A MITEL
PRODUCT
GUIDE

Mitel OpenScape Contact Center Enterprise V12

SDK V12

Programming Guide
10/2024

A31003-S22C0-R102-01-7620

Mitel

Notices

The information contained in this document is believed to be accurate in all respects but is not warranted by Mitel Europe Limited. The information is subject to change without notice and should not be construed in any way as a commitment by Mitel or any of its affiliates or subsidiaries. Mitel and its affiliates and subsidiaries assume no responsibility for any errors or omissions in this document. Revisions of this document or new editions of it may be issued to incorporate such changes. No part of this document can be reproduced or transmitted in any form or by any means - electronic or mechanical - for any purpose without written permission from Mitel Networks Corporation.

Trademarks

The trademarks, service marks, logos, and graphics (collectively “Trademarks”) appearing on Mitel’s Internet sites or in its publications are registered and unregistered trademarks of Mitel Networks Corporation (MNC) or its subsidiaries (collectively “Mitel), Unify Software and Solutions GmbH & Co. KG or its affiliates (collectively “Unify”) or others. Use of the Trademarks is prohibited without the express consent from Mitel and/or Unify. Please contact our legal department at iplegal@mitel.com for additional information. For a list of the worldwide Mitel and Unify registered trademarks, please refer to the website: http://www.mitel.com/trademarks.

© Copyright 2024, Mitel Networks Corporation

All rights reserved

Contents

1 About this guide . . . 27
1.1 Who should use this guide . . . 28
1.2 Formatting conventions . . . 28
1.3 Documentation feedback . . . 28
2 About the OpenScape Contact Center SDK . . . 29
2.1 System requirements . . . 29
2.2 Installing the OpenScape Contact Center SDK . . . 30
2.2.1 Installing the SDK for all users . . . 31
2.3 Redistributing and patching the SDK . . . 31
2.4 Writing SDK applications in a multitenant environment . . . 31
2.5 What the SDK includes . . . 32
2.6 Creating the Manager objects . . . 34
2.6.1 Creating a Manager object with events . . . 34
2.6.1.1 Creating the object as a global object . . . 34
2.6.1.2 Before using a Manager object . . . 34
2.6.1.3 Creating an event handler automatically . . . 35
2.6.2 Creating a Manager object without events . . . 35
2.6.3 Creating a Manager object locally . . . 36
2.7 Destroying the Manager objects . . . 36
2.8 About events . . . 37
2.8.1 Listening for events . . . 37
2.8.2 Stop listening for events . . . 37
2.8.3 ListenForEvent and StopListeningForEvents parameters . . . 37
2.8.3.1 Type parameter . . . 38
2.8.3.2 Resource parameter . . . 38
2.8.4 Examples of events . . . 38
2.8.4.1 Example 1: Making calls . . . 38
2.8.4.2 Example 2: Receiving all voice events . . . 39
2.8.4.3 Example 3: Receiving voice events for an extension . . . 39
2.9 Handling connectivity issues . . . 40
3 Enumerations . . . 43
3.1 enAdministrationBaseObjectTypes . . . 43
3.2 enAdministrationEventCodes . . . 44
3.3 enAdministrationEventObjectTypes . . . 45
3.4 enAdministrationEventTypes . . . 47
3.5 enAdministrationFunctionalities . . . 48
3.6 enCallbackOrigins . . . 48
3.7 enCallStates . . . 49
3.8 enCapabilityType . . . 52
3.9 enContactTypes . . . 52
3.10 enDeliveredReasons . . . 53
3.11 enDequeueReasons . . . 53
3.12 enDiagnosticFilters . . . 54
3.13 enDisconnectReasons . . . 55
3.14 enDivertReasons . . . 56
3.15 enEmailsHistoryQueryCriteria . . . 57
3.16 enEmailTemplateTypes . . . 57

3.17 enEmailTypes . . . 57
3.18 enErrors . . . 59
3.19 enEventModes . . . 81
3.20 enHandlingStates . . . 82
3.21 enHiPathProCenterEventCodes . . . 83
3.22 enHiPathProCenterEventObjectTypes . . . 84
3.23 enHiPathProCenterEventTypes . . . 84
3.24 enHiPathProCenterFunctionalities . . . 85
3.25 enHoldReasons . . . 85
3.26 enKeyValuePairTypes . . . 86
3.27 enLocaleIDs . . . 86
3.28 enManagerStates . . . 87
3.29 enMediaBaseObjectTypes . . . 87
3.30 enMediaEventCodes . . . 89
3.31 enMediaEventObjectTypes . . . 95
3.32 enMediaEventTypes . . . 98
3.33 enMediaFunctionalities . . . 100
3.34 enMediaTypes . . . 100
3.35 enPartyTypes . . . 101
3.36 enPresenceStates . . . 101
3.37 enResults . . . 102
3.38 enRoutingBaseObjectTypes . . . 102
3.39 enRoutingEventCodes . . . 103
3.40 enRoutingEventObjectTypes . . . 105
3.41 enRoutingEventTypes . . . 106
3.42 enRoutingFunctionalities . . . 107
3.43 enRoutingStates . . . 107
3.44 enServers . . . 108
3.45 enServerStates . . . 109
3.46 enStatisticsEventCodes . . . 111
3.47 enStatisticsEventObjectTypes . . . 112
3.48 enStatisticsEventTypes . . . 113
3.49 enStatisticsFunctionalities . . . 113
3.50 enUserTypes . . . 114
3.51 enWebCollaborationTemplateTypes . . . 115

4 Common Interfaces . . . 117
4.1 ContactData . . . 117
4.1.1 Properties . . . 117
4.1.1.1 _NewEnum . . . 117
4.1.1.2 Count . . . 118
4.1.2 Methods . . . 119
4.1.2.1 Add . . . 119
4.1.2.2 Item . . . 120
4.1.2.3 Remove . . . 121
4.1.2.4 RemoveAll . . . 122
4.2 EnumList . . . 122
4.2.1 Properties . . . 122
4.2.1.1 _NewEnum . . . 123
4.2.1.2 Count . . . 124
4.2.1.3 Item . . . 124
4.2.2 Methods . . . 125

4.2.2.1 Add . . . 125
4.2.2.2 Remove . . . 126
4.2.2.3 RemoveAll . . . 126
4.3 KeyList . . . 127
4.3.1 Properties . . . 127
4.3.1.1 _NewEnum . . . 127
4.3.1.2 Count . . . 128
4.3.2 Methods . . . 128
4.3.2.1 Add . . . 129
4.3.2.2 Item . . . 129
4.3.2.3 Remove . . . 130
4.3.2.4 RemoveAll . . . 130
4.3.2.5 RemoveAt . . . 131
4.4 KeyValuePair . . . 132
4.4.1 Properties . . . 132
4.4.1.1 Key . . . 132
4.4.1.2 Type . . . 133
4.4.1.3 Value . . . 133
4.4.1.4 Visible . . . 134
4.5 ManagerStateChangedEvent . . . 135
4.5.1 Properties . . . 135
4.5.1.1 State . . . 135
4.6 MediaTypes . . . 136
4.6.1 Properties . . . 137
4.6.1.1 _NewEnum . . . 137
4.6.1.2 Count . . . 138
4.6.1.3 Item . . . 138
4.7 Parameters . . . 139
4.7.1 Properties . . . 139
4.7.1.1 _NewEnum . . . 139
4.7.1.2 Count . . . 140
4.7.2 Methods . . . 141
4.7.2.1 Add . . . 141
4.7.2.2 Item . . . 141
4.7.2.3 Remove . . . 143
4.7.2.4 RemoveAll . . . 144
4.8 XMLAccess . . . 144
4.8.1 Methods . . . 144
4.8.1.1 AsXML . . . 145
4.8.1.2 FromXML . . . 146

5 OpenScape Contact Center Manager . . . 147
5.1 HiPathProCenterManager object . . . 147
5.1.1 Properties . . . 147
5.1.1.1 BusinessUnitName . . . 148
5.1.1.2 DiagnosticFileName . . . 148
5.1.1.3 DiagnosticsFilterEnabled . . . 149
5.1.1.4 LocaleID . . . 149
5.1.1.5 LoggedOnUserKey . . . 150
5.1.1.6 State . . . 150
5.1.2 Methods . . . 150
5.1.2.1 DisableDiagnosticFilter . . . 151

5.1.2.2 EnableDiagnosticFilter . . . 152
5.1.2.3 GetFunctionalityState . . . 153
5.1.2.4 HireAdministrationManager . . . 154
5.1.2.5 HireMediaManager . . . 155
5.1.2.6 HireRoutingManager . . . 156
5.1.2.7 HireStatisticsManager . . . 158
5.1.2.8 Initialize . . . 159
5.1.2.9 ListenForEvents . . . 161
5.1.2.10 Logon . . . 163
5.1.2.11 StopListeningForEvents . . . 164
5.1.2.12 WriteToDiagnosticFile . . . 165
5.1.3 Events . . . 166
5.1.3.1 EventOccurred . . . 166
5.2 HiPathProCenterEvent . . . 167
5.2.1 Properties . . . 168
5.2.1.1 Code . . . 168
5.2.1.2 EventType . . . 169
5.2.1.3 ObjectType . . . 171
5.2.1.4 Resource . . . 171
5.3 ServerStateChangedEvent . . . 173
5.3.1 Properties . . . 173
5.3.1.1 Server . . . 174
5.3.1.2 ServerState . . . 174

6 Administration Manager . . . 175
6.1 AdministrationManager object . . . 175
6.1.1 Properties . . . 175
6.1.1.1 State . . . 175
6.1.2 Methods . . . 176
6.1.2.1 GetFunctionalityState . . . 176
6.1.2.2 ListenForEvents . . . 177
6.1.2.3 QueryAggregates . . . 179
6.1.2.4 QueryCallbackDeleteReasons . . . 180
6.1.2.5 QueryCallbackRetryReasons . . . 180
6.1.2.6 QueryDepartments . . . 181
6.1.2.7 QueryEmailDiscardReasons . . . 181
6.1.2.8 QueryEmailTemplates . . . 182
6.1.2.9 QueryGroups . . . 183
6.1.2.10 QueryPostProcessingReasons . . . 183
6.1.2.11 QueryQueues . . . 184
6.1.2.12 QueryUnavailableReasons . . . 184
6.1.2.13 QueryUsers . . . 185
6.1.2.14 QueryWebCollaborationLanguages . . . 185
6.1.2.15 QueryWebCollaborationTemplates . . . 186
6.1.2.16 QueryWorkReasons . . . 187
6.1.2.17 QueryWrapupReasons . . . 187
6.1.2.18 StopListeningForEvents . . . 188
6.1.3 Events . . . 189
6.1.3.1 EventOccurred . . . 189
6.2 AdministrationEvent . . . 191
6.2.1 Properties . . . 192
6.2.1.1 Code . . . 192

6.2.1.2 EventType . . . 193
6.2.1.3 ObjectType . . . 193
6.2.1.4 Resource . . . 194
6.3 Aggregate . . . 195
6.3.1 Properties . . . 195
6.3.1.1 Description . . . 195
6.3.1.2 Key . . . 196
6.3.1.3 MediaType . . . 196
6.3.1.4 Name . . . 196
6.4 Aggregates . . . 197
6.4.1 Properties . . . 197
6.4.1.1 _NewEnum . . . 197
6.4.1.2 Count . . . 198
6.4.1.3 Item . . . 199
6.5 DeleteList . . . 200
6.5.1 Properties . . . 200
6.5.1.1 _NewEnum . . . 200
6.5.1.2 Count . . . 201
6.5.1.3 Item . . . 201
6.5.1.4 ObjectType . . . 202
6.6 Department . . . 203
6.6.1 Properties . . . 203
6.6.1.1 Description . . . 204
6.6.1.2 Key . . . 204
6.6.1.3 Name . . . 204
6.7 Departments . . . 205
6.7.1 Properties . . . 205
6.7.1.1 _NewEnum . . . 205
6.7.1.2 Count . . . 206
6.7.1.3 Item . . . 206
6.8 EmailDiscardReason . . . 207
6.8.1 Properties . . . 207
6.8.1.1 Description . . . 207
6.8.1.2 Key . . . 208
6.8.1.3 Name . . . 208
6.9 EmailDiscardReasons . . . 209
6.9.1 Properties . . . 209
6.9.1.1 _NewEnum . . . 209
6.9.1.2 Count . . . 210
6.9.1.3 Item . . . 210
6.10 EmailTemplate . . . 211
6.10.1 Properties . . . 211
6.10.1.1 Content . . . 212
6.10.1.2 Description . . . 212
6.10.1.3 Key . . . 213
6.10.1.4 Name . . . 213
6.10.1.5 QueueKeys . . . 213
6.10.1.6 Type . . . 214
6.11 EmailTemplates . . . 214
6.11.1 Properties . . . 214
6.11.1.1 _NewEnum . . . 215
6.11.1.2 Count . . . 215

6.11.1.3 Item. 216
6.12 Group 217
6.12.1 Properties. 217
6.12.1.1 Description 217
6.12.1.2 Key 218
6.12.1.3 Name 218
6.13 Groups 218
6.13.1 Properties. 219
6.13.1.1 _NewEnum 219
6.13.1.2 Count 220
6.13.1.3 Item. 220
6.14 Language 221
6.14.1 Properties. 221
6.14.1.1 Description 221
6.14.1.2 Key 222
6.14.1.3 Name 222
6.15 Languages 223
6.15.1 Properties. 223
6.15.1.1 _NewEnum 223
6.15.1.2 Count 224
6.15.1.3 Item. 224
6.16 PostProcessingReason 225
6.16.1 Properties. 225
6.16.1.1 Description 226
6.16.1.2 Name 226
6.16.1.3 ReasonKey 226
6.17 PostProcessingReasons 227
6.17.1 Properties. 227
6.17.1.1 _NewEnum 227
6.17.1.2 Count 228
6.17.1.3 Item. 229
6.18 Queue 230
6.18.1 Properties. 230
6.18.1.1 Description 231
6.18.1.2 Key 231
6.18.1.3 MediaType 232
6.18.1.4 Name 232
6.19 Queues 233
6.19.1 Properties. 233
6.19.1.1 _NewEnum 233
6.19.1.2 Count 234
6.19.1.3 Item. 235
6.20 RoutingStateReason 236
6.20.1 Properties. 236
6.20.1.1 Description 236
6.20.1.2 Name 237
6.20.1.3 ReasonKey 237
6.21 RoutingStateReasons 238
6.21.1 Properties. 238
6.21.1.1 _NewEnum 239
6.21.1.2 Count 240
6.21.1.3 Item. 240

6.22 User . . . 241
6.22.1 Properties . . . 242
6.22.1.1 ACDGroupNumber . . . 242
6.22.1.2 Department . . . 242
6.22.1.3 DepartmentKey . . . 243
6.22.1.4 FirstName . . . 243
6.22.1.5 ID . . . 244
6.22.1.6 Key . . . 244
6.22.1.7 LastName . . . 245
6.22.1.8 MaximumCapacity . . . 245
6.22.1.9 MediaTypes . . . 245
6.22.1.10 MiddleInitial . . . 246
6.22.1.11 MultipleContactHandlingCapable . . . 246
6.22.1.12 Type . . . 247
6.22.1.13 UserName . . . 247
6.23 Users . . . 247
6.23.1 Properties . . . 248
6.23.1.1 _NewEnum . . . 248
6.23.1.2 Count . . . 249
6.23.1.3 Item . . . 249
6.24 WebCollaborationTemplate . . . 250
6.24.1 Properties . . . 250
6.24.1.1 Content . . . 250
6.24.1.2 Description . . . 251
6.24.1.3 Key . . . 251
6.24.1.4 LanguageKey . . . 251
6.24.1.5 Name . . . 252
6.24.1.6 QueueKeys . . . 252
6.24.1.7 Type . . . 252
6.25 WebCollaborationTemplates . . . 254
6.25.1 Properties . . . 254
6.25.1.1 _NewEnum . . . 254
6.25.1.2 Count . . . 255
6.25.1.3 Item . . . 255
6.26 WrapupReason . . . 256
6.26.1 Properties . . . 257
6.26.1.1 Description . . . 257
6.26.1.2 Key . . . 257
6.26.1.3 Name . . . 258
6.27 WrapupReasons . . . 258
6.27.1 Properties . . . 258
6.27.1.1 _NewEnum . . . 259
6.27.1.2 Count . . . 259
6.27.1.3 Item . . . 260

7 Media Manager . . . 261
7.1 MediaManager object . . . 261
7.1.1 Properties . . . 261
7.1.1.1 State . . . 261
7.1.2 Methods . . . 262
7.1.2.1 GetFunctionalityState . . . 262
7.1.2.2 NewAgent . . . 263

7.1.2.3 NewCallbackCall . . . 263
7.1.2.4 NewEmailCall . . . 264
7.1.2.5 NewServicesEnabled . . . 264
7.1.2.6 NewVoiceCall . . . 265
7.1.2.7 NewWebCollaborationCall . . . 265
7.1.2.8 Query . . . 266
7.1.2.9 QueryAsyncClientEmailsHistory . . . 267
7.1.2.10 QueryAsyncConversationEmailsHistory . . . 269
7.1.2.11 QueryAsyncEmailsHistory . . . 270
7.1.2.12 QueryAsyncScheduledCallbacks . . . 272
7.1.2.13 QueryEmailSummaries . . . 273
7.1.2.14 Update . . . 274
7.1.3 Events . . . 275
7.1.3.1 EventOccurred . . . 275
7.1.3.2 ListenForEvents . . . 277
7.1.3.3 StopListeningForEvents . . . 278
7.2 Agent . . . 279
7.2.1 Properties . . . 280
7.2.1.1 Extension . . . 280
7.2.1.2 ForwardingState . . . 280
7.2.1.3 HandlingStates . . . 281
7.2.1.4 ID . . . 282
7.2.1.5 IsMandatoryWrapupRequired . . . 282
7.2.1.6 Key . . . 283
7.2.1.7 LoggedOnForMediaType . . . 283
7.2.1.8 PreferredDevice . . . 284
7.2.1.9 PresenceState . . . 285
7.2.1.10 RoutingState . . . 285
7.2.1.11 RoutingStateReasonKey . . . 286
7.2.2 Methods . . . 286
7.2.2.1 AddCapability . . . 286
7.2.2.2 GetVoicePreferredDevice . . . 287
7.2.2.3 Logoff . . . 288
7.2.2.4 Logon . . . 289
7.2.2.5 RemoveCapability . . . 290
7.2.2.6 SetVoicePreferredDevice . . . 292
7.2.2.7 SetRoutingState . . . 293
7.3 AgentStatusEvent . . . 294
7.3.1 Properties . . . 294
7.3.1.1 AgentID . . . 294
7.3.1.2 AgentKey . . . 295
7.3.1.3 Extension . . . 295
7.3.1.4 HandlingStates . . . 296
7.3.1.5 LoggedOnMediaTypes . . . 296
7.3.1.6 PresenceState . . . 297
7.3.1.7 RoutingState . . . 297
7.3.1.8 RoutingStateReasonKey . . . 298
7.4 AutoAcknowledgedEvent . . . 298
7.4.1 Properties . . . 298
7.4.1.1 CallID . . . 299
7.4.1.2 ConversationID . . . 299
7.4.1.3 OriginalFromParty . . . 300

7.4.1.4 OriginalToParty . . . 300
7.4.1.5 SequenceNumber . . . 301
7.5 CallbackCall . . . 301
7.5.1 Properties . . . 301
7.5.1.1 AgentKey . . . 302
7.5.1.2 CallbackSchedules . . . 302
7.5.1.3 CallID . . . 303
7.5.1.4 ContactData . . . 303
7.5.1.5 CreatorAgentKey . . . 304
7.5.1.6 CustomerName . . . 304
7.5.1.7 Description . . . 305
7.5.1.8 Priority . . . 305
7.5.1.9 QueueKey . . . 306
7.5.1.10 Result . . . 306
7.5.1.11 RetryDelay . . . 306
7.5.1.12 SequenceNumber . . . 307
7.5.1.13 State . . . 307
7.5.1.14 WaitTime . . . 308
7.5.2 Methods . . . 308
7.5.2.1 Accept . . . 308
7.5.2.2 AddCallbackSchedule . . . 309
7.5.2.3 Complete . . . 310
7.5.2.4 Create . . . 313
7.5.2.5 Delete . . . 314
7.5.2.6 RemoveCallbackSchedule . . . 316
7.5.2.7 UpdateCallbackSchedule . . . 317
7.6 CallbackReason . . . 318
7.6.1 Properties . . . 318
7.6.1.1 DefaultRetryInterval . . . 318
7.6.1.2 Description . . . 319
7.6.1.3 Key . . . 319
7.6.1.4 Name . . . 320
7.7 CallbackReasons . . . 320
7.7.1 Properties . . . 320
7.7.1.1 _NewEnum . . . 321
7.7.1.2 Count . . . 322
7.7.1.3 Item . . . 322
7.8 CallbackSchedule . . . 323
7.8.1 Properties . . . 323
7.8.1.1 EndTime . . . 323
7.8.1.2 StartTime . . . 324
7.8.1.3 TelephoneNumber . . . 324
7.9 CallbackSchedules . . . 324
7.9.1 Properties . . . 325
7.9.1.1 _NewEnum . . . 325
7.9.1.2 Count . . . 326
7.9.1.3 Item . . . 326
7.10 CallInformationUpdatedEvent . . . 327
7.10.1 Properties . . . 327
7.10.1.1 CallID . . . 328
7.10.1.2 ContactData . . . 328
7.10.1.3 ConversationID . . . 329

7.10.1.4 Description . . . 329
7.10.1.5 SequenceNumber . . . 330
7.11 CallMeEvent . . . 330
7.11.1 Properties . . . 330
7.11.1.1 CallID . . . 331
7.11.1.2 OriginalFromParty . . . 331
7.11.1.3 OriginalToParty . . . 332
7.11.1.4 Party . . . 332
7.11.1.5 SequenceNumber . . . 333
7.11.1.6 TelephoneNumber . . . 333
7.12 ConferencedEvent . . . 334
7.12.1 Properties . . . 334
7.12.1.1 CallID . . . 334
7.12.1.2 InvitedParty . . . 335
7.12.1.3 InvitingParty . . . 335
7.12.1.4 OriginalFromParty . . . 336
7.12.1.5 OriginalToParty . . . 336
7.12.1.6 SequenceNumber . . . 337
7.13 ContentAddedEvent . . . 337
7.13.1 Properties . . . 337
7.13.1.1 CallID . . . 338
7.13.1.2 OriginalFromParty . . . 338
7.13.1.3 OriginalToParty . . . 339
7.13.1.4 Party . . . 339
7.13.1.5 SequenceNumber . . . 340
7.13.1.6 Text . . . 340
7.14 Criterion . . . 341
7.14.1 Properties . . . 341
7.14.1.1 Type . . . 341
7.14.1.2 Value . . . 342
7.15 CriterionList . . . 342
7.15.1 Properties . . . 342
7.15.1.1 _NewEnum . . . 343
7.15.1.2 Count . . . 344
7.15.1.3 Item . . . 344
7.15.2 Methods . . . 344
7.15.2.1 Add . . . 345
7.15.2.2 Remove . . . 346
7.15.2.3 RemoveAll . . . 347
7.16 DeliveredEvent . . . 348
7.16.1 Properties . . . 348
7.16.1.1 CallID . . . 348
7.16.1.2 ConversationID . . . 349
7.16.1.3 DeliveredToParty . . . 349
7.16.1.4 FromParty . . . 350
7.16.1.5 HeldParty . . . 350
7.16.1.6 OriginalFromParty . . . 351
7.16.1.7 OriginalToParty . . . 351
7.16.1.8 Reason . . . 352
7.16.1.9 RedirectParty . . . 352
7.16.1.10 SequenceNumber . . . 353
7.16.1.11 ToParty . . . 353

7.17 DisconnectedEvent . . . 354
7.17.1 Properties . . . 354
7.17.1.1 CallID . . . 354
7.17.1.2 ConversationID . . . 354
7.17.1.3 DiscardReasonKey . . . 355
7.17.1.4 DisconnectedParty . . . 356
7.17.1.5 OriginalFromParty . . . 357
7.17.1.6 OriginalToParty . . . 357
7.17.1.7 Reason . . . 358
7.17.1.8 SequenceNumber . . . 358
7.18 DivertedEvent . . . 358
7.18.1 Properties . . . 359
7.18.1.1 CallID . . . 359
7.18.1.2 ConversationID . . . 359
7.18.1.3 DestinationParty . . . 360
7.18.1.4 DivertedParty . . . 360
7.18.1.5 DivertingParty . . . 361
7.18.1.6 OriginalFromParty . . . 361
7.18.1.7 OriginalToParty . . . 362
7.18.1.8 Reason . . . 362
7.18.1.9 SequenceNumber . . . 363
7.19 EmailAddress . . . 363
7.19.1 Properties . . . 363
7.19.1.1 DisplayName . . . 364
7.19.1.2 EmailAddress . . . 364
7.20 EmailAddresses . . . 364
7.20.1 Properties . . . 365
7.20.1.1 _NewEnum . . . 365
7.20.1.2 Count . . . 366
7.20.1.3 Item . . . 366
7.20.2 Methods . . . 367
7.20.2.1 Add . . . 367
7.20.2.2 Remove . . . 368
7.21 EmailAttachment . . . 368
7.21.1 Properties . . . 369
7.21.1.1 NameUTF8 . . . 369
7.21.1.2 Path . . . 369
7.21.2 Methods . . . 370
7.21.2.1 Retrieve . . . 371
7.22 EmailAttachments . . . 371
7.22.1 Properties . . . 372
7.22.1.1 _NewEnum . . . 372
7.22.1.2 Count . . . 373
7.22.1.3 Item . . . 373
7.22.2 Methods . . . 374
7.22.2.1 Add . . . 374
7.22.2.2 Remove . . . 375
7.23 EmailCall . . . 375
7.23.1 Properties . . . 376
7.23.1.1 AgentKey . . . 376
7.23.1.2 Attachments . . . 376
7.23.1.3 BodyUTF8 . . . 377

7.23.1.4 CallID . . . 377
7.23.1.5 ContactData . . . 378
7.23.1.6 ConversationID . . . 378
7.23.1.7 Description . . . 379
7.23.1.8 ExternalConsultCallID . . . 379
7.23.1.9 FromEmailAddress . . . 380
7.23.1.10 LastDeferTimeStamp . . . 380
7.23.1.11 QueueKey . . . 381
7.23.1.12 ReplyAttachments . . . 381
7.23.1.13 ReplyBodyUTF8 . . . 382
7.23.1.14 ReplyFromEmailAddress . . . 382
7.23.1.15 ReplySubjectUTF8 . . . 383
7.23.1.16 ReplyToBCCEmailAddresses . . . 383
7.23.1.17 ReplyToCCEmailAddresses . . . 384
7.23.1.18 ReplyToEmailAddress . . . 384
7.23.1.19 SequenceNumber . . . 385
7.23.1.20 State . . . 385
7.23.1.21 SubjectUTF8 . . . 386
7.23.1.22 ToBCCEmailAddresses . . . 386
7.23.1.23 ToCCEmailAddresses . . . 387
7.23.1.24 ToEmailAddress . . . 387
7.23.1.25 ToEmailAddresses . . . 388
7.23.1.26 WaitTime . . . 388
7.23.2 Methods . . . 389
7.23.2.1 Accept . . . 389
7.23.2.2 Defer . . . 390
7.23.2.3 Discard . . . 391
7.23.2.4 ExternalConsult . . . 392
7.23.2.5 ExternalTransfer . . . 393
7.23.2.6 Reply . . . 395
7.23.2.7 Requeue . . . 396
7.23.2.8 Resume . . . 397
7.23.2.9 Transfer . . . 398
7.24 EmailsHistoryQueryResultEvent . . . 399
7.24.1 Properties . . . 399
7.24.1.1 _NewEnum . . . 399
7.24.1.2 Count . . . 400
7.24.1.3 IsOverflow . . . 400
7.24.1.4 Item . . . 402
7.24.1.5 QueryID . . . 403
7.24.1.6 Result . . . 403
7.25 EmailSummaries . . . 404
7.25.1 Properties . . . 404
7.25.1.1 _NewEnum . . . 404
7.25.1.2 Count . . . 405
7.25.1.3 Item . . . 405
7.26 EmailSummary . . . 406
7.26.1 Properties . . . 406
7.26.1.1 CallID . . . 407
7.26.1.2 DeferTimeStamp . . . 407
7.26.1.3 FromEmailAddress . . . 407
7.26.1.4 QueueKey . . . 409

7.26.1.5 State . . . 409
7.26.1.6 Subject . . . 410
7.27 EstablishedEvent . . . 410
7.27.1 Properties . . . 410
7.27.1.1 CallID . . . 411
7.27.1.2 ConversationID . . . 411
7.27.1.3 FromParty . . . 412
7.27.1.4 OriginalFromParty . . . 412
7.27.1.5 OriginalToParty . . . 413
7.27.1.6 RedirectParty . . . 413
7.27.1.7 SequenceNumber . . . 414
7.27.1.8 ToParty . . . 414
7.27.1.9 TrunkID . . . 415
7.28 FailedEvent . . . 415
7.28.1 Properties . . . 415
7.28.1.1 CallID . . . 416
7.28.1.2 FromParty . . . 416
7.28.1.3 OriginalFromParty . . . 417
7.28.1.4 OriginalToParty . . . 417
7.28.1.5 SequenceNumber . . . 418
7.28.1.6 ToParty . . . 418
7.29 ForwardingStateEvent . . . 419
7.29.1 Properties . . . 419
7.29.1.1 Extension . . . 419
7.29.1.2 ForwardingState . . . 420
7.29.1.3 PreferredDevice . . . 420
7.30 HeldEvent . . . 421
7.30.1 Properties . . . 421
7.30.1.1 CallID . . . 421
7.30.1.2 ConversationID . . . 422
7.30.1.3 HeldParty . . . 422
7.30.1.4 HoldingParty . . . 423
7.30.1.5 OriginalFromParty . . . 423
7.30.1.6 OriginalToParty . . . 424
7.30.1.7 Reason . . . 424
7.30.1.8 SequenceNumber . . . 425
7.31 MediaBase . . . 425
7.31.1 Properties . . . 426
7.31.1.1 Type . . . 426
7.32 MediaEvent . . . 426
7.32.1 Properties . . . 427
7.32.1.1 Code . . . 428
7.32.1.2 EventType . . . 428
7.32.1.3 MediaType . . . 429
7.32.1.4 ObjectType . . . 429
7.32.1.5 Resource . . . 430
7.33 NetworkReachedEvent . . . 430
7.33.1 Properties . . . 432
7.33.1.1 CallID . . . 432
7.33.1.2 FromParty . . . 432
7.33.1.3 OriginalFromParty . . . 433
7.33.1.4 OriginalToParty . . . 433

7.33.1.5 SequenceNumber . . . 434
7.33.1.6 ToParty . . . 434
7.34 OffHookEvent . . . 435
7.34.1 Properties . . . 435
7.34.1.1 CallID . . . 435
7.34.1.2 ForConsult . . . 436
7.34.1.3 FromParty . . . 436
7.34.1.4 OriginalFromParty . . . 437
7.34.1.5 OriginalToParty . . . 437
7.34.1.6 SequenceNumber . . . 438
7.35 OriginatedEvent . . . 438
7.35.1 Properties . . . 438
7.35.1.1 CallID . . . 439
7.35.1.2 FromParty . . . 439
7.35.1.3 OriginalFromParty . . . 440
7.35.1.4 OriginalToParty . . . 440
7.35.1.5 ReferenceCallID . . . 441
7.35.1.6 SequenceNumber . . . 441
7.35.1.7 ToParty . . . 442
7.36 Parties . . . 442
7.36.1 Properties . . . 442
7.36.1.1 _NewEnum . . . 443
7.36.1.2 Count . . . 444
7.36.1.3 Item . . . 444
7.37 Party . . . 445
7.37.1 Properties . . . 445
7.37.1.1 AgentKey . . . 446
7.37.1.2 Device . . . 446
7.37.1.3 PartyType . . . 446
7.38 QueuedEvent . . . 447
7.38.1 Properties . . . 447
7.38.1.1 CallID . . . 447
7.38.1.2 ConversationID . . . 448
7.38.1.3 FromParty . . . 448
7.38.1.4 OriginalFromParty . . . 449
7.38.1.5 OriginalToParty . . . 449
7.38.1.6 QueuedToParty . . . 450
7.38.1.7 SequenceNumber . . . 450
7.38.1.8 ToParty . . . 451
7.39 RetrievedEvent . . . 451
7.39.1 Properties . . . 451
7.39.1.1 CallID . . . 452
7.39.1.2 ConversationID . . . 452
7.39.1.3 HeldParty . . . 453
7.39.1.4 OriginalFromParty . . . 453
7.39.1.5 OriginalToParty . . . 453
7.39.1.6 RetrievingParty . . . 454
7.39.1.7 SequenceNumber . . . 455
7.40 ScheduledCallbackElement . . . 455
7.40.1 Properties . . . 455
7.40.1.1 CallID . . . 456
7.40.1.2 CreatedByKey . . . 456

7.40.1.3 CustomerName . . . 457
7.40.1.4 Description . . . 457
7.40.1.5 NextScheduledPhoneNumber . . . 458
7.40.1.6 NextScheduledStart . . . 458
7.40.1.7 Origin . . . 459
7.40.1.8 Priority . . . 459
7.40.1.9 QueueKey . . . 460
7.41 ScheduledCallbacksQueryResultEvent . . . 460
7.41.1 Properties . . . 460
7.41.1.1 _NewEnum . . . 461
7.41.1.2 Count . . . 462
7.41.1.3 Item . . . 462
7.41.1.4 QueryID . . . 463
7.41.1.5 Result . . . 464
7.42 ServicesEnabled . . . 464
7.42.1 Properties . . . 464
7.42.1.1 Answer . . . 465
7.42.1.2 Available . . . 465
7.42.1.3 Conference . . . 466
7.42.1.4 Consult . . . 466
7.42.1.5 Dial . . . 467
7.42.1.6 Disconnect . . . 467
7.42.1.7 Divert . . . 468
7.42.1.8 Extension . . . 468
7.42.1.9 Hold . . . 468
7.42.1.10 Logoff . . . 469
7.42.1.11 Logon . . . 469
7.42.1.12 Reconnect . . . 470
7.42.1.13 Retrieve . . . 470
7.42.1.14 Transfer . . . 471
7.42.1.15 Unavailable . . . 471
7.42.1.16 Work . . . 472
7.43 ServicesEnabledEvent . . . 472
7.43.1 Properties . . . 472
7.43.1.1 Answer . . . 473
7.43.1.2 Available . . . 473
7.43.1.3 Conference . . . 474
7.43.1.4 Consult . . . 474
7.43.1.5 Dial . . . 475
7.43.1.6 Disconnect . . . 475
7.43.1.7 Divert . . . 476
7.43.1.8 Hold . . . 476
7.43.1.9 Logoff . . . 477
7.43.1.10 Logon . . . 477
7.43.1.11 Reconnect . . . 478
7.43.1.12 Retrieve . . . 478
7.43.1.13 Transfer . . . 479
7.43.1.14 Unavailable . . . 479
7.43.1.15 Work . . . 480
7.44 TransferredEvent . . . 480
7.44.1 Properties . . . 480
7.44.1.1 CallID . . . 481

7.44.1.2 ConversationID . . . 481
7.44.1.3 FromParty . . . 482
7.44.1.4 OriginalFromParty . . . 482
7.44.1.5 OriginalToParty . . . 482
7.44.1.6 RedirectParty . . . 483
7.44.1.7 SequenceNumber . . . 484
7.44.1.8 ToParty . . . 484
7.44.1.9 TransferredParty . . . 485
7.45 UserEventHandlingState . . . 485
7.45.1 Properties . . . 485
7.45.1.1 CallID . . . 486
7.45.1.2 HandlingState . . . 486
7.45.1.3 HandlingStateReasonKey . . . 487
7.45.1.4 IsPrimary . . . 487
7.45.1.5 MediaType . . . 488
7.45.1.6 SequenceNumber . . . 488
7.46 UserEventHandlingStates . . . 488
7.46.1 Properties . . . 489
7.46.1.1 _NewEnum . . . 489
7.46.1.2 Count . . . 491
7.46.1.3 Item . . . 491
7.47 UserHandlingState . . . 492
7.47.1 Properties . . . 492
7.47.1.1 CallID . . . 493
7.47.1.2 ContactMediaType . . . 493
7.47.1.3 HandlingState . . . 493
7.47.1.4 HandlingStateReasonKey . . . 494
7.47.1.5 IsMandatoryWrapUp . . . 494
7.47.1.6 IsPrimary . . . 495
7.47.1.7 SequenceNumber . . . 495
7.47.2 Methods . . . 496
7.47.2.1 CancelMandatoryWrappingUp . . . 496
7.47.2.2 ExitPostProcessing . . . 496
7.47.2.3 RequestPostProcessing . . . 497
7.47.2.4 SetPrimary . . . 497
7.48 UserHandlingStates . . . 497
7.48.1 Properties . . . 498
7.48.1.1 _NewEnum . . . 498
7.48.1.2 Count . . . 499
7.48.1.3 Item . . . 499
7.49 UserMediaNotReadyEvent . . . 500
7.49.1 Properties . . . 500
7.49.1.1 UserMediaType . . . 501
7.50 UserMediaReadyEvent . . . 501
7.50.1 Properties . . . 501
7.50.1.1 UserMediaType . . . 502
7.51 VoiceCall . . . 502
7.51.1 Properties . . . 502
7.51.1.1 AgentKey . . . 503
7.51.1.2 ANI . . . 503
7.51.1.3 CallID . . . 504
7.51.1.4 ContactData . . . 504

7.51.1.5 ContactType . . . 505
7.51.1.6 Description . . . 505
7.51.1.7 DNIS . . . 506
7.51.1.8 Parties . . . 506
7.51.1.9 QueueKey . . . 507
7.51.1.10 SequenceNumber . . . 507
7.51.1.11 State . . . 508
7.51.1.12 WaitTime . . . 508
7.51.2 Methods . . . 508
7.51.2.1 Answer . . . 509
7.51.2.2 Conference . . . 510
7.51.2.3 Consult . . . 511
7.51.2.4 Dial . . . 512
7.51.2.5 Disconnect . . . 513
7.51.2.6 Divert . . . 514
7.51.2.7 Hold . . . 515
7.51.2.8 Reconnect . . . 516
7.51.2.9 Retrieve . . . 517
7.51.2.10 Transfer . . . 518
7.52 WebCollaborationCall . . . 519
7.52.1 Properties . . . 519
7.52.1.1 AgentKey . . . 519
7.52.1.2 CallID . . . 520
7.52.1.3 ContactData . . . 520
7.52.1.4 CustomerName . . . 521
7.52.1.5 Description . . . 521
7.52.1.6 Destination . . . 522
7.52.1.7 QueueKey . . . 522
7.52.1.8 SequenceNumber . . . 523
7.52.1.9 Source . . . 523
7.52.1.10 State . . . 524
7.52.1.11 Transcript . . . 524
7.52.1.12 WaitTime . . . 525
7.52.2 Methods . . . 525
7.52.2.1 Accept . . . 525
7.52.2.2 ConferenceRequest . . . 526
7.52.2.3 Disconnect . . . 527
7.52.2.4 PushCallMeForm . . . 528
7.52.2.5 Reject . . . 529
7.52.2.6 Requeue . . . 530
7.52.2.7 SendContent . . . 531

8 Routing Manager . . . 533
8.1 RoutingManager object . . . 533
8.1.1 Properties . . . 533
8.1.1.1 State . . . 533
8.1.2 Methods . . . 534
8.1.2.1 GetFunctionalityState . . . 534
8.1.2.2 ListenForEvents . . . 535
8.1.2.3 NewRoutingCall . . . 536
8.1.2.4 Query . . . 537
8.1.2.5 StopListeningForEvents . . . 538

8.1.2.6 Update . . . 540
8.1.3 Events . . . 541
8.1.3.1 EventOccurred . . . 541
8.2 AssignedEvent . . . 542
8.2.1 Properties . . . 542
8.2.1.1 AgentExtension . . . 542
8.2.1.2 AgentKey . . . 543
8.2.1.3 CallID . . . 543
8.3 DequeuedEvent . . . 543
8.3.1 Properties . . . 544
8.3.1.1 CallID . . . 544
8.3.1.2 DequeueReason . . . 544
8.3.1.3 Description . . . 546
8.3.1.4 Priority . . . 546
8.3.1.5 QueueKey . . . 547
8.4 EnqueuedEvent . . . 547
8.4.1 Properties . . . 547
8.4.1.1 CallID . . . 548
8.4.1.2 Description . . . 548
8.4.1.3 Priority . . . 549
8.4.1.4 QueueKey . . . 549
8.5 RoutingBase . . . 550
8.5.1 Properties . . . 550
8.5.1.1 Type . . . 551
8.6 RoutingCall . . . 551
8.6.1 Properties . . . 551
8.6.1.1 AssignAgentKey . . . 552
8.6.1.2 CallID . . . 552
8.6.1.3 ContactData . . . 553
8.6.1.4 Description . . . 553
8.6.1.5 MediaType . . . 554
8.6.1.6 Priority . . . 554
8.6.1.7 QueueKey . . . 555
8.6.1.8 ReserveAgentKey . . . 555
8.6.1.9 ReserveAgentMaximumTime . . . 556
8.6.1.10 State . . . 556
8.6.2 Methods . . . 556
8.6.2.1 Dequeue . . . 557
8.6.2.2 Enqueue . . . 558
8.7 RoutingEvent . . . 560
8.7.1 Properties . . . 561
8.7.1.1 Code . . . 561
8.7.1.2 EventType . . . 562
8.7.1.3 MediaType . . . 562
8.7.1.4 ObjectType . . . 563
8.7.1.5 Resource . . . 563
8.8 RoutingInformationUpdatedEvent . . . 564
8.8.1 Properties . . . 564
8.8.1.1 CallID . . . 564
8.8.1.2 ContactData . . . 566
8.8.1.3 Description . . . 566
8.8.1.4 Priority . . . 567

8.8.1.5 QueueKey . . . 567
8.9 TimedOutEvent . . . 568
8.9.1 Properties . . . 568
8.9.1.1 CallID . . . 568
8.9.1.2 TelephoneNumber . . . 569
8.10 UnassignedEvent . . . 569
8.10.1 Properties . . . 569
8.10.1.1 AgentKey . . . 570
8.10.1.2 CallID . . . 570
9 Statistics Manager . . . 571
9.1 StatisticsManager object . . . 571
9.1.1 Properties . . . 571
9.1.1.1 State . . . 571
9.1.2 Methods . . . 572
9.1.2.1 GetFunctionalityState . . . 572
9.1.2.2 ListenForEvents . . . 573
9.1.2.3 StopListeningForEvents . . . 574
9.2 AggregateCumulativeElement . . . 574
9.2.1 Properties . . . 574
9.2.1.1 Abandoned . . . 575
9.2.1.2 AbandonedRate . . . 575
9.2.1.3 AggregateKey . . . 575
9.2.1.4 Answered . . . 576
9.2.1.5 AverageWaitTime . . . 576
9.2.1.6 MaximumWaitTime . . . 576
9.2.1.7 Received . . . 577
9.2.1.8 Redirected . . . 577
9.2.1.9 ServiceLevel . . . 577
9.3 AggregateCumulativeEvent . . . 578
9.3.1 Properties . . . 578
9.3.1.1 _NewEnum . . . 578
9.3.1.2 Count . . . 579
9.3.1.3 Item . . . 579
9.3.1.4 QueryID . . . 580
9.4 AggregateRealtimeElement . . . 581
9.4.1 Properties . . . 581
9.4.1.1 AbandonedRate . . . 581
9.4.1.2 AggregateKey . . . 582
9.4.1.3 AverageAbandonedWaitTime . . . 582
9.4.1.4 AverageAnsweredWaitTime . . . 583
9.4.1.5 Contacts . . . 583
9.4.1.6 EstimatedAnsweredWaitTime . . . 583
9.4.1.7 EstimatedServiceLevel . . . 584
9.4.1.8 OldestContactWaitTime . . . 584
9.4.1.9 OverflowedContacts . . . 584
9.4.1.10 ServiceLevel . . . 585
9.5 AggregateRealtimeEvent . . . 585
9.5.1 Properties . . . 585
9.5.1.1 Count . . . 586
9.5.1.2 Item . . . 586
9.5.1.3 QueryID . . . 587

9.6 AggregateWrapupReasonCumulativeElement . . . 588
9.6.1 Properties. . . . 588
9.6.1.1 AggregateKey . . . 588
9.6.1.2 Count . . . 589
9.6.1.3 WrapupReasonKey . . . 589
9.7 AggregateWrapupReasonCumulativeEvent . . . 589
9.7.1 Properties. . . . 590
9.7.1.1 _NewEnum . . . 590
9.7.1.2 Count . . . 591
9.7.1.3 Item. . . . 591
9.7.1.4 QueryID . . . 592
9.8 CallbackRealtimeElement . . . 593
9.8.1 Properties. . . . 593
9.8.1.1 AttemptCount . . . 593
9.8.1.2 CallID . . . 593
9.8.1.3 CustomerName . . . 594
9.8.1.4 Description . . . 594
9.8.1.5 Origin . . . 594
9.8.1.6 State . . . 595
9.8.1.7 TimeInState . . . 595
9.9 CallbackRealtimeEvent . . . 595
9.9.1 Properties. . . . 596
9.9.1.1 _NewEnum . . . 596
9.9.1.2 Count . . . 597
9.9.1.3 Item. . . . 597
9.9.1.4 QueryID . . . 598
9.10 ContactRealtimeElement . . . 599
9.10.1 Properties. . . . 599
9.10.1.1 CallID . . . 599
9.10.1.2 CurrentPriority . . . 599
9.10.1.3 CurrentState . . . 600
9.10.1.4 Description . . . 600
9.10.1.5 Destination . . . 600
9.10.1.6 MediaType . . . 601
9.10.1.7 QualifyingAgentsCount . . . 601
9.10.1.8 QueueKey. . . . 602
9.10.1.9 Source . . . 602
9.10.1.10 TimeInState . . . 602
9.10.1.11 TotalWaitTime . . . 603
9.11 ContactRealtimeEvent . . . 603
9.11.1 Properties. . . . 603
9.11.1.1 _NewEnum . . . 603
9.11.1.2 Count . . . 604
9.11.1.3 Item. . . . 605
9.11.1.4 QueryID . . . 606
9.12 GroupCumulativeElement . . . 606
9.12.1 Properties. . . . 606
9.12.1.1 ConsultOut . . . 607
9.12.1.2 GroupKey . . . 607
9.12.1.3 Offered . . . 607
9.12.1.4 Received. . . . 608
9.12.1.5 ReceivedHereOfferedElsewhere . . . 608

9.12.1.6 TransferOut . . . 608
9.13 GroupCumulativeEvent . . . 609
9.13.1 Properties . . . 609
9.13.1.1 _NewEnum . . . 609
9.13.1.2 Count . . . 610
9.13.1.3 Item . . . 610
9.13.1.4 QueryID . . . 611
9.14 GroupRealtimeElement . . . 613
9.14.1 Properties . . . 613
9.14.1.1 AwayUsers . . . 613
9.14.1.2 BusyUsers . . . 613
9.14.1.3 CallsWaiting . . . 614
9.14.1.4 GroupKey . . . 614
9.14.1.5 HandlingDirectUsers . . . 614
9.14.1.6 HandlingRoutedUsers . . . 615
9.14.1.7 IdleUsers . . . 615
9.14.1.8 LoggedOnUsers . . . 615
9.15 GroupRealtimeEvent . . . 616
9.15.1 Properties . . . 616
9.15.1.1 _NewEnum . . . 616
9.15.1.2 Count . . . 617
9.15.1.3 Item . . . 617
9.15.1.4 QueryID . . . 618
9.16 QueueCumulativeElement . . . 619
9.16.1 Properties . . . 619
9.16.1.1 Abandoned . . . 619
9.16.1.2 AbandonedRate . . . 619
9.16.1.3 Answered . . . 620
9.16.1.4 AverageWaitTime . . . 620
9.16.1.5 MaximumWaitTime . . . 620
9.16.1.6 QueueKey . . . 621
9.16.1.7 Received . . . 621
9.16.1.8 Redirected . . . 621
9.16.1.9 ServiceLevel . . . 622
9.17 QueueCumulativeEvent . . . 622
9.17.1 Properties . . . 622
9.17.1.1 _NewEnum . . . 623
9.17.1.2 Count . . . 624
9.17.1.3 Item . . . 624
9.17.1.4 QueryID . . . 625
9.18 QueueRealtimeElement . . . 626
9.18.1 Properties . . . 626
9.18.1.1 AbandonedRate . . . 626
9.18.1.2 AverageAbandonedWaitTime . . . 626
9.18.1.3 AverageAnsweredWaitTime . . . 627
9.18.1.4 Contacts . . . 627
9.18.1.5 EstimatedAnsweredWaitTime . . . 627
9.18.1.6 EstimatedServiceLevel . . . 628
9.18.1.7 OldestContactWaitTime . . . 628
9.18.1.8 OverflowedContacts . . . 628
9.18.1.9 QueueKey . . . 629
9.18.1.10 ServiceLevel . . . 629

9.19 QueueRealtimeEvent . . . 630
9.19.1 Properties . . . 630
9.19.1.1 _NewEnum . . . 630
9.19.1.2 Count . . . 632
9.19.1.3 Item . . . 632
9.19.1.4 QueryID . . . 633
9.20 QueueWrapupReasonCumulativeElement . . . 634
9.20.1 Properties . . . 634
9.20.1.1 Count . . . 634
9.20.1.2 QueueKey . . . 635
9.20.1.3 WrapupReasonKey . . . 635
9.21 QueueWrapupReasonCumulativeEvent . . . 635
9.21.1 Properties . . . 636
9.21.1.1 _NewEnum . . . 636
9.21.1.2 Count . . . 637
9.21.1.3 Item . . . 637
9.21.1.4 QueryID . . . 638
9.22 StatisticsEvent . . . 639
9.22.1 Properties . . . 640
9.22.1.1 Code . . . 640
9.22.1.2 EventType . . . 641
9.22.1.3 ObjectType . . . 641
9.22.1.4 Resource . . . 642
9.23 TimeRange . . . 642
9.23.1 Properties . . . 643
9.23.1.1 _NewEnum . . . 643
9.23.1.2 Count . . . 644
9.23.1.3 Item . . . 645
9.23.1.4 TimeRange . . . 646
9.24 UserCumulativeElement . . . 647
9.24.1 Properties . . . 647
9.24.1.1 AbandonedWhileRinging . . . 647
9.24.1.2 ConsultedOut . . . 647
9.24.1.3 Disconnected . . . 648
9.24.1.4 Handled . . . 648
9.24.1.5 MaximumRoutedHandlingTime . . . 648
9.24.1.6 Offered . . . 649
9.24.1.7 Requeued . . . 649
9.24.1.8 RoutedHeld . . . 649
9.24.1.9 TotalAwayTime . . . 651
9.24.1.10 TotalBusyTime . . . 651
9.24.1.11 TotalHandledTime . . . 651
9.24.1.12 TotalHoldTime . . . 652
9.24.1.13 TotalIdleTime . . . 652
9.24.1.14 TotalLoggedOnTime . . . 652
9.24.1.15 TotalPendingTime . . . 653
9.24.1.16 TotalPostProcessingTime . . . 653
9.24.1.17 TotalRingTime . . . 653
9.24.1.18 TotalTalkTime . . . 654
9.24.1.19 TotalTimeOther . . . 654
9.24.1.20 TransferredOut . . . 654
9.24.1.21 Undelivered . . . 655

9.24.1.22 Unhandled . . . 655
9.24.1.23 UserKey . . . 655
9.24.1.24 Utilization . . . 656
9.25 UserCumulativeEvent . . . 656
9.25.1 Properties . . . 656
9.25.1.1 _NewEnum . . . 656
9.25.1.2 Count . . . 657
9.25.1.3 Item . . . 658
9.25.1.4 QueryID . . . 659
9.26 UserRealtimeElement . . . 659
9.26.1 Properties . . . 659
9.26.1.1 Extension . . . 660
9.26.1.2 HandlingStates . . . 660
9.26.1.3 MaximumUnusedCapacity . . . 661
9.26.1.4 MinimumUnusedCapacity . . . 661
9.26.1.5 PresenceState . . . 661
9.26.1.6 RoutingState . . . 662
9.26.1.7 RoutingStateReasonKey . . . 662
9.26.1.8 TimeInPresenceState . . . 662
9.26.1.9 TimeInRoutingState . . . 663
9.26.1.10 UserKey . . . 663
9.27 UserRealtimeEvent . . . 663
9.27.1 Properties . . . 664
9.27.1.1 _NewEnum . . . 664
9.27.1.2 Count . . . 665
9.27.1.3 Item . . . 665
9.27.1.4 QueryID . . . 666
9.28 UserRTHandlingState . . . 666
9.28.1 Properties . . . 667
9.28.1.1 AssociatedQueueKey . . . 667
9.28.1.2 CallID . . . 667
9.28.1.3 ContactDescription . . . 668
9.28.1.4 ContactType . . . 668
9.28.1.5 HandlingState . . . 668
9.28.1.6 HandlingStateReason . . . 669
9.28.1.7 IsPrimary . . . 669
9.28.1.8 SequenceNumber . . . 669
9.28.1.9 TimeInHandlingState . . . 670
9.28.1.10 TotalContactActiveTime . . . 670
9.28.1.11 TotalContactHandlingTime . . . 670
9.29 UserRTHandlingStates . . . 671
9.29.1 Properties . . . 671
9.29.1.1 _NewEnum . . . 671
9.29.1.2 Count . . . 672
9.29.1.3 Item . . . 672
9.30 UserWrapupReasonCumulativeElement . . . 673
9.30.1 Properties . . . 673
9.30.1.1 Count . . . 674
9.30.1.2 UserKey . . . 674
9.30.1.3 WrapupReasonKey . . . 674
9.31 UserWrapupReasonCumulativeEvent . . . 675
9.31.1 Properties . . . 675

9.31.1.1 _NewEnum 675
9.31.1.2 Count 676
9.31.1.3 Item. 676
9.31.1.4 QueryID 677
Index 679

1 About this guide

This guide provides reference information for the OpenScape Contact Center Software Development Kit (SDK). The information provided in this guide is generally structured as shown in the following diagram.

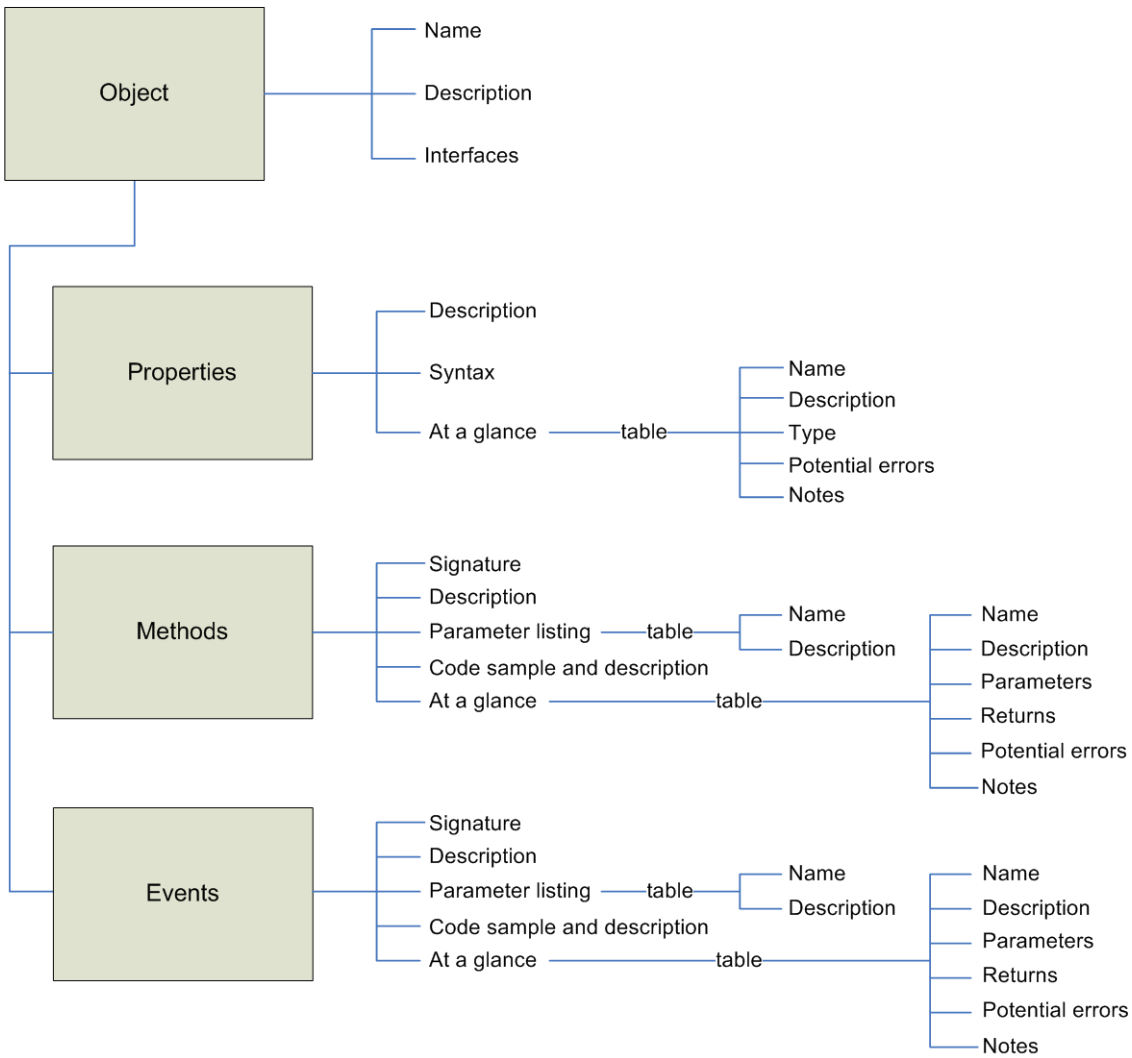

Figure 1 SDK Programming Guide reference information structure

1.1 Who should use this guide

This guide is intended for system integrators and application developers who intend to use the SDK to develop multimedia applications that integrate with the OpenScape Contact Center system.

1.2 Formatting conventions

The following formatting conventions are used in this guide:

Bold
This font identifies <System Group> components, window and dialog box titles, and item names.

Italic
This font identifies references to related documentation.

`Monospace Font`
This font distinguishes text that you should type, or that the computer displays in a message.

NOTE: Notes emphasize information that is useful but not essential, such as tips or alternative methods for performing a task.

IMPORTANT: Important notes draw special attention to actions that could adversely affect the operation of the application or result in a loss of data.

1.3 Documentation feedback

To report an issue with this document, call the Customer Support Center.

When you call, be sure to include the following information. This will help identify which document you are having issues with.

- **Title:** SDK Programming Guide
- **Order Number:** A31003-S22B1-R102-02-7620

2 About the OpenScape Contact Center SDK

The OpenScape Contact Center SDK is a 32-bit COM control that exposes OpenScape Contact Center functionality to custom applications through a set of interfaces so that developers can create a fully integrated desktop application featuring voice, e-mail, callback, and Web collaboration functionality.

The SDK provides application developers with the tools they will need to build applications that integrate with the OpenScape Contact Center system and can be used with any language that properly implements COM, including Visual Basic and Visual C++.

NOTE: We strongly recommend that system integrators and application developers update their SDK-based applications to use the latest interfaces. Older interfaces contain obsolete functionality that might not work correctly with the latest version of the product and these interfaces might be removed in subsequent releases.

2.1 System requirements

The OpenScape Contact Center SDK can be installed on a machine that is running one of the following Microsoft Windows operating systems.

- Windows Server 2012 R2 Standard or Datacenter
- Windows Server 2012 Standard or Datacenter
- Windows Server 2008 R2 Standard or Enterprise Edition with Service Pack 1
- Windows 10 Professional or Enterprise Edition
- Windows 8 or 8.1 Professional or Enterprise Edition or later
- Windows 7 Professional or Enterprise Edition or later
- Windows Vista Business or Enterprise Edition with Service Pack 2 or later

---

NOTE: For any application that you develop with the SDK, the server and client machines must meet the system requirements described in the *Installation Guide.*

2.2 Installing the OpenScape Contact Center SDK

This section describes how to install the SDK for the logged-on user. To install the SDK for all users, see Section 2.2.1, "Installing the SDK for all users", on page 31.

The setup program installs the following two products on the developer machine:

- **OpenScape Contact Center SDK** — Contains the SDK samples and documentation.
- **OpenScape Contact Center SDK Runtime** — Contains the files needed for the SDK to communicate with an OpenScape Contact Center server.

To install the OpenScape Contact Center SDK:

1. Insert the medium that contains the OpenScape Contact Center SDK.
2. Browse to the **OpenScape Contact Center SDK** folder, and then double-click **setup.exe**. This launches the OpenScape Contact Center SDK setup program, which will guide you through the rest of the installation process.
3. In the **Welcome** dialog box, click **Next**.
4. In the **Destination** dialog box, click **Next**.

   IMPORTANT: We strongly recommend that you do not change the default location unless advised to do so by your support representative.

5. In the **Setup Type** dialog box, select one of the following options, and then click **Next**:
   - **Complete** — Select this option to install all the SDK program features to the default location. This is the recommended option.
   - **Custom** — Select this option to choose which program features you want installed. This option is for advanced users only.
6. In the **Ready to Install** dialog box, click **Install** to begin the installation.
7. When the installation process is complete, click **Finish**.

2.2.1 Installing the SDK for all users

If the SDK is already installed and you want to install it for all users, you must first uninstall the SDK and then reinstall it as described here.

To install the SDK for all users:

1. Open a command prompt window.
2. Type the following and then press **ENTER**.

```
msiexec /i "<full_path>\HiPath ProCenter SDK.msi" ALLUSERS=1
```

2.3 Redistributing and patching the SDK

The OpenScape Contact Center SDK Runtime is automatically installed on the developer machine during the SDK installation and must be redistributed and installed on each machine that is running an SDK-based application.

NOTE: The SDK Runtime Setup Program is included in the **OpenScape Contact Center SDK Runtime** folder on the OpenScape Contact Center DVD.

When necessary, patches for the SDK Runtime product will be created and distributed. You can install a patch on the developer machine, and also redistribute it to a site to patch the SDK Runtime product on the machines that are running the SDK-developed application.

If the SDK samples and documentation need to be updated, OpenScape Contact Center SDK patches will be created and distributed separately from the OpenScape Contact Center SDK Runtime patches.

NOTE: The OpenScape Contact Center server and the OpenScape Contact Center SDK Runtime product must be at a compatible patch level.

2.4 Writing SDK applications in a multitenant environment

The OpenScape Contact Center system can be used in a multitenant environment containing separate business units. If you are working in a multitenant environment, you must log on to a specific business unit.

In a multitenant environment, the business unit name is a required parameter in the Initialize method. The Initialize method is always called prior to calling any other methods. In a non-multitenant environment, the business unit name parameter is optional and is ignored. For more information on the Initialize method, see Section 5.1.2.8, “Initialize”, on page 159.

2.5 What the SDK includes

The SDK includes five Managers. As shown in the following diagram, the Managers in the SDK abstract a server away from the user so that the user does not need to know which OpenScape Contact Center server is required for which function.

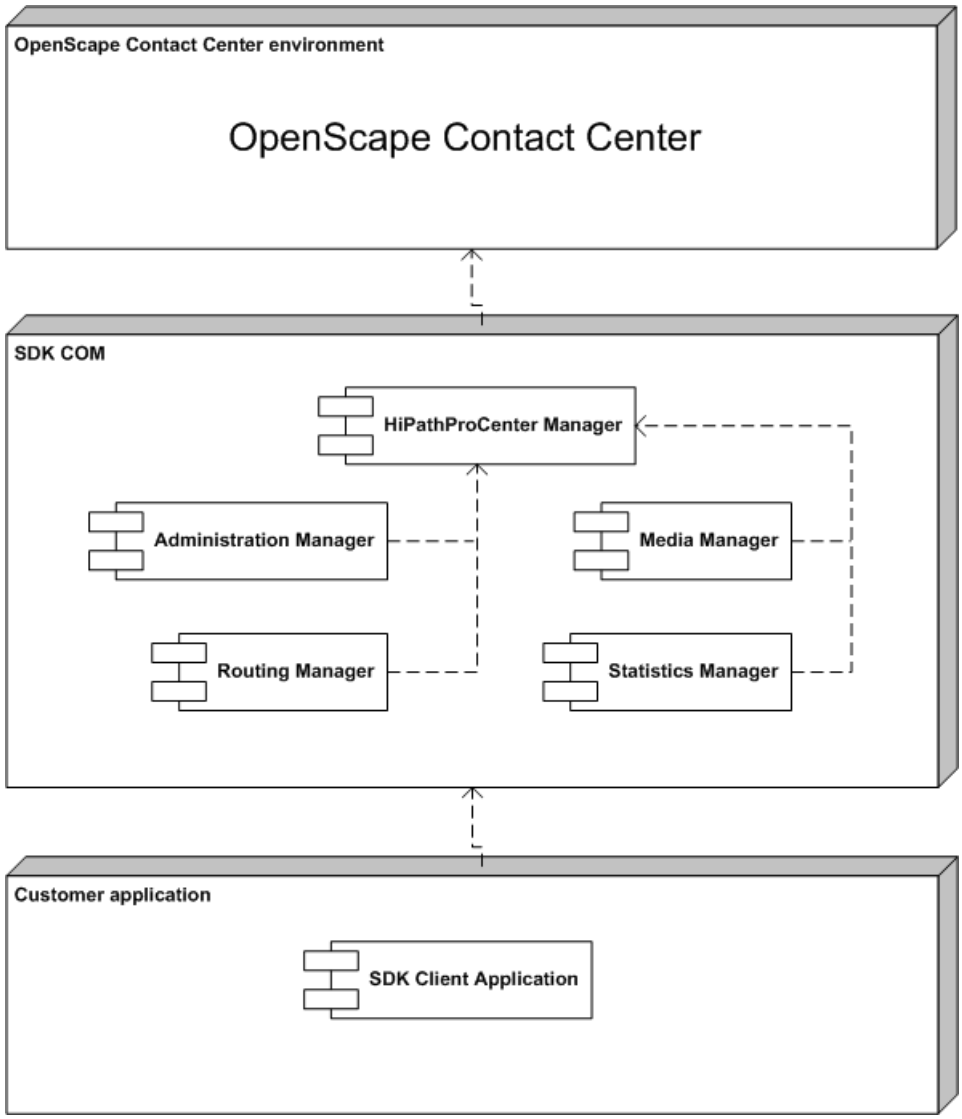

Figure 2 OpenScape Contact Center SDK Managers

Manager	Description
OpenScape Contact Center Manager	The OpenScape Contact Center Manager is responsible for the following: ● Creating the other Managers ● Logging on to the OpenScape Contact Center system ● Providing basic status and administrative functionality such as logging on to the OpenScape Contact Center system ● Providing a single point of access into the OpenScape Contact Center system by maintaining all the pointers to the servers, enabling them to be shared throughout the rest of the SDK ● Handling event registration
Administration Manager	The Administration Manager is the access point to any administration information that is stored in the OpenScape Contact Center system. The Administration Manager is responsible for the following: ● Read-only access to information about OpenScape Contact Center users, queues, Unavailable reasons, Work reasons, Delete reasons, and Retry reasons. ● Dynamic update events to indicate when these collections have been updated
Media Manager	The Media Manager is responsible for the following: ● Events and objects for each media type — voice, callback, e-mail, and Web collaboration (media-specific contact objects can be used to interact with the contact) ● A list of the current contacts in the system ● Agent status information
Routing Manager	The Routing Manager is responsible for the following: ● Events generated by routing actions ● A routing object used to enqueue or dequeue contacts and query contacts queued by the Routing Server
Statistics Manager	The Statistics Manager is responsible for the following: ● An event-based model for retrieval of real-time statistics, current statistics (snapshot of pertinent statistics) ● Cumulative statistics (historical, over the last 24 hour period)

Table 1 OpenScape Contact Center SDK Managers

2.6 Creating the Manager objects

A Manager object can be created either with events or without events. There is also a difference between creating the OpenScape Contact Center Manager and creating the other Manager objects (Administration, Media, Routing, and Statistics).

2.6.1 Creating a Manager object with events

2.6.1.1 Creating the object as a global object

If you want to receive events, you must create the object as a global object. For example, in your global area, you should have one or more of the following lines:

- For the OpenScape Contact Center Manager:

```
Public WithEvents g_oHiPathProCenterManager as
HiPathProCenterLibrary.HiPathProCenterManager
```

- For another Manager, such as Administration Manager:

```
Public WithEvents g_oAdministrationManager as
HiPathProCenterLibrary.AdministrationManager
```

Use the appropriate HiPathProCenterLibrary object type for the other Managers.

2.6.1.2 Before using a Manager object

Before using a Manager object, you must do the following:

- For the OpenScape Contact Center Manager:

```
Set g_oHiPathProCenterManager = new
HiPathProCenterLibrary.HiPathProCenterManager
```

- For another Manager, such as Administration Manager:

```
Set g_oAdministrationManager =
g_oHiPathProCenterManager.HireAdministrationManager
(EventMode_FireEvents)
```

Use the appropriate OpenScape Contact Center Manager creation methods for the other Managers. For more information, see Section 5.1.2.4, “HireAdministrationManager”, on page 154 through Section 5.1.2.7, “HireStatisticsManager”, on page 158.

2.6.1.3 Creating an event handler automatically

By creating the Managers with events, Visual Basic automatically creates the event handler through which all events from the appropriate Managers will be sent. These subroutines adhere to the following format:

- For the OpenScape Contact Center Manager:

```
Private Sub g_oHiPathProCenterManager_EventOccurred (ByVal
HiPathProCenterEvent as
HiPathProCenterLibrary.HiPathProCenterEvent)
```

- For another Manager, such as Administration Manager:

```
Private Sub g_oAdministrationManager_EventOccurred (ByVal
AdministrationEvent as
HiPathProCenterLibrary.AdministrationEvent)
```

2.6.2 Creating a Manager object without events

If you do not want to receive events, there are two ways you can do this. The recommended way is to use the same event method, but omit the 'WithEvents' portion. For example:

- For the OpenScape Contact Center Manager:

```
Public g_oHiPathProCenterManager as
HiPathProCenterLibrary.HiPathProCenterManager
```

- For another Manager, such as Administration Manager:

```
Public g_oAdministrationManager as
HiPathProCenterLibrary.AdministrationManager
```

Use the appropriate HiPathProCenterLibrary object type for the other Managers.

You will notice in the previous example that the 'WithEvents' portion has been omitted. However, you still need to allocate the memory for the Manager objects as follows:

- For a OpenScape Contact Center Manager:

```
Set g_oHiPathProCenterManager = new
HiPathProCenterLibrary.HiPathProCenterManager
```

- For another Manager, such as Administration Manager:

```
Set g_oAdministrationManager =
g_objHiPathProCenterManager.HireAdministrationManager
(EventMode_IgnoreEvents)
```

Use the appropriate HiPathProCenterManager creation methods for the other Managers. For more information, see Section 5.1.2.4, "HireAdministrationManager", on page 154 through Section 5.1.2.7, "HireStatisticsManager", on page 158.

When you create the Manager objects, you pass an EventMode_IgnoreEvents. It is critical that you match the state in which you have declared the object (with or without events) with the way in which you have created the object.

2.6.3 Creating a Manager object locally

The other way to create a Manager is to create it locally in a subroutine. This can be accomplished as follows:

- For the OpenScape Contact Center Manager:

```
Dim oHiPathProCenterManager as New
HiPathProCenterLibrary.HiPathProCenterManager
```

- For another Manager, such as Administration Manager:

```
Dim oAdministrationManager as
HiPathProCenterLibrary.AdministrationManager

Set objAdministrationManager =
g_objHiPathProCenterManager.HireAdministrationManager
(EventMode_IgnoreEvents)
```

This method is only recommended if you are using the object for a single method call in this one and only method. There is some overhead associated with creating the Managers, and you would not want to do this every time you perform a function.

2.7 Destroying the Manager objects

There is one restriction when using the SDK. Since the Manager objects share some common resources, they must be destroyed in order. The OpenScape Contact Center Manager is destroyed last. All the other Manager objects must be destroyed first. It is recommended that you add the following lines before shutting down your application.

```
Set g_oAdministrationManager = Nothing
Set g_oMediaManager = Nothing
Set g_oRoutingManager = Nothing
Set g_oStatisticsManager = Nothing
Set g_oHiPathProCenterManager = Nothing
```

You can include the lines at any time during the execution of the application, but they should be added in the order shown. For example, you can create the Administration Manager, destroy it elsewhere in your code, and then recreate it. However, before you exit the application, you must destroy it.

2.8 About events

Events are sent from the various servers in the system when certain things happen. For example, when the telephone starts to ring, the T-Server sends the Delivered event to any applications that are registered for the Delivered event for that particular telephone.

2.8.1 Listening for events

To allow applications to receive these events, each of the Manager objects must use the following method:

```
ListenForEvents (EventType as enEventTypes, (Optional) Resource
as String = "*")
```

Use a different enEventType enumeration for each of the various objects.

2.8.2 Stop listening for events

To allow the application to stop receiving events, you use the following method:

```
StopListeningForEvents (EventType as enEventTypes, (Optional)
Resource as String = "*")
```

Use a different enEventType enumeration for each of the various objects.

When you use the StopListeningForEvents method, it is important that you use the same resource that you used when calling ListenForEvents.

2.8.3 ListenForEvent and StopListeningForEvents parameters

ListenForEvent and StopListeningForEvents both make use of a type parameter and a resource parameter.

2.8.3.1 Type parameter

Both of these methods accept the event type that the application wants to listen for. For example, if you are connected to the Media Manager, you could pass in the event type MediaEventType_VoiceEvents to listen for the events for the telephone (that is, when it is ringing, answered, and so on).

2.8.3.2 Resource parameter

The resource parameter is the resource on which you want to register for the event. The resource will vary depending on the event that you are registering for.

For example, if you are registering for the MediaEventType_VoiceEvents, then you pass the device that you want to register for the events from. An example is the extension of the telephone.

If you are registering for MediaEventType_AgentMessages events, then you pass the user key of the user you want to receive messages for. If you want to receive all events, you can use the '*' as the resource, and you will receive all events for that event type. This can cause a significant increase in network traffic and should be used sparingly for most event types.

For more information on the resource that should be used for each of the event types, see the following:

- Section 3.4, “enAdministrationEventTypes”, on page 47
- Section 3.22, “enHiPathProCenterEventObjectTypes”, on page 84
- Section 3.32, “enMediaEventTypes”, on page 98
- Section 3.41, “enRoutingEventTypes”, on page 106

2.8.4 Examples of events

This section provides several examples of events.

2.8.4.1 Example 1: Making calls

If you make the following calls:

```
Call g_oMediaManager.ListenForEvents
(MediaEventType_VoiceEvents, "1203")
```

then later:

```
Call g_oMediaManager.StopListeningForEvents
(MediaEventType_VoiceEvents, "1203")
```

After making the ListenForEvents call, you begin receiving voice events for extension 1203 (for example, when someone takes that telephone off the hook, or when that telephone is ringing). After making the StopListeningForEvents request, you no longer receive any voice events for extension 1203.

2.8.4.2 Example 2: Receiving all voice events

If you want to receive all voice events for all extensions, you could do the following:

```
Call g_oMediaManager.ListenForEvents
(MediaEventType_VoiceEvents, "*")
```

then later:

```
Call g_oMediaManager.StopListeningForEvents
(MediaEventType_VoiceEvents, "*")
```

After making the ListenForEvents call, you begin receiving voice events for all extensions, since you have used ‘*’ as the resource.

After making the StopListeningForEvents request, you no longer receive any voice events for any extensions because you have used '*' as the resource again. Since this is the same resource you registered for, it will effectively erase the previous ListenForEvents request with '*' as the resource. In this instance, you could omit '*' and the application will register for the event with '*' by default.

2.8.4.3 Example 3: Receiving voice events for an extension

In this example, you register for voice events for extension 1203, as well as a call to ListenForEvents for all extensions (in this case, you use the default value of the optional strResource parameter):

```
Call g_oMediaManager.ListenForEvents
(MediaEventType_VoiceEvents, "1203")

Call g_oMediaManager.ListenForEvents
(MediaEventType_VoiceEvents)
```

then later:

```
Call g_objMediaManager.StopListeningForEvents
(MediaEventType_VoiceEvents)
```

After making the first ListenForEvents call, you begin receiving voice events for extension 1203. After making the second ListenForEvents call, you receive an event whenever anything happens on any extension.

When something happens on extension 1203, you receive two events — one for the request to listen for events for extension 1203, and one for the request to listen for events for all extensions. After making the StopListeningForEvents for all extensions, you stop getting voice events for all extensions, but you still get the voice events for extension 1203.

Each time you call ListenForEvents with a different resource, you create a new instance to receive an event. The use of the '*' is really a specific instance that is independent of the individual requests you have made (that is, from our example, the 1203 registration). It is its own entity and does not affect the other ListenForEvents requests that you have made.

2.9 Handling connectivity issues

If connectivity is lost between an SDK-based application and an OpenScape Contact Center server (or the entire OpenScape Contact Center server machine), the SDK automatically reestablishes connectivity with the appropriate servers. However, it is important to understand that an interruption in connectivity can last for some time, depending on the type of issue — for example, an individual server failure, a failover of the system in a high availability (warm standby) deployment, or network connectivity issues. Therefore, we recommend that an SDK-based application implement the code required to handle such scenarios, and the resulting interruption in application functionality, and recover gracefully to a normal state.

The SDK provides notifications about changes in the server states, including connectivity loss, by way of the ServerStateChangedEvent. We recommend that an application provide logic to react to all reported server state changes.

For information on the ServerStateChangedEvent, see Section 5.3, “ServerStateChangedEvent”, on page 173. For information on event handling, see Section 5.1.3.1, “EventOccurred”, on page 166.

When the SDK reestablishes connectivity with a server, the SDK-based application should resynchronize application state information, so that the application can continue with the operations it was performing when the interruption occurred. For example, if the application is tracking an active contact when an interruption occurs, after connectivity is reestablished, the application should use the Query()

method to confirm that the contact is still active in the system, and in an appropriate state, before performing any new operations on it (such as disconnecting the contact).

3 Enumerations

This section describes the enumerations provided by the SDK. The SDK uses enumerations whenever there are a fixed number of values. For example, any types, states, or codes are returned through enumerations.

3.1 enAdministrationBaseObjectTypes

The enAdministrationObjectTypes enumeration represents the various types of objects that can be passed into the main methods of the AdministrationManager (that is, into methods such as Query). These are used with the AdministrationBase interface to provide a standard method of passing Administration objects. This is useful for event handling, as well as for passing parameters into common methods (as seen in the AdministrationManager).

When you have an AdministrationBase object, you can retrieve the interface to the underlying type by setting a variable of the underlying type to the AdministrationBase object. For example:

```
Dim objQueue as HiPathProCenterLibrary.Queue
If (objAdminBase.ObjectType =
   AdministrationBaseObjectType_Queue) Then
   Set objQueue = objAdminBase
   '...then can call the various queue methods
   ' and properties as required
Endif
```

Enumeration	Value
AdministrationBaseObjectType_Aggregate An AdministrationBase object is a simplified interface to the Aggregate object.	27
AdministrationBaseObjectType_Base This object type does not correspond to an acceptable Administration object type. This should never be seen.	0
AdministrationBaseObjectType_CallbackReason An AdministrationBase object is a simplified interface to the callbackReason object. The reason could be either a callback Retry reason or Delete reason that is uniquely identified by the Key property.	26
AdministrationBaseObjectType_Department	31
AdministrationBaseObjectType_EmailDiscardReason	33
AdministrationBaseObjectType_EmailMessageTemplate	35

Table 2 enAdministrationObjectTypes enumeration (Seite 1 von 2)

Enumeration	Value
`AdministrationBaseObjectType_Group` An AdministrationBase object is a simplified interface to the Group object.	28
`AdministrationBaseObjectType_Language`	34
`AdministrationBaseObjectType_PostProcessingReason` This indicates that an AdministrationBase object is a simplified interface to the Post-processing object.	37
`AdministrationBaseObjectType_Queue` This indicates that an AdministrationBase object is a simplified interface to the Queue object.	6
`AdministrationBaseObjectType_RoutingStateReason` An AdministrationBase object is a simplified interface to the RoutingStateReason object. The routing state reason could be either an Unavailable reason or a Work reason that is uniquely identified by the RoutingStateReasonKey property.	25
`AdministrationBaseObjectType_User` An AdministrationBase object is a simplified interface to the User object.	24
`AdministrationBaseObjectType_WebCollaborationTemplate`	36
`AdministrationBaseObjectType_WrapupReason`	32

Table 2 enAdministrationObjectTypes enumeration (Seite 2 von 2)

3.2 enAdministrationEventCodes

The enAdministrationEventCodes enumeration represents the various types of events that can occur in the AdministrationManager. To receive these events, you must have created the AdministrationManager with events (that is, EventMode_FireEvents), and registered for an event type through the ListenForEvents method, passing in a valid enAdministrationEventType.

Enumeration	Value
`AdministrationEventCode_Added` An object has been added to the database. To receive this, you only have to start the AdministrationManager with events and listen for AdministrationEventType_AdministrationDatabase.	2102
`AdministrationEventCode_Deleted` An object has been deleted from the database. To receive this, you only have to start the AdministrationManager with events and listen for AdministrationEventType_AdministrationDatabase.	2104

Table 3 enAdministrationEventCodes enumeration (Seite 1 von 2)

Enumeration	Value
`AdministrationEventCode_ManagerStateChanged` A ManagerStateChanged event has occurred. This is sent when the Administration Manager has a necessary background server that has either gone down or come back up. To receive this, you only have to start the AdministrationManager with events. You do not need to explicitly listen for any events.	2003
`AdministrationEventCode_None` An AdministrationEvent does not correspond to something that has happened. This should never be seen.	2000
`AdministrationEventCode_Updated` An object has been updated in the database. To receive this, you only have to start the AdministrationManager with events and listen for AdministrationEventType_AdministrationDatabase.	2103

Table 3 enAdministrationEventCodes enumeration (Seite 2 von 2)

3.3 enAdministrationEventObjectTypes

The enAdministrationEventObjectTypes enumeration represents the various types of event objects that can be returned from the AdministrationManager. When an event occurs in the AdministrationManager, the event is returned through the EventOccurred event handler from the AdministrationManager as an AdministrationEvent object. You can then check the ObjectType property to determine what type of underlying object you have.

Enumeration	Value
`AdministrationEventObjectType_Aggregate` An AdministrationEvent corresponds to an Aggregate object. This is returned if an Aggregate is added or updated in the database after registering for ConfigurationUpdate events.	29
`AdministrationEventObjectType_Base` An AdministrationEvent does not correspond to a valid object. This should never be seen.	0
`AdministrationEventObjectType_CallbackDeleteReason` An Administration event corresponds to a CallbackReason object. This is returned if a Callback Delete reason is added or updated in the database after registering for ConfigurationUpdate events.	28
`AdministrationEventObjectType_CallbackRetryReason` An AdministrationEvent corresponds to a CallbackReason object. This is returned if a callback Retry reason is added or updated in the database after registering for ConfigurationUpdate events.	27

Table 4 enAdministrationEventObjectTypes enumeration (Seite 1 von 2)

Enumeration	Value
`AdministrationEventObjectType_DeleteList` An AdministrationEvent corresponds to a Delete List object. This is returned if one or more objects of the same type is deleted from the database after registering for ConfigurationUpdate events.	9
`AdministrationEventObjectType_Department`	31
`AdministrationEventObjectType_EmailDiscardReason`	33
`AdministrationEventObjectType_EmailMessageTemplate`	35
`AdministrationEventObjectType_Group` An AdministrationEvent corresponds to a Group object. This is returned if a Group is added or updated in the database after registering for ConfigurationUpdate events.	30
`AdministrationEventObjectType_Language`	34
`AdministrationEventObjectType_ManagerStateChanged` An Administration event corresponds to a ManagerStateChanged object. This is returned when there is a change in the Manager state, such as when you have created the AdministrationManager with events.	9899
`AdministrationEventObjectType_PostProcessingReason` An AdministrationEvent corresponds to a PostProcessingReason object. This is returned if a Post-processing reason is added or updated in the database after registering for ConfigurationUpdate events.	39
`AdministrationEventObjectType_Queue` An AdministrationEvent corresponds to a Queue object. This is returned if a queue is added or updated in the database after registering for ConfigurationUpdate events.	25
`AdministrationEventObjectType_RoutingUnavailableReason` An AdministrationEvent corresponds to a RoutingStateReason object. This is returned if an Unavailable reason is added or updated in the database after registering for ConfigurationUpdate events.	37
`AdministrationEventObjectType_User` An AdministrationEvent corresponds to a User object. This is returned if a user is added or updated in the database after registering for ConfigurationUpdate events.	24
`AdministrationEventObjectType_WebCollaborationTemplate`	36
`AdministrationEventObjectType_RoutingWorkReason` An AdministrationEvent corresponds to a RoutingStateReason object. This is returned if a Work reason is added or updated in the database after registering for ConfigurationUpdate events.	38
`AdministrationEventObjectType_WrapupReason`	32

Table 4 enAdministrationEventObjectTypes enumeration (Seite 2 von 2)

3.4 enAdministrationEventTypes

The enAdministrationEventTypes enumeration represents the various types of event groups that can be listened for from the AdministrationManager. When these events are passed into the ListenForEvents and StopListeningForEvents methods, the AdministrationManager will register for a variety of individual events. For more information, see Section 3.2, “enAdministrationEventCodes”, on page 44.

Enumeration	Value
`AdministrationEventType_AdministrationDatabaseUpdates` The group of events related to database updates in the AdministrationManager. These include such things as users getting updated or queues getting deleted. The resource used to register for these events should be '*'. However, you can use another resource if necessary.	2100
`AdministrationEventType_ConfigurationSynchronizationEvents` The group of events related to configuration synchronization updates from the AdministrationManager. Synchronization updates occur when the database and the information contained in the communication platform are synchronized. The resource used to register for these events should be '*'. However, you can use another resource if necessary.	2200
`AdministrationEventType_NotSet` The event that is received does not correspond to an event that has been explicitly listened for (that is, a ManagerStateChanged event).	2000

Table 5 enAdministrationEventTypes enumeration

3.5 enAdministrationFunctionalities

The enAdministrationFunctionalities enumeration represents the various function groups that exist in the AdministrationManager. This is used when asking the AdministrationManager for the functionality that is supported. You should check to see if a particular functionality is available before allowing related actions in your application.

For example, if you know that the database is unavailable, you may want to prevent people from attempting to add or update items using the AdministrationManager.

Enumeration	Value
`AdministrationFunctionality_AdministrationDatabase` This represents the database functionality group. This includes such things as updating a user or asking for the Queue Summaries collection.	1
`AdministrationFunctionality_ConfigurationSynchronization` This represents the Configuration Synchronization functionality group. This includes such things as notifications of synchronization between the database and the information contained in the communication platform.	2

Table 6 enAdministrationFunctionalities enumeration

3.6 enCallbackOrigins

The enCallbackOrigins enumerations represents the callback origin.

Enumeration	Value
`CallbackOrigin_Abandoned`	4
`CallbackOrigin_All`	7
`CallbackOrigin_Desktop`	2
`CallbackOrigin_IVR`	1
`CallbackOrigin_None`	0
`CallbackOrigin_Outbound`	5
`CallbackOrigin_SDK`	6
`CallbackOrigin_WebCollaboration`	3

Table 7 enCallbackOrigins enumeration

3.7 enCallStates

The enCallStates enumeration represents the various states that a contact can be in.

Enumeration	Value
`CallState_Abandoned` The contact was disconnected before being addressed by the system or a user.	24
`CallState_Accepted` A callback is accepted by a user.	35
`CallState_Assigned` The call is assigned to a user, but has not yet been routed to the user.	11
`CallState_AssignmentSuspended` A call is in the suspended state. This occurs when all the users who are eligible for the call at this step are also in the suspended state (that is, they are handling call routed by the communication platform, not calls routed by the system).	18
`CallState_AutoAcknowledged` The e-mail state when an e-mail message has had an auto-acknowledgment sent. An auto-acknowledgment is a notification that can be sent from the E-mail Server to the customer, stating that the e-mail message is received by the contact center, perhaps with a message stating the normal response times.	7
`CallState_Calling` A call is in the Calling state. This occurs when a telephone is placing an outgoing call. At this point, the call associated with that device (the telephone) is in the Calling state.	2
`CallState_Conferenced` A call is in the Conferenced state. This occurs after a customer is connected with a contact center user, and this first contact center user consults with a second contact center user, and then the first contact center user conferences in the second contact center user to the call.	20
`CallState_Connected` A call is connected between two or more parties. This occurs when a customer is connected with a contact center user.	15
`CallState_Consulting` A call is in the Consulting state. This occurs after a customer is connected with a contact center user, and that contact center user consults with another party.	19
`CallState_Deferred` An e-mail call is deferred by a user.	16
`CallState_Deleted` A callback call is deleted from the Callback Server.	29
`CallState_Delivered` The call has arrived at the E-mail Server of the OpenScape Manager.	12

Table 8 enCallStates enumeration (Seite 1 von 4)

Enumeration	Value
`CallState_Dequeued` A call is dequeued. This occurs when a call previously enqueued to the system is dequeued (for example, the caller has disconnected).	8
`CallState_Discarded` An e-mail message has been discarded. An e-mail message could be discarded for many reasons: duplicate, spam, or required no response.	23
`CallState_Disconnected` A voice call between the user and customer is disconnected.	30
`CallState_Expired` A callback call was not completed successfully and the existing callback schedules have expired.	27
`CallState_ExternallyConsulted` An e-mail message is 'deferred' pending an external consultation.	31
`CallState_ExternallyTransferred` An incoming e-mail message that is transferred to an external resource.	32
`CallState_Failed` A call has failed for various reasons.	36
`CallState_Finished` A call is Finished. This occurs after all parties that are involved in a call have disconnected.	22
`CallState_Held` A call is held. This occurs when a contact center user who is on a call presses hold.	17
`CallState_HeldInQueue` A call is held in queue. This can occur when a call is in an IVR, on IVR hold. The customer may be in the middle of an action, and the IVR has requested that the system not route the call immediately, but wait until the IVR tells the system that the caller is ready once more for routing to a user. If the caller is near the front of the queue, the next call can be assigned first, without displacing the first caller from their place in queue.	6
`CallState_Invalid` Missing information for CallState_Invalid	39
`CallState_IVRSuspended` The contact is waiting in the queue but the routing process (user assignment) is suspended by the system routing engine because the IVR application requested the suspended using the existing OpenScape Contact Center Immediate Connect to user (also called ICA hold) feature.	41
`CallState_LoadBalanced` A call has been Load Balanced. This occurs when a call is transferred to another site in a multisite environment.	3
`CallState_MaxRetry` A callback call was not completed successfully during the maximum retry number assigned to the callback.	28

Table 8 enCallStates enumeration (Seite 2 von 4)

Enumeration	Value
`CallState_MediaSuspended` The contact is waiting in the queue but routing process (user assignment) is suspended by the routing engine because of either a system-related or a callback-schedule suspend reason. Media-suspended contacts are the ones that are suspended because the Contact Center is closed (off hours), the associated media server (E-mail and Callback Servers only) is down, or the callback schedule has expired and it is applicable to callbacks and e-mail messages only.	38
`CallState_New` A callback is created and will be tried in the future.	33
`CallState_Offered` The call is assigned to a user, and is routed to the user.	10
`CallState_Pending` A call is pending assignment to a user. This occurs when a call is matched to a user, and the Routing Server has requested that the call be transferred to that user, but the transfer has not yet taken place.	25
`CallState_PostProcessing` A contact has been disconnected, but the user is still performing tasks related to the contact, such as making notes or sending follow-up information.	42
`CallState_Processing`	37
`CallState_Queued` A call is queued. This occurs when a call is queued on an ACD group, UCD group, or hunt group.	5
`CallState_Replied` An e-mail call has generated a reply. An e-mail reply can be either a response from a contact center user, or an automatic reply, generated by the E-mail Server (for example, for product information).	21
`CallState_Reserved` The call is reserved for a user.	9
`CallState_Retry` A callback is waiting for processing.	34
`CallState_Routing` A call is enqueued to the Routing Server. This occurs after a call arrives on a queue, and the T-Server informs the Routing Server about the call.	4
`CallState_Scheduled` A callback call is created in the Callback Server and is scheduled to be queued with the Routing Server at the time specified in the callback schedule.	26
`CallState_Start` A new call has been started. This occurs when a telephone is taken off-hook.	1
`CallState_TimedOut` The call in the system has timed out while waiting to be assigned to a user.	14

Table 8 enCallStates enumeration (Seite 3 von 4)

Enumeration	Value
`CallState_Unanswered` The call that is assigned to a user has not been answered.	13
`CallState_Unknown` An unknown state for a call. This occurs, when the T-Server cannot determine the state that a call is in.	0
`CallState_UserInteraction` For calls, this state indicates that the caller was connected with a user and the caller concerned is being handled. For callbacks, this indicates that the offered user has accepted the callback offer and a callback attempt is in process. For e-mail contacts, this state indicates that user is processing this e-mail.	40

Table 8 enCallStates enumeration (Seite 4 von 4)

3.8 enCapabilityType

The enCapabilityType enumeration represents the capabilities that can be assigned to a user.

NOTE: The only capability that is currently supported is "Campaign Director".

Enumeration	Value
`CapType_CampaignDirector`	2

Table 9 enCapability

3.9 enContactTypes

Enumeration	Value
`ContactType_DirectIncomingVoice`	2
`ContactType_DirectInternalVoice`	4
`ContactType_DirectOutgoingEmail`	7
`ContactType_DirectOutgoingVoice`	3
`ContactType_None`	0
`ContactType_RoutedCallback`	5
`ContactType_RoutedEmail`	6
`ContactType_RoutedVoice`	1

Table 10 enContactTypes (Seite 1 von 2)

Enumeration	Value
ContactType_RoutedWebCollaboration	8
ContactType_UnknownVoice	9

Table 10 enContactTypes (Seite 2 von 2)

3.10 enDeliveredReasons

Enumeration	Value
DeliveredReason_Conferenced A user is being requested to join a Web collaboration session.	4
DeliveredReason_InitialArrival A contact arrived at the Web server.	1
DeliveredReason_Normal A contact is being assigned to a user (no conference).	2
DeliveredReason_Requeued A requeued contact is being assigned to a user.	3
DeliveredReason_Undefined	0

Table 11 enDeliveredReasons enumeration

3.11 enDequeueReasons

The enDequeueReasons enumeration represents the various reasons that a call could be dequeued from the Routing Server.

Enumeration	Value
DequeueReason_WebCollaborationConnectFailed A call is dequeued because the Web Interaction Server could not connect the contact center user and that customer. This can occur for many reasons. For example, the customer may have lost their connection to the internet.	3
DequeueReason_CallAbandoned A call is dequeued because the call is abandoned. After a call is abandoned, there is no reason to keep it in the Routing Server queue.	6
DequeueReason_CallAnswered A call is dequeued because the call is answered. After a call is assigned to a user, and the user has answered the call, there is no reason to keep it in the Routing Server queue.	5
DequeueReason_CallTimedOut A call is dequeued because the call has timed out. After a call has timed out, there is no reason to keep it in the Routing Server queue, as it is transferred to the time out target.	7

Table 12 enDequeueReasons enumeration (Seite 1 von 2)

Enumeration	Value
`DequeueReason_None` This occurs when the Routing Server is unable to determine why the call is dequeued.	0
`DequeueReason_RequestFromEmailServer` A call is dequeued through a request from the E-mail Server. This can occur when a contact center user determines that an e-mail that is queued no longer needs to be handled and chooses to dequeue.	2
`DequeueReason_RequestFromWebServer` A call is dequeued through a request from the Web Interaction Server. This can occur when a customer that is queued for a Web collaboration session decides that they do not want to wait any longer for an answer, and requests the Web collaboration request to be dequeued.	1
`DequeueReason_SystemError` A call is dequeued because of a system error. This can occur if there is a problem with a server in the system.	4
`DequeueReason_WorkflowTransfer` A call is dequeued because a contact has been transferred from the queue through a Workflow Transfer component.	8

Table 12 enDequeueReasons enumeration (Seite 2 von 2)

3.12 enDiagnosticFilters

The enDiagnosticFilters enumeration represents the various diagnostic filters that can be turned on or off in the system. Turning on diagnostics can adversely affect the performance of your application.

Enumeration	Value
`DiagnosticFilter_Application` Enables diagnostics. The system exposes its diagnostic writing functions to applications that are using the OpenScape Manager. To ensure that these messages will be logged properly, you must enable the DiagnosticFilter_Application filter.	2
`DiagnosticFilter_HiPathProCenterFunctionEntryExit` Enable diagnostics for function entry exit messages from the system. These can greatly affect performance, as a message will be logged every time a function is entered and exited, and should only be enabled at the request of a software developer. These diagnostics are strictly designed for software developers, and in some instances, software developers may request that scenarios be reproduced with one or more filters enabled to aid in determining issues.	6

Table 13 enDiagnosticFilters enumeration (Seite 1 von 2)

Enumeration	Value
DiagnosticFilter_HiPathProCenterInformation Enable diagnostics for informative messages from the system. This includes such things as documenting return values from procedures. These diagnostics are strictly designed for software developers, and in some instances, software developers may request that scenarios be reproduced with one or more filters enabled to aid in determining issues.	4
DiagnosticFilter_HiPathProCenterSevere Enable diagnostics for when something severe happens in the system. This includes such things as memory exceptions. These diagnostics are strictly designed for software developers, and in some instances, software developers may request that scenarios be reproduced with one or more filters enabled to aid in determining issues.	3
DiagnosticFilter_HiPathProCenterWarning Enable diagnostics for warning messages from the system. This includes such things as the passing of invalid parameters. These diagnostics are strictly designed for software developers, and in some instances, software developers may request that scenarios be reproduced with one or more filters enabled to aid in determining issues.	5
DiagnosticFilter_Toolkit Affects diagnostics for the SDK. This usually aids in debugging issues in the SDK code, or by developers using the SDK to determine what is happening with their code. These diagnostics are designed to be read by software developers, but may aid in your debugging efforts as well.	1

Table 13 enDiagnosticFilters enumeration (Seite 2 von 2)

3.13 enDisconnectReasons

The enDisconnectReasons enumeration represents the reasons that a disconnect has taken place.

Enumeration	Value
DisconnectReason_AbandonedInQueue	104
DisconnectReason_AbandonedInWorkflow	103
DisconnectReason_AbandonedWhilePending	106
DisconnectReason_AgentDeleted	110
DisconnectReason_AgentReplied	1
DisconnectReason_AutoResponded	2
DisconnectReason_BrowserTimedOut	108
DisconnectReason_CriticalServerDown	5
DisconnectReason_Discarded	3

Table 14 enDisconnectReasons enumeration (Seite 1 von 2)

Enumeration	Value
DisconnectReason_FailedToStartWorkflow	102
DisconnectReason_Normal	101
DisconnectReason_SessionExpired	114
DisconnectReason_ShuttingDown	112
DisconnectReason_TimedOut	4
DisconnectReason_TimedOutInQueue	105
DisconnectReason_TransferredOut	6
DisconnectReason_Undefined	0
DisconnectReason_WorkflowDisconnect	111

Table 14 enDisconnectReasons enumeration (Seite 2 von 2)

3.14 enDivertReasons

The enDivertReasons enumeration represents the reason for a diversion.

Enumeration	Value
DivertReason_AbandonedWhilePending	105
DivertReason_DeferTakeBack	1
DivertReason_DisconnectedWhilePending	106
DivertReason_ExternalConsultReply	4
DivertReason_ExternalConsultTakeBack	5
DivertReason_InvitationRejected	102
DivertReason_InvitationTimeOut	103
DivertReason_NoAnswer	2
DivertReason_RoutingTimedOut	3
DivertReason_SystemRouted	101
DivertReason_Undefined	0

Table 15 enDivertReasons enumeration

3.15 enEmailsHistoryQueryCriteria

The enEmailsHistoryQueryCriteria

Enumeration	Value
EmailsHistoryQueryCriterion_CallCenterAddress	4
EmailsHistoryQueryCriterion_ConversationId	0
EmailsHistoryQueryCriterion_CustomAddress	5
EmailsHistoryQueryCriterion_LastAgentKey	2
EmailsHistoryQueryCriterion_OriginalAgentKey	1
EmailsHistoryQueryCriterion_QueueKey	3
EmailsHistoryQueryCriterion_Subject	6

Table 16 enEmailsHistoryQueryCriteria enumeration

3.16 enEmailTemplateTypes

The enEmailTemplateTypes represents the e-mail template type.

Enumeration	Value
EmailTemplateType_AutoAcknowledgement	2
EmailTemplateType_NewMessage	4
EmailTemplateType_Prolog	0
EmailTemplateType_Response	3
EmailTemplateType_Signature	1
EmailTemplateType_Unknown	99

Table 17 enEmailTemplateTypes enumeration

3.17 enEmailTypes

The enEmailTypes represents the e-mail types.

Enumeration	Value
EmailType_AgentOutbound	6
EmailType_CustomerReply	3
EmailType_ExternalIncoming	5
EmailType_ExternalOutgoing	4
EmailType_Internal	7
EmailType_None	0

Table 18 enEmailTypes enumeration

Enumeration	Value
`EmailType_Original`	1
`EmailType_Reply`	2
`EmailType_Resend`	8

Table 18 enEmailTypes enumeration

3.18 enErrors

The enErrors enumeration represents all the various errors that are returned from the objects in the SDK.

Enumeration	Value
`Error_Generic_ActionFailed` The action you have requested to perform on an object, such as a Transfer on a VoiceCall or an Enqueue on a RoutingCall, has failed. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	-2147217373
`Error_Generic_BaseObjectDoesNotExist` The underlying object in the SDK object does not exist. Some objects in the SDK create a 'wrapper' around other classes. When this error occurs, it means that the underlying 'base object' has not been created. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	-2147217405
`Error_Generic_CallDataItemFailed`	-2147217151
`Error_Generic_CallDataKeyValuePairReadOnly`	-2147217149
`Error_Generic_CollectionIsEmpty`	-2147217358
`Error_Generic_ContactDataAddFailed`	-2147217152
`Error_Generic_ContactDataAddFailedDuplicateKey` You have attempted to add a KeyValuePair object to the ContactData collection and the add attempt failed. This is because a KeyValuePair already exists in the ContactData collection with the same 'Key' string. KeyValuePair with duplicate 'Key' are not allowed in the ContactData collection.	-2147217148
`Error_Generic_ContactDataAddFailed` You have attempted to add a KeyValuePair object to the ContactData collection and the add attempt failed. This failure message occurs if there is an issue creating the information that is stored in the ContactData collection. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	-2147217152

Table 19 enErrors enumeration (Seite 1 von 23)

Enumeration	Value
`Error_Generic_ContactDataItemFailed` You have attempted to query a KeyValuePair object from the ContactData collection and the update attempt failed. This failure message occurs if there is an issue creating a copy of the information that is stored in the ContactData collection. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217151
`Error_Generic_ContactDataKeyValuePairReadOnly` The KeyValuePAir object is read-only. If you want to store a value in the ContactData collection, you cannot use 'DDE' or 'UUI' for Key in KeyValuePair or add Binary KeyValuePair to the ContactData collection.	–2147217149
`Error_Generic_CreatingObject` Your attempt to create an object has failed. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217375
`Error_Generic_InitializationTimedOut` A Manager has timed-out while attempting to create the event threads for the servers that it uses. This typically indicates that one or more of the event threads are not fully connected to the servers in question. This could be because the server is down, or the machine attempting to connect with the servers is not connected to the network.	–2147217388
`Error_Generic_ItemAlreadyExists` You tried to add an item that already exists to the collection. For more information, consult your documentation or contact your system administrator.	–2147217360
`Error_Generic_ItemNotFound` You have attempted to query a KeyValuePair object from the ContactData collection and the item does not exist in the collection. If this occurs, there was most likely an error in the user input.	–2147217404
`Error_Generic_IterationGetNewEnum` The _NewEnum property is called for iterating through a collection (this is called by Visual Basic when For Each is used). This indicates that an error occurred while building the enumeration of all the items in the collection. If this occurs, there may be an issue with the items in the collection. There also may be an issue with the SDK, and you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217403
`Error_Generic_ListeningForEventsWhenIgnoringEvents` You have requested to listen for an event type after initializing a Manager without events. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147217390

Table 19 enErrors enumeration (Seite 2 von 23)

Enumeration	Value
`Error_Generic_NetworkBusy` The action you have requested to perform on an object, such as a Transfer on a VoiceCall or an Enqueue on a RoutingCall, has failed because the communication platform is either very busy, or there is something wrong with the communication platform. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217374
`Error_Generic_NotSupportedObject` You tried to access an object which is not supported. For more information, consult your documentation or contact your system administrator.	–2147217357
`Error_Generic_ObjectReadOnly`	–2147217367
`Error_Generic_QueryFailed` Your query request has failed. This can occur for several reasons, including if the CallID does not match a call in the system or the server is down. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217391
`Error_Generic_QueryNotSupported` An object you have passed into the Query method of the MediaManager or RoutingManager is not supported. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217384
`Error_Generic_RequestTimedOut` The action you have attempted to perform on an object, such as a Transfer on a VoiceCall or an Enqueue on a RoutingCall, has failed because the request has timed out. This typically indicates an issue with the T-Server, or that the communication platform is very busy. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217371
`Error_Generic_StopListeningForEventsWhenIgnoringEvents` You have requested to stop listening for an event type after initializing the Manager without events. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147217389

Table 19 enErrors enumeration (Seite 3 von 23)

Enumeration	Value
`Error_Generic_UnableToAllocateMemory` The SDK was unable to perform your request because of a memory allocation issue. If this occurs, try closing any open application windows to free up more memory, or investigate whether the machine requires more memory to run the application properly. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217406
`Error_Generic_UnableToCreateXMLParser` Unable to create XML Parser Object. For more information, consult your documentation or contact your system administrator.	–2147217359
`Error_Generic_UnableToDetermineState` Your attempt to determine the state of a Manager has failed. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217370
`Error_Generic_UnableToReadObjectInformation` An error occurred while reading the information from the object that was passed into the Query or Update methods of the MediaManager, RoutingManager, or StatisticsManager. If this occurs, the object you passed in may not be supported or there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217386
`Error_Generic_UnableToWriteObjectInformation` An error occurred while writing the information to the object that was passed into the Query or Update methods of the MediaManager, RoutingManager, or StatisticsManager. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217385
`Error_Generic_UnknownError` An unknown error has been returned from the SDK in one of the Managers. The likelihood of this error is very small, but if it is returned, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217407
`Error_Generic_UnknownEventType` You have requested to listen for or stop listening for an event type that does not exist. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147217399

Table 19 enErrors enumeration (Seite 4 von 23)

Enumeration	Value
`Error_Generic_UnknownObjectType` The Media Manager or RoutingManager could not determine which object you wanted to query or update. This occurs when it attempts to check the MediaBase or RoutingBase interface for the object type that we want to query or update. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217387
`Error_Generic_UnknownState` The SDK was unable to determine the state of a voice or Web collaboration contact. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217392
`Error_Generic_UpdateFailed` The Manager was unable to update an object on the appropriate server in the system. This can occur for several reasons, including if the CallID does not match a call in the system, or the server is down. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217376
`Error_Generic_UpdateNotSupported` An object you have passed into the Update method of the MediaManager or RoutingManager is not supported. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217383
`Error_Generic_ValidationFailed` The action you have attempted to perform on an object, such as a Transfer on a VoiceCall or an Enqueue on a RoutingCall, could not be completed at this time and has failed. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217372
`Error_Generic_VariantTypeNotSupported` You have attempted to pass in an invalid variant object type for the Value of a KeyValuePair object. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147217401
`Error_Generic_WrongPatch`	–2147217368

Table 19 enErrors enumeration (Seite 5 von 23)

Enumeration	Value
`Error_Generic_WrongVersion` The version of the SDK you are running on the client machine is not up-to-date with the version that is running on the server. If this occurs, there is an issue with the auto-update portion of the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147217369
`Error_Generic_XMLEventReadOnly` You have attempted to build a read-only event from an XML string. The operation you are attempting is not supported, because the read-only events are not designed to allow their properties to be updated by any means. If this occurs, there is an issue in the code that is attempting to populate the read-only event with the FromXML string.	–2147217400
`Error_Generic_XMLObjectReadOnly` You have attempted to build a read-only object from an XML string. The operation you are attempting is not supported, because the read-only objects are not designed to allow their properties to be updated by any means. If this occurs, there is an issue in the code that is attempting to populate the read-only object with the FromXML string.	–2147217402
`Error_HiPathProCenter_Auto_Authentication_Failed`	–2147213008
`Error_HiPathProCenter_ConnectingToAdministrationServer` An error has occurred while attempting to connect to the Administration Server. If this occurs, this could be an issue with the Administration Server (perhaps it is not running), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213056
`Error_HiPathProCenter_ConnectingToCallbackServer` An error has occurred while attempting to connect to the Callback Server. If this occurs, this could be an issue with the Callback Server (perhaps it is not running, or the server is not defined in the database), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213052
`Error_HiPathProCenter_ConnectingToConfigurationSynchronizationServer` An error has occurred while attempting to connect to the Configuration Server. If this occurs, this could be an issue with the Configuration Server (perhaps it is not running, or the server is not defined in the database), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213054

Table 19 enErrors enumeration (Seite 6 von 23)

Enumeration	Value
`Error_HiPathProCenter_ConnectingToEmailServer` An error has occurred while attempting to connect to the E-mail Server. If this occurs, this could be an issue with the E-mail Server (perhaps it is not running, or the server is not defined in the database), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213051
`Error_HiPathProCenter_ConnectingToNetworkRoutingServer` An error has occurred while attempting to connect to the Network Routing Server. If this occurs, this could be an issue with the Network Routing Server (perhaps it is not running, or the server is not defined in the database), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213048
`Error_HiPathProCenter_ConnectingToPresenceServer` An error has occurred while attempting to connect to the Presence Server. If this occurs, this could be an issue with the Presence Server (perhaps it is not running, or the server is not defined in the database), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213004
`Error_HiPathProCenter_ConnectingToRealTimeServer` An error has occurred while attempting to connect to the Real-Time Server. If this occurs, this could be an issue with the Real-Time Server (perhaps it is not running, or the server is not defined in the database), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213047
`Error_HiPathProCenter_ConnectingToRoutingServer` An error has occurred while attempting to connect to the Routing Server. If this occurs, this could be an issue with the Routing Server (perhaps it is not running, or the server is not defined in the database), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213049
`Error_HiPathProCenter_ConnectingToStatisticsServer` An error has occurred while attempting to connect to the Statistics Server. If this occurs, this could be an issue with the Statistics Server (perhaps it is not running, or the server is not defined in the database), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213040

Table 19 enErrors enumeration (Seite 7 von 23)

Enumeration	Value
`Error_HiPathProCenter_ConnectingToTelephonyServer` An error has occurred while attempting to connect to the T-Server. If this occurs, this could be an issue with the T-Server (perhaps it is not running, or the server is not defined in the database), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213053
`Error_HiPathProCenter_ConnectingToWatchdogServer` An error has occurred while attempting to connect to the Watchdog Server. If this occurs, this could be an issue with the Watchdog Server (perhaps it is not running, or the server is not defined in the database), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213055
`Error_HiPathProCenter_ConnectingToWebCollaborationServer` An error has occurred while attempting to connect to the Web Interaction Server. If this occurs, this could be an issue with the Web Interaction Server (perhaps it is not running, or the server is not defined in the database), an issue in the code, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213050
`Error_HiPathProCenter_CouldNotFindCallbackServerAddress` The HiPathProCenterManager cannot determine the address of the Callback Server. If this occurs, it typically indicates that the server is not defined or configured in the database, or the HiPathProCenterManager encountered an error while attempting to query the Server summary collection from the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213031
`Error_HiPathProCenter_CouldNotFindConfigurationSynchronizationServerAddress` The HiPathProCenterManager could not determine the address of the Configuration Synchronization Server. If this occurs, it typically indicates that the server is not defined or configured in the database, or the HiPathProCenterManager encountered an error while attempting to query the Server summary collection from the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213033
`Error_HiPathProCenter_CouldNotFindEmailServerAddress` The HiPathProCenterManager cannot determine the address of the E-mail Server. If this occurs, it typically indicates that the server is not defined or configured in the database, or the HiPathProCenterManager encountered an error while attempting to query the Server summary collection from the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213024

Table 19 enErrors enumeration (Seite 8 von 23)

Enumeration	Value
`Error_HiPathProCenter_CouldNotFindNetworkRoutingServerAddress` The HiPathProCenterManager cannot determine the address of the Network Routing Server. If this occurs, it typically indicates that the server is not defined or configured in the database, or the HiPathProCenterManager encountered an error while attempting to query the Server summary collection from the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	-2147213021
`Error_HiPathProCenter_CouldNotFindPresenceServerAddress` The HiPathProCenterManager cannot determine the address of the Presence Server. If this occurs, it typically indicates that the server is not defined or configured in the database, or the HiPathProCenterManager encountered an error while attempting to query the Server summary collection from the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	-2147213003
`Error_HiPathProCenter_CouldNotFindRealTimeServerAddress` The HiPathProCenterManager cannot determine the address of the Real-Time Server. If this occurs, it typically indicates that the server is not defined or configured in the database, or the HiPathProCenterManager encountered an error while attempting to query the Server summary collection from the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	-2147213020
`Error_HiPathProCenter_CouldNotFindRoutingServerAddress` The HiPathProCenterManager cannot determine the address of the Routing Server. If this occurs, it typically indicates that the server is not defined or configured in the database, or the HiPathProCenterManager encountered an error while attempting to query the Server summary collection from the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	-2147213022
`Error_HiPathProCenter_CouldNotFindStatisticsServerAddress` The HiPathProCenterManager cannot determine the address of the Statistics Server. If this occurs, it typically indicates that the server is not defined or configured in the database, or the HiPathProCenterManager encountered an error while attempting to query the Server summary collection from the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	-2147213019

Table 19 enErrors enumeration (Seite 9 von 23)

Enumeration	Value
`Error_HiPathProCenter_CouldNotFindTelephonyServerAddress` The HiPathProCenterManager cannot determine the address of the T-Server. If this occurs, it typically indicates that the server is not defined or configured in the database, or the HiPathProCenterManager encountered an error while attempting to query the Server summary collection from the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213032
`Error_HiPathProCenter_CouldNotFindWatchdogServerAddress` The HiPathProCenterManager could not determine the address of the Watchdog Server. If this occurs, it typically indicates that the server is not defined or configured in the database, or the HiPathProCenterManager encountered an error while attempting to query the Server summary collection from the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213034
`Error_HiPathProCenter_CouldNotFindWebCollaborationServerAddress` The HiPathProCenterManager cannot determine the address of the Web Interaction Server. If this occurs, it typically indicates that the server is not defined or configured in the database, or the HiPathProCenterManager encountered an error while attempting to query the Server summary collection from the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213023
`Error_HiPathProCenter_CouldNotQueryUser` The HiPathProCenterManager encountered an error attempting to query the user from the database. This can occur for a variety of reasons (underlying support objects are not created, the system was unable to allocate memory for the User object, the database was locked, and so on). If this occurs, there may be an error with the code, with the machine that the application is running on, with the database, or an issue with the SDK. If the latter is the case, you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213037
`Error_HiPathProCenter_IncorrectPassword` A user has attempted to log on with an incorrect password. Ensure that the password has been entered correctly and try again. If this error enumeration occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147213036
`Error_HiPathProCenter_Invalid_Admin_Address`	–2147213016

Table 19 enErrors enumeration (Seite 10 von 23)

Enumeration	Value
`Error_HiPathProCenter_Invalid_BusinessUnitName` The HiPathProCenterManager could not find the specified business unit name in the database. This error occurs only in a multitenant environment.	–2147213006
`Error_HiPathProCenter_MismatchBUKeyLogonInfoReq` The business unit name from the Client Desktop application does not match the business unit name that was Initialized by the SDK application. The SDK application must log on using a valid user name and password. This error occurs only in a multitenant environment.	–2147213005
`Error_HiPathProCenter_NoLicenseAvailable` HiPathProCenterManager could not initialize because it could not allocate an SDK license in the Administration Server. If this occurs, it means that the SDK application had consumed all available SDK licenses.	–2147213007
`Error_HiPathProCenter_QueryingServerSummariesTimedOut` The HiPathProCenterManager could not query the list of servers from the Administration Server. The HiPathProCenterManager requires the list of servers, so it knows the addresses to use when connecting to the various servers. This error can occur for a variety of reasons, including that the address to the Administration Server is incorrect, or the machine the application is running on is not connected to the Administration Server. There may also be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147213017
`Error_HiPathProCenter_ReaddingToEventQueue` The HiPathProCenterManager encountered an error attempting to move an event to the event queue. This occurs when the HiPathProCenterManager has popped an event from the head of the event queue and attempted to send that event to the application waiting for events. If for some reason, an error occurs trying to send the event to the application, the event is then moved to the head of the event queue. This error is returned when an error is encountered attempting to move the event to the head of the queue. If this occurs, in all likelihood there is an issue with the SDK (or memory issue), and you should notify Support to determine if this is a known issue, and request a fix.	–2147213038
`Error_HiPathProCenter_UserKey_Password_Empty`	–2147213015
`Error_HiPathProCenter_UserNotLoggedOn` You have attempted to create either the Media, Routing or Statistics Manager without first logging a user on. Before attempting to create any Manager other then the Administration Manager, the Logon method must be called. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147213035

Table 19 enErrors enumeration (Seite 11 von 23)

Enumeration	Value
`Error_Media_ActionResultCSTAONSNotEnabled` The T-Server could not perform the requested action because the One Number Service feature is not enabled.	–214720484 3
`Error_Media_ActionResultInvalidCSTADeviceID` The T-Server could not perform the requested action because the CSTA Device ID is invalid.	–214720484 5
`Error_Media_ActionResultInvalidExtension` The T-Server could not perform the requested action, such as a Transfer on a VoiceCall or an Enqueue on a RoutingCall, because the extension is invalid. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–214720511 8
`Error_Media_ActionResultInvalidForwardingDestination` The T-Server could not perform the requested action because the forwarding destination is invalid or restricted.	–214720484 4
`Error_Media_ActionResultLimitExceeded` The T-Server could not perform the requested action, such as a Transfer on a VoiceCall or an Enqueue on a RoutingCall, because the limit is exceeded. This can occur if you already have the maximum number of users logged onto the system. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–214720511 7
`Error_Media_ActionResultLineBusy` The code has attempted to perform an action on an object, such as a Transfer on a VoiceCall or an Enqueue on a RoutingCall, because the telephone device that you are calling is busy. This is normally returned after a Dial request. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–214720511 1
`Error_Media_ActionResultOutOfService` The T-Server could not perform the requested action, such as a Transfer of a VoiceCall or an Enqueue of a RoutingCall, because it is out of service. If this occurs, you should check to ensure that the T-Server on the server machine is running, and correct any issues that it might be having. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–214720511 2

Table 19 enErrors enumeration (Seite 12 von 23)

Enumeration	Value
`Error_Media_ActionResultResourceBusy` The T-Server could not perform the requested action, such as a Transfer on a VoiceCall or an Enqueue on a RoutingCall, because a resource, such as an extension, is in use. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205115
`Error_Media_ActionResultResourceError` The T-Server could not perform the requested action, such as a Transfer on a VoiceCall or an Enqueue on a RoutingCall, because of a resource error. If this occurs, it could be because the resource specified does not exist, or is unknown in the system. Ensure that the extension is defined in the database. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205114
`Error_Media_ActionResultStateError` The T-Server could not perform the requested action, such as a Transfer on a VoiceCall or an Enqueue on a RoutingCall, because of a state error. This can occur if you attempt to log a user off when they are not already in the logged on state. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205113
`Error_Media_ActionResultTelephonyServerNotReady` The T-Server could not perform the requested action, such as a Transfer on a VoiceCall or an Enqueue on a RoutingCall, because it is not ready to process the request. This can occur if the T-Server is just starting. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205116
`Error_Media_AgentInvalidMediaTypeForAction` That the SDK could not perform the requested operation because it requires a specific media type. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205119

Table 19 enErrors enumeration (Seite 13 von 23)

Enumeration	Value
`Error_Media_AgentUnknownMediaType` This indicates that the SDK could not recognize the media type. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205120
`Error_Media_Callback_AddScheduleFailed` This indicates that the Callback Server could not add a Callback Schedule to Callback Schedules collection. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205016
`Error_Media_Callback_CallbackScheduleExpired` This indicates that the Callback Server could not add a Callback Schedule to Callback Schedules collection because the Callback Schedules has an EndDate property in the past. The EndDate has expired. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205017
`Error_Media_Callback_RemoveScheduleFailed` This indicates that the Callback Server could not delete the Callback Schedule from Callback Schedules collection. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205015
`Error_Media_Callback_UpdateScheduleFailed` This indicates that the Callback Server could not update the Callback Schedule in Callback Schedules collection. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205008
`Error_Media_CallbackActionResultCallbackAlreadyActive` This indicates that the Callback Server failed to update callback because the callback is already active. Please try again. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	Unknown
`Error_Media_CallbackActionResultCallbackCallNotFound` The Callback Server could not perform the requested action, such as a Create or Delete request, because the CallbackCall specified by the CallID is not found in the system. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205070
`Error_Media_CallbackActionResultCallbackServerNotReady` The Callback Server could not perform the requested action, such as a Create or Delete request, because it is not ready to process the request. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205102
`Error_Media_CallbackActionResultCallDataTooLong`	–2147205084

Table 19 enErrors enumeration (Seite 14 von 23)

Enumeration	Value
`Error_Media_CallbackActionResultCallIDToolong` The Callback Server could not perform the requested action, such as a Create or Delete request, because the CallID you have entered is too long. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205100
`Error_Media_CallbackActionResultCannotDeleteOngoingCall` The Callback Server could not perform the requested action, such as a Create or Delete request, because the application could not delete a callback call that is being handled by a user. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205055
`Error_Media_CallbackActionResultCannotUpdateOngoingCall` The Callback Server could not perform the requested action, such as a Create or Delete request, because it could not update an ongoing call. If this occurs, there is an issue in the code that is using the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205101
`Error_Media_CallbackActionResultCaptionTooLong`	–2147205085
`Error_Media_CallbackActionResultContactDataTooLong` The Callback Server could not perform the requested action, such as a Create or Delete request, because the Contact Data you have entered is longer than the length defined in the Callback Server. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205084
`Error_Media_CallbackActionResultCountryCodeTooLong` The Callback Server could not perform the requested action, such as a Create or Delete request, because the country code you have entered is longer than the length defined in the Callback Server. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205087
`Error_Media_CallbackActionResultCustomerNameTooLong` The Callback Server could not perform the requested action, such as a Create or Delete request, because the customer name you have entered is longer than the length defined in the Callback Server. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205088
`Error_Media_CallbackActionResultDatabaseOperationError` The Callback Server could not perform the requested action, such as a Create or Delete request, because an error was encountered in the database. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205071

Table 19 enErrors enumeration (Seite 15 von 23)

Enumeration	Value
`Error_Media_CallbackActionResultDDETooLong` The Callback Server could not perform the requested action, such as a Create or Delete request, because the DDE you have entered is longer than the length defined in the Callback Server. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205083
`Error_Media_CallbackActionResultDescriptionTooLong` The Callback Server could not perform the requested action, such as a Create or Delete request, because the description you have entered is longer than the length defined in the Callback Server. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205085
`Error_Media_CallbackActionResultDuplicateCallback` This indicates that the Callback Server could not perform the requested action because it has detected duplicate callback: callback with the same queue and destination number. Please try again. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205003
`Error_Media_CallbackActionResultEmptyCallID` The Callback Server could not perform the requested action, such as a Create or Delete request, because there is no CallID to specify the CallbackCall. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205069
`Error_Media_CallbackActionResultEndTimeEarlyThanStartTime` The Callback Server could not perform the requested action, such as a Create or Delete request, because you have entered an end time that is earlier than the start time associated with the callback schedule. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205096
`Error_Media_CallbackActionResultFoundExcludedNumber` The Callback Server could not perform the requested action, such as a Create or Delete request, because one of the telephone numbers entered is an excluded number for the contact center. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205086
`Error_Media_CallbackActionResultInvalidCallbackState` The Callback Server could not perform the requested action, such as a Create or Delete request, because the CallbackCall is not in the proper state. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205079
`Error_Media_CallbackActionResultInvalidCallOrigin` This indicates that the Callback Server could not perform the requested action because the call origin is invalid. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205007

Table 19 enErrors enumeration (Seite 16 von 23)

Enumeration	Value
`Error_Media_CallbackActionResultInvalidPriority` The Callback Server could not perform the requested action, such as a Create or Delete request, because the priority for the CallbackCall is invalid. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205072
`Error_Media_CallbackActionResultInvalidQueueKey` The Callback Server could not perform the requested action, such as a Create or Delete request, because you have entered an invalid queue key. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205098
`Error_Media_CallbackActionResultInvalidQueueName` This indicates that the Callback Server could not perform the requested action because the queue name is invalid. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205004
`Error_Media_CallbackActionResultInvalidRPCType` This indicates that the Callback Server could not perform the requested action because of an internal error. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205065
`Error_Media_CallbackActionResultInvalidScheduleTime` The Callback Server could not perform the requested action, such as a Create or Delete request, because you have entered an invalid schedule time. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205097
`Error_Media_CallbackActionResultInvalidSiteKey` The Callback Server could not perform the requested action, such as a Create or Delete request, because you have entered an invalid site key. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205099
`Error_Media_CallbackActionResultInvalidSiteName` This indicates that the Callback Server could not perform the requested action because the site name is invalid. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205063
`Error_Media_CallbackActionResultInvalidTelphoneNumber` The Callback Server could not perform the requested action, such as a Create or Delete request, because the telephone number for the callback schedule is invalid. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205066

Table 19 enErrors enumeration (Seite 17 von 23)

Enumeration	Value
`Error_Media_CallbackActionResultReceivedCorruptedData` The Callback Server could not perform the requested action, such as a Create or Delete request, because the data that was sent to the Callback Server was corrupted. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205064
`Error_Media_CallbackActionResultResourceError` The Callback Server could not perform the requested action, such as a Create or Delete request, because of a resource error. If this occurs, it could be because the resource specified does not exist, or is unknown in the system. Ensure that all of the resources specified exist in the database. If the problem persists, there may be an error in the code that is using the SDK or in the input that you are passing in. It is also possible that there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205103
`Error_Media_CallbackActionResultScheduleExceedMaxDaysAhead` The Callback Server could not perform the requested action, such as a Create or Delete request, because a call is scheduled too many days ahead of the current date. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205054
`Error_Media_CallbackActionResultSchedulesExpired` The Callback Server could not perform the requested action, such as a Create or Delete request, because the times used in the schedule have expired. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205068
`Error_Media_CallbackActionResultSchedulesHaveNoMutualTime` The Callback Server could not perform the requested action, such as a Create or Delete request, because the schedules associated with the CallbackCall do not have a mutual time. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205095
`Error_Media_CallbackActionResultSchedulesOverlapped` The Callback Server could not perform the requested action, such as a Create or Delete request, because the times used in the schedule overlap. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205067
`Error_Media_CallbackScheduleInvalidKey` The Callback Server could not perform the requested action because the key for the callback schedule falls outside the valid range (1-3). If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205053

Table 19 enErrors enumeration (Seite 18 von 23)

Enumeration	Value
`Error_Media_EmailAddressNotComplete` The SDK could not perform the requested action because the EmailAddress property is missing from one of the e-mail addresses that you are trying to assign to the ReplyToCCEmailAddresses collection or the ReplyTo BCCEmailAddresses collection. Ensure that this property is populated with the correct data before assigning an EmailAddresses collection to the ReplyTOCCEmailAddresses property or the ReplyToBCCEmailAddresses property.	–2147205018
`Error_Media_EmailAdministrationServerError` An error has occurred in the communication between the Administration Server and the E-mail Server. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205052
`Error_Media_EmailAttachmentNotComplete` The SDK could not perform the requested action because either the Name or Path property is missing from one of the e-mail attachments that you are trying to assign to the ReplyAttachments collection. Ensure that both these properties are populated with the correct data before assigning an EmailAttachments collection to the ReplyAttachments property.	–2147205023
`Error_Media_EmailAttachmentNotLinked` The SDK could not perform the requested action because an isolated e-mail attachment cannot be retrieved. This e-mail attachment is not linked to an EmailCall object. Ensure that the e-mail attachment is associated with an EmailCall object.	–2147205024
`Error_Media_EmailAttachmentTooLargeError` The E-mail Server could not perform the requested action because the e-mail attachments are larger than the maximum size allowed. Ensure that the size is less than the maximum allowed size and try again.	–2147205022
`Error_Media_EmailConfigurationError` The E-mail Server could not perform the requested action because of an invalid configuration setting. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205051
`Error_Media_EmailCorporateEmailServerDown` The E-mail Server could not perform the requested action because the corporate E-mail Server is down. If this occurs, you should check to see if the E-mail Server is actually running. If it is, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205036

Table 19 enErrors enumeration (Seite 19 von 23)

Enumeration	Value
`Error_Media_EmailCorporateEmailServerError` The E-mail Server could not perform the requested action because an error was received from the corporate E-mail Server. If this occurs, you should ensure that the corporate E-mail Server is actually running. If it is, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205035
`Error_Media_EmailDatabaseConnectionFailed` The E-mail Server could not perform the requested action because it encountered an error while attempting to connect to the database. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205048
`Error_Media_EmailDatabaseError` The E-mail Server could not perform the requested action because it encountered an error while attempting to update the database. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205049
`Error_Media_EmailDataCorrupted` The E-mail Server could not perform the requested action because the e-mail data is corrupted. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205050
`Error_Media_EmailInvalidAgent` The E-mail Server could not perform the requested action because the user specified is not a valid user. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205047
`Error_Media_EmailInvalidCallID` The E-mail Server could not perform the requested action because the Call ID specified is not a valid Call ID. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147205040
`Error_Media_EmailMessageNotFound` The E-mail Server could not perform the requested action because the e-mail message could not be found on the corporate E-mail Server. If this occurs, there is an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205033
`Error_Media_EmailRoutingServerError` The E-mail Server could not perform the requested action because there was an issue with the E-mail Server communicating with the Routing Server. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205039

Table 19 enErrors enumeration (Seite 20 von 23)

Enumeration	Value
`Error_Media_EmailServerTerminating` The E-mail Server could not perform the requested action because the E-mail Server is shutting down. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147205038
`Error_Media_OperationNotSupportedOnCommunciationPlatform` The SDK could not perform the requested action because the operation is not supported on the communication platform.	–2147204846
`Error_Media_PresenceAddCapabilityException` The Presence Server could not perform the requested action because the AddCapability method call failed.	-2147204831
`Error_Media_PresenceCapabilityHandlingNailUpCall` The Presence Server could not perform the requested action because the specified number is already being used by Campaign Director to set up a connection for a blended agent for an outbound call.	-2147204842
`Error_Media_PresenceCapabilityNotSupported` The Presence Server could not perform the requested action because the specified capability is not supported.	-2147204841
`Error_Media_PresenceInvalidState` The Presence Server could not perform the requested action because the specified routing state is invalid.	-2147204839
`Error_Media_PresenceRemoveCapabilityException` The Presence Server could not perform the requested action because the RemoveCapability method call failed.	-2147204830
`Error_Media_PresenceRequestFailed` The Presence Server could not perform the requested action.	-2147204840
`Error_Media_PresenceUnableToSetPrimaryContact`	–2147204848
`Error_Media_PropertyNotSupportedForCallback`	–2147205025
`Error_Media_PropertyNotSupportedForNonVoiceContact`	–2147204847
`Error_Media_ServerNotConfigured` The SDK could not perform the requested action on the server because this server is not configured (SID-enabled). Ensure that the server is SID-enabled before attempting that action again.	–2147205019

Table 19 enErrors enumeration (Seite 21 von 23)

Enumeration	Value
`Error_Media_UnknownQueryCallState` The Query failed because the CallStates being queried are not supported. For a list of supported CallStates please refer to the Query.	–2147205007
`Error_Media_WebCollaborationAgentAlreadyInSession` This indicates that the Web Interaction Server could not perform the requested action because the user is handling a contact in another session.	–2147204856
`Error_Media_WebCollaborationDataCorrupted` This indicates that the Web Interaction Server could not perform the requested action because the request or reply data is corrupted and unusable.	–2147204862
`Error_Media_WebCollaborationInvalidAgentKey` The Web Interaction Server could not perform the requested action because an invalid agent Key was provided. If this occurs, there may be an error in the code that is using the SDK or in the input that you are assign in.	–2147204863
`Error_Media_WebCollaborationInvalidCallID` The Web Interaction Server could not perform the requested action because an invalid call ID was provided. If this occurs, there may be an error in the code that is using the SDK or in the input that you are passing in.	–2147204864
`Error_Media_WebCollaborationRPCFailed` This indicates that the Web Interaction Server could not perform the requested action because the system was unable to send the request to the server (Web Interaction Server may not be running)	–2147204860
`Error_Media_WebCollaborationRPCTimeout` This indicates that the Web Interaction Server could not perform the requested action because the request exceeded the maximum allowed time (Routing Server may not be running)	–2147204861
`Error_Media_WebCollaborationServerTerminating` This indicates that the Web Interaction Server could not perform the requested action because the Web Interaction Server is terminating.	–2147204857
`Error_Media_WebCollaborationNotConnected` SDK does not have connection to Web Interaction Server. Server may be down or maybe there are network problems. For more information, consult your documentation or contact your system administrator.	Unknown
`Error_Media_WebCollaborationUnableToAllocateMemory` This indicates that the Web Interaction Server could not perform the requested action because the application was unable to allocate sufficient memory for the data to be returned.	–2147204858
`Error_Media_WebCollaborationWrongVersion` This indicates that the Web Interaction Server could not perform the requested action because the request was not compatible with the current version of the Routing Server	–2147204859

Table 19 enErrors enumeration (Seite 22 von 23)

Enumeration	Value
`Error_Routing_InvalidQueue` The Routing Server could not perform the requested action because you have entered an invalid QueueKey. If this occurs, there may be an issue with the SDK, in which case you may want to notify Support to determine if this is a known issue, and request a fix.	–2147201024
`Error_Statistics_StopListeningForEventsFailed` The Statistics Manager could not perform your request to Stop Listening For Events. For more information, consult your documentation or contact your system administrator.	–2147196927

Table 19 enErrors enumeration (Seite 23 von 23)

3.19 enEventModes

The enEventModes enumeration represents the various event modes that the SDK can be run in. These event modes determine if events can be listened for by the client applications through the various Manager objects, as well as how those events are sent to the client application.

For more information about events, see Section 2.8, "About events", on page 37.

Enumeration	Value
`EventMode_FireEvents` The event mode for the various Manager objects should listen for events and then send the event back to the client application when they occur. If the client application is busy when the event is sent back, the event will remain at the head of the queue and the Manager will attempt to send it again after a brief, system defined, polling interval.	1
`EventMode_IgnoreEvents` The e-mail state after an e-mail has received a reply. An e-mail reply can be either a response from a contact center user, or an automatic reply, generated by the E-mail Server (for example, for product information).	100
`EventMode_NotSet` The event mode for the various Manager objects has not been set. This should only be seen when first constructing the OpenScape Contact Center Manager and before it has been initialized.	0

Table 20 enEventModes enumeration

3.20 enHandlingStates

The enHandlingStates enumeration indicates the state of a user while they are handling a contact. For example, if a user is talking on the telephone to a customer, then the HandlingState_Talking enumeration represents this action.

Enumeration	Value
`HandlingState_Consulting` The user is consulting with another user.	7
`HandlingState_Dialing` A user is dialing from an extension.	1
`HandlingState_Holding` The user is handling a call or callback and has placed the call on hold.	6
`HandlingState_LineBusy` The user's extension is busy.	2
`HandlingState_LineQueued` The user’s call is queued on another extension (for example, camped on or parked).	5
`HandlingState_Pending` The user has a contact that is pending for them. This occurs when a contact has been assigned to a user, but in the process of getting routed to the user.	8
`HandlingState_PostProcessing` The user has disconnected from a contact but is still performing tasks related to the contact, such as making notes or sending follow-up information.	10
`HandlingState_Processing` Indicates one of the following: ● For callback, the user is previewing a routed callback offer, but has not yet dialed the telephone number or the routed callback has been accepted by the system and the user is reviewing the callback data before dialing the telephone number. ● For e-mail, the user is handling a routed e-mail message, or has resumed handling a deferred or externally consulted e-mail message. ● For Web collaboration, the user is handling a routed Web collaboration session.	9
`HandlingState_Ringing` The user's extension is ringing.	3
`HandlingState_Talking` The user is connected to a call.	4
`HandlingState_Unknown` A user's current activity is unknown.	0

Table 21 enHandlingStates enumeration

3.21 enHiPathProCenterEventCodes

The enHiPathProCenterEventCodes enumeration represents the various types of events that can occur in the OpenScape Contact Center Manager. To receive these events, you must have initialized the HiPathProCenterManager with events (that is, EventMode_FireEvents), and registered for an event type through the ListenForEvents method, passing in a valid enHiPathProCenterEventType.

Enumeration	Value
`HiPathProCenterEventCode_ManagerStateChanged` You have a ManagerStateChanged event. These are sent when the HiPathProCenterManager has a necessary background server that has gone down or come back up. To receive these, start the HiPathProCenterManager with events. You do not need to explicitly listen for any events.	1002
`HiPathProCenterEventCode_None` You have a HiPathProCenterEvent that does not correspond to something that has happened. This should never be seen.	1000
`HiPathProCenterEventCode_ServerStateChanged` You have a ServerStateChanged event. These are sent when a server in the system has gone down. To receive these events, you need to explicitly listen for the HiPathProCenterEventType_ServerStateChangedEvents. It is strongly recommended that you use the '*' resource.	1003

Table 22 enHiPathProCenterEventCodes enumeration

3.22 enHiPathProCenterEventObjectTypes

The enHiPathProCenterEventObjectTypes enumeration represents the various types of event objects that can be returned from the HiPathProCenterManager. When an event occurs in the HiPathProCenterManager, the event is returned through the EventOccurred event handler from the HiPathProCenterManager as a HiPathProCenterEvent object. You can then check the ObjectType property to determine what type of underlying object you have.

Enumeration	Value
`HiPathProCenterEventObjectType_Base` You have a HiPathProCenterEvent that does not correspond to a valid object. This should never be seen.	0
`HiPathProCenterEventObjectType_ManagerStateChanged` You have a HiPathProCenterEvent that corresponds to a ManagerStateChanged object. This is returned when there is a change in the Manager state (when you have initialized the HiPathProCenterManager with events). See Section 4.5, “ManagerStateChangedEvent”, on page 135.	9899
`HiPathProCenterEventObjectType_ServerStateChanged` You have a HiPathProCenterEvent that corresponds to a ServerStateChanged object. This is returned when there is a change in the state of the HiPathProCenterManager connection to any of the servers. This can be used for system status type functionality.	9799

Table 23 enHiPathProCenterEventObjectTypes enumeration

3.23 enHiPathProCenterEventTypes

The enHiPathProCenterEventTypes enumeration represents the various types of event groups that can be listened for from the HiPathProCenterManager. When these events are passed into the ListenForEvents and StopListeningForEvents methods, the HiPathProCenterManager will register for a variety of individual events. For specific events, see Section 3.21, “enHiPathProCenterEventCodes”, on page 83.

Enumeration	Value
`HiPathProCenterEventType_NotSet` The event that is received does not correspond to an event that has been explicitly listened for (that is, a ManagerStateChanged event).	1000
`HiPathProCenterEventType_ServerStateChangedEvents` This represents the Server State changes in the HiPathProCenterManager group of events These include such things as connecting to servers or being disconnected from a server. It is strongly recommended that you use the '*' resource.	1100

Table 24 enHiPathProCenterEventTypes enumeration

3.24 enHiPathProCenterFunctionalities

The enHiPathProCenterFunctionalities enumeration represents the various function groups that exist in the HiPathProCenterManager. These are used when asking the HiPathProCenterManager what functionality is supported. It might be worthwhile to check if certain functionality is available before allowing certain actions in your application. For example, if you know that the Administration Server is unavailable, you may want to prevent people from attempting to do anything else, such as creating other Managers.

Enumeration	Value
`HiPathProCenterFunctionality_Connectivity` This indicates whether the HiPathProCenterManager has connectivity to the Administration Server. This is critical, since all the connections to the other servers that are used by the rest of the Managers in the SDK is based on the information contained in the database.	0
`HiPathProCenterFunctionality_SystemStatus` This indicates whether the HiPathProCenterManager has connectivity to the Watchdog Server. This retrieves detailed system status information.	1

Table 25 enHiPathProCenterFunctionalities enumeration

3.25 enHoldReasons

The enHoldReasons enumeration represents the reasons why a contact is held.

Enumeration	Value
`HoldReason_AgentDefer`	1
`HoldReason_AutoDefer`	2
`HoldReason_ExternalConsult`	3
`HoldReason_Undefined`	0

Table 26 enHoldReasonsTypes enumeration

3.26 enKeyValuePairTypes

The enKeyValuePairTypes enumeration represents the various types of key/value pairs that are used in the ContactData collection.

Enumeration	Value
`KeyValuePairType_Binary` The key/value pair type has a value that is binary data.	1
`KeyValuePairType_String` The key/value pair type has a value of type string.	0

Table 27 enKeyValuePairTypes enumeration

3.27 enLocaleIDs

The enLocaleIDs enumeration sets and returns the value of the locale ID associated with the language used when a hard-coded localized string is required by the system. The LocaleID is currently only used when handling e-mail messages. By default, the locale ID is set to LocaleID_None.

Enumeration	Value
`LocaleID_English`	1033
`LocaleID_French`	1036
`LocaleID_German`	1031
`LocaleID_Italian`	1040
`LocaleID_None`	0
`LocaleID_Portuguese`	1046
`LocaleID_Spanish`	1034

Table 28 enLocaleIDs enumeration

3.28 enManagerStates

The enManagerStates enumeration represents the various states of Managers. It is also used to represent the functionality states for the various functions of the Manager objects (through the GetFunctionalityState methods).

Enumeration	Value
`ManagerState_Available` A Manager is available. When a ManagerStateChangedEvent is returned with this state, it means that all of the servers in the Manager that are returning this event are up. If this is returned through the GetFunctionalityState method, it means that the functionality requested is available.	3
`ManagerState_PartiallyAvailable` A Manager is partially available. When a ManagerStateChangedEvent is returned with this state, it means that one or more of the servers in the Manager that is returning this event are down, but at least one server is up. To determine what functionality is available, use the GetFunctionalityState method of the appropriate Manager. This is never returned through the GetFunctionalityState method.	2
`ManagerState_Unavailable` A Manager is unavailable. When a ManagerStateChangedEvent is returned with this state, it means that all the servers in the Manager that is returning this event are down. If this is returned through the GetFunctionalityState method, it means that the functionality requested is not available.	1

Table 29 enManagerStates enumeration

3.29 enMediaBaseObjectTypes

The enMediaObjectTypes enumeration represents the various types of objects that can be passed into the main methods of the Media Manager (that is, into methods such as Query). These are used with the MediaBase interface to provide a standard way to pass around Media objects.

When you have a MediaBase object, you can retrieve the interface to the underlying type by setting a variable of the underlying type to the AdministrationBase object. For example:

```
Dim objCall as HiPathProCenterLibrary.Call
If (objMediaBase.ObjectType =
   MediaBaseObjectType_VoiceCall) Then
Set objCall = objMediaBase
   '...then can call the various call methods
   ' and properties as required
```

```
Endif
```

Enumeration	Value
`MediaBaseObjectType_Agent` The MediaBase object is actually a simplified interface to the Agent object.	9
`MediaBaseObjectType_Base` This represents a MediaBase object type that does not correspond to an acceptable Media Manager object type. This should never be seen.	0
`MediaBaseObjectType_Callback` The MediaBase object is actually a simplified interface to the CallbackCall object.	2
`MediaBaseObjectType_Email` The MediaBase object is actually a simplified interface to the EmailCall object.	3
`MediaBaseObjectType_ServicesEnabled` The MediaBase object is actually a simplified interface to the ServicesEnabled object.	5
`MediaBaseObjectType_VoiceCall` The MediaBase object is actually a simplified interface to the VoiceCall object.	1
`MediaBaseObjectType_WebCollaboration` The MediaBase object is actually a simplified interface to the WebCollaborationCall object.	4

Table 30 enMediaObjectTypes enumeration

3.30 enMediaEventCodes

The enMediaEventCodes enumeration represents the various types of events that can occur in the MediaManager. To receive these events, you must have created the MediaManager with events (that is, EventMode_FireEvents), and registered for an event type through the ListenForEvents method, passing in a valid enMediaEventType.

Enumeration	Value
`MediaEventCode_AgentStatus` An AgentStatus event has occurred. An AgentStatus event is sent when a user changes state (Available, Unavailable, Work, and so on). To receive these, start the MediaManager with events and listen for MediaEventType_AgentStatusEvents (specifying the agent key for the user you want to monitor, or '*' for all users)	3900
`MediaEventCode_AutoAcknowledged` An AutoAcknowledged event has occurred. An AutoAcknowledged event is sent when the e-mail server has sent an auto-acknowledgment to the customer (that is, "Thanks for your e-mail. Someone will answer it within 24 hours."). To receive these, start the MediaManager with events and listen for MediaEventType_EmailEvents (specifying the agent key for the user you want to monitor, or '*' for all users).	3190
`MediaEventCode_CallInformationUpdated` A CallInformationUpdated event has occurred. A DetailsUpdated event is sent when the details for the Call (specified by the Call ID) have changed (for voice), or when the details for the call that the user is on have been changed (for e-mail and Web collaboration). To receive these, start the MediaManager with events and listen for the following: • MediaEventType_VoiceDetailsUpdated (specifying the Call ID for the call you want to monitor, or '*' for all calls) • MediaEventType_EmailDetailsUpdated (specifying the agent key for the user you want to monitor, or '*' for all users) • MediaEventType_ChatDetailsUpdated (specifying the agent key for the user you want to monitor, or '*' for all users), depending on the media types you want.	3210
`MediaEventCode_CallMeAnswered` A CallMeAnswered event has occurred. A CallMeAnswered event is sent when a customer has requested that a user call them at a telephone number. The customer will make the request for the user to call them back (through a Web collaboration message in their conversation). The user will send the PushCallMeForm to the customer. When the customer submits the telephone number from the CallMe form, this event will be sent. To receive these, start the MediaManager with events and listen for MediaEventType_ChatEvents (specifying the agent key for the user you want to monitor, or '*' for all users).	3230

Table 31 enMediaEventCodes enumeration (Seite 1 von 6)

Enumeration	Value
`MediaEventCode_Conferenced` A Conferenced event has occurred. A Conferenced event is sent when new party joins a 'call' (either a voice call or a Web collaboration session). To receive these, start the MediaManager with events and listen for the following: ● MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices) ● MediaEventType_ChatEvents (specifying the agent key for the user you want to monitor, or '*' for all users), depending on the media types you want.	3270
`MediaEventCode_ConferenceRequested` A ConferenceRequested event has occurred. A ConferenceRequested event is sent when a user in a call (for Web collaboration) wants to add another user to the Web collaboration session. The user in the current Web collaboration session will send this event to the user who they want to add. To receive these, start the MediaManager with events and listen for MediaEventType_ChatEvents (specifying the agent key for the user you want to monitor, or '*' for all users).	3260
`MediaEventCode_ConferenceRequestRefused` A ConferenceRequestRefused event has occurred. A ConferenceRequestRefused event is sent when a user wants to add another user to the Web collaboration session. The first user will have sent a ConferenceRequested event to the second user. When the second user refuses the conference request (through the RefuseParticipantRequest), the ConferenceRequestRefused event is sent out. To receive these, start the MediaManager with events and listen for MediaEventType_ChatEvents (specifying the agent key for the user you want to monitor, or '*' for all users).	3280
`MediaEventCode_ContentAdded` A ContentAdded event has occurred. A ContentAdded event is sent when someone involved in a Web collaboration session sends a new message to all parties in the Web collaboration session. To receive these, start the MediaManager with events and listen for MediaEventType_ChatEvents (specifying the agent key for the user you want to monitor, or '*' for all users).	3220

Table 31 enMediaEventCodes enumeration (Seite 2 von 6)

Enumeration	Value
`MediaEventCode_Delivered` A Delivered event has occurred. A Delivered event is sent when a 'call' has been delivered to the user's workstation. For example, in the case that the monitored device is an extension, then the telephone would be ringing at this point. For e-mail and Web collaboration, the user would receive notification through the desktop PC. For voice events, these can also be received when a call is delivered to a Route Control Group, ACD group, UCD group, or hunt troup, although an extension is usually the target. To receive these, start the MediaManager with events and listen for the following: ● MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices) ● MediaEventType_EmailEvents (specifying the agent key for the user you want to monitor, or '*' for all users) ● MediaEventType_ChatEvents (specifying the agent key for the user you want to monitor, or '*' for all users), depending on the media types you want.	3170
`MediaEventCode_Disconnected` A Disconnect event has occurred. A Disconnect event is sent when a party disconnects from a call (for any media type). To receive these, start the MediaManager with events and listen for the following: ● MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices) ● MediaEventType_EmailEvents (specifying the agent key for the user you want to monitor, or '*' for all users) ● MediaEventType_ChatEvents (specifying the agent key for the user you want to monitor, or '*' for all users), depending on the media types you want.	3310
`MediaEventCode_Diverted` A Diverted event has occurred. A Diverted event is sent when a call has been diverted to the monitored device. This is usually an extension, but can be to another Route Control Group, ACD group, UCD group or hunt group. To receive these, start the MediaManager with events and listen for the following: ● MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices) ● MediaEventType_EmailEvents (specifying the agent key for the user you want to monitor, or '*' for all users), depending on the media types you want.	3160
`MediaEventCode_EmailsHistoryQueryResult` A ScheduledCallbacksQueryResultEvent event has occurred. This is used to deliver the results of a previous call to QueryAsyncScheduledCallbacks ().	3360

Table 31 enMediaEventCodes enumeration (Seite 3 von 6)

Enumeration	Value
`MediaEventCode_Established` An Established event has occurred. An Established event is sent when the connection between the customer and the user is established. For voice, this is when the telephone is answered. For e-mail messages, this is when the e-mail message is accepted by the user. For Web collaboration, this is when the first message is sent from the customer to the contact center user. To receive these, start the MediaManager with events and listen for the following: ● MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices) ● MediaEventType_EmailEvents (specifying the agent key for the user you want to monitor, or '*' for all users) ● MediaEventType_ChatEvents (specifying the agent key for the user you want to monitor, or '*' for all users), depending on the media types you want.	3200
`MediaEventCode_Failed` A Failed event has occurred. A Failed event is sent when the destination of a call is busy (for voice) or during a time out from the Routing Server (for Web collaboration). To receive these, start the MediaManager with events and listen for the following: ● MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices) ● MediaEventType_ChatEvents (specifying the agent key for the user you want to monitor, or '*' for all users), depending on the media types you want.	3140
`MediaEventCode_Held` A Held event has occurred. A Held event is sent when a call is placed on hold or an e-mail message is deferred. To receive these, start the MediaManager with events and listen for the following: ● MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices) ● MediaEventType_EmailEvents (specifying the agent key for the user you want to monitor, or '*' for all users), depending on the media types you are interested in.	3240
`MediaEventCode_ManagerStateChanged` A ManagerStateChanged event has occurred. A ManagerStateChanged event is sent when the MediaManager Manager has a necessary background server that has gone down or come back up. To receive these, start the MediaManager with events. You do not need to explicitly listen for these events.	3002

Table 31 enMediaEventCodes enumeration (Seite 4 von 6)

Enumeration	Value
`MediaEventCode_MediaNotReady` A MediaNotReady event has occurred. A MediaNotReady event is sent when the system is not ready to route contacts. This can occur if the T-Server, Callback Server, E-mail Server, Web Interaction Server, communication platform, corporate e-mail server, or corporate Web server is not operational. When the SDK application calls the HireMediaManager method, it listens for the MediaNotReady event by default.	3951
`MediaEventCode_MediaReady` A MediaReady event has occurred. A MediaReady event is sent when the system is ready to route contacts. When the SDK application calls the HireMediaManager method, it listens for the MediaReady event by default.	3950
`MediaEventCode_NetworkReached` A NetworkReached event has occurred. A NetworkReached event is sent when a monitored resource (extension) calls an external, T-Server monitored, device. In most cases, this will be at another site, in a multisite environment. To receive these, start the MediaManager with events and listen for the following: ● MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices) ● MediaEventType_EmailEvents (specifying the agent key for the user you want to monitor, or '*' for all users), depending on the media types you want.	3300
`MediaEventCode_OffHook` An OffHook event has occurred. An OffHook event is sent when a user on the monitored device takes their telephone off hook. To receive these, start the MediaManager with events and listen for MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices).	3110
`MediaEventCode_Originated` An Originated event has occurred. An Originated event is sent when a user on the monitored device completes dialing the number and the call is begun. To receive these, start the MediaManager with events and listen for MediaEventType_VoiceEvents (specifying the device you want monitor, or '*' for all devices).	3120

Table 31 enMediaEventCodes enumeration (Seite 5 von 6)

Enumeration	Value
`MediaEventCode_Queued` A Queued event has occurred. A Queued event is sent when a call has been queued to any monitored device. For voice, this can be any RCG (including the queue), or the users extension (in a camp on scenario). This event is also sent back when an e-mail or Web collaboration session that a user is on is requeued. To receive these, start the MediaManager with events and listen for the following: ● MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices) ● MediaEventType_EmailEvents (specifying the agent key for the user you want to monitor, or '*' for all users) ● MediaEventType_ChatEvents (specifying the agent key for the user you want to monitor, or '*' for all users, depending on the media types you want).	3150
`MediaEventCode_Retrieved` A Retrieved event has occurred. A Retrieved event is sent when a call that was placed on hold or an e-mail message that was deferred is retrieved. To receive these, start the MediaManager with events and listen for the following: ● MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices) ● MediaEventType_EmailEvents (specifying the agent key for the user you want to monitor, or '*' for all users), depending on the media types you want.	3250
`MediaEventCode_ScheduledCallbacksQueryResult`	3330
`MediaEventCode_ScheduledCallbacksQueryResult`	3330
`MediaEventCode_ServicesEnabledEvent` A ServicesEnabled event has occurred. A ServicesEnabled event is sent whenever a routing state or line state changes. To receive this event, start the MediaManager with events and listen for MediaEventType_VoiceEvents (specifying the Extension as a resource you want to monitor, or '*' for all extensions)MediaEventCode_Disconnected=3310	3320
`MediaEventCode_Transferred` A Transferred event has occurred. A Transferred event is sent when a call or an e-mail message has been transferred to another queue or extension. To receive these, start the MediaManager with events and listen for the following: ● MediaEventType_VoiceEvents (specifying the device you want to monitor, or '*' for all devices) ● MediaEventType_EmailEvents (specifying the agent key for the user you want to monitor, or '*' for all users), depending on the media types you want.	3290

Table 31 enMediaEventCodes enumeration (Seite 6 von 6)

3.31 enMediaEventObjectTypes

The enMediaEventObjectTypes enumeration represents the various types of event objects that can be returned from the MediaManager. When an event occurs in the MediaManager, the event is returned through the EventOccurred event handler from the MediaManager as a MediaEvent object. You can then check the ObjectType property to determine what type of underlying object you have.

Enumeration	Value
`MediaEventObjectType_AgentStatus` You have an AgentStatusEvent object. An AgentStatus event is sent when a user changes their routing state (Available, Unavailable, Work, and so on).	3900
`MediaEventObjectType_AutoAcknowledged` You have an AutoAcknowledgedEvent object. This is returned when an e-mail is received by the E-mail Server and an auto-acknowledgment is sent to them (that is, "Thanks for your e-mail. We'll get back to you within 24 hours.".	3190
`MediaEventObjectType_Base` You have a MediaEvent that does not correspond to a valid object. This should never be seen.	0
`MediaEventObjectType_CallInformationUpdated` You have a CallInformationUpdatedEvent object. This is returned when a call's details (for example, Description) are updated in the system. The ContactData property will not be exposed. For Contact Data updates use MediaEventObjectType_ContactDataUpdated.	3210
`MediaEventObjectType_CallMeAnswered` You have a CallMeAnsweredEvent object. This is returned when a customer has requested that a user call them at a telephone number. The customer will ask the user to call them (through text in the Web collaboration session), the user will then push the Call Me form to the customer asking for their telephone number. When the customer submits that page, the CallMeAnswered event is sent back.	3230
`MediaEventObjectType_Conferenced` You have a VoiceConferencedEvent object. This is returned when another party has joined the call (voice or Web collaboration).	3270
`MediaEventObjectType_ConferenceRequested` You have a ConferenceRequestedEvent object. This is returned when a user in the Web collaboration session, sends a request to a user not in the Web collaboration session. The new user can choose to join the conference or refuse the request.	3260
`MediaEventObjectType_ConferenceRequestRefused` You have a ConferenceRequestRefusedEvent object. This is returned when a new party refuses the request to join the Web collaboration session.	3280

Table 32 enMediaEventObjectTypes enumeration (Seite 1 von 3)

Enumeration	Value
`MediaEventObjectType_ContactDataUpdated` You have a CallInformationUpdatedEvent object. This is returned when a call's Contact Data is updated in the system. Only the CallID and ContactData properties will be exposed.	3211
`MediaEventObjectType_ContentAdded` You have a ContentAddedEvent object. This is returned when there is data that is entered in the Web collaboration session. This is sent to all parties that are participating in the Web collaboration session.	3220
`MediaEventObjectType_Delivered` You have a DeliveredEvent object. This is returned when a call has been delivered to the monitored device (for example for voice, the monitored extension will be ringing at this point) or to a user (e-mail or Web collaboration).	3170
`MediaEventObjectType_Disconnected` You have a DisconnectedEvent object. This is returned when a call connected to a user is disconnected (voice, Web collaboration or e-mail).	3310
`MediaEventObjectType_Diverted` You have a DivertedEvent object. This is returned when a call has been diverted to the monitored device (voice) or to a user (e-mail). Usually this is an extension, but can be another Route Control Group, ACD group, UCD group, or hunt group.	3160
`MediaEventObjectType_EmailsHistoryQueryResult`	3360
`MediaEventObjectType_EmailsHistoryQueryResult`	3360
`MediaEventObjectType_Established` You have an EstablishedEvent object. This is returned when a call has been answered.	3200
`MediaEventObjectType_Failed` You have a FailedEvent object. This is returned when an action performed on the monitored extension experiences an error (for example, the number dialed is busy), or when a Web collaboration contact experiences a time out from the Routing Server. This is sent for voice and Web collaboration contacts.	3140
`MediaEventObjectType_Held` You have a VoiceHeldEvent object. This is returned when a call on the monitored device has been placed on hold, or when an e-mail is deferred (held) to be completed later. For example, this may occur if a user is awaiting an answer from another source.	3240
`MediaEventObjectType_ManagerStateChanged` You have a MediaEvent that corresponds to a ManagerStateChanged object. This is returned when there is a change in the Manager state (when have created the MediaManager with events). See Section 4.5, “ManagerStateChangedEvent”, on page 135.	9899
`MediaEventObjectType_MediaNotReady` You have a UserMediaNotReadyEvent object. This is returned when the system is not ready to route contacts.	3951

Table 32 enMediaEventObjectTypes enumeration (Seite 2 von 3)

Enumeration	Value
`MediaEventObjectType_MediaReady` You have a UserMediaReadyEvent object. This is returned when the system is ready to route contacts.	3950
`MediaEventObjectType_NetworkReached` You have a NetworkReachedEvent object. This is returned when a monitored resource (extension) calls an external device (monitored by the T-Server, usually at another site). This is usually in a multisite environment.	3300
`MediaEventObjectType_OffHookEvent` You have an OffHookEvent object. This is returned when the user on a monitored device takes their telephone off the hook (for example, when starting a new outgoing telephone call). This is sent only for voice calls.	3110
`MediaEventObjectType_OriginatedEvent` You have a VoiceOriginatedEvent object. This is returned when the user on a monitored device completes dialing a valid telephone number and the call is begun. This is sent only for voice calls.	3120
`MediaEventObjectType_Queued` You have a QueuedEvent object. This is returned when a call has been queued to a monitored device (voice), or to a monitored user (e-mail or Web collaboration). For most instances this is a queue, but it is possible for this to be returned when the monitored device is an extension (for example, in a camp-on scenario).	3150
`MediaEventObjectType_Retrieved` You have a VoiceRetrievedEvent object. This is returned when a call that had previously been placed on hold, has been taken off of hold, or when an e-mail that has been deferred (held) earlier, is retrieved.	3250
`MediaEventObjectType_ScheduledCallbacksQueryResult`	3350
`MediaEventObjectType_ScheduledCallbacksQueryResult` You have a ScheduledCallbacksQueryResultEvent object. This is returned to deliver the results of a previous call to QueryAsyncScheduledCallbacks ().	3350
`MediaEventObjectType_ServicesEnabled` You have a ServicesEnabledEvent object. This is returned whenever there is a Line event or user event at a user's extension and CTI services information changes in the system.	3320
`MediaEventObjectType_Transferred` You have a TransferredEvent object. This is returned when a call has been transferred. To do this, a call is first placed on hold. Then, a new number is dialed. After the call has been delivered to the new extension (that is, the telephone at the number you dialed is ringing), the transferred event will be sent. This can also be returned when a user has transferred an e-mail back to the queue or has reserved it for another user (that is, the e-mail is no longer assigned to the user).	3290

Table 32 enMediaEventObjectTypes enumeration (Seite 3 von 3)

3.32 enMediaEventTypes

The enMediaEventTypes enumeration represents the various types of event groups that can be listened for from the MediaManager. When these events are passed into the ListenForEvents and StopListeningForEvents methods, the MediaManager will register for a variety of individual events. For specific events, see Section 3.30, “enMediaEventCodes”, on page 89.

Enumeration	Value
`MediaEventType_AgentStatusEvents` This represents the group of AgentStatus events. An AgentStatus event is sent when a user changes their routing state (Available, Unavailable, Work, and so on). The resource you use to register for these events is the user key for the user you want to receive AgentStatus events for. You can use '*' to receive AgentStatus events for all users in the system.	3900
`MediaEventType_CallbackEvents` This represents the group of Callback events. Listening for callback events with this event type will cause events DeliveredEvent and FailedEvent to be raised with the media type set to MediaType_Callback.	3200
`MediaEventType_EmailDetailsUpdates` This represents the group of e-mail Details Updates events. The E-mail Details Updates events occur when the details of an e-mail are changed (for example, the Contact Data). The resource you use to register for these events is the agent key of the user handling e-mail messages that have the e-mail Details Updated. You can use '*' to receive detailed updates for all e-mail messages in the system.	3301
`MediaEventType_EmailEvents` This represents the group of e-mail events. The e-mail events are any of the various events that can occur with an e-mail message. These refer to such things as Transferred events, Held events, and so on. The resource you use to register for these events is the agent key for the user who you want to receive e-mail events for. You can use '*' to receive e-mail events for all users in the system.	3300
`MediaEventType_MediaReadinessEvents` This represents the group of media readiness events. The MediaReadyEvent is sent when the system is ready to route contacts. The MediaNotReady event is sent when the system is not ready to route contacts.	3950
`MediaEventType_NotSet` The event that is received does not correspond to an event that has been explicitly listened for (that is, a ManagerStateChanged event).	3000

Table 33 enMediaEventTypes enumeration (Seite 1 von 2)

Enumeration	Value
`MediaEventType_WebCollaborationDetailsUpdates` This represents the group of WebCollaborationDetailsUpdates events. WebCollaborationDetailsUpdates events occur when the details of a Web collaboration contact are changed (for example, the Contact Data). The resource you use to register for these events is the agent key of the user handling the Web collaboration sessions that have the e-mail details updated. You can use '*' to receive details updates for all Web collaboration sessions in the system.	3401
`MediaEventType_WebCollaborationEvents` This represents the group of Web collaboration events. Web collaboration events are any of the various events that can occur with a Web collaboration session. These refer to such things as Content Added events, Conference Join Requests, and so on. The resource you use to register for these events is the agent key for the user who you want to receive Web collaboration events for. You can use '*' to receive Web collaboration events for all users in the system.	3400
`MediaEventType_VoiceDetailsUpdates` This represents the group of Voice Details Updates events. Voice Details Updates events occur when the details of a Voice call are changed (for example, the Contact Data). The resource you use to register for these events is the device, on which you will receive events when calls are updated. You can use '*' to receive details updates for all voice calls in the system.	3101
`MediaEventType_VoiceEvents` This represents the group of Voice events. Voice events are any of the various events that can occur with a voice call. These refer to such things as Established events, Connected events, and so on. The resource you use to register for these events is the device that you want to receive Voice events for. For example, this could be an extension or a queue. You can use '*' to receive Voice events for all monitored devices in the system.	3100

Table 33 enMediaEventTypes enumeration (Seite 2 von 2)

3.33 enMediaFunctionalities

The enMediaFunctionalities enumeration represents the various function groups that exist in the MediaManager. These are used when asking the MediaManager what functionality is supported. It might be worthwhile to check if certain functionality is available before allowing certain actions in your application. For example, if you know that the Telephony functionality is unavailable, you may want to prevent people from querying Calls objects.

Enumeration	Value
`MediaFunctionality_Callback` This represents the Callback functionality group. These include such things as updating a querying a callback, or creating a callback.	2
`MediaFunctionality_Email` This represents the E-mail functionality group. These include such things as updating and querying an e-mail message, or deferring an e-mail message.	4
`MediaFunctionality_WebCollaboration` This represents the Web collaboration functionality group. These include such things as updating a querying a Web collaboration object, or adding text to a Web collaboration session.	3
`MediaFunctionality_Voice` This represents the T-Server (voice) functionality group. These include such things as updating a querying a Call, or answering a Call.	1

Table 34 enMediaFunctionalities enumeration

3.34 enMediaTypes

The enMediaTypes enumeration represents the various Media Types that are used in the system.

Enumeration	Value
`MediaType_All` All the Media Types. This is not used, but it is included for future expansion.	3500
`MediaType_Callback`	3200
`MediaType_Email`	3300
`MediaType_NotSet` The Media Type is not set. This applies mainly when the Media Type property is not applicable for an event (for example, the ManagerStateChangedEvent from the Media Manager will have the Media Type set to MediaType_NotSet from the MediaEvent interface).	0

Table 35 enMediaTypes enumeration (Seite 1 von 2)

Enumeration	Value
MediaType_Voice	3100
MediaType_WebCollaboration	3400

Table 35 enMediaTypes enumeration (Seite 2 von 2)

3.35 enPartyTypes

The enPartyTypes enumeration represents the various Party Types that are used in the Party object.

Enumeration	Value
PartyType_Agent	3
PartyType_External	1
PartyType_Internal	2
PartyType_IVR	7
PartyType_Queue	6
PartyType_RCG	5
PartyType_Trunk	4
PartyType_Unknown An Unknown Party Type. This should never be seen.	0

Table 36 enPartyTypes enumeration

3.36 enPresenceStates

The enPresenceStates enumeration represents the various presence states that the user can be in.

Enumeration	Value
PresenceState_Active	4
PresenceState_Away	3
PresenceState_Busy	2
PresenceState_Idle	1
PresenceState_LoggedOff	5
PresenceState_Unknown	0

Table 3-1

3.37 enResults

The enResults enumeration

Enumeration	Value
Result_Failure	1
Result_Success	0

Table 37 enResults enumeration

3.38 enRoutingBaseObjectTypes

The enRoutingObjectTypes enumeration represents the various types of objects that can be passed into the main methods of the RoutingManager (that is, into methods such as Query). These are used with the RoutingBase interface to provide a standard way to pass around Routing objects. This is useful for event handling, as well as passing of parameters into common methods (as seen in the RoutingManager).

When you have an RoutingBase object, you can retrieve the interface to the underlying type by setting a variable of the underlying type to the RoutingBase object. For example:

```
Dim objSBRCall as HiPathProCenterLibrary.SkillsBasedRoutingCall
If (objRoutingBase.ObjectType =
      RoutingBaseObjectType_RoutingCall) Then
   Set objRoutingCall = objRoutingBase
   '...then can call the various skills based routing
   ' call method and properties as required
Endif
```

Enumeration	Value
RoutingBaseObjectType_Base A RoutingBase object type that does not correspond to an acceptable Routing Manager object type. This should never be seen.	0
RoutingBaseObjectType_RoutingCall The RoutingBase object is actually a simplified interface to the RoutingCall object.	1

Table 38 enRoutingObjectTypes enumeration

3.39 enRoutingEventCodes

The enRoutingEventCodes enumeration represents the various types of events that can occur in the RoutingManager. To receive these events, first you must have created the RoutingManager with events (that is, EventMode_FireEvents). Second, you will have had to register for an event type through the ListenForEvents method, passing in a valid enRoutingEventType.

Enumeration	Value
`RoutingEventCode_Assigned` A call has been assigned. This occurs when a call is assigned to a user. To receive these, start the RoutingManager with events and listen for one or more of the following (depending on the media types you want): ● RoutingEventType_VoiceEvents ● RoutingEventType_CallbackEvents ● RoutingEventType_EmailEvents ● RoutingEventType_WebCollaborationEvents	4300
`RoutingEventCode_Dequeued` A call has been dequeued. This occurs when a call is dequeued from the Routing Server. To receive these, start the RoutingManager with events and listen for one or more of the following (depending on the media types you want): ● RoutingEventType_VoiceEvents ● RoutingEventType_CallbackEvents ● RoutingEventType_EmailEvents, ● RoutingEventType_WebCollaborationEvents	4600
`RoutingEventCode_DetailsUpdated` A call has had its details updated. This mainly occurs when a supervisor updates a call that is in queue. To receive these, start the RoutingManager with events and listen for one or more of the following (depending on the media types you want): ● RoutingEventType_VoiceEvents ● RoutingEventType_CallbackEvents ● RoutingEventType_EmailEvents ● RoutingEventType_WebCollaborationEvents	4200
`RoutingEventCode_Enqueued` A call has been Enqueued. This occurs when a call is added to the Call Table and has been queued to the Routing Server, and is ready to be matched to a user. To receive these, start the RoutingManager with events and listen for one or more of the following (depending on the media types you want): ● RoutingEventType_VoiceEvents ● RoutingEventType_CallbackEvents ● RoutingEventType_EmailEvents ● RoutingEventType_WebCollaborationEvents	4100

Table 39 enRoutingEventCodes enumeration (Seite 1 von 2)

Enumeration	Value
`RoutingEventCode_ManagerStateChanged` You have a ManagerStateChanged event. These are sent when the Routing Manager has a necessary background server that has gone down or come back up. To receive these, start the RoutingManager with events. You do not need to explicitly listen for any events.	4003
`RoutingEventCode_None` You have a RoutingEvent that does not correspond to something that has happened. This should never be seen.	4000
`RoutingEventCode_TimedOut` A call has timed-out. This occurs when a call has been enqueued for either the maximum amount of time (the call has reached the last step in the queue), or there are no users logged on that can handle this call. To receive these, start the RoutingManager with events and listen for one or more of the following (depending on the media types you want): • RoutingEventType_VoiceEvents • RoutingEventType_CallbackEvents • RoutingEventType_EmailEvents, • RoutingEventType_WebCollaborationEvents	4500
`RoutingEventCode_Unassigned` A call has been unassigned. This occurs when a call is unassigned from a user. To receive these, start the RoutingManager with events and listen for one or more of the following (depending on the media types you want): • RoutingEventType_VoiceEvents • RoutingEventType_CallbackEvents • RoutingEventType_EmailEvents • RoutingEventType_WebCollaborationEvents	4400

Table 39 enRoutingEventCodes enumeration (Seite 2 von 2)

3.40 enRoutingEventObjectTypes

The enRoutingEventObjectTypes enumeration represents the various types of event objects that can be returned from the RoutingManager. When an event occurs in the RoutingManager, the event is returned through the EventOccurred event handler from the RoutingManager as an RoutingEvent object. You can then check the ObjectType property to determine what type of underlying object you have.

Enumeration	Value
`RoutingEventObjectType_Assigned` You have a RoutingEvent that corresponds to an AssignedEvent object. This is returned when a call is assigned to a user in the Routing Server.	3001
`RoutingEventObjectType_Base` You have a RoutingEvent that does not correspond to a valid object. This should never be seen.	0
`RoutingEventObjectType_ContactDataUpdated` You have a RoutingEvent that corresponds to a RoutingInformationUpdatedEvent object. This is returned when a call ContactData is updated in the Routing Server.	2002
`RoutingEventObjectType_Dequeued` You have a RoutingEvent that corresponds to a DequeuedEvent object. This is returned when a call has been dequeued in the Routing Server.	6001
`RoutingEventObjectType_Enqueued` You have a RoutingEvent that corresponds to an EnqueuedEvent object. This is returned when a call is enqueued to the Routing Server.	1001
`RoutingEventObjectType_ManagerStateChanged` You have a RoutingEvent that corresponds to a ManagerStateChanged object. This is returned when there is a change in the Manager state (when have created the RoutingManager with events). See Section 4.5, “ManagerStateChangedEvent”, on page 135.	9899
`RoutingEventObjectType_TimedOut` You have a RoutingEvent that corresponds to a TimedOutEvent object. This is returned when a call has timed-out in the Routing Server.	5001
`RoutingEventObjectType_Unassigned` You have a RoutingEvent that corresponds to an UnassignedEvent object. This is returned when a call is unassigned from a user in the Routing Server.	4001
`RoutingEventObjectType_Updated` You have a RoutingEvent that corresponds to a RoutingInformationUpdatedEvent object. This is returned when a call details (such as Description, Priority) are updated in the Routing Server. For Contact Data updates use type RoutingEventObjectType_ContactDataUpdated.	2001

Table 40 enRoutingEventObjectTypes enumeration

3.41 enRoutingEventTypes

The enRoutingEventTypes enumeration represents the various types of event groups that can be listened for from the RoutingManager. When these events are passed into the ListenForEvents and StopListeningForEvents methods, the RoutingManager will register for a variety of individual events. For specific events, see Section 3.39, “enRoutingEventCodes”, on page 103.

Enumeration	Value
`RoutingEventType_CallbackEvents` This represents the group of Callback Call Routing events. These include such things as calls enqueued, assigned, and dequeued. The resource you use to register for these is the Call ID of the call that you want Callback Routing events for, or you can use '*' to receive all Callback Routing events for all callback calls.	4200
`RoutingEventType_EmailEvents` This represents the group of e-mail Routing events. These include such things as calls enqueued, assigned, and dequeued. The resource you use to register for these is the Call ID of the call that you want e-mail Routing events for, or you can use '*' to receive all e-mail Routing events for all e-mail messages.	4300
`RoutingEventType_NotSet` The event that is received does not correspond to an event that has been explicitly listened for (that is, a ManagerStateChanged event).	4000
`RoutingEventType_VoiceEvents` This represents the group of voice call Routing events. These include such things as calls enqueued, assigned, and dequeued. The resource you use to register for these is the Call ID of the call that you want voice Routing events for, or you can use '*' to receive all Voice Routing events for all voice calls.	4100
`RoutingEventType_WebCollaborationEvents` This represents the group of a Web collaboration session Routing events. These include such things as calls enqueued, assigned, and dequeued. The resource you use to register for these is the Call ID of the call that you want Web collaboration routing events for, or you can use '*' to receive all Web collaboration routing events for all Web collaboration sessions.	4400

Table 41 enRoutingEventTypes enumeration

3.42 enRoutingFunctionalities

The enRoutingFunctionalities enumeration represents the various function groups that exist in the RoutingManager. These are used when asking the RoutingManager what functionality is supported. You should check if certain functionality is available before allowing certain actions in your application. For example, if you know that the Routing Server is unavailable, you may want to prevent people from attempting to query a Skills Based Routing call.

Enumeration	Value
`RoutingFunctionality_LocalRouting` Indicates whether the RoutingManager has connectivity to the Routing Server. If the Routing Manager does not have connectivity with the Routing Server, then Routing Functionality will be unavailable (that is, using the SkillsBasedRoutingCall object to do such things as changing the queue).	1
`RoutingFunctionality_NetworkRouting` Indicates whether the RoutingManager has connectivity to the Routing Server. If the Routing Manager does not have connectivity with the Routing Server, then Network Routing Functionality will be unavailable.	2
`RoutingFunctionality_RealTime` Indicates whether the RoutingManager has connectivity to the Real-Time Server. If the Routing Manager does not have connectivity with the Real-Time Server, then Real Time information (that is, call queue) will be unavailable.	3

Table 42 enRoutingFunctionalities enumeration

3.43 enRoutingStates

The enRoutingStates enumeration represents the various routing states that the user can be in. For example, if a user is in the Available state, then the RoutingState_Available represents the Available state in the various objects that use the enRoutingStates enumeration.

Enumeration	Value
`RoutingState_Available` The user is logged on and available for calls.	1
`RoutingState_LoggedOff` The user is logged off from the system.	4
`RoutingState_Unavailable` The user is logged on and unavailable for calls.	2

Table 43 enRoutingStates enumeration (Seite 1 von 2)

Enumeration	Value
RoutingState_Unknown	0
RoutingState_Work The user is logged on and in Work state.	3

Table 43 enRoutingStates enumeration (Seite 2 von 2)

3.44 enServers

The enServers enumeration represents the various servers that exist in the system.

Enumeration	Value
enServers_Administration Administration Server. The Administration Server's primary role is to handle requests for configuration information from other servers and the client applications and control access to administration and configuration information stored in the database.	1
enServers_Callback Callback Server. The Callback Server's primary role is to handle the creation, scheduling, and processing of callbacks.	4
enServers_ConfigurationSynchronization Config Sync Server. The Config Sync Server's primary role is to synchronize resource configuration between the system and the communication platform.	2
enServers_Email E-mail Server. The E-mail Server's primary role is to coordinate the handling and management of e-mail messages, in conjunction with the corporate e-mail server.	5
enServers_None There is no server type set. This should never be seen.	0
enServers_Presence Presence Server. The Presence Server's primary role is to manage user states (handling state, presence state, and routing state).	12
enServers_RealTime Real-Time Server. The Real-Time Server's primary role is to collect real-time user and contact-related performance and adherence data from the other servers and provide the data for real-time and cumulative reports.	9
enServers_Routing Routing Server. The Routing Server's primary role is to determine the routing of all incoming contacts for all supported media types. The Routing Server is responsible for the execution of routing strategy and queue processing workflows and all processing associated with routing a contact to a user.	7

Table 44 enServers enumeration (Seite 1 von 2)

Enumeration	Value
enServers_Statistics Statistics Server. The Statistics Server's primary role is to monitor all transactions in the system and record meaningful information into the database for future reporting purposes.	10
enServers_Telephony T-Server. The T-Server's primary role is to handle communication between the communication platform and the system. This involves such tasks as monitoring the devices in the contact center (extensions) and passing information to the users desktop PC's when an event occurs.	3
enServers_Watchdog Watchdog Server. The Watchdog Server's primary role is to monitor all servers in the system, and ensure that the system is running smoothly. If any problems arise, the Watchdog Server should correct those problems.	11
enServers_WebCollaboration Interaction Server. The Web Interaction Server's primary role is to handle incoming Web collaboration session requests in the system.	6

Table 44 enServers enumeration (Seite 2 von 2)

3.45 enServerStates

The enServerStates enumeration represents the various server states that exist in the system. It's important to note that these represent the SDK's connection to these servers, and may not necessarily reflect the state of the server itself. For example, if the machine that is running an application built with the SDK loses its connection to the server, all servers will show as reconnecting. This is because the SDK is reconnecting its client side connection to the servers.

Enumeration	Value
enServerState_AttemptingReconnect The SDK is attempting to reconnect to the appropriate server. This happens after the SDK has initially connected to the appropriate server, but has lost its connection to the server. the SDK will attempt to reconnect to the server until the application is shut down.	2
enServerState_Connected The SDK is connected to the appropriate server. This is the most common state for the Server.	1
enServerState_Connecting The SDK is attempting to connect to the appropriate server. This happens when the SDK is connecting to the server for the first time.	0
enServerState_Disconnected The SDK has disconnected from the appropriate server. This occurs when shutting down an application built using the SDK.	4

Table 45 enServerStates enumeration (Seite 1 von 2)

Enumeration	Value
`enServerState_Disconnecting` The SDK is disconnecting from the appropriate server. This occurs when shutting down an application built using the SDK.	3
`ServerState_NotLicensed` The SDK has detected that The application is not licensed. This occurs when an already licensed SDK application detects the Administration Server shutting down/starting and before it reconsumes the license it was using, another SDK application consumes that license instead.	8
`enServerState_OutOfService` The out of service state in the SDK. This occurs when you still have a connection to a server, but that server is not functional. For example, in some configurations, the T-Server requires a heartbeat card configured. If the T-Server 'loses its heartbeat', then it thinks the communication platform is down and goes into the Out of Service state.	6
`enServerState_Overloaded` The overloaded state in the SDK. This occurs when there is an abundance of network traffic.	7
`enServerState_Unknown` An unknown server state in the SDK. The T-Server informs the SDK that it is in the Unknown state.	5
`enServerState_OutOfService` The out of service state in the SDK. This occurs when you still have a connection to a server, but that server is not currently functional. The SDK will not fire a ManagerStateChanged event and a call to the GetFunctionalityState method will return ManagerState_Available since the SDK is still connected to the server. For example, in some configurations, the T-Server requires a heartbeat card configured. If the T-Server 'loses its heartbeat', then it thinks the communication platform is down and goes into the Out of Service state.	9

Table 45 enServerStates enumeration (Seite 2 von 2)

3.46 enStatisticsEventCodes

The enStatisticsEventCodes enumeration represents the various types of events that can occur in the StatisticsManager Manager. To receive these events, you must have initialized the StatisticsManager with events (that is, EventMode_FireEvents).

Enumeration	Value
`StatisticsEventCode_ManagerStateChanged` You have a ManagerStateChanged event. These are sent when the StatisticsManager has a necessary background server that has gone down or come back up. To receive these, start the StatisticsManager with events. You do not need to explicitly listen for any events.	5003
`StatisticsEventCode_None` You have a StatisticsEvent that does not correspond to something that has happened. This should never be seen.	5000
`StatisticsEventCode_QueryResultsReceived` You have a statistic event that contains a collection of requested items. This event is sent after issuing the ListenForEvents request from Statistics Manager. The initial event will arrive shortly after the request is issued per the current Real-Time Server design. Subsequent events will arrive at the configured OpenScape Contact Center Manager refresh rate.	5102

Table 46 enStatisticsEventCodes enumeration

3.47 enStatisticsEventObjectTypes

The enStatisticsEventObjectTypes enumeration represents the various types of event objects that can be returned from the StatisticsManager. When an event occurs in the StatisticsManager, the event is returned through the EventOccurred event handler from the StatisticsManager as a StatisticsEvent object. You can then check the ObjectType property to determine the type of the underlying object.

Enumeration	Value
`StatisticsEventObjectType_AggregateCumulative`	5401
`StatisticsEventObjectType_AggregateRealtime`	5400
`StatisticsEventObjectType_AggregateWrapupReasonCumulative`	5702
`StatisticsEventObjectType_Base` You have a StatisticsEvent that does not correspond to a valid object. This should never be seen.	0
`StatisticsEventObjectType_CallbackRealtime`	5600
`StatisticsEventObjectType_ContactRealtime`	5500
`StatisticsEventObjectType_GroupCumulative`	5201
`StatisticsEventObjectType_GroupRealtime`	5200
`StatisticsEventObjectType_ManagerStateChanged` You have a StatisticsEvent that corresponds to a ManagerStateChanged object. This is returned when there is a change in the Manager state (when you have initialized the StatisticsManager with events). See Section 4.5, “ManagerStateChangedEvent”, on page 135.	9899
`StatisticsEventObjectType_QueueCumulative`	5301
`StatisticsEventObjectType_QueueRealtime`	5300
`StatisticsEventObjectType_QueueWrapupReasonCumulative`	5701
`StatisticsEventObjectType_UserCumulative`	5101
`StatisticsEventObjectType_UserRealtime`	5100
`StatisticsEventObjectType_UserWrapupReasonCumulative`	5700

Table 47 enStatisticsEventObjectTypes enumeration

3.48 enStatisticsEventTypes

The enStatisticsEventTypes enumeration represents the various types of event groups that can be listened for from the StatisticsManager. For specific events, see Section 3.46, “enStatisticsEventCodes”, on page 111.

Enumeration	Value
StatisticsEventType_AggregateCumulativeEvent	5401
StatisticsEventType_AggregateRealtimeEvent	5400
StatisticsEventType_CallbackRealtimeEvent	5600
StatisticsEventType_ContactRealtimeEvent	5500
StatisticsEventType_GroupCumulativeEvent	5201
StatisticsEventType_GroupRealtimeEvent	5200
StatisticsEventType_NotSet The event that is received does not correspond to an event that has been explicitly listened for, for example a ManagerStateChanged event.	5000
StatisticsEventType_QueueCumulativeEvent	5301
StatisticsEventType_QueueRealtimeEvent	5300
StatisticsEventType_UserCumulativeEvent	5101
StatisticsEventType_UserRealtimeEvent	5100

Table 48 enStatisticsEventTypes enumeration

3.49 enStatisticsFunctionalities

The enStatisticsFunctionalities enumeration represents the various function groups that exist in the StatisticsManager. These are used when asking the StatisticsManager what functionality is supported. It might be worth to check if certain functionality is available before allowing certain actions in your application. For example, if you know that the Real-Time Server is unavailable, you may want to prevent people from attempting to query Agent Details.

Enumeration	Value
StatisticsFunctionality_Historical This indicates whether the StatisticsManager has connectivity to the Statistics Server.	2

Table 49 enStatisticsFunctionalities enumeration (Seite 1 von 2)

Enumeration	Value
`StatisticsFunctionality_Network` This indicates whether the StatisticsManager has connectivity to the Routing Server.	3
`StatisticsFunctionality_RealTime` This indicates whether the StatisticsManager has connectivity to the Real-Time Server. If the StatisticsManager does not have connectivity with the Real-Time Server, then Real Time information (for example, Agent Details) will be unavailable.	1

Table 49 enStatisticsFunctionalities enumeration (Seite 2 von 2)

3.50 enUserTypes

The enUserTypes enumeration represents the various types of users who can be defined in the database. Different user types have different authorizations in the system and can perform different actions based on their authorizations.

Enumeration	Value
`UserType_Administrator` An Administrator user type.	3
`UserType_AssociateA` Associate A user type is for agents who frequently handle contacts but are not high-volume contact center agents. These users are not measured on how efficiently they handle contacts.	8
`UserType_AssociateE` Associate E user type is for senior agents. These agents have the ability to monitor other agents and monitor contact center performance.	9
`UserType_Basic` The Basic user type. This is usually for a non-telephony user.	7
`UserType_CRM` This is the CRM User type.	10
`UserType_Master` The Master Administrator user type. There can only be one master administrator for a site.	4
`UserType_Service` The Service user type. This is the system user that is used by support in the event of a database problem.	5
`UserType_Supervisor` A Supervisor user type.	2
`UserType_Unknown` An Unknown Party Type. This should never be seen.	0

Table 50 enUserTypes enumeration

3.51 enWebCollaborationTemplateTypes

The enWebCollaborationTemplateTypes enumeration

Enumeration	Value
WebCollaborationTemplateType_StandardMessage	0
WebCollaborationTemplateType_Unknown	99
WebCollaborationTemplateType_URL	1

Table 51 enWebCollaborationTemplateTypes enumeration

Enumerations

enWebCollaborationTemplateTypes

4 Common Interfaces

This section describes the common interfaces provided by the SDK. Although each Manager object uses specific objects, there are some that are shared between Managers, such as common events or common data elements.

4.1 ContactData

As part of the call, various pieces of information are kept. This can be information that is entered in a voice processing system (for example, Call Director or an IVR), or information entered by users from a desktop application. This information is kept with the call and can be used by applications for various purposes such as display or screen pops. This information is stored in the form of KeyValuePairs. For more information, see Section 4.4, “KeyValuePair”, on page 132.

Interfaces supported:

- IContactData (default)
- IXMLAccess

4.1.1 Properties

This section contains properties exposed through the IContactData interface.For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

4.1.1.1 _NewEnum

For Each oKeyValuePair in colContactData

...

Next

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following:

```
Dim oKeyValuePair as HiPathProCenterLibrary.KeyValuePair
For Each oKeyValuePair in colContactData
   If oKeyValuePair.Type = KeyValuePairType_String Then
   txtTextBox.Text = txtTextBox.Text &
   oKeyValuePair.Key & " " &
   oKeyValuePair.Value & vbCrLf
   End If
Next
```

This sets the Text property of the txtTextBox to contain all key and the value for each of the KeyValuePair objects (that are of the string type) in our Contact Data collection.

Property:	_NewEnum (of ContactData)
Description:	Allows for the 'For Each' operator on our ContactData collection.
Parameters:	None
Type:	objKeyValuePair — For each item in the ContactData collection until all items have been returned (In C++ this returns an iterator object, consistent with parameters passed in).
Potential Errors:	None
Notes:	This property allows the ContactData collection to support the 'For Each' iterator. In Visual Basic, this property is hidden.

4.1.1.2 Count

The Count property returns the number of KeyValuePair items that are in the ContactData collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colContactData.Count
```

Property:	Count (of ContactData)
Description:	The number of KeyValuePair items that are in the ContactData collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

4.1.2 Methods

This section contains methods exposed through the ContactData interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

4.1.2.1 Add

```
Add ( KeyValuePair as KeyValuePair )
```

The Add method adds items to the ContactData collection.

This method accepts the following parameters:

- KeyValuePair as KeyValuePair

This is the KeyValuePair object that you want to add to the ContactData collection. For more information, see Section 4.4, "KeyValuePair", on page 132.

This method does not return anything.

To use this method, do the following:

```
Call colContactData.Add ( oKeyValuePair )
```

This adds the oKeyValuePair KeyValuePair object into the ContactData collection.

Method:	Add (of ContactData)
Description:	Adds a KeyValuePair to the ContactData collection.
Parameters:	KeyValuePair — The KeyValuePair object that you want to add to the ContactData collection.
Returns:	None
Potential Errors:	Error_Generic_ContactDataKeyValuePairReadOnly Error_Generic_ContactDataAddFailed Error_Generic_ContactDataAddFailedDuplicateKey
Notes:	None

4.1.2.2 Item

```
oKeyValuePair = Item (Key as String)
```

The Item method finds an item in the ContactData collection that is indexed by the strKey value. If an item is found in the collection at that key, the Item method returns the KeyValuePair object that is stored there. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

- Key as String

This is the string key of the item in the collection. For example, if you were searching for an account number, you can put the "Account #: " key.

This method returns the following parameters:

- KeyValuePair as String

This is the KeyValuePair object that is stored at the strKey key in the ContactData collection.

To use this method, do the following:

```
Dim oKeyValuePair as HiPathProCenterLibrary.KeyValuePair
strValue = colContactData.Item ("Account #: ")
```

This retrieves the KeyValuePair object from the ContactData collection that is stored at the "Account #: " key.

Property:	Item (of ContactData)
Description:	Queries the item stored at the string Key value from our ContactData collection.
Parameters:	Key — The collection key for the value you are querying.
Type:	KeyValuePair
Potential Errors:	Error_Generic_ItemNotFound Error_Generic_ContactDataItemFailed
Notes:	If the item does not exist in the collection, an error is raised.

4.1.2.3 Remove

```
Remove (Key as String)
```

The Remove method removes an item from our ContactData collection that is indexed by the strKey value. If an item is found in the collection at that key, the Remove method will remove the KeyValuePair from our ContactData collection. If the item does not exist in the collection, then the Remove returns without raising an error.

This method accepts the following parameters:

- strKey as String

This is the string key of the item in the collection. For example, if you were searching for the account number, you can put the "Account #: " key.

This method does not return anything.

To use this method, do the following:

```
Call colContactData.Remove ("Account #: ")
```

This removes the KeyValuePair from our ContactData collection that is stored at the "Account #: " key.

Method:	Remove (of ContactData)
Description:	Removes the KeyValuePair item stored at the strKey value from our ContactData collection.
Parameters:	Key — The collection key for the value you are removing.
Returns:	None
Potential Errors:	Error_Generic_ContactDataKeyValuePairReadOnly
Notes:	If the item does not exist in the collection, no error is raised.

4.1.2.4 RemoveAll

```
RemoveAll()
```

The RemoveAll method removes all the items from our ContactData collection. If the collection is empty, this method returns without raising an error.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following:

```
Call colContactData.RemoveAll
```

This removes all the KeyValuePairs from our ContactData collection.

Method:	RemoveAll (of ContactData)
Description:	Removes all the KeyValuePair items stored in our ContactData collection.
Parameters:	None
Returns:	None
Potential Errors:	None
Notes:	If the collection is already empty, no error is raised.

4.2 EnumList

The EnumList object is a collection of the enumeration This method does not accept any parameters.. The enumeration This method does not accept any parameters. are considered of type short (in C++) or Integer (in VB).

Interfaces supported:

- EnumList (default)

4.2.1 Properties

This section contains properties exposed through the EnumList interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

4.2.1.1 _NewEnum

```
For Each sValue in colEnumList
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following:

```
Dim sValue as Integer
For Each sValue in colEnumList
txtTextBox.Text = txtTextBox.Text & _
   "Value: " & _
   sValue & _
vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the enumeration This method does not accept any parameters. in the EnumList collection.

Property:	_NewEnum (of EnumList)
Description:	Allows for the 'For Each' operator on the EnumList collection.
Returns:	Integer — For each item in the EnumList collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum
Notes:	This method allows the EnumList collection to support the 'For Each' iterator.

4.2.1.2 Count

The Count property returns the number of enumeration This method does not accept any parameters. that exist in the EnumList collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colEnumList.Count
```

Property:	Count (of EnumList)
Description:	The number of enumeraton This method does not accept any parameters. that are in the EnumList collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

4.2.1.3 Item

The Item method accepts a zero-based index as a parameter and returns the enumeration value associated with the index location. For example, if the collection has five enumeration This method does not accept any parameters., the zero-based index would be zero to four (0 to 4).

```
Dim sValue as Integer
sValue = Item (Index as Long)
```

This method accepts the following parameters:

- Index as Long

This is the zero based index which identifies the position of enumeration value that need to be accessed. This allows an array like access to the enumeration This method does not accept any parameters., stored in the EnumList collection.

This method returns the following parameters:

- pVal as Integer

This is the enumeration value stored at the Index position.

To use this method, do the following:

Example usage to iterate through the EnumList collection

```
Dim List as EnumList
Dim sValue as Integer
For index = 0 To (list.count () -1) Step 1
```

```
sValue = list.Item(index)
Next index
```

Property:	Item (of EnumList)
Description:	Finds an enumeration value in the EnumList collection that is indexed by the Index value.
Parameters:	Index — The zero-based index of the enumeration value you are looking for.
Returns:	pVal — The enumeration value stored at the Index position.
Potential Errors:	Error_Generic_XMLObjectReadOnly
Notes:	None

4.2.2 Methods

This section contains methods exposed through the EnumList interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

4.2.2.1 Add

```
Add (Value as Integer)
```

The Add method adds items to the EnumList collection. This method accepts the following parameters:

- Value as Integer

This is the enumeration value that you want to add to the EnumList collection.

This method does not return anything.

To use this method, do the following:

Example to create colEnumList a collection of enCallStates enumeration This method does not accept any parameters.:-

```
Call colEnumList.Add ( CallState_Deferred )
```

This adds the enCallStates enumeration value CallState_Deferred into the EnumList collection.

Method:	Add (of EnumList)
Description:	Adds an enumeration value to the EnumList collection.
Parameters:	Value — The enumeration value that you want to add to the EnumList collection.
Returns:	None

Potential Errors:	None
Notes:	None

4.2.2.2 Remove

```
Remove (Value as Integer)
```

The Remove method removes the enumeration value from the EnumList collection that is identified by the Value.

This method accepts the following parameters:

- Value as Integer

This is the enumeration value that needs to removed from the collection.

This method does not return anything.

To use this method, do the following:

```
Call colEnumList.Remove (CallState_Deferred)
```

This removes the enCallStates enumeration value CallState_Deferred from the EnumList collection.

Method:	Remove (of EnumList)
Description:	Removes the enumeration value stored in the EnumList collection.
Parameters:	Value — The enumeration value you are removing.
Returns:	None
Potential Errors:	None
Notes:	If the item does not exist in the collection, no error is raised.

4.2.2.3 RemoveAll

```
RemoveAll ()
```

The RemoveAll method removes all the items from our EnumList collection. If the collection is empty, this method returns without raising an error.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following:

```
Call colEnumList.RemoveAll
```

Method:	RemoveAll (of EnumList)
Description:	Removes all enumeration This method does not accept any parameters. stored in our EnumList collection.
Parameters:	None
Returns:	None
Potential Errors:	None
Notes:	If the collection is already empty, no error is raised.

4.3 KeyList

The KeyList object represents the list of keys for the selected queues to be queried. It is a read/write collection of integers (longs). This is used to pass the collection of keys that are set by the SDK client application to narrow down the query. This collection differs in each request depending of the type of statistics events queried, they could be Queue IDs or User Keys or something else.

Interfaces supported:

- IKeyList (default)

4.3.1 Properties

This section contains methods exposed through the KeyList interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

4.3.1.1 _NewEnum

```
For Each lKey in colKeyList
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed via the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic via the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim lKey as Long
For Each lKey in colKeyList
txtTextBox.Text = txtTextBox.Text & _
"Key: " & _
lKey & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all Keys in the collection.

Property:	_NewEnum (of KeyList)
Description:	Allows for the 'For Each' operator on the Aggregates collection.
Returns:	OAggregate — For each item in the Aggregates collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum
Notes:	This method allows the Aggregates collection to support the 'For Each' iterator.

4.3.1.2 Count

The Count property returns the number of Keys that exist in the KeyList collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colKeyList.Count
```

Property:	Count (of KeyList)
Description:	The number of Keys that exist in the KeyList collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

4.3.2 Methods

The following methods are exposed through the KeyList interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

4.3.2.1 Add

```
Add (Key as Long)
```

The Add method adds keys to the KeyList collection.

This method accepts the following parameters:

Key as Long	This is the Key that you want to add to the KeyList collection.

This method does not return anything.

To use this method, do the following:

```
Call colKeyList.Add (lKey)
```

This adds the Key lKey into the KeyList collection.

Method:	Add (of KeyList)
Description:	Adds a Key to the KeyList collection.
Parameters:	Key — The Key that you want to add to the KeyList collection.
Returns:	None
Potential Errors:	Error_Generic_ItemAlreadyExists
Notes:	None

4.3.2.2 Item

```
lKey = Item (Index as Long)
```

The Item method finds the Key item in the KeyList collection that is indexed by the Index value. If an item is found in the collection at the Index value, the Item method returns the Key. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

Index as Long	This is the Index for the Key you are requesting.

This method returns the following parameters:

Key as Long	This is the Key specified by the Index value.

To use this method, do the following:

```
Dim lKey As Long
lKey = colKeyList.Item (Index)
```

This checks to see if the Key specified by the Index exists. If it does, lKey will contain the Key. If it does not exist in the KeyList collection, this will raise an error.

Property:	Item (of KeyList)
Description:	Finds the Key in the KeyList collection that is indexed by the Index value.
Parameters:	Index — The Index for the Key that you are looking for. This is an index starting from one.
Returns:	Key — The Key that you have requested.
Potential Errors:	Error_Generic_ItemNotFound
Notes:	If the Key does not exist in our collection, you raise an error.

4.3.2.3 Remove

The Remove method removes the specified Key from the KeyList collection. If the Key is found in the collection, the Remove method will remove that Key from our KeyList collection. If the item does not exist in the collection, then the Remove returns after raising an error.

This method accepts the following parameters:

Key as Long	This is the key that you want to remove from the collection.

This method does not return anything.

To use this method, do the following:

```
Call colKeyList.RemoveAt (Key)
```

This removes the Key from our KeyList collection.

Method:	Remove (of KeyList)
Description:	Removes the Key from our KeyList collection.
Parameters:	Key — This is the Key you want to remove from the collection.
Returns:	None
Potential Errors:	Error_Generic_ItemNotFound
Notes:	If the item does not exist in the collection, an error will be raised.

4.3.2.4 RemoveAll

```
RemoveAll()
```

The RemoveAll method removes all the items from our KeyList collection. If the collection is empty, this method returns without raising an error.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following:

```
Call colKeyList.RemoveAll
```

This removes all the Keys from our KeyList collection.

Method:	RemoveAll (of KeyList)
Description:	Removes all the Keys stored in our KeyList collection.
Parameters:	None
Returns:	None
Potential Errors:	None
Notes:	If the collection is already empty, no error is raised.

4.3.2.5 RemoveAt

```
RemoveAt (Index as Long)
```

The RemoveAt method removes an item from our KeyList collection that is indexed by the Index value. If an item is found in the collection at that key, the RemoveAt method will remove the Key from our KeyList collection. If the item does not exist in the collection, then the RemoveAt returns after raising an error.

This method accepts the following parameters:

Index as Long	This is the index of the key that you want to remove from the collection.

This method does not return anything.

To use this method, do the following:

```
Call colKeyList.RemoveAt (Index)
```

This removes the Key index by Index value from our KeyList collection.

Method:	RemoveAt (of KeyList)
Description:	Removes the Key stored at the Index value from our KeyList collection.
Parameters:	Index — The collection Index for the value you are removing.
Returns:	None
Potential Errors:	Error_Generic_ItemNotFound
Notes:	If the item does not exist in the collection, an error will be raised.

4.4 KeyValuePair

For each item in the ContactData collection there is a KeyValuePair object. These items are returned when the user uses the 'For Each' iterator on the collection. These are the actual pieces of Contact Data that you pass along with the calls.

Interfaces supported:

- IKeyValuePair (default)
- IXMLAccess

4.4.1 Properties

This section contains properties exposed through the KeyValuePair interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

4.4.1.1 Key

The Key property returns the string key of a KeyValuePair item from the ContactData collection. The Key property is used as the Key for the KeyValuePair item in the Contact Data collection.

To use this property, do the following:

```
Dim strKey as String
strKey = oKeyValuePair.Key
```

For example, if you were storing a customer's account number in the ContactData collection, this could be a string such as "Account #: ".

Property:	Key (of KeyValuePair)
Description:	The key of the KeyValuePair item.
Type:	String
Potential Errors:	None
Notes:	None

4.4.1.2 Type

The Type property returns the type of a KeyValuePair item from the ContactData collection. This type will match one of the This method does not accept any parameters. in the enKeyValuePairTypes enumeration (for example, if the value in the Value field is a string or binary data).

To use this property, do the following:

```
Dim enType as HiPathProCenterLibrary.enKeyValuePairTypes
enType = oKeyValuePair.Type
```

For example, if you were storing a customer's account number in the ContactData collection, this could be a type such as KeyValuePairType_String.

Property:	Type (of KeyValuePair)
Description:	The type of the KeyValuePair item.
Type:	enKeyValuePairTypes
Potential Errors:	None
Notes:	Read-only

4.4.1.3 Value

The Value property returns the value of the KeyValuePair item. Of all the properties of the KeyValuePair item, this is probably the most informative, as this is the actual value that was stored in the ContactData collection.

To use this property, do the following:

```
Dim strValue as String
strValue = oKeyValuePair.Value
```

For example, if you were storing a customer's account number in the ContactData collection, this could be a string such as "123456".

Property:	Value (of KeyValuePair)
Description:	The value of the KeyValuePair item.
Type:	String
Potential Errors:	Error_Generic_VariantTypeNotSupported (on Set)
Notes:	None

4.4.1.4 Visible

The Visible property returns if the KeyValuePair item should be visible. This is a boolean flag that represents whether or not the value that is contained here should be displayed.

In the future, this is planned to be used to flag if the data in the KeyValuePair should be displayed through the GUIs. This is just used as a suggestion for the GUIs to determine if they should display the KeyValuePair value or not. Even if the Visible flag is true, the data can still be displayed. This is useful for such information as data for screen pops, or other information that is important for the call, but not necessarily something that the contact center user wants to see.

To use this property, do the following:

```
Dim strValue as String
strValue = oKeyValuePair.Value
```

For example, if you were storing a customer's account number in the ContactData collection, this could be a string such as "123456".

Property:	Value (of KeyValuePair)
Description:	The value of the KeyValuePair item.
Type:	String
Potential Errors:	Error_Generic_VariantTypeNotSupported (on Set)
Notes:	None

4.5 ManagerStateChangedEvent

The ManagerStateChangedEvent interface is sent back when there is a change in the state of one of the Managers. This object, that cannot be created, is sent back when the state of one of the servers that is used by the Manager changes state. This is returned from all the Managers in the SDK.

Interfaces supported:

- IManagerStateChangedEvent (default)
- IHiPathProCenterEvent
- IAdministrationEvent
- IMediaEvent
- IRoutingEvent
- IStatisticsEvent

4.5.1 Properties

This section contains properties exposed through the ManagerStateChangedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

4.5.1.1 State

The State property returns the state that the Manager has changed to. This property returns a value from the enManagerStates enumeration, representing the state that the Manager has changed to. For more information, see Section 3.28, “enManagerStates”, on page 87.

This property is read-only, and it is set internally in the various Managers before the event is returned through the EventOccurred event handler.

To use this property, do the following:

```
Dim enManagerState as HiPathProCenterLibrary.enManagerStates
```

```
enManagerState = oManagerStateChangedEvent.State
```

Property:	State (of ManagerStateChangedEvent)
Description:	The state that the Manager has changed to.
Type:	enManagerStates
Potential Errors:	None
Notes:	Returned from the various Manager objects in the EventOccurred event handler. Read-only

4.6 MediaTypes

The MediaTypes object is a collection of various media types. This is used by the User object to track the various media types that the user is eligible for.

Interfaces supported:

- IMediaTypes (default)
- IXMLAccess

4.6.1 Properties

This section contains properties exposed through the MediaTypes interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

4.6.1.1 _NewEnum

```
For Each enMediaType in colMediaTypes
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim enMediaType as HiPathProCenterLibrary.enMediaTypes
For Each enMediaType in colMediaTypes
   txtTextBox.Text = txtTextBox.Text & _
      "MediaType: " & _
      CStr(enMediaTypes) & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the media types that the user is eligible for.

Property:	_NewEnum (of MediaTypes)
Description:	Allows for the 'For Each' operator on the Media Types collection.
Returns:	enMediaType — For each item in the MediaType collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory
Notes:	This method allows the MediaTypes collection to support the 'For Each' iterator.

To allow for the iteration of collections in C# through the 'foreach' iterator for the user.mediatypes collection a cast to “Int16” or “short” must be used instead of “enMediaTypes” cast. For example:

```
IUsers objUsers = oAdminManager.QueryUsers();
string sMediaTypes = "";
foreach (User objUser in objUsers)
```

```
{
foreach (short enMediaType in objUser.MediaTypes)
{
sMediaTypes = sMediaTypes + "," + enMediaType.ToString()
}
}
```

4.6.1.2 Count

The Count property returns the number of media types in the MediaTypes list.

To use this property, do the following:

```
Dim lCount as Long
lCount = colMediaTypes.Count
```

Property:	Count (of MediaTypes)
Description:	The number of media types that are in the MediaTypes list.
Type:	Long
Potential Errors:	Error_Generic_BaseObjectDoesNotExist
Notes:	Read-only

4.6.1.3 Item

```
enValue = Item (Key as String)
```

The Item method finds an item in the MediaTypes collection that is indexed by the Key value. If an item is found in the collection at that key, the Item method returns an enMediaTypes value. If the item does not exist in the collection, then the Item method raises an error.

This method accepts the following parameters:

Key as enMediaTypes	This is a media type that you want to see if the user is associated with.

This method returns the following parameters:

enMediaType as enMediaTypes	This returns the value of Key if the user is associated with Key. If the user is not associated with Key, an error is raised.

To use this method, do the following:

```
Dim enMediaType as HiPathProCenterLibary.enMediaTypes
enMediaType = colMediaTypes.Item (MediaType_Voice)
```

This checks to see if the user is eligible for the MediaType_Voice media type. If the user is eligible, enMediaType will equal MediaType_Voice. If they are not eligible for the voice media type, this will raise an error.

Property:	Item (of MediaTypes)
Description:	Checks if a user is eligible for a media type.
Parameters:	Key — The media type that you want to see if the user is associated with.
Returns:	enMediaTypes
Potential Errors:	Error_Generic_ItemNotFound
Notes:	None.

4.7 Parameters

The Parameters object is a collection of key/value pairs associated with a capability. For more information, see Section 4.4, “KeyValuePair”, on page 132.

Interfaces supported:

- IParameters (default)
- IXMLAccess

4.7.1 Properties

This section contains properties exposed through the IParameters interface.For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

4.7.1.1 _NewEnum

For Each oKeyValuePair in colParameters

...

Next

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following:

```
Dim oKeyValuePair as HiPathProCenterLibrary.KeyValuePair
For Each oKeyValuePair in colParameters
   If oKeyValuePair.Type = KeyValuePairType_String Then
   txtTextBox.Text = txtTextBox.Text &
   oKeyValuePair.Key & " " &
   oKeyValuePair.Value & vbCrLf
   End If
Next
```

This sets the Text property of the txtTextBox to contain all key and the value for each of the KeyValuePair objects (that are of the string type) in the Parameters collection.

Property:	_NewEnum (of Parameters)
Description:	Allows for the 'For Each' operator on our Parameters collection.
Parameters:	None
Type:	objKeyValuePair — For each item in the Parameters collection until all items have been returned (In C++ this returns an iterator object, consistent with parameters passed in).
Potential Errors:	None
Notes:	This property allows the Parameters collection to support the 'For Each' iterator. In Visual Basic, this property is hidden.

4.7.1.2 Count

The Count property returns the number of KeyValuePair items that are in the Parameters collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colCParameters.Count
```

Property:	Count (of Parameters)
Description:	The number of KeyValuePair items that are in the Parameters collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

4.7.2 Methods

This section contains methods exposed through the Parameters interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

4.7.2.1 Add

```
Add ( KeyValuePair as KeyValuePair )
```

The Add method adds items to the Parameters collection.

This method accepts the following parameters:

- KeyValuePair as KeyValuePair

This is the KeyValuePair object that you want to add to the Parameters collection. For more information, see Section 4.4, "KeyValuePair", on page 132.

This method does not return anything.

To use this method, do the following:

```
Call colParameters.Add ( oKeyValuePair )
```

This adds the oKeyValuePair KeyValuePair object into the Parameters collection.

Method:	Add (of Parameters)
Description:	Adds a KeyValuePair to the Parameters collection.
Parameters:	KeyValuePair — The KeyValuePair object that you want to add to the Parameters collection.
Returns:	None
Potential Errors:	Error_Generic_ContactDataKeyValuePairReadOnly Error_Generic_ContactDataAddFailed Error_Generic_ContactDataAddFailedDuplicateKey
Notes:	None

4.7.2.2 Item

```
oKeyValuePair = Item (Key as String)
```

The Item method finds an item in the Parameters collection that is indexed by the strKey value. If an item is found in the collection at that key, the Item method returns the KeyValuePair object that is stored there. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

- Key as String

This is the string key of the item in the collection. For example, if you were searching for a number, you can put the "Number: 1234" key.

This method returns the following parameters:

- KeyValuePair as String

This is the KeyValuePair object that is stored at the strKey key in the Parameters collection.

To use this method, do the following:

```
Dim oKeyValuePair as HiPathProCenterLibrary.KeyValuePair
strValue = colParameters.Item ("Number: 1234”)
```

This retrieves the KeyValuePair object from the Parameters collection that is stored at the "Number: 1234" key.

Property:	Item (of Parameters)
Description:	Queries the item stored at the string Key value from our Parameters collection.
Parameters:	Key — The collection key for the value you are querying.
Type:	KeyValuePair
Potential Errors:	Error_Generic_ItemNotFound Error_Generic_ContactDataItemFailed
Notes:	If the item does not exist in the collection, an error is raised.

4.7.2.3 Remove

```
Remove (Key as String)
```

The Remove method removes an item from our Parameters collection that is indexed by the strKey value. If an item is found in the collection at that key, the Remove method will remove the KeyValuePair from our Parameters collection. If the item does not exist in the collection, then the Remove returns without raising an error.

This method accepts the following parameters:

- strKey as String

This is the string key of the item in the collection. For example, if you were searching for the number, you can put the "Number: 1234” key.

This method does not return anything.

To use this method, do the following:

```
Call colParameters.Remove ("Number: 1234”)
```

This removes the KeyValuePair from our Parameters collection that is stored at the "Number: "1234” key.

Method:	Remove (of Parameters)
Description:	Removes the KeyValuePair item stored at the strKey value from our Parameters collection.
Parameters:	Key — The collection key for the value you are removing.
Returns:	None
Potential Errors:	Error_Generic_ContactDataKeyValuePairReadOnly
Notes:	If the item does not exist in the collection, no error is raised.

4.7.2.4 RemoveAll

```
RemoveAll()
```

The RemoveAll method removes all the items from our Parameters collection. If the collection is empty, this method returns without raising an error.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following:

```
Call colParameters.RemoveAll
```

This removes all the KeyValuePairs from our Parameters collection.

Method:	RemoveAll (of Parameters)
Description:	Removes all the KeyValuePair items stored in our Parameters collection.
Parameters:	None
Returns:	None
Potential Errors:	None
Notes:	If the collection is already empty, no error is raised.

4.8 XMLAccess

Most of the objects used in the SDK support what we call Access Interfaces. These interfaces provide a secondary way to get the details of the object.

The second of these interfaces is the XMLAccess interface. This interface allows users to retrieve the information contained inside an object in an XML-formatted string form.

Interfaces supported:

- IXMLAccess (default)

4.8.1 Methods

The XMLAccess interface provides a means to retrieve an XML-formatted string that represents the object the interface is exposed through, as well as a way to update the writeable properties of the object.

For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

4.8.1.1 AsXML

```
strXML = AsXML()
```

The AsXML method returns the underlying object that the XMLAccess interface is exposed through as an XML-formatted string representing the XML representation of the object.

This method does not accept any parameters.

This method returns the following parameters:

- strXML as String

This is the XML-formatted string that represents the XML representation of the Object (for example, in the following sample, it would represent the data about the call). Inside an XML-formatted string, some XML tags or nodes have attribute "RefNode" defined. The value of attribute "RefNode" refers to the name of the sibling tag or node from which the value of this node should be determined.

To use this method, do the following:

```
Dim oXMLAccess as HiPathProCenterLibrary.XMLAccess
Dim strXML as String
Set oXMLAccess = oCall
strXML = oXMLAccess.AsXML
```

This accepts the data of the Call object and stores it in an XML-formatted string.

Method:	AsXML (of XMLAccess)
Description:	Converts an object to an XML-formatted string.
Parameters:	None
Returns:	strXML — an XML-formatted string representing the object that the interface is exposed through.
Potential Errors:	None
Notes:	None

4.8.1.2 FromXML

```
FromXML (XML as String)
```

The FromXML method takes an XML-formatted string and updates the underlying object through the AsXML interface.

This method takes the following parameters:

- strXML as String

This is the XML-formatted string that represents the object data contained in the object (for example, in the following sample, it would represent the data about the call).

This method does not return any parameters.

To use this method, do the following:

```
Dim oXMLAccess as HiPathProCenterLibrary.XMLAccess
Dim oXMLAccess2 as HiPathProCenterLibrary.XMLAccess
Dim strXML as String
Set oXMLAccess = oCall
Set oXMLAccess2 = oCall2
strXML = oXMLAccess.AsXML
Call oXMLAccess2.FromXML (strXML)
```

This accepts the data of the Call object and stores it in an XML-formatted string in the strXML string variable, then you will pass this data into the objXMLAccess2 XMLAccess interface, thereby updating oCall2.

Method:	FromXML (of XMLAccess)
Description:	Updates an object with an XML-formatted string.
Parameters:	XML — an XML-formatted string to update our object with.
Returns:	None
Potential Errors:	Error_Generic_XMLObjectReadOnly
Notes:	None

5 OpenScape Contact Center Manager

The OpenScape Contact Center Manager is a creatable object that provides access to the other Managers in the system and information about their status. This is accomplished through the following methods:

- HireAdministrationManager
- HireMediaManager
- HireRoutingManager
- HireStatisticsManager

5.1 HiPathProCenterManager object

The HiPathProCenterManager object represents the system as follows:

- Provides access to the Managers that supply the necessary functions for the contact center.
- Provides information about the status of the contact center.

Interfaces supported:

- IHiPathProCenterManager5 (default)

5.1.1 Properties

This section contains properties exposed through the HiPathProCenterManager interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

5.1.1.1 BusinessUnitName

The BusinessUnitName property returns the name of the business unit that the user is logged on to or that was passed as a parameter in the Initialize method.

To use this property, do the following:

```
Dim BusinessUnitName as String
BusinessUnitName = g_oHiPathProCenterManager.BusinessUnitName
```

Property:	BusinessUnitName (Set on HiPathProCenterManager)
Description:	Returns the name of the business unit that the user is logged on to or that was passed as a parameter in the Initialize method.
Type:	String
Potential Errors:	None
Notes:	Read-only

5.1.1.2 DiagnosticFileName

The DiagnosticFileName property sets and returns the value of the file to which the diagnostic information is written. Diagnostics information helps determine what is going on in an application, under the covers. This can track both information in the SDK, as well as information in applications that you have written. For more information, see Section 5.1.2.12, "WriteToDiagnosticFile", on page 165.

To use this property, do the following:

```
Dim strFileName as String
strFileName = g_oHiPathProCenterManager.DiagnosticFileName
```

Property:	DiagnosticFileName (of HiPathProCenterManager)
Description:	The name of the file to which diagnostic information is written.
Type:	String
Potential Errors:	None
Notes:	By default, the SDK diagnostic file name is TTKDiag.log.

5.1.1.3 DiagnosticsFilterEnabled

The DiagnosticFilterEnabled property determines if the various diagnostic filters are turned on or off. For diagnostics of a specific level to be written to a file, the appropriate diagnostic filter must be enabled. This is true for both SDK diagnostics and external applications.

To use this property, do the following:

```
Dim bFilterEnabled as Boolean

bFilterEnabled =
g_oHiPathProCenterManager.DiagnosticFilterEnabled
(DiagnosticFilter_SDK)
```

Property:	DiagnosticFilterEnabled (of HiPathProCenterManager)
Description:	Determines if a diagnostic filter is enabled or not.
Parameters:	enDiagnosticFilter — The level of diagnostics for which to determine if it is enabled or not.
Type:	Boolean
Potential Errors:	None
Notes:	By default, the SDK diagnostic filter (DiagnosticFilter_SDK) is turned on.

5.1.1.4 LocaleID

The LocaleID property sets and returns the value of the locale ID associated with the language used when a hard-coded localized string is required by the system. The LocaleID property is currently only used when handling e-mail messages.

To use this property, do the following:

```
Dim enLocaleID as HiPathProCenterLibrary.enLocaleIDs

enLocaleID = g_oHiPathProCenterManager.LocaleID
```

Property:	LocaleID (of HiPathProCenterManager)
Description:	The locale ID of the OpenScape Contact Center Manager.
Type:	enLocaleIDs
Potential Errors:	None
Notes:	This property sets and returns a value from the enLocaleIDs enumeration as described in Section 3.27, "enLocaleIDs", on page 86. By default, the locale ID is set to LocaleID_None.

5.1.1.5 LoggedOnUserKey

The LoggedOnUserKey property returns the key that is associated with the logged on user after the Logon method has been invoked.

To use this property, do the following:

```
Dim LoggedOnUserKey as Long
LoggedOnUserKey = g_oHiPathProCenterManager. LoggedOnUserKey
```

Property:	LoggedOnUserKey (Set on HiPathProCenterManager)
Description:	Returns the key that is associated with the logged on user after the Logon method has been invoked.
Type:	Long
Potential Errors:	None
Notes:	Read-only

5.1.1.6 State

To use this property, do the following:

```
Dim enState as HiPathProCenterLibrary.enStates
enState = g_oHiPathProCenterManager.State
```

Property:	State (of HiPathProCenterManager)
Description:	The state of the OpenScape Contact Center Manager (Available/Unavailable).
Type:	enManagerStates
Potential Errors:	Error_Generic_UnableToDetermineState
Notes:	Read-only This property returns a value from the enManagerStates enumerations as described in Section 3.28, "enManagerStates", on page 87. This value is read-only.

5.1.2 Methods

This section contains methods exposed through the HiPathProCenterManager interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

5.1.2.1 DisableDiagnosticFilter

```
DisableDiagnosticFilter (DiagnosticFilter as enDiagnosticFilter)
```

The SDK uses diagnostics to log the behavior of the underlying code in the SDK. This is useful for determining why applications are behaving in various ways. These diagnostics provide technical information about why an error has occurred. To turn off diagnostics after they have been turned on, you must turn off the various diagnostic filters.

The diagnostic infrastructure that is used by the SDK is also exposed for external applications to use to log information. If you want to turn off this logging but continue to leave it implemented in your application, you must turn off the DiagnosticFilter_Application filter.

This method accepts the following parameters:

DiagnosticFilter as enDiagnosticFilter	This is the Diagnostic Filter that is to be turned off. A Diagnostic Filter must be turned off if you do not want the diagnostics to be written to the text file. See Section 3.12, "enDiagnosticFilters", on page 54.

This method does not return anything.

To use this method, do the following after creating the HiPathProCenterManager:

```
Call g_oHiPathProCenterManager.DisableDiagnosticFilter
(DiagnosticFilter_SDK)
```

Method:	DisableDiagnosticFilter (of HiPathProCenterManager)
Description:	Turns off the various diagnostic filters in the SDK.
Parameters:	DiagnosticFilter — The diagnostic filter to be turned off.
Returns:	None
Potential Errors:	None
Notes:	By default, the SDK diagnostic filter (DiagnosticFilter_SDK) is turned on.

5.1.2.2 EnableDiagnosticFilter

```
EnableDiagnosticFilter (DiagnosticFilter as enDiagnosticFilter)
```

The SDK uses diagnostics to log the behavior of the underlying code in the SDK. This is useful for determining why applications are behaving in various ways. These diagnostics provide technical information about why an error has occurred. To enable diagnostics, you must enable the various diagnostic filters.

The diagnostic infrastructure that is used by the SDK is also exposed for external applications to use to log information. To do this you must enable the DiagnosticFilter Application filter.

This method accepts the following parameters:

DiagnosticFilter as enDiagnosticFilter	This is the Diagnostic Filter that is to be enabled. A Diagnostic Filter must be turned on before the type of diagnostics will be written to the text file. See Section 3.12, "enDiagnosticFilters", on page 54.

This method does not return anything.

To use this method, do the following after creating the HiPathProCenterManager:

```
Call g_oHiPathProCenterManager.EnableDiagnosticFilter
(DiagnosticFilter_SDK)
```

Method:	EnableDiagnosticFilter (of HiPathProCenterManager)
Description:	Enables the various diagnostic filters in the SDK.
Parameters:	DiagnosticFilter — The diagnostic filter to be enabled.
Returns:	None
Potential Errors:	None
Notes:	By default, the SDK diagnostic filter (DiagnosticFilter_SDK) is turned on.

5.1.2.3 GetFunctionalityState

```
enManagerStates = GetFunctionalityState
(HiPathProCenterFunctionality as
enHiPathProCenterFunctionalities)
```

The HiPathProCenterManager may occasionally lose its connection to one or more servers in the system. When this occurs, you can check to see what functionality is supported. To do this, you can use the GetFunctionalityState method to pass in the functionality that you are interested in and the state of that functionality will be returned.

This method accepts the following parameters:

HiPathProCenter Functionality as enHiPathProCenter Functionalities	This is the functionality group that you want to enquire about. See Section 3.24, "enHiPathProCenterFunctionalities", on page 85.

This method returns the ManagerState of the functionality group you have requested. For more information, see Section 3.28, "enManagerStates", on page 87.

To use this method, do the following after creating the HiPathProCenterManager:

```
Dim enManagerState as HiPathProCenterLibrary.enManagerStates

enManagerState = g_oHiPathProCenterManager.GetFunctionalityState
(HiPathProCenterFunctionality_SystemStatus)
```

This checks with the HiPathProCenterManager to see if you have the ability to perform the system status functionality.

Method:	GetFunctionalityState (of HiPathProCenterManager)
Description:	Queries the state of various functionality groups in the HiPathProCenterManager.
Parameters:	HiPathProCenterFunctionalities — The functionality group you want to check the state of.
Returns:	enManagerStates — The state of the functionality group you requested.
Potential Errors:	Error_Generic_UnknownState
Notes:	None

5.1.2.4 HireAdministrationManager

```
AdministrationManager = HireAdministrationManager (EventMode as
enEventModes)
```

The HireAdministrationManager method returns the AdministrationManager object associated with the HiPathProCenterManager. This is the only means of retrieving an Administration Manager object. For more information, see Section 6.1, "AdministrationManager object", on page 175.

This method accepts the following parameters:

EventMode as HiPathProCenterLibrary.enEventModes	This is the event mode in which you want to start the AdministrationManager. These event modes work the same as the HiPathProCenterManager's event modes. The event mode is only used the first time the AdministrationManager is created. If you attempt to create the AdministrationManager with a different event mode, an error is returned to the caller. See Section 3.19, "enEventModes", on page 81.

This method returns the shared AdministrationManager object associated with the HiPathProCenterManager object. Subsequent calls to the HireAdministrationManager method returns the same AdministrationManager object that has already been created.

To use this method, do the following:

In global memory:

```
Public WithEvents g_oAdministrationManager as
HiPathProCenterLibrary.AdministrationManager
```

Somewhere else in your code after creating the HiPathProCenterManager:

```
Set g_oAdministrationManager =
g_oHiPathProCenterManager.HireAdministrationManager
(EventMode_FireEvents)
```

Method:	HireAdministrationManager (of HiPathProCenterManager)
Description:	Returns the AdministrationManager object associated with the HiPathProCenterManager.
Parameters:	EventModes — The event mode to start the AdministrationManager in (ignored after first call).
Returns:	AdministrationManager associated with this HiPathProCenterManager.

Potential Errors:	• Error_Generic_UnableToAllocateMemory • Error_HiPathProCenter_QueryingServerSummariesTimedOut • Error_HiPathProCenter_ConnectingToAdministrationServer • Error_HiPathProCenter_CouldNotFindConfigurationSynchronizationServerAddress • Error_HiPathProCenter_ConnectingToConfigurationSynchronizationServer • Error_Generic_InitializationTimedOut
Notes:	None

5.1.2.5 HireMediaManager

```
MediaManager = HireMediaManager (EventMode as enEventModes)
```

The HireMediaManager method returns the MediaManager object associated with the HiPathProCenterManager. This is the only means of retrieving a Media Manager object. For more information, see Section 7.1, "MediaManager object", on page 261.

This method accepts the following parameters:

EventMode as HiPathProCenterLibrary. enEventModes	This is the event mode in which you want to start the MediaManager. These event modes work the same as the HiPathProCenterManager's event modes. The event mode is only used the first time the MediaManager is created. If you attempt to create the MediaManager with a different event mode, an error is returned to the caller. See Section 3.19, "enEventModes", on page 81.

This method returns the shared MediaManager object associated with the HiPathProCenterManager object. Subsequent calls to the HireMediaManager method returns the same MediaManager object that has already been created.

To use this method, do the following:

In global memory:

```
Public WithEvents g_oMediaManager as
HiPathProCenterLibrary.MediaManager
```

Somewhere else in your code after creating the HiPathProCenterManager:

```
Set g_oMediaManager =
g_objHiPathProCenterManager.HireMediaManager
(EventMode_FireEvents)
```

Method:	HireMediaManager (of HiPathProCenterManager)
Description:	Returns the MediaManager object associated with the HiPathProCenterManager.

Parameters:	EventModes — The event mode to start the MediaManager in (ignored after first call).
Returns:	MediaManager associated with this HiPathProCenterManager.
Potential Errors:	• Error_HiPathProCenter_UserNotLoggedOn • Error_Generic_UnableToAllocateMemory • Error_HiPathProCenter_QueryingServerSummariesTimedOut • Error_HiPathProCenter_ConnectingToAdministrationServer • Error_HiPathProCenter_CouldNotFindTelephonyServerAddress • Error_HiPathProCenter_ConnectingToTelephonyServer • Error_HiPathProCenter_CouldNotFindCallbackServerAddress • Error_HiPathProCenter_ConnectingToCallbackServer • Error_HiPathProCenter_CouldNotFindEmailServerAddress • Error_HiPathProCenter_ConnectingToEmailServer • Error_HiPathProCenter_ConnectingToWebCollaborationServer • Error_HiPathProCenter_CouldNotFindRealTimeServerAddress • Error_HiPathProCenter_ConnectingToRealTimeServer • Error_Generic_InitializationTimedOut
Notes:	The MediaManager cannot be created until the Logon call has been successfully made.

5.1.2.6 HireRoutingManager

```
RoutingManager = HireRoutingManager (EventMode as enEventModes)
```

The HireRoutingManager method returns the RoutingManager object associated with the HiPathProCenterManager. This is the only means of retrieving a Routing Manager object. For more information, see Section 8.1, "RoutingManager object", on page 533.

This method accepts the following parameters:

EventMode as HiPathProCenterLibrary. enEventModes	This is the event mode in which you want to start the RoutingManager. These event modes work the same as the HiPathProCenterManager's event modes. The event mode is only used the first time the RoutingManager is created. If you attempt to create the RoutingManager with a different event mode, an error is returned to the caller. See Section 3.19, "enEventModes", on page 81.

This method returns the shared RoutingManager object associated with the HiPathProCenterManager object. Subsequent calls to the HireRoutingManager method returns the same RoutingManager object that has already been created.

To use this method, do the following:

In global memory:

```
Public WithEvents g_oRoutingManager as
HiPathProCenterLibrary.RoutingManager
```

Somewhere else in your code after creating the HiPathProCenterManager:

```
Set g_oRoutingManager =
g_oHiPathProCenterManager.HireRoutingManager
(EventMode_FireEvents)
```

Method:	HireRoutingManager (of HiPathProCenterManager)
Description:	Returns the RoutingManager object associated with the HiPathProCenterManager.
Parameters:	EventModes — The event mode to start the RoutingManager in (ignored after first call).
Returns:	RoutingManager associated with this HiPathProCenterManager.
Potential Errors:	• Error_HiPathProCenter_UserNotLoggedOn • Error_Generic_UnableToAllocateMemory • Error_HiPathProCenter_QueryingServerSummariesTimedOut • Error_HiPathProCenter_ConnectingToAdministrationServer • Error_HiPathProCenter_CouldNotFindRoutingServerAddress • Error_HiPathProCenter_ConnectingToRoutingServer • Error_HiPathProCenter_CouldNotFindNetworkRoutingServer Addres • Error_HiPathProCenter_ConnectingToNetworkRoutingServer • Error_Generic_InitializationTimedOut
Notes:	The RoutingManager cannot be created until the Logon call has been successfully made.

5.1.2.7 HireStatisticsManager

```
StatisticsManager = HireStatisticsManager (EventMode as
enEventModes)
```

The HireStatisticsManager method returns the StatisticsManager object associated with the HiPathProCenterManager. This is the only means of retrieving a StatisticsManager object. For more information, see Section 9.1, “StatisticsManager object”, on page 571.

This method accepts the following parameters:

EventMode as HiPathProCenterLibrary. enEventModes	This is the event mode in which you want to start the StatisticsManager. These event modes work the same as the HiPathProCenterManager's event modes. The event mode is only used the first time the StatisticsManager is created. If you attempt to create the StatisticsManager with a different event mode, an error is returned to the caller. See Section 3.19, “enEventModes”, on page 81.

This method returns the shared StatisticsManager object associated with the HiPathProCenterManager object. Subsequent calls to the HireStatisticsManager method returns the same StatisticsManager object that has already been created.

To use this method, do the following:

In global memory:

```
Public WithEvents g_oStatisticsManager as
HiPathProCenterLibrary.StatisticsManager
```

Somewhere else in your code after creating the HiPathProCenterManager:

```
Set g_oStatisticsManager =
g_oHiPathProCenterManager.HireStatisticsManager
(EventMode_FireEvents)
```

Method:	HireStatisticsManager (of HiPathProCenterManager)
Description:	Returns the StatisticsManager object associated with the HiPathProCenterManager.
Parameters:	EventModes — The event mode to start the StatisticsManager in (ignored after first call).
Returns:	StatisticsManager associated with this HiPathProCenterManager.
Potential Errors:	• Error_HiPathProCenter_UserNotLoggedOn • Error_Generic_UnableToAllocateMemory • Error_HiPathProCenter_QueryingServerSummariesTimedOut • Error_HiPathProCenter_ConnectingToAdministrationServer • Error_HiPathProCenter_CouldNotFindRealTimeServerAddress • Error_HiPathProCenter_ConnectingToRealTimeServer • Error_HiPathProCenter_CouldNotFindStatisticsServerAddress • Error_HiPathProCenter_ConnectingToStatisticsServer • Error_Generic_InitializationTimedOut
Notes:	The StatisticsManager cannot be created until the Logon call has been successfully made.

5.1.2.8 Initialize

NOTE: The Initialize method can be called only once for each SDK application.

```
Initialize (AdminServerAddress as String, EventMode as
enEventModes, (Optional) LocalPortNumber as Integer = 6100)
(Optional) WorkingThreads as Integer = 5) (Optional)
BusinessUnitName as String = "")
```

The Initialize method initializes the HiPathProCenterManager, and must be called prior to calling any other methods. This creates the various internal structures that the HiPathProCenterManager requires.

This method accepts the following parameters:

AdminServerAddress as String	This is the port number and TCP/IP address of the Administration Server to which you want to connect (for example, 6000@servername).
EventMode as HiPathProCenterLibrary. enEventModes	This is the event mode in which you want to start the HiPathProCenterManager. See Section 3.19, "enEventModes", on page 81.
LocalPortNumber as Integer	This is the local TCP/IP port number that will be used by the HiPathProCenterManager to communicate with the system. This must be unique for each application running on the same desktop machine.
WorkingThreads as Integer	This is the number of threads that will be used internally by the SDK to interact with the system. Please note that adjusting the default value to a much higher number could adversely affect the system on which the SDK application is running if the system does not have enough resources to handle that many threads.
BusinessUnitName as String	This is the name of the business unit in a multitenant environment. In a multitenant environment, this parameter is required. In a non-multitenant environment, this parameter is optional and is ignored.

This method does not return anything.

To use this method, do the following after creating the HiPathProCenterManager:

```
Call g_oHiPathProCenterManager.Initialize ("6000@servername",
EventMode_FireEvents)
```

This initializes the HiPathProCenterManager with the Administration Server running on the servername machine (on port 6000), fires events back as soon as they are received, and uses port 6100 (the default) on the local machine to communicate with the servers.

Method:	Initialize (of HiPathProCenterManager)
Description:	Initializes the HiPathProCenterManager.

Parameters:	AdminServerAddress — The address of the Administration Server to which you want to connect. EventModes — The event mode in which to start the HiPathProCenterManager. LocalPortNumber(6100) — The local port used to communicate with the servers. WorkingThreads (5) — The number of threads used internally by the SDK to interact with the system. BusinessUnitName (“”)— The name of the business unit in a multitenant environment.
Returns:	None
Potential Errors:	• Error_Generic_UnableToAllocateMemory • Error_HiPathProCenter_QueryingServerSummariesTimedOut • Error_HiPathProCenter_ConnectingToAdministrationServer • Error_HiPathProCenter_CouldNotFindWatchdogServerAddress • Error_HiPathProCenter_ConnectingToWatchdogServer • Error_Generic_InitializationTimedOut • Error_HiPathProCenter_InvalidBusinessUnitName
Notes:	Must be the first method called on the HiPathProCenterManager.

5.1.2.9 ListenForEvents

```
ListenForEvents (HiPathProCenterEventType as
enHiPathProCenterEventTypes, (Optional) Resource as String = "*"
)
```

The HiPathProCenterManager has a variety of events that you can request to listen for. To do this, you must make a call to ListenForEvents with the event type and the resource that you want to listen for the event on. The HiPathProCenterManager then notifies the servers that it wants to know when these events occur, and will wait for them. When they occur, the HiPathProCenterManager will receive the events and pass them back in the EventOccurred event handler. This function can only be called if the HiPathProCenterManager was started with events.

This method accepts the following parameters:

HiPathProCenterEventType as enHiPathProCenterEventTypes	This is the event type on which you want to listen for events. It's important to note that event types usually refer to more than one specific event in the HiPathProCenterManager. See Section 3.23, "enHiPathProCenterEventTypes", on page 84.
Resource as String	This is the resource on which you want to listen for the event. The resource is specific for the event type being listened for. For a list of what the resource represents for the various HiPathProCenterEventTypes, see Section 3.23, "enHiPathProCenterEventTypes", on page 84.

This method does not return anything.

To use this method, do the following after creating the HiPathProCenterManager:

```
Call g_oHiPathProCenterManager.ListenForEvents
(HiPathProCenterEventType_ServerStateChangedEvents)
```

or

```
Call g_oHiPathProCenterManager.ListenForEvents
(HiPathProCenterEventType_ServerStateChangedEvents, "*")
```

Although both of the preceding examples will do the same thing, the difference between the two is that in one you explicitly state the resource you want to listen for the event on. The resource parameter for this is optional, and if you do not pass it in, it will default to '*' (or all events). For more information, see Section 2.8, "About events", on page 37.

Method:	ListenForEvents (of HiPathProCenterManager)
Description:	Listens for events from the HiPathProCenterManager.
Parameters:	HiPathProCenterEventType — The event group you want to listen for. Resource("*") — The resource that you want to listen for the events on.
Returns:	None
Potential Errors:	Error_Generic_ListeningForEventsWhenIgnoringEvents Error_Generic_UnknownEventType
Notes:	This function can only be called if the HiPathProCenterManager was created with events.

5.1.2.10 Logon

```
Logon (UserKey as Long, UserPassword as String)
```

The Logon method ensures that the user specified by the UserKey is a known user in the system. To determine this, the OpenScape Contact Center Manager queries the user to ensure that the Password that they have specified matches the password for this user.

This method accepts the following parameters:

UserKey as Long	This is the agent key of the user that you want to logon to the SDK with.
UserPassword as String	This is the password for the user key that was passed into the Logon function.

This method does not return anything.

To use this method, do the following after creating the HiPathProCenterManager:

```
Call g_oHiPathProCenterManager.Logon (123, "password")
```

Method:	Logon (of HiPathProCenterManager)
Description:	Ensures that the user specified is a known user in the system.
Parameters:	UserKey — The agent key of the user that you want to logon. UserPassword — The password for this user.
Returns:	None
Potential Errors:	• Error_HiPathProCenter_CouldNotQueryUser • Error_HiPathProCenter_IncorrectPassword • Error_HiPathProCenter_MismatchBUKeyLogonInfoReq • Error_HiPathProCenter_UserKey_Password_Empty
Notes:	Must be called before hiring any Manager except for the Administration Manager. The Administration Manager can be created before calling Logon, so the users list can be retrieved.

5.1.2.11 StopListeningForEvents

```
StopListeningForEvents (HiPathProCenterEventType as
enHiPathProCenterEventTypes,(Optional) Resource as String = "*")
```

The HiPathProCenterManager has a variety of events that you can request to listen for There may come a time though when you no longer want to listen for these events. To stop listening for these events, you must call the StopListeningForEvents method. To do this, you must make a call to StopListeningForEvents with the event type and the resource you previously listened for the event on. You must use the exact same resource that you used when calling ListenForEvents. The HiPathProCenterManager will then tell the servers that it no longer wants to know when these events occur, and is no longer listening for them. This function can only be called if the HiPathProCenterManager was started without ignoring events.

This method accepts the following parameters:

HiPathProCenterEventType as enHiPathProCenterEventTypes	This is the event type that you want to stop listening for events on. It's important to note that event types usually refer to more than one specific event in the HiPathProCenterManager. See Section 3.23, "enHiPathProCenterEventTypes", on page 84.
Resource as String	This is the resource that you want to stop listening for the event on. The resource is specific for the event type you are stopping listening for. This must match the exact resource that you used when you issued the ListenForEvent request. For a list of what the resource represents for the various HiPathProCenterEventTypes, see Section 3.23, "enHiPathProCenterEventTypes", on page 84.

This method does not return anything.

To use this method, do the following after creating the HiPathProCenterManager:

```
Call
g_oHiPathProCenterManager.StopListeningForEvents(HiPathProCenter
EventType_ServerStateChangedEvents)
```

or

```
Call
g_oHiPathProCenterManager.StopListeningForEvents(HiPathProCenter
EventType_ServerStateChangedEvents, "*")
```

Although both of the preceding examples will do the same thing, the difference between the two is that in one you explicitly state the resource you want to stop listening for the event on. The resource parameter for this is optional, and if you do not pass it in, it will default to '*' (or all events). It is important that you use the same resource that you used when you call ListenForEvents. For example, if you registered for the event with '1234' you must use that to deregister the

event. Although '*' can be used for all events when you listen and stop listening for events, '*' only refers to stop listening for events that you listened for with '*'. For more information, see Section 2.8, "About events", on page 37.

Method:	StopListeningForEvents (of HiPathProCenterManager)
Description:	Stops listening for events from the HiPathProCenterManager.
Parameters:	HiPathProCenterEventType — The event group you want to stop listening for events for. Resource("*") — The resource that you want to listen for the events on.
Returns:	None
Potential Errors:	• Error_Generic_StopListeningForEventsWhenIgnoringEvents • Error_Generic_UnknownEventType
Notes:	This function can only be called if the HiPathProCenterManager was created with events. The resource must be the exact same as the resource you used when you called ListenForEvents.

5.1.2.12 WriteToDiagnosticFile

```
WriteToDiagnosticFile (Message as String,(Optional) Number as
Long = 0,(Optional) Source as String = "")
```

The WriteToDiagnosticFile method is exposed to provide external applications with the ability to log diagnostic information to a text file. To do this, you must enable the DiagnosticFilter_Application, and then call the WriteToDiagnosticFile method, passing along the message that is to be logged. Optionally, a number (for example, an error number or line number) can be passed, as well as a source of the error (for example, a file name or function name).

This method accepts the following parameters:

Message as String	This is the message that will be logged to a file.
Number as Long	This is the number that will be logged to a file. This is usually the error number or the line number from which the diagnostic is logged.
Source as String	This is the source that will be logged to a file. This is usually the file name or the function name from which the diagnostic is logged.

This method does not return anything.

To use this method, do the following after creating the HiPathProCenterManager:

```
Call g_oHiPathProCenterManager.WriteToDiagnosticFile ("Starting
Application with Parms: x: " & iX, 0,"Sub Main")
```

or

```
Call g_oHiPathProCenterManager.WriteToDiagnosticFile("Error
Occurred: " & Err.Description,Err.Number, "cmdStart_Click")
```

Method:	WriteToDiagnosticFile (of HiPathProCenterManager)
Description:	Logs diagnostic messages to the diagnostic text file.
Parameters:	• Message — The message to log to the diagnostic text file (ex: error information or parameters). • Number(0) — The number to log to the diagnostic text file (ex: error number or line number) • Source("") — The source to log to the diagnostic text file (ex: function name or file name)
Returns:	None
Potential Errors:	None
Notes:	None

5.1.3 Events

The following events are exposed through the HiPathProCenterManager interface.

For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

5.1.3.1 EventOccurred

```
g_oHiPathProCenterManager_EventOccurred (HiPathProCenterEvent as
HiPathProCenterEvent)
```

The EventOccurred event handler returns all events that occur in the HiPathProCenterManager object. All events returned through this event interface support the HiPathProCenterEvent interface (see Section 5.2, “HiPathProCenterEvent”, on page 167).

This method returns the following parameters:

HiPathProCenterEvent as HiPathProCenterLibrary. HiPathProCenterEvent	This is the event that has occurred in the HiPathProCenterManager. This returns the interface pointer to the base HiPathProCenterEvent interface. You can accept this object and get the full object interface pointer from it as follows.

To use this method, do the following after creating the HiPathProCenterManager:

```
Private Sub g_oHiPathProCenterManager_EventOccurred
(HiPathProCenterEvent as HiPathProCenterEvent)

dim oServerStateChanged as HiPathProCenterLibrary
.ServerStateChangedEvent

   if HiPathProCenterEvent.ObjectType =
   HiPathProCenterEventObjectType_ServerStateChanged Then

      ' Here you get the ServerStateChangedEvent interface

      ' from the HiPathProCenterEvent interface you received

      ' in the Event, and then check the server for the event.

      Set oServerStateChanged = pHiPathProCenterEvent

      MsgBox "ServerStateChanged for Server:" &
      oServerStateChanged.Server

   End If

   End Sub
```

Event Handler:	EventOccurred (of HiPathProCenterManager)
Description:	Returns all events that occur in the HiPathProCenterManager.
Parameters:	HiPathProCenterEvent — The HiPathProCenterEvent interface for the event that has occurred.
Returns:	None
Notes:	The event that is returned in this method can be 'recast' to other interfaces that are supported by the object. See Section 5.2, “HiPathProCenterEvent”, on page 167.

5.2 HiPathProCenterEvent

The HiPathProCenterEvent is an interface that contains a summary of the events that have occurred. All events sent back from the HiPathProCenterManager support the HiPathProCenterEvent interface. This non-creatable object also provides a common method of passing all events that occur in the HiPathProCenterManager to the applications using the SDK. This enables various pieces of information to be returned from the HiPathProCenterManager and allows for some simple handling to be performed before determining how (and if) you want to handle this event.

Applications can get specific information about the event by querying for other interfaces that are supported by the event interface. To determine what detailed object type the event returned, use the ObjectType property, and then ask for the appropriate interface.

Interfaces supported:

- IHiPathProCenterEvent (default)

For example:

```
Private Sub g_objHPPCManager_EventOccurred
   (ByVal pHiPathProCenterEvent as
   HiPathProCenterLibrary.HiPathProCenterEvent)
Select Case pHiPathProCenterEvent.ObjectType
   Case HiPathProCenterEventObjectType_ServerStateChanged
       Here since you have a ServerStateChanged event,
      ' you query for the ServerStateChanged interface.
      Dim objServerStateChanged as
      HiPathProCenterLibrary.ServerStateChangedEvent
      Set objServerStateChanged = pHiPathProCenterEvent
      MsgBox objServerStateChanged.Server &
       " has changed states."
   Case HiPathProCenterEventObjectType_ManagerStateChanged
      ' Here though you have a ManagerStateChanged event,
      ' so you query for the ManagerStateChanged interface.
      Dim objManagerStateChanged as
      HiPathProCenterLibrary.ManagerStateChangedEvent
      Set objManagerStateChanged = pHiPathProCenterEvent
      MsgBox "HiPathManager is now " &
       objManagerStateChanged.State
   Default
      MsgBox "Unknown ObjectType"
End Select
End Sub
```

5.2.1 Properties

This section contains properties exposed through the HiPathProCenterEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

5.2.1.1 Code

The Code property returns the event code of the event that has occurred in the HiPathProCenterManager. In most instances this will be directly related to the ObjectType that is also contained in the

HiPathProCenterEvent interface. This property determines what has happened. Since the ObjectType refers to the object that is returned, they will be related.

This property returns a value from the enHiPathProCenterEventCodes enumerations described in Section 3.21, "enHiPathProCenterEventCodes", on page 83. This property is read-only, and is set internally in the OpenScape Contact Center Manager before it fires the event back through the EventOccurred event handler. For more information, see Section 5.1.3.1, "EventOccurred", on page 166.

To use this property, do the following:

```
Dim enCode as HiPathProCenterLibrary.enHiPathProCenterEventCodes
enCode = oHiPathProCenterEvent.Code
```

Property:	Code (of HiPathProCenterEvent)
Description:	The code of what has occurred in the HiPathProCenterManager.
Type:	enHiPathProCenterEventCodes
Potential Errors:	None
Notes:	Read-only

5.2.1.2 EventType

The EventType property returns the event type that this event corresponds with. This event type is the event that was listened for this event to be sent. If the event is always going to be sent (Error or ManagerStateChanged event), then this property will be HiPathProCenterEventType_NotSet.

This property returns a value from the enHiPathProCenterEventTypes enumerations described in Section 3.23, "enHiPathProCenterEventTypes", on page 84. This property is read-only, and is set internally in the OpenScape Contact Center Manager before it fires the event back through the EventOccurred event handler. For more information, see Section 5.1.3.1, "EventOccurred", on page 166.

To use this property, do the following:

```
Dim enEventType as
HiPathProCenterLibrary.enHiPathProCenterEventTypes

enEventType = oHiPathProCenterEvent.EventType
```

Property:	EventType (of HiPathProCenterEvent)
Description:	The event type of the HiPathProCenterManager event that is sent back.
Type:	enHiPathProCenterEventTypes
Potential Errors:	None
Notes:	This determines what event group the object belongs to.

5.2.1.3 ObjectType

The ObjectType property returns the object interface type of the detailed event information contained in the HiPathProCenterEvent. In most instances this will be directly related to the Code that is also contained in the HiPathProCenterEvent interface. This property determines what the detailed interface is that is contained in the HiPathProCenterEvent.

This property returns a value from the enHiPathProCenterEventObjectTypes enumerations described in Section 3.22, "enHiPathProCenterEventObjectTypes", on page 84. This property is read-only, and is set internally in the OpenScape Contact Center Manager before it fires the event back through the EventOccurred event handler. For more information, see Section 5.1.3.1, "EventOccurred", on page 166.

To use this property, do the following:

```
Dim enObjectType as
HiPathProCenterLibrary.enHiPathProCenterEventObjectTypes

enObjectType = oHiPathProCenterEvent.ObjectType
```

Property:	ObjectType (of HiPathProCenterEvent)
Description:	The detailed object type of the HiPathProCenterManager event that is sent back.
Type:	enHiPathProCenterEventObjectTypes
Potential Errors:	None
Notes:	This determines what object type you can set the object to for detailed event information.

5.2.1.4 Resource

The Resource property returns the resource that this event corresponds with. The resource is a string value that is specific for an event. This is the resource that is used when registering for an event through the ListenForEvent method.

To use this property, do the following:

```
Dim strResource as String

strResource = oHiPathProCenterEvent.Resource
```

Property:	Resource (of HiPathProCenterEvent)
Description:	The resource that corresponds to the HiPathProCenterManager event that is sent back.
Type:	String

Potential Errors:	None
Notes:	This determines the resource the event is associated with.

5.3 ServerStateChangedEvent

The ServerStateChangedEvent interface is sent back when there is a change in the state of one of the servers that SDK has established a connection with. This non-creatable object represents a change in the state of the server from the client machines perspective.

For example, if the client loses connection to the server, it will report that it is attempting to reconnect to the server. This does not necessarily mean that the server is down, as the event could be sent back for a variety of reasons (for example, if the network cable of the client machine had been disconnected). This is returned through the EventOccurred event handler on the HiPathProCenterManager object.

NOTE: In order for SDK to establish a connection with a specific server, you need to Hire the SDK Manager that corresponds to that server.

Interfaces supported:

- ServerStateChangedEvent (default)
- IHiPathProCenterEvent
- IXMLAccess

5.3.1 Properties

This section contains properties exposed through the ServerStateChangedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

5.3.1.1 Server

The Server property returns a value from the enServers enumeration described in Section 3.44, "enServers", on page 108, representing the server that has changed state. This property is read-only, and is set internally in the HiPathProCenterManager before it fires the event back through the EventOccurred event handler. For more information, see Section 5.1.3.1, "EventOccurred", on page 166.

To use this property, do the following:

```
Dim enServer as HiPathProCenterLibrary.enServers
enServer = oServerStateChangedEvent.Server
```

Property:	Server (of ServerStateChangedEvent)
Description:	The server that has changed state in the system.
Type:	enServers
Potential Errors:	None
Notes:	Returned from the HiPathProCenterManager, EventOccurred event handler.

5.3.1.2 ServerState

The ServerState property returns a value from the enServerStates enumeration described in Section 3.45, "enServerStates", on page 109, representing the state that the server has changed to. This property is read-only, and is set internally in the HiPathProCenterManager before it fires the event back through the EventOccurred event handler. For more information, see Section 5.1.3.1, "EventOccurred", on page 166.

To use this property, do the following:

```
Dim enServerState as HiPathProCenterLibrary.enServerStates
enServerState = oServerStateChangedEvent.ServerState
```

Property:	ServerState (of ServerStateChangedEvent)
Description:	The state that the server in the system has changed to.
Type:	enServerStates
Potential Errors:	None
Notes:	None

6 Administration Manager

The Administration Manager is responsible for handling the administrative features for the system. This includes such things as updating the database, managing the configuration synchronization information from the Configuration Synchronization Server, and running and displaying simulation cases.

6.1 AdministrationManager object

The AdministrationManager is a non-creatable object that provides access to the various administrative functionalities in the system. The AdministrationManager is the access point to any administration and configuration information that is stored in the system.

Interfaces supported:

- IAdministrationManager5 (default)

6.1.1 Properties

This section contains properties exposed through the AdministrationManager interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.1.1.1 State

The State property returns the current state of the Administration Manager. This refers to the state that the AdministrationManager: available or unavailable.

This property returns a value from the enManagerStates enumerations described in Section 3.28, “enManagerStates”. This value is read-only.

To use this property, do the following:

```
Dim enState as HiPathProCenterLibrary.enStates

enState = g_oAdministrationManager.State
```

Property:	State (of AdministrationManager)
Description:	The state of the Administration Manager: available or unavailable.

Type:	enManagerStates
Potential Errors:	Error_Generic_UnableToDetermineState
Notes:	Read-only

6.1.2 Methods

This section contains methods exposed through the AdministrationManager interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

6.1.2.1 GetFunctionalityState

```
enManagerStates = GetFunctionalityState
(AdministrationFunctionality as enAdministrationFunctionalities)
```

The AdministrationManager may occasionally lose its connection to one or more servers in the system. When this occurs, you can check to see what functionality is supported using the GetFunctionalityState method. This method will pass in the functionality that you are interested in and the state of that functionality will be returned.

This method accepts the following parameter:

Administration Functionality as enAdministration Functionalities	This is the functionality group that you want to enquire about. See section Section 3.5, "enAdministrationFunctionalities", on page 48.

This method returns the ManagerState of the functionality group you have requested. For more information, see section Section 3.28, "enManagerStates", on page 87.

To use this method, do the following after creating the AdministrationManager:

```
Dim enManagerState as HiPathProCenterLibrary.enManagerStates
enManagerState = g_oAdministrationManager.GetFunctionalityState
   (AdministrationFunctionality_AdministrationDatabaseUpdates)
```

This checks with the AdministrationManager to see if you have the ability to receive configuration updates from the database.

Method:	GetFunctionalityState (of AdministrationManager)
Description:	Queries the state of various functionality groups in the AdministrationManager.
Parameters:	AdministrationFunctionality — The functionality group you want to check the state of.
Returns:	enManagerStates — The state of the functionality group you requested.
Potential Errors:	Error_Generic_UnableToDetermineState
Notes:	None

6.1.2.2 ListenForEvents

```
ListenForEvents (AdministrationEventType as
enAdministrationEventTypes, (Optional) Resource as String = "*")
```

The AdministrationManager has a variety of events that you can request to listen for. To do this, you must make a call to ListenForEvents with the event type and the resource you want to listen for the event on. The AdministrationManager will then tell the servers that it wants to know when these events occur, and will wait for them. When they occur, the AdministrationManager will receive the events and pass them back in the EventOccurred event handler.

This function can only be called if the AdministrationManager was created with events.

This method accepts the following parameters:

AdministrationEventType as enAdministrationEventTypes	This is the event type that you want to listen for events on. It's important to note that event types usually refer to more than one specific event in the AdministrationManager. See Section 3.4, “enAdministrationEventTypes”, on page 47.
Resource as String	This is the resource that you want to listen for the event on. The resource is specific for the event type being listened for. For a list of what the resource represents for the various AdministrationEventTypes, see Section 3.4, “enAdministrationEventTypes”, on page 47.

This method does not return anything.

To use this method, do the following after creating the AdministrationManager:

Example 1

```
Call g_oAdministrationManager.ListenForEvents
(AdministrationEventType_AdministrationDatabaseUpdates)
```

Example 2

```
Call g_oAdministrationManager.ListenForEvents
(AdministrationEventType_AdministrationDatabaseUpdates, "*")
```

Although both of the preceding examples will accomplish the same result, the difference between the two is that in Example 1, you explicitly state the resource you want to listen for the event on. The resource parameter for this is optional, and if you do not pass it in, it will default to '*' (or all events). For more information, see Section 2.8, “About events”, on page 37.

Method:	ListenForEvents (of AdministrationManager)
Description:	Listens for events from the AdministrationManager.
Parameters:	AdministrationEventType — The event group you want to listen for. Resource("*") — The resource that you want to listen for the events on.
Returns:	None
Potential Errors:	Error_Generic_ListeningForEventsWhenIgnoringEvents
Notes:	This function can only be called if the AdministrationManager was created with events.

6.1.2.3 QueryAggregates

The QueryAggregates method returns the collection of aggregates that are defined in the database. This method returns the Aggregates object, containing a collection of Aggregate objects that are defined in the database.

To use this method, do the following:

```
Dim oAggregates as HiPathProCenterLibrary.Aggregates
oAggregates = g_oAdministrationManager.QueryAggregates
```

Method:	QueryAggregates (of AdministrationManager)
Description:	Returns the collection of the Aggregates defined in the database.
Parameters:	None
Returns:	Aggregates — The Aggregates collection containing all the Aggregate objects defined in the database.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.4 QueryCallbackDeleteReasons

The QueryCallbackDeleteReasons method queries callback Delete reasons as a CallbackReasons collection.

This method does not accept any parameters.

This method returns a CallbackReasons collection object.

To use this method, do the following after creating the AdministrationManager object:

```
Dim oCallbackReasons as HiPathProCenterLibrary.CallbackReasons

Set oCallbackReasons =
oAdministrationManager.QueryCallbackDeleteReasons
```

Method:	QueryCallbackDeleteReasons (of AdministrationManager)
Description:	Queries callback Delete reasons.
Parameters:	None
Returns:	CallbackReasons — Callback Delete reasons queried from Administration Server.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.5 QueryCallbackRetryReasons

The QueryCallbackRetryReasons method queries callback Retry reasons as a CallbackReasons collection.

This method does not accept any parameters.

This method returns a CallbackReasons collection object.

To use this method, do the following after creating AdministrationManager object:

```
Dim oCallbackReasons as HiPathProCenterLibrary.CallbackReasons

Set oCallbackReasons =
oAdministrationManager.QueryCallbackRetryReasons
```

Method:	QueryCallbackRetryReasons (of AdministrationManager)
Description:	Queries callback Retry reasons.
Parameters:	None
Returns:	CallbackReasons — Callback Retry reasons queried from the Administration Server.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.6 QueryDepartments

The QueryDepartments method returns the collection of Department objects that are defined in the database. This method returns the Departments object, containing a collection of Department objects that are defined in the database.

To use this method, do the following:

```
Dim objDepartment as HiPathProCenterLibrary.Department
Dim objDepartments as HiPathProCenterLibrary.Departments
Set objDepartments = m_objAdmin.QueryDepartments
For Each objDepartment In objDepartments
...
Next
```

Method:	QueryDepartments (of AdministrationManager)
Description:	Returns the collection of the Departments defined in the database.
Parameters:	None
Returns:	Departments — The Departments collection containing all the Department objects defined in the database.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.7 QueryEmailDiscardReasons

The QueryEmailDiscardReasons method returns the collection of EmailDiscardReason objects that are defined in the database. This method returns the EmailDiscardReasons object, containing a collection of EmailDiscardReason objects that are defined in the database.

To use this method, do the following:

```
Dim objEmailDiscardReason as
HiPathProCenterLibrary.EmailDiscardReason
Dim objWrapupReasons as
HiPathProCenterLibrary.EmailDiscardReasons
Set objEmailDiscardReasons = m_objAdmin.QueryEmailDiscardReasons
For Each objEmailDiscardReason In objEmailDiscardReasons
...
```

```
Next
```

Method:	QueryEmailDiscardReasons (of AdministrationManager)
Description:	Returns the collection of the EmailDiscardReasons defined in the database.
Parameters:	None
Returns:	EmailDiscardReasons — The EmailDiscardReasons collection containing all the EmailDiscardReason objects defined in the database.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.8 QueryEmailTemplates

The QueryEmailTemplates method returns the collection of EmailTemplate objects that are defined in the database. This method returns the EmailTemplates object, containing a collection of EmailDiscard objects that are defined in the database.

To use this method, do the following:

```
Dim objEmailTemplate as HiPathProCenterLibrary.EmailTemplate
Dim objEmailTemplates as HiPathProCenterLibrary.EmailTemplates
Set objEmailTemplates =
m_objAdmin.QueryEmailTemplates(EmailTemplateType_Prolog,1)
For Each objEmailTemplate In objEmailTemplates
...
Next
```

Method:	QueryEmailTemplates (of AdministrationManager)
Description:	Returns the collection of the EmailTemplates defined in the database.
Parameters:	Type — Enumeration enEmailTemplateTypes, indicates the type of templates that will be returned by this query QueueKey — Queue key, zero (0) indicates all queues
Returns:	EmailTemplates — The EmailTemplates collection containing all the EmailTemplate objects defined in the database.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.9 QueryGroups

The QueryGroups method returns the collection of groups that are defined in the database. This method returns the Groups object, containing a collection of Group objects that are defined in the database.

To use this method, do the following:

```
Dim oGroups as HiPathProCenterLibrary.Groups
oGroups = g_oAdministrationManager.QueryGroups
```

Method:	QueryGroups (of AdministrationManager)
Description:	Returns the collection of the Groups defined in the database.
Parameters:	None
Returns:	Groups- the Groups collection containing all the Group objects defined in the database.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.10 QueryPostProcessingReasons

The QueryPostProcessingReasons method queries Post-processing reason codes and keys; it returns a read-only collection of PostProcessingReason objects which contains all Post-processing reasons defined in the system. This method does not accept any parameters.

To use this method, do the following after creating the AdministrationManager object:

```
Dim oPostProcessingReasons as
HiPathProCenterLibrary.PostProcessingReasons
Set oPostProcessingReasons =
oAdministrationManager.QueryPostProcessingReasons
```

Method:	QueryPostProcessingReasons (of AdministrationManager)
Description:	Queries Post-processing reason codes and keys.
Parameters:	None
Returns:	Post-processing reasons queried from Administration Server.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.11 QueryQueues

```
Queues = QueryQueues()
```

The QueryQueues method returns the collection of queues that are defined in the database. This method returns the Queues object, containing a collection of Queue objects that are defined in the database.

To use this method, do the following:

```
Dim oQueues as HiPathProCenterLibrary.Queues
oQueues = g_oAdministrationManager.QueryQueues
```

Method:	QueryQueues (of AdministrationManager)
Description:	Returns the collection of the queue defined in the database.
Parameters:	None
Returns:	Queues — The Queues collection containing all the Queue objects defined in the database.
Potential Errors:	Error_Generic_UnknownError Error_Generic_QueryFailed
Notes:	None

6.1.2.12 QueryUnavailableReasons

The QueryUnavailableReasons method queries Unavailable reason codes and keys; it returns a read-only collection of RoutingStateReason objects which contains all Unavailable reasons defined in the system.

This method does not accept any parameters.

To use this method, do the following after creating the AdministrationManager object:

```
Dim oRoutingStateReasons as
HiPathProCenterLibrary.RoutingStateReasons
Set oRoutingStateReasons =
oAdministrationManager.QueryUnavailableReasons
```

Method:	QueryUnavailableReasons (of AdministrationManager)
Description:	Queries Unavailable reason codes and keys.
Parameters:	None
Returns:	Unavailable reasons queried from Administration Server.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.13 QueryUsers

```
Users = QueryUsers()
```

The QueryUsers method returns the collection of users who are defined in the database. This method returns the Users object, containing a collection of User objects that are defined in the database.

To use this method, do the following:

```
Dim oUsers as HiPathProCenterLibrary.Users
oUsers = g_oAdministrationManager.QueryUsers
```

Method:	QueryUsers (of AdministrationManager)
Description:	Returns the collection of the users defined in the database.
Parameters:	None
Returns:	Users — The users collection containing all the User objects defined in the database.
Potential Errors:	Error_Generic_UnknownError Error_Generic_QueryFailed
Notes:	None

6.1.2.14 QueryWebCollaborationLanguages

The QueryWebCollaborationLanguages method returns the collection of Language objects that are defined in the database. This method returns the Languages object, containing a collection of Language objects that are defined in the database.

To use this method, do the following:

```
Dim objLanguage as HiPathProCenterLibrary.Language
Dim objLanguages as HiPathProCenterLibrary. Languages
Set objLanguages = m_objAdmin.QueryWebCollaborationLanguages
For Each objLanguage In objLanguages
...
Next
```

Method:	QueryWebCollaborationLanguages (of AdministrationManager)
Description:	Returns the collection of the Languages defined in the database.
Parameters:	None
Returns:	Languages — The Languages collection containing all the Language objects defined in the database.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.15 QueryWebCollaborationTemplates

The QueryWebCollaborationTemplates method returns the collection of WebCollaborationTemplate objects that are defined in the database. This method returns the WebCollaborationTemplates object, containing a collection of WebCollaborationTemplate objects that are defined in the database.

To use this method, do the following:

```
Dim objWebCollaborationTemplate as HiPathProCenterLibrary.
WebCollaborationTemplate

Dim objWebCollaborationTemplates as HiPathProCenterLibrary.
WebCollaborationTemplates

Set objWebCollaborationTemplates =
m_objAdmin.QueryWebCollaborationTemplates(WebCollaborationTempla
teType_StandardMessage,1)

For Each objWebCollaborationTemplate In
objWebCollaborationTemplates

...

Next
```

Method:	QueryWebCollaborationTemplates (of AdministrationManager)
Description:	Returns the collection of the WebCollaborationTemplates defined in the database.
Parameters:	Type — Enumeration enWebCollaborationTemplateTypes, indicates the type of templates that will be returned by this query QueueKey — Queue key, zero (0) indicates all queues
Returns:	WebCollaborationTemplates — The WebCollaborationTemplates collection containing all the WebCollaborationTemplate objects defined in the database.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.16 QueryWorkReasons

The QueryWorkReasons method queries Work reason codes and keys; it returns a read-only collection of RoutingStateReason objects which contains all Work reasons defined in the system. This method does not accept any parameters.

To use this method, do the following after creating the AdministrationManager object:

```
Dim oRoutingStateReasons as
HiPathProCenterLibrary.RoutingStateReasons

Set oRoutingStateReasons =
oAdministrationManager.QueryWorkReasons
```

Method:	QueryWorkReasons (of AdministrationManager)
Description:	Queries Work reason codes and keys.
Parameters:	None
Returns:	Work reasons queried from Administration Server.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.17 QueryWrapupReasons

The QueryWrapupReasons method returns the collection of WrapupReason objects that are defined in the database. This method returns the WrapupReasons object, containing a collection of WrapupReason objects that are defined in the database.

To use this method, do the following:

```
Dim objWrapupReason as HiPathProCenterLibrary.WrapupReason

Dim objWrapupReasons as HiPathProCenterLibrary.WrapupReasons

Set objWrapupReasons = m_objAdmin.QueryWrapupReasons

For Each objWrapupReason In objWrapupReasons

...

Next
```

Method:	QueryWrapupReasons (of AdministrationManager)
Description:	Returns the collection of the WrapupReasons defined in the database.
Parameters:	None

Returns:	WrapupReasons — The WrapupReasons collection containing all the WrapupReason objects defined in the database.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

6.1.2.18 StopListeningForEvents

```
StopListeningForEvents (AdministrationEventType as
enAdministrationEventTypes,(Optional) Resource as String = "*")
```

The AdministrationManager has a variety of events that you can request to listen for. There may come a time though when you no longer want to listen for these events. To stop listening for these events, call the StopListeningForEvents method with the event type and the resource you previously listened for the event on. You must use the exact same resource that you used when calling ListenForEvents. The AdministrationManager then notifies the servers that it no longer wants to know when these events occur, and is no longer listening for them.

This function can only be called if the AdministrationManager was created with events.

This method accepts the following parameters:

AdministrationEventType as enAdministrationEventTypes	This is the event type that you want to stop listening for events on. It's important to note that event types usually refer to more than one specific event in the AdministrationManager. See Section 3.4, “enAdministrationEventTypes”, on page 47.
Resource as String	This is the resource that you want to stop listening for the event on. The resource is specific for the event type you are stopping listening for. This must match the exact resource that you used when you issued the ListenForEvent request. For a list of what the resource represents for the various AdministrationEventTypes, see Section 3.4, “enAdministrationEventTypes”, on page 47.

This method does not return anything.

To use this method, do either of the following after creating the HiPathProCenterManager:

Example 1

```
Call g_oAdministrationManager.StopListeningForEvents
(AdministrationEventType_AdministrationDatabaseUpdates)
```

Example 2

```
Call g_oAdministrationManager.StopListeningForEvents
(AdministrationEventType_AdministrationDatabaseUpdates, "*")
```

Although both of the preceding examples will do the same thing, the difference between the two is that in example 1 you explicitly state the resource you want to stop listening for the event on. The resource parameter for this is optional, and if you do not pass it in, it will default to '*' (or all events). It is important that you use the same resource that you used when you call ListenForEvents. For example, if you registered for the event with '1234' you must use that to deregister for the event. Although '*' can be used for all events when you listen and stop listening for events, '*' only refers to stop listening for events that you listened for with '*'. For more information, see Section 2.8, "About events", on page 37.

Method:	StopListeningForEvents (of AdministrationManager)
Description:	Stops listening for events from the AdministrationManager.
Parameters:	AdministrationEventType — The event group you want to stop listening for events for. Resource("*") — The resource that you want to listen for the events on.
Returns:	None
Potential Errors:	Error_Generic_StopListeningForEventsWhenIgnoringEvents
Notes:	This function can only be called if the AdministrationManager was created with events. The resource must be the exact same as the resource you used when you called ListenForEvents.

6.1.3 Events

The following events are exposed through the AdministrationManager interface.

For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

6.1.3.1 EventOccurred

```
g_oAdministrationManager_EventOccurred (AdministrationEvent as
AdministrationEvent)
```

The EventOccurred event handler returns all events that occur in the AdministrationManager object. All events returned through this event interface support the AdministrationEvent interface. For more

information, see Section 6.2, "AdministrationEvent", on page 191.

This method returns the following parameter:

Administration Event as HiPathProCenter Library. Administration Event	This is the event that has occurred in the AdministrationManager. This returns the interface pointer to the base AdministrationEvent interface. You can take this object and get the full object interface pointer from it as follows.

To use this method, do the following after creating the AdministrationManager:

```
Private Sub g_oAdministrationManager_EventOccurred
   (AdministrationEvent as AdministrationEvent)
   dim oUser as HiPathProCenterLibrary.User
   if AdministrationEvent.ObjectType =
      AdministrationEventObjectType_User Then
      ' Here you get the User interface from the
      ' AdministrationEvent interface you received
      ' in the Event, and then check the first
      ' name property.
      Set oUser = AdministrationEvent
      MsgBox "First Name: " & oUser.FirstName
   End If
End Sub
```

Event Handler:	EventOccurred (of AdministrationManager)
Description:	Returns all events that occur in the AdministrationManager.
Parameters:	AdministrationEvent — The AdministrationEvent interface for the event that has occurred.
Returns:	None
Notes:	The event that is returned in this method can be 'recast' to other interfaces that are supported by the object. See Section 6.2, "AdministrationEvent", on page 191.

6.2 AdministrationEvent

The AdministrationEvent is an interface that contains a summary of the event that has occurred. All events sent back from the AdministrationManager support the AdministrationEvent interface. This object, that cannot be created, also provides a common method of passing all events that occur in the AdministrationManager to the applications using the SDK. This enables various pieces of information to be returned from the AdministrationManager and some simple handling to be performed before determining how (and if) you want to handle this event.

Applications can get specific information about the event by querying for other interfaces that are supported by the event interface. To determine what detailed object type the event returned, you can use the ObjectType property, and then ask for the appropriate interface.

Interfaces supported:

- AdministrationEvent (default)

For example:

```
Private Sub g_oAdministrationManager_EventOccurred
   (ByVal AdministrationEvent as
   HiPathProCenterLibrary.AdministrationEvent)
Select Case AdministrationEvent.ObjectType
   Case AdministrationEventObjectType_User
      ' Here since you have a User event,
      ' you query for the User interface.
      Dim oUser as HiPathProCenterLibrary.User
      Set oUser = AdministrationEvent
      MsgBox oUser.FirstName & " " & _
      oUser.LastName & " has been updated."
   Case AdministrationEventObjectType_ManagerStateChanged
      ' Here though you have a ManagerStateChanged event,
      ' so you query for the ManagerStateChanged interface.
      Dim oManagerStateChanged as
      HiPathProCenterLibrary.ManagerStateChangedEvent
      Set oManagerStateChanged = HiPathProCenterEvent
      MsgBox "Administration Manager is now " &
      oManagerStateChanged.State
   Default
      MsgBox "Unknown ObjectType"
End Select
```

```
End Sub
```

6.2.1 Properties

This section contains properties exposed through the AdministrationEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

6.2.1.1 Code

The Code property returns the event code of the event that has occurred in the AdministrationManager. In most instances this will be directly related to the ObjectType that is also contained in the AdministrationEvent interface. This property determines what has happened. Since the ObjectType refers to the object that is returned, they will be related.

This property returns a value from the enAdministrationEventCodes enumerations described in Section 3.2, "enAdministrationEventCodes", on page 44. This property is read-only, and is set internally in the AdministrationManager before it fires the event back through the EventOccurred event handler. For more information, see section Section 6.1.3.1, "EventOccurred", on page 189.

To use this property, do the following:

```
Dim enCode as HiPathProCenterLibrary.enAdministrationEventCodes
enCode = oAdministrationEvent.Code
```

Property:	Code (of AdministrationEvent)
Description:	The code of what has occurred in the AdministrationManager.
Type:	enAdministrationEventCodes
Potential Errors:	None
Notes:	Read-only

6.2.1.2 EventType

The EventType property returns the event type that this event corresponds with. This is the event that was listened for this event to be sent. If this event is always going to be sent (Error or ManagerStateChanged event), then this property will be AdministrationEventType_NotSet.

This property returns a value from the enAdministratonEventTypes enumerations described in Section 3.4, "enAdministrationEventTypes", on page 47. This property is read-only, and is set internally in the Administration Manager before it fires the event back through the EventOccurred event handler. For more information, see section Section 6.1.3.1, "EventOccurred", on page 189.

To use this property, do the following:

```
Dim enEventType as
HiPathProCenterLibrary.AdministrationEventTypes

enEventType = oAdministrationEvent.EventType
```

Property:	EventType (of AdministrationEvent)
Description:	The event type of the AdministrationEvent event that is sent back.
Type:	enAdministrationEventTypes
Potential Errors:	None
Notes:	This determines what event group the object belongs to.

6.2.1.3 ObjectType

The ObjectType property returns the object interface type of the detailed event information contained in the AdministrationEvent. In most instances this will be directly related to the Code that is also contained in the AdministrationEvent interface. This property determines what the detailed interface is that is contained in the AdministrationEvent.

This property returns a value from the enAdministrationEventObjectTypes enumerations described in Section 3.22, "enHiPathProCenterEventObjectTypes", on page 84. This property is read-only, and is set internally in the AdministrationManager before it fires the event back through the EventOccurred event handler. For more information, see Section 6.1.3.1, "EventOccurred", on page 189.

To use this property, do the following:

```
Dim enObjectType as
HiPathProCenterLibrary.enAdministrationEventObjectTypes

enObjectType = oAdministrationEvent.ObjectType
```

Property:	ObjectType (of AdministrationEvent)
Description:	The detailed object type of the AdministrationManager event that is sent back.
Type:	enAdministrationEventObjectTypes
Potential Errors:	None
Notes:	This determines what object type you can query for more detailed event information.

6.2.1.4 Resource

The Resource property returns the resource that this event corresponds with. The resource is a string value that is specific for an event. This is the resource that is used when registering for an event through the ListenForEvent method.

To use this property, do the following:

```
Dim strResource as String

strResource = oAdministrationEvent.Resource
```

Property:	Resource (of AdministrationEvent)
Description:	The resource that corresponds to the AdministrationManager event that is sent back.
Type:	String
Potential Errors:	None
Notes:	This determines the resource the event is associated with.

6.3 Aggregate

The Aggregate object provides an aggregate which is a logical grouping of queues defined in the system. Aggregates are useful for reporting purposes.

Interfaces supported:

- IAggregate (default)
- IAdministrationEvent
- IXMLAccess

6.3.1 Properties

This section contains properties exposed through the Aggregate interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.3.1.1 Description

The Description property contains the description of the Aggregate object.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oAggregate.Description
```

Property:	Description (of Aggregate)
Description:	Specifies the description for the Aggregate which is the logical grouping of voice queues used for reporting purposes.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.3.1.2 Key

The Key property is the key associated with the aggregate object in the database.

To use this property, do the following:

```
Dim lKey as Long
lKey = oAggregate.Key
```

Property:	Key (of Aggregate)
Description:	The key associated with the aggregate object in the database.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.3.1.3 MediaType

The MediaType property is the media type of the aggregate.

To use this property, do the following:

```
Dim enMediaType as HiPathProCenterLibrary.enMediaTypes
enMediaType = oAggregate.MediaType
```

Property:	MediaType (of Aggregate)
Description:	The media type of the Aggregate.
Type:	enMediaTypes
Potential Errors:	None
Notes:	Read-only

6.3.1.4 Name

The Name property is the name associated with the aggregate object in the database.

To use this property, do the following:

```
Dim strName as String
strName = oAggregate.Name
```

Property:	Name (of Aggregate)
Description:	The name of the Aggregate.

Type:	String
Potential Errors:	None
Notes:	Read-only

6.4 Aggregates

The Aggregates object represents a read-only collection of zero or more aggregate objects containing all the aggregates that are defined in the database.

Interfaces supported:

- IAggregates (default)
- IXMLAccess

6.4.1 Properties

This section contains properties exposed through the Aggregates interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.4.1.1 _NewEnum

```
For Each oAggregate in colAggregates
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oAggregate as HiPathProCenterLibrary.Aggregate
For Each oAggregate in colAggregates
txtTextBox.Text = txtTextBox.Text & _
"Aggregate Name: " & _
oAggregate.Name & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the names of all Aggregate.

Property:	_NewEnum (of Aggregates)
Description:	Allows for the 'For Each' operator on the Aggregates collection.
Returns:	OAggregate — For each item in the Aggregates collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemoryError_Generic_IterationGetNewEnum
Notes:	This method allows the Aggregates collection to support the 'For Each' iterator.

6.4.1.2 Count

The Count property returns the number of Aggregate objects that exist in the Aggregates collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colAggregates.Count
```

Property:	Count (of Aggregates)
Description:	The number of Aggregate objects that are in the Aggregates collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.4.1.3 Item

```
oAggregate = Item (Key as Long)
```

The Item method finds an Aggregate object in the Aggregates collection that is indexed by the Key value. If an item is found in the collection at the Aggregate key value, the Item method returns the Aggregate object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

Key as Long	This is the Aggregate key for the Aggregate you are requesting.

This method returns the following parameters:

oAggregate as Aggregate	This is the Aggregate object for Aggregate specified by the Key value. This contains the information for the Aggregate.

To use this method, do the following:

```
Dim oAggregate as HiPathProCenterLibary.Aggregate
Set oAggregate = colAggregates.Item (lKey)
```

This checks to see if the Aggregate specified by the Key exists. If it does, oAggregate will contain the Aggregate object. If it does not exist in the Aggregates collection, this will raise an error.

Property:	Item (of Aggregates)
Description:	Finds an Aggregate object in the Aggregates collection that is indexed by the Key key value.
Parameters:	Key — the key for the Aggregate that you are looking for. This is an index starting from one (1).
Returns:	oAggregate — The summary for the Aggregate that you have requested.
Potential Errors:	Error_Generic_CreatingObject Error_Generic_ItemNotFound
Notes:	If the Aggregate does not exist in your collection, you raise an error.

6.5 DeleteList

The DeleteList object is a collection of keys of a single type of object that have been deleted. This is to notify applications when one or more objects have been deleted from the database. The reason this is presented in the form of a list is so that, if there are multiple items that are deleted, they can be sent as a single update.

Interfaces supported:

- IDeleteList (default)
- IAdministrationEvent

6.5.1 Properties

This section contains properties exposed through the DeleteList interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.5.1.1 _NewEnum

```
For Each lKey in colDeleteList
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim lKey as String
For Each lKey in colDeleteList
   txtTextBox.Text = txtTextBox.Text & _
      "Key: " & lKey & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the keys of the objects that were deleted.

Property:	_NewEnum (of DeleteList)
Description:	Allows for the 'For Each' operator on the DeleteList collection.
Returns:	Key — For each item in the DeleteList collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum Error_Generic_BaseObjectDoesNotExist
Notes:	This method allows the DeleteList collection to support the 'For Each' iterator.

6.5.1.2 Count

The Count property returns the number of items in our DeleteList list.

To use this property, do the following:

```
Dim lCount as Long
lCount = colDeleteList.Count
```

Property:	Count (of DeleteList)
Description:	The number of items in our DeleteList list.
Type:	Long
Potential Errors:	Error_Generic_BaseObjectDoesNotExist
Notes:	Read-only

6.5.1.3 Item

```
lValue = Item (Key as Long)
```

The Item method finds an item in the DeleteList collection that is indexed by the Key value. If an item is found in the collection at the key, the Item method returns the key of the object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

Key as Long	This is the key for the object that has been deleted. If the key for the item you pass into the DeleteList collection is not found, then the item has not been deleted.

This method returns the following parameters:

Value as String	This is the key of the item that has been deleted. This is the same key that was passed in and does not really provide much more information, except that since the Item method didn't cause an error, you know that the item was deleted.

To use this method, do the following:

```
Dim lKey as Long
lValue = colDeleteList.Item (lKey)
```

This checks to see if the item specified by the lKey key has been deleted. If it has, lValue will contain the key of the object that has been deleted. If they do not have this key in our DeleteList collection, this will raise an error.

Method:	Item (of DeleteList)
Description:	Checks if an object has been deleted.
Parameters:	Key — The key of the object you want to use to check to see if it has been deleted.
Returns:	Value — The key of the object that you have deleted.
Potential Errors:	Error_Generic_BaseObjectDoesNotExist Error_Generic_ItemNotFound
Notes:	If the item specified by the key has not been deleted, raise an error.

6.5.1.4 ObjectType

The ObjectType property returns the type of objects that have been deleted from the database. This type corresponds to one of the AdministrationBase objects.

To use this property, do the following:

```
Dim enObjectType as
HiPathProCenterLibrary.enAdministrationBaseObjectTypes
enObjectType = colDeleteList.ObjectType
```

Property:	ObjectType (of DeleteList)
Description:	The type of objects that have been deleted from the database.
Type:	AdministrationBaseObjectTypes
Potential Errors:	None
Notes:	Read-only

6.6 Department

The Department objects are used in Departments collection in AdministrationManager::QueryDepartments() query.

Interfaces supported:

- IDepartment (default)
- IAdministrationEvent
- IXMLAccess

6.6.1 Properties

This section contains properties exposed through the Department interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.6.1.1 Description

The Description property returns the department description.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oDepartment.Description
```

Property:	Description (of Department)
Description:	This property returns the department description.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.6.1.2 Key

The Key property returns the department key.

To use this property, do the following:

```
Dim lKey as Long
lKey = oDepartment.Key
```

Property:	Key (of Department)
Description:	This property returns the department key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.6.1.3 Name

The Name property returns the department name.

To use this property, do the following:

```
Dim strName as String
strName = oDepartment.Name
```

Property:	Name (of Department)
Description:	This property returns the department name.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.7 Departments

The Department object represents a collection of one or more of Department objects.

Interfaces supported:

- IDepartments (default)
- IXMLAccess

6.7.1 Properties

This section contains properties exposed through the Departments interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.7.1.1 _NewEnum

```
For Each oDepartment in colDepartments
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oDepartment as HiPathProCenterLibrary.Department
For Each oDepartment in colDepartments
Next
```

Property:	_NewEnum (of Departments)
Description:	Provides access to elements of the Departments by means of operator FOREACH.
Returns:	oDepartment - for each item in the Departments collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum
Notes:	This method allows the Departments collection to support the 'For Each' iterator.

6.7.1.2 Count

The Count property returns the number of Department objects that exist in the Departments collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colDepartments.Count
```

Property:	Count (of Departments)
Description:	The number of Department objects in Departments object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.7.1.3 Item

```
oTimeRange = Item (departmentKey as Integer)
```

The Item method finds a Department object in the Departments collection that is indexed by the department key. If an item is found in the collection, the Item method returns the Department object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

departmentKey as Integer	This is the department key for the item you are requesting.

This method returns the following parameters:

oDepartment as IDepartment	This is the Department object for the department key value.

To use this method, do the following:

```
Dim oDepartment as HiPathProCenterLibary.Department
Set oDepartment = colDepartments.Item (departmentKey)
```

This checks to see if the Department specified by the departmentKey value exists. If it does, oDepartment will contain the Department object. If it does not exist in the Departments, this will raise an error.

Method:	Item (of Departments)
Description:	Finds a Department in the Departments collection that is indexed by the department key.
Parameters:	departmentKey — The department key for the Department that you are looking for.

Returns:	oDepartment — The Department object specified by the departmentKey value.
Potential Errors:	Error_Generic_CreatingObject Error_Generic_ItemNotFound
Notes:	If the Department object does not exist in the Departments, the call will raise an error.

6.8 EmailDiscardReason

The EmailDiscardReason objects are used in EmailDiscardReasons collection in AdministrationManager::QueryEmailDiscardReasons() query.

Interfaces supported:

- IEmailDiscardReason (default)
- IAdministrationEvent
- IXMLAccess

6.8.1 Properties

This section contains properties exposed through the EmailDiscardReason interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.8.1.1 Description

The Description property returns the e-mail discard description.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oEmailDiscardReason.Description
```

Property:	Description (of EmailDiscardReason)
Description:	This property returns the EmailDiscardReason description.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.8.1.2 Key

The Key property returns the department key.

To use this property, do the following:

```
Dim lKey as Long
lKey = oEmailDiscardReason.Key
```

Property:	Key (of EmailDiscardReason)
Description:	This property returns the EmailDiscardReason key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.8.1.3 Name

The Name property returns the e-mail discard name.

To use this property, do the following:

```
Dim strName as String
strName = oEmailDiscardReason.Name
```

Property:	Name (of EmailDiscardReason)
Description:	This property returns the e-mail discard name.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.9 EmailDiscardReasons

The EmailDiscardReasons object represents a collection of one or more of EmailDiscardReason objects.

Interfaces supported:

- IEmailDiscardReasons (default)
- IXMLAccess

6.9.1 Properties

This section contains properties exposed through the EmailDiscardReasons interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.9.1.1 _NewEnum

```
For Each oEmailDiscardReason in colEmailDiscardReasons
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oEmailDiscardReason as
HiPathProCenterLibrary.EmailDiscardReason
For Each oEmailDiscardReason in colEmailDiscardReasons
...
Next
```

Property:	_NewEnum (of EmailDiscardReasons)
Description:	Provides access to elements of the EmailDiscardReasons by means of operator FOREACH.

Returns:	oEmailDiscardReason — For each item in the EmailDiscardReasons collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum
Notes:	This method allows the EmailDiscardReasons collection to support the 'For Each' iterator.

6.9.1.2 Count

The Count property returns the number of EmailDiscardReason objects that exist in the EmailDiscardReasons collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colEmailDiscardReasons.Count
```

Property:	Count (of EmailDiscardReasons)
Description:	Number of EmailDiscardReason objects in EmailDiscardReasons object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.9.1.3 Item

```
oEmailDiscardReason = Item (EmailDiscardReasonKey as Integer)
```

The Item method finds a EmailDiscardReason object in the EmailDiscardReasons collection that is indexed by the EmailDiscardReasonKey. If an item is found in the collection, the Item method returns the EmailDiscardReason object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

EmailDiscardReasonKey as Integer	This is the EmailDiscardReason key for the item you are requesting.

This method returns the following parameters:

oEmailDiscardReason as IWrapupReason	This is the EmailDiscardReason object for the e-mail Discard reason key value.

To use this method, do the following:

```
Dim oEmailDiscardReason as
HiPathProCenterLibary.EmailDiscardReason

Set oEmailDiscardReason = colEmailDiscardReasons.Item
(emailDiscardReasonKey)
```

This checks to see if the EmailDiscardReason specified by the EmailDiscardReasonKey value exists. If it does, oEmailDiscardReason will contain the EmailDiscardReason object. If it does not exist in the EmailDiscardReasons, this will raise an error.

Method:	Item (of EmailDiscardReasons)
Description:	Finds a EmailDiscardReason in the EmailDiscardReasons collection that is indexed by the e-mail discard key.
Parameters:	reasonKey — The EmailDiscard reason key for the EmailDiscardReason that you are looking for.
Returns:	oEmailDiscardReason — The EmailDiscardReason object specified by the wrapup reason key value.
Potential Errors:	Error_Generic_CreatingObject Error_Generic_ItemNotFound
Notes:	If the EmailDiscardReason object does not exist in the EmailDiscardReasons collection, the call will raise an error.

6.10 EmailTemplate

The EmailTemplate objects are used in EmailTemplates collection in AdministrationManager::QueryEmailTemplates() query.

Interfaces supported:

- IEmailTemplate (default)
- IAdministrationEvent
- IXMLAccess

6.10.1 Properties

This section contains properties exposed through the EmailTemplate interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.10.1.1 Content

The Content property returns the e-mail template content in plain text. When the Content property is invoked, the SDK will request the message content from the Administration Server. The information will not be cached.

If your application makes use of this property often, for performance reasons, it must cache the information. If the information is cached, then the application must also listen to administration update events from the EmailMessageTemplate ObjectType to determine if the cache needs to be updated.

To use this property, do the following:

```
Dim strText as String
strText = oEmailTemplate.Content
```

Property:	Content (of EmailTemplate)
Description:	This property returns the EmailTemplate content.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.10.1.2 Description

The Description property returns the e-mail template description.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oEmailTemplate.Description
```

Property:	Description (of EmailTemplate)
Description:	This property returns the EmailTemplate description.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.10.1.3 Key

The Key property returns the e-mail template key.

To use this property, do the following:

```
Dim lKey as Long
lKey = oEmailTemplate.Key
```

Property:	Key (of EmailTemplate)
Description:	This property returns the EmailTemplate key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.10.1.4 Name

The Name property returns the e-mail template name.

To use this property, do the following:

```
Dim strName as String
strName = oEmailTemplate.Name
```

Property:	Name (of EmailTemplate)
Description:	This property returns the EmailTemplate name.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.10.1.5 QueueKeys

The QueueKeys property returns a collection of queue keys associated with the e-mail template.

To use this property, do the following:

```
Dim oQueueKeys as HiPathProCenterLibrary.KeyList
oQueueKeys = oEmailTemplate.QueueKeys
```

Property:	QueueKeys (of EmailTemplate)
Description:	This property returns a collection of queue keys associated with the EmailTemplate.
Type:	KeyList
Potential Errors:	None
Notes:	Read-only

6.10.1.6 Type

The Type property returns the e-mail template type.

To use this property, do the following:

```
Dim strDescription as EmailTemplateTypes
enType = oEmailTemplate.Type
```

Property:	Type (of EmailTemplate)
Description:	This property returns the EmailTemplate type.
Type:	EmailTemplateTypes
Potential Errors:	None
Notes:	Read-only

6.11 EmailTemplates

The EmailTemplates object represents a collection of one or more of EmailTemplate objects.

Interfaces supported:

- IEmailTemplates (default)
- IXMLAccess

6.11.1 Properties

This section contains properties exposed through the EmailTemplates interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.11.1.1 _NewEnum

```
For Each oEmailTemplate in colEmailTemplates
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++. This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oEmailTemplate as HiPathProCenterLibrary.EmailTemplate
For Each oEmailTemplate in colEmailTemplates
...
Next
```

Property:	_NewEnum (of EmailTemplates)
Description:	Provides access to elements of the EmailTemplates through operator FOREACH.
Returns:	oEmailTemplate — For each item in the EmailTemplates collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum
Notes:	This method allows the EmailTemplates collection to support the 'For Each' iterator.

6.11.1.2 Count

The Count property returns the number of EmailTemplate objects in the EmailTemplates collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colEmailTemplates.Count
```

Property:	Count (of EmailTemplates)
Description:	Number of EmailTemplate objects in EmailTemplates object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.11.1.3 Item

```
oEmailTemplate = Item (EmailTemplateKey as Integer)
```

The Item method finds a EmailTemplate object in the EmailTemplates collection that is indexed by the EmailTemplateKey. If an item is found in the collection, the Item method returns the EmailTemplate object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

EmailTemplate Key as Integer	This is the EmailTemplate key for the item you are requesting.

This method returns the following parameters:

oEmailTemplate as IEmailTemplate	This is the EmailTemplate object for the EmailTemplate key value.

To use this method, do the following:

```
Dim oEmailTemplate as HiPathProCenterLibary.EmailTemplate
Set oEmailTemplate = colEmailTemplates.Item (EmailTemplateKey)
```

This checks to see if the EmailTemplate specified by the EmailTemplateKey value exists. If it does, oEmailTemplate will contain the EmailTemplate object. If it does not exist in the EmailTemplates, this will raise an error.

Method:	Item (of EmailTemplates)
Description:	Finds a EmailTemplate in the EmailTemplates collection that is indexed by the EmailTemplate key.
Parameters:	EmailTemplateKey — The EmailTemplate key for the EmailTemplate that you are looking for.
Returns:	oEmailTemplate — The EmailTemplate object specified by the EmailTemplate key value.
Potential Errors:	Error_Generic_CreatingObject Error_Generic_ItemNotFound
Notes:	If the EmailTemplate object does not exist in the EmailTemplate, the call will raise an error.

6.12 Group

The Group object represents a group object as defined in the system. In the case of skills based routing, a group refers to a configured virtual group.

Interfaces supported:

- IGroup (default)
- IAdministrationEvent
- IXMLAccess

6.12.1 Properties

This section contains properties exposed through the Group interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

6.12.1.1 Description

The Description property contains the description of the Group object.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oGroup.Description
```

Property:	Description (of Group)
Description:	This property returns the group description.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.12.1.2 Key

The Key property is the key associated with the group object in the database.

To use this property, do the following:

```
Dim lKey as Long
lKey = oGroup.Key
```

Property:	Key (of Group)
Description:	Key associated with the Group object in the database.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.12.1.3 Name

The Name property is the name associated with the Group object in the database.

To use this property, do the following:

```
Dim strName as String
strName = oGroup.Name
```

Property:	Name (of Group)
Description:	The name of the Group.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.13 Groups

The Groups object represents a read-only collection of zero or more group objects containing all the groups or virtual groups that are defined in the database.

Interfaces supported:

- IGroups (default)
- IXMLAccess

6.13.1 Properties

This section contains properties exposed through the Groups interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.13.1.1 _NewEnum

```
For Each oGroup in colGroups
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oGroup as HiPathProCenterLibrary.Group
For Each oGroup in colGroups
txtTextBox.Text = txtTextBox.Text & _
"Group Name: " & _
oGroup.Name & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the names of all Groups.

Property:	_NewEnum (of Groups)
Description:	Allows for the 'For Each' operator on the Groups collection.
Returns:	OGroup — For each item in the Groups collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum
Notes:	This method allows the Groups collection to support the 'For Each' iterator.

6.13.1.2 Count

The Count property returns the number of Group objects that exist in the Groups collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colGroups.Count
```

Property:	Count (of Groups)
Description:	The number of Group objects that are in the Groups collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.13.1.3 Item

```
oGroup = Item (Key as Long)
```

The Item method finds a Group object in the Groups collection that is indexed by the Key value. If an item is found in the collection at the Group key value, the Item method returns the Group object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

Key as Long	This is the Group key for the Group you are requesting.

This method returns the following parameters:

oGroup as Group	This is the Group object for Group specified by the Key value. This contains the information for the Group.

To use this method, do the following:

```
Dim oGroup as HiPathProCenterLibary.Group
Set oGroup = colGroups.Item (lKey)
```

This checks to see if the Group specified by the Key exists. If it does, oGroup will contain the Group object. If it does not exist in the Groups collection, this will raise an error.

Method:	Item (of Groups)
Description:	Finds a Group object in the Groups collection that is indexed by the Key key value.
Parameters:	Key — The key for the Group that you are looking for. This is an index starting from 1.

Returns:	oGroup — The summary for the Group that you have requested.
Potential Errors:	Error_Generic_CreatingObject Error_Generic_ItemNotFound
Notes:	If the Group does not exist in our collection, you raise an error.

6.14 Language

The Language objects are used in Language s collection in AdministrationManager::QueryWebCollaborationLanguages() query.

Interfaces supported:

- ILanguage (default)
- IAdministrationEvent
- IXMLAccess

6.14.1 Properties

This section contains properties exposed through the Language interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.14.1.1 Description

The Description property returns the language description.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oLanguage.Description
```

Property:	Description (of Language)
Description:	This property returns the Language description.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.14.1.2 Key

The Key property returns the language key.

To use this property, do the following:

```
Dim lKey as Long
lKey = oLanguage.Key
```

Property:	Key (of Language)
Description:	This property returns the language key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.14.1.3 Name

The Name property returns the language name.

To use this property, do the following:

```
Dim strName as String
strName = oLanguage.Name
```

Property:	Name (of Language)
Description:	This property returns the language name.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.15 Languages

The Languages object represents a collection of one or more of Language objects.

Interfaces supported:

- ILanguages (default)
- IXMLAccess

6.15.1 Properties

This section contains properties exposed through the Languages interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.15.1.1 _NewEnum

```
For Each oLanguage in colLanguages
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oLanguage as HiPathProCenterLibrary.Language
For Each oLanguage in colLanguages
...
Next
```

Property:	_NewEnum (of Languages)
Description:	Provides access to elements of the Languages by means of operator FOREACH.
Returns:	oLanguage — For each item in the Languages collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum
Notes:	None

This property allows the Languages collection to support the 'For Each' iterator.

6.15.1.2 Count

The Count property returns the number of Language objects that exist in the Languages collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colLanguages.Count
```

Property:	Count (of Languages)
Description:	The number of Language objects in Languages object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.15.1.3 Item

```
oLanguage = Item (LanguageKey as Integer)
```

The Item method finds a Language object in the Languages collection that is indexed by the LanguageKey. If an item is found in the collection, the Item method returns the Language object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

LanguageKey as Integer	This is the language key for the item you are requesting.

This method returns the following parameters:

oLanguage as ILanguage	This is the Language object for the language key value.

To use this method, do the following:

```
Dim oLanguage as HiPathProCenterLibary.Language
Set oLanguage = colLanguages.Item (LanguageKey)
```

This checks to see if the Language specified by the LanguageKey value exists. If it does, oLanguage will contain the Language object. If it does not exist in the Languages, this will raise an error.

Property:	Item (of Languages)
Description:	Finds a Language in the Languages collection that is indexed by the language key.
Parameters:	languageKey — The language key for the Language that you are looking for.
Returns:	oLanguage — The Language object specified by the language key value.
Potential Errors:	Error_Generic_CreatingObject Error_Generic_ItemNotFound
Notes:	If the Language object does not exist in the Languages, the call will raise an error.

6.16 PostProcessingReason

The PostProcessingReason object contains information about a Post-processing reason. This includes the reasonkey, name as well as description.

Interfaces supported:

- IPostProcessingReason (default)
- IXMLAccess

6.16.1 Properties

This section contains properties exposed through the PostProcessingReason interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on

page 59.

6.16.1.1 Description

The Description property contains the description of the Post-processing reason.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oPostProcessingReason.Description
```

Property:	Description (of PostProcessingReason)
Description:	Specifies the description name of the Post-processing reason.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.16.1.2 Name

The Name property contains the name of the Post-processing reason code.

To use this property, do the following:

```
Dim strName as String
strName = oPostProcessingReason. Name
```

Property:	Name (of PostProcessingReason)
Description:	The name of Post-processing reason code.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.16.1.3 ReasonKey

The ReasonKey property contains the reason key in the database for the user going into Post-processing state.

To use this property, do the following:

```
Dim lReasonKey as Long
lReasonKey = oPostProcessingReason.ReasonKey
```

Property:	ReasonKey (of PostProcessingReason)
Description:	The reason key for the user going into Post-processing state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.17 PostProcessingReasons

The PostProcessingReasons object is a collection of the PostProcessingReason objects.

Interfaces supported:

- IPostProcessingReasons (default)
- IXMLAccess

6.17.1 Properties

This section contains properties exposed through the PostProcessingReasons interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.17.1.1 _NewEnum

```
For Each oPostProcessingReason in colPostProcessingReasons
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator.

For example, to use this method, do the following.

```
Dim oPostProcessingReason as
HiPathProCenterLibrary.PostProcessingReason
For Each oPostProcessingReason in m_colPostProcessingReasons
   txtTextBox.Text = txtTextBox.Text & _
```

```
        "ReasonKey: " & _
        oPostProcessingReason.ReasonKey & _
        "Name: " & _
        oPostProcessingReason.Name & _
        "Description: " & _
        oPostProcessingReason.Description & _
        vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the reasons ReasonKey, Name and Description of all the PostProcessingReason objects in the PostProcessingReasons collection.

Property:	_NewEnum (of PostProcessingReasons)
Description:	Allows for the 'For Each' operator on the PostProcessingReasons collection.
Returns:	PostProcessingReason — For each item in the PostProcessingReasons collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum
Notes:	This method allows the PostProcessingReasons collection to support the 'For Each' iterator.

6.17.1.2 Count

The Count property returns the number of PostProcessingReason objects that exist in the PostProcessingReasons collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colPostProcessingReasons.Count
```

Property:	Count (of PostProcessingReasons)
Description:	The number of PostProcessingReason objects that are in the PostProcessingReasons collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.17.1.3 Item

```
oPostProcessingReason = Item (ReasonKey as Long)
```

Use the Item method to find a PostProcessingReason object in the PostProcessingReasons collection that is indexed by the ReasonKey value. If an item is found in the collection at the ReasonKey value, the Item method returns the PostProcessingReason object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameter:

ReasonKey as Long	This is the key for the Post-processing reason you are requesting.

This method returns the following parameters:

oPostProcessingReason as PostProcessingReason	The PostProcessingReason object for the Post-processing reason specified by the ReasonKey value. This contains the PostProcessingReason for the key specified.

To use this method, do the following:

```
Dim oPostProcessingReason as HiPathProCenterLibary.
PostProcessingReason

Set oPostProcessingReason = colPostProcessingReasons.Item
(ReasonKey)
```

This checks to see if the PostProcessingReason specified by the ReasonKey exists. If it does, oPostProcessingReason will contain the PostProcessingReason object. If it does not exist in the PostProcessingReasons collection, this will raise an error.

Method:	Item (of PostProcessingReasons)
Description:	Finds a PostProcessingReason object in the PostProcessingReasons collection that is indexed by the ReasonKey value.
Parameters:	ReasonKey — The key for the PostProcessingReason that you are looking for.
Returns:	oPostProcessingReason — The PostProcessingReason for the key specified.
Potential Errors:	Error_Generic_ItemNotFound
Notes:	If the PostProcessingReason object for the user does not exist in the collection, it will raise an error.

6.18 Queue

The Queue object represents a queue defined in the system. These define how you should handle certain contacts (voice, callback, e-mail, or Web collaboration).

Interfaces supported:

- IQueue2 (default)
- IAdministrationEvent
- IXMLAccess

6.18.1 Properties

This section contains properties exposed through the Queue interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.18.1.1 Description

The Description property contains the description of the Queue object.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oQueue.Description
```

Property:	Description (of Queue)
Description:	Specifies the description for the queue which is a logical container where calls wait until they are handled.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.18.1.2 Key

The Key property stores the unique key from the Queues table in the database.

To use this property, do the following:

```
Dim lKey as Long
lKey = oQueue.Key
```

Property:	Key (of Queue)
Description:	The database table unique key for the Queue object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.18.1.3 MediaType

The MediaType property stores what media type this queue is. Each queue can only correspond to a single media type (voice, callback, e-mail or Web collaboration).

To use this property, do the following:

```
Dim enMediaTypes as HiPathProCenterLibrary.enMediaTypes
enMediaTypes = oQueue.MediaType
```

Property:	MediaType (of Queue)
Description:	The media type for the queue.
Type:	enMediaTypes
Potential Errors:	None
Notes:	Read-only

6.18.1.4 Name

The Name property stores the name of the queue (for example, Sales or Support).

To use this property, do the following:

```
Dim strName as String
strName = oQueue.Name
```

Property:	Name (of Queue)
Description:	The name of the queue.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.19 Queues

The Queues object is a collection of the Queue objects, containing all the queues that are defined in the database.

Interfaces supported:

- IQueues (default)
- IXMLAccess

6.19.1 Properties

This section contains properties exposed through the Queues interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.19.1.1 _NewEnum

```
For Each oQueue in colQueues
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the FOREACH iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'FOREACH' iterator. For example, to make use of this method, do the following.

```
Dim oQueue as HiPathProCenterLibrary.Queue
For Each oQueue in colQueues
   txtTextBox.Text = txtTextBox.Text & _
   "Queue Name: " & _
   oQueue.Name & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the names of the different steps in the queue.

Property:	_NewEnum (of Queues)
Description:	Allows for the 'FOREACH' operator on the Queues collection.
Returns:	oQueue — For each item in the Queues collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum Error_Generic_BaseObjectDoesNotExist
Notes:	This method allows the Queues collection to support the 'FOREACH' iterator.

6.19.1.2 Count

The Count property returns the number of Queue objects that exist in the Queues collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colQueues.Count
```

Property:	Count (of Queues)
Description:	The number of Queue objects that are in the Queues collection.
Type:	Long
Potential Errors:	Error_Generic_BaseObjectDoesNotExist
Notes:	Read-only

6.19.1.3 Item

```
oQueue = Item (QueueKey as Long)
```

The Item method finds a Queue object in the Queues collection that is indexed by the Queuekey value. If an item is found in the collection at the queue key value, the Item method returns the Queue object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

QueueKey as Long	This is the queue key for the queue you are requesting.

This method returns the following parameters:

oQueue as Queue	This is the Queue object for a queue specified by the QueueKey value. This contains the information for the queue.

To use this method, do the following:

```
Dim oQueue as HiPathProCenterLibary.Queue
Set oQueue = colQueues.Item (lQueueKey)
```

This checks to see if the queue specified by the QueueKey exists. If it does, oQueue will contain the Queue object. If it does not exist in the Queues collection, this will raise an error.

Method:	Item (of Queues)
Description:	Finds a Queue object in the Queues collection that is indexed by the QueueKey key value.
Parameters:	QueueKey — The key for the queue that you are looking for. This is an index starting from 1.
Returns:	oQueue — The summary for the queue that you have requested.
Potential Errors:	Error_Generic_BaseObjectDoesNotExist Error_Generic_ItemNotFound
Notes:	If the queue does not exist in our collection, you raise an error.

6.20 RoutingStateReason

The RoutingStateReason object contains information about an Unavailable or Work reason. This includes the reasonkey, name as well as description.

Interfaces supported:

- IRoutingStateReason (default)
- IXMLAccess

6.20.1 Properties

This section contains properties exposed through the RoutingStateReason interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.20.1.1 Description

The Description property contains the description of the Unavailable/ Work reason.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oRoutingStateReason.Description
```

Property:	Description (of RoutingStateReason)
Description:	Specifies the description name of the Unavailable/Work reason.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.20.1.2 Name

The Name property contains the name of the Unavailable/Work reason code.

To use this property, do the following:

```
Dim strName as String
strName = oRoutingStateReason. Name
```

Property:	Name (of RoutingStateReason)
Description:	The name of Unavailable/Work reason code.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.20.1.3 ReasonKey

The ReasonKey property contains the reason key in the database for the user going into Unavailable/Work state.

To use this property, do the following:

```
Dim lReasonKey as Long
lReasonKey = oRoutingStateReason.ReasonKey
```

Property:	ReasonKey (of RoutingStateReason)
Description:	The reason key for the user going into Unavailable/Work state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.21 RoutingStateReasons

The RoutingStateReasons object is a collection of the RoutingStateReason objects.

Interfaces supported:

- IRoutingStateReasons (default)
- IXMLAccess

6.21.1 Properties

This section contains properties exposed through the RoutingStateReasons interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.21.1.1 _NewEnum

```
For Each oRoutingStateReason in colRoutingStateReasons
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator.

For example, to use this method, do the following.

```
Dim oRoutingStateReason as
HiPathProCenterLibrary.RoutingStateReason
For Each oRoutingStateReason in m_colRoutingStateReasons
   txtTextBox.Text = txtTextBox.Text & _
      "ReasonKey: " & _
      oRoutingStateReason.ReasonKey & _
      "Name: " & _
      oRoutingStateReason.Name & _
      "Description: " & _
      oRoutingStateReason.Description & _
      vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the reasons ReasonKey, Name and Description of all the RoutingStateReason objects in the RoutingStateReasons collection.

Property:	_NewEnum (of RoutingStateReasons)
Description:	Allows for the 'For Each' operator on the RoutingStateReasons collection.
Returns:	RoutingStateReason — For each item in the RoutingStateReasons collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum
Notes:	This method allows the RoutingStateReasons collection to support the 'For Each' iterator.

6.21.1.2 Count

The Count property returns the number of RoutingStateReason objects that exist in the RoutingStateReasons collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colRoutingStateReasons.Count
```

Property:	Count (of RoutingStateReasons)
Description:	The number of RoutingStateReason objects that are in the RoutingStateReasons collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.21.1.3 Item

```
oRoutingStateReason = Item (ReasonKey as Long)
```

Use the Item method to find a RoutingStateReason object in the RoutingStateReasons collection that is indexed by the ReasonKey value. If an item is found in the collection at the ReasonKey value, the Item method returns the RoutingStateReason object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameter:

ReasonKey as Long	This is the key for the routing state reason you are requesting.

This method returns the following parameters:

oRoutingStateReason as RoutingStateReason	The RoutingStateReason object for the routing state reason specified by the ReasonKey value. This contains the RoutingStateReason for the key specified.

To use this method, do the following:

```
Dim oRoutingStateReason as HiPathProCenterLibary.
RoutingStateReason
Set oRoutingStateReason = colRoutingStateReasons.Item
(ReasonKey)
```

This checks to see if the RoutingStateReason specified by the ReasonKey exists. If it does, oRoutingStateReason will contain the RoutingStateReason object. If it does not exist in the RoutingStateReasons collection, this will raise an error.

Method:	Item (of RoutingStateReasons)
Description:	Finds a RoutingStateReason object in the RoutingStateReasons collection that is indexed by the ReasonKey value.
Parameters:	ReasonKey — The key for the RoutingStateReason that you are looking for.
Returns:	oRoutingStateReason — The RoutingStateReason for the key specified.
Potential Errors:	Error_Generic_ItemNotFound
Notes:	If the RoutingStateReason object for the user does not exist in the collection, it will raise an error.

6.22 User

The User object represents a user that is defined in the system. This can be a user, a supervisor, or any other valid user type.

Interfaces supported:

- IUser4 (default)
- IAdministrationEvent
- IXMLAccess

6.22.1 Properties

This section contains properties exposed through the User interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

6.22.1.1 ACDGroupNumber

The ACDGroupNumber property stores the group number of the backup ACD group, UCD group, or hunt group that the user belongs to.

To use this property, do the following:

```
Dim strACDGroupNumber as String
strACDGroupNumber = oUser.ACDGroupNumber
```

Property:	ACDGroupNumber (of User)
Description:	Stores the group number of the backup ACD group, UCD group, or hunt group that the user belongs to.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.22.1.2 Department

The Department property stores the name of the department that the user belongs to.

To use this property, do the following:

```
Dim strDepartment as String
strDepartment = objUser.Department
```

Property:	Department (of User)
Description:	The name of the department that the user belongs to.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.22.1.3 DepartmentKey

The DepartmentKey property returns the DepartmentKey number defined in the database that indexes the Department object associated with a user.

To use this property, do the following:

```
Dim lDepartmentKey as Long
lDepartmentKey = colUsers.DepartmentKey
```

Property:	DepartmentKey (of User)
Description:	The DepartmentKey number of the Department associated with the user.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.22.1.4 FirstName

The FirstName property stores the first name of the user.

To use this property, do the following:

```
Dim strFirstName as String
strFirstName = oUser.FirstName
```

Property:	FirstName (of User)
Description:	The first name of the user.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.22.1.5 ID

The ID property stores the ID that the user will use to log on. This is a global unique string (in the system) that is defined by the system administrator, and used by the user to identify themselves.

To use this property, do the following:

```
Dim strID as String
strID = oUser.ID
```

Property:	ID (of User)
Description:	The ID of the user.
Type:	String
Potential Errors:	None
Notes:	Read-only. While this is similar to the Key property (both are unique ways to identify the user), the ID is user defined, whereas the Key is system defined. In most of the SDK, Key is the preferred way of interacting with the objects, although in most cases, the customer will use the ID to originally identify the user.

6.22.1.6 Key

The Key property is the unique key for the user in the database. This is a system generated value.

To use this property, do the following:

```
Dim lKey as Long
lKey = oUser.Key
```

Property:	Key (of User)
Description:	The database table unique key for the User object.
Type:	Long
Potential Errors:	None
Notes:	Read-only. While this is similar to the ID property (both are unique ways to identify the user), this Key is system defined, whereas the ID is user defined. In most of the SDK, Key is the preferred way of interacting with the objects although in most cases, the customer will use the ID to originally identify the user.

6.22.1.7 LastName

The LastName property stores the last name of the user.

To use this property, do the following:

```
Dim strLastName as String
strLastName = oUser.LastName
```

Property:	LastName (of User)
Description:	The last name of the user.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.22.1.8 MaximumCapacity

The MaximumCapacity property returns the maximum concurrent contacts that the user can handle among all the configured contact handling rules.

To use this property, do the following:

```
Dim lMaximumCapacity as long
lMaximumCapacity = oUser.MaximumCapacity
```

Property:	MaximumCapacity (of User)
Description:	The maximum concurrent contacts that the user can handle.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.22.1.9 MediaTypes

The MediaTypes property stores a collection of the various media types that the user is eligible to handle.

To use this property, do the following:

```
Dim colMediaTypes as HiPathProCenterLibrary.MediaTypes
```

```
Set colMediaTypes = oUser.MediaTypes
```

Property:	MediaTypes (of User)
Description:	A collection of the various media types that the user is eligible for.
Type:	MediaTypes
Potential Errors:	None
Notes:	Read-only

6.22.1.10 MiddleInitial

The MiddleInitial property stores the middle initial of the user.

To use this property, do the following:

```
Dim strMiddleInitial as String
strMiddleInitial = oUser.MiddleInitial
```

Property:	MiddleInitial (of User)
Description:	The middle initial of the user.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.22.1.11 MultipleContactHandlingCapable

The MultipleContactHandlingCapable property returns false if the user's contact handling rules are only system contact handling rules.

To use this property, do the following:

```
Dim bMultipleContactHandlingCapable as Boolean
bMultipleContactHandlingCapable =
oUser.MultipleContactHandlingCapable
```

Property:	MultipleContactHandlingCapable (of User)
Description:	Returns true or false depending on whether the user can handle multiple routed contacts at the same time.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

6.22.1.12 Type

The Type property stores the user type of the user. For example, is this type a user, a Supervisor, or an Administrator.

To use this property, do the following:

```
Dim enType as HiPathProCenterLibrary.enUserTypes
enType = oUser.Type
```

Property:	Type (of User)
Description:	The user type for this user.
Type:	enUserTypes
Potential Errors:	None
Notes:	Read-only

6.22.1.13 UserName

The UserName property returns the UserName of the user that is defined in the database.

To use this property, do the following:

```
Dim strUserName as String
strUserName = oUser.UserName
```

Property:	UserName (of User)
Description:	The UserName of the User
Type:	String
Potential Errors:	None
Notes:	Read-only

6.23 Users

The Users object is a collection of the user objects, containing all the users who are defined in the database.

Interfaces supported:

- IUsers (default)
- IXMLAccess

6.23.1 Properties

This section contains properties exposed through the Users interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.23.1.1 _NewEnum

```
For Each oUser in colUsers
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oUser as HiPathProCenterLibrary.User
For Each oUser in colUsers
   txtTextBox.Text = txtTextBox.Text & _
      "User Name: " & _
      oUser.FirstName & " " & _
      oUser.LastName & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the names of the users in the users collection.

Property:	_NewEnum (of Users)
Description:	Allows for the 'For Each' operator on the User collection.
Parameters:	UserKey — The key for the user that you are looking for.
Returns:	oUser — For each item in the Users collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum Error_Generic_BaseObjectDoesNotExist
Notes:	This method allows the Users collection to support the 'For Each' iterator.

6.23.1.2 Count

The Count property returns the number of user objects that exist in the Users collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colUsers.Count
```

Property:	Count (of Users)
Description:	The number of user objects that are in the Users collection.
Type:	Long
Potential Errors:	Error_Generic_BaseObjectDoesNotExist
Notes:	Read-only

6.23.1.3 Item

```
User = Item (UserKey as Long)
```

The Item method finds a User object in the Users collection that is indexed by the UserKey key value. If an item is found in the collection at the user key value, the Item method returns the User object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameter:

UserKey as Long	This is the user key for the user you are requesting.

This method returns the following parameter:

User as User	This is the User object for user specified by the UserKey value. This contains the user information for the user.

To use this method, do the following:

```
Dim oUser as HiPathProCenterLibary.User
Set oUser = colUsers.Item (lUserKey)
```

This checks to see if the user for the user specified by the UserKey exists. If it does, oUser will contain the User object. If it does not exist in the Users collection, this will raise an error.

Method:	Item (of Users)
Description:	Finds a User object in the Users collection that is indexed by the UserKey key value.
Parameters:	UserKey — The key for the user that you are looking for.

Returns:	User — The user that you have requested.
Potential Errors:	Error_Generic_BaseObjectDoesNotExist Error_Generic_ItemNotFound
Notes:	If the queue does not exist in our collection, you raise an error.

6.24 WebCollaborationTemplate

The WebCollaborationTemplate objects are used in WebCollaborationTemplates collection in AdministrationManager::QueryWebCollaborationTemplates() query.

Interfaces supported:

- IWebCollaborationTemplate (default)
- IAdministrationEvent
- IXMLAccess

6.24.1 Properties

This section contains properties exposed through the WebCollaborationTemplate interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

6.24.1.1 Content

The Content property returns the Web collaboration template content text.

To use this property, do the following:

```
Dim strText as String
strText = oWebCollaborationTemplate.Content
```

Property:	Content (of WebCollaborationTemplate)
Description	This property returns the WebCollaborationTemplate content.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.24.1.2 Description

The Description property returns the Web collaboration template description.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oWebCollaboration.Description
```

Property:	Description (of WebCollaborationTemplate)
Description:	This property returns the WebCollaborationTemplate description.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.24.1.3 Key

The Key property returns the e-mail template key.

To use this property, do the following:

```
Dim lKey as Long
lKey = oWebCollaborationTemplate.Key
```

Property:	Key (of WebCollaborationTemplate)
Description:	This property returns the Web collaboration template key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.24.1.4 LanguageKey

The LanguageKey property returns the Web collaboration template language key.

To use this property, do the following:

```
Dim lKey as Long
lKey = oWebCollaborationTemplate.LanguageKey
```

Property:	LanguageKey (of WebCollaborationTemplate)
Description:	This property returns the Web collaboration template language key.

Type:	Long
Potential Errors:	None
Notes:	Read-only

6.24.1.5 Name

The Name property returns the Web collaboration template name.

To use this property, do the following:

```
Dim strName as String
strName = oWebCollaborationTemplate.Name
```

Property:	Name (of WebCollaborationTemplate)
Description:	This property returns the Web collaboration template name.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.24.1.6 QueueKeys

The QueueKeys property returns a collection of queue keys associated with the e-mail template.

To use this property, do the following:

```
Dim oQueueKeys as HiPathProCenterLibrary.KeyList
oQueueKeys = oWebCollaborationTemplate.QueueKeys
```

Property:	QueueKeys (of WebCollaborationTemplate)
Description:	This property returns a collection of queue keys associated with the WebCollaborationTemplate.
Type:	KeyList
Potential Errors:	None
Notes:	Read-only

6.24.1.7 Type

The Type property returns the Web collaboration template type.

To use this property, do the following:

```
Dim strDescription as WebCollaborationTemplateTypes
```

```
enType = oWebCollaborationTemplate.Type
```

Property:	Type (of WebCollaborationTemplate)
Description:	This property returns the WebCollaborationTemplate type.
Type:	WebCollaborationTemplateTypes
Potential Errors:	None
Notes:	Read-only

6.25 WebCollaborationTemplates

The WebCollaborationTemplates object represents a collection of one or more of WebCollaborationTemplate objects.

Interfaces supported:

- IWebCollaborationTemplates (default)
- IXMLAccess

6.25.1 Properties

This section contains properties exposed through the WebCollaborationTemplates interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.25.1.1 _NewEnum

```
For Each oWebCollaborationTemplate in
colWebCollaborationTemplates

...

Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oWebCollaborationTemplate as
HiPathProCenterLibrary.WebCollaborationTemplate

For Each oWebCollaborationTemplate in
colWebCollaborationTemplates

...

Next
```

Property:	_NewEnum (of WebCollaborationTemplates)
Description:	Provides access to elements of the WebCollaborationTemplates by means of operator FOREACH.
Returns:	oWebCollaborationTemplate — For each item in the WebCollaborationTemplates collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum
Notes:	This method allows the WebCollaborationTemplates collection to support the 'For Each' iterator.

6.25.1.2 Count

The Count property returns the number of WebCollaborationTemplate objects that exist in the WebCollaborationTemplates collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colWebCollaborationTemplates.Count
```

Property:	Count (of WebCollaborationTemplates)
Description:	The number of WebCollaborationTemplate objects in WebCollaborationTemplates object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.25.1.3 Item

```
oWebCollaborationTemplate = Item (WebCollaborationTemplateKey as Integer)
```

The Item method finds a WebCollaborationTemplate object in the WebCollaborationTemplates collection that is indexed by the WebCollaborationTemplateKey. If an item is found in the collection, the Item method returns the WebCollaborationTemplate object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

WebCollaborationTemplateKey as Integer	This is the WebCollaborationTemplate key for the item you are requesting.

This method returns the following parameters:

oWeb CollaborationTemplate as IWebCollaborationTemplate	This is the WebCollaborationTemplate object for the WebCollaborationTemplate key value.

To use this method, do the following:

```
Dim oWebCollaborationTemplate as
HiPathProCenterLibary.WebCollaborationTemplate

Set oWebCollaborationTemplate =
colWebCollaborationTemplates.Item (WebCollaborationTemplateKey)
```

This checks to see if the WebCollaborationTemplate specified by the WebCollaborationTemplateKey value exists. If it does, oWebCollaborationTemplate will contain the WebCollaborationTemplate object. If it does not exist in the WebCollaborationTemplates, this will raise an error.

Method:	Item (of WebCollaborationTemplates)
Description:	Finds a WebCollaborationTemplate in the WebCollaborationTemplates collection that is indexed by the WebCollaborationTemplate key.
Parameters:	WebCollaborationTemplateKey — The WebCollaborationTemplate key for the WebCollaborationTemplate that you are looking for.
Returns:	oWebCollaborationTemplate — The WebCollaborationTemplate object specified by the WebCollaborationTemplate key value.
Potential Errors:	Error_Generic_CreatingObject Error_Generic_ItemNotFound
Notes:	If the WebCollaborationTemplate object does not exist in the WebCollaborationTemplate, the call will raise an error.

6.26 WrapupReason

The WrapupReason objects are used in WrapupReasons collection in AdministrationManager::Query WrapupReasons() query.

Interfaces supported:

- IWrapupReason (default)
- IAdministrationEvent
- IXMLAccess

6.26.1 Properties

This section contains properties exposed through the WrapupReason interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.26.1.1 Description

The Description property returns the wrapup reason description.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oWrapupReason.Description
```

Property:	Description (of WrapupReason)
Description:	This property returns the WrapupReason description.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.26.1.2 Key

The Key property returns the wrapup reason key.

To use this property, do the following:

```
Dim lKey as Long
lKey = oWrapupReason.Key
```

Property:	Key (of WrapupReason)
Description:	This property returns the wrapup reason key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.26.1.3 Name

The Name property returns the wrapup reason name.

To use this property, do the following:

```
Dim strName as String
strName = oWrapupReason.Name
```

Property:	Name (of WrapupReason)
Description:	This property returns the wrapup reason name.
Type:	String
Potential Errors:	None
Notes:	Read-only

6.27 WrapupReasons

The WrapupReasons object represents a collection of one or more of WrapupReason objects.

Interfaces supported:

- IWrapupReasons (default)
- IXMLAccess

6.27.1 Properties

This section contains properties exposed through the WrapupReasons interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

6.27.1.1 _NewEnum

```
For Each oWrapupReason in colWrapupReasons
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oWrapupReason as HiPathProCenterLibrary.WrapupReason
For Each oWrapupReason in colWrapupReasons
...
Next
```

Property:	_NewEnum (of WrapupReasons)
Description:	Provides access to elements of the WrapupReasons by means of operator FOREACH.
Returns:	oWrapupReason — For each item in the WrapupReasons collection until all items have been returned.
Potential Errors:	Error_Generic_UnableToAllocateMemory Error_Generic_IterationGetNewEnum
Notes:	This method allows the WrapupReasons collection to support the 'For Each' iterator.

6.27.1.2 Count

The Count property returns the number of WrapupReason objects that exist in the WrapupReasons collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colWrapupReasons.Count
```

Property:	Count (of WrapupReasons)
Description:	The number of WrapupReason objects in WrapupReasons object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

6.27.1.3 Item

```
oWrapupReason = Item (ReasonKey as Integer)
```

The Item method finds a WrapupReason object in the WrapupReasons collection that is indexed by the wrapupReasonKey. If an item is found in the collection, the Item method returns the WrapupReason object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

ReasonKey as Integer	This is the wrapup reason key for the item you are requesting.

This method returns the following parameters:

oWrapupReason as IWrapupReason	This is the WrapupReason object for the wrapup reason key value.

To use this method, do the following:

```
Dim oWrapupReason as HiPathProCenterLibary.WrapupReason
Set oWrapupReason = colWrapupReasons.Item (wrapupReasonKey)
```

This checks to see if the WrapupReason specified by the wrapupReasonKey value exists. If it does, o WrapupReason will contain the WrapupReason object. If it does not exist in the WrapupReasons, this will raise an error.

Method:	Item (of WrapupReasons)
Description:	Finds a WrapupReason in the WrapupReasons collection that is indexed by the department key.
Parameters:	reasonKey — the wrapup reason key for the WrapupReason that you are looking for.
Returns:	o WrapupReason — the WrapupReason object specified by the wrapup reason key value.
Potential Errors:	Error_Generic_CreatingObject Error_Generic_ItemNotFound
Notes:	If the WrapupReason object does not exist in the WrapupReasons, the call will raise an error.

7 Media Manager

The Media Manager is responsible for handling all the contacts in the system. This includes voice, callback, e-mail, and Web collaboration contacts. The Media Manager also determines the current status of contacts and users in the system.

7.1 MediaManager object

The MediaManager is a non-creatable object that provides access to the various media functionalities in the system. This includes tasks such as creating the various contact objects, passing along events about contacts, and providing a method of viewing the current contacts in queue.

The MediaManager is the access point for anything that has to do with current contacts in the system.

Interfaces supported:

- IMediaManager4 (default)

7.1.1 Properties

This section contains properties exposed through the MediaManager interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.1.1.1 State

The State property returns the current state of the MediaManager. This refers to the state that the MediaManager is in (that is, Available or Unavailable).

This property returns a value from the enManagerStates enumerations described in Section 3.28, “enManagerStates”, on page 87.

To use this property, do the following:

```
Dim enState as HiPathProCenterLibrary.enStates
enState = g_oMediaManager.State
```

Property: State (of MediaManager)

Description: The state of the Media Manager (Available/Unavailable).

Type:	enManagerStates
Potential Errors:	Error_Generic_UnableToDetermineState
Notes:	Read-only

7.1.2 Methods

This section contains methods exposed through the MediaManager interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.1.2.1 GetFunctionalityState

```
enManagerStates = GetFunctionalityState (MediaFunctionality as
enMediaFunctionalities)
```

The MediaManager may occasionally lose its connection to one or more servers in the system. When this occurs, you can check to see what functionality is supported. To do this, you can use the GetFunctionalityState method to pass in the functionality that you are interested in to return the state of that functionality.

This method accepts the following parameters:

Media Functionality as enMedia Functionalities	This is the functionality group that you want to enquire about. See Section 3.33, "enMediaFunctionalities", on page 100.

This method returns the ManagerState of the functionality group you have requested. For more information, see Section 3.28, "enManagerStates", on page 87.

To use this method, do the following after creating the MediaManager:

```
Dim enManagerState as HiPathProCenterLibrary.enManagerStates

enManagerState = g_oMediaManager.GetFunctionalityState
(MediaFunctionality_Voice)
```

This checks with the MediaManager to see if you have the ability to receive voice events.

Method:	GetFunctionalityState (of MediaManager)
Description:	Queries the state of various functionality groups in the MediaManager.
Parameters:	MediaFunctionality — the functionality group whose state you want to check.

Returns:	enManagerStates — the state of the functionality group you requested.
Potential Errors:	Error_Generic_UnableToDetermineState
Notes:	None

7.1.2.2 NewAgent

```
oAgent = NewAgent()
```

To create a User object, you must have the MediaManager create one on your behalf. To do this, you must call NewAgent.

This method does not accept any parameters.

This method returns a newly created Agent object. For more information, see Section 7.2, "Agent", on page 279.

To use this method, do the following after creating the MediaManager:

```
Dim oAgent as HiPathProCenterLibrary.Agent
Set oAgent = g_oMediaManager.NewAgent
```

Method:	NewAgent (of MediaManager)
Description:	Creates a new Agent object.
Parameters:	None
Returns:	Agent — a newly created Agent object.
Potential Errors:	Error_Generic_CreatingObject
Notes:	None

7.1.2.3 NewCallbackCall

```
oCallbackCall = NewCallbackCall()
```

To create a CallbackCall object, you must have the MediaManager create one on your behalf. To do this, you must call NewCallbackCall.

This method does not accept any parameters.

This method returns a newly created CallbackCall object. For more information, see Section 7.5, "CallbackCall", on page 301.

To use this method, do the following after creating the MediaManager:

```
Dim oCallbackCall as HiPathProCenterLibrary.CallbackCall
```

```
Set oCallbackCall = g_oMediaManager.NewCallbackCall
```

Method:	NewCallbackCall (of MediaManager)
Description:	Creates a new CallbackCall object.
Parameters:	None
Returns:	CallbackCall — a newly created CallbackCall object.
Potential Errors:	Error_Generic_CreatingObject
Notes:	None

7.1.2.4 NewEmailCall

```
oEmailCall = NewEmailCall()
```

To create an EmailCall object, you must have the MediaManager create one on your behalf. To do this, you must call NewEmailCall.

This method does not accept any parameters.

This method returns a newly created EmailCall object. For more information, see Section 7.23, "EmailCall", on page 375.

To use this method, do the following after creating the MediaManager:

```
Dim oEmailCall as HiPathProCenterLibrary.EmailCall
Set oEmailCall = g_oMediaManager.NewEmailCall
```

Method:	NewEmailCall (of MediaManager)
Description:	Creates a new EmailCall object.
Parameters:	None
Returns:	EmailCall — A newly created EmailCall object.
Potential Errors:	Error_Generic_CreatingObject
Notes:	None

7.1.2.5 NewServicesEnabled

```
oServicesEnabled = NewServicesEnabled()
```

To create a ServicesEnabled object, you must have the MediaManager create one on your behalf.

To do this, you must call NewServicesEnabled.

This method does not accept any parameters.

This method returns a newly created ServicesEnabled object. For more information, see Section 7.42, "ServicesEnabled", on page 464

To use this method, do the following after creating the MediaManager:

```
Dim oServicesEnabled as HiPathProCenterLibrary.ServicesEnabled
Set oServicesEnabled = g_oMediaManager.NewServicesEnabled
```

Method:	NewServicesEnabled (of MediaManager)
Description:	Creates a new NewServicesEnabled object.
Parameters:	None
Returns:	ServicesEnabled — A newly created ServicesEnabled object.
Potential Errors:	Error_Generic_CreatingObject

7.1.2.6 NewVoiceCall

```
oVoiceCall = NewVoiceCall()
```

To create a VoiceCall object, you must have the MediaManager create one on your behalf. To do this, you must call NewVoiceCall.

This method does not accept any parameters.

This method returns a newly created VoiceCall object. For more information, see Section 7.51, “VoiceCall”, on page 502.

To use this method, do the following after creating the MediaManager:

```
Dim oVoiceCall as HiPathProCenterLibrary.VoiceCall
Set oVoiceCall = g_oMediaManager.NewVoiceCall
```

Method:	NewVoiceCall (of MediaManager)
Description:	Creates a new VoiceCall object.
Parameters:	None
Returns:	VoiceCall — a newly created VoiceCall object.
Potential Errors:	Error_Generic_CreatingObject
Notes:	None

7.1.2.7 NewWebCollaborationCall

```
oWebCollaborationCall = NewWebCollaborationCall ()
```

To create a WebCollaborationCall object, you must have the MediaManager create one on your behalf. To do this, you must call NewWebCollaborationCall.

This method does not accept any parameters.

This method returns a newly created WebCollaborationCall object. For more information, see Section 7.52, “WebCollaborationCall”, on page 519.

To use this method, do the following after creating the MediaManager:

```
Dim oWebCollaborationCall as
HiPathProCenterLibrary.WebCollaborationCall

Set oWebCollaborationCall =
g_oMediaManager.NewWebCollaborationCall
```

Method:	NewWebCollaborationCall (of MediaManager)
Description:	Creates a new WebCollaborationCall object.
Parameters:	None
Returns:	WebCollaborationCall — A newly created WebCollaborationCall object.
Potential Errors:	• Error_Generic_CreatingObject • Error_Media_ServerNotConfigured

7.1.2.8 Query

```
Query (MediaBase as MediaBase)
```

To Query an appropriate object through the MediaManager you must pass the object into the Query method. The object being passed in must support the MediaBase interface. If you are querying a user, ensure that the agent Key has been set. If you are querying a call of any sort, ensure that the CallID has been set.

This method accepts the following parameters:

MediaBase as MediaBase	This is the MediaBase object that you want to query.

This method does not return any new value, but does update the object passed into the method with the updated object data. For example, if a VoiceCall object was passed in to be queried, the updated VoiceCall object would be returned.

To use this method, do the following after creating the MediaManager:

```
oVoiceCall.CallID = "1234"

Call g_oMediaManager.Query (oVoiceCall)

MsgBox "Call Description: " & oVoiceCall.Description
```

CallbackCall object can now be queried using the existing Query method.

To use this method, do the following after creating an empty CallbackCall:

```
oCallbackCall.CallID = strCallID
```

```
Call oMediaManager.Query(oCallbackCall)
```

Method:	Query (of MediaManager)
Description:	Queries an object through the MediaManager.
Parameters:	MediaBase — An object that supports the event base interface that you want to query.
Returns:	None
Potential Errors:	• Error_Generic_QueryFailed • Error_Generic_UnableToReadObjectInformation • Error_Generic_UnableToWriteObjectInformation • Error_Generic_UnknownObjectType • Error_HiPathProCenter_CouldNotFindTelephonyServerAddress • Error_HiPathProCenter_CouldNotFindWebCollaborationServerAddress
Notes:	The object being passed in must support the MediaBase interface. The object being passed in will be updated with the updated object data.

7.1.2.9 QueryAsyncClientEmailsHistory

QueryAsyncClientEmailsHistory method enables SDK client applications to query e-mail messages stored in the system. This method is asynchronous. It submits the request to the E-mail Server, and sometime later, the E-mail Server replies by sending EmailsHistoryQueryResultEvent to the SDK which then triggers a corresponding event to the VB application.

This method accepts the following parameters:

clientAddress as String	This is the client e-mail address to be used as a search criterion. The e-mail messages with all states and types that match the client e-mail address will be returned by this query.

To use this method, do the following:

```
Dim clientAddress as String

clientAddress = "email65dit1-cust@trangotest.com"

Dim queryID as Long

queryID =
m_oMediaManager.QueryAsyncClientEmailsHistory(clientAddress)
```

Method:	QueryAsyncClientEmailsHistory (of MediaManager)
Description:	Enables SDK client applications to trigger the retrieval of historical e-mail messages.
Parameters:	clientAddress — The client e-mail address

Returns:	QueryID — The QueryID associated with the query that has been created.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

7.1.2.10 QueryAsyncConversationEmailsHistory

QueryAsyncClientEmailsHistory method enables SDK client applications to query e-mail messages stored in the system. This method is asynchronous. It submits the request to the E-mail Server, and sometime later, the E-mail Server replies by sending EmailsHistoryQueryResultEvent to the SDK which in turn fires up a corresponding event to the VB application.

This method accepts the following parameters:

ConversationId as String	This is the conversation ID to be used as a search criterion. This query returns e-mail messages with all states and types that match the client e-mail address.

To use this method, do the following:

```
Dim conversationId as String

conversationId = "EB10ED6430100EB10ED6430100"

Dim queryID as Long

queryID =
m_oMediaManager.QueryAsyncConversationEmailsHistory(conversation
Id)
```

Method:	QueryAsyncConversationEmailsHistory (of MediaManager)
Description:	Enables SDK client applications to trigger the retrieval of historical e-mail messages.
Parameters:	ConversationId — the conversation Id string
Returns:	QueryID - The QueryID associated with the query that has been created.
Potential Errors:	Error_Generic_QueryFailed
Notes:	None

7.1.2.11 QueryAsyncEmailsHistory

QueryAsyncEmailsHistory method enables SDK client applications to query e-mail messages stored in the system. This method is asynchronous. It submits the request to the E-mail Server, and sometime later, the E-mail Server replies by sending EmailsHistoryQueryResultEvent to the SDK which in turn fires up a corresponding event to the VB application.

This method accepts the following parameters:

from as Date to as Date	Describe the required time interval for which the e-mail messages will be returned.
criteria as ICriterionList	Indicates the required criteria for which the e-mail messages will be returned. The criteria are represented as collection of pairs of enEmailHistoryQueryCriteria enumeration value and a string.
states as IEnumList	This is the list of e-mail state This method does not accept any parameters.. Supported enCallStates This method does not accept any parameters. are: • CallState_Assigned • CallState_Deferred • CallState_Discarded • CallState_ExternallyConsulted • CallState_ExternallyTransferred • CallState_Queued • CallState_Replied
types as IEnumList	This is the list of e-mail state types. Supported enEmailTypes This method does not accept any parameters. are: • EmailType_AgentOutbound • EmailType_CustomerReply • EmailType_ExternalIncoming • EmailType_ExternalOutgoing • EmailType_Internal • EmailType_None • EmailType_Original • EmailType_Reply • EmailType_Resend

To use this method, do the following:

```
frmTimeInput.Caption = "Query Async Emails History"
frmTimeInput.lblLabelText = "from"
frmTimeInput.Show vbModal, Me
Dim dFrom as Date
dFrom = m_sTime
frmTimeInput.Caption = "Query Async Emails History"
frmTimeInput.lblLabelText = "to"
frmTimeInput.Show vbModal, Me
```

```
Dim dTo as Date

dTo = m_sTime

Dim criteria as CriterionList

Set criteria = New CriterionList

criteria.Add EmailsHistoryQueryCriterion_ConversationId,
"EB10ED6430100"

Dim states as EnumList

Set states = New EnumList

states.Add CallState_Discarded

Dim types as EnumList

Set types = New EnumList

types.Add EmailType_Original

Dim queryID as Long

queryID = m_oMediaManager.QueryAsyncEmailsHistory(dFrom, dTo,
criteria, states, types)
```

Method:	QueryAsyncEmailsHistory (of MediaManager)
Description:	Enables SDK client applications to trigger the retrieval of historical e-mail messages.
Parameters:	• Criteria — Search criteria • From — The start of the time interval • States — List of e-mail states, specifies the states for which to search e-mail server database. • To — The end of the time interval • Types — List of e-mail types, specifies the types for which to search e-mail server database.
Returns:	queryID — The QueryID associated with the query that has been created.
Potential Errors	• Error_Generic_QueryFailed • Error_Media_UnknownQueryCallState
Notes	None

7.1.2.12 QueryAsyncScheduledCallbacks

QueryAsyncScheduledCallbacks method enables SDK client applications to trigger the retrieval of scheduled callbacks.

This method accepts the following parameters:

from as Date to as Date	Describes the required time interval for which the scheduled callbacks will be returned.
origin as enCallbackOrigins	Indicates the required origin type for which the scheduled callback summary information will be returned
keyList as IKeyList	This is the list of queue IDs for the selected queues to be queried. It is a read/write collection of integers (longs). This is used to pass the collection of keys that are set by the SDK client application to narrow down the query. If keyList is empty, the SDK will query for all callback queues in the system.

To use this method, do the following:

```
queryID = m_oMediaManager.QueryAsyncScheduledCallbacks(from, to,
origin, keyList)
```

Method:	QueryAsyncScheduledCallbacks
Description:	Enables SDK client applications to trigger the retrieval of scheduled callbacks.
Parameters:	• From, to — The required time interval for which the scheduled callbacks will be returned. • KeyList — The list of keys for the selected queues to be queried. • Origin — The required origin type for which the scheduled callback summary information will be returned.
Returns:	queryID — The QueryID associated with the query that has been created.
Potential Errors:	• Error_Generic_QueryFailed • Error_Generic_UnableToAllocateMemory

7.1.2.13 QueryEmailSummaries

The QueryEmailSummaries method queries a summary of existing e-mail messages in the system as an EmailSummaries collection. The query only allows e-mail messages in the following states to be queried: Connected, Deferred, ExternallyConsulted, Queued, and Assigned. Multiple states can be queried through a single query. The query can run for one user or for all users.

This method accepts the following parameters:

pEnumList as EnumList	This is a collection of enCallStates enumeration This method does not accept any parameters. that SDK Application can request to be queried. It represents the e-mail states that need to be queried. The following enCallStates enumeration This method does not accept any parameters. are considered valid for this query: ● CallState_Assigned ● CallState_Connected ● CallState_Deferred ● CallState_ExternallyConsulted ● CallState_Queued
AgentKey as Long	This value would be either the AgentKey of the user for which the e-mail messages are to be queried, or 0 (to query e-mail messages for all users).

This method returns an EmailSummaries collection object.

To use this method, do the following after creating the MediaManager object:

Example to query connected and deferred e-mail messages for a user (AgentKey = 10)

```
Dim List as EnumList

List.Add CallState_Connected

List.Add CallState_Deferred

Dim objEmailSummaries as EmailSummaries

Set objEmailSummaries =
objMediaManager.QueryEmailSummaries(List, 10)
```

Example to query all e-mail messages in queued state in the system

```
Dim List as EnumList

List.Add CallState_Queued

Dim objEmailSummaries as EmailSummaries
```

```
Set objEmailSummaries =
objMediaManager.QueryEmailSummaries(List, 0)
```

Method:	QueryEmailSummaries (of MediaManager)
Description:	Queries e-mail summaries.
Parameters:	None
Returns:	EmailSummaries — The e-mail summaries queried from the E-mail Server.
Potential Errors:	Error_Media_UnknownQueryCallState
Notes:	None

7.1.2.14 Update

```
Update (MediaBase as MediaBase)
```

To update an appropriate object through the MediaManager you must pass the object into the Update method. The object being passed in must support the RoutingBase interface. If you are updating a user, ensure that the AgentKey has been set. If you are updating a call of any sort, ensure that the CallID has been set.

This method accepts the following parameters:

MediaBase as MediaBase	This is the MediaBase object that you want to update.

This method does not return any new value, but does update the object passed into the method with the updated object data. In most instances, this should be the exact same properties that you have passed in, but it is possible that some properties may have changed (for example, State). Update returns an updated object.

To use this method, do the following after creating the MediaManager:

```
oVoiceCall.CallID = "1234"
Call g_oMediaManager.Query (oVoiceCall)
oVoiceCall.Description = "New Description"
Call g_oMediaManager.Update (oVoiceCall)
```

CallbackCall object can now be updated on the Callback Server using the existing Update method.

To use this method, do the following after updating an existing queried CallbackCall:

```
Call oMediaManager.Update(oCallbackCall)
```

Method:	Update (of MediaManager)
Description:	Updates an object through the MediaManager.
Parameters:	MediaBase — An object that supports the event base interface that you want to update.
Returns:	None
Potential Errors:	• Error_Generic_UnableToReadObjectInformation • Error_Generic_UnableToWriteObjectInformation • Error_Generic_UnknownObjectType • Error_Generic_UpdateFailed • Error_Generic_UpdateNotSupported • Error_HiPathProCenter_CouldNotFindTelephonyServerAddress
Notes:	The object being passed in must support the MediaBase interface. The object being passed will update the server and return fully updated if there have been other changes in the object.

7.1.3 Events

The following events are exposed through the MediaManager interface.

For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.1.3.1 EventOccurred

```
g_oMediaManager_EventOccurred (MediaEvent as MediaEvent)
```

The EventOccurred event handler returns all events that occur in the MediaManager object. All events returned through this event interface support the MediaEvent interface. For more information, see Section 7.32, “MediaEvent”, on page 426.

This method returns the following parameter:

MediaEvent as HiPathProCenterLibrary.MediaEvent	This is the event that has occurred in the MediaManager. This returns the interface pointer to the base MediaEvent interface. You can take this object and get the full object interface pointer.

To use this method, do the following after creating the MediaManager:

```
Private Sub g_oMediaManager_EventOccurred
   (MediaEvent as MediaEvent)
   Dim oDisconnectedEvent as
   HiPathProCenterLibrary.DisconnectedEvent
   if MediEvent.ObjectType =
      MediaEventObjectType_Disconnected Then
      ' Here you get the DisconnectedEvent interface from
      ' the MediaEvent interface you received in the Event,
      ' and then check the Call ID property.
      Set oDisconnectedEvent = MediaEvent
      MsgBox "Call Completed—CallID: " &
      oDisconnectedEvent.CallID
   End If
End Sub
```

Event Handler:	EventOccurred (of MediaManager)
Description:	Returns all events that occur in the MediaManager.
Parameters:	MediaEvent — the MediaEvent interface for the event that has occurred.
Returns:	None
Potential Errors:	Error_Generic_QueryFailed
Notes:	The event that is returned in this method can be 'recast' to other interfaces that are supported by the object. See Section 7.32, “MediaEvent”, on page 426.

7.1.3.2 ListenForEvents

```
ListenForEvents (MediaEventType as enMediaEventTypes, (Optional)
Resource as String = "*")
```

The MediaManager has a variety of events that you can request to listen for. To do this, you must make a call to ListenForEvents with the event type and the resource you want to listen for the event on. The MediaManager will then tell the servers that it wants to know when these events occur, and will wait for them. When they occur, the MediaManager will receive the events and pass them back in the EventOccurred event handler.

This function can only be called if the MediaManager was created with events.

This method accepts the following parameters:

MediaEventType as enMediaEventTypes	This is the event type that you want to listen for events on. It's important to note that event types usually refer to more than one specific event in the MediaManager. See Section 3.32, "enMediaEventTypes", on page 98.
Resource as String	This is the resource that you want to listen for the event on. The resource is specific for the event type being listened for. For a list of what the resource represents for the various MediaEventTypes, see Section 3.32, "enMediaEventTypes", on page 98.

This method does not return anything.

To use this method, do the following after creating the MediaManager:

```
Call g_oMediaManager.ListenForEvents
(MediaEventType_VoiceEvents)
```

or

```
Call g_oMediaManager.ListenForEvents
(MediaEventType_VoiceEvents, "*")
```

Although both of the preceding examples will do the same thing, the difference between the two is that in one you explicitly state the resource you want to listen for the event on. The resource parameter for this is optional, and if you do not pass it in, it will default to '*' (or all events). For more information, see Section 2.8, "About events", on page 37.

Method:	ListenForEvents (of MediaManager)
Description:	Listens for events from the MediaManager.
Parameters:	• MediaEventType — the event group you want to listen for. • Resource("*") — the resource that you want to listen for the events on.

Returns:	None
Potential Errors:	• Error_Generic_ListeningForEventsWhenIgnoringEvents • Error_Generic_UnknownEventType
Notes:	This function can only be called if the MediaManager was created with events.

7.1.3.3 StopListeningForEvents

```
StopListeningForEvents (MediaEventType as
enMediaEventTypes,(Optional) Resource as String = "*")
```

The MediaManager has a variety of events that you can request to listen for. There may come a time though when you no longer want to listen for these events. To stop listening for these events, call the StopListeningForEvents method. To do this, you must make a call to StopListeningForEvents with the event type and the resource you previously Listened for the event on. You must use the exact same resource that you used when calling ListenForEvents. The MediaManager will then tell the servers that it no longer wants to know when these events occur, and is no longer listening for them.

This function can only be called if the MediaManager was created with events.

This method accepts the following parameters:

MediaEvent Type as enMediaEvent Types	This is the event type that you want to stop listening for events on. It's important to note that event types usually refer to more than one specific event in the MediaManager. See Section 3.32, "enMediaEventTypes", on page 98.
Resource as String	This is the resource that you want to stop listening for the event on. The resource is specific for the event type you are stopping listening for. This must match the exact resource that you used when you issued the ListenForEvent request. For a list of what the resource represents for the various MediaEventTypes, see Section 3.32, "enMediaEventTypes", on page 98.

This method does not return anything.

To use this method, do the following after creating the MediaManager:

```
Call g_oMediaManager.StopListeningForEvents
(MediaEventType_VoiceEvents)
```

or

```
Call g_oMediaManager.StopListeningForEvents
(MediaEventType_VoiceEvents, "*")
```

Although both of the preceding examples will do the same thing, the difference between the two is that in one you explicitly state the resource you want to stop listening for the event on. The resource parameter for this is optional, and if you do not pass it in, it will default to '*' (or all events).

It is important that you use the same resource that you used when you call ListenForEvents. For example, if you registered for the event with '1234' you must use that to deregister for the event. Although '*' can be used for all events when you listen and stop listening for events, when you stop listening for events, '*' only refers to stop listening for events that you listened for with '*'. For more information, see Section 2.8, “About events”, on page 37.

Method:	StopListeningForEvents (of MediaManager)
Description:	Stops listening for events from the MediaManager.
Parameters:	• MediaEventType — the event group you want to stop listening for events for. • Resource("*") — the resource that you want to listen for the events on.
Returns:	None
Potential Errors:	• Error_Generic_StopListeningForEventsWhenIgnoringEvents • Error_Generic_UnknownEventType
Notes:	This function can only be called if the MediaManager was created with events. The resource must be the exact same as the resource you used when you called ListenForEvents.

7.2 Agent

Use the Agent object to handle routing state changes in the system. This includes logging a user on and off of the system, as well as changing their routing state.

When users are Available, they are available for all media types for which they are logged on. For example, if a user is logged on to voice, then logs on to e-mail, and then selects Available state, the user is available to handle both voice and e-mail contacts. If they no longer want to be eligible for calls, they must log off voice.

Interfaces supported:

- IAgent5 (default)
- IBinaryAccess
- IUserHandlingState2
- IUserHandlingStates2

- IMediaBase
- IXMLAccess

7.2.1 Properties

This section contains properties exposed through the Agent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.2.1.1 Extension

Use the Extension property to store the extension that the user is logged on to.

To use this property, do the following:

```
Dim strExtension as String
strExtension = oAgent.Extension
```

Property:	Extension (of Agent)
Description:	The extension that the user is logged on to.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.2.1.2 ForwardingState

The ForwardingState property specifies whether call forwarding from the user extension to the associated preferred device is active or inactive.

NOTE: This property is supported only when the system is connected to an OpenScape Voice V7 R1 or V8 communication platform.

To use this property, do the following:

```
Dim bForwardingState as Boolean
bForwardingState = oAgent.ForwardingState
```

Property:	ForwardingState (of Agent)
Description:	Specifies whether call forwarding from the user extension to the associated preferred device is active or inactive.

Type:	Boolean
Potential Errors:	Error_Media_OperationNotSupportedOnCommunicationPlatform
Notes:	None

7.2.1.3 HandlingStates

The HandlingStates property queries the UserHandlingStates interface and returns a read-only collection of HandlingState objects that contains all the handling states defined in the system. This property does not accept any parameters.

To use this property, do the following:

```
Dim oHandlingStates as HiPathProCenterLibrary.UserHandlingStates
set oHandlingStates = oAgent.HandlingStates
```

Property:	HandlingStates (of Agent)
Description:	The state of a user while handling a contact.
Type:	UserHandlingStates
Potential Errors:	Error_Generic_UnknownState
Notes:	Read-only

7.2.1.4 ID

Use the ID property to store the ID that the user will use to log on. This is a global unique string (in the system) that is defined by the system administration, and used by the user for identification purposes.

To use this property, do the following:

```
Dim strID as String
strID = oAgent.ID
```

Property:	ID (of Agent)
Description:	The ID of the user.
Type:	String
Potential Errors:	None
Notes:	• Read-only • While this is similar to the Key property (both are unique ways to identify the user), the ID is user defined, whereas the Key is system defined. In most of the SDK, Key is the preferred way of interacting with the objects, although in most cases, the customer will use the ID to originally identify the user.

7.2.1.5 IsMandatoryWrapupRequired

The IsMandatoryWrapupRequired property sets the mandatory wrap-up flag on the Agent object.

To use this property, do the following:

```
Dim bMandatoryWrapupRequired as Boolean
bMandatoryWrapupRequired = oAgent.IsMandatoryWrapupRequired
```

Property:	IsMandatoryWrapupRequired (of Agent)
Description:	This property sets the mandatory wrap-up flag on the Agent object.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.2.1.6 Key

The Key property is the unique key for the user in the database. This is a system generated value.

To use this property, do the following:

```
Dim lKey as Long
lKey = oAgent.Key
```

Property:	Key (of Agent)
Description:	The database table unique key for the User object.
Type:	Long
Potential Errors:	None
Notes:	While this is similar to the ID property (both are unique ways to identify the user), the Key is system defined, whereas the ID is user defined. In most of the SDK, Key is the preferred way of interacting with the objects, although in most cases, the customer will use the ID to originally identify the user.

7.2.1.7 LoggedOnForMediaType

```
bLoggedOn = LoggedOnForMediaType (enMediaTypes as
HiPathProCenterLibrary.MediaTypes)
```

Use the LoggedOnForMediaType property to return whether the user who is logged on, is logged on for the media type specified. If the user is logged on for the specified media type, TRUE is returned. If the user is not logged on, FALSE is returned.

This property accepts the following parameter:

enMediaType as enMediaTypes	This is the media types that you want to see if the user is logged on for. For example, if you were searching for the voice media type, you would pass MediaType_Voice.

This property returns the following parameter:

bLoggedOn as Boolean	This is a boolean value that represents if the user is logged on for the media type specified. If the user is logged on for the specified media type, TRUE is returned. If the user is not logged on, FALSE is returned.

To use this property, do the following:

```
Dim bLoggedOn as Boolean
bLoggedOn = oAgent.LoggedOnForMediaType (MediaType_Voice)
```

This checks to see if the user is logged o for the MediaType_Voice media type. If they are, LoggedOnForMediaType returns TRUE. If they are not logged on for the voice media type, this returns FALSE.

Property:	LoggedOnForMediaType (of Agent)
Description:	Checks if a user is logged on for a media type.
Parameters:	enMediaType — the media type you want to check to see if the user is logged on for.
Returns:	bLoggedOn — a boolean value representing if the user is logged on for the media type specified (TRUE) or not (FALSE).
Potential Errors:	Error_Media_AgentInvalidMediaTypeForAction
Notes:	None

7.2.1.8 PreferredDevice

The PreferredDevice property specifies the preferred device of a user.

NOTE: This property is supported only when the system is connected to an OpenScape Voice V7 R1 or V8 communication platform.

To use this property, do the following:

```
Dim strPreferredDevice as String
strPreferredDevice = oAgent.PreferredDevice
```

Property:	PreferredDevice (of Agent)
Description:	The preferred device of a user.
Type:	String
Potential Errors:	Error_Media_OperationNotSupportedOnCommunicationPlatform
Notes:	None

7.2.1.9 PresenceState

The PresenceState property specifies the presence state of the user. This is one of the valid enumeration This method does not accept any parameters. from the enPresenceStates enumeration.

To use this property, do the following:

```
Dim enPresenceState as HiPathProCenterLibrary.enPresenceStates
enPresenceState = oAgent.PresenceState
```

Property:	PresenceState (of Agent)
Description:	The presence state of the user.
Type:	enPresenceStates
Potential Errors:	Error_Generic_UnknownState
Notes:	Read-only

7.2.1.10 RoutingState

The RoutingState property specifies the routing state of the user. This is one of the valid enumeration This method does not accept any parameters. from the enRoutingStates enumeration.

To use this property, do the following:

```
Dim enRoutingState as HiPathProCenterLibrary.enRoutingStates
enRoutingState = oAgent.RoutingState
```

Property:	RoutingState (of Agent)
Description:	The routing state of the user.
Type:	enRoutingStates
Potential Errors:	Error_Generic_UnknownState
Notes:	Read-only

7.2.1.11 RoutingStateReasonKey

The RoutingStateReasonKey property returns Unavailable or Work reason keys.

To use this property, do the following:

```
Dim lKey as Integer
lKey = oAgent.RoutingStateReasonKey
```

Property:	RoutingStateReasonKey (of Agent)
Description:	This property returns Unavailable or Work reason key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.2.2 Methods

This section contains methods exposed through the Agent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.2.2.1 AddCapability

```
oAgent.AddCapability ()
```

Use the AddCapability method to add a capability to a user.

NOTE: The only capability that is currently supported is "Campaign Director".

This method accepts the following parameters:

enCapabilityType	This is an enumerated value that sets the capability of the user.
Parameters	This is a collection of key/value pairs that provide information about a capability.

To use this method, do the following:

```
Dim oParameters as HiPathProCenterLibrary.Parameters
Set oContactData = new HiPathProCenterLibrary.Parameters
```

For example:

```
Dim oKVPair As HiPathProCenterLibrary.KeyValuePair
Set oKVPair = New HiPathProCenterLibrary.KeyValuePair
```

For the "Campaign Director" capability the object contains two key/value pairs. The first key is the number used by Campaign Director to set up a connection for a blended agent for an outbound call. The second key is optional and specifies a value indicating the reason that the user's routing state changed to Work.

Method:	AddCapability (of Agent)
Description:	Adds a capability to a user.
Parameters:	enCapabilityType — This is an enumerated value that sets the capability of the user Parameters — This is a collection of key/value pairs associated with the capability.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_UnableToDetermineState • Error_Media_PresenceAddCapabilityException • Error_Media_PresenceCapabilityHandlingNailUpCall • Error_Media_PresenceCapabilityNotSupported • Error_Media_PresenceInvalidState • Error_Media_PresenceRequestFailed
Notes:	None

7.2.2.2 GetVoicePreferredDevice

```
oAgent.GetVoicePreferredDevice ()
```

Use the GetVoicePreferredDevice method to obtain the preferred device and determine whether call forwarding from the user extension to the preferred device is active or inactive. This information is stored in the PreferredDevice and ForwardingState properties respectively.

NOTE: This method is supported only when the system is connected to an OpenScape Voice V7 R1 or V8 communication platform.

This method does not accept any parameters.

Method:	GetVoicePreferredDevice (of Agent)
Description:	Obtains the preferred device and determines whether call forwarding from the user extension to the preferred device is active or inactive.
Parameters:	None

Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Media_ActionResultResourceError • Error_Media_ActionResultTelephonyServerNotReady • Error_Media_OperationNotSupportedOnCommunicationPlatform
Notes:	None

7.2.2.3 Logoff

```
Logoff (MediaType as enMediaTypes)
```

Use the Logoff method to log a user off of a specific media type. After a user is logged off a specific media type, they are no longer eligible to take those types of calls.

This method accepts the following parameter:

MediaType as enMediaTypes	This is the media type for which you are logging the user off.

This method does not return anything.

To use this method, do the following after creating the user:

```
Call oAgent.Logoff (MediaType_Voice)
```

This logs the user off for the MediaType_Voice media type.

Method:	Logoff (of Agent)
Description:	Logs a user off for a specific media type of calls.
Parameters:	MediaType — the media type that you want to log the user off for.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

7.2.2.4 Logon

```
Logon (MediaType as enMediaTypes,(Optional) Extension as String
= "", (Optional) PreferredDevice as String)
```

Use the Logon method to log a user on for a specific media type. If the media type is voice, you must set the extension as well or an error will be raised. A user must be logged on for a specific media type if they want to be considered to be eligible for a call of that media type. For example, if a user wants to be eligible for e-mail calls, they must logon for the MediaType_Email media type. When the user then enters the Available state, they are now eligible to take e-mail calls (as well as any other media types that they are logged on for).

This method accepts the following parameters:

MediaType as enMediaTypes	The media type that you are logging the user on for.
Extension as String	The extension that is associated with the user for voice calls.
PreferredDevice as String	The preferred device of the user. This parameter is supported only when the system is connected to an OpenScape Voice V7 R1 or V8 communication platform.

This method does not return anything.

To use this method, do the following after creating the user:

```
Call oAgent.Logon (MediaType_Voice, txtExtension.Text,
txtPreferredDevice.Text)
```

This logs the user on to the voice media type with the specified extension and sets the specified preferred device.

Method:	Logon (of Agent)
Description:	Logs a user on for a specific media type of calls.
Parameters:	• MediaType — the media type you want to log the user on for. • Extension("") — the extension for voice calls that the user should be logged onto. • PreferredDevice(“”) — The preferred device of the user.

Returns: None

Potential Errors:
- Error_Generic_ActionFailed
- Error_Generic_NetworkBusy
- Error_Generic_RequestTimedOut
- Error_Generic_ValidationFailed
- Error_Media_ActionResultCSTAONSNotEnabled
- Error_Media_ActionResultInvalidCSTADeviceID
- Error_Media_ActionResultInvalidExtension
- Error_Media_ActionResultInvalidForwardingDestination
- Error_Media_ActionResultLineBusy
- Error_Media_ActionResultOutOfService
- Error_Media_ActionResultResourceBusy
- Error_Media_ActionResultResourceError
- Error_Media_ActionResultStateError
- Error_Media_ActionResultTelephonyServerNotReady

Notes: None

7.2.2.5 RemoveCapability

```
oAgent.RemoveCapability ()
```

Use the RemoveCapability method to remove a capability from a user.

NOTE: The only capability that is currently supported is "Campaign Director".

This method accepts the following parameters:

enCapabilityType	This is an enumerated value that sets the capability of the user.
Parameters	This is a collection of key/value pairs that provide information about a capability.

To use this method, do the following:

```
Dim oParameters as HiPathProCenterLibrary.Parameters
Set oContactData = new HiPathProCenterLibrary.Parameters
```

For example:

```
Dim oKVPair As HiPathProCenterLibrary.KeyValuePair
Set oKVPair = New HiPathProCenterLibrary.KeyValuePair
```

For the "Campaign Director" capability the object contains two key/value pairs. The first key is a value indicating the user's routing state (Available or Unavailable). The second key is optional and specifies a value indicating the reason that the user's routing state changed to Available or Unavailable.

Method:	RemoveCapability (of Agent)
Description:	Removes a capability from a user.
Parameters:	enCapabilityType — This is an enumerated value that sets the capability of the user Parameters — This is a collection of key/value pairs associated with the capability.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_UnableToDetermineState • Error_Media_PresenceCapabilityHandlingNailUpCall • Error_Media_PresenceCapabilityNotSupported • Error_Media_PresenceInvalidState • Error_Media_PresenceRemoveCapabilityException • Error_Media_PresenceRequestFailed
Notes:	None

7.2.2.6 SetVoicePreferredDevice

```
oAgent.SetVoicePreferredDevice ()
```

Use the SetVoicePreferredDevice method to set the preferred device and whether call forwarding from the user extension to the preferred device is active or inactive. Before using this method, the preferred device and forwarding state must be set in the PreferredDevice and ForwardingState properties respectively.

NOTE: This method is supported only when the system is connected to an OpenScape Voice V7 R1 or V8 communication platform.

This method does not accept any parameters.

Method:	SetVoicePreferredDevice (of Agent)
Description:	Sets the preferred device and whether call forwarding from the user extension to the preferred device is active or inactive.
Parameters:	None
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Media_ActionResultCSTAONSNotEnabled • Error_Media_ActionResultInvalidCSTADeviceID • Error_Media_ActionResultInvalidForwardingDestination • Error_Media_ActionResultResourceError • Error_Media_ActionResultTelephonyServerNotReady • Error_Media_OperationNotSupportedOnCommunicationPlatform
Notes:	None

7.2.2.7 SetRoutingState

```
SetRoutingState (enRoutingStates, (Optional)
RoutingStateReasonKey as Long = 0)
```

Use the SetRoutingState method to set the routing state of the user.

This method accepts the following parameters:

enRoutingStates	Sets the routing state of the user.
RoutingStateReasonKey as Long	This is the routing state reason for the user going into Unavailable or Work state.

This method does not return anything.

To use this method, do the following:

```
Call oAgent.SetRoutingState (RoutingState_Available)
```

This sets the user's routing state to Available for all media types that they are logged on for.

Method:	SetRoutingState (of Agent)
Description:	Sets the routing state of the user for all media types for which they are logged on.
Parameters:	ReasonKey(0) — the routing state reason for the user going into Unavailable or Work state.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

7.3 AgentStatusEvent

The AgentStatusEvent object is sent whenever there is a change in the state of a user.

Interfaces supported:

- IAgentStatusEvent4 (default)
- IMediaEvent
- IXMLAccess

7.3.1 Properties

This section contains properties exposed through the AgentStatusEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.3.1.1 AgentID

Use the AgentID property to store the ID that the user will use to log on. This is a global unique string (in the system) that is defined by the system administration, and used by the user for identification purposes.

To use this property, do the following:

```
Dim strAgentID as String
strAgentID = oAgentStatusEvent.AgentID
```

Property:	AgentID (of AgentStatusEvent)
Description:	The AgentID of the user.
Type:	String
Potential Errors:	None
Notes:	• Read-only. • While this is similar to the Key property (both are unique ways to identify the user), the ID is user defined, whereas the Key is system defined. In most of the SDK, Key is the preferred way of interacting with the objects, although in most cases, the customer will use the ID to originally identify the user.

7.3.1.2 AgentKey

The AgentKey property is the key for the user in the database. This is a system generated value.

To use this property, do the following:

```
Dim lKey as Long
lKey = oAgentStatusEvent.AgentKey
```

Property:	AgentKey (of AgentStatusEvent)
Description:	The database table unique key for the Agent object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.3.1.3 Extension

Use the Extension property to store the extension that the user is logged on to.

To use this property, do the following:

```
Dim strExtension as String
strExtension = oAgentStatusEvent.Extension
```

Property:	Extension (of AgentStatusEvent)
Description:	The extension that the user is logged on to.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.3.1.4 HandlingStates

The HandlingStates property queries the UserEventHandlingStates interface and returns a read-only collection of the user's HandlingState objects. This property does not accept any parameters.

To use this property, do the following:

```
Dim oHandlingStates as
HiPathProCenterLibrary.UserEventHandlingStates
Set oHandlingStates = oAgentStatusEvent.HandlingStates
```

Property:	HandlingStates (of AgentStatusEvent)
Description:	The state of a user while handling a contact.
Type:	UserEventHandlingStates
Potential Errors:	Error_Generic_UnknownState
Notes:	Read-only

7.3.1.5 LoggedOnMediaTypes

The LoggedOnMediaTypes property returns a list of the media types that the user is logged on to.

To use this property, do the following:

```
Dim oMediaTypes as HiPathProCenterLibrary.mediaTypes
Set oMediaTypes = oAgentStatusEvent.LoggedOnMediaTypes
```

Property:	LoggedOnMediaTypes(of AgentStatusEvent)
Description:	The media types that the user is logged on to
Type:	MediaTypes
Potential Errors:	None
Notes:	Read-only

7.3.1.6 PresenceState

The PresenceState property specifies the presence state of the user. This is one of the valid enumeration This method does not accept any parameters. from the enPresenceStates enumeration.

To use this property, do the following:

```
Dim enPresenceState as HiPathProCenterLibrary.enPresenceStates
enPresenceState = oAgentStatusEvent.PresenceState
```

Property:	PresenceState (of AgentStatusEvent)
Description:	The presence state of the user.
Type:	enPresenceStates
Potential Errors:	None
Notes:	Read-only

7.3.1.7 RoutingState

The RoutingState property specifies the routing state of the user. This is one of the valid enumeration This method does not accept any parameters. from the enRoutingStates enumeration.

To use this property, do the following:

```
Dim enRoutingState as HiPathProCenterLibrary.enRoutingStates
enRoutingState = oAgentStatusEvent.RoutingState
```

Property:	RoutingState (of AgentStatusEvent)
Description:	The routing state of the user.
Type:	enRoutingStates
Potential Errors:	None
Notes:	Read-only

7.3.1.8 RoutingStateReasonKey

The RoutingStateReasonKey property is the reason key in the database for the user going into Unavailable/Work state. This is a system generated value.

To use this property, do the following:

```
Dim lRoutingStateReasonKey as Long
lRoutingStateReasonKey = oAgentStatusEvent.RoutingStateReasonKey
```

Property:	RoutingStateReasonKey (of AgentStatusEvent)
Description:	The reason key for the user going into Unavailable/Work state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.4 AutoAcknowledgedEvent

The AutoAcknowledgedEvent object is sent when an auto-acknowledged e-mail message is sent. The auto-acknowledge message is a message from the e-mail server letting the customer know that their e-mail message has been received and usually contains information as to when they can expect a reply. The AutoAcknowledgedEvent is used by the e-mail media type.

Interfaces supported:

IAutoAcknowledgedEvent3 (default)

IMediaEvent

IXMLAccess

7.4.1 Properties

This section contains properties exposed through the AutoAcknowledgedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.4.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oAutoAcknowledgedEvent.CallID
```

Property:	CallID (of AutoAcknowledgedEvent)
Description:	.The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.4.1.2 ConversationID

The ConversationID property determines which interactions are associated with the same e-mail message. Conversation IDs are used only for e-mail messages. In an e-mail message, there may be various messages sent back and forth between the customer and the user. Each of these e-mail messages will have a different CallID, but each e-mail message that is associated with the same e-mail thread will have the same conversation ID.

To use this property, do the following:

```
Dim strConversationID as String
strConversationID = oAutoAcknowledgedEvent.ConversationID
```

Property:	ConversationID (of AutoAcknowledgedEvent)
Description:	The conversation ID used to determine which interactions are associated with the same call.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.4.1.3 OriginalFromParty

The OriginalFromParty property is the original calling party that placed the call. For example: for voice, this is the ANI; for e-mail, this is the From e-mail address; for Web collaboration, this is the customer's IP address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oAutoAcknowledgedEvent.OriginalFromParty
```

Property:	OriginalFromParty (of AutoAcknowledgedEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Notes:	Read-only

7.4.1.4 OriginalToParty

The OriginalToParty property is the original destination of the call. For example: for a voice call, this is the DNIS; for an e-mail message, this is the To e-mail address; for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oAutoAcknowledgedEvent.OriginalToParty
```

Property:	OriginalToParty (of AutoAcknowledgedEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Notes:	Read-only

7.4.1.5 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oAutoAcknowledgedEvent.SequenceNumber
```

Property:	SequenceNumber (of AutoAcknowledgedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.5 CallbackCall

The CalbackCall object represents a callback in the system. This creates and deletes callbacks.

Interfaces supported:

- ICallbackCall5 (default)
- IBinaryAccess
- IMediaBase
- IXMLAccess

7.5.1 Properties

This section contains properties exposed through the CallbackCall interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.5.1.1 AgentKey

The AgentKey property is the key for the user assigned to the callback. This is a system generated value.

To use this property, do the following:

```
Dim lKey as Long
lKey = oCallbackCall.AgentKey
```

Property:	AgentKey (of CallbackCall)
Description:	The unique key for the Agent object.
Type:	Long
Potential Errors:	None
Notes:	None

7.5.1.2 CallbackSchedules

The CallbackSchedules property is a collection of CallbackSchedule objects for that callbackcall. It contains information about the various times and numbers including schedule status that should be used when performing the callback. For more information, see Section 7.8, “CallbackSchedule”, on page 323.

To use this property, do the following:

```
Dim oSchedules as HiPathProCenterLibrary.CallbackSchedules
Set oSchedules = oCallbackCall.CallbackSchedules
```

Property:	CallbackSchedules (of CallbackCall)
Description:	A collection of the various times and numbers to be used for performing a callback.
Type:	CallbackSchedules
Potential Errors:	Error_Generic_CreatingObject
Notes:	Read-only

7.5.1.3 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oCallbackCall.CallID
```

Property:	CallID (of CallbackCall)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Notes:	This property should be set before querying a CallbackCall from the MediaManager object.

7.5.1.4 ContactData

The ContactData property is the callback data that is associated with this CallbackCall object. The ContactData property can provide the most information about a call. The ContactData object is a collection of key/value pairs associated with the Call. This can be used to hold information entered in the IVR, or information from other agents that have talked to this customer. For more information, see Section 4.1, “ContactData”, on page 117. This read/write property is also used so that the Contact Data for this contact can be updated.

To use this property, do the following:

```
Dim oContactData as HiPathProCenterLibrary.ContactData
Set oContactData = oCallbackCall.ContactData
```

Property:	ContactData (of CallbackCall)
Description:	The contact data that is associated with this contact in the system.
Type:	ContactData
Potential Errors:	Error_Generic_CreatingObject
Notes:	Read-only

7.5.1.5 CreatorAgentKey

The CreatorAgentKey property is the key for the user who created the callback. This is a system generated value.

To use this property, do the following:

```
Dim lKey as Long
lKey = oCallbackCall.CreatorAgentKey
```

Property:	CreatorAgentKey (of CallbackCall)
Description:	The unique key for the Agent object.
Type:	Long
Potential Errors:	None
Notes:	None

7.5.1.6 CustomerName

The CustomerName property is the name of the person who you want to call back. This should be set when creating the callback so the user performing the callback will know who it is that they are calling.

To use this property, do the following:

```
Dim strCustomerName as String
strCustomerName = oCallbackCall.CustomerName
```

Property:	CustomerName (of CallbackCall)
Description:	The name of the person that you want to call back.
Type:	String
Potential Errors:	None
Notes:	None

7.5.1.7 Description

The Description property is the description that is associated with this CallbackCall object. The description is a single line of text that can be displayed in the Screen Pop portion of the Client Desktop application. This can be used to provide some information about the queue that the queue itself may not be able to do. For example, if the queue is 'Sales', the description might be 'Sales Call for All Wood Furniture'. This read/write property is also used so that the description for this contact can be updated.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oCallbackCall. Description
```

Property:	Description (of CallbackCall)
Description:	The description that is associated with this contact in the system.
Type:	String
Potential Errors:	None
Notes:	None

7.5.1.8 Priority

The Priority property is the priority that is associated with this CallbackCall object. The priority determines how this contact is scored compared to other contacts. The higher the priority of the contact relative to other contacts, the higher the importance of the contact. This read/write property is also used so that the priority for this contact can be updated.

To use this property, do the following:

```
Dim lPriority as Long
lPriority = oCallbackCall.Priority
```

Property:	Priority (of CallbackCall)
Description:	The priority that is associated with the contactin the system.
Type:	Long
Potential Errors:	None
Notes:	None

7.5.1.9 QueueKey

The Queuekey property is the key for the queue in the database. This is a system generated value.

To use this property, do the following:

```
Dim lKey as Long
lKey = oCallbackCall.Queuekey
```

Property:	Queuekey (of CallbackCall)
Description:	The database table unique key for the queue object.
Type:	Long
Potential Errors:	None
Notes:	None

7.5.1.10 Result

The Result property returns a callback Delete or Retry reason key.

To use this property, do the following:

```
Dim lKey as Integer
lKey = oCallbackCall.Result
```

Property:	Result (of CallbackCall)
Description:	The Result property returns a callback Delete or Retry reason key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.5.1.11 RetryDelay

The RetryDelay property returns a callback retry delay.

To use this property, do the following:

```
Dim lDelay as Integer
lDelay = oCallbackCall.RetryDelay
```

Property:	RetryDelay (of CallbackCall)
Description:	The RetryDelay property returns a callback retry delay.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.5.1.12 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oCallbackCall.SequenceNumber
```

Property:	SequenceNumber (of CallbackCall)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.5.1.13 State

The State property is the state that the contact is in. This is one of the valid enumeration This method does not accept any parameters. from the enCallStates enumeration.

To use this property, do the following:

```
Dim enCallState as HiPathProCenterLibrary.enCallStates
enCallState = oCallbackCall.State
```

Property:	State (of CallbackCall)
Description:	The state of the call.
Type:	EnCallStates
Potential Errors:	None
Notes:	Read-only

7.5.1.14 WaitTime

The WaitTime property is the amount of time in seconds that the contact was queued prior to being delivered to the agent.

To use this property, do the following:

```
Dim lWaitTime as Long
lWaitTime = oEmailCall.WaitTime
```

Property:	WaitTime (of CallbackCall)
Description:	The amount of time in seconds the contact was queued prior to being delivered to the agent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.5.2 Methods

This section contains methods exposed through the CallbackCall interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.5.2.1 Accept

```
Accept()
```

The Accept method accepts a callback offered to a user. This is required when a callback is offered to a user, so that the Callback Server knows that the user will handle the callback.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following after creating the CallbackCall:

```
Call oCallbackCall.Accept
```

This accepts the callback using the parameters set in the CallbackCall object.

Method:	Accept (of CallbackCall)
Description:	Accepts a callback offered to a user.
Parameters:	None

Returns: None

Potential Errors:
- Error_Generic_ActionFailed
- Error_Generic_NetworkBusy
- Error_Generic_RequestTimedOut
- Error_Generic_WrongVersion
- Error_Media_CallbackActionResultCallbackServerNotReady
- Error_Media_CallbackActionResultCallIDToolong
- Error_Media_CallbackActionResultCannotDeleteOngoingCall
- Error_Media_CallbackActionResultCannotUpdateOngoingCall
- Error_Media_CallbackActionResultContactDataTooLong
- Error_Media_CallbackActionResultCountryCodeTooLong
- Error_Media_CallbackActionResultCustomerNameTooLong
- Error_Media_CallbackActionResultDDETooLong
- Error_Media_CallbackActionResultDescriptionTooLong
- Error_Media_CallbackActionResultEmptyCallID
- Error_Media_CallbackActionResultEndTimeEarlyThanStartTime
- Error_Media_CallbackActionResultFoundExcludedNumber
- Error_Media_CallbackActionResultInvalidCallbackState
- Error_Media_CallbackActionResultInvalidPriority
- Error_Media_CallbackActionResultInvalidQueuekey
- Error_Media_CallbackActionResultInvalidScheduleTime
- Error_Media_CallbackActionResultInvalidSiteKey
- Error_Media_CallbackActionResultInvalidTelphoneNumber
- Error_Media_CallbackActionResultReceivedCorruptedData
- Error_Media_CallbackActionResultResourceError
- Error_Media_CallbackActionResultScheduleExceedMaxDays
  Ahead
- Error_Media_CallbackActionResultSchedulesExpired
- Error_Media_CallbackActionResultSchedulesHaveNoMutualTime
- Error_Media_CallbackActionResultSchedulesOverlapped

Notes: None

7.5.2.2 AddCallbackSchedule

```
AddCallbackSchedule (CallbackSchedule as CallbackSchedule)
```

This method adds callbackSchedule to CallbackSchedules collection in CallbackCall object if possible.

It will return an error if the client tries to add an expired callbackschedule or the schedule, whose addition was attempted breaks the integrity of that set of schedules.

This method accepts the following parameter:

Callback Schedule as Callback Schedule	This is the CallbackSchedule that you want to add in our CallbackSchedules collection inside the CallbackCall object.

This method does not return anything.

To use this method, do the following:

```
Dim oCallbackSchedule as HiPathProCenterLibary.CallbackSchedule
Set oCallbackSchedule = New
HiPathProCenterLibrary.CallbackSchedule
oCallbackSchedule.StartTime = txtStartTime.Text
oCallbackSchedule.EndTime = txtEndTime.Text
oCallbackSchedule.TelephoneNumber = txtTelephoneNumber.Text
Call oCallbackCall.AddCallbackSchedule (oCallbackSchedule)
```

This adds the CallbackShedule in our StartTime indexed CallbackSchedules collection inside the CallbackCall object.

Method:	AddCallbackSchedule (of CallbackCall)
Description:	Adds callbackSchedule to CallbackSchedules collection in CallbackCall object.
Parameters:	CallbackSchedule- the CallbackSchedule that you want to add in our collection.
Returns:	None
Potential Errors:	• Error_Media_Callback_AddScheduleFailed • Error_Media_Callback_CallbackScheduleExpired
Notes:	None

7.5.2.3 Complete

```
Complete()
```

The Complete method notifies CallbackServer of callback success or failure. To do this, the user should first ensure that they have set the CallID property of the CallbackCall object then call Complete.

This method accepts the following parameters:

Success as Boolean	• False — callback failed. • True — callback completed successfully
ReasonKey as Long	Retry Reason Key queried using AdministrationManager method QueryCallbackRetryReasons. This parameter is optional and defaults to a value of negative one (-1). The ReasonKey should be filled-in if the Success parameter is False. This will throw an error if Success parameter is False and reason key has a value of negative one (-1).
RetryInterval as Long	Optional. This value is in minutes. Specifying a value of negative one (-1) or leaving it at the default indicates that the default retry interval associated with the specified reason should be used. If the retry interval specified does not fall within callback Schedules, an error is thrown back to the SDK application.

This method does not return anything.

To use this method, do either one of the following after creating the CallbackCall:

```
Call oCallbackCall.Complete(True)
```

Or

```
Call oCallbackCall.Complete(False, 5)
```

Method:	Complete (of CallbackCall)
Description:	Notifies CallbackServer of callback success or failure.
Parameters:	• False — callback failed. • ReasonKey — Retry Reason Key queried using AdministrationManager method QueryCallbackRetryReasons. • RetryInterval — Retry interval in minutes. • Success (True) — callback completed successfully

Returns: None

Potential Errors:
- Error_Generic_ActionFailed
- Error_Generic_NetworkBusy
- Error_Generic_RequestTimedOut
- Error_Generic_WrongVersion
- Error_Media_CallbackActionResultCallbackCallNotFound
- Error_Media_CallbackActionResultCallbackServerNotReady
- Error_Media_CallbackActionResultCallIDToolong
- Error_Media_CallbackActionResultCannotDeleteOngoingCall
- Error_Media_CallbackActionResultCannotUpdateOngoingCall
- Error_Media_CallbackActionResultContactDataTooLong
- Error_Media_CallbackActionResultCountryCodeTooLong
- Error_Media_CallbackActionResultCustomerNameTooLong
- Error_Media_CallbackActionResultDatabaseOperationError
- Error_Media_CallbackActionResultDDETooLong
- Error_Media_CallbackActionResultDescriptionTooLong
- Error_Media_CallbackActionResultEmptyCallID
- Error_Media_CallbackActionResultEndTimeEarlyThanStartTime
- Error_Media_CallbackActionResultFoundExcludedNumber
- Error_Media_CallbackActionResultInvalidCallbackState
- Error_Media_CallbackActionResultInvalidPriority
- Error_Media_CallbackActionResultInvalidQueuekey
- Error_Media_CallbackActionResultInvalidScheduleTime
- Error_Media_CallbackActionResultInvalidSiteKey
- Error_Media_CallbackActionResultInvalidTelphoneNumber
- Error_Media_CallbackActionResultReceivedCorruptedData
- Error_Media_CallbackActionResultResourceError
- Error_Media_CallbackActionResultScheduleExceedMaxDays
  Ahead
- Error_Media_CallbackActionResultSchedulesExpired
- Error_Media_CallbackActionResultSchedulesHaveNoMutualTime
- Error_Media_CallbackActionResultSchedulesOverlapped

Notes: None

7.5.2.4 Create

```
Create()
```

The Create method creates a callback. To do this, the user should first ensure that they have set all the necessary properties of the CallbackCall object then call Create.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following after creating the CallbackCall:

```
Call oCallbackCall.Create
```

This creates the callback using the parameters set in the CallbackCall object.

Method:	Create (of CallbackCall)
Description:	Creates a callback in the system.
Parameters:	None

Returns: None

Potential Errors:
- Error_Generic_ActionFailed
- Error_Generic_NetworkBusy
- Error_Generic_RequestTimedOut
- Error_Generic_WrongVersion
- Error_Media_CallbackActionResultCallbackCallNotFound
- Error_Media_CallbackActionResultCallbackServerNotReady
- Error_Media_CallbackActionResultCallIDToolong
- Error_Media_CallbackActionResultCannotUpdateOngoingCall
- Error_Media_CallbackActionResultContactDataTooLong
- Error_Media_CallbackActionResultCountryCodeTooLong
- Error_Media_CallbackActionResultCustomerNameTooLong
- Error_Media_CallbackActionResultDatabaseOperationError
- Error_Media_CallbackActionResultDDETooLong
- Error_Media_CallbackActionResultDescriptionTooLong
- Error_Media_CallbackActionResultEmptyCallID
- Error_Media_CallbackActionResultEndTimeEarlyThanStartTime
- Error_Media_CallbackActionResultFoundExcludedNumber
- Error_Media_CallbackActionResultInvalidCallbackState
- Error_Media_CallbackActionResultInvalidPriority
- Error_Media_CallbackActionResultInvalidQueuekey
- Error_Media_CallbackActionResultInvalidScheduleTime
- Error_Media_CallbackActionResultInvalidSiteKey
- Error_Media_CallbackActionResultInvalidTelphoneNumber
- Error_Media_CallbackActionResultReceivedCorruptedData
- Error_Media_CallbackActionResultResourceError
- Error_Media_CallbackActionResultScheduleExceedMaxDayAhead
- Error_Media_CallbackActionResultSchedulesExpired
- Error_Media_CallbackActionResultSchedulesHaveNoMutualTime
- Error_Media_CallbackActionResultSchedulesOverlapped

Notes: None

7.5.2.5 Delete

The Delete method deletes a callback that has been created. To do this, the user should first ensure that they have set the CallID property of the CallbackCall object then call Delete.

This method accepts the following parameters:

ReasonKey as Long	Delete Reason Key queried using AdministrationManager method QueryCallbackDeleteReasons. This parameter is optional and defaults to a value of negative one (-1).

This method does not return anything.

To use this method, do the following after creating the CallbackCall:

```
Call oCallbackCall.Delete
```

This deletes the callback specified by the CallID property of the CallbackCall object.

Method:	Delete (of CallbackCall)
Description:	Deletes a callback in the system.
Parameters:	ReasonKey — Delete Reason Key queried using AdministrationManager method QueryCallbackDeleteReasons.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_WrongVersion • Error_Media_CallbackActionResultCallbackCallNotFound • Error_Media_CallbackActionResultCallbackServerNotReady • Error_Media_CallbackActionResultCallIDToolong • Error_Media_CallbackActionResultCannotDeleteOngoingCall • Error_Media_CallbackActionResultCannotUpdateOngoingCall • Error_Media_CallbackActionResultContactDataTooLong • Error_Media_CallbackActionResultCountryCodeTooLong • Error_Media_CallbackActionResultCustomerNameTooLong • Error_Media_CallbackActionResultDatabaseOperationError • Error_Media_CallbackActionResultDDETooLong • Error_Media_CallbackActionResultDescriptionTooLong • Error_Media_CallbackActionResultEmptyCallID • Error_Media_CallbackActionResultEndTimeEarlyThanStartTime • Error_Media_CallbackActionResultFoundExcludedNumber • Error_Media_CallbackActionResultInvalidCallbackState • Error_Media_CallbackActionResultInvalidPriority • Error_Media_CallbackActionResultInvalidQueuekey • Error_Media_CallbackActionResultSchedulesOverlapped • Error_Media_CallbackActionResultInvalidScheduleTime • Error_Media_CallbackActionResultInvalidSiteKey • Error_Media_CallbackActionResultInvalidTelphoneNumber • Error_Media_CallbackActionResultReceivedCorruptedData • Error_Media_CallbackActionResultResourceError • Error_Media_CallbackActionResultScheduleExceedMaxDays Ahead • Error_Media_CallbackActionResultSchedulesExpired • Error_Media_CallbackActionResultSchedulesHaveNoMutualTime
Notes:	None

7.5.2.6 RemoveCallbackSchedule

```
RemoveCallbackSchedule (StartTime as Date)
```

This method deletes schedule for the specified StartTime from SchedulesCollection; RemoveCallbackSchedule().

It will return an error on an attempt to remove current or past schedule.

This method accepts the following parameters:

StartTime as Date	This is the start time to be used to specify the CallbackSchedule you want to be removed from our CallbackSchedules collection inside the CallbackCall object.

This method does not return anything.

To use this method, do the following:

```
Dim oStartTime as Date
oStartTime = txtStartTime.Text
Call oCallbackCall.RemoveCallbackSchedule (oStartTime)
```

This deletes the CallbackShedule specified by oStartTime from our StartTime indexed CallbackSchedules collection inside the CallbackCall object.

Method:	RemoveCallbackSchedule (of CallbackCall)
Description:	Deletes schedule for the specified StartTime from SchedulesCollection;
Parameters:	StartTime: specifies the CallbackSchedule you want to be removed from our CallbackSchedules collection inside the CallbackCall object.
Returns:	None
Potential Errors:	Error_Media_Callback_ RemoveScheduleFailed
Notes:	None

7.5.2.7 UpdateCallbackSchedule

```
UpdateCallbackSchedule (StartTime as Date , oUpdatedSchedule as
CallbackSchedule)
```

This method updates the CallbackSchedules collection by assigning new This method does not accept any parameters. to the Callbackschedule record specified by StartTime.

This method accepts the following parameters:

StartTime as Date	This is the start time to be used to specify the CallbackSchedule you want to be removed from our CallbackSchedules collection inside the CallbackCall object.
oUpdatedSchedule as CallbackSchedule	This is the CallbackSchedule that contains the new This method does not accept any parameters. that you want to update in our CallbackSchedules collection inside the CallbackCall object.

This method does not return anything.

To use this method, do the following:

```
Dim oUpdatedSchedule as HiPathProCenterLibary.CallbackSchedule

Dim oStartTime as Date

Set oUpdatedSchedule = New
HiPathProCenterLibrary.CallbackSchedule

oStartTime = txtStartTime.Text

oUpdatedSchedule.StartTime = txtStartTime.Text

oUpdatedSchedule.EndTime = txtEndTime.Text

oUpdatedSchedule.TelephoneNumber = txtTelephoneNumber.Text

Call oCallbackCall.UpdateCallbackSchedule (oStartTime,

oUpdatedSchedule)
```

This updates the CallbackShedule specified by oStartTime in our StartTime indexed CallbackSchedules collection inside the CallbackCall object with the new This method does not accept any parameters. in oUpdatedSchedule.

Method:	UpdateCallbackSchedule (of CallbackCall)
Description:	Updates the CallbackSchedules collection by assigning new This method does not accept any parameters. to the Callbackschedule record specified by StartTime.
Parameters:	• StartTime — specifies the CallbackSchedule you want to be updated in our CallbackSchedules collection inside the CallbackCall object. • oUpdatedSchedule — contains the new This method does not accept any parameters. that you want to update in our CallbackSchedules collection inside the CallbackCall object.

Returns:	None
Potential Errors:	Error_Media_Callback_ UpdateScheduleFailed
Notes:	None

7.6 CallbackReason

The CallbackReason object contains information about callback Retry reasons or Delete reasons as defined in the Administration Server database.

Interfaces supported:

- ICallbackReason (default)
- IXMLAccess

7.6.1 Properties

This section contains properties exposed through the CallbackReason interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.6.1.1 DefaultRetryInterval

The DefaultRetryInterval property is the interval in minutes; the Callback Server will wait before attempting to dial to customer, after unsuccessful previous attempt.

To use this property, do the following:

```
Dim oDefaultRetryInterval as Long
oDefaultRetryInterval = oCallbackReason. DefaultRetryInterval
```

Property:	DefaultRetryInterval (of CallbackReason)
Description:	The retry interval in minutes.
Type:	Long
Potential Errors:	None
Notes:	• Read-only. • This property should only be used for callback Retry reasons. Has no context with callback Delete reasons.

7.6.1.2 Description

The Description property is the callback Retry or Delete reason description as defined in the database.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oCallbackReason. Description
```

Property:	Description (of CallbackReason)
Description:	The callback Retry or Delete reason description as defined in the database.
Type:	String
Potential Errors:	None
Notes:	None

7.6.1.3 Key

The Key property is the key for the CallbackReason in the CallbackReasons collection.

To use this property, do the following:

```
Dim lKey as Long
lKey = oCallbackReason.Key
```

Property:	Key (of CallbackReason)
Description:	The key for the CallbackReason in the CallbackReasons collection.
Type:	Long
Potential Errors:	None
Notes:	None

7.6.1.4 Name

The Name property is the callback Retry or Delete reason name as defined in the database.

To use this property, do the following:

```
Dim sName as String
sName = oCallbackReason.Name
```

Property:	Name (of CallbackReason)
Description:	The callback Retry or Delete reason name as defined in the database.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.7 CallbackReasons

The CallbackReasons object is a collection of the CallbackReason object.

Interfaces supported:

- ICallbackReasons (default)
- IXMLAccess

7.7.1 Properties

This section contains properties exposed through the CallbackReasons interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.7.1.1 _NewEnum

```
For Each oCallbackReason in colCallbackReasons
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oCallbackReason as HiPathProCenterLibrary.CallbackReason
For Each oCallbackReason in m_colCallbackReasons
   txtTextBox.Text = txtTextBox.Text & _
      "Key: " & _
      oCallbackReason.Key & _
      "Name: " & _
      oCallbackReason.Name & _
      vbCrLf
Next
```

Property:	_NewEnum (of CallbackReasons)
Description:	Allows for the 'For Each' operator on the CallbackSummaries collection.
Returns:	oCallbackSummary — For each item in the CallbackSummaries collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the CallbackSummaries collection to support the 'For Each' iterator.

7.7.1.2 Count

The Count property returns the number of CallbackReason objects that exist in the CallbackReasons collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colCallbackReasons.Count
```

Property:	Count (of CallbackReasons)
Description:	The number of CallbackReason objects that are in the CallbackReasons collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.7.1.3 Item

```
oCallbackReason = Item (ReasonKey as Long)
```

Use the Item method to find a CallbackReason object in the CallbackReasons collection that is indexed by the ReasonKey value. If an item is found in the collection which matches the ReasonKey value, the Item method returns the CallbackReason object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameter:

ReasonKey as Long	This is the Key of the CallbackReason object.

This method returns the following parameter:

CallbackReason	This is the callback Retry or Delete reason object.

To use this method, do the following:

```
Dim oCallbackReason as HiPathProCenterLibary.CallbackReason
Set oCallbackReason = colCallbackReasons.Item (lReasonKey)
```

This checks to see if the CallbackReason specified by the ReasonKey exists. If it does, oCallbackReason will contain the CallbackReason object. If it does not exist in the CallbackReasons collection, this will raise an error.

Method:	Item (of CallbackReasons)
Description:	Finds a CallbackReason object in the CallbackReasons collection that is indexed by the ReasonKey value.
Parameters:	ReasonKey — The Key of the CallbackReason object.

Returns:	CallbackReason — The callback Retry or Delete reason object.
Potential Errors:	• Error_Generic_ItemNotFound • Error_Generic_UnableToAllocateMemory
Notes:	If the CallbackReason object does not exist in our collection, an error is raised.

7.8 CallbackSchedule

The CallbackSchedule object contains information for a callback schedule. Use the callback schedule to determine when a callback should occur. It contains such information as the telephone number that should be called, as well as the start and end time for the period represented by the CallbackSchedule.

Interfaces supported:

- ICallbackSchedule3 (default)
- IXMLAccess

7.8.1 Properties

This section contains properties exposed through the CallbackSchedule interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.8.1.1 EndTime

The EndTime property is the time that this CallbackSchedule period should end.

To use this property, do the following:

```
Dim dteEndTime as Date
dteEndTime = oCallbackSchedule.EndTime
```

Property:	EndTime (of CallbackSchedule)
Description:	The time that this CallbackSchedule period should end.
Type:	Date
Potential Errors:	None
Notes:	None

7.8.1.2 StartTime

The StateTime property is the time that the CallbackSchedule period should start.

To use this property, do the following:

```
Dim dteStartTime as Date
dteStartTime = oCallbackSchedule.StartTime
```

Property:	StartTime (of CallbackSchedule)
Description:	The time that the CallbackSchedule period should start.
Type:	Date
Potential Errors:	None
Notes:	None

7.8.1.3 TelephoneNumber

The TelephoneNumber property is the telephone number that should be used to call back the customer for this CallbackSchedule period.

To use this property, do the following:

```
Dim strTelephoneNumber as String
strTelephoneNumber = oCallbackSchedule.TelephoneNumber
```

Property:	TelephoneNumber (of CallbackSchedule)
Description:	The telephone number that should be used to call back the customer for this CallbackSchedule period.
Type:	String
Potential Errors:	None
Notes:	None

7.9 CallbackSchedules

The CallbackSchedules object is a collection of the CallbackSchedule objects.

Interfaces supported:

- ICallbackSchedules2 (default)
- IXMLAccess

7.9.1 Properties

This section contains properties exposed through the CallbackSchedules interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.9.1.1 _NewEnum

```
For Each oCallbackSchedule in colCallbackSchedules
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oCallbackSchedule as HiPathProCenterLibrary.CallbackSchedule
For Each oCallbackSchedule in m_colCallbackSchedule
   txtTextBox.Text = txtTextBox.Text & _
      “Schedule Start Time: ” & _
      oCallbackSchedule.StartTime & _
      “ Telephone Number: ” & _
      oCallbackSchedule.TelephoneNumber & _
      vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the start time of all the CallbackSchedule objects in the CallbackSchedules collection, as well as the telephone number that should be used for each period.

Property:	_NewEnum (of CallbackSchedules)
Description:	Allows for the 'For Each' operator on the CallbackSchedules collection.
Parameters:	None

Returns:	oCallbackSchedule — For each item in the CallbackSchedules collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This property allows the CallbackSchedules collection to support the 'For Each' iterator.

7.9.1.2 Count

The Count property returns the number of CallbackSchedule objects that exist in the CallbackSchedules collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colCallbackSchedules.Count
```

Property:	Count (of CallbackSchedules)
Description:	The number of CallbackSchedule objects that are in the CallbackSchedules collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.9.1.3 Item

```
oCallbackSchedule = Item (ScheduleStartTime as Date
```

Use the Item method to find a CallbackSchedule object in the CallbackSchedules collection that is indexed by the ScheduleStartTime value. If an item is found in the collection at the ScheduleStartTime value, the Item method returns the CallbackSchedule object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

ScheduleStartTime as Date	This is the time that the CallbackSchedule period should start.

This method returns the following parameter:

oCallbackSchedule as CallbackSchedule	This is the CallbackSchedule object for period specified by the ScheduleStartTime value. This contains the CallbackSchedule for the ScheduleStartTime specified.

To use this method, do the following:

```
Dim oCallbackSchedule as HiPathProCenterLibary.CallbackSchedule
Set oCallbackSchedule = colCallbackSchedules.Item
(ScheduleStartTime)
```

This checks to see if the CallbackSchedule specified by the ScheduleStartTime exists. If it does, oCallbackSchedule will contain the CallbackSchedule object. If it does not exist in the CallbackSchedules collection, this will raise an error.

Method:	Item (of CallbackSchedules)
Description:	Finds a CallbackSchedule object in the CallbackSchedules collection that is indexed by the ScheduleStartTime value.
Parameters:	ScheduleStartTime — The time that the CallbackSchedule period that you are looking for should start.
Returns:	oCallbackSchedule — The callback schedule for the ScheduleStartTime specified.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the CallbackSchedule object for the user does not exist in our collection, you raise an error.

7.10 CallInformationUpdatedEvent

The CallInformationUpdatedEvent object is sent whenever the ContactData or Description has changed for a contact.

Interfaces supported:

- ICallInformationUpdatedEvent3 (default)
- IMediaEvent
- IXMLAccess

7.10.1 Properties

This section contains properties exposed through the CallInformationUpdatedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.10.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oCallInformationUpdatedEvent.CallID
```

Property:	CallID (of CallInformationUpdatedEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.10.1.2 ContactData

The ContactData property is the contact data that is associated with the contact specified by the CallID. The ContactData property can provide the most information about a contact. The ContactData object is a collection of key/value pairs associated with the contact. This can be used to hold information entered in the IVR, or information from other agents that have talked to this customer. For more information, see Section 4.1, “ContactData”, on page 117.

To use this property, do the following:

```
Dim oContactData as HiPathProCenterLibrary.ContactData
If oCallInformationUpdatedEvent. ObjectType =
MediaEventObjectType_ContactDataUpdated Then
   Set oContactData = oCallInformationUpdatedEvent.ContactData
End If
```

Property:	ContactData (of CallInformationUpdatedEvent)
Description:	The contact data that is associated with this contact in the system.
Type:	ContactData
Potential Errors:	None
Notes:	Read-only

7.10.1.3 ConversationID

The ConversationID property determines which interactions are associated with the same e-mail message. In an e-mail message, there may be various messages sent back and forth between the customer and the user. Each of these e-mail messages will have a different CallID, but each e-mail message that is associated with the same e-mail thread will have the same conversation ID.

To use this property, do the following:

```
Dim strConversationID as String
strConversationID = oCallInformationUpdatedEvent.ConversationID
```

Property:	ConversationID (of CallInformationUpdatedEvent)
Description:	The conversation ID used to determine which interactions are associated with the same call.
Type:	String
Potential Errors:	None
Supported Media:	E-mail
Notes:	Read-only

7.10.1.4 Description

The Description property is the description that is associated with the contact specified by the CallID. The description is a single line of text that can be displayed in the Screen Pop portion of the Client Desktop application. This can be used to provide some information about the queue that the queue itself may not be able to do. For example, if the queue is 'Sales' the description might be 'Sales Call for All Wood Furniture'.

To use this property, do the following:

```
Dim strDescription as String
If oCallInformationUpdatedEvent. ObjectType =
MediaEventObjectType_CallInformationUpdated Then
   strDescription = oCallInformationUpdatedEvent.Description
End If
```

Property:	Description (of CallInformationUpdatedEvent)
Description:	The description that is associated with this contact in the system.

Type:	String
Potential Errors:	None
Notes:	Read-only

7.10.1.5 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oCallInformationUpdatedEvent.SequenceNumber
```

Property:	SequenceNumber (of CallInformationUpdatedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.11 CallMeEvent

The CallMeEvent object is sent when a participant in a Web collaboration session requests that the user ask them for the form to request the user call them. When the customer enters their telephone number and submits it, the CallMeEvent is generated with the telephone number of the customer.

Interfaces supported:

- ICallMeEvent3 (default)
- IMediaEvent
- IXMLAccess

7.11.1 Properties

This section contains properties exposed through the CallMeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.11.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oCallMeEvent.CallID
```

Property:	CallID (of CallMeEvent)
Description:	The unique key for the contact in the system..
Type:	String
Potential Errors:	None
Supported Media:	Web collaboration
Notes:	Read-only

7.11.1.2 OriginalFromParty

The OriginalFromParty property is the original calling party that placed the call. For example: for a voice call, this is the ANI; for an e-mail message, this is the from e-mail address; for a Web collaboration session, this is the customer's IP address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oCallMeEvent.OriginalFromParty
```

Property:	OriginalFromParty (of CallMeEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Web collaboration (Customer IP Address)
Notes:	Read-only

7.11.1.3 OriginalToParty

The OriginalToParty property is the original destination of the call. For example: for a voice call, this is the DNIS; for an e-mail message, this is the to e-mail address; for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oCallMeEvent.OriginalToParty
```

Property:	OriginalToParty (of CallMeEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Web collaboration (Initial URL)
Notes:	Read-only

7.11.1.4 Party

The Party property is the party to which the CallMeEvent is sent (the user who should be calling the customer back, or scheduling the callback, and so on). This is provided so you can display both the message and who sent it. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oCallMeEvent.Party
```

Property:	Party (of CallMeEvent)
Description:	The party that the CallMeEvent is sent to.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Web collaboration
Notes:	Read-only

7.11.1.5 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oCallMeEvent.SequenceNumber
```

Property:	SequenceNumber (of CallMeEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.11.1.6 TelephoneNumber

The TelephoneNumber property is telephone number that the customer has submitted for the user to return their call.

To use this property, do the following:

```
Dim strTelephoneNumber as String
oParty = oCallMeEvent.TelephoneNumber
```

Property:	TelephoneNumber (of CallMeEvent)
Description:	The telephone number that the customer has submitted for the user to return their call.
Type:	String
Potential Errors:	None
Supported Media:	Web collaboration
Notes:	Read-only

7.12 ConferencedEvent

The ConferencedEvent object is sent when another user has joined a call. For Web collaboration sessions, this may or may not be preceded by a ConferenceRequestedEvent.

Interfaces supported:

- IConferencedEvent3 (default)
- IMediaEvent
- IXMLAccess

7.12.1 Properties

This section contains properties exposed through the ConferencedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.12.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oConferencedEvent.CallID
```

Property:	CallID (of ConferencedEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Web collaboration
Notes:	Read-only

7.12.1.2 InvitedParty

The InvitedParty property of the ConferenceRequestedEvent is the party that has joined the conference. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oConferencedEvent.InvitedParty
```

Property:	InvitedParty (of ConferencedEvent)
Description:	The party that has joined the conference.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Web collaboration
Notes:	Read-only

7.12.1.3 InvitingParty

The InvitingParty property of the ConferenceRequestedEvent is the party that is inviting another user to the conference. This is the party that is initiating the conference request. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oConferencedEvent.InvitingParty
```

Property:	InvitingParty (of ConferencedEvent)
Description:	The party that is inviting another user to the conference.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Web collaboration
Notes:	Read-only

7.12.1.4 OriginalFromParty

The OriginalFromParty property is the original calling party that placed the call. For example: for a voice call, this is the ANI; for an e-mail message, this is the from e-mail address; for a Web collaboration session, this is the customer's IP address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oConferencedEvent.OriginalFromParty
```

Property:	OriginalFromParty (of ConferencedEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Web collaboration (Customer IP Address)
Notes:	Read-only

7.12.1.5 OriginalToParty

The OriginalToParty property is the original destination of the call. For example: for a voice call, this is the DNIS; for an e-mail message, this is the to e-mail address; for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oConferencedEvent.OriginalToParty
```

Property:	OriginalToParty (of ConferencedEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Web collaboration (Initial URL)
Notes:	Read-only

7.12.1.6 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oConferencedEvent.SequenceNumber
```

Property:	SequenceNumber (of ConferencedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.13 ContentAddedEvent

The ContentAddedEvent object is sent whenever a participant in a Web collaboration session sends a message to the other Web collaboration participants.

Interfaces supported:

- IContentAddedEvent3 (default)
- IMediaEvent
- IXMLAccess

7.13.1 Properties

This section contains properties exposed through the ContentAddedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.13.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oContentAddedEvent.CallID
```

Property:	CallID (of ContentAddedEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Web collaboration
Notes:	Read-only

7.13.1.2 OriginalFromParty

The OriginalFromParty property is the original calling party that placed the call. For example: for a voice call, this is the ANI; for an e-mail message, this is the from e-mail address; for a Web collaboration session, this is the customer's IP address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oContentAddedEvent.OriginalFromParty
```

Property:	OriginalFromParty (of ContentAddedEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Web collaboration (Customer IP Address)
Notes:	Read-only

7.13.1.3 OriginalToParty

The OriginalToParty property is the original destination of the call. For example: for a voice call, this is the DNIS; for an e-mail message, this is the to e-mail address; for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oContentAddedEvent.OriginalToParty
```

Property:	OriginalToParty (of ContentAddedEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Web collaboration (Initial URL)
Notes:	Read-only

7.13.1.4 Party

The Party property is the party that is sending the message. This is provided so you can display both the message and who sent it. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oContentAddedEvent.Party
```

Property:	Party (of ContentAddedEvent)
Description:	The party that is sending the message.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Web collaboration
Notes:	Read-only

7.13.1.5 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oContentAddedEvent.SequenceNumber
```

Property:	SequenceNumber (of ContentAddedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.13.1.6 Text

The Text property is the message that is sent by the Party. This is provided so you can display both the message and who sent it.

To use this property, do the following:

```
Dim strMessage as String
strMessage = oContentAddedEvent.Text
```

Property:	Text (of ContentAddedEvent)
Description:	The message that is sent.
Type:	String
Potential Errors:	None
Supported Media:	Web collaboration
Notes:	Read-only

7.14 Criterion

The Criterion object is a pair comprising an enEmailsHistoryQueryCriteria enumeration and a corresponding string value. This object represents an individual e-mail search criterion.

Interfaces supported:

- ICriterion (default)
- IAdministrationEvent
- IXMLAccess

7.14.1 Properties

This section contains properties exposed through the ICriterion interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.14.1.1 Type

The Type property returns the criterion type as an enEmailsHistoryQueryCriteria enumeration. To use this property, do the following:

```
Dim enType as enEmailsHistoryQueryCriteria
enType = oCriterion.Type
```

Property:	Type (of Criterion)
Description:	This property returns the criterion type.
Type:	enEmailsHistoryCriterionType
Potential Errors:	None
Notes:	Read-only

7.14.1.2 Value

The Value property returns a value of the criterion object.

To use this property, do the following:

```
Dim criterionType as enEmailsHistoryQueryCriteria
Dim criterionValue as String
criterionType = EmailsHistoryQueryCriterion_QueueKey
criterionValue = "1"
Dim oCriterionList as HiPathProCenterLibrary.CriterionList
oCriterionList = New HiPathProCenterLibrary.CriterionList
oCriterionList.Add(criterionType, criterionValue)
Dim oCriterion as HiPathProCenterLibrary.Criterion
oCriterion = New HiPathProCenterLibrary.Criterion
oCriterion =
oCriterionList.Item(EmailsHistoryQueryCriterion_QueueKey)
criterionValue = oCriterion.Value
```

Property:	Value (of Criterion)
Description:	This property returns a string representation of the Criterion value.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.15 CriterionList

The CriterionList object is a collection of pairs each of which comprises an enEmailsHistoryQueryCriteria enumeration and a corresponding string value.

Interfaces supported:

- ICriterionList (default)

7.15.1 Properties

This section contains properties exposed through the ICriterionList interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.15.1.1 _NewEnum

For Each oCriterion in colCriterionList

...

Next

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oCriterion as HiPathProCenterLibrary.Criterion
Dim oStr as String
For Each oCriterion in colCriterionList
oStr = "Type: " & oCriterion.Type & _
" Value: " & oCriterion.Value vbCrLf
Next
```

This example shows how to iterate through a CriterionList collection and retrieve type and This method does not accept any parameters. from the individual Criterion elements.

Property:	_NewEnum (of CriterionList)
Description:	Provides access to elements of the CriterionList by means of operator FOREACH.
Returns:	oCriterion — For each item in the CriterionList collection until all items have been returned.
Potential Errors:	Error_Generic_IterationGetNewEnum Error_Generic_UnableToAllocateMemory
Notes:	This method allows the CriterionList collection to support the 'For Each' iterator.

7.15.1.2 Count

The Count property returns number of elements in the list.

To use this property, do the following:

```
Dim lCount as Long
lCount = oCriterionList.Count
```

Property:	Count (of CriterionList)
Description:	This property returns a number of elements in the list.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.15.1.3 Item

The Item property returns a Criterion object pointed by a given index key.

To use this property, do the following:

```
Dim lIndex as Long
Dim oCriterion as ICriterion
oCriterion = New Criterion
oCriterion = oCriterionList.Item(lIndex)
```

Property:	Item (of CriterionList)
Description:	This property returns a Criterion object pointed by a given index.
Type:	ICriterion
Potential Errors:	None
Notes:	Read-only

7.15.2 Methods

This section contains methods exposed through the ICriterionList interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.15.2.1 Add

The Add method allows a caller to add a new pair criterion to the CriterionList collection.

The signature of this method is:

```
Add( enType as enEmailsHistoryQueryCriteria, strValue as String)
```

This method accepts the following parameters:

enType as enEmailsHistoryQueryCriteria	This enumeration value identifies the criterion type.
strValue as String	This is the criterion value.

To use this method, do the following:

```
Dim criterionType as enEmailsHistoryQueryCriteria
Dim criterionValue as String
criterionType = EmailsHistoryQueryCriterion_QueueKey
criterionValue = "1"
Dim oCriterionList as HiPathProCenterLibrary.CriterionList
oCriterionList = New HiPathProCenterLibrary.CriterionList
oCriterionList.Add(criterionType, criterionValue)
```

Method:	Add (of CriterionList)
Description:	Enables SDK client applications to add new Criteria to CriterionList object.
Parameters:	• enType — the criterion type • strValue — the criterion value
Returns:	0 — success
Potential Errors:	None
Notes:	None

7.15.2.2 Remove

The Remove method allows a caller to remove a pair criterion from the CriterionList collection.

The signature of this method is:

```
Remove( enType as enEmailsHistoryQueryCriteria )
```

This method accepts the following parameters:

enType as enEmailsHistoryQueryCriteria	This enumeration value identifies the criterion type.

To use this method, do the following:

```
Dim criterionType as enEmailsHistoryQueryCriteria
Dim criterionValue as String
criterionType = EmailsHistoryQueryCriterion_QueueKey
criterionValue = "1"
Dim oCriterionList as HiPathProCenterLibrary.CriterionList
oCriterionList = New HiPathProCenterLibrary.CriterionList
oCriterionList.Add(criterionType, criterionValue)
oCriterionList.Remove(criterionType)
```

The preceding sample code adds and then removes a single "queue key" type of criterion. Please note that the Remove method only accepts the criterion type.

Method:	Remove (of CriterionList)
Description:	Enables SDK client applications to remove a Criterion object from the CriterionList collection
Parameters:	enType — the criterion type
Returns:	0 — success
Potential Errors:	None
Notes:	None

7.15.2.3 RemoveAll

Add method allows a caller to clear (remove all elements) from the CriterionList collection.

The signature of this method is:

```
RemoveAll()
```

This method does not have any parameters:

To use this method, do the following:

```
Dim criterionType as enEmailsHistoryQueryCriteria
Dim criterionValue as String
criterionType = EmailsHistoryQueryCriterion_QueueKey
criterionValue = "1"
Dim oCriterionList as HiPathProCenterLibrary.CriterionList
oCriterionList = New HiPathProCenterLibrary.CriterionList
oCriterionList.Add(criterionType, criterionValue)
oCriterionList.RemoveAll
```

The preceding sample code adds one element to the criterion list and then removes all elements (one element in this case) from the list.

Method:	RemoveAll (of CriterionList)
Description:	Enables SDK client applications to remove all elements from the CriterionList collection
Parameters:	None
Returns:	0 — success
Potential Errors:	None
Notes:	None

7.16 DeliveredEvent

The DeliveredEvent object is sent when a contact is delivered to a monitored device in the system. In most cases, this is when a contact has been assigned and sent to a user.

Interfaces supported:

- IDeliveredEvent5 (default)
- IMediaEvent
- IXMLAccess

NOTE: For DeliveredEvent with MediaType property set to MediaType_Callback, only the CallID property is available. All other properties return error code Error_Media_PropertyNotSupportedForCallback.

7.16.1 Properties

This section contains properties exposed through the DeliveredEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.16.1.1 CallID

The CallID property is the unique key for the contact in the system. This value is unique whether the contact is voice, callback, e-mail, or Web collaboration.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oDeliveredEvent.CallID
```

Property:	CallID (of DeliveredEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice, callback e-mail, Web collaboration
Notes:	Read-only

7.16.1.2 ConversationID

The ConversationID property determines which interactions are associated with the same e-mail message. In an e-mail message, there may be various messages sent back and forth between the customer and the user. Each of these e-mail messages will have a different CallID, but each e-mail that is associated with the same e-mail thread will have the same conversation ID.

To use this property, do the following:

```
Dim strConversationID as String
strConversationID = oDeliveredEvent.ConversationID
```

Property:	ConversationID (of DeliveredEvent)
Description:	The conversation ID used to determine which interactions are associated with the same call.
Type:	String
Potential Errors:	None
Supported Media:	E-mail
Notes:	Read-only

7.16.1.3 DeliveredToParty

The DeliveredToParty property is the party the contact is delivered to. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDeliveredEvent.DeliveredToParty
```

Property:	DeliveredToParty (of DeliveredEvent)
Description:	The party the contact is delivered to.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.16.1.4 FromParty

The FromParty property is the party that the contact is coming from. In some instances, this will be the same as the OriginalFromParty, but in some scenarios (for example, a call being transferred by another user) this may be different. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDeliveredEvent.FromParty
```

Property:	FromParty (of DeliveredEvent)
Description:	The from party when the contact is queued.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.16.1.5 HeldParty

The HeldParty property is the party (if any) that is on hold when the call is delivered. This is used for example in a consultation scenario, when user A is consulting with user B, and the customer is the held party. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDivertedEvent.HeldParty
```

Property:	HeldParty (of DeliveredEvent)
Description:	The party (if any) that is on hold when the contact is delivered.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.16.1.6 OriginalFromParty

The OriginalFromParty property is the original calling party that placed the call. For example: for a voice call, this is the ANI; for an e-mail message, this is the From e-mail address; for a Web collaboration session, this is the customer's IP address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDeliveredEvent.OriginalFromParty
```

Property:	OriginalFromParty (of DeliveredEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	voice (ANI), e-mail (From e-mail address), Web collaboration (customer IP address)
Notes:	Read-only

7.16.1.7 OriginalToParty

The OriginalToParty property is the original destination of the call. For example: for a voice call, this is the DNIS; for an e-mail message, this is the To e-mail address; for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDeliveredEvent.OriginalToParty
```

Property:	OriginalToParty (of DeliveredEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	voice (DNIS), e-mail (To e-mail address), Web collaboration (initial URL)
Notes:	Read-only

7.16.1.8 Reason

The Reason property returns the delivered reason enumeration value. For media type of WebCollaboration, the Reason property is used to distinguish if the Delivered event indicates a new call or a conference invitation from other agent. For other media types this property is not applicable and it will return DeliveredReason_NotApplicable.

To use this property, do the following:

```
Dim enReason as HiPathProCenterLibrary.enDeliveredReasons
enReason = oDeliveredEvent.Reason
```

Property:	Reason (of DeliveredEvent)
Description:	This property returns a delivered reason.
Type:	enDeliveredReasons
Potential Errors:	None
Notes:	Read-only

7.16.1.9 RedirectParty

The RedirectParty property returns the telephone number from which the call was last forwarded or deflected. This field is populated only when a redirect occurs and the system is connected to an OpenScape Voice, OpenScape 4000, or HiPath 4000 communication platform.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDeliveredEvent.RedirectParty
```

Property:	Redirect (of DeliveredEvent)
Description:	The telephone number from which the call was last forwarded or deflected. This field is populated only when a redirect occurs and the system is connected to an OpenScape Voice, OpenScape 4000, or HiPath 4000 communication platform.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject Error_Media_PropertyNotSupportedForNonVoiceContact
Supported Media:	Voice
Notes:	Read-only

7.16.1.10 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oDeliveredEvent.SequenceNumber
```

Property:	SequenceNumber (of DeliveredEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.16.1.11 ToParty

The ToParty property is the party the call was transferred to so that the call could be queued. In some instances, this will be the same as the OriginalToParty, but in some scenarios (for example, a call being transferred by another user) this may be different. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDeliveredEvent.ToParty
```

Property:	ToParty (of DeliveredEvent)
Description:	The to party the call was transferred to for the call to be queued.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.17 DisconnectedEvent

The DisconnectedEvent object is sent whenever a party that is on a contact (voice, e-mail, Web collaboration) in the system disconnects (hangs-up).

Interfaces supported:

- IDisconnectedEvent4 (default)
- IMediaEvent
- IXMLAccess

7.17.1 Properties

This section contains properties exposed through the EstablishedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.17.1.1 CallID

The CallID property is the unique key for the contact in the system. This value is unique whether the contact is a voice, callback, e-mail, or Web collaboration.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oDisconnectedEvent.CallID
```

Property:	CallID (of DisconnectedEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice, callback, e-mail, Web collaboration
Notes:	Read-only

7.17.1.2 ConversationID

The ConversationID property determines which interactions are associated with the same e-mail message. This property is only used for e-mail messages. In an e-mail message, there may be various messages sent back and forth between the customer and the user.

Each of these e-mail messages will have a different CallID, but each e-mail message that is associated with the same e-mail thread will have the same conversation ID.

To use this property, do the following:

```
Dim strConversationID as String
strConversationID = oDisconnectedEvent.ConversationID
```

Property:	ConversationID (of DisconnectedEvent)
Description:	The conversation ID used to determine which interactions are associated with the same call.
Type:	String
Potential Errors:	None
Supported Media:	E-mail
Notes:	Read-only

7.17.1.3 DiscardReasonKey

This DiscardReasonKey property returns a Discard reason key as defined in the database. Applications can use QueryEmailDiscardReasons to get more information on the actual Discard reason.

To use this property, do the following:

```
Dim lKey as Long
lKey = oDisconnectedEvent.DiscardReasonKey
```

Property:	DiscardReasonKey (of DisconnectedEvent)
Description:	This property returns a Discard reason key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.17.1.4 DisconnectedParty

The DisconnectedParty property is the party that is disconnecting from the contact. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDisconnectedEvent.DisconnectedParty
```

Property:	DisconnectedParty (of DisconnectedEvent)
Description:	The party that is disconnecting from the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.17.1.5 OriginalFromParty

The OriginalFromParty property is the original party that placed the contact. For example: for a voice call, this is the ANI; for an e-mail message, this is the From e-mail address; for a Web collaboration session, this is the customers IP address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDisconnectedEvent.OriginalFromParty
```

Property:	OriginalFromParty (of DisconnectedEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (ANI), e-mail (From e-mail address), Web collaboration (customer IP address)
Notes:	Read-only

7.17.1.6 OriginalToParty

The OriginalToParty property is the original destination of the contact. For example: for a voice call, this is the DNIS; for an e-mail message, this is the To e-mail address; for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDisconnectedEvent.OriginalToParty
```

Property:	OriginalToParty (of DisconnectedEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (DNIS), e-mail (To e-mail address), Web collaboration (initial URL)
Notes:	Read-only

7.17.1.7 Reason

The Reason property returns the disconnect reason enumeration value.

To use this property, do the following:

```
Dim enReason as HiPathProCenterLibrary.enDisconnectReasons
enReason = oDisconnectedEvent.Reason
```

Property:	Reason (of DisconnectedEvent)
Description:	This property returns a disconnect reason
Type:	enDisconnectReason
Potential Errors:	None
Notes:	Read-only

7.17.1.8 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oDisconnectedEvent.SequenceNumber
```

Property:	SequenceNumber (of DisconnectedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.18 DivertedEvent

The DivertedEvent object is sent when a contact is diverted from the queue to a user.

Interfaces supported:

- IDivertedEvent4 (default)
- IMediaEvent
- IXMLAccess

7.18.1 Properties

This section contains properties exposed through the DivertedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.18.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oDivertedEvent.CallID
```

Property:	CallID (of DivertedEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice, e-mail
Notes:	Read-only

7.18.1.2 ConversationID

The ConversationID property determines which interactions are associated with the same e-mail message. This property is only used for an e-mail message. In an e-mail message, there may be various messages sent back and forth between the customer and the user. Each of these e-mail messages will have a different CallID, but each e-mail message that is associated with the same e-mail thread will have the same conversation ID.

To use this property, do the following:

```
Dim strConversationID as String
strConversationID = oDivertedEvent.ConversationID
```

Property:	ConversationID (of DivertedEvent)
Description:	The conversation ID used to determine which interactions are associated with the same call.
Type:	String

Potential Errors:	None
Supported Media:	E-mail
Notes:	Read-only

7.18.1.3 DestinationParty

The DestinationParty property is the party the call is diverted to. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDivertedEvent.DestinationParty
```

Property:	DestinationParty (of DivertedEvent)
Description:	The party the call is diverted to.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail
Notes:	Read-only

7.18.1.4 DivertedParty

The DivertedParty property is the party that is diverted. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDivertedEvent.DivertedParty
```

Property:	DivertedParty (of DivertedEvent)
Description:	The party that is diverted.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail
Notes:	Read-only

7.18.1.5 DivertingParty

The DivertingParty property is the party that is diverting the call. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDivertedEvent.DivertingParty
```

Property:	DivertingParty (of DivertedEvent)
Description:	The party that is diverting the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail
Notes:	Read-only

7.18.1.6 OriginalFromParty

The OriginalFromParty property is the original calling party that placed the call. For example: for a voice call, this is the ANI; for an e-mail message, this is the From e-mail address; for a Web collaboration session, this is the customers IP address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDivertedEvent.OriginalFromParty
```

Property:	OriginalFromParty (of DivertedEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (ANI), e-mail (From e-mail address)
Notes:	Read-only

7.18.1.7 OriginalToParty

The OriginalToParty property is the original destination of the call. For example: for a voice call, this is the DNIS; for an e-mail message, this is the To e-mail address; for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oDivertedEvent.OriginalToParty
```

Property:	OriginalToParty (of DivertedEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (DNIS), e-mail (To e-mail address)
Notes:	Read-only

7.18.1.8 Reason

The Reason property returns the diverted reason enumeration value.

To use this property, do the following:

```
Dim enReason as HiPathProCenterLibrary.enDivertReasons
enReason = oDivertedEvent.Reason
```

Property:	Reason (of DivertedEvent)
Description:	This property returns a diverted reason
Type:	enDivertReasons
Potential Errors:	None
Notes:	Read-only

7.18.1.9 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oDivertedEvent.SequenceNumber
```

Property:	SequenceNumber (of DivertedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.19 EmailAddress

The EmailAddress object contains information about an e-mail address. This includes the e-mail address (that is, name@address.com) as well as the displayed name (Mr. First Last).

Interfaces supported:

- IEmailAddress (default)
- IXMLAccess

7.19.1 Properties

This section contains properties exposed through the EmailAddress interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.19.1.1 DisplayName

The DisplayName property is the name that is displayed with the e-mail address. This provides an alternative method of determining who an e-mail is to or from.

To use this property, do the following:

```
Dim strDisplayName as String
strDisplayName = oEmailAddress.DisplayName
```

Property:	DisplayName (of EmailAddress)
Description:	The name that is displayed with the e-mail address.
Type:	String
Potential Errors:	None
Notes:	None

7.19.1.2 EmailAddress

The EmailAddress property is the e-mail address that the e-mail is either to or from. This is in the form name@address.com.

To use this property, do the following:

```
Dim strEmailAddress as String
strEmailAddress = oEmailAddress.EmailAddress
```

Property:	EmailAddress (of EmailAddress)
Description:	The e-mail address that the e-mail is either to or from.
Type:	String
Potential Errors:	None
Notes:	None

7.20 EmailAddresses

The EmailAddresses object is a collection of the EmailAddress objects.

Interfaces supported:

- IEmailAddresses (default)
- IXMLAccess

7.20.1 Properties

This section contains properties exposed through the EmailAddresses interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.20.1.1 _NewEnum

```
For Each oEmailAddress in colEmailAddresses
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oEmailAddress as HiPathProCenterLibrary.EmailAddress
For Each oEmailAddress in m_colEmailAddresses
   txtTextBox.Text = txtTextBox.Text & _
      "Display Name: " & _
      oEmailAddress. DisplayName & _
      " Email Address: " & _
      oEmailAddress.EmailAddress& _
      vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the display names of all the EmailAddress objects in the EmailAddresses collection, as well as the e-mail address properties.

Property:	_NewEnum (of EmailAddresses)
Description:	Allows for the 'For Each' operator on the EmailAddresses collection.
Returns:	OEmailAddress — for each item in the EmailAddresses collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the EmailAddresses collection to support the 'For Each' iterator.

7.20.1.2 Count

The Count property returns the number of EmailAddress objects that exist in the EmailAddresses collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colEmailAddresses.Count
```

Property:	Count (of EmailAddresses)
Description:	The number of EmailAddress objects that are in the EmailAddresses collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.20.1.3 Item

```
oEmailAddress = Item (Key as Long)
```

Use the Item method to find an EmailAddress object in the EmailAddresses collection that is indexed by the key value. If an item is found in the collection at the key value, the Item method returns the EmailAddress object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameter:

Key as Long	This is the key for the e-mail address you are requesting.

This method returns the following parameters:

oEmailAddress as EmailAddress	This is the EmailAddress object for e-mail address specified by the Key value. This contains the EmailAddress for the key specified.

To use this method, do the following:

```
Dim oEmailAddress as HiPathProCenterLibary. EmailAddress
Set oEmailAddress = colEmailAddresses.Item (lKey)
```

This checks to see if the EmailAddress specified by the Key exists. If it does, oEmailAddress will contain the EmailAddress object. If it does not exist in the EmailAddresses collection, this will raise an error.

Method:	Item (of EmailAddresses)
Description:	Finds an EmailAddress object in the EmailAddresses collection that is indexed by the key value
Parameter:	Key — the key for the EmailAddress that you are looking for. This is an index starting from 1.
Returns:	oEmailAddress — the e-mail address for the key specified.
Potential Errors:	Error_Generic_ItemNotFound
Notes:	None

7.20.2 Methods

This section contains methods exposed through the EmailAddresses interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.20.2.1 Add

```
Add (EmailAddress as EmailAddress)
```

Use the Add method to add a new EmailAddress object to the EmailAddresses collection.

This method accepts the following parameters:

EmailAddress as EmailAddress	This is the EmailAddress object that you want to add to EmailAddresses collection.

This method does not return anything.

To use this method, do the following:

```
Dim oEmailAddress as HiPathProCenterLibary. EmailAddress
oEmailAddress.DisplayName = txtDisplayName.Text
oEmailAddress.EmailAddress = txtPath.EmailAddress
Call colEmailAddresses.Add(oEmailAddress)
```

This adds an EmailAddress to the EmailAddresses collection.

Method:	Add (of EmailAddresses)
Description:	Adds an EmailAddress object to the EmailAddresses collection.
Parameters:	EmailAddress — the EmailAddress object that you want to add to the collection.

Returns:	None
Potential Errors:	Error_Generic_UnableToAllocateMemory
Notes:	Note that EmailCall.Attachments is read-only and you can not add or remove an attachment from there

7.20.2.2 Remove

```
Remove (lKey as Long)
```

The Remove method removes an item from our EmailAddresses collection that is indexed by the lKey value. If an item is found in the collection at that key, the Remove method will remove the EmailAddress from our EmailAddresses collection. If the item does not exist in the collection, then the Remove returns without raising an error.

This method accepts the following parameters:

lKey as Long	This is the long key of the item in the collection.

This method does not return anything.

To use this method, do the following:

```
Call EmailAddresses.Remove (lKey)
```

This removes the EmailAddress object from the EmailAddresses collection that is indexed by lkey.

Method:	Remove (of EmailAddresses)
Description:	Removes the lKeyth EmailAddress object from the EmailAddresses collection. lKey is an index starting from 1.
Parameters:	Key — the key for the EmailAddress that you are removing.
Returns:	None
Potential Errors:	Error_Generic_ItemNotFound
Notes:	If the item does not exist in the collection, no error is raised.

7.21 EmailAttachment

The EmailAttachment object contains information about an e-mail attachment. This includes the e-mail name as well as the full path.

Interfaces supported:

- IEmailAttachment2 (default)
- IXMLAccess

7.21.1 Properties

This section contains properties exposed through the EmailAttachment interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.21.1.1 NameUTF8

The NameUTF8 property contains the file name of the attachment for both incoming (set by the user e-mail interface) and outgoing (set by the application developer to indicate what name should be used in the e-mail message).

To use this property, do the following:

```
Dim strNameUTF8 as String
strNameUTF8 = oEmailAttachment.NameUTF8
```

Property:	NameUTF8 (of EmailAttachment)
Description:	The file name of the incoming/outgoing attachment.
Type:	String
Potential Errors:	None
Notes:	None

7.21.1.2 Path

The Path property specifies the full path (including file name) for an e-mail attachment file.

To use this property, do the following:

```
Dim strPath as String
```

```
strPath = oEmailAttachment.Path
```

Property:	Path (of EmailAttachment)
Description:	Specifies the full path (including file name) for an e-mail attachment file.
Type:	String
Potential Errors:	None
Notes:	None

7.21.2 Methods

This section contains methods exposed through the EmailAttachment interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.21.2.1 Retrieve

```
Retrieve (DestinationPath as String)
```

Use the Retrieve method to obtain an e-mail attachment from the corporate e-mail server and save the attachment to a specified location.

This method accepts the following parameters:

DestinationPath as String	This is the full path (including file name) where you want to save the e-mail attachment.

This method does not return anything.

To use this method, do the following:

```
Call oEmailAttachment.Retrieve(DestinationPath)
```

This method obtains the e-mail attachment from the corporate e-mail server and saves the attachment to the specified location.

Method:	Retrieve (of EmailAttachment)
Description:	Obtains the e-mail attachment from the corporate e-mail server and saves the attachment to the specified location.
Parameters:	DestinationPath — the full path (including file name) where you want to save the e-mail attachment.
Returns:	None
Potential Errors:	Error_Media_EmailAttachmentNotLinked
Notes:	The file name specified for the e-mail attachment must contain Latin 1 (ISO-8859-1) characters. The file can be renamed later if required.

7.22 EmailAttachments

The EmailAttachments object is a collection of the EmailAttachment objects.

Interfaces supported:

- IEmailAttachments (default)
- IXMLAccess

7.22.1 Properties

This section contains properties exposed through the EmailAttachments interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.22.1.1 _NewEnum

```
For Each oEmailAttachment in colEmailAttachments
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oEmailAttachment as HiPathProCenterLibrary.EmailAttachment
For Each oEmailAttachment in m_colEmailAttachment
   txtTextBox.Text = txtTextBox.Text & _
      "Attachment Name: " & _
      oEmailAttachment.Name & _
      " Attachment Path: " & _
      oEmailAttachment.Path& _
      vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the attachments names of all the EmailAttachment objects in the EmailAttachments collection, as well as the attachments Paths.

Property:	_NewEnum (of EmailAttachments)
Description:	Allows for the 'For Each' operator on the EmailAttachments collection.
Returns:	OEmailAttachment- for each item in the EmailAttachments collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the EmailAttachments collection to support the 'For Each' iterator.

7.22.1.2 Count

The Count property returns the number of EmailAttachment objects that exist in the EmailAttachments collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colEmailAttachments.Count
```

Property:	Count (of EmailAttachments)
Description:	The number of EmailAttachment objects that are in the EmailAttachments collection.
Type:	Long
Potential Errors:	None
Notes:	None

7.22.1.3 Item

```
oEmailAttachment = Item (Key as Long)
```

Use the Item method to find an EmailAttachment object in the EmailAttachments collection that is indexed by the key value. If an item is found in the collection at the key value, the Item method returns the EmailAttachment object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

Key as Long	This is the key for the e-mail attachment you are requesting.

This method returns the following parameters:

oEmail Attachment as Email Attachment	This is the EmailAttachment object for e-mail attachment specified by the Key value. This contains the EmailAttachment for the key specified.

To use this method, do the following:

```
Dim oEmailAttachment as HiPathProCenterLibary.EmailAttachment
Set oEmailAttachment = colEmailAttachments.Item (lKey)
```

This checks to see if the EmailAttachment specified by the Key exists. If it does, oEmailAttachment will contain the EmailAttachment object. If it does not exist in the EmailAttachments collection, this will raise an error.

Method:	Item (of EmailAttachments)
Description:	Finds an EmailAttachment object in the EmailAttachments collection that is indexed by the key value.
Parameters:	Key — the key for the EmailAttachment that you are looking for. This is an index starting from one (1).
Returns:	oEmailAttachment — the e-mail attachment for the key specified.
Potential Errors:	Error_Generic_ItemNotFound
Notes:	None

7.22.2 Methods

This section contains methods exposed through the EmailAttachments interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.22.2.1 Add

```
Add (EmailAttachment as EmailAttachment)
```

Use the Add method to add a new EmailAttachment to the EmailAttachments collection.

This method accepts the following parameters:

EmailAttachment as EmailAttachment	This is the EmailAttachment that you want to add to EmailAttachments collection.

This method does not return anything.

To use this method, do the following:

```
Dim oEmailAttachment as HiPathProCenterLibary.EmailAttachment
oEmailAttachment.Name = txtName.Text
oEmailAttachment.Path = txtPath.Text
Call colEmailAttachments.Add(oEmailAttachment)
```

This adds an EmailAttachment to the EmailAttachments collection.

Property:	Add (of EmailAttachments)
Description:	Adds an EmailAttachment object to the EmailAttachments collection.
Parameters:	EmailAttachment — the EmailAttachment that you want to add to the collection.

Returns:	None
Potential Errors:	Error_Generic_UnableToAllocateMemory
Notes:	None

7.22.2.2 Remove

```
Remove (lKey as Long)
```

The Remove method removes an item from the EmailAttachments collection that is indexed by the lKey value. If an item is found in the collection at that key, the Remove method will remove the EmailAttachment from the EmailAttachments collection. If the item does not exist in the collection, then the Remove returns without raising an error.

This method accepts the following parameter:

lKey as Long	This is the long key of the item in the collection.

This method does not return anything.

To use this method, do the following:

```
Call EmailAttachments.Remove (lKey)
```

This removes the EmailAttachment object from the EmailAttachments collection that is indexed by lkey.

Method:	Remove (of EmailAttachments)
Description:	Removes the lKeyth EmailAttachment object from the Emailattachments collection. lKey is an index starting from 1.
Parameters:	Key — the key for the EmailAttachment that you are removing.
Returns:	None
Potential Errors:	Error_Generic_ItemNotFound
Notes:	None

7.23 EmailCall

The EmailCall object represents an e-mail message in the system. This does such things as reply to, or discard, e-mail messages.

Interfaces supported:

- IEmailCall7 (default)
- IBinaryAccess

- IMediaBase
- IXMLAccess

7.23.1 Properties

This section contains properties exposed through the EmailCall interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.23.1.1 AgentKey

The AgentKey property is the key for the user in the database. This key represents the user who is associated with the call. This can be the user who is placing an outgoing call, or a user who the call has been assigned to. This is a system generated value.

To use this property, do the following:

```
Dim lKey as Long
lKey = oEmailCall.AgentKey
```

Property:	AgentKey (of EmailCall)
Description:	The database table unique key for the Agent object associated with this call.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.23.1.2 Attachments

The Attachments property is a collection of incoming e-mail attachments.

To use this property, do the following:

```
Dim colAttachments as HiPathProCenterLibrary.EmailAttachments
Set colAttachments = oEmailCall. Attachments
```

Property:	Attachments (of EmailCall)
Description:	A collection of incoming e-mail attachments.
Type:	EmailAttachments
Potential Errors:	Error_Generic_CreatingObject
Notes:	You cannot set this property because it is read-only.

7.23.1.3 BodyUTF8

The BodyUTF8 property contains the original body of the e-mail message that was sent. This property cannot be changed.

To use this property, do the following:

```
Dim strBodyUTF8 as String
strBodyUTF8 = oEmailCall.BodyUTF8
```

Property:	BodyUTF8 (of EmailCall)
Description:	The original body of the e-mail message that was sent.
Type:	String
Potential Errors:	None
Notes:	Read-only. The body of the e-mail message is encoded in UTF8 format.

7.23.1.4 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oEmailCall.CallID
```

Property:	CallID (of EmailCall)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Notes:	This property should be set before querying an EmailCall from the MediaManager object.

7.23.1.5 ContactData

The ContactData property is the contact data that is associated with this EmailCall object. The ContactData property can provide the most information about a call. The ContactData object is a collection of key/value pairs associated with the Call. This can be used to hold information entered in the IVR, or information from other agents that have talked to this customer.For more information, see Section 4.1, "ContactData", on page 117. This read/write property is also used so that the Contact Data for this contact can be updated.

To use this property, do the following:

```
Dim oContactData as HiPathProCenterLibrary.ContactData
Set oContactData = oEmailCall.ContactData
```

Property:	ContactData (of EmailCall)
Description:	The contact data that is associated with this contact in the system.
Type:	ContactData
Potential Errors:	None
Notes:	Read-only

7.23.1.6 ConversationID

The ConversationID property is a key value that is set for all e-mail messages that are part of the same conversation. For example, it is reasonable to expect that a scenario could involve multiple e-mail messages being sent between two parties in the process of resolving an issue. Each one of these e-mail messages will have a unique CallID, but will all share the same ConversationID.

To use this property, do the following:

```
Dim strConversationID as String
strConversationID = oEmailCall.ConverstationID
```

Property:	ConversationID (of EmailCall)
Description:	The key for all e-mail messages that are part of the same conversation thread.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.23.1.7 Description

The Description property is the description that is associated with this EmailCall object. The description is a single line of text that can be displayed in the Screen Pop portion of the Client Desktop application. This can be used to provide some information about the queue that the queue itself may not be able to do. For example, if the queue is 'Sales' the description might be 'Sales Call for All Wood Furniture'. This read/write property is also used so that the description for this contact can be updated.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oEmailCall.Description
```

Property:	Description (of EmailCall)
Description:	The description that is associated with this contact in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.23.1.8 ExternalConsultCallID

The ExternalConsultCallID property provides the string that represents the call ID of the external consultation reply for a message.

This property is especially important in identifying the expert reply after using externalConsult method. After the contact center receives a reply from the third party expert, it will send a CallInformationUpdated event. To distinguish between normal CallInformationUpdated events and the ones that are being sent after receiving the expert reply use ExternalConsultCallID property. After you receive a CallInformationUpdated event, query for the customer's e-mail and check for ExternalConsultCallID property. If it is not empty, it means that the expert has replied. Query for a newly created EmailCall using the value of ExternalConsultCallID as its CallerID to get the expert reply.

To use this property, do the following:

```
Dim str ExternalConsultCallID as String
```

```
str ExternalConsultCallID = oEmailCall. ExternalConsultCallID
```

Property:	ExternalConsultCallID (of EmailCall)
Description:	The string that represents the call ID of the external consultation reply for a message.
Type:	String
Potential Errors:	None
Notes:	None

7.23.1.9 FromEmailAddress

The FromEmailAddress property contains the original e-mail address that the e-mail was sent from. This property cannot be changed.

To use this property, do the following:

```
Dim oEmailAddress as HiPathProCenterLibrary.EmailAddress
Set oEmailAddress = oEmailCall.FromEmailAddress
```

Property:	FromEmailAddress (of EmailCall)
Description:	The original e-mail address that the e-mail was sent from.
Type:	EmailAddress
Potential Errors:	Error_Generic_CreatingObject
Notes:	Read-only

7.23.1.10 LastDeferTimeStamp

The LastDeferTimeStamp property returns the time at which the e-mail message has been last deferred.

To use this property, do the following:

```
Dim oTime as Date
oTime = oEmailCall.LastDeferTimeStamp
```

Property:	LastDeferTimeStamp (of EmailCall)
Description:	The LastDeferTimeStamp property returns the time at which the e-mail message has been last deferred.
Type:	Date
Potential Errors:	None
Notes:	Read-only

7.23.1.11 QueueKey

The QueueKey property is the key for the queue in the database. This is a system generated value.

To use this property, do the following:

```
Dim lKey as Long
lKey = oEmailCall.QueueKey
```

Property:	QueueKey (of EmailCall)
Description:	The database table unique key for the Queue object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.23.1.12 ReplyAttachments

The ReplyAttachments property is a collection of outgoing e-mail attachments.

To use this property, do the following:

```
Dim colAttachments as HiPathProCenterLibrary.EmailAttachments
Set colAttachments = oEmailCall. ReplyAttachments
```

Property:	ReplyAttachments (of EmailCall)
Description:	A collection of outgoing e-mail attachments.
Type:	EmailAttachments
Potential Errors:	Error_Generic_CreatingObject
Notes:	To pass outgoing attachments, you must set this property prior to calling the Reply method.

7.23.1.13 ReplyBodyUTF8

The ReplyBodyUTF8 property contains the body that should be used when replying.

To use this property, do the following:

```
Dim strBodyUTF8 as String
strBodyUTF8 = oEmailCall.ReplyBodyUTF8
```

Property:	ReplyBodyUTF8 (of EmailCall)
Description:	The body that should be used when replying.
Type:	String
Potential Errors:	None
Notes:	You must pass a string in UTF8 format as this property value.

7.23.1.14 ReplyFromEmailAddress

The ReplyFromEmailAddress property contains the from e-mail address that should be used when replying.

To use this property, do the following:

```
Dim oEmailAddress as HiPathProCenterLibrary.EmailAddress
Set oEmailAddress = oEmailCall.ReplyFromEmailAddress
```

Property:	ReplyFromEmailAddress (of EmailCall)
Description:	The from e-mail address that should be used when replying.
Type:	EmailAddress
Potential Errors:	Error_Generic_CreatingObject (Get only)
Notes:	None

7.23.1.15 ReplySubjectUTF8

The ReplySubjectUTF8 property contains the subject that should be used when replying.

To use this property, do the following:

```
Dim strSubjectUTF8 as String
strSubjectUTF8 = oEmailCall.ReplySubjectUTF8
```

Property:	ReplySubjectUTF8 (of EmailCall)
Description:	The subject that should be used when replying.
Type:	String
Potential Errors:	None
Notes:	You must pass a string in UTF8 format as this property value.

7.23.1.16 ReplyToBCCEmailAddresses

The ReplyToBCCEmailAddresses property is a collection of all replied BCC e-mail addresses that the reply or partial reply e-mail was blank copied to. For more information, see Section 7.20, “EmailAddresses”, on page 364.

To use this property, do the following:

```
Dim colEmailAddresses as
HiPathProCenterLibrary.ReplyToBCCEmailAddresses
Set colEmailAddresses = oEmailCall. ReplyToBCCEmailAddresses
```

Property:	ReplyToBCCEmailAddresses (of EmailCall)
Description:	A collection of all replied BCC e-mail addresses that the reply or partial reply e-mail was blank copied to.
Type:	EmailAddresses
Potential Errors:	Error_Generic_CreatingObject Error_Media_EmailAttachmentNotComplete
Notes:	None

7.23.1.17 ReplyToCCEmailAddresses

The ReplyToCCEmailAddresses property is a collection of all replied CC e-mail addresses that the reply or partial reply e-mail was copied to. For more information, see Section 7.20, "EmailAddresses", on page 364.

To use this property, do the following:

```
Dim colEmailAddresses as
HiPathProCenterLibrary.ReplyToCCEmailAddresses
Set colEmailAddresses = oEmailCall. ReplyToCCEmailAddresses
```

Property:	ReplyToCCEmailAddresses (of EmailCall)
Description:	A collection of all replied CC e-mail addresses that the reply or partial reply e-mail was copied to.
Type:	EmailAddresses
Potential Errors:	Error_Generic_CreatingObject Error_Media_EmailAttachmentNotComplete
Notes:	None

7.23.1.18 ReplyToEmailAddress

The ReplyToEmailAddress property contains the to e-mail address that should be used when replying.

To use this property, do the following:

```
Dim oEmailAddress as HiPathProCenterLibrary.EmailAddress
Set oEmailAddress = oEmailCall.ReplyToEmailAddress
```

Property:	ReplyToEmailAddress (of EmailCall)
Description:	The to e-mail address that should be used when replying.
Type:	EmailAddress
Potential Errors:	Error_Generic_CreatingObject (Get only)
Notes:	None

7.23.1.19 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oEmailCall.SequenceNumber
```

Property:	SequenceNumber (of EmailCall)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.23.1.20 State

The State property is the state that the contact is in. This is one of the valid enumeration This method does not accept any parameters. from the enCallStates enumeration.

To use this property, do the following:

```
Dim enCallState as HiPathProCenterLibrary.enCallStates
enCallState = oEmailCall.State
```

Property:	State (of EmailCall)
Description:	The state of the call.
Type:	EnCallStates
Potential Errors:	None
Notes:	Read-only

7.23.1.21 SubjectUTF8

The SubjectUTF8 property contains the original subject of the e-mail that was sent. This property cannot be changed.

To use this property, do the following:

```
Dim strSubjectUTF8 as String
strSubjectUTF8 = oEmailCall.SubjectUTF8
```

Property:	SubjectUTF8 (of EmailCall)
Description:	The original subject of the e-mail that was sent.
Type:	String
Potential Errors:	None
Notes:	Read-only. The subject of the e-mail message is encoded in UTF8 format

7.23.1.22 ToBCCEmailAddresses

The ToBCCEmailAddresses property is a collection of all BCC e-mail addresses that the deferred, forwarded to another user, or requeued e-mail was blank copied to. For more information, see Section 7.20, “EmailAddresses”, on page 364.

To use this property, do the following:

```
Dim colEmailAddresses as
HiPathProCenterLibrary.ToBCCEmailAddresses
Set colEmailAddresses = oEmailCall.ToBCCEmailAddresses
```

Property:	ToBCCEmailAddresses (of EmailCall)
Description:	A collection of all BCC e-mail addresses that the deferred, forwarded to another user, or requeued e-mail was blank copied to.
Type:	EmailAddresses
Potential Errors:	Error_Generic_CreatingObject
Notes:	Read-only

7.23.1.23 ToCCEmailAddresses

The ToCCEmailAddresses property is a collection of all CC e-mail addresses that the incoming e-mail was copied to. For more information, see Section 7.20, “EmailAddresses”, on page 364.

To use this property, do the following:

```
Dim colEmailAddresses as
HiPathProCenterLibrary.ToCCEmailAddresses
Set colEmailAddresses = oEmailCall.ToCCEmailAddresses
```

Property:	ToCCEmailAddresses (of EmailCall)
Description:	A collection of all CC e-mail addresses that the incoming e-mail was copied to.
Type:	EmailAddresses
Potential Errors:	Error_Generic_CreatingObject
Notes:	Read-only

7.23.1.24 ToEmailAddress

The ToEmailAddress property contains the original to e-mail address that the e-mail was sent to. This property cannot be changed.

To use this property, do the following:

```
Dim oEmailAddress as HiPathProCenterLibrary.EmailAddress
Set oEmailAddress = oEmailCall.ToEmailAddress
```

Property:	ToEmailAddress (of EmailCall)
Description:	The original to e-mail address that the e-mail was sent to.
Type:	EmailAddress
Potential Errors:	Error_Generic_CreatingObject
Notes:	Read-only

7.23.1.25 ToEmailAddresses

The ToEmailAddresses property is a collection of all e-mail addresses that the e-mail was sent to. For more information, see Section 7.20, "EmailAddresses", on page 364.

To use this property, do the following:

```
Dim colEmailAddresses as HiPathProCenterLibrary.ToEmailAddresses
Set colEmailAddresses = oEmailCall.ToEmailAddresses
```

Property:	ToEmailAddresses (of EmailCall)
Description:	A collection of all e-mail addresses that the e-mail was sent to.
Type:	EmailAddresses
Potential Errors:	Error_Generic_CreatingObject
Notes:	Read-only

7.23.1.26 WaitTime

The WaitTime property is the amount of time in seconds the contact was queued prior to being delivered to the agent.

To use this property, do the following:

```
Dim lWaitTime as Long
lWaitTime = oCallbackCall.WaitTime
```

Property:	WaitTime (of EmailCall)
Description:	The amount of time in seconds the contact was queued prior to being delivered to the agent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.23.2 Methods

This section contains methods exposed through the EmailCall interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.23.2.1 Accept

```
Accept()
```

The Accept method accepts an e-mail that a user has been offered. This is required when an e-mail message is offered to a user so the E-mail Server knows that the user will handle the e-mail message.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following after creating the EmailCall:

```
Call oEmailCall.Accept
```

This informs the E-mail Server that the user who has accepted the e-mail contact will handle the e-mail message.

Method:	Accept (of EmailCall)
Description:	Accepts an e-mail message that is offered to a user.
Parameters:	None
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_RequestTimedOut • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownError • Error_Generic_ValidationFailed • Error_Generic_WrongVersion • Error_Media_EmailAdministrationServerError • Error_Media_EmailConfigurationError • Error_Media_EmailCorporateEmailServerDown • Error_Media_EmailCorporateEmailServerError • Error_Media_EmailDatabaseConnectionFailed • Error_Media_EmailDatabaseError • Error_Media_EmailDataCorrupted • Error_Media_EmailInvalidAgent • Error_Media_EmailInvalidCallID • Error_Media_EmailMessageNotFound • Error_Media_EmailRoutingServerError • Error_Media_EmailServerTerminating
Notes:	None

7.23.2.2 Defer

```
Defer()
```

The Defer method defers an e-mail message that a user is handling. When an e-mail message is deferred, it is basically put on hold to be handled later. For example, if a user is waiting for more information, they may defer the e-mail message until that information is received.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following after creating the EmailCall:

```
Call oEmailCall.Defer
```

This defers the e-mail message that the user is working on.

Method:	Defer (of EmailCall)
Description:	Defers an e-mail message that a user is handling.
Parameters:	None
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_RequestTimedOut • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownError • Error_Generic_ValidationFailed • Error_Generic_WrongVersion • Error_Media_EmailAdministrationServerError • Error_Media_EmailConfigurationError • Error_Media_EmailCorporateEmailServerDown • Error_Media_EmailCorporateEmailServerError • Error_Media_EmailDatabaseConnectionFailed • Error_Media_EmailDatabaseError • Error_Media_EmailDataCorrupted • Error_Media_EmailInvalidAgent • Error_Media_EmailInvalidCallID • Error_Media_EmailMessageNotFound • Error_Media_EmailRoutingServerError • Error_Media_EmailServerTerminating
Notes:	None

7.23.2.3 Discard

```
Discard((Optional) ReasonKey as Long = 0)
```

The Discard method discards an e-mail message after it has been handled. To do this, the user must call Discard, passing along the appropriate reason key if required. An e-mail message should be discarded after it has been handled.

This method accepts the following parameters:

ReasonKey as Long	This is the reason key for discarding the e-mail message.

This method does not return anything.

To use this method, do the following after creating the EmailCall:

```
Call oEmailCall.Discard (txtDiscardReasons.Value)
```

This discards the e-mail message.

Method:	Discard (of EmailCall)
Description:	Discards an e-mail message after it has been handled.
Parameters:	ReasonKey(0) — the reason that the e-mail message is discarded.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_RequestTimedOut • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownError • Error_Generic_ValidationFailed • Error_Generic_WrongVersion • Error_Media_EmailAdministrationServerError • Error_Media_EmailConfigurationError • Error_Media_EmailCorporateEmailServerDown • Error_Media_EmailCorporateEmailServerError • Error_Media_EmailDatabaseConnectionFailed • Error_Media_EmailDatabaseError • Error_Media_EmailDataCorrupted • Error_Media_EmailInvalidAgent • Error_Media_EmailInvalidCallID • Error_Media_EmailMessageNotFound • Error_Media_EmailRoutingServerError • Error_Media_EmailServerTerminating
Notes:	None

7.23.2.4 ExternalConsult

```
ExternalConsult(ThirdPartyEmailAddress as IEmailAddress)
```

The ExternalConsult method transfers an e-mail message, with reply text and attachments, to a third party outside the contact center, with the expectation that the outside resource will reply back to the contact center. The e-mail message is placed into an externally-consulted state, and is reactivated and queued with reserve for the original user when the reply from the outside resource is received.

After the expert replies, you will receive a CallInformationUpdated event.

To distinguish between normal CallInformationUpdated events and the one that you receive after the expert replies, query for EmailCall and check for ExternalConsultCallID property. If it is not empty, create another EmailCall and query for the second EmailCall using ExternalConsultCallID as its CallID.

To use this method, do the following after creating the EmailCall:

```
Dim ThirdPartyEmailAddress
HiPathProCenterLibrary.EmailAddressCall
oEmailCall.ExternalConsult (ThirdPartyEmailAddress)
```

Method:	ExternalConsult (of EmailCall)
Description:	Transfers an e-mail message to a third party outside the contact center. The original e-mail message will be automatically deferred.
Parameters:	ThirdPartyEmailAddress — e-mail address of the third party expert you want to externally consult.

Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_RequestTimedOut • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownError • Error_Generic_ValidationFailed • Error_Generic_WrongVersion • Error_Media_EmailAdministrationServerError • Error_Media_EmailConfigurationError • Error_Media_EmailCorporateEmailServerDown • Error_Media_EmailCorporateEmailServerError • Error_Media_EmailDatabaseConnectionFailed • Error_Media_EmailDatabaseError • Error_Media_EmailDataCorrupted • Error_Media_EmailInvalidAgent • Error_Media_EmailInvalidCallID • Error_Media_EmailMessageNotFound • Error_Media_EmailRoutingServerError • Error_Media_EmailServerTerminating
Notes:	The user should ensure that the appropriate 'ReplyXXX' (the subject line ReplySubjectUTF8, the outgoing e-mail body text ReplyBodyUTF8 and vector of outgoing attachments ReplyAttachments) properties have been set.

7.23.2.5 ExternalTransfer

```
ExternalTransfer(ThirdPartyEmailAddress as IEmailAddress)
```

The ExternalTransfer method transfers an e-mail message, with reply text and attachments, to a third party outside the contact center. This will complete the original e-mail interaction, which will be flagged with a new 'externally transferred' state. The expectation is that the third party will reply directly to the customer.

To use this method, do the following after creating the EmailCall:

```
Dim ThirdPartyEmailAddress
HiPathProCenterLibrary.EmailAddressCall
oEmailCall.ExternalTransfer(ThirdPartyEmailAddress)
```

Method:	ExternalTransfer (of EmailCall)
Description:	Transfers an e-mail to a third party outside the contact center. This will complete the original e-mail interaction.
Parameters:	ThirdPartyEmailAddress — the e-mail address of the third party expert you want to externally transfer the e-mail message to.

Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_RequestTimedOut • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownError • Error_Generic_ValidationFailed • Error_Generic_WrongVersion • Error_Media_EmailAdministrationServerError • Error_Media_EmailConfigurationError • Error_Media_EmailCorporateEmailServerDown • Error_Media_EmailCorporateEmailServerError • Error_Media_EmailDatabaseConnectionFailed • Error_Media_EmailDatabaseError • Error_Media_EmailDataCorrupted • Error_Media_EmailInvalidAgent • Error_Media_EmailInvalidCallID • Error_Media_EmailMessageNotFound • Error_Media_EmailRoutingServerError • Error_Media_EmailServerTerminating
Notes:	The user should ensure that the appropriate 'ReplyXXX' (the subject line ReplySubjectUTF8, the outgoing e-mail body text ReplyBodyUTF8 and vector of outgoing attachments ReplyAttachments) properties have been set.

7.23.2.6 Reply

```
Reply()
```

The Reply method replies to an e-mail that has been assigned to a user. Before calling Reply, all 'ReplyXXX' (ReplyToEmailAddress, ReplyFromEmailAddress, ReplySubjectUTF8, ReplyBodyUTF8) properties should be set.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following after creating the EmailCall:

```
Call oEmailCall.Reply
```

This sends a reply e-mail to the ReplyToEmailAddress.

Method:	Reply (of EmailCall)
Description:	Replies to an e-mail that has been assigned to a user.
Parameters:	None
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_RequestTimedOut • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownError • Error_Generic_ValidationFailed • Error_Generic_WrongVersion • Error_Media_EmailAdministrationServerError • Error_Media_EmailConfigurationError • Error_Media_EmailCorporateEmailServerDown • Error_Media_EmailCorporateEmailServerError • Error_Media_EmailDatabaseConnectionFailed • Error_Media_EmailDatabaseError • Error_Media_EmailDataCorrupted • Error_Media_EmailInvalidAgent • Error_Media_EmailInvalidCallID • Error_Media_EmailMessageNotFound • Error_Media_EmailRoutingServerError • Error_Media_EmailServerTerminating
Notes:	The user should ensure that the appropriate 'ReplyXXX' properties have been set.

7.23.2.7 Requeue

```
Requeue(Queuekey as Long)
```

The Requeue method requeues an e-mail message to another queue. To do this, the user must pass the queue key that they want to requeue the e-mail message with.

This method accepts the following parameter:

Queuekey as Long	This is the key of the queue the user wants to requeue the e-mail message with.

This method does not return anything.

To use this method, do the following after creating the EmailCall:

```
Call oEmailCall.Requeue (txtQueue.Value)
```

This requeues the e-mail using the Queuekey parameter.

Method:	Requeue (of EmailCall)
Description:	Requeues an e-mail with a specific queue.
Parameters:	Queuekey — the key for the queue that you want to requeue the e-mail message to.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_RequestTimedOut • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownError • Error_Generic_ValidationFailed • Error_Generic_WrongVersion • Error_Media_EmailAdministrationServerError • Error_Media_EmailConfigurationError • Error_Media_EmailCorporateEmailServerDown • Error_Media_EmailCorporateEmailServerError • Error_Media_EmailDatabaseConnectionFailed • Error_Media_EmailDatabaseError • Error_Media_EmailDataCorrupted • Error_Media_EmailInvalidAgent • Error_Media_EmailInvalidCallID • Error_Media_EmailMessageNotFound • Error_Media_EmailRoutingServerError • Error_Media_EmailServerTerminating
Notes:	None

7.23.2.8 Resume

```
Resume()
```

The Resume method resumes an e-mail that has been deferred. When an e-mail is deferred, it is basically put on hold to be handled later. When the user decides its time to handle the e-mail again, they need to call Resume.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following after creating the EmailCall:

```
Call oEmailCall.Resume
```

This resumes the e-mail that the user had previously deferred.

Method:	Resume (of EmailCall)
Description:	Resumes an e-mail that a user had previously deferred.
Parameters:	None
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_RequestTimedOut • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownError • Error_Generic_ValidationFailed • Error_Generic_WrongVersion • Error_Media_EmailAdministrationServerError • Error_Media_EmailConfigurationError • Error_Media_EmailCorporateEmailServerDown • Error_Media_EmailCorporateEmailServerError • Error_Media_EmailDatabaseConnectionFailed • Error_Media_EmailDatabaseError • Error_Media_EmailDataCorrupted • Error_Media_EmailInvalidAgent • Error_Media_EmailInvalidCallID • Error_Media_EmailMessageNotFound • Error_Media_EmailRoutingServerError • Error_Media_EmailServerTerminating
Notes:	None

7.23.2.9 Transfer

```
Transfer (AgentKey as Long, Queuekey as Long)
```

The Transfer method transfers an e-mail message to another user. To do this, the user must pass the agent key that they want to transfer the e-mail message to, as well as a backup queue that they want to assign to the e-mail message.

This method accepts the following parameters:

AgentKey as Long	This is the agent key of another user that the original user wants to transfer the e-mail message to.
Queuekey as Long	This is the key of the queue the user wants to use as the backup queue for the e-mail message.

This method does not return anything.

To use this method, do the following after creating the EmailCall:

```
Call oEmailCall.Transfer (txtAgent.Value, txtQueue.Value)
```

This transfers the call to the user specified by the AgentKey parameter. If the user doesn't accept the e-mail message, the call will be reassigned using the Queuekey parameter.

Method:	Transfer (of EmailCall)
Description:	Transfers an e-mail message to another user.
Parameters:	• AgentKey — the key for the user who you want to transfer the e-mail message to. • Queuekey — the key for the queue that you want to use as the backup queue for the e-mail message.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_RequestTimedOut • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownError • Error_Generic_ValidationFailed • Error_Generic_WrongVersion • Error_Media_EmailAdministrationServerError • Error_Media_EmailConfigurationError • Error_Media_EmailCorporateEmailServerDown • Error_Media_EmailCorporateEmailServerError • Error_Media_EmailDatabaseConnectionFailed • Error_Media_EmailDatabaseError • Error_Media_EmailDataCorrupted • Error_Media_EmailInvalidAgent • Error_Media_EmailInvalidCallID • Error_Media_EmailMessageNotFound • Error_Media_EmailRoutingServerError • Error_Media_EmailServerTerminating
Notes:	None

7.24 EmailsHistoryQueryResultEvent

The EmailsHistoryQueryResultEvent object represents a collection of one or more EmailSummary objects.

Interfaces supported:

- IEmailsHistoryQueryResultEvent (default)
- IMediaEvent
- IXMLAccess

7.24.1 Properties

This section contains properties exposed through the EmailsHistoryQueryResultEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.24.1.1 _NewEnum

```
For Each oEmailSummary in colEmailsHistoryQueryResultEvent
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oEmailSummary as HiPathProCenterLibrary.EmailSummary
For Each oEmailsSummary in colEmailsHistoryQueryResultEvent
.
```

```
Next
```

Property:	_NewEnum (of EmailsHistoryQueryResultEvent)
Description:	Provides access to elements of the EmailsHistoryQueryResultEvent by means of operator FOREACH.
Returns:	oEmailSummary — For each item in the EmailsHistoryQueryResultEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the EmailsHistoryQueryResultEvent collection to support the 'For Each' iterator.

7.24.1.2 Count

The Count property returns the number of EmailSummary objects that exist in the EmailsHistoryQueryResultEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colEmailsHistoryQueryResultEvent.Count
```

Property:	Count (of EmailsHistoryQueryResultEvent)
Description:	The number of EmailSummary objects EmailsHistoryQueryResultEvent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.24.1.3 IsOverflow

The IsOverflow property returns true if number of elements in the event result object is greater than a "hidden" system parameter (5000 by default) value. If the number of rows returned in the result set is less than this hidden value, the IsOverflow property returns false.

To use this property, do the following:

```
Dim bOverflow as Boolean
```

```
bOverflow = colEmailsHistoryQueryResultEvent.IsOverflow
```

Property:	IsOverflow (of EmailsHistoryQueryResultEvent)
Description:	The IsOverflow property returns success or failure depending whether the number of rows in the result set exceeded a predefined maximum.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.24.1.4 Item

```
oEmailSummary = Item (callID as String)
```

The Item method finds an E-mailSummary object in the EmailsHistoryQueryResultEvent collection that is indexed by the callID. If an item is found in the collection at the callID value, the Item method returns the EmailSummary object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

callID as String	This is the callID for the item you are requesting.

This method returns the following parameters:

oEmailSummary as EmailSummary	This is the EmailSummary object for the specified callID value. This contains the e-mail details information.

To use this method, do the following:

```
Dim oEmailSummary as HiPathProCenterLibary.EmailSummary

Set oEmailSummary = colEmailsHistoryQueryResultSummary.Item
(callID)
```

This checks to see if the EmailSummary specified by the callID value exists. If it does, oEmailSummary will contain the EmailSummary object. If it does not exist in the EmailsHistoryQueryResultEvent, this will raise an error.

Method:	Item (of EmailsHistoryQueryResultEvent)
Description:	Finds an EmailSummary in the EmailsHistoryQueryResultEvent collection that is indexed by the call ID.
Parameters:	callID — the call ID for the EmailSummary that you are looking for.
Returns:	oEmailSummary — the EmailSummary object specified by the callID value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the EmailSummary object does not exist in the EmailsHistoryQueryResultEvent, the call will raise an error.

7.24.1.5 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a QueryAsync*EmailsHistory requests. This QueryID is uniquely generated every time the client calls QueryAsync*EmailsHistory methods.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colEmailsHistoryQueryResultEvent.QueryID
```

Property:	QueryID (of EmailsHistoryQueryResultEvent)
Description:	The QueryID associated with EmailsHistoryQueryResultEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.24.1.6 Result

The Result property returns enResult enumeration to indicate the success or failure of the previous e-mail messages history query.

To use this property, do the following:

```
Dim enResult as HiPathProCenterLibrary.enResults
enResult = colEmailsHistoryQueryResultEvent.Result
```

Property:	Result (of EmailsHistoryQueryResultEvent)
Description:	The Result property returns a success or failure return code of the associated e-mail messages history query.
Type:	enResults
Potential Errors:	None
Notes:	Read-only

7.25 EmailSummaries

The EmailSummaries object is a collection of the EmailSummary objects.

Interfaces Supported:

IEmailSummaries

IXMLAccess

7.25.1 Properties

This section contains properties exposed through the EmailSummaries interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.25.1.1 _NewEnum

```
For Each oEmailSummary in colEmailSummaries
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oEmailSummary As HiPathProCenterLibrary.EmailSummary
For Each oEmailSummary in m_colEmailSummaries
   txtTextBox.Text = txtTextBox.Text & _
   "CallID: " & _
   oEmailSummary.CallID & _
   "Subject: " & _
   oEmailSummary.Subject & _
   vbCrLf
```

```
Next
```

Property:	_NewEnum (of EmailSummaries)
Description:	Allows for the 'For Each' operator on the EmailSummaries collection.
Returns:	oEmailSummary — For each item in the EmailSummaries collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the EmailSummaries collection to support the 'For Each' iterator.

7.25.1.2 Count

The Count property returns the number of EmailSummary objects that exist in the EmailSummaries collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colEmailSummaries.Count
```

Property:	Count (of EmailSummaries)
Description:	The number of EmailSummary objects that are in the EmailSummaries collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.25.1.3 Item

```
oEmailSummary =
Item (CallID As String)
```

Use the Item method to find an EmailSummary object in the EmailSummaries collection that is indexed by the CallID value. If an item is found in the collection which matches the CallID value, the Item method returns the EmailSummary object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameter:

CallID As String	This is the CallID for the e-mail message being retrieved.

This method returns the following parameter:

oEmailSummary As EmailSummary — This is the EmailSummary indexed by CallID.

To use this method, do the following:

```
Dim oEmailSummary As HiPathProCenterLibary.EmailSummary
Set oEmailSummary = colEmailSummaries.Item (sCallID)
```

This checks to see if the EmailSummary specified by the CallID exists. If it does, oEmailSummary will contain the EmailSummary object. If it does not exist in the EmailSummaries collection, this will raise an error.

Method:	Item (of EmailSummary)
Description:	Finds a EmailSummary object in the EmailSummaries collection that is indexed by the CallID value.
Parameters:	CallID — the CallID for the EmailSummary that you are looking for.
Returns:	EmailSummary — the EmailSummary for the specified CallID used as a key.
Potential Errors:	• Error_Generic_ItemNotFound • Error_Generic_UnableToAllocateMemory
Notes:	If the EmailSummary object does not exist in our collection, an error is raised.

7.26 EmailSummary

The EmailSummary object contains summary information for an e-mail message.

Interfaces Supported:

EmailSummary2 (default)

IXMLAccess

7.26.1 Properties

This section contains properties exposed through the EmailSummary interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.26.1.1 CallID

The CallID property is the unique e-mail CallID. It is also the key for the EmailSummary in the EmailSummaries collection. This value is system assigned.

To use this property, do the following:

```
Dim sCallID As String
sCallID = oEmailSummary.CallID
```

Property:	CallID (of EmailSummary)
Description:	The unique e-mail CallID
Type:	String
Potential Errors:	None
Notes:	None

7.26.1.2 DeferTimeStamp

The DeferTimeStamp property contains the date/time e-mail message was last deferred. This property cannot be changed.

To use this property, do the following:

```
Dim dteDeferTime As Date
dteDeferTime = oEmailSummary.DeferTimeStamp
```

Property:	DeferTimeStamp (of EmailSummary)
Description:	The date and time the e-mail message was last deferred.
Type:	Date
Potential Errors:	None
Notes:	Read-only

7.26.1.3 FromEmailAddress

The FromEmailAddress property contains the original e-mail address that the e-mail was sent from. This property cannot be changed.

To use this property, do the following:

```
Dim oEmailAddress As HiPathProCenterLibrary.EmailAddressSet
oEmailAddress = oEmailSummary.FromEmailAddress
```

Property:	FromEmailAddress (of EmailSummary)
Description:	The original e-mail address that the e-mail message was sent from.

Type:	EmailAddress
Potential Errors:	Error_Generic_CreatingObject
Notes:	Read-only

7.26.1.4 QueueKey

The QueueKey property is the key for the call type in the database. This is a system generated value.

To use this property, do the following:

```
Dim lKey As Long
lKey = oEmailSummary.QueueKey
```

Property:	QueueKey (of EmailSummary)
Description:	The database table unique key for the Call Type object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.26.1.5 State

The State property is the state that the contact is in. This is one of the valid enumeration This method does not accept any parameters. from the enCallStates enumeration.

To use this property, do the following:

```
Dim enCallState As HiPathProCenterLibrary.enCallStates
enCallState = oEmailSummary.State
```

Property:	State (of EmailSummary)
Description:	The current state of the call.
Type:	enCallStates
Potential Errors:	None
Notes:	Read-only

7.26.1.6 Subject

The Subject property contains the original subject of the e-mail that was sent. This property cannot be changed.

To use this property, do the following:

```
Dim strSubject As String
strSubject = oEmailSummary.Subject
```

Property:	Subject (of EmailSummary)
Description:	The original subject of the e-mail that was sent.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.27 EstablishedEvent

The EstablishedEvent object is sent whenever a contact that is in the system is connected with a user.

Interfaces supported:

- IEstablishedEvent5 (default)
- IMediaEvent
- IXMLAccess

7.27.1 Properties

This section contains properties exposed through the EstablishedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.27.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oEstablishedEvent.CallID
```

Property:	CallID (of EstablishedEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.27.1.2 ConversationID

The ConversationID property determines which interactions are associated with the same call. In an e-mail message, there may be various messages sent back and forth between the customer and the user. Each of these e-mail messages will have a different call ID, but each e-mail message that is associated with the same e-mail thread will have the same conversation ID.

To use this property, do the following:

```
Dim strConversationID as String
strConversationID = oEstablishedEvent.ConversationID
```

Property:	ConversationID (of EstablishedEvent)
Description:	The conversation ID used to determine which interactions are associated with the same call.
Type:	String
Potential Errors:	None
Supported Media:	E-mail
Notes:	Read-only

7.27.1.3 FromParty

The FromParty property is the party that the call is coming from. In some instances, this will be the same as the OriginalFromParty, but in some scenarios (for example, a call being transferred by another user) this may be different. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oEstablishedEvent.FromParty
```

Property:	FromParty (of EstablishedEvent)
Description:	The from party when the call is queued.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.27.1.4 OriginalFromParty

The OriginalFromParty property is the original calling party that placed the call. For example, for a voice call, this is the ANI, for an e-mail message, this is the From e-mail address, and for a Web collaboration session, this is the customers IP address. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oEstablishedEvent.OriginalFromParty
```

Property:	OriginalFromParty (of EstablishedEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (ANI), e-mail (From e-mail address), Web collaboration (customer IP address)
Notes:	Read-only

7.27.1.5 OriginalToParty

The OriginalToParty property is the original destination of the call. For example, for a voice call, this is the DNIS, for an e-mail message, this is the To e-mail address, and for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oEstablishedEvent.OriginalToParty
```

Property:	OriginalToParty (of EstablishedEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (DNIS), e-mail (To e-mail address), Web collaboration (initial URL)
Notes:	Read-only

7.27.1.6 RedirectParty

The RedirectParty property returns the telephone number from which the call was last forwarded or deflected. This field is populated only when a redirect occurs and the system is connected to an OpenScape Voice, OpenScape 4000, or HiPath 4000 communication platform.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oEstablishedEvent.RedirectParty
```

Property:	Redirect (of EstablishedEvent)
Description:	The telephone number from which the call was last forwarded or deflected. This field is populated only when a redirect occurs and the system is connected to an OpenScape Voice, OpenScape 4000, or HiPath 4000 communication platform.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject Error_Media_PropertyNotSupportedForNonVoiceContact
Notes:	Read-only

7.27.1.7 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oEstablishedEvent.SequenceNumber
```

Property:	SequenceNumber (of EstablishedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.27.1.8 ToParty

The ToParty property is the party the call was transferred to so that the call could be queued. In some instances, this will be the same as the OriginalToParty, but in some scenarios (for example, a call being transferred by another user) this may be different. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oEstablishedEvent.ToParty
```

Property:	ToParty (of EstablishedEvent)
Description:	The to party the call was transferred to for the call to be queued.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.27.1.9 TrunkID

The TrunkID property is the device number used in the communication platform which identifies the trunk devices that connect the local communication platform to the public telephone network. If TrunkID is not available, this property will contain an empty string.

The OpenScape Voice communication platform does not support TrunkID. For internal calls TrunkID will be an empty string.

To use this property, do the following:

```
Dim strTruncID as String
strTruncID = oEstablishedEvent.TruncID
```

Property:	TrunkID (of EstablishedEvent)
Description:	The TrunkID.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.28 FailedEvent

The FailedEvent object is sent when an action that is attempted on a device (for example, an extension) has failed.

Interfaces supported:

- IFailedEvent3 (default)
- IMediaEvent
- IXMLAccess

NOTE: For FailedEvent with MediaType property set to MediaType_Callback, only the CallID property is available. All other properties return error code Error_Media_PropertyNotSupportedForCallback.

7.28.1 Properties

This section contains properties exposed through the FailedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.28.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oFailedEvent.CallID
```

Property:	CallID (of FailedEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice
Notes:	Read-only

7.28.1.2 FromParty

The FromParty property is the party that is placing the call. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oFailedEvent.FromParty
```

Property:	FromParty (of FailedEvent)
Description:	The party that is placing the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice
Notes:	Read-only

7.28.1.3 OriginalFromParty

The OriginalFromParty property is the original calling party that is involved in the call. For example, for a voice call, this is the ANI. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oFailedEvent.OriginalFromParty
```

Property:	OriginalFromParty (of FailedEvent)
Description:	The original calling party that is placing the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (ANI)
Notes:	Read-only

7.28.1.4 OriginalToParty

The OriginalToParty property is the original destination of the call. For example, for a voice call, this is the DNIS. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oFailedEvent.OriginalToParty
```

Property:	OriginalToParty (of FailedEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (DNIS)
Notes:	Read-only

7.28.1.5 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oFailedEvent.SequenceNumber
```

Property:	SequenceNumber (of FailedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.28.1.6 ToParty

The ToParty property is the party that is called. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oFailedEvent.ToParty
```

Property:	ToParty (of FailedEvent)
Description:	The party that is called.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice
Notes:	Read-only

7.29 ForwardingStateEvent

The ForwardingStateEvent object is sent whenever the state of the forwarding device changes.

NOTE: This object is supported only when the system is connected to an OpenScape Voice V7 R1 or V8 communication platform.

Interfaces supported:

- IForwardingStateEvent (default)

7.29.1 Properties

This section contains properties exposed through the ForwardingStateEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.29.1.1 Extension

Use the Extension property to store the extension that the user is logged on to.

To use this property, do the following:

```
Dim strExtension as String
strExtension = oForwardingStateEvent.Extension
```

Property:	Extension (of Agent)
Description:	The extension that the user is logged on to.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.29.1.2 ForwardingState

The ForwardingState property specifies whether call forwarding from the user extension to the preferred device is active or inactive.

To use this property, do the following:

```
Dim bForwardingState as Boolean
bForwardingState = oForwardingStateEvent.ForwardingState
```

Property:	ForwardingState (of Agent)
Description:	Specifies whether call forwarding from the user extension to the preferred device is active or inactive.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.29.1.3 PreferredDevice

The PreferredDevice property specifies the preferred device of a user.

To use this property, do the following:

```
Dim strPreferredDevice as String
strPreferredDevice = oForwardingStateEvent.PreferredDevice
```

Property:	PreferredDevice (of Agent)
Description:	The preferred device of the user.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.30 HeldEvent

The HeldEvent object is sent whenever a contact that is in the system is placed on hold. For voice calls, this is the traditional hold that a contact center user can place callers on to confer with other agents, and so on. For e-mail calls, this is when an e-mail is deferred to be handled later (perhaps they are waiting for information from another source, and so on).

Interfaces supported:

- IHeldEvent5 (default)
- IMediaEvent
- IXMLAccess

7.30.1 Properties

This section contains properties exposed through the HeldEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.30.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oHeldEvent.CallID
```

Property:	CallID (of HeldEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice, e-mail
Notes:	Read-only

7.30.1.2 ConversationID

The ConversationID property determines which interactions are associated with the same call. This is only used for an e-mail message. In an e-mail message, there may be various messages sent back and forth between the customer and the user. Each of these e-mail messages will have a different CallID, but each e-mail message that is associated with the same e-mail thread will have the same conversation ID.

To use this property, do the following:

```
Dim strConversationID as String
strConversationID = oHeldEvent.ConversationID
```

Property:	ConversationID (of HeldEvent)
Description:	The conversation ID used to determine which interactions are associated with the same call.
Type:	String
Potential Errors:	None
Supported Media:	E-mail
Notes:	Read-only

7.30.1.3 HeldParty

The HeldParty property is the party that has been placed on hold. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oHeldEvent.HeldParty
```

Property:	HeldParty (of HeldEvent)
Description:	The party that has been placed on hold.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail
Notes:	Read-only

7.30.1.4 HoldingParty

The HoldingParty property is the party that has placed the call on hold. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oHeldEvent.HoldingParty
```

Property:	HoldingParty (of HeldEvent)
Description:	The party that has placed the call on hold.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail
Notes:	Read-only

7.30.1.5 OriginalFromParty

The OriginalFromParty property is the original calling party that placed the call. For a voice call, this is the ANI; for an e-mail message, this is the From e-mail address; for a Web collaboration session, this is the customer's IP address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oHeldEvent.OriginalFromParty
```

Property:	OriginalFromParty (of HeldEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (ANI), e-mail (From e-mail address)
Notes:	Read-only

7.30.1.6 OriginalToParty

The OriginalToParty property is the original destination of the call. For a voice call, this is the DNIS; for an e-mail message, this is the To e-mail address; for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oHeldEvent.OriginalToParty
```

Property:	OriginalToParty (of HeldEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (DNIS), e-mail (To e-mail address)
Notes:	Read-only

7.30.1.7 Reason

The Reason property returns the held reason enumeration value.

To use this property, do the following:

```
Dim enReason as HiPathProCenterLibrary.enHoldReasons
enReason = oHeldEvent.Reason
```

Property:	Reason (of HeldEvent)
Description:	This property returns a hold reason
Type:	enHeldReasons
Potential Errors:	None
Notes:	Read-only

7.30.1.8 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oHeldEvent.SequenceNumber
```

Property:	SequenceNumber (of HeldEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.31 MediaBase

The MediaBase interface is implemented by any objects that are valid objects for performing functions with the MediaManager with (for example, VoiceCall). This non-creatable object provides a common method of passing these database objects into the MediaManager.

The use of this interface is implicit. When you have an object that supports the MediaBase interface, you can pass that object into the COM method that takes a MediaBase object as a parameter.

Interfaces supported:

- IMediaBase (default)

For example:

```
Private Sub cmdQuery_Click()
   m_oVoiceCall.CallID = txtCallID.Text
   Call g_oMediaManager.Query (m_oVoiceCall)
End Sub
```

This takes the VoiceCall interface and converts it under the covers to the MediaBase interface for the Query method to accept.

In most instances, this interface is not something that you would directly use, but instead is a convenient way to pass around grouped types. For example, you could write a method that would display all properties of the various MediaManager objects (that is, DisplayCallProperties (oMediaBase as MediaBase)).

7.31.1 Properties

This section contains properties exposed through the MediaBase interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.31.1.1 Type

The Type property returns the object type that the MediaBase object refers to, which can then be used to determine what to 'cast' the object to.

This property returns a value from the enMediaBaseObjectTypes enumeration described in Section 3.29, “enMediaBaseObjectTypes”, on page 87. This property is read-only, and is set internally in the parent object that the MediaBase interface is exposed through.

To use this property, do the following:

```
Dim enType as HiPathProCenterLibrary.enMediaBaseObjectType
enType = oMediaBase.Type
```

Property:	Type (of MediaBase)
Description:	The object type of the MediaBase object the property is exposed through.
Type:	enMediaBaseObjectTypes
Potential Errors:	None
Notes:	Read-only

7.32 MediaEvent

The MediaEvent is an interface that contains a summary of the event that has occurred. All events sent back from the MediaManager will support the MediaEvent interface. This non-creatable object also provides a common method of passing all events that occur in the MediaManager to the applications using the SDK. This enables various pieces of information to be returned from the MediaManager and some simple handling to be performed before determining how (and if) you want to handle this event.

Applications can get specific information about the event by querying for other interfaces that are supported by the event interface. To determine what detailed object type the event returned, you can use the ObjectType property, and then ask for the appropriate interface.

Interfaces supported:

- IMediaEvent (default)

For example:

```
Private Sub g_oMediaManager_EventOccurred
   (ByVal MediaEvent as HiPathProCenterLibrary.MediaEvent)
Select Case MediaEvent.ObjectType
   Case MediaEventObjectType_Disconnected
      ' Here since you have a Disconnected event,
      ' you query for the Disconnected event interface.
      Dim oDisconnectedEvent as
      HiPathProCenterLibrary.DisconnectedEvent
      Set oDisconnectedEvent = MediaEvent
      MsgBox oMediaEvent.CallID &
      " has been disconnected."
   Case MediaEventObjectType_ManagerStateChanged
      ' Here though you have a ManagerStateChanged event,
      ' so you query for the ManagerStateChanged interface.
      Dim oManagerStateChanged as
      HiPathProCenterLibrary.ManagerStateChangedEvent
      Set oManagerStateChanged = MediaEvent
      MsgBox "MediaManager is now " &
      oManagerStateChanged.State
   Default
      MsgBox "Unknown ObjectType"
End Select
End Sub
```

7.32.1 Properties

This section contains properties exposed through the MediaEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.32.1.1 Code

The Code property returns the event code of the event that has occurred in the MediaManager. In most instances this will be directly related to the ObjectType that is also contained in the MediaEvent interface. This property determines what has happened. Since the ObjectType refers to the object that is returned, they will be related.

This property returns a value from the enMediaEventCodes enumerations described in Section 3.30, "enMediaEventCodes", on page 89. This property is read-only, and is set internally in the MediaManager before it fires the event back through the EventOccurred event handler. For more information, see Section 7.1.3.1, "EventOccurred", on page 275.

To use this property, do the following:

```
Dim enCode as HiPathProCenterLibrary.enMediaEventCodes
enCode = oMediaEvent.Code
```

Property:	Code (of MediaEvent)
Description:	The code of what has occurred in the MediaManager.
Type:	enMediaEventCodes
Potential Errors:	None
Notes:	Read-only

7.32.1.2 EventType

The EventType property returns the event type that this event corresponds with. This event type is the event that was listened for this event to be sent. If this event is always going to be sent (Error or ManagerStateChanged event), then this property will be MediaEventType_NotSet.

This property returns a value from the enMediaEventTypes enumerations described in Section 3.32, "enMediaEventTypes", on page 98. This property is read-only, and is set internally in the MediaManager before it fires the event back through the EventOccurred event handler. For more information, see Section 7.1.3.1, "EventOccurred", on page 275.

To use this property, do the following:

```
Dim enEventType as HiPathProCenterLibrary.MediaEventTypes
enEventType = oMediaEvent.EventType
```

Property:	EventType (of MediaEvent)
Description:	The event type of the MediaManager event that is sent back.
Type:	enMediaEventTypes
Potential Errors:	None
Notes:	This determines what event group the object belongs to.

7.32.1.3 MediaType

The MediaType property returns the media type that the MediaEvent applies to.

This property returns a value from the enMediaTypes enumerations described in Section 3.34, “enMediaTypes”, on page 100. This property is read-only, and is set internally in the MediaManager before it fires the event back through the EventOccurred event handler. For more information, see Section 7.1.3.1, “EventOccurred”, on page 275.

To use this property, do the following:

```
Dim enMediaType as HiPathProCenterLibrary.enMediaTypes
enMediaType = oMediaEvent.MediaType
```

Property:	MediaType (of MediaEvent)
Description:	The media type of the MediaManager event that is sent back.
Type:	enMediaTypes
Potential Errors:	None
Notes:	None

7.32.1.4 ObjectType

The ObjectType property returns the object interface type of the detailed event information contained in the MediaEvent. In most instances this will be directly related to the Code that is also contained in the MediaEvent interface. This property determines what the detailed interface is that is contained in the MediaEvent.

This property returns a value from the enMediaEventObjectTypes enumerations described in Section 3.31, “enMediaEventObjectTypes”, on page 95. This property is read-only, and is set internally in the MediaManager before it fires the event back through the

EventOccurred event handler. For more information, see Section 7.1.3.1, "EventOccurred", on page 275.

To use this property, do the following:

```
Dim enObjectType as
HiPathProCenterLibrary.enMediaEventObjectTypes

enObjectType = oMediaEvent.ObjectType
```

Property:	ObjectType (of MediaEvent)
Description:	The detailed object type of the MediaManager event that is sent back.
Type:	enMediaEventObjectTypes
Potential Errors:	None
Notes:	This determines what object type you can query for more detailed event information.

7.32.1.5 Resource

The Resource property returns the resource that this event corresponds with. The resource is a string value that is specific for an event. This is the resource that is used when registering for an event through the ListenForEvent method.

To use this property, do the following:

```
Dim strResource as String

strResource = oMediaEvent.Resource
```

Property:	Resource (of MediaEvent)
Description:	The resource that relates to the MediaManager event that is sent back.
Type:	String
Potential Errors:	None
Notes:	This determines the resource the event is associated with.

7.33 NetworkReachedEvent

The NetworkReachedEvent object is sent out when a monitored resource (extension) calls an external device. This is usually only for multisite environments.

Interfaces supported:

- INetworkReachedEvent3 (default)
- IMediaEvent

- IXMLAccess

7.33.1 Properties

This section contains properties exposed through the NetworkReachedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.33.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oNetworkReachedEvent.CallID
```

Property:	CallID (of NetworkReachedEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice
Notes:	Read-only

7.33.1.2 FromParty

The FromParty property is the party that is placing the call. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oNetworkReachedEvent.FromParty
```

Property:	FromParty (of NetworkReachedEvent)
Description:	The party that is placing the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice
Notes:	Read-only

7.33.1.3 OriginalFromParty

The OriginalFromParty property is the original calling party that is placing the call. For example, for a voice call, this is the ANI. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oNetworkReachedEvent.OriginalFromParty
```

Property:	OriginalFromParty (of NetworkReachedEvent)
Description:	The original calling party that is placing the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (ANI)
Notes:	Read-only

7.33.1.4 OriginalToParty

The OriginalToParty property is the original destination of the call. For example, for a voice call, this is the DNIS. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oNetworkReachedEvent.OriginalToParty
```

Property:	OriginalToParty (of NetworkReachedEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (DNIS)
Notes:	Read-only

7.33.1.5 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oNetworkReachedEvent.SequenceNumber
```

Property:	SequenceNumber (of NetworkReachedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.33.1.6 ToParty

The ToParty property is the party that is called. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oNetworkReachedEvent.ToParty
```

Property:	ToParty (of NetworkReachedEvent)
Description:	The party that is called.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice
Notes:	Read-only

7.34 OffHookEvent

The OffHookEvent object signifies when a monitored device (for example, a telephone) has been taken off the hook. This could be for the start of an outgoing telephone call, or as part of a consultation.

All events are media-neutral. This means that they are used for all media types calls; voice, callback (in a future release), e-mail, and Web collaboration. Some events may not be used for all media types at this time, but do have the potential to be used for the other media types in the future.

Interfaces supported:

- IOffHookEvent3 (default)
- IMediaEvent
- IXMLAccess

7.34.1 Properties

This section contains properties exposed through the OffHookEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.34.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oOffHookEvent.CallID
```

Property:	CallID (of OffHookEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice
Notes:	Read-only

7.34.1.2 ForConsult

The ForConsult property is used when the OffHookEvent is generated because of a consultation (that is, the OffHookEvent is for a consultation). If this property is True, then the OffHookEvent is the result of the initiation of a consultation. If it is False, it is not.

To use this property, do the following:

```
Dim bForConsult as Boolean
bForConsult = oOffHookEvent.ForConsult
```

Property:	ForConsult (of OffHookEvent)
Description:	Whether or not the OffHookEvent is the result of a consultation (True) or not (False).
Type:	Boolean
Potential Errors:	None
Supported Media:	Voice
Notes:	Read-only

7.34.1.3 FromParty

The FromParty property is the party that is causing the off hook event to be generated. This can be used in two scenarios; when placing an outgoing call, and when initiating a consultation with another user. When placing an outgoing call, the FromParty and the OriginalFromParty will be the same. When initiating a consultation with another user, the FromParty will be the user initiating the consultation, where the OriginalFromParty will be the person that initially originated the call. This may not necessarily be the same as the FromParty property. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oOffHookEvent.FromParty
```

Property:	FromParty (of OffHookEvent)
Description:	The party that is causing the off hook event to be generated.
Type:	Party
Potential Errors:	None
Supported Media:	Voice
Notes:	Read-only

7.34.1.4 OriginalFromParty

The OriginalFromParty property is the original calling party that is placing the call. For example, for a voice call, this is the ANI. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oOffHookEvent.OriginalFromParty
```

Property:	OriginalFromParty (of OffHookEvent)
Description:	The original calling party that is placing the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (ANI)
Notes:	Read-only

7.34.1.5 OriginalToParty

The OriginalToParty property is the original destination of the call. For example, for a voice call, this is the DNIS. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oOffHookEvent.OriginalToParty
```

Property:	OriginalToParty (of OffHookEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (DNIS)
Notes:	Read-only

7.34.1.6 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oOffHookEvent.SequenceNumber
```

Property:	SequenceNumber (of OffHookEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.35 OriginatedEvent

The OriginatedEvent object is sent out after a number has been entered and the call is begun. For example, this occurs after a number has been dialed and you can hear the ringing.

All events are media-neutral. This means that they are used for all media types calls; voice, callback (in a future release), e-mail, and Web collaboration. Some events may not be used for all media types at this time, but do have the potential to be used for the other media types in the future.

Interfaces supported:

- IOriginatedEvent4 (default)
- IMediaEvent
- IXMLAccess

7.35.1 Properties

This section contains properties exposed through the OriginatedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.35.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oOriginatedEvent.CallID
```

Property:	CallID (of OriginatedEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice
Notes:	Read-only

7.35.1.2 FromParty

The FromParty property is the party that is placing the call. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oOriginatedEvent.FromParty
```

Property:	FromParty (of OriginatedEvent)
Description:	The party that is placing the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice
Notes:	Read-only

7.35.1.3 OriginalFromParty

The OriginalFromParty property is the original calling party that is placing the call. For example, for a voice call, this is the ANI. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oOriginatedEvent.OriginalFromParty
```

Property:	OriginalFromParty (of OriginatedEvent)
Description:	The original calling party that is placing the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (ANI)
Notes:	Read-only

7.35.1.4 OriginalToParty

The OriginalToParty property is the original destination of the call. For example, for a voice call, this is the DNIS. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oOriginatedEvent.OriginalToParty
```

Property:	OriginalToParty (of OriginatedEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (DNIS)
Notes:	Read-only

7.35.1.5 ReferenceCallID

The ReferenceCallID property is the original callback CallID. This property should be used to associate the OriginatedEvent with the callback call, which initiated this event.

To use this property, do the following:

```
Dim strReferenceCallID as String
strReferenceCallID = oOriginatedEvent.ReferenceCallID
```

Property:	ReferenceCallID (of OriginatedEvent)
Description:	The original callback CallID, which initiated the OriginatedEvent.
Type:	String
Potential Errors:	None
Supported Media:	Voice
Notes:	Read-only

7.35.1.6 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oOriginatedEvent.SequenceNumber
```

Property:	SequenceNumber (of OriginatedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.35.1.7 ToParty

The ToParty property is the party that is called. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oOriginatedEvent.ToParty
```

Property:	ToParty (of OriginatedEvent)
Description:	The party that is called.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice
Notes:	Read-only

7.36 Parties

The Parties object is a collection of the Party objects that contain information about different parties involved in a call.

Interfaces supported:

- IParties (default)
- IXMLAccess

7.36.1 Properties

This section contains properties exposed through the Parties interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.36.1.1 _NewEnum

```
For Each oParty in colParties
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oParty as HiPathProCenterLibrary.Party
For Each oParty in colParties
   txtTextBox.Text = txtTextBox.Text & _
   "Device: " & _
   oParty.Device & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the devices of the different parties in the collection.

Property:	_NewEnum (of Parties)
Description:	Allows for the 'For Each' operator on the Parties collection.
Parameters:	None
Returns:	OParty — For each item in the Parties collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the Parties collection to support the 'For Each' iterator.

7.36.1.2 Count

The Count property returns the number of Party objects that exist in the Parties collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colParties.Count
```

Property:	Count (of Parties)
Description:	The number of Party objects that are in the Parties collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.36.1.3 Item

```
oParty = Item (partyType as enPartyTypes, DeviceID as String)
```

The Item method finds a Party object in the Parties collection that is indexed by the partyType and DeviceID This method does not accept any parameters.. If an item is found in the collection at the partyType and DeviceID This method does not accept any parameters., the Item method returns the Party object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

partyType as enPartyTypes	This is the type of the Party object you are requesting.
DeviceID as String	This is the DeviceID of the Party object you are requesting.

This method returns the following parameters:

oParty as Party	This is the Party object for Party specified by the partyType and DeviceID This method does not accept any parameters..

To use this method, do the following:

```
Dim oParty as HiPathProCenterLibary.Party
Set oParty = colParties.Item (partyType,DeviceID)
```

This checks to see if the Party specified by the partyType and DeviceID exists. If it does, oParty will contain the Party object. If it does not exist in the Parties collection, this will raise an error.

Method:	Item (of Parties)
Description:	Finds a Party object in the Parties collection that is by the partyType and DeviceID This method does not accept any parameters..
Parameters:	• DeviceID — the DeviceID for the Party object that you are looking for. • partyType — the type for the Party object that you are looking for.
Returns:	oParty — the Party object that you have requested.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the Party does not exist in our collection, you raise an error.

7.37 Party

The Party object contains information about a specific party involved in a call. This party can represent a user, an external customer, an IVR, and so on.

Interfaces supported:

- IParty (default)
- IXMLAccess

7.37.1 Properties

This section contains properties exposed through the Party interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.37.1.1 AgentKey

The AgentKey property is the user, if any, that is associated with the party object.

To use this property, do the following:

```
Dim lAgentKey as Long
lAgentKey = oParty.AgentKey
```

Property:	AgentKey (of Party)
Description:	The user, if any, that is associated with the party object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.37.1.2 Device

The Device property describes what device, if any, is a part of the party.

To use this property, do the following:

```
Dim strDevice as String
strDevice = oParty.Device
```

Property:	Device (of Party)
Description:	The device that is part of the party.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.37.1.3 PartyType

The PartyType property describes the type of party object you have. This may also provide more information about the Device, depending on what the party object represents.

To use this property, do the following:

```
Dim enPartyType as enPartyTypes
enPartyType = oParty.PartyType
```

Property:	PartyType (of Party)
Description:	Describes the type of party object you have.

Type:	enPartyTypes
Potential Errors:	Error_Generic_ActionFailed
Notes:	Read-only

7.38 QueuedEvent

The QueuedEvent object is sent whenever a contact is queued in the system.

Interfaces supported:

- IQueuedEvent3 (default)
- IMediaEvent
- IXMLAccess

7.38.1 Properties

This section contains properties exposed through the QueuedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.38.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oQueuedEvent.CallID
```

Property:	CallID (of QueuedEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.38.1.2 ConversationID

The ConversationID property determines which interactions are associated with the same call. This is only used for an e-mail message. In an e-mail message, there may be various messages sent back and forth between the customer and the user. Each of these e-mail messages will have a different CallID, but each e-mail message that is associated with the same e-mail thread will have the same conversation ID.

To use this property, do the following:

```
Dim strConversationID as String
strConversationID = oQueuedEvent.ConversationID
```

Property:	ConversationID (of QueuedEvent)
Description:	The conversation ID used to determine which interactions are associated with the same call.
Type:	String
Potential Errors:	None
Supported Media:	E-mail
Notes:	Read-only

7.38.1.3 FromParty

The FromParty property is the party that the call is coming from. In some instances, this will be the same as the OriginalFromParty, but in some scenarios (for example, a call being transferred by another user) this may be different. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oQueuedEvent.FromParty
```

Property:	FromParty (of QueuedEvent)
Description:	The from party when the call is queued.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.38.1.4 OriginalFromParty

The OriginalFromParty property is the original calling party that placed the call. For example: for a voice call, this is the ANI; for an e-mail message, this is the From e-mail address; for a Web collaboration session, this is the customer's IP address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oQueuedEvent.OriginalFromParty
```

Property:	OriginalFromParty (of QueuedEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (ANI), e-mail (From e-mail address), Web collaboration (customer IP address)
Notes:	Read-only

7.38.1.5 OriginalToParty

The OriginalToParty property is the original destination of the call. For example: for a voice call, this is the DNIS; for an e-mail message, this is the To e-mail address; for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oQueuedEvent.OriginalToParty
```

Property:	OriginalToParty (of QueuedEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (DNIS), e-mail (To e-mail Address), Web collaboration (initial URL)
Notes:	Read-only

7.38.1.6 QueuedToParty

The QueuedToParty property is the party the call was queued to. In some instances, this will be the same as the OriginalToParty, but in some scenarios (for example, a call being transferred by another user) this may be different. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oQueuedEvent.QueuedToParty
```

Property:	QueuedToParty (of QueuedEvent)
Description:	The to party when the call is queued.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.38.1.7 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oQueuedEvent.SequenceNumber
```

Property:	SequenceNumber (of QueuedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.38.1.8 ToParty

The ToParty property is the party the call was transferred to so that the call could be queued. In some instances, this will be the same as the OriginalToParty, but in some scenarios (for example, a call being transferred by another user) this may be different. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oQueuedEvent.ToParty
```

Property:	ToParty (of QueuedEvent)
Description:	The to party the call was transferred to for the call to be queued.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail, Web collaboration
Notes:	Read-only

7.39 RetrievedEvent

The RetrievedEvent object is sent whenever a contact that has been placed on hold is retrieved from hold. For voice calls, this is the traditional hold that a contact center user can place callers on to confer with other agents, and so on. For e-mail calls, this is when an e-mail is deferred to be handled later (perhaps they are waiting for information from another source, and so on).

Interfaces supported:

- IRetrievedEvent3 (default)
- IMediaEvent
- IXMLAccess

7.39.1 Properties

This section contains properties exposed through the RetrievedEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.39.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oRetrievedEvent.CallID
```

Property:	CallID (of RetrievedEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice, e-mail
Notes:	Read-only

7.39.1.2 ConversationID

The ConversationID property determines which interactions are associated with the same call. In an e-mail message, there may be messages sent between the customer and the user. Each of these e-mail messages will have a different CallID, but each e-mail message that is associated with the same e-mail thread will have the same conversation ID.

To use this property, do the following:

```
Dim strConversationID as String
strConversationID = oRetrievedEvent.ConversationID
```

Property:	ConversationID (of RetrievedEvent)
Description:	The conversation ID used to determine which interactions are associated with the same call.
Type:	String
Potential Errors:	None
Supported Media:	E-mail
Notes:	Read-only

7.39.1.3 HeldParty

The HeldParty property is the party that has been placed on hold. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oRetrievedEvent.HeldParty
```

Property:	HeldParty (of RetrievedEvent)
Description:	The party that has been placed on hold.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail
Notes:	Read-only

7.39.1.4 OriginalFromParty

The OriginalFromParty property is the original calling party that placed the call. For example: for a voice call, this is the ANI; for an e-mail message, this is the From e-mail address; for a Web collaboration session, this is the customer's IP address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oRetrievedEvent.OriginalFromParty
```

Property:	OriginalFromParty (of RetrievedEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (ANI), e-mail (From e-mail address)
Notes:	Read-only

7.39.1.5 OriginalToParty

The OriginalToParty property is the original destination of the call. For example: for a voice call, this is the DNIS; for an e-mail message, this is the To e-mail address; for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on

page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oRetrievedEvent.OriginalToParty
```

Property:	OriginalToParty (of RetrievedEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (DNIS), e-mail (To e-mail address)
Notes:	Read-only

7.39.1.6 RetrievingParty

The RetrievingParty property is the party that is retrieving the call from hold. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oRetrievedEvent.RetrievingParty
```

Property:	RetrievingParty (of RetrievedEvent)
Description:	The party that has placed the call on hold.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail
Notes:	Read-only

7.39.1.7 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oRetrievedEvent.SequenceNumber
```

Property:	SequenceNumber (of RetrievedEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.40 ScheduledCallbackElement

The ScheduledCallbackElement object provides the information for a single scheduled callback.

Interfaces supported:

- IScheduledCallbackElement (default)
- IXMLAccess

7.40.1 Properties

This section contains properties exposed through the ScheduledCallbackElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.40.1.1 CallID

The CallID property is the unique callback CallID. It is also the key for the ScheduledCallbackElement in the ScheduledCallbacksQueryResultEvent collection.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oScheduledCallback.CallID
```

Property:	CallID (of ScheduledCallbackElement)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.40.1.2 CreatedByKey

The CreatedByKey property returns the key of the user that created the callback. If there is not key, this property returns negative one (–1).

To use this property, do the following:

```
Dim lCreatedByKey as Long
lCreatedByKey = oScheduledCallbackElement.CreatedByKey
```

Property:	CreatedByKey (of ScheduledCallbackElement)
Description:	The key of the user that created the callback.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.40.1.3 CustomerName

The CustomerName property is the name of the customer to be called back.

To use this property, do the following:

```
Dim strCustomerName as String
strCustomerName = oScheduledCallbackElement.CustomerName
```

Property:	CustomerName (of ScheduledCallbackElement)
Description:	The name of the customer to be called back.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.40.1.4 Description

The Description property is the Description that is associated with this ScheduledCallbackElement object.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oScheduledCallbackElement.Description
```

Property:	Description (of ScheduledCallbackElement)
Description:	The Description that is associated with this ScheduledCallbackElement object.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.40.1.5 NextScheduledPhoneNumber

The NextScheduledPhoneNumber property returns the next schedule telephone number.

To use this property, do the following:

```
Dim strNextScheduledPhoneNumber as String

strNextScheduledPhoneNumber =
oScheduledCallbackElement.NextScheduledPhoneNumber
```

Property:	NextScheduledPhoneNumber (of ScheduledCallbackElement)
Description:	The next scheduled telephone number.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.40.1.6 NextScheduledStart

The NextScheduledStart property returns the next scheduled start time.

To use this property, do the following:

```
Dim dteNextScheduledStart as Date

dteNextScheduledStart =
oScheduledCallbackElement.NextScheduledStart
```

Property:	NextScheduledStart (of ScheduledCallbackElement)
Description:	The next scheduled start time.
Type:	Date
Potential Errors:	None
Notes:	Read-only

7.40.1.7 Origin

The Origin property is the Origin of the callback. This value could be: IVR, Client Desktop, Web, Customer Abandoned, Import File or SDK.

To use this property, do the following:

```
Dim enOrigin as HiPathProCenterLibrary.enCallbackOrigins
enOrigin = oScheduledCallbackElement.Origin
```

Property:	Origin (of ScheduledCallbackElement)
Description:	The Origin of the callback.
Type:	enCallbackOrigins
Potential Errors:	None
Notes:	Read-only

7.40.1.8 Priority

The Priority property returns the priority that is associated with this ScheduledCallbackElement object.

To use this property, do the following:

```
Dim sPriority as Integer
sPriority = oScheduledCallbackElement.Priority
```

Property:	Priority (of ScheduledCallbackElement)
Description:	The priority that is associated with this ScheduledCallbackElement object.
Type:	Short
Potential Errors:	None
Notes:	Read-only

7.40.1.9 QueueKey

The QueueKey property is the key of the queue entity associated with this scheduled callback.

To use this property, do the following:

```
Dim lQueueKey as Long
lQueueKey = oScheduledCallbackElement.QueueKey
```

Property:	QueueKey (of ScheduledCallbackElement)
Description:	The key of the queue entity associated with this scheduled callback.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.41 ScheduledCallbacksQueryResultEvent

The ScheduledCallbacksQueryResultEvent object consists of a collection of one or more scheduled callbacks. Each scheduled callback contains the summary for one scheduled callback.

Interfaces supported:

- IScheduledCallbacksQueryResultEvent (default)
- IMediaEvent
- IXMLAccess

7.41.1 Properties

This section contains properties exposed through the ScheduledCallbacksQueryResultEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.41.1.1 _NewEnum

```
For Each oScheduledCallback in
colScheduledCallbacksQueryResultEvent

   ...

Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oScheduledCallback as
HiPathProCenterLibrary.ScheduledCallback

For Each oScheduledCallback in
colScheduledCallbacksQueryResultEvent

   txtTextBox.Text = txtTextBox.Text & _

   "ContactID: " & _

   oScheduledCallback.ContactID & vbCrLf

Next
```

This sets the Text property of the txtTextBox to contain all the ContactIDs in the ScheduledCallbacksQueryResultEvent.

Property:	_NewEnum (of ScheduledCallbacksQueryResultEvent)
Description:	Provides access to elements of the ScheduledCallbacksQueryResultEvent by means of operator FOREACH.
Returns:	OScheduledCallback — for each item in the ScheduledCallbacksQueryResultEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the ScheduledCallbacksQueryResultEvent collection to support the 'For Each' iterator.

7.41.1.2 Count

The Count property returns the number of ScheduledCallback objects that exist in the ScheduledCallbacksQueryResultEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colScheduledCallbacksQueryResultEvent.Count
```

Property:	Count (of ScheduledCallbacksQueryResultEvent)
Description:	The number of ScheduledCallback objects in ScheduledCallbacksQueryResultEvent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.41.1.3 Item

```
oScheduledCallback = Item (ContactID as String)
```

The Item method finds a ScheduledCallback object in the ScheduledCallbacksQueryResultEvent collection that is indexed by the ContactID. If an item is found in the collection at the ContactID value, the Item method returns the ScheduledCallback object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

ContactID as String	This is the ContactID for the ScheduledCallback item you are requesting.

This method returns the following parameters:

oScheduledCallback as ScheduledCallback	This is the ScheduledCallback object specified by the ContactID value. This contains the statistics for the ScheduledCallback.

To use this method, do the following:

```
Dim oScheduledCallback as
HiPathProCenterLibary.ScheduledCallback
Set oScheduledCallback =
colScheduledCallbacksQueryResultEvent.Item (ContactID)
```

This checks to see if the ScheduledCallback specified by the ContactID value exists. If it does, oScheduledCallback will contain the ScheduledCallback object. If it does not exist in the ScheduledCallbacksQueryResultEvent, this will raise an error.

Property:	Item (of ScheduledCallbacksQueryResultEvent)
Description:	Finds a ScheduledCallback in the ScheduledCallbacksQueryResultEvent collection that is indexed ContactID.
Parameters:	ContactID — the ContactID for the ScheduledCallback that you are looking for.
Returns:	oScheduledCallback — the ScheduledCallback object specified by the ContactID value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the ScheduledCallback object does not exist in the ScheduledCallbacksQueryResultEvent, It will raise an error.

7.41.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a QueryAsyncScheduledCallbacks () request. This QueryID is uniquely generated every time the client calls QueryAsyncScheduledCallbacks () method.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colScheduledCallbacksQueryResultEvent.QueryID
```

Property:	QueryID (of ScheduledCallbacksQueryResultEvent)
Description:	The QueryID associated with ScheduledCallbacksQueryResultEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.41.1.5 Result

The Result property is the result of the query of scheduled callback from the Callback Server.

To use this property, do the following:

```
Dim enResult as HiPathProCenterLibrary.enResults
enResult = colScheduledCallbacksQueryResultEvent.Result
```

Property:	Result (of ScheduledCallbacksQueryResultEvent)
Description:	The result of the query of scheduled callback from the Callback Server.
Type:	enResults
Potential Errors:	None
Notes:	Read-only

7.42 ServicesEnabled

The ServicesEnabled represents the CTI services available for an Agent's monitored Extension. This event is sent whenever there is a Line event or Agent event at an Agent's extension and CTI services information changes in the system.

Interfaces supported:

- IServicesEnabled (default)
- MediaBase
- IBinaryAccess
- IXMLAccess

7.42.1 Properties

The following properties are exposed through the ServicesEnabled interface.

7.42.1.1 Answer

The Answer property represents Answer Call service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bAnswerCall as Boolean
bAnswerCall = oServicesEnabled.Answer
```

Property:	Answer (of ServicesEnabled)
Description:	The Answer Call service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.2 Available

The Available property represents the Available service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bAvailable as Boolean
bAvailable = oServicesEnabled.Available
```

Property:	Available (of ServicesEnabled)
Description:	The Available service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.3 Conference

The Conference property represents the Conference Call service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bConferenceCall as Boolean
bConferenceCall = oServicesEnabled.Conference
```

Property:	Conference (of ServicesEnabled)
Description:	The Conference Call service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.4 Consult

The Consult property represents the Consult Call service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bConsultCall as Boolean
bConsultCall = oServicesEnabled.Consult
```

Property:	Consult (of ServicesEnabled)
Description:	The Consult Call service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.5 Dial

The Dial property represents the Make Call service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bMakeCall as Boolean
bMakeCall = oServicesEnabled.Dial
```

Property:	Dial (of ServicesEnabled)
Description:	The Make Call service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.6 Disconnect

The Disconnect property represents the Disconnect Call service availability for an Agent's monitored Extension

To use this property, do the following:

```
Dim bDisconnectCall as Boolean
bDisconnectCall = oServicesEnabled.Disconnect
```

Property:	Disconnect (of ServicesEnabled)
Description:	The Disconnect Call service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.7 Divert

The Divert property represents the Divert Call service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bDivertCall as Boolean
bDivertCall = oServicesEnabled.Divert
```

Property:	Divert (of ServicesEnabled)
Description:	The Divert Call service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.8 Extension

The Extension property represents the Agent's monitored Extension.

To use this property, do the following:

```
Dim strExtension as String
strExtension = oServicesEnabled.Extension
```

Property:	Extension (of ServicesEnabled)
Description:	The Agent's monitored Extension.
Type:	String
Potential Errors:	None
Notes:	Read/Write

7.42.1.9 Hold

The Hold property represents the Hold Call service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bHoldCall as Boolean
bHoldCall = oServicesEnabled.Hold
```

Property:	Hold (of ServicesEnabled)
Description:	The Hold Call service availability for Agent's monitored Extension.

Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.10 Logoff

The Logoff property represents the Logoff service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bLogoff as Boolean
bLogoff = oServicesEnabled.Logoff
```

Property:	Logoff (of ServicesEnabled)
Description:	The Logoff service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.11 Logon

The Logon property represents the Logon service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bLogon as Boolean
bLogon = oServicesEnabled.Logon
```

Property:	Logon (of ServicesEnabled)
Description:	The Logon service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.12 Reconnect

The Reconnect property represents the Reconnect Call service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bReconnectCall as Boolean
bReconnectCall = oServicesEnabled.Reconnect
```

Property:	Reconnect (of ServicesEnabled)
Description:	The Reconnect Call service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.13 Retrieve

The Retrieve property represents the Retrieve Call service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bRetrieveCall as Boolean
bRetrieveCall = oServicesEnabled.Retrieve
```

Property:	Retrieve (of ServicesEnabled)
Description:	The Retrieve Call service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.14 Transfer

The Transfer property represents the Transfer Call service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bTransferCall as Boolean
bTransferCall = oServicesEnabled.Transfer
```

Property:	Transfer (of ServicesEnabled)
Description:	The Transfer Call service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.15 Unavailable

The Unavailable property represents the Unavailable service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bUnavailable as Boolean
bUnavailable = oServicesEnabled.Unavailable
```

Property:	Unavailable (of ServicesEnabled)
Description:	The Unavailable service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.42.1.16 Work

The Work property represents the Work service availability for an Agent's monitored Extension.

To use this property, do the following:

```
Dim bWork as Boolean
bWork = oServicesEnabled.Work
```

Property:	Work (of ServicesEnabled)
Description:	The Work service availability for Agent's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43 ServicesEnabledEvent

The ServicesEnabledEvent represents the CTI services available for a user's monitored Extension. This event is sent whenever there is a Line event or user event at a user's extension and CTI services information changes in the system.

Interfaces supported:

- IServicesEnabledEvent (default)
- IMediaEvent
- IXMLAccess

7.43.1 Properties

This section contains properties exposed through the ServicesEnabledEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.43.1.1 Answer

The Answer property represents Answer Call service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bAnswerCall as Boolean
bAnswerCall = oServicesEnabledEvent.Answer
```

Property:	Answer (of ServicesEnabledEvent)
Description:	The database table unique key for the Agent object.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.2 Available

The Available property represents the Available service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bAvailable as Boolean
bAvailable = oServicesEnabledEvent.Available
```

Property:	Available (of ServicesEnabledEvent)
Description:	The Available service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.3 Conference

The Conference property represents the Conference Call service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bConferenceCall as Boolean
bConferenceCall = oServicesEnabledEvent.Conference
```

Property:	Conference (of ServicesEnabledEvent)
Description:	The Conference Call service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.4 Consult

The Consult property represents the Consult Call service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bConsultCall as Boolean
bConsultCall = oServicesEnabledEvent.Consult
```

Property:	Consult (of ServicesEnabledEvent)
Description:	The Consult Call service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.5 Dial

The Dial property represents the Make Call service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bMakeCall as Boolean
bMakeCall = oServicesEnabledEvent.Dial
```

Property:	Dial (of ServicesEnabledEvent)
Description:	The Make Call service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.6 Disconnect

The Disconnect property represents the Disconnect Call service availability for a user's monitored Extension

To use this property, do the following:

```
Dim bDisconnectCall as Boolean
bDisconnectCall = oServicesEnabledEvent.Disconnect
```

Property:	Disconnect (of ServicesEnabledEvent)
Description:	The Disconnect Call service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.7 Divert

The Divert property represents the Divert Call service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bDivertCall as Boolean
bDivertCall = oServicesEnabledEvent.Divert
```

Property:	Divert (of ServicesEnabledEvent)
Description:	The Divert Call service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.8 Hold

The Hold property represents the Hold Call service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bHoldCall as Boolean
bHoldCall = oServicesEnabledEvent.Hold
```

Property:	Hold (of ServicesEnabledEvent)
Description:	The Hold Call service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.9 Logoff

The Logoff property represents the Logoff service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bLogoff as Boolean
bLogoff = oServicesEnabledEvent.Logoff
```

Property:	Logoff (of ServicesEnabledEvent)
Description:	The Logoff service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.10 Logon

The Logon property represents the Logon service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bLogon as Boolean
bLogon = oServicesEnabledEvent.Logon
```

Property:	Logon (of ServicesEnabledEvent)
Description:	The Logon service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.11 Reconnect

The Reconnect property represents the Reconnect Call service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bReconnectCall as Boolean
bReconnectCall = oServicesEnabledEvent.Reconnect
```

Property:	Reconnect (of ServicesEnabledEvent)
Description:	The Reconnect Call service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.12 Retrieve

The Retrieve property represents the Retrieve Call service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bRetrieveCall as Boolean
bRetrieveCall = oServicesEnabledEvent.Retrieve
```

Property:	Retrieve (of ServicesEnabledEvent)
Description:	The Retrieve Call service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.13 Transfer

The Transfer property represents the Transfer Call service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bTransferCall as Boolean
bTransferCall = oServicesEnabledEvent.Transfer
```

Property:	Transfer (of ServicesEnabledEvent)
Description:	The Transfer Call service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.14 Unavailable

The Unavailable property represents the Unavailable service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bUnavailable as Boolean
bUnavailable = oServicesEnabledEvent.Unavailable
```

Property:	Unavailable (of ServicesEnabledEvent)
Description:	The Unavailable service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.43.1.15 Work

The Work property represents the Work service availability for a user's monitored Extension.

To use this property, do the following:

```
Dim bWork as Boolean
bWork = oServicesEnabledEvent.Work
```

Property:	Work (of ServicesEnabledEvent)
Description:	The Work service availability for the user's monitored Extension.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.44 TransferredEvent

The TransferredEvent is sent whenever a contact is transferred from one device (for example, a user) to another device.

Interfaces supported:

- ITransferredEvent4 (default)
- IMediaEvent
- IXMLAccess

7.44.1 Properties

This section contains properties exposed through the TransferredEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.44.1.1 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oTransferredEvent.CallID
```

Property:	CallID (of TransferredEvent)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Supported Media:	Voice, e-mail
Notes:	Read-only

7.44.1.2 ConversationID

The ConversationID property determines which interactions are associated with the same call. , this is only used for an e-mail message. In an e-mail message, there may be various messages sent back and forth between the customer and the user. Each of these e-mail messages will have a different CallID, but each e-mail message that is associated with the same e-mail thread will have the same conversation ID.

To use this property, do the following:

```
Dim strConversationID as String
strConversationID = oTransferredEvent.ConversationID
```

Property:	ConversationID (of TransferredEvent)
Description:	The conversation ID used to determine which interactions are associated with the same call.
Type:	String
Potential Errors:	None
Supported Media:	E-mail
Notes:	Read-only

7.44.1.3 FromParty

The FromParty property is the party that the call is transferred from. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oTransferredEvent.FromParty
```

Property:	FromParty (of TransferredEvent)
Description:	The party the call is transferred from.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail
Notes:	Read-only

7.44.1.4 OriginalFromParty

The OriginalFromParty property is the original calling party that placed the call. For example: for a voice call, this is the ANI; for an e-mail message, this is the From e-mail address; for a Web collaboration session, this is the customers IP address. For more information, see Section 7.37, "Party", on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oTransferredEvent.OriginalFromParty
```

Property:	OriginalFromParty (of TransferredEvent)
Description:	The original calling party that placed the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (ANI), e-mail (From e-mail address)
Notes:	Read-only

7.44.1.5 OriginalToParty

The OriginalToParty property is the original destination of the call. For example: for a voice call, this is the DNIS; for an e-mail message, this is the To e-mail address; for a Web collaboration session, this is the initial URL address. For more information, see Section 7.37, "Party", on

page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oTransferredEvent.OriginalToParty
```

Property:	OriginalToParty (of TransferredEvent)
Description:	The original destination of the call.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice (DNIS), e-mail (To e-mail Address)
Notes:	Read-only

7.44.1.6 RedirectParty

The RedirectParty property returns the telephone number from which the call was last forwarded or deflected. This field is populated only when a redirect occurs and the system is connected to an OpenScape Voice, OpenScape 4000, or HiPath 4000 communication platform.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oTransferredEvent.RedirectParty
```

Property:	Redirect (of TransferredEvent)
Description:	The telephone number from which the call was last forwarded or deflected. This field is populated only when a redirect occurs and the system is connected to an OpenScape Voice, OpenScape 4000, or HiPath 4000 communication platform.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject Error_Media_PropertyNotSupportedForNonVoiceContact
Notes:	Read-only

7.44.1.7 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oTransferredEvent.SequenceNumber
```

Property:	SequenceNumber (of TransferredEvent)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.44.1.8 ToParty

The ToParty property is the party the call is transferred to. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oTransferredEvent.ToParty
```

Property:	ToParty (of TransferredEvent)
Description:	The party the call is transferred to.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail
Notes:	Read-only

7.44.1.9 TransferredParty

The TransferredParty property is the party that is transferred. , this is only applicable for voice calls. For more information, see Section 7.37, “Party”, on page 445.

To use this property, do the following:

```
Dim oParty as HiPathProCenterLibrary.Party
oParty = oTransferredEvent.TransferredParty
```

Property:	TransferredParty (of TransferredEvent)
Description:	The party that is transferred.
Type:	Party
Potential Errors:	Error_Generic_CreatingObject
Supported Media:	Voice, e-mail
Notes:	Read-only

7.45 UserEventHandlingState

The UserEventHandlingState represents the state of a user while handling a contact.

Interfaces supported:

- IUserEventHandlingState2 (default)
- IXMLAccess

7.45.1 Properties

This section contains properties exposed through the UserEventHandlingState interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.45.1.1 CallID

The CallID property is the CallID for the contact that the user is handling.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oUserEventHandlingState.CallID
```

Property:	CallID (of UserEventHandlingState)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.45.1.2 HandlingState

The HandlingState property specifies the state of a user while handling a contact.

To use this property, do the following:

```
Dim enHandlingStates as HiPathProCenterLibrary.enHandlingStates
enHandlingStates = oUserEventHandlingState.HandlingState
```

Property:	HandlingState (of UserEventHandlingState)
Description:	The state of a user while handling a contact.
Type:	enHandlingStates
Potential Errors:	Error_Generic_UnknownState
Notes:	Read-only

7.45.1.3 HandlingStateReasonKey

The HandlingStateReasonKey property is the reason for a user being in a particular handling state. The HandlingStateReasonKey property is currently only used for Post-processing reasons.

To use this property, do the following:

```
Dim lHandlingStateReasonKey as Long

lHandlingStateReasonKey =
oUserEventHandlingState.HandlingStateReasonKey
```

Property:	HandlingStateReasonKey (of UserEventHandlingState)
Description:	The reason for a user being in a particular handling state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.45.1.4 IsPrimary

The IsPrimary property specifies whether the contact being handled is the primary contact.

To use this property, do the following:

```
Dim bIsPrimary as Boolean

bIsPrimary = oUserEventHandlingState.IsPrimary
```

Property:	IsPrimary (of UserEventHandlingState)
Description:	Returns true or false depending on whether the contact being handled is the primary contact.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.45.1.5 MediaType

The MediaType property is the media type for the contact that the user is handling.

To use this property, do the following:

```
Dim enMediaType as HiPathProCenterLibrary.enMediaTypes
enMediaType = oUserEventHandlingState.MediaType
```

Property:	MediaType (of UserEventHandlingState)
Description:	The media type of the contact that the user is handling.
Type:	enMediaTypes
Potential Errors:	Error_Media_AgentUnknownMediaType
Notes:	Read-only

7.45.1.6 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oUserEventHandlingState.SequenceNumber
```

Property:	SequenceNumber (of UserEventHandlingState)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.46 UserEventHandlingStates

The UserEventHandlingStates object is a collection of UserEventHandlingState objects.

Interfaces supported:

- IUserEventHandlingStates2 (default)
- IXMLAccess

7.46.1 Properties

This section contains properties exposed through the UserEventHandlingStates interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.46.1.1 _NewEnum

```
For Each oUserEventHandlingState in colUserEventHandlingStates
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oUserEventHandlingState as
HiPathProCenterLibrary.UserEventHandlingState
For Each oUserEventHandlingState in m_colUserEventHandlingStates
   txtTextBox.Text = txtTextBox.Text & _
      " CallID: " & _
      oUserEventHandlingState.CallID & _
      " MediaType: " & _
      oUserEventHandlingState.MediaType & _
      " HandlingState: " & _
      oUserEventHandlingState.HandlingState &
      vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain a collection of handling states for the user while they are handling a contact.

Property:	_NewEnum (of UserEventHandlingStates)
Description:	Allows for the 'For Each' operator on the UserEventHandlingStates collection.

Returns:	oUserEventHandlingState — For each item in the UserEventHandlingStates collection until all items have been returned.
Potential Errors:	None
Notes:	This method allows the UserEventHandlingStates collection to support the 'For Each' iterator.

7.46.1.2 Count

The Count property returns the number of UserEventHandlingState objects that exist in the UserEventHandlingStates collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colUserEventHandlingStates.Count
```

Property:	Count (of UserEventHandlingStates)
Description:	The number of UserEventHandlingState objects that are in the UserEventHandlingStates collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.46.1.3 Item

```
oUserEventHandlingState = Item (ContactID as String,
SequenceNumber as Long)
```

The Item method finds a UserEventHandlingState object in the UserEventHandlingStates collection that is indexed by the ContactID and SequenceNumber. If an item is found in the collection, the Item method returns the UserEventHandlingState object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameter:

ContactID as String	This is the ContactID for the period you are requesting.
SequenceNumber as Long	This is the SequenceNumber for the handling state you are requesting.

This method returns the following parameter:

oUserEventHandlingState as UserEventHandlingState	This is the UserEventHandlingState object that is indexed by the ContactID and SequenceNumber.

To use this method, do the following:

```
Dim oUserEventHandlingState as
HiPathProCenterLibary.UserEventHandlingState
Set oUserEventHandlingState = colUserEventHandlingStates.Item
(strContactID, lSequenceNumber)
```

This checks to see if the UserEventHandlingState exists. If it does, oUserEventHandlingState will contain the UserEventHandlingState object. If it does not exist in the UserEventHandlingStates collection, this will raise an error.

Method:	Item (of UserEventHandlingStates)
Description:	Finds an UserEventHandlingState object in the UserEventHandlingStates collection that is indexed by the ContactID and SequenceNumber.
Parameters:	ContactID — The ContactID value for the UserEventHandlingState that you are requesting. SequenceNumber — The SequenceNumber for the UserEventHandlingState that you are requesting.
Returns:	oUserEventHandlingState — The user status for the user specified.
Potential Errors:	None
Notes:	If the UserEventHandlingState object for the user does not exist in our collection, you raise an error.

7.47 UserHandlingState

The UserHandlingState represents the state of a user while handling a contact.

Interfaces supported:

- IUserHandlingState2 (default)
- IXMLAccess

7.47.1 Properties

This section contains properties exposed through the UserHandlingState interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.47.1.1 CallID

The CallID property is the CallID for the contact that the user is handling.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oUserHandlingState.CallID
```

Property:	CallID (of UserHandlingState)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.47.1.2 ContactMediaType

The ContactMediaType property is the media type for the contact that the user is handling.

To use this property, do the following:

```
Dim enMediaType as HiPathProCenterLibrary.enMediaTypes
enMediaType = oUserHandlingState.ContactMediaType
```

Property:	ContactMediaType (of UserHandlingState)
Description:	The media type of the contact that the user is handling.
Type:	enMediaTypes
Potential Errors:	Error_Media_AgentUnknownMediaType
Notes:	Read-only

7.47.1.3 HandlingState

The HandlingState property specifies the state of a user while handling a contact.

To use this property, do the following:

```
Dim enHandlingStates as HiPathProCenterLibrary.enHandlingStates
enHandlingStates = oUserHandlingState.HandlingState
```

Property:	HandlingState (of UserHandlingState)
Description:	The state of a user while handling a contact.

Type:	enHandlingStates
Potential Errors:	Error_Generic_UnknownState
Notes:	Read-only

7.47.1.4 HandlingStateReasonKey

The HandlingStateReasonKey property is the reason for a user being in a particular handling state. The HandlingStateReasonKey property is currently only used for Post-processing reasons.

To use this property, do the following:

```
Dim lHandlingStateReasonKey as Long

lHandlingStateReasonKey =
oUserHandlingState.HandlingStateReasonKey
```

Property:	HandlingStateReasonKey (of UserHandlingState)
Description:	The user's current handling state reason.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.47.1.5 IsMandatoryWrapUp

If the IsMandatoryWrapUpRequired property of the Agent object is set to true, then the IsMandatoryWrapUp property can return true or false depending on the state of the user while handling the contact.

For example, if the IsMandatoryWrapUpRequired property is set and the user is in the Post-processing handling state, then the IsMandatoryWrapUp property returns true.

To use this property, do the following:

```
Dim bMandatoryWrapUp as Boolean

bMandatoryWrapUp = oUserHandlingState.IsMandatoryWrapUp
```

Property:	IsMandatoryWrapUp (of UserHandlingState)
Description:	Returns true or false depending on the state of the user while handling the contact.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.47.1.6 IsPrimary

The IsPrimary property specifies whether the contact being handled is the primary contact.

To use this property, do the following:

```
Dim bIsPrimary as Boolean
bIsPrimary = oUserHandlingState.IsPrimary
```

Property:	IsPrimary (of UserHandlingState)
Description:	Returns true or false depending on whether the contact being handled is the primary contact.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

7.47.1.7 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oUserHandlingState.SequenceNumber
```

Property:	SequenceNumber (of UserHandlingState)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.47.2 Methods

This section contains methods exposed through the UserHandlingState interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

7.47.2.1 CancelMandatoryWrappingUp

The CancelMandatoryWrappingUp method can be called to reset the mandatory wrap-up flag of a contact.

To use this method, do the following:

```
call oUserHandlingState.CancelMandatoryWrappingUp
```

Method:	CancelMandatoryWrappingUp (of UserHandlingState)
Description:	This method resets the mandatory wrap-up flag of a contact.
Parameters	None
Returns	None
Potential Errors:	None
Notes:	Read-only

7.47.2.2 ExitPostProcessing

The ExitPostProcessing method can be called to exit from the post-processing handling state.

To use this method, do the following:

```
call oUserHandlingState.ExitPostProcessing
```

Method:	ExitPostProcessing (of UserHandlingState)
Description:	This method exits the post-processing handling state of a contact.
Parameters	None
Returns	None
Potential Errors:	None
Notes:	Read-only

7.47.2.3 RequestPostProcessing

The RequestPostProcessing method can be called to request the post-processing handling state.

To use this method, do the following:

```
call oUserHandlingState.RequestPostProcessing
(lPostProcessingReasonKey)
```

Property:	RequestPostProcessing (of UserHandlingState)
Description:	This method requests the post-processing handling state of a contact.
Parameters:	ReasonKey
Returns:	None
Potential Errors:	None
Notes:	Read-only

7.47.2.4 SetPrimary

The SetPrimary method can be called to set the contact as the primary contact.

To use this method, do the following:

```
call oUserHandlingState.SetPrimary
```

Property:	SetPrimary (of UserHandlingState)
Description:	This method sets the contact as the primary contact.
Parameters:	None
Returns:	None
Potential Errors:	Error_Media_PresenceUnableToSetPrimaryContact
Notes:	Read-only

7.48 UserHandlingStates

The UserHandlingStates object is a collection of the UserHandlingState objects.

Interfaces supported:

- IUserHandlingStates2 (default)
- IXMLAccess

7.48.1 Properties

This section contains properties exposed through the Agent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.48.1.1 _NewEnum

```
For Each oUserHandlingState in colUserHandlingStates
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oUserHandlingState as
HiPathProCenterLibrary.UserHandlingState
For Each oUserHandlingState in m_colUserHandlingStates
   txtTextBox.Text = txtTextBox.Text & _
      " CallID: " & _
      oUserHandlingState.CallID & _
      " MediaType: " & _
      oUserHandlingState.MediaType & _
      " HandlingState: " & _
      oUserHandlingState.HandlingState &
      vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain a collection of handling states for the user while they are handling a contact.

Property:	_NewEnum (of UserHandlingStates)
Description:	Allows for the 'For Each' operator on the UserHandlingStates collection.
Returns:	oUserHandlingState — For each item in the UserHandlingStates collection until all items have been returned.
Potential Errors:	None
Notes:	This method allows the UserHandlingStates collection to support the 'For Each' iterator.

7.48.1.2 Count

The Count property returns the number of UserHandlingState objects that exist in the UserHandlingStates collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colUserHandlingStates.Count
```

Property:	Count (of UserHandlingStates)
Description:	The number of UserHandlingState objects that are in the UserHandlingStates collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.48.1.3 Item

```
oUserHandlingState = Item (ContactID as String, SequenceNumber
as Long)
```

The Item method finds a UserHandlingState object in the UserHandlingStates collection that is indexed by the ContactID and SequenceNumber. If an item is found in the collection, the Item method returns the UserHandlingState object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameter:

ContactID as String	This is the ContactID for the period you are requesting.
SequenceNumber as Long	This is the SequenceNumber for the handling state you are requesting.

This method returns the following parameter:

oUserHandlingState as UserHandlingState	This is the UserHandlingState object that is indexed by the ContactID and SequenceNumber.

To use this method, do the following:

```
Dim oUserHandlingState as
HiPathProCenterLibary.UserHandlingState
Set oUserHandlingState = colUserHandlingStates.Item
(strContactID, lSequenceNumber)
```

This checks to see if the UserHandlingState exists. If it does, oUserHandlingState will contain the UserHandlingState object. If it does not exist in the UserHandlingStates collection, this will raise an error.

Method:	Item (of UserHandlingStates)
Description:	Finds an UserHandlingState object in the UserHandlingStates collection that is indexed by the ContactID and SequenceNumber.
Parameters:	ContactID — The ContactID value for the UserHandlingState that you are requesting. SequenceNumber — The SequenceNumber for the UserHandlingState that you are requesting.
Returns:	oUserHandlingState — The user status for the user specified.
Potential Errors:	None
Notes:	If the UserHandlingState object for the user does not exist in our collection, you raise an error.

7.49 UserMediaNotReadyEvent

The UserMediaNotReadyEvent is sent when the system is not ready to route contacts (voice, callback, e-mail, or Web collaboration).

Interfaces supported:

- IUserMediaNotReadyEvent (default)
- IMediaEvent
- IXMLAccess

7.49.1 Properties

This section contains properties exposed through the UserMediaNotReadyEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.49.1.1 UserMediaType

The UserMediaType property is the media type (voice, callback, e-mail, or Web collaboration) of the contact that the system is not ready to route.

To use this property, do the following:

```
Dim notReadyMediaType as HiPathProCenterLibrary.enMediaTypes
notReadyMediaType = oUserMediaNotReadyEvent.UserMediaType
```

Property:	UserMediaType (of UserMediaNotReadyEvent)
Description:	The media type (voice, callback, e-mail, or Web collaboration) of the contact that the system is not ready to route.
Type:	HiPathProCenterLibrary.enMediaTypes
Potential Errors:	None
Supported Media:	Voice, callback, e-mail, Web collaboration
Notes:	Read-only

7.50 UserMediaReadyEvent

The UserMediaReadyEvent is sent when the system is ready to route contacts (voice, callback, e-mail, or Web collaboration).

Interfaces supported:

- IUserMediaReadyEvent (default)
- IMediaEvent
- IXMLAccess

7.50.1 Properties

This section contains properties exposed through the UserMediaReadyEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.50.1.1 UserMediaType

The UserMediaType property is the media type (voice, callback, e-mail, or Web collaboration) of the contact that the system is ready to route.

To use this property, do the following:

```
Dim readyMediaType as HiPathProCenterLibrary.enMediaTypes
readyMediaType = oUserMediaReadyEvent.UserMediaType
```

Property:	UserMediaType (of UserMediaReadyEvent)
Description:	The media type (voice, callback, e-mail, or Web collaboration) of the contact that the system is ready to route.
Type:	HiPathProCenterLibrary.enMediaTypes
Potential Errors:	None
Supported Media:	Voice, callback, e-mail, Web collaboration
Notes:	Read-only

7.51 VoiceCall

The VoiceCall object represents a voice contact in the system. This performs the various actions that you would with a telephone call, such as Dial, Answer, and so on. You can also use this object to update a call after it has been assigned to a user, adding new Contact Data information, or updating the Description.

Interfaces supported:

- IVoiceCall6 (default)
- IMediaBase
- IXMLAccess

7.51.1 Properties

This section contains properties exposed through the VoiceCall interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.51.1.1 AgentKey

The AgentKey property is the key for the user in the database. This key represents the user who is associated with this call. This can be the user who is placing an outgoing call, or a user who the call has been assigned to. This is a system generated value.

To use this property, do the following:

```
Dim lKey as Long
lKey = oVoiceCall.AgentKey
```

Property:	AgentKey (of VoiceCall)
Description:	The database table unique key for the Agent object associated with this call.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.51.1.2 ANI

The ANI property is the telephone number of the original calling party. This is the source of the call. For example, this is the number that is displayed for call display.

To use this property, do the following:

```
Dim strTelephoneNumber as String
strTelephoneNumber = oVoiceCall.ANI
```

Property:	ANI (of VoiceCall)
Description:	The telephone number of the original calling party.
Type:	String
Potential Errors:	None
Notes:	None

7.51.1.3 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oVoiceCall.CallID
```

Property:	CallID (of VoiceCall)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Notes:	This property should be set before querying a VoiceCall from the MediaManager object.

7.51.1.4 ContactData

The ContactData property is the contact data that is associated with this VoiceCall object. The ContactData property can provide the most information about a call. The ContactData object is a collection of key/value pairs associated with the Call. This can be used to hold information entered in the IVR, or information from other agents that have talked to this customer. For more information, see Section 4.1, "ContactData", on page 117. This read/write property is also used so that the contact data for this contact can be updated.

To use this property, do the following:

```
Dim oContactData as HiPathProCenterLibrary.ContactData
Set oContactData = oVoiceCall.ContactData
```

Property:	ContactData (of VoiceCall)
Description:	The contact data that is associated with this contact in the system.
Type:	ContactData
Potential Errors:	None
Notes:	Read-only

7.51.1.5 ContactType

The ContactType property returns the type of the contact that a user is currently involved in. Contact Type This method does not accept any parameters. are Routed Voice, Direct Incoming Voice, Direct Outgoing Voice, Direct Internal Voice, Routed Callback, Routed E-mail, Direct Outgoing E-mail, Routed Web Collaboration, and Unknown Voice.

To use this property, do the following:

```
Dim enContactType as HiPathProCenterLibrary.enContactTypes
enContactType = oVoiceCall.ContactType
```

Property:	ContactType (of VoiceCall)
Description:	The type of contact that the user is handling.
Type:	enContactTypes
Potential Errors:	None
Notes:	Read-only

7.51.1.6 Description

The Description property is the description that is associated with a VoiceCall object. The description is a single line of text that is displayed in the Screen Pop portion of the Client Desktop application. This text provides some information about the queue that the queue itself may not be able to do. For example, if the queue is 'Sales', the description might be 'Sales Call for All Wood Furniture'. This read/write property is also used so that the description for this contact can be updated.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oVoiceCall.Description
```

Property:	Description (of VoiceCall)
Description:	The description that is associated with this contact in the system.
Type:	String
Potential Errors:	None
Notes:	None

7.51.1.7 DNIS

The DNIS property is the telephone number that was originally dialed by the caller. This is the destination of the call.

To use this property, do the following:

```
Dim strTelephoneNumber as String
strTelephoneNumber = oVoiceCall.DNIS
```

Property:	DNIS (of VoiceCall)
Description:	The telephone number that was originally dialed by the caller.
Type:	String
Potential Errors:	None
Notes:	None

7.51.1.8 Parties

The Parties property is the collection of parties that contains information about different parties involved in a call. These parties can represent agents, external customers, IVRs, and so on.

To use this property, do the following:

```
Dim colParties as HiPathProCenterLibrary.Parties
Set colParties = oVoiceCall.Parties
```

Property:	Parties (of VoiceCall)
Description:	A collection of parties that contains information about different parties involved in a call.
Type:	Parties
Potential Errors:	Error_Generic_CreatingObject
Notes:	Read-only

7.51.1.9 QueueKey

The QueueKey property is the key for the queue in the database. This is a system generated value.

To use this property, do the following:

```
Dim lKey as Long
lKey = oVoiceCall.QueueKey
```

Property:	QueueKey (of VoiceCall)
Description:	The database table unique key for the Queue object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.51.1.10 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oVoiceCall.SequenceNumber
```

Property:	SequenceNumber (of VoiceCall)
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.51.1.11 State

The State property is the state that the contact is in. This is one of the valid enumeration This method does not accept any parameters. from the enCallStates enumeration.

To use this property, do the following:

```
Dim enCallState as HiPathProCenterLibrary.enCallStates
enCallState = oVoiceCall.State
```

Property:	State (of VoiceCall)
Description:	The state of the call.
Type:	EnCallStates
Potential Errors:	None
Notes:	Read-only

7.51.1.12 WaitTime

The WaitTime property is the amount of time in seconds the contact was queued prior to being delivered to the agent.

To use this property, do the following:

```
Dim lWaitTime as Long
lWaitTime = oVoiceCall.WaitTime
```

Property:	WaitTime (of VoiceCall)
Description:	The amount of time in seconds the contact was queued prior to being delivered to the agent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.51.2 Methods

This section contains methods exposed through the VoiceCall interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.51.2.1 Answer

```
Answer (From as String)
```

The Answer method enables users to answer incoming telephone calls. To do this, they must specify the extension that they want to answer.

This method accepts the following parameter:

From as String	This is the telephone extension that the user wants to answer.

This method does not return anything.

To use this method, do the following after creating the VoiceCall:

```
Call oVoiceCall.Answer (txtExtension.Text)
```

This answers the incoming telephone call on the extension specified.

Method:	Answer (of VoiceCall)
Description:	Answers an incoming telephone call.
Parameters:	From — the telephone extension that user wants to answer.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

7.51.2.2 Conference

```
Conference (From as String)
```

The Conference method is used after a user has performed a consultation and wants to conference all three parties together. To do this, they must pass their extension and call Conference.

This method accepts the following parameters:

From as String	This is the telephone extension that the user is calling from.

This method does not return anything.

To use this method, do the following after creating the VoiceCall:

```
Call oVoiceCall.Conference (txtExtension.Text)
```

This essentially connects the original caller (on hold from the consultation), the user, and the party the user is consulting with.

Method:	Conference (of VoiceCall)
Description:	Conferences the user with the party they are consulting with, as well as the caller placed on hold when the consultation was performed.
Parameters:	From — the telephone extension of the user that wants to perform the conference from.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

7.51.2.3 Consult

```
Consult (From as String, To as String)
```

The Consult method enables users to place the call on hold and consult with another user. To do this, they must pass their extension, as well as the number that they want to consult with.

This method accepts the following parameters:

From as String	This is the telephone extension that the user is calling from.
To as String	This is the telephone number that the user wants to consult with.

This method does not return anything.

To use this method, do the following after creating the VoiceCall:

```
Call oVoiceCall.Consult (txtExtension.Text, txtNumber.Text)
```

This essentially places the party that the current user is talking to on hold, and will dial the txtNumber.Text number and connect the user to that party.

Method:	Consult (of VoiceCall)
Description:	Consults with another party (usually another user in the contact center).
Parameters:	From — the telephone extension of the user that wants to perform the consultation. To — the telephone number that the user wants to consult with.
Returns:	None
Potential Errors:	• Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultTelephonyServerNotReady • Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultOutOfService • Error_Media_ActionResultLineBusy • Error_Generic_ValidationFailed • Error_Generic_RequestTimedOut
Notes:	None

7.51.2.4 Dial

```
Dial (From as String, To as String)
```

The Dial method enables users to place outgoing telephone calls. To do this, they must specify the extension that they are dialing from, as well as the telephone number that they want to dial (this can be another extension, or a complete telephone number).

This method accepts the following parameters:

From as String	This is the telephone extension that the call is originating from (the outgoing caller).
To as String	This is the telephone number that the call should be going to (the destination).

To use this method, do the following after creating the VoiceCall:

```
Call oVoiceCall.Dial (txtExtension.Text, txtNumber.Text)
```

This calls the telephone number you have specified in the To field, from the extension that you have specified in the From field.

Method:	Dial (of VoiceCall)
Description:	Makes a telephone call.
Parameters:	From — the extension that the call is originating from (the caller). To — the telephone number that you want to dial (the destination).
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

7.51.2.5 Disconnect

```
Disconnect (From as String)
```

The Disconnect method disconnects a telephone call. To do this, they have to pass their extension and call Disconnect.

This method accepts the following parameters:

From as String	This is the telephone extension that the user wants to disconnect.

This method does not return anything.

To use this method, do the following after creating the VoiceCall:

```
Call oVoiceCall.Disconnect (txtExtension.Text)
```

This essentially disconnects the user with the caller they are connected with (or will disconnect on an outgoing Dial if the user is on an outgoing call).

Method:	Disconnect (of VoiceCall)
Description:	Disconnects the call that the user is on.
Parameters:	From — the telephone extension of the user that wants to perform the disconnect.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

7.51.2.6 Divert

```
Divert (To as String)
```

The Divert method transfers a call from an ACD (usually a Flex Routing ACD, in a system when the system is performing the routing) to a user's extension. To do this, they must pass the extension to which they want to divert the call.

This method accepts the following parameters:

To as String	This is the user extension to which the call is to be diverted.

This method does not return anything.

To use this method, do the following after creating the VoiceCall:

```
Call oVoiceCall.Divert (txtNumber.Text)
```

This method is quite different from the other methods as it is performed on a call that residing in an ACD queue rather than a call that is not assigned to a user.

Method:	Divert (of VoiceCall)
Description:	Diverts a call from an ACD (Flex Routing usually) queue to an agents extension.
Parameters:	To — the user's extension to which the user wants to divert the call.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

7.51.2.7 Hold

```
Hold (From as String)
```

The Hold method enables users to place telephone calls on hold. To do this, they must specify the extension that they want to place on hold.

This method accepts the following parameters:

From as String	This is the telephone extension that the user wants to place on hold.

This method does not return anything.

To use this method, do the following after creating the VoiceCall:

```
Call oVoiceCall.Hold (txtExtension.Text)
```

This is similar to pressing the hold button on the extension specified by the txtExtension.Text parameter.

Method:	Hold (of VoiceCall)
Description:	Places a telephone call on hold.
Parameters:	From — the telephone extension that user wants to place on hold.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

7.51.2.8 Reconnect

```
Reconnect (From as String)
```

The Reconnect method is used after a user has performed a consultation and is talking to another party. After they have determined that they want to return to the caller that is on hold, they need to call Reconnect. To do this, they must pass their extension and call Reconnect.

This method accepts the following parameter:

From as String	This is the telephone extension that the user is calling from.

This method does not return anything.

To use this method, do the following after creating the VoiceCall:

```
Call oVoiceCall.Reconnect (txtExtension.Text)
```

This essentially reconnects the user to the caller that was placed on hold when the Consultation was performed.

Method:	Reconnect (of VoiceCall)
Description:	Reconnects the user with a caller placed on hold during a consultation.
Parameters:	From — the telephone extension of the user that wants to perform the reconnect.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

7.51.2.9 Retrieve

```
Retrieve (From as String)
```

The Retrieve method enables users to retrieve telephone calls that are on hold. To do this, they must specify the extension that they want to retrieve the call from.

This method accepts the following parameter:

From as String	This is the telephone extension that the user wants to retrieve the call from.

This method does not return anything.

To use this method, do the following after creating the VoiceCall:

```
Call oVoiceCall.Retrieve (txtExtension.Text)
```

This is similar to pressing the hold button on the extension specified by the txtExtension.Text parameter after a call has been placed on hold.

Method:	Retrieve (of VoiceCall)
Description:	Retrieves a telephone call that is on hold.
Parameters:	From — the telephone extension that user wants to retrieve the call that is on hold from.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

7.51.2.10 Transfer

```
Transfer (From as String)
```

The Transfer method is used after a user has performed a consultation and wants to transfer the caller on hold to the party the user has consulted with. To do this, they must pass their extension and call Transfer.

This method accepts the following parameters:

From as String	This is the telephone extension that the user is calling from.

This method does not return anything.

To use this method, do the following after creating the VoiceCall:

```
Call oVoiceCall.Transfer (txtExtension.Text)
```

This essentially connects the caller with the party the user consulted with, in effect, performing a warm transfer.

Method:	Transfer (of VoiceCall)
Description:	Transfers the caller placed on hold when the consultation was performed, with the party the user consulted with.
Parameters:	From — the telephone extension of the user that wants to perform the transfer from.
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

7.52 WebCollaborationCall

The WebCollaborationCall object represents a WebCollaboration contact in the system. It is used to send WebCollaboration messages to callers or to invite people to join Web collaboration sessions.

Interfaces supported:

- IWebCollaborationCall2 (default)
- IBinaryAccess
- IMediaBase
- IXMLAccess

7.52.1 Properties

This section contains properties exposed through the WebCollaborationCall interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.52.1.1 AgentKey

The AgentKey property is the key for the user associated with the call in the database, either placing an outgoing call, or a user who the call has been assigned to. This is a system-generated value.

You must set the AgentKey property before calling any methods on the WebCollaborationCall object because you need to ensure that you associate the user with the Web collaboration session that the WebCollaborationCall corresponds to. When messages are sent, it is known who is saying what. If this property is not set, an error is raised when a method is attempted.

To use this property, do the following:

```
Dim lKey as Long
lKey = oWebCollaborationCall.AgentKey
```

Property:	AgentKey (of WebCollaborationCall)
Description:	The database table unique key for the Agent object associated with this call.
Type:	Long
Potential Errors:	None
Notes:	None

7.52.1.2 CallID

The CallID property is the unique key for the contact in the system.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oWebCollaborationCall.CallID
```

Property:	CallID (of WebCollaborationCall)
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Notes:	This property should be set before querying a WebCollaborationCall from the MediaManager object.

7.52.1.3 ContactData

The ContactData property is the contact data that is associated with this WebCollaborationCall object. The ContactData property can provide the most information about a call. The ContactData object is a collection of key/value pairs associated with the Call. This can be used to hold information entered in the IVR, or information from other agents that have talked to this customer. This read/write property is also used so that the contact data for this contact can be updated.

To use this property, do the following:

```
Dim oContactData as HiPathProCenterLibrary.ContactData
Set oContactData = oWebCollaborationCall.ContactData
```

Property:	ContactData (of WebCollaborationCall)
Description:	The contact data that is associated with the contact in the system.
Type:	ContactData
Potential Errors:	None
Notes:	None

7.52.1.4 CustomerName

The CustomerName property is the surfer's name that is currently associated with this WebCollaboration object.

To use this property, do the following:

```
Dim strCustomerName as String
strCustomerName = oWebCollaborationCall.CustomerName
```

Property:	CustomerName (of WebCollaborationCall)
Description:	The surfer's name that is currently associated with this WebCollaboration object.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.52.1.5 Description

The Description property is the Description that is currently associated with this WebCollaborationCall object in the system.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oWebCollaborationCall.Description
```

Property:	Description (of WebCollaborationCall)
Description:	The Description that is currently associated with this WebCollaborationCall object.
Type:	String
Potential Errors:	None
Notes:	None

7.52.1.6 Destination

The Destination property is the URL from which the customer requested the Web collaboration session. For our purposes in the SDK, this is similar to the dialed number, in that it was the 'door' the customer used to get into the system.

To use this property, do the following:

```
Dim strDestination as String
strDestination = oWebCollaborationCall. Destination
```

Property:	Destination (of WebCollaborationCall)
Description:	The URL from which the customer requested the Web collaboration session.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.52.1.7 QueueKey

The QueueKey property is the key of the queue entity associated with this WebCollaborationCall object in the database.

To use this property, do the following:

```
Dim lQueueKey as Long
lQueueKey = oWebCollaborationCall.QueueKey
```

Property:	QueueKey (of oWebCollaborationCall)
Description:	The database table unique key for the WebCollaborationCall object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.52.1.8 SequenceNumber

The SequenceNumber property is an internal number used to uniquely identify contacts.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oWebCollaborationCall.SequenceNumber
```

Property:	SequenceNumber (of WebCollaborationCall))
Description:	An internal number used to uniquely identify contacts.
Type:	Long
Potential Errors:	None
Notes:	Read/Write

7.52.1.9 Source

The Source property is the Contact source such as a customer IP or the account number of the customer that is requesting a Web collaboration session. This is similar to the FromAddress or ANI of an e-mail message or voice call.

To use this property, do the following:

```
Dim strSource as String
strSource = oWebCollaborationCall.Source
```

Property:	Source (of WebCollaborationCall)
Description:	The Contact source such as a customer IP or the account number of the customer that is requesting a Web collaboration session.
Type:	String
Potential Errors:	None
Notes:	Read-only

7.52.1.10 State

The State property is the state that the contact is currently in. This is one of the valid enumeration This method does not accept any parameters. from the enCallStates enumeration.

To use this property, do the following:

```
Dim enCallState as HiPathProCenterLibrary.enCallStates
enCallState = oWebCollaborationCall.State
```

Property:	State (of WebCollaborationCall)
Description:	The state that the contact is currently in.
Type:	enCallStates
Potential Errors:	Error_Generic_UnknownState
Notes:	Read-only

7.52.1.11 Transcript

The Transcript property contains the transcript of the Web collaboration session so far. This is designed to be used for such scenarios as when a new party joins the Web collaboration session. They can get the transcript of the Web collaboration session so far, and then handle the ContentAdded events when WebCollaboration members send new text.

To use this property, do the following:

```
Dim strTranscript as String
strTranscript = oWebCollaborationCall.Transcript
```

Property:	Transcript (of WebCollaborationCall)
Description:	The transcript of the Web collaboration session
Type:	String
Potential Errors:	None
Notes:	Read-only

The transcript is current only at the time the Query is performed. The transcript property does not contain new text that is added after the Query.

7.52.1.12 WaitTime

The WaitTime property is the amount of time in seconds the contact was queued prior to being delivered to the agent.

To use this property, do the following:

```
Dim lWaitTime as Long
lWaitTime = oWebCollaborationCall.WaitTime
```

Property:	WaitTime (of oWebCollaborationCall)
Description:	The amount of time in seconds the contact was queued prior to being delivered to the agent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

7.52.2 Methods

This section contains methods exposed through the WebCollaborationCall interface. For more information on the Potential Error s listed in the following sections, see Section 3.18, “enErrors”, on page 59.

7.52.2.1 Accept

The Accept method accepts a WebCollaboration call that a user has been offered. This is required when a WebCollaboration call is offered to a user so the Web Interaction Server knows that the agent will handle the WebCollaboration call.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following after creating the WebCollaborationCall:

```
Call oWebCollaborationCall.Accept
```

This informs the WebCollaboration server that the user has accepted the Web collaboration session and will handle the WebCollaboration call.

Method:	Accept (of WebCollaborationCall)
Description:	Accepts a WebCollaboration call that is offered to a user.
Parameters:	None

Returns:	None
Potential Errors:	• Error_Media_WebCollaborationDataCorrupted • Error_Media_WebCollaborationRPCTimeout • Error_Media_WebCollaborationRPCFailed • Error_Media_WebCollaborationWrongVersion • Error_Media_WebCollaborationUnableToAllocateMemory • Error_Media_WebCollaborationServerTerminating
Notes:	None

7.52.2.2 ConferenceRequest

The ConferenceRequest method sends a request to another agent to see if they want to join the conference.

A DeliveredEvent will be sent to the agent being requested. The value of Reason property will be DeliveredReason_WebCollaboration_Conferenced. If the requested agent calls Reject () a DivertedEvent will be sent to the agent who initiated the request. The DivertReason will be: DivertReason_InvitationRejected. If the agent did not answer the request in time, the DivertReason will be DivertReason_InvitationTimeOut.

This method accepts the following parameter:

AgentKeyToAdd as Long	This is the agent key for the user you are requesting to join the conference.

This method does not return anything.

To use this method, do the following after creating the WebCollaborationCall:

```
Call oWebCollaborationCall.ConferenceRequest (txtAgents.Value)
```

This sends a message to the user who is specified by the agent key to see if they would like to join the Web collaboration session. They can then either Accept or Reject.

Method:	ConferenceRequest (of WebCollaborationCall)
Description:	Sends a request to another user to see if they want to join the conference.
Parameters:	AgentKeyToAdd — the agent key for the user you are requesting to join your conference.

Returns:	None
Potential Errors:	• Error_Media_WebCollaborationDataCorrupted • Error_Media_WebCollaborationRPCTimeout • Error_Media_WebCollaborationRPCFailed • Error_Media_WebCollaborationWrongVersion • Error_Media_WebCollaborationUnableToAllocateMemory • Error_Media_WebCollaborationServerTerminating • Error_Media_WebCollaborationAgentAlreadyInSession • Error_Media_WebCollaborationInvalidAgentKey
Notes:	None

7.52.2.3 Disconnect

The Disconnect method is used to disconnect a user from a Web collaboration session. This is similar to hanging up a telephone call.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following after creating the WebCollaborationCall:

```
Call oWebCollaborationCall.Disconnect
```

This disconnects the user who is associated with this WebCollaborationCall object from the Web collaboration session.

Method:	Disconnect (of WebCollaborationCall)
Description:	Disconnects from a Web collaboration session.
Parameters:	None
Returns:	None
Potential Errors:	• Error_Media_WebCollaborationDataCorrupted • Error_Media_WebCollaborationRPCTimeout • Error_Media_WebCollaborationRPCFailed • Error_Media_WebCollaborationWrongVersion • Error_Media_WebCollaborationUnableToAllocateMemory • Error_Media_WebCollaborationServerTerminating
Notes:	None

7.52.2.4 PushCallMeForm

The PushCallMeForm method is used to send a Web page to a customer to request their telephone number. This would be used if perhaps the customer did not feel comfortable with the Web collaboration session and would prefer to have a live agent call them. The agent chatting with the customer can call this method which will ask the customer for their telephone number.

This method does not accept any parameters.

This method does not return anything.

To use this method, do the following after creating the WebCollaborationCall:

```
Call oWebCollaborationCall.PushCallMeForm
```

Method:	PushCallMeForm (of WebCollaborationCall)
Description:	Requests a telephone number from a customer.
Parameters:	None
Returns:	None
Potential Errors:	• Error_Media_WebCollaborationDataCorrupted • Error_Media_WebCollaborationRPCTimeout • Error_Media_WebCollaborationRPCFailed • Error_Media_WebCollaborationWrongVersion • Error_Media_WebCollaborationUnableToAllocateMemory • Error_Media_WebCollaborationServerTerminating • Error_Media_WebCollaborationInvalidAgentKey
Notes:	None

7.52.2.5 Reject

The Reject method is used by a user who has been requested to join a Web collaboration session (through the ConferenceRequest method) to reject the conference request.

This method accepts the following parameter:

RefusingAgent Key as Long	This is the agent key that is refusing the Web collaboration session conference request.

This method does not return anything.

To use this method, do the following after creating the WebCollaborationCall:

```
Call oWebCollaborationCall.Reject (txtAgents.Value)
```

This results in a diverted event.

Method:	Reject (of WebCollaborationCall)
Description:	Refuses a Web collaboration session conference request.
Parameters:	RefusingAgentKey — the agent key that is refusing the Web collaboration session conference request.
Returns:	None
Potential Errors:	• Error_Media_WebCollaborationDataCorrupted • Error_Media_WebCollaborationRPCTimeout • Error_Media_WebCollaborationRPCFailed • Error_Media_WebCollaborationWrongVersion • Error_Media_WebCollaborationUnableToAllocateMemory • Error_Media_WebCollaborationServerTerminating • Error_Media_WebCollaborationInvalidAgentKey
Notes:	None

7.52.2.6 Requeue

The Requeue method is used by a user to requeue a Web collaboration session to a different queue. To do this, the user must specify the queue that the Web collaboration session should be requeued with.

This method accepts the following parameters:

QueueKey as Long	This is the key for the Queue that you want to requeue the Web collaboration session with.
ReservedAgentKey as Long	This is the agent to warm requeue the call to.
Priority as Long	This is the new priority.

This method does not return anything.

To use this method, do the following after creating the WebCollaborationCall:

```
Call oWebCollaborationCall.Requeue (lQueueKey,
lReservedAgentKey, lPriority)
```

This requeues the Web collaboration session for the QueueKey specified by lQueueKey. The following table shows how the session is requeued:

Interface	Value	Result
IQueueKey	zero	The existing queue will be used for the requeue.
lReservedAgentKey value	zero	The first available agent in the queue that is eligible to take the contact is assigned the contact.
	non-zero	The contact will be warm-requeued to this agent key.
IPriority	zero	The existing priority is used.

Table 52 Session requeueing results

Method:	Requeue (of WebCollaborationCall)
Description:	Used by a user to requeue a Web collaboration session to a different queue.
Parameters:	• QueueKey — the key for the Queue that you want to requeue the Web collaboration session with. • ReservedAgentKey — the agent to warm requeue the call to. • Priority — the new priority.

Returns:	None
Potential Errors:	• Error_Media_WebCollaborationDataCorrupted • Error_Media_WebCollaborationRPCTimeout • Error_Media_WebCollaborationRPCFailed • Error_Media_WebCollaborationWrongVersion • Error_Media_WebCollaborationUnableToAllocateMemory • Error_Media_WebCollaborationServerTerminating • Error_Media_WebCollaborationInvalidAgentKey • Error_Media_WebCollaborationInvalidCallID
Notes:	None

7.52.2.7 SendContent

The SendContent method sends a Web collaboration message member of the Web collaboration session.

This method accepts the following parameter:

Message as Long	This is the message that the user of the WebCollaborationCall object wants to send.

This method does not return anything.

To use this method, do the following after creating the WebCollaborationCall:

```
Call oWebCollaborationCall.SendContent (txtMessageToSend.Text)
```

This sends the Web collaboration message specified in the text field of txtMessageToSend.Text to members of the Web collaboration session.

Method:	SendContent (of WebCollaborationCall)
Description:	Sends message to members of the Web collaboration session.
Parameters:	Message — for members of the Web collaboration session.
Returns:	None
Potential Errors:	• Error_Media_WebCollaborationDataCorrupted • Error_Media_WebCollaborationRPCTimeout • Error_Media_WebCollaborationRPCFailed • Error_Media_WebCollaborationWrongVersion • Error_Media_WebCollaborationUnableToAllocateMemory • Error_Media_WebCollaborationServerTerminating
Notes:	None

8 Routing Manager

The Routing Manager is responsible for handling the features of the system related to routing. This includes updating queue information for calls that are enqueued.

The Routing Manager is also responsible for networking functionality.

8.1 RoutingManager object

The RoutingManager is a non-creatable object that provides access to the routing functionalities in the system. This includes querying and updating contacts from the Routing Server.

The RoutingManager is the access point to any routing functionality. This includes local routing for the current site, as well as inter-site routing.

Interfaces supported:

- IRoutingManager

8.1.1 Properties

This section contains properties exposed through the RoutingManager interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

8.1.1.1 State

The State property returns the current state of the RoutingManager. This refers to the state that the RoutingManager is in (Available or Unavailable).

This property returns a value from the enManagerStates enumerations described in Section 3.28, “enManagerStates”, on page 87. This value is read-only.

To use this property, do the following:

```
Dim enState as HiPathProCenterLibrary.enStates
enState = g_oRoutingManager.State
```

Property:	State (of RoutingManager)
Description:	The state of the Routing Manager (Available/Unavailable).
Type:	enManagerStates
Potential Errors:	Error_Generic_UnableToDetermineState
Notes:	Read-only

8.1.2 Methods

This section contains methods exposed through the RoutingManager interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

8.1.2.1 GetFunctionalityState

```
enManagerStates = GetFunctionalityState (Routing Functionality
as enRoutingFunctionalities)
```

The RoutingManager may occasionally lose its connection to one or more servers in the system. When this occurs, you can check to see what functionality is supported. To do this, you can use the GetFunctionalityState method to pass in the functionality that you are interested in to return the state of that functionality.

This method accepts the following parameter:

Routing Functionality as enRouting Functionalities	This is the functionality group that you want to enquire about its. See Section 3.42, "enRoutingFunctionalities", on page 107.

This method returns the ManagerState of the functionality group you have requested. For more information, see Section 3.28, "enManagerStates", on page 87.

To use this method, do the following after creating the RoutingManager:

```
Dim enManagerState as HiPathProCenterLibrary.enManagerStates

enManagerState =

   g_oRoutingManager.GetFunctionalityState
   RoutingFunctionality_LocalRouting)
```

This checks with the RoutingManager to see if you have the ability to receive local site routing information for this site.

Method:	GetFunctionalityState (of RoutingManager)
Description:	Queries the state of various functionality groups in the RoutingManager.
Parameters:	RoutingFunctionality — The functionality group you want to check the state of.
Returns:	enManagerStates — The state of the functionality group you requested.
Potential Errors:	Error_Generic_UnableToDetermineState
Notes:	None

8.1.2.2 ListenForEvents

```
ListenForEvents (RoutingEventType as enRoutingEventTypes,
(Optional) Resource as String = "*")
```

The RoutingManager has a variety of events that you can request to listen for. To do this, you must make a call to ListenForEvents with the event type and the resource they want to listen for the event on. The RoutingManager will then tell the servers that it wants to know when these events occur, and will wait for them. When they occur, the RoutingManager will receive the events and pass them back in the EventOccurred event handler.

This function can only be called if the RoutingManager was created with events.

This method accepts the following parameters:

RoutingEventType as enRoutingEventTypes	This is the event type that you want to listen for events on. It's important to note that event types usually refer to more than one specific event in the RoutingManager. See Section 3.41, "enRoutingEventTypes", on page 106.
Resource as String	This is the resource that you want to listen for the event on. The resource is specific for the event type being listened for. For a list of what the resource represents for the various RoutingEventTypes, see Section 3.41, "enRoutingEventTypes", on page 106.

This method does not return anything.

To use this method, do the following after creating the RoutingManager:

```
Call g_oRoutingManager.ListenForEvents
(RoutingEventType_VoiceEvents)
```

or

```
Call g_oRoutingManager.ListenForEvents
(RoutingEventType_VoiceEvents, "*")
```

Although both of the preceding examples will do the same thing, the difference between the two is that in one you explicitly state the resource you want to listen for the event on. The resource parameter for this is optional, and if you do not pass it in, it will default to '*' (or all events). For more information, see Section 2.8, "About events", on page 37.

Method:	ListenForEvents (of RoutingManager)
Description:	Listens for events from the RoutingManager.
Parameters:	• Resource("*") — The resource that you want to listen for the events on. • RoutingEventType — The event group you want to listen for.
Returns:	None
Potential Errors:	Error_Generic_ListeningForEventsWhenIgnoringEvents
Notes:	This function can only be called if the RoutingManager was created with events.

8.1.2.3 NewRoutingCall

```
oRoutingCall = NewRoutingCall()
```

To create a RoutingCall object, you must have the RoutingManager create one on your behalf. To do this, call NewRoutingCall.

This method does not accept any parameters.

This method returns a newly created RoutingCall object. For more information, see Section 8.6, "RoutingCall", on page 551.

To use this method, do the following after creating the RoutingManager:

```
Dim oRoutingCall as HiPathProCenterLibrary.RoutingCall
Set oRoutingCall = g_oRoutingManager.NewRoutingCall
```

Method:	NewRoutingCall (of RoutingManager)
Description:	Creates a new RoutingCall object.
Parameters:	None
Returns:	RoutingCall — A newly created RoutingCall object.
Potential Errors:	Error_Generic_CreatingObject
Notes:	None

8.1.2.4 Query

```
Query (RoutingBase as RoutingBase)
```

To Query any appropriate object through the RoutingManager you must pass the object into the Query method. The object being passed in must support the RoutingBase interface.

, the only object that can be queried is the RoutingCall object. To Query the RoutingCall object, you need to ensure that the CallID is set before calling Query.

This method accepts the following parameters:

RoutingBase as RoutingBase	This is the RoutingBase object that you want to query.

This method does not return any new value, but does update the object passed into the method with the updated object data. For example, if a RoutingCall object was passed in to be queried, the updated RoutingCall object would be returned.

To use this method, do the following after creating the RoutingManager:

```
oRoutingCall.CallID = "1234"
Call g_oRoutingManager.Query (oRoutingCall)
MsgBox "Call Description: " & oRoutingCall.Description
```

Method:	Query (of RoutingManager)
Description:	Queries an object through the RoutingManager.
Parameters:	RoutingBase — An object that supports the event base interface that you want to query.
Returns:	None
Potential Errors:	• Error_Generic_QueryFailed • Error_Generic_QueryNotSupported • Error_Generic_UnableToDetermineState • Error_Generic_UnableToReadObjectInformation • Error_Generic_UnableToWriteObjectInformation
Notes:	The object being passed in must support the RoutingBase interface. The object being passed in will be updated with the updated object data.

8.1.2.5 StopListeningForEvents

```
StopListeningForEvents (RoutingEventType as enRoutingEventTypes,
(Optional) Resource as String = "*")
```

The RoutingManager has a variety of events that you can request to listen for. There may come a time though you no longer want to listen for these events. To stop listening for these events, you must call the StopListeningForEvents method. To do this, you must make a call to StopListeningForEvents with the event type and the resource you previously listened for the event on. You must use the exact same resource that you used when calling ListenForEvents. The RoutingManager will then tell the servers that it no longer wants to know when these events occur, and is no longer listening for them.

This function can only be called if the RoutingManager was created with events.

This method accepts the following parameters:

RoutingEvent Type as enRoutingEvent Types	This is the event type that you want to stop listening for events on. It's important to note that event types usually refer to more than one specific event in the RoutingManager. See Section 3.41, “enRoutingEventTypes”, on page 106.
Resource as String	This is the resource that you want to stop listening for the event on. The resource is specific for the event type you are stopping listening for. This must match the exact resource that you used when you issued the ListenForEvent request. For a list of what the resource represents for the various RoutingEventTypes, see Section 3.41, “enRoutingEventTypes”, on page 106.

This method does not return anything.

To use this method, do the following after creating the RoutingManager:

```
Call g_oRoutingManager.StopListeningForEvents
(RoutingEventType_VoiceEvents)
```

or

```
Call g_oRoutingManager.StopListeningForEvents
(RoutingEventType_VoiceEvents, "*")
```

Although both of the preceding examples will do the same thing, the difference between the two is that in one you explicitly state the resource you want to listen for the event on. The resource parameter for this is optional, and if you do not pass it in, it will default to '*' (or all events). It is important that you use the same resource that you used when you call ListenForEvents. For example, if you registered for the event with '1234' you must use that to deregister for the event. Although '*' can be used for all events when you listen and stop

listening for events, when you stop listening for events, '*' only refers to stop listening for events that you listened for with '*'. For more information, see Section 2.8, “About events”, on page 37.

Method:	StopListeningForEvents (of RoutingManager)
Description:	Stops listening for events from the RoutingManager.
Parameters:	• Resource("*") — The resource that you want to listen for the events on. • RoutingEventType — The event group you want to stop listening for events for.
Returns:	None
Potential Errors:	Error_Generic_StopListeningForEventsWhenIgnoringEvents
Notes:	• This function can only be called if the RoutingManager was created with events. • The resource must be the exact same as the resource you used when you called ListenForEvents.

8.1.2.6 Update

```
Update (RoutingBase as RoutingBase)
```

To Update any appropriate object through the RoutingManager you must pass the object into the Update method. The object being passed in must support the RoutingBase interface.

, the only object that can be updated is the RoutingCall object. To Update the RoutingCall object, you must take a RoutingCall object (ideally after querying it), and update the fields that you want to update. After you have made all the changes to the object that you want, you must call Update on the RoutingManager and pass along the RoutingCall object that you want to update.

This method accepts the following parameters:

RoutingBase as RoutingBase	This is the RoutingBase object that you want to update.

This method does not return any new value, but does update the object passed into the method with the updated object data. In most instances, this should be the exact same properties that you have passed in, but it is possible that some properties may have changed (for example, State). Update returns an updated object.

To use this method, do the following after creating the RoutingManager:

```
oRoutingCall.CallID = "1234"
Call g_oRoutingManager.Query (oRoutingCall)
oRoutingCall.Description = "New Description"
Call g_oRoutingManager.Update (oRoutingCall)
```

Method:	Update (of RoutingManager)
Description:	Updates an object through the RoutingManager.
Parameters:	RoutingBase — An object that supports the event base interface that you want to update.
Returns:	None
Potential Errors:	• Error_Generic_UpdateFailed • Error_Generic_UpdateNotSupported • Error_Generic_UnknownObjectType • Error_Routing_InvalidQueue
Notes:	The object being passed in must support the RoutingBase interface. The object being passed will update the server and return fully updated if there have been other changes in the object.

8.1.3 Events

The following events are exposed through the RoutingManager interface.

For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

8.1.3.1 EventOccurred

```
g_oRoutingManager_EventOccurred (RoutingEvent as RoutingEvent)
```

The EventOccurred event handler returns all events that occur in the RoutingManager object. All events returned through this event interface support the RoutingEvent interface. For more information, see Section 8.7, "RoutingEvent", on page 560.

This method returns the following parameters:

RoutingEvent as HiPathProCenterLibrary.RoutingEvent	This is the event that has occurred in the RoutingManager. This returns the interface pointer to the base RoutingEvent interface. You can take this object and get the full object interface pointer from it as follows.

To use this method, do the following after creating the RoutingManager:

```
Private Sub g_oRoutingManager_EventOccurred
   (RoutingEvent as RoutingEvent)
   dim oEnqueuedEvent as HiPathProCenterLibrary.EnqueuedEvent
   if RoutingEvent.ObjectType =
      RoutingEventObjectType_Enqueued Then
      ' Here you get the EnqueuedEvent interface from
      ' the RoutingEvent interface you received in the
      ' Event, and then check the Call ID property.
      Set oEnqueuedEvent = RoutingEvent
      MsgBox "Enqueued Call—CallID: " &
      oEnqueuedEvent.CallID
   End If
End Sub
```

Event Handler:	EventOccurred (of RoutingManager)
Description:	Returns all events that occur in the RoutingManager.
Parameters:	RoutingEvent — The RoutingEvent interface for the event that has occurred.

Returns:	None
Potential Errors:	None
Notes:	The event that is returned in this method can be 'recast' to other interfaces that are supported by the object. See Section 8.7, "RoutingEvent", on page 560.

8.2 AssignedEvent

The AssignedEvent object is an event object that is sent whenever a call is assigned to a user in the system. These can be e-mail, voice, Web collaboration or callback calls.

Interfaces supported:

- IAssignedEvent
- IRoutingEvent
- IXMLAccess

8.2.1 Properties

This section contains properties exposed through the AssignedEvent interface. All properties from event objects are Read-only. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

8.2.1.1 AgentExtension

The AgentExtension property is the extension of the user who the call has been assigned to.

To use this property, do the following:

```
Dim strAgentExtension as String
strAgentExtension = oAssignedEvent.AgentExtension
```

Property:	AgentExtension (of AssignedEvent)
Description:	The user extension of the user who this call has been assigned to.
Type:	String
Potential Errors:	None
Notes:	Read-only

8.2.1.2 AgentKey

The AgentKey property is the key for the user who the call has been assigned to.

To use this property, do the following:

```
Dim lAgentKey as Long
lAgentKey = oAssignedEvent.AgentKey
```

Property:	AgentKey (of AssignedEvent)
Description:	The agent key for the user who this call has been assigned to.
Type:	Long
Potential Errors:	None
Notes:	Read-only

8.2.1.3 CallID

The CallID property is the unique key for the call object in the system. This value is unique whether the call is a voice, e-mail, Web collaboration or callback call. This is the same CallID that would match the VoiceCall, EmailCall, WebCollaborationCall, or CallbackCall CallID property.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oAssignedEvent.CallID
```

Property:	CallID (of AssignedEvent)
Description:	The unique key for a call in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

8.3 DequeuedEvent

The DequeuedEvent object is an event object that is sent whenever a call is dequeued from the system. These can be e-mail, voice, Web collaboration or callback calls.

Interfaces supported:

- IDequeuedEvent2 (default)

- IRoutingEvent
- IXMLAccess

8.3.1 Properties

This section contains properties exposed through the DequeuedEvent interface. All properties from event objects are Read-only. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

8.3.1.1 CallID

The CallID property is the unique key for the call object in the system. This value is unique whether the call is a voice, e-mail, Web collaboration, or callback call. This is the same CallID that would match the VoiceCall, EmailCall, WebCollaborationCall, or CallbackCall CallID property.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oDequeuedEvent.CallID
```

Property:	CallID (of DequeuedEvent)
Description:	The unique key for a call in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

8.3.1.2 DequeueReason

The DequeueReason property is the reason for why the call was dequeued.

To use this property, do the following:

```
Dim enDequeueReason as HiPathProCenterLibrary.DequeueReasons
enDequeueReason = oEnqueuedEvent.DequeueReason
```

Property:	DequeueReason (of DequeuedEvent)
Description:	The reason why the call was dequeued from the Routing Server.

Type:	enDequeueReasons
Potential Errors:	None
Notes:	Read-only

8.3.1.3 Description

The Description property is the description that is associated with call that is being dequeued. The description is a single line of text that can be displayed in the Screen Pop portion of the Client Desktop application. This can be used to provide some information about the queue that the queue itself may not be able to provide. For example, if the queue is 'Sales' the description might be 'Sales Call for All Wood Furniture'.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oDequeuedEvent.Description
```

Property:	Description (of DequeuedEvent)
Description:	The description that is associated with the dequeued call in the system.
Type:	String
Potential Errors:	None
Notes:	None

8.3.1.4 Priority

The Priority property is the priority that is associated with the call that is being dequeued. The priority determines how this call is scored compared to other calls. The higher the priority of the call relative to other calls, the higher the importance of the call. This read/write property is also used so that the priority for this call object can be updated.

To use this property, do the following:

```
Dim lPriority as Long
lPriority = oDequeuedEvent.Priority
```

Property:	Priority (of DequeuedEvent)
Description:	The priority that is associated with the dequeued call in the system.
Type:	Long
Potential Errors:	None
Notes:	Read-only

8.3.1.5 QueueKey

The QueueKey property is the key for the queue for the call that is being dequeued.

To use this property, do the following:

```
Dim lKey as Long
lKey = oDequeuedEvent.QueueKey
```

Property:	QueueKey (of DequeuedEvent)
Description:	The queue that is associated with the dequeued call in the system.
Type:	Long
Potential Errors:	None
Notes:	Read-only

8.4 EnqueuedEvent

The EnqueuedEvent object is an event object that is sent whenever a call is enqueued. These can be e-mail, voice, Web collaboration, or callback calls.

Interfaces supported:

- IEnqueuedEvent2 (default)
- IRoutingEvent
- IXMLAccess

8.4.1 Properties

This section contains properties exposed through the EnqueuedEvent interface. All properties from event objects are Read-only. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

8.4.1.1 CallID

The CallID property is the unique key for the call object in the system. This value is unique whether the call is a voice, e-mail, Web collaboration, or callback call. This is the same CallID that would match the VoiceCall, EmailCall, WebCollaborationCall, or CallbackCall CallID property.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oEnqueuedEvent.CallID
```

Property:	CallID (of EnqueuedEvent)
Description:	The unique key for a call in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

8.4.1.2 Description

The Description property is the description that is associated with call that is being enqueued. The description is a single line of text that can be displayed in the Screen Pop portion of the Client Desktop application. This can be used to provide some information about the queue that the queue itself may not be able to provide. For example, if the queue is 'Sales' the description might be 'Sales Call for All Wood Furniture'.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oEnqueuedEvent.Description
```

Property:	Description (of EnqueuedEvent)
Description:	The description that is associated with the enqueued call.
Type:	String
Potential Errors:	None
Notes:	None

8.4.1.3 Priority

The Priority property is the priority that is associated with the call that is being enqueued. The priority determines how this call is scored compared to other calls. The higher the priority of the call relative to other calls, the higher the importance of the call. This read/write property is also used so that the priority for this call object can be updated.

To use this property, do the following:

```
Dim lPriority as Long
lPriority = oEnqueuedEvent.Priority
```

Property:	Priority (of EnqueuedEvent)
Description:	The priority that is associated with the enqueued call.
Type:	Long
Potential Errors:	None
Notes:	Read-only

8.4.1.4 QueueKey

The QueueKey property is the key for the queue of the call that is being enqueued.

To use this property, do the following:

```
Dim lKey as Long
lQueuekey = oEnqueuedEvent. QueueKey
```

Property:	QueueKey (of EnqueuedEvent)
Description:	The queue that is associated with the enqueued call.
Type:	Long
Potential Errors:	None
Notes:	Read-only

8.5 RoutingBase

The RoutingBase interface is implemented by any objects that are valid objects for performing functions with the RoutingManager (for example, RoutingCall). This non-creatable object provides a common method of passing these database objects into the RoutingManager.

The use of this interface is implicit. When you have an object that supports the RoutingBase interface, you can pass that object into the COM method that takes a RoutingBase object as a parameter.

Interfaces supported:

- IRoutingBase

For example:

```
Private Sub cmdQuery_Click()
   m_oRoutingCall.CallID = txtCallID.Text
   Call g_oRoutingManager.Query (m_oRoutingCall)
End Sub
```

This accepts the RoutingCall interface and will convert it under the covers to the RoutingBase interface for the Query method to accept.

In most instances, this interface is not something that you would directly use, but rather is a convenient way to pass around grouped types. For example, you could write a method that would display all properties of the various RoutingManager objects (DisplayRoutingProperties (oRoutingBase as RoutingBase)).

8.5.1 Properties

This section contains properties exposed through the RoutingBase interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

8.5.1.1 Type

The Type property returns the object type that the RoutingBase object refers to. This can then be used to determine what to 'cast' the object to.

This property returns a value from the enRoutingBaseObjectTypes enumeration described in Section 3.38, “enRoutingBaseObjectTypes”, on page 102. This property is read-only, and is set internally in the parent object that the RoutingBase interface is exposed through.

To use this property, do the following:

```
Dim enType as HiPathProCenterLibrary.enRoutingBaseObjectType
enType = oRoutingBase.Type
```

Property:	Type (of RoutingBase)
Description:	The object type of the RoutingBase object the property is exposed through.
Type:	enRoutingBaseObjectTypes
Potential Errors:	None
Notes:	Read-only

8.6 RoutingCall

The RoutingCall object represents a call that is in the system and in the process of being matched to a user. These can be e-mail, voice, Web collaboration, or callback calls.

Interfaces supported:

- IRoutingCall2 (default)
- IRoutingBase
- IXMLAccess

8.6.1 Properties

This section contains properties exposed through the User interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

8.6.1.1 AssignAgentKey

The AssignAgentKey property is the user who is assigned to the call. This is set when the call has been assigned to a user.

To use this property, do the following:

```
Dim lAgentKey as Long
Set lAgentKey = oRoutingCall.AssignAgentKey
```

Property:	AssignAgentKey (of RoutingCall)
Description:	The agent key of the user who is assigned to the call.
Type:	Long
Potential Errors:	None
Notes:	Read-only

8.6.1.2 CallID

The CallID property is the unique key for the call object in the system. This value is unique whether the call is a voice, e-mail, Web collaboration, or callback call. This is the same CallID that would match the VoiceCall, EmailCall, WebCollaboration Call, or CallbackCall CallID property.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oRoutingCall.CallID
```

Property:	CallID (of RoutingCall)
Description:	The unique key for a call in the system.
Type:	String
Potential Errors:	None
Notes:	None

8.6.1.3 ContactData

The ContactData property is the contact data that is associated with this RoutingCall object. The ContactData property can provide the most information about a call. The ContactData object is a collection of key/value pairs associated with the Call. This can be used to hold information entered in the IVR, or information from other agents that have talked to this customer. For more information, see Section 4.1, "ContactData", on page 117. This read/write property is also used so that the contact data for this call object can be updated.

To use this property, do the following:

```
Dim oContactData as HiPathProCenterLibrary.ContactData
Set oContactData = oRoutingCall.ContactData
```

Property:	ContactData (of RoutingCall)
Description:	The contact data that is associated with this call in the system.
Type:	ContactData
Potential Errors:	None
Notes:	Read-only

8.6.1.4 Description

The Description property is the description that is associated with this RoutingCall object. The description is a single line of text that can be displayed in the Screen Pop portion of the Client Desktop application. This can be used to provide some information about the queue that the queue itself may not be able to do. For example, if the queue is 'Sales' the description might be 'Sales Call for All Wood Furniture'. This read/write property is also used so that the description for this call object can be updated.

To use this property, do the following:

```
Dim strDescription as String
strDescription = oRoutingCall.Description
```

Property:	Description (of RoutingCall)
Description:	The description that is associated with this call in the system.
Type:	String
Potential Errors:	None
Notes:	None

8.6.1.5 MediaType

The MediaType property is the media type for this RoutingCall object. This is required since a RoutingCall can be representing a call object of any of the valid media types.

To use this property, do the following:

```
Dim enMediaType as HiPathProCenterLibrary.enMediaTypes
enMediaType = oRoutingCall.MediaType
```

Property:	MediaType (of RoutingCall)
Description:	The media type for a call in the system.
Type:	enMediaTypes
Potential Errors:	None
Notes:	Read-only

8.6.1.6 Priority

The Priority property is the priority that is associated with this RoutingCall object. The priority determines how this call is scored compared to other calls. The higher the priority of the call relative to other calls, the higher the importance of the call. This read/write property is also used so that the priority for this call object can be updated.

To use this property, do the following:

```
Dim lPriority as Long
lPriority = oRoutingCall.Priority
```

Property:	Priority (of RoutingCall)
Description:	The priority that is associated with this call in the system.
Type:	Long
Potential Errors:	None
Notes:	None

8.6.1.7 QueueKey

The QueueKey property is the key for the queue that is associated with this RoutingCall object. This read/write property is also used so that the queue for this call object can be updated.

To use this property, do the following:

```
Dim lKey as Long
lKey = oRoutingCall.QueueKey
```

Property:	QueueKey (of RoutingCall)
Description:	The queue that is associated with this call in the system.
Type:	Long
Potential Errors:	None
Notes:	Read-only

8.6.1.8 ReserveAgentKey

The ReserveAgentKey property is the agent key that the call is reserved for. This is set when the call has been reserved for a user.

To use this property, do the following:

```
Dim lAgentKey as Long
Set lAgentKey = oRoutingCall.ReserveAgentKey
```

Property:	ReserveAgentKey (of RoutingCall)
Description:	The agent key of the user who this call is reserved for.
Type:	Long
Potential Errors:	None
Notes:	None

8.6.1.9 ReserveAgentMaximumTime

The ReserveAgentMaximumTime property is the amount of time (in seconds) that a call is to be reserved for a user. This is set when the call has been reserved for a user.

To use this property, do the following:

```
Dim lTimeInSeconds as Long
Set lTimeInSeconds = oRoutingCall.ReserveAgentMaximumTime
```

Property:	ReserveAgentMaximumTime (of RoutingCall)
Description:	The amount of time (in seconds) that a call is to be reserved.
Type:	Long
Potential Errors:	None
Notes:	None

8.6.1.10 State

The State property is the current state of this RoutingCall object. This tracks at what step in the routing process the RoutingCall is at. For example, has the called just entered the contact center, has the call already been queued, and so on.

To use this property, do the following:

```
Dim enState as HiPathProCenterLibrary.enCallStates
enState = oRoutingCall.State
```

Property:	State (of RoutingCall)
Description:	The state of a call in the system.
Type:	enCallStates
Potential Errors:	None
Notes:	Read-only

8.6.2 Methods

This section contains methods exposed through the RoutingCall interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

8.6.2.1 Dequeue

```
Dequeue (AbandonCall as Boolean)
```

The RoutingCall method provides the functionality to dequeue a call. To do this, you can use the Enqueue method. The only thing that you need to pass in is whether or not this call should be considered abandoned or not.

This method accepts the following parameter:

AbandonedCall as Boolean	This is whether or not you want to track the call as an abandoned call (True) or not (False).

This method does not return anything.

To use this method, do the following after creating the RoutingCall:

```
Call oRoutingCall.Dequeue
```

This dequeues the call from the system.

Method:	Dequeue (of RoutingCall)
Description:	Dequeues a call to the Routing Server to be assigned to a user.
Parameters:	AbandonCall — If the call should be tracked as an abandon (True) or not (False).
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

8.6.2.2 Enqueue

```
Enqueue (AutomatedTransfer as Boolean, (Optional) Destination as
String = "", (Optional) Source as String = "")
```

The RoutingCall provides the functionality to enqueue a call to a queue and optionally reserve a call for a user. To do this, you can use the Enqueue method to determine whether you want the Routing Server to automatically transfer the call to the assigned user, or if you want to be notified and will handle the transfer yourself (for example, in an IVR hold scenario). You may also pass the Destination and Source that you want to associate with the Enqueued call.

This method accepts the following parameters:

Automated Transfer as Boolean	This determines whether or not you want the call to be automatically transferred to a user when the user becomes available (Queue Hold) or if you want to be notified when the user is available and will handle routing the call to the user yourself (IVR Hold). For Queue Hold, pass in True, for IVR Hold, pass in False.
Destination as String	This is the destination that you want to associate with the call (for example, the DNIS associated with a voice call).
Source as String	This is the source that you want to associate with the call (for example, the ANI associated with a voice call).

This method does not return anything.

To use this method, do the following after creating the RoutingCall:

```
Call oRoutingCall.Enqueue (True)
```

or…

```
Call oRoutingCall.Enqueue (False, "1203", "555-5555")
```

This enqueues the call to the system. In the first scenario, you will enqueue the call with the default value for Destination and Source, and you will have the Routing Server handle routing the call to the user (Queue Hold). In the second case, you will update the Destination and Source parameters, and you will also handle routing the call to the user when the user becomes available (IVR Hold).

Some of the possible scenarios for enqueueing a call are as follows:

Queue Key and Source and Destination Specified	Routing Server uses the queue key to determine an eligible user. The source and destination are specified in the resulting Call object that's created.
No Queue Key but Source and Destination Specified	Routing Server uses the source and destination to determine a queue to use.
No Queuekey and No Source and Destination	An error is returned.

In all cases, if the Routing Server cannot determine the queuekey, an error is returned.

NOTE: Although the Routing Server will process an Enqueue request, such as find an eligible user and send an Assigned event to the corresponding server, it should not be performed on e-mail, Web, or callback calls, because the normal behavior for these servers is to enqueue calls directly.

Method:	Enqueue (of RoutingCall)
Description:	Enqueues a call to the Routing Server to be assigned to a user.
Parameters:	• AutomatedTransfer — If the Routing Server will transfer the call (Queue Hold) or not (IVR Hold). • Destination("") — The destination to associate with the call (DNIS for a call). • Source("") — The source to associate with the call (ANI for a call).
Returns:	None
Potential Errors:	• Error_Generic_ActionFailed • Error_Generic_NetworkBusy • Error_Generic_RequestTimedOut • Error_Generic_ValidationFailed • Error_Media_ActionResultInvalidExtension • Error_Media_ActionResultLineBusy • Error_Media_ActionResultOutOfService • Error_Media_ActionResultResourceBusy • Error_Media_ActionResultResourceError • Error_Media_ActionResultStateError • Error_Media_ActionResultTelephonyServerNotReady
Notes:	None

8.7 RoutingEvent

The RoutingEvent is an interface that contains a summary of the event that has occurred. All events sent back from the RoutingManager will support the RoutingEvent interface. This non-creatable object also provides a common method of passing all events that occur in the RoutingManager to the applications using the SDK. This enables various pieces of information to be returned from the RoutingManager and performs simple handling before determining how (and if) you want to handle this event.

Applications can get specific information about the event by querying for other interfaces that are supported by the event interface. To determine what detailed object type the event returned, you can use the ObjectType property, and then ask for the appropriate interface.

Interfaces supported:

- IRoutingEvent

For example:

```
Private Sub g_oRoutingManager_EventOccurred
   (ByVal RoutingEvent as HiPathProCenterLibrary.RoutingEvent)
Select Case RoutingEvent.ObjectType
   Case RoutingEventObjectType_Enqueued
      ' Here since you have a Enqueued event,
      ' you query for the Enqueued event interface.
      Dim oEnqueuedEvent as HiPathProCenterLibrary.EnqueuedEvent
      Set oEnqueuedEvent = RoutingEvent
      MsgBox oEnqueuedEvent.CallID &
      " has been enqueued."
   Case RoutingEventObjectType_ManagerStateChanged
      ' Here though you have a ManagerStateChanged event,
      ' so you query for the ManagerStateChanged interface.
      Dim oManagerStateChanged as
      HiPathProCenterLibrary.ManagerStateChangedEvent
      Set oManagerStateChanged = RoutingEvent
      MsgBox "RoutingManager is now " &
      oManagerStateChanged.State
   Default
      MsgBox "Unknown ObjectType"
End Select
End Sub
```

8.7.1 Properties

This section contains properties exposed through the RoutingEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

8.7.1.1 Code

The Code property returns the event code of the event that has occurred in the RoutingManager. In most instances this will be directly related to the ObjectType that is also contained in the RoutingEvent interface. This property determines what has happened. Since the ObjectType refers to the object that is returned, they will be related.

This property returns a value from the enRoutingEventCodes enumerations described in Section 3.39, “enRoutingEventCodes”, on page 103. This property is read-only, and is set internally in the RoutingManager before it fires the event back through the EventOccurred event handler. For more information , see Section 8.1.3.1, “EventOccurred”, on page 541.

To use this property, do the following:

```
Dim enCode as HiPathProCenterLibrary.enRoutingEventCodes
enCode = oRoutingEvent.Code
```

Property:	Code (of RoutingEvent)
Description:	The code of what has occurred in the RoutingManager.
Parameters:	RoutingEvent — The RoutingEvent interface for the event that has occurred.
Type:	enRoutingEventCodes
Potential Errors:	None
Notes:	Read-only

8.7.1.2 EventType

The EventType property returns the event type that this event corresponds with. This event type is the event that was listened for this event to be sent. If the event is always going to be sent (Error or ManagerStateChanged event), then this property will be RoutingEventType_NotSet.

This property returns a value from the enRoutingEventTypes enumerations described in Section 3.41, "enRoutingEventTypes", on page 106. This property is read-only, and is set internally in the RoutingManager before it fires the event back through the EventOccurred event handler. For more information, see Section 8.1.3.1, "EventOccurred", on page 541.

To use this property, do the following:

```
Dim enEventType as HiPathProCenterLibrary.RoutingEventTypes
enEventType = oRoutingEvent.EventType
```

Property:	EventType (of RoutingEvent)
Description:	The event type of the RoutingManager event that is sent back.
Type:	enRoutingEventTypes
Potential Errors:	None
Notes:	This determines what event group the object belongs to.

8.7.1.3 MediaType

The MediaType property returns the media type to which the RoutingEvent applies.

This property returns a value from the enMediaTypes enumerations described in Section 3.34, "enMediaTypes", on page 100. This property is read-only, and is set internally in the RoutingManager before it fires the event back through the EventOccurred event handler. For more information, see Section 8.1.3.1, "EventOccurred", on page 541.

To use this property, do the following:

```
Dim enMediaType as HiPathProCenterLibrary.enMediaTypes
enMediaType = oRoutingEvent.MediaType
```

Property:	MediaType (of RoutingEvent)
Description:	The media type of the RoutingManager event that is sent back.

Type:	enMediaTypes
Potential Errors:	None
Notes:	None

8.7.1.4 ObjectType

The ObjectType property returns the object interface type of the detailed event information contained in the RoutingEvent. In most instances this will be directly related to the Code that is also contained in the RoutingEvent interface. This property determines what the detailed interface is that is contained in the RoutingEvent.

This property returns a value from the enRoutingEventObjectTypes enumerations described in Section 3.40, "enRoutingEventObjectTypes", on page 105. This property is read-only, and is set internally in the RoutingManager before it fires the event back through the EventOccurred event handler. For more information, see Section 8.1.3.1, "EventOccurred", on page 541.

To use this property, do the following:

```
Dim enObjectType as
HiPathProCenterLibrary.enRoutingEventObjectTypes

enObjectType = oRoutingEvent.ObjectType
```

Property:	ObjectType (of RoutingEvent)
Description:	The detailed object type of the RoutingManager event that is sent back.
Type:	enRoutingEventObjectTypes
Potential Errors:	None
Notes:	This determines the object type you can query for more detailed event information.

8.7.1.5 Resource

The Resource property returns the resource that this event corresponds with. The resource is a string value that is specific for an event. This is the resource that is used when registering for an event through the ListenForEvent method.

To use this property, do the following:

```
Dim strResource as String

strResource = oRoutingEvent.Resource
```

Property:	Resource (of RoutingEvent)
Description:	The resource that corresponds to the RoutingManager event that is sent back.
Type:	String
Potential Errors:	None
Notes:	This determines the resource the event is associated with.

8.8 RoutingInformationUpdatedEvent

The RoutingInformationUpdatedEvent object is an event object that is sent when the call details have been updated in the system. These events are sent from the Routing Manager when the call is enqueued and waiting to be assigned to a user, after it has been assigned to a user, these events come from the Media Manager. These can be e-mail, voice, Web collaboration, or callback calls.

Interfaces supported:

- IRoutingInformationUpdatedEvent2 (default)
- IRoutingEvent
- IXMLAccess

8.8.1 Properties

This section contains properties exposed through the RoutingInformationUpdatedEvent interface. All properties from event objects are Read-only. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

8.8.1.1 CallID

The CallID property is the unique key for the call object in the system. This value is unique whether the call is a voice, e-mail, Web collaboration, or callback call. This is the same CallID that would match the VoiceCall, EmailCall, WebCollaboration Call, or CallbackCall CallID property.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oRoutingInformationUpdatedEvent.CallID
```

Property:	CallID (of RoutingInformationUpdatedEvent)
Description:	The unique key for a call in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

8.8.1.2 ContactData

The ContactData property is the contact data that is associated with the call object specified by the CallID. The ContactData property can provide the most information about a call. The ContactData object is a collection of key/value pairs associated with the Call. This can be used to hold information entered in the IVR, or information from other agents that have talked to this customer. For more information, see Section 4.1, "ContactData", on page 117. If ObjectType property is RoutingEventObjectType_ContactDataUpdated then ContactData property has updated ContactData.

To use this property, do the following:

```
Dim oContactData as HiPathProCenterLibrary.ContactData
If oRoutingInformationUpdatedEvent. ObjectType =
RoutingEventObjectType_ContactDataUpdated Then
   Set oContactData =
   oRoutingInformationUpdatedEvent.ContactData
End If
```

Property:	ContactData (of RoutingInformationUpdatedEvent)
Description:	The contact data that is associated with the this call in the system.
Type:	Contact Data
Potential Errors:	None
Notes:	Read-only

8.8.1.3 Description

The Description property is the description that is associated with the call object specified by the CallID. The description is a single line of text that can be displayed in the Screen Pop portion of the Client Desktop application. This can be used to provide some information about the queue that the queue itself may not be able to do. For example, if the queue is 'Sales' the description might be 'Sales Call for All Wood Furniture'.

To use this property, do the following:

```
Dim strDescription as String
If oRoutingInformationUpdatedEvent. ObjectType =
RoutingEventObjectType_Updated Then
strDescription = oRoutingInformationUpdatedEvent.Description
```

```
End If
```

Property:	Description (of RoutingInformationUpdatedEvent)
Description:	The description that is associated with this call in the system.
Type:	String
Potential Errors:	None
Notes:	None

8.8.1.4 Priority

The Priority property is the priority that is associated with the call that has been updated. The priority determines how this call is scored compared to other calls. The higher the priority of the call relative to other calls, the higher the importance of the call. This read/write property is also used so that the priority for this call object can be updated.

To use this property, do the following:

```
Dim lPriority as Long
lPriority = oRoutingInformationUpdatedEvent.Priority
```

Property:	Priority (of RoutingInformationUpdatedEvent)
Description:	The priority that is associated with the updated call in the system.
Type:	Long
Potential Errors:	None
Notes:	Read-only

8.8.1.5 QueueKey

The QueueKey property is the key for the queue for the call that has been updated.

To use this property, do the following:

```
Dim lKey as Long
lKey = oRoutingInformationUpdatedEvent.QueueKey
```

Property:	QueueKey (of RoutingInformationUpdatedEvent)
Description:	The queue that is associated with this updated call in the system.

Type:	Long
Potential Errors:	None
Notes:	Read-only

8.9 TimedOutEvent

The TimedOutEvent object is an event object that is sent whenever a call has timed out. This can occur when a call has sat in queue, waiting to be assigned to a user, and has reached the maximum amount of time that the call should wait. These can be e-mail, voice, Web collaboration, or callback calls.

Interfaces supported:

- ITimedOutEvent
- IRoutingEvent
- IXMLAccess

8.9.1 Properties

This section contains properties exposed through the TimedOutEvent interface. All properties from event objects are Read-only. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

8.9.1.1 CallID

This value is unique whether the call is a voice, e-mail, Web collaboration, or callback call. This is the same CallID that would match the VoiceCall, EmailCall, WebCollaborationCall, or CallbackCall CallID property.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oTimedOutEvent.CallID
```

Property:	CallID (of TimedOutEvent)
Description:	The unique key for a call in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

8.9.1.2 TelephoneNumber

The TelephoneNumber property is the telephone number of where the call has been transferred to (the time out extension).

To use this property, do the following:

```
Dim strTelephoneNumber as String
strTelephoneNumber = oTimedOutEvent.TelephoneNumber
```

Property:	TelephoneNumber (of TimedOutEvent)
Description:	The telephone number of where the call has been transferred to (the timeout extension).
Type:	String
Potential Errors:	None
Notes:	Read-only

8.10 UnassignedEvent

The UnassignedEvent object is an event object that is sent whenever a call is unassigned from a user in the system. These can be e-mail, voice, Web collaboration, or callback calls.

Interfaces supported:

- IUnassignedEvent
- IRoutingEvent
- IXMLAccess

8.10.1 Properties

This section contains properties exposed through the UnassignedEvent interface. All properties from event objects are Read-only. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

8.10.1.1 AgentKey

The AgentKey property is the key for the user who the call has been unassigned from.

To use this property, do the following:

```
Dim lAgentKey as Long
lAgentKey = oUnassignedEvent.AgentKey
```

Property:	AgentKey (of UnassignedEvent)
Description:	The agent key for the user who this call has been unassigned from.
Type:	Long
Potential Errors:	None
Notes:	Read-only

8.10.1.2 CallID

The CallID property is the unique key for the call object in the system. This value is unique whether the call is a voice, e-mail, Web collaboration, or callback call. This is the same CallID that would match the VoiceCall, EmailCall, WebCollaborationCall, or CallbackCall CallID property.

To use this property, do the following:

```
Dim strCallID as String
strCallID = oUnassignedEvent.CallID
```

Property:	CallID (of UnassignedEvent)
Description:	The unique key for a call in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

9 Statistics Manager

The Statistics Manager handles the features of the system that are related to statistics, current and historical.

9.1 StatisticsManager object

The StatisticsManager is a non-creatable object that provides access to the various statistic functionalities in the system.

Interfaces supported:

- IStatisticsManager2 (default)

9.1.1 Properties

This section contains properties exposed through the StatisticsManager interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.1.1.1 State

The State property returns the current state of the StatisticsManager. This refers to the state that the StatisticsManager is in (Available or Unavailable).

This property returns a value from the enManagerStates enumerations described in Section 3.28, “enManagerStates”, on page 87. This value is read-only.

To use this property, do the following:

```
Dim enState as HiPathProCenterLibrary.enStates
enState = g_oStatisticsManager.State
```

Property:	State
Description:	The state of the Statistics Manager (Available/Unavailable).
Type:	enManagerStates
Potential Errors:	None
Notes:	Read-only

9.1.2 Methods

This section contains methods exposed through the StatisticsManager interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.1.2.1 GetFunctionalityState

```
enManagerStates = GetFunctionalityState (StatisticsFunctionality
as enStatisticsFunctionalities)
```

The StatisticsManager may occasionally lose its connection to one or more servers in the system. When this occurs, you can check to see what functionality is supported. To do this, you can use the GetFunctionalityState method to pass in the functionality that you are interested in to return the state of that functionality.

This method accepts the following parameter:

Statistics Functionality as enStatistics Functionalities	This is the functionality group that you want to enquire about its. See Section 3.49, "enStatisticsFunctionalities", on page 113.

This method returns the ManagerState of the functionality group you have requested. For more information, see Section 3.28, "enManagerStates", on page 87.

To use this method, do the following after creating the StatisticsManager:

```
Dim enManagerState as HiPathProCenterLibrary.enManagerStates

enManagerState = g_oStatisticsManager.GetFunctionalityState
(StatisticsFunctionality_Historical)
```

This checks with the StatisticsManager to see if you have the ability to receive historical statistical information.

Method:	GetFunctionalityState
Description:	Queries the state of various functionality groups in the StatisticsManager.
Parameters:	StatisticsFunctionality — The functionality group you want to check the state of.
Returns:	enManagerStates — The state of the functionality group you requested.
Potential Errors:	None
Notes:	None

9.1.2.2 ListenForEvents

ListenForEvents () method enables SDK client applications to initiate statistics queries. After a query is created, you will start to receive StatisticsEvents that contain the results. You will keep receiving these result events until you call the corresponding StopListeningForEvents method.

This method accepts the following parameters:

enStatisticsEventType as enStatisticsEventTypes	This is the type of event required.
keyList as IKeyList	This is the list of keys for the selected queues to be queried. It is a read/write collection of integers (longs). This is used to pass the collection of keys that are set by the SDK client application to narrow down the query. This collection differs in each request depending of the type of statistics events queried: for example, they could be queue IDs or user Keys.
from as Date	This is the start time of the statistical period for the original event (the equivalent of RID_SINCE RangeType).

To use this method, do the following:

```
queryID = m_oStatsManager.ListenForEvents(enStatisticsEventType,
keyList, from)
```

Method:	ListenForEvents
Description:	Enables SDK client applications to initiate statistics queries.
Parameters:	• enStatisticsEventType — The type of event required. • From — The start time of the statistical period for the original event • keyList — The list of keys for the selected queues to be queried.
Returns:	queryID — The QueryID associated with the query that has been created.
Potential Errors:	• Error_Generic_QueryFailed • Error_Generic_UnknownEventType

9.1.2.3 StopListeningForEvents

StopListeningForEvents () method is used to stop receiving event notifications for a previous request.

This method accepts the following parameters:

queryID as Long	This is the query ID passed back from the ListenForEvents () request.

To use this method, do the following:

```
Call m_oStatsManager.StopListeningforEvents(queryID)
```

Method:	StopListeningForEvents
Description:	Stops receiving event notifications for a previous request.
Parameters:	queryID — The query ID passed back from the ListenForEvents () request.
Returns:	None
Potential Errors:	Error_Statistics_StopListeningForEventsFailed

9.2 AggregateCumulativeElement

The AggregateCumulativeElement object provides the cumulative statistics for a single aggregate in a given time range.

Interfaces supported:

- IAggregateCumulativeElement (default)
- IXMLAccess

9.2.1 Properties

This section contains properties exposed through the AggregateCumulativeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.2.1.1 Abandoned

To use this property, do the following:

```
Dim lAbandoned as Long
lAbandoned = oAggregateCumulativeElement.Abandoned
```

Property:	Abandoned (of AggregateCumulativeElement)
Description:	The number of contacts that were abandoned while waiting in the aggregate or while being offered to the user.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.2.1.2 AbandonedRate

To use this property, do the following:

```
Dim fAbandonedRate as Single
fAbandonedRate = oAggregateCumulativeElement.AbandonedRate
```

Property:	AbandonedRate (of AggregateCumulativeElement)
Description:	The percentage of contacts routed to the aggregate that were abandoned before being answered.
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.2.1.3 AggregateKey

The AggregateKey property is the key of the aggregate that you want to get the statistics for.

To use this property, do the following:

```
Dim lAggregateKey as Long
lAggregateKey = oAggregateCumulativeElement.AggregateKey
```

Property:	AggregateKey (of AggregateCumulativeElement)
Description:	The database table unique key for aggregate that you want to get the statistics for.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.2.1.4 Answered

To use this property, do the following:

```
Dim lAnswered as Long
lAnswered = oAggregateCumulativeElement.Answered
```

Property:	Answered (of AggregateCumulativeElement)
Description:	The number of contacts routed to the aggregate that were answered.
Type:	Long
Potential Errors:	None
Notes:	Read-Only

9.2.1.5 AverageWaitTime

To use this property, do the following:

```
Dim fAverageWaitTime as Single
fAverageWaitTime = oAggregateCumulativeElement.AverageWaitTime
```

Property:	AverageWaitTime (of AggregateCumulativeElement)
Description:	The average amount of time that contacts waited in the aggregate before being answered, abandoned, or redirected.
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.2.1.6 MaximumWaitTime

To use this property, do the following:

```
Dim lMaximumWaitTime as Long
lMaximumWaitTime = oAggregateCumulativeElement.MaximumWaitTime
```

Property:	MaximumWaitTime (of AggregateCumulativeElement)
Description:	The maximum amount of time that a contact waited in the aggregate before being answered, abandoned, or redirected.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.2.1.7 Received

To use this property, do the following:

```
Dim lReceived as Long
lReceived = oAggregateCumulativeElement.Received
```

Property:	Received (of AggregateCumulativeElement)
Description:	The number of contacts that were received in the aggregate.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.2.1.8 Redirected

To use this property, do the following:

```
Dim lRedirected as Long
lRedirected = oAggregateCumulativeElement.Redirected
```

Property:	Redirected (of AggregateCumulativeElement)
Description:	The number of contacts received in the aggregate that were redirected.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.2.1.9 ServiceLevel

To use this property, do the following:

```
Dim fServiceLevel as Single
fServiceLevel = oAggregateCumulativeElement.ServiceLevel
```

Property:	ServiceLevel (of AggregateCumulativeElement)
Description:	The percentage of contacts routed to the aggregate that were answered within the site-defined or queue-defined service level.
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.3 AggregateCumulativeEvent

The AggregateCumulativeEvent object represents a collection of zero (0) or more time ranges. Each time range contains a collection of zero (0) or more Aggregate Cumulative Elements. Each Aggregate Cumulative Element contains the statistics for the specified aggregate within the given time range.

Interfaces supported:

- IAggregateCumulativeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.3.1 Properties

This section contains properties exposed through the AggregateCumulativeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.3.1.1 _NewEnum

```
For Each oTimeRange in colAggregateCumulativeEvent
   ...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oTimeRange as HiPathProCenterLibrary.TimeRange
For Each oTimeRange in colAggregateCumulativeEvent
   txtTextBox.Text = txtTextBox.Text & _
   "Time Range: " & _
   oTimeRange.TimeRange & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the Time Ranges associated with all the TimeRange collections in AggregateCumulativeEvent.

Property:	_NewEnum
Description:	Provides access to elements of the AggregateCumulativeEvent collection by means of operator FOREACH.
Returns:	OTimeRange- for each item in the AggregateCumulativeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownEventType
Notes:	This method allows the AggregateCumulativeEvent collection to support the 'For Each' iterator.

9.3.1.2 Count

The Count property returns the number of TimeRange collections of AggregateCumulativeElement objects that exist in the AggregateCumulativeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colAggregateCumulativeEvent.Count
```

Property:	Count
Description:	The number of TimeRange collections of AggregateCumulativeElement objects in AggregateCumulativeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.3.1.3 Item

```
oTimeRange = Item (time as Date)
```

The Item method finds a TimeRange collection of Aggregate Cumulative Elements in the AggregateCumulativeEvent collection that is indexed and sorted on time interval. If an item is found in the collection at the specified time interval, the Item method returns the TimeRange collection of AggregateCumulativeElement objects. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

time as Date	This is the time interval associated with the TimeRange collection item you are requesting.

This method returns the following parameters:

oTimeRange as TimeRange	This is the TimeRange collection object for TimeRange specified by the time value. This contains a collection of AggregateCumulativeElement objects.

To use this method, do the following:

```
Dim oTimeRange as HiPathProCenterLibary.TimeRange
Set oTimeRange = colAggregateCumulativeEvent.Item (time)
```

This checks to see if the TimeRange specified by the time value exists. If it does, oTimeRange will contain the TimeRange object. If it does not exist in the AggregateCumulativeEvent collection, this will raise an error.

Property:	Item
Description:	Finds a TimeRange collection in the AggregateCumulativeEvent collection that is indexed and sorted on time interval.
Parameters:	Time — The interval associated with the TimeRange collection item you are requesting.
Returns:	oTimeRange — The TimeRange collection object for TimeRange specified by the time value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the TimeRange collection does not exist in the AggregateCumulativeEvent, It will raise an error.

9.3.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a ListenForEvents request. This QueryID is uniquely generated every time the client calls ListenForEvents method.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colAggregateCumulativeEvent.QueryID
```

Property:	QueryID
Description:	The QueryID associated with AggregateCumulativeEvent collection.

Type:	Long
Potential Errors:	None
Notes:	Read-only

9.4 AggregateRealtimeElement

The AggregateRealtimeElement object provides the statistics for a single aggregate.

Interfaces supported:

- IAggregateRealtimeElement (default)
- IXMLAccess

9.4.1 Properties

This section contains properties exposed through the AggregateRealtimeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.4.1.1 AbandonedRate

To use this property, do the following:

```
Dim fAbandonedRate as Single
fAbandonedRate = oAggregateRealtimeElement.AbandonedRate
```

Property:	AbandonedRate
Description:	The weighted percentage of the last 24 contacts routed to the aggregate that were abandoned.
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.4.1.2 AggregateKey

The AggregateKey property is the key of the aggregate that you want to get the statistics for.

To use this property, do the following:

```
Dim lAggregateKey as Long
lAggregateKey = oAggregateRealtimeElement.AggregateKey
```

Property:	AggregateKey
Description:	The database table unique key of the aggregate that you want to get the statistics for.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.4.1.3 AverageAbandonedWaitTime

To use this property, do the following:

```
Dim fAverageAbandonedWaitTime as Single
fAverageAbandonedWaitTime =
oAggregateRealtimeElement.AverageAbandonedWaitTime
```

Property:	AverageAbandonedWaitTime
Description:	The average amount of time that contacts waited in the aggregate before being abandoned, based on the last 10 contacts that were abandoned..
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.4.1.4 AverageAnsweredWaitTime

To use this property, do the following:

```
Dim fAverageAnsweredWaitTime as Single

fAverageAnsweredWaitTime =
oAggregateRealtimeElement.AverageAnsweredWaitTime
```

Property:	AverageAnsweredWaitTime
Description:	The average amount of time that contacts waited in the aggregate before being answered, based on the last 10 contacts that were answered.
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.4.1.5 Contacts

To use this property, do the following:

```
Dim sContacts as Integer

sContacts = oAggregateRealtimeElement.Contacts
```

Property:	Contacts
Description:	The number of contacts that are waiting in the aggregate.
Type:	Short
Potential Errors:	None
Notes:	Read-only

9.4.1.6 EstimatedAnsweredWaitTime

To use this property, do the following:

```
Dim lEstimatedAnsweredWaitTime as Long

lEstimatedAnsweredWaitTime =
oAggregateRealtimeElement.EstimatedAnsweredWaitTime
```

Property:	EstimatedAnsweredWaitTime
Description:	The estimated amount of time that contacts will spend in the aggregate before being routed to a user.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.4.1.7 EstimatedServiceLevel

To use this property, do the following:

```
Dim fEstimatedServiceLevel as Single

fEstimatedServiceLevel =
oAggregateRealtimeElement.EstimatedServiceLevel
```

Property:	EstimatedServiceLevel
Description:	The estimated service level of the aggregate..
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.4.1.8 OldestContactWaitTime

To use this property, do the following:

```
Dim lOldestContactWaitTime as Long

lOldestContactWaitTime =
oAggregateRealtimeElement.OldestContactWaitTime
```

Property:	OldestContactWaitTime
Description:	The amount of time that the oldest contact has been waiting in the aggregate..
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.4.1.9 OverflowedContacts

To use this property, do the following:

```
Dim sOverflowedContacts as Integer

sOverflowedContacts =
oAggregateRealtimeElement.OverflowedContacts
```

Property:	OverflowedContacts
Description:	The number of contacts waiting in the aggregate that have overflowed.
Type:	Short
Potential Errors:	None
Notes:	Read-only

9.4.1.10 ServiceLevel

To use this property, do the following:

```
Dim fServiceLevel as Single
fServiceLevel = oAggregateRealtimeElement.ServiceLevel
```

Property:	ServiceLevel
Description:	The current service level, calculated based on the last 24 contacts routed to the aggregate.
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.5 AggregateRealtimeEvent

The AggregateRealtimeEvent object represents a collection of one or more Aggregate Real-time Elements. Each Aggregate Real-time Element contains the statistics for one aggregate.

Interfaces supported:

- IAggregateRealtimeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.5.1 Properties

This section contains properties exposed through the AggregateRealtimeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.5.1.1 Count

The Count property returns the number of AggregateRealtimeElement objects that exist in the AggregateRealtimeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colAggregateRealtimeEvent.Count
```

Property:	Count
Description:	The number of AggregateRealtimeElement objects in AggregateRealtimeEvent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.5.1.2 Item

```
oAggregateRealtimeElement = Item (Key as long)
```

The Item method finds an Aggregate Real-time Element object in the AggregateRealtimeEvent collection that is indexed by the AggregateKey. If an item is found in the collection at the key value, the Item method returns the AggregateRealtimeElement object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

Key as Long	This is the AggregateKey for the AggregateRealtimeElement item you are requesting.

This method returns the following parameters:

oAggregate RealtimeElemen t as Aggregate RealtimeElemen t	This is the AggregateRealtimeElement object for AggregateRealtimeElement specified by the Key value. This contains the statistics for the AggregateRealtimeElement.

To use this method, do the following:

```
Dim oAggregateRealtimeElement as
HiPathProCenterLibary.AggregateRealtimeElement
Set oAggregateRealtimeElement = colAggregateRealtimeEvent.Item
(Key)
```

This checks to see if the AggregateRealtimeElement specified by the Key value exists. If it does, oAggregateRealtimeElement will contain the AggregateRealtimeElement object. If it does not exist in the AggregateRealtimeEvent, this will raise an error.

Property:	Item
Description:	Finds an AggregateRealtimeElement in the AggregateRealtimeEvent collection that is indexed by the Aggregate Key.
Parameters:	Key — The key for the AggregateRealtimeElement that you are looking for.
Returns:	oAggregateRealtimeElement — The AggregateRealtimeElement object specified by the key value.
Potential Errors:	Error_Generic_CreatingObject Error_Generic_ItemNotFound
Notes:	If the AggregateRealtimeElement object does not exist in the AggregateRealtimeEvent, It will raise an error.

9.5.1.3 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a ListenForEvents request. This QueryID is uniquely generated every time the client calls ListenForEvents method.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colAggregateRealtimeEvent.QueryID
```

Property:	QueryID
Description:	The QueryID associated with AggregateRealtimeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.6 AggregateWrapupReasonCumulativeElement

The AggregateWrapupReasonCumulativeElement object represents an individual record in the AggregateWrapupReasonCumulativeEvent collection.

Interfaces supported:

- IAggregateWrapupReasonCumulativeElement (default)
- IXMLAccess

9.6.1 Properties

This section contains properties exposed through AggregateWrapupReasonCumulativeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.6.1.1 AggregateKey

The AggregateKey property returns the aggregate key value.

To use this property, do the following:

```
Dim lKey as Long
lKey = colAggregateWrapupReasonCumulativeElement.AggregateKey
```

Property:	AggregateKey (of AggregateWrapupReasonCumulativeElement)
Description:	Returns an aggregate key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.6.1.2 Count

The Count property returns the wrapup reason statistic value.

To use this property, do the following:

```
Dim lCount as Long
lCount = colAggregateWrapupReasonCumulativeElement.Count
```

Property:	Count (of AggregateWrapupReasonCumulativeElement)
Description:	The number of TimeRange objects AggregateWrapupReasonCumulativeElement.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.6.1.3 WrapupReasonKey

The WrapupReasonKey property returns the wrapup reason key value.

To use this property, do the following:

```
Dim lKey as Long
lKey = colAggregateWrapupReasonCumulativeElement.WrapupReasonKey
```

Property:	WrapupReasonKey (of AggregateWrapupReasonCumulativeElement)
Description:	Returns a wrapup reason key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.7 AggregateWrapupReasonCumulativeEvent

The AggregateWrapupReasonCumulativeEvent object represents a collection of one or more of TimeRange objects each containing one or more AggregateWrapupReasonCumulativeElement objects.

Interfaces supported:

- IAggregateWrapupReasonCumulativeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.7.1 Properties

This section contains properties exposed through the AggregateWrapupReasonCumulativeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.7.1.1 _NewEnum

```
For Each oTimeRange in colAggregateWrapupReasonCumulativeEvent
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oTimeRange as HiPathProCenterLibrary.TimeRange
For Each oTimeRange in colAggregateWrapupReasonCumulativeEvent
...
Next
```

Property:	_NewEnum (of AggregateWrapupReasonCumulativeEvent)
Description:	Provides access to elements of the AggregateWrapupReasonCumulativeEvent by means of operator FOREACH.
Returns:	oTimeRange — For each item in the AggregateWrapupReasonCumulativeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the AggregateWrapupReasonCumulativeEvent collection to support the 'For Each' iterator.

9.7.1.2 Count

The Count property returns the number of TimeRange objects that exist in the AggregateWrapupReasonCumulativeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colAggregateWrapupReasonCumulativeEvent.Count
```

Property:	Count (of AggregateWrapupReasonCumulativeEvent)
Description:	The number of TimeRange objects AggregateWrapupReasonCumulativeEvent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.7.1.3 Item

```
oTimeRange = Item (range as Date)
```

The Item method finds a Time Range object in the AggregateWrapupReasonCumulativeEvent collection that is indexed by the range parameter. If an item is found in the collection at the range value, the Item method returns the TimeRange object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

range as Date	This is the time range for the item you are requesting.

This method returns the following parameters:

oTimeRange as ITimeRange	This is the TimeRange object for the specified "range" value. This contains the collection of one or more of AggregateWrapupReasonCumulativeElement objects.

To use this method, do the following:

```
Dim oTimeRange as HiPathProCenterLibary.TimeRange
Set oTimeRange = colAggregateWrapupReasonCumulativeElement.Item
(range)
```

This checks to see if the TimeRange specified by the range value exists. If it does, oTimeRange will contain the TimeRange object. If it does not exist in the AggregateWrapupReasonCumulativeEvent, this will raise an error.

Property:	Item (of AggregateWrapupReasonCumulativeEvent)
Description:	Finds a TimeRange in the AggregateWrapupReasonCumulativeEvent collection that is indexed by the range parameter.
Parameters:	Range — The time range for the TimeRange that you are looking for.
Returns:	oTimeRange — The TimeRange object specified by the range value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the TimeRange object does not exist in the AggregateWrapupReasonCumulativeEvent, the call will raise an error.

9.7.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a AggregateWrapupReasonCumulative requests. This QueryID is uniquely generated every time the client starts to listen for AggregateWrapupReasonCumulative events.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colAggregateWrapupReasonCumulativeEvent.QueryID
```

Property:	QueryID (of AggregateWrapupReasonCumulativeEvent)
Description:	The QueryID associated with AggregateWrapupReasonCumulativeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.8 CallbackRealtimeElement

The CallbackRealtimeElement object provides the information for a single callback.

Interfaces supported:

- ICallbackRealtimeElement (default)
- IXMLAccess

9.8.1 Properties

This section contains properties exposed through the CallbackRealtimeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.8.1.1 AttemptCount

To use this property, do the following:

```
Dim lAttemptCount as Long
lAttemptCount = oCallbackRealtimeElement.AttemptCount
```

Property:	AttemptCount
Description:	The number of callback attempts made.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.8.1.2 CallID

To use this property, do the following:

```
Dim strCallID as String
strCallID = oCallbackRealtimeElement.CallID
```

Property:	CallID
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

9.8.1.3 CustomerName

To use this property, do the following:

```
Dim strCustomerName as String
strCustomerName = oCallbackRealtimeElement.CustomerName
```

Property:	CustomerName
Description:	The name of the customer who is to be called back.
Type:	String
Potential Errors:	None
Notes:	Read-only

9.8.1.4 Description

To use this property, do the following:

```
Dim strDescription as String
strDescription = oCallbackRealtimeElement.Description
```

Property:	Description
Description:	The description associated with the callback..
Type:	String
Potential Errors:	None
Notes:	Read-only

9.8.1.5 Origin

To use this property, do the following:

```
Dim enOrigin as HiPathProCenterLibrary.enCallbackOrigins
enOrigin = oCallbackRealtimeElement.Origin
```

Property:	Origin
Description:	The origin of the callback.
Type:	enCallbackOrigins
Potential Errors:	None
Notes:	Read-only

9.8.1.6 State

To use this property, do the following:

```
Dim enState as HiPathProCenterLibrary.enCallStates
enState = oCallbackRealtimeElement.State
```

Property:	State
Description:	The current state of the callback, for example, Handled, Queued, Reserved, Pending, Scheduled..
Type:	enCallStates
Potential Errors:	None
Notes:	Read-only

9.8.1.7 TimeInState

To use this property, do the following:

```
Dim lTimeInState as Long
lTimeInState = oCallbackRealtimeElement.TimeInState
```

Property:	TimeInState
Description:	The amount of time that the callback has been in this state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.9 CallbackRealtimeEvent

The CallbackRealtimeEvent object represents a collection of one or more callback Real-time Elements. Each callback Real-time Element contains the statistics for one callback.

Interfaces supported:

- ICallbackRealtimeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.9.1 Properties

This section contains properties exposed through the CallbackRealtimeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.9.1.1 _NewEnum

```
For Each oCallbackRealtimeElement in colCallbackRealtimeEvent
   ...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oCallbackRealtimeElement as
HiPathProCenterLibrary.CallbackRealtimeElement
For Each oCallbackRealtimeElement in colCallbackRealtimeEvent
   txtTextBox.Text = txtTextBox.Text & _
   "ContactID: " & _
   oCallbackRealtimeElement.ContactID & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the ContactIDs in the CallbackRealtimeEvent.

Property:	_NewEnum
Description:	Provides access to elements of the CallbackRealtimeEvent by means of operator FOREACH.
Returns:	OCallbackRealtimeElement- for each item in the CallbackRealtimeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the CallbackRealtimeEvent collection to support the 'For Each' iterator.

9.9.1.2 Count

The Count property returns the number of CallbackRealtimeElement objects that exist in the CallbackRealtimeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colCallbackRealtimeEvent.Count
```

Property:	Count
Description:	The number of CallbackRealtimeElement objects in CallbackRealtimeEvent.
Type:	Long
Potential Errors	None
Notes:	Read-only

9.9.1.3 Item

```
oCallbackRealtimeElement = Item (ContactID as String)
```

The Item method finds a callback Real-time Element object in the CallbackRealtimeEvent collection that is indexed by the ContactID. If an item is found in the collection at the ContactID value, the Item method returns the CallbackRealtimeElement object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

ContactID as String	This is the ContactID for the CallbackRealtimeElement item you are requesting.

This method returns the following parameters:

oCallbackRealtimeElement as CallbackRealtimeElement	This is the CallbackRealtimeElement object specified by the ContactID value. This contains the statistics for the CallbackRealtimeElement.

To use this method, do the following:

```
Dim oCallbackRealtimeElement as
HiPathProCenterLibary.CallbackRealtimeElement
Set oCallbackRealtimeElement = colCallbackRealtimeEvent.Item
(ContactID)
```

This checks to see if the CallbackRealtimeElement specified by the ContactID value exists. If it does, oCallbackRealtimeElement will contain the CallbackRealtimeElement object. If it does not exist in the CallbackRealtimeEvent, this will raise an error.

Property:	Item
Description:	Finds a CallbackRealtimeElement in the CallbackRealtimeEvent collection that is indexed ContactID.
Parameters:	ContactID — The ContactID for the CallbackRealtimeElement that you are looking for.
Returns:	oCallbackRealtimeElement — The CallbackRealtimeElement object specified by the ContactID value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the CallbackRealtimeElement object does not exist in the CallbackRealtimeEvent, It will raise an error.

9.9.1.4 QueryID

The QueryID property returns the QueryID associated with the event which helps the client to correlate the received events with a ListenForEvents request. This QueryID is uniquely generated every time the client calls ListenForEvents method.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colCallbackRealtimeEvent.QueryID
```

Property:	QueryID
Description:	The QueryID associated with CallbackRealtimeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.10 ContactRealtimeElement

The ContactRealtimeElement object provides the information for a single callback.

Interfaces supported:

- IContactRealtimeElement (default)
- IXMLAccess

9.10.1 Properties

This section contains properties exposed through the ContactRealtimeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.10.1.1 CallID

To use this property, do the following:

```
Dim strContactID as String
strContactID = oContactRealtimeElement.ContactID
```

Property:	ContactID
Description:	The unique key for the contact in the system.
Type:	String
Potential Errors:	None
Notes:	Read-only

9.10.1.2 CurrentPriority

To use this property, do the following:

```
Dim sCurrentPriority as Integer
sCurrentPriority = oContactRealtimeElement.CurrentPriority
```

Property:	CurrentPriority
Description:	The current priority of the contact.
Type:	Short
Potential Errors:	None
Notes:	Read-only

9.10.1.3 CurrentState

To use this property, do the following:

```
Dim enCurrentState as HiPathProCenterLibrary.enCallStates
enCurrentState = oContactRealtimeElement.CurrentState
```

Property:	CurrentState
Description:	The current state of the contact. For example, Handled, Queued, Reserved, Pending, Deferred.
Type:	enCallStates
Potential Errors:	None
Notes:	Read-only

9.10.1.4 Description

To use this property, do the following:

```
Dim strDescription as String
strDescription = oContactRealtimeElement.Description
```

Property:	Description
Description:	The description of the contact.
Type:	String
Potential Errors:	None
Notes:	Read-only

9.10.1.5 Destination

To use this property, do the following:

```
Dim strDestination as String
strDestination = oContactRealtimeElement. Destination
```

Property:	Destination
Description:	The destination to which the contact was sent.
Type:	String
Potential Errors:	None
Notes:	Read-only

9.10.1.6 MediaType

To use this property, do the following:

```
Dim enMediaType as HiPathProCenterLibrary.enMediaTypes
enMediaType = oContactRealtimeElement.MediaType
```

Property:	MediaType
Description:	The media type of the contact.
Type:	enMediaTypes
Potential Errors:	None
Notes:	Read-only

9.10.1.7 QualifyingAgentsCount

The QualifyingAgentsCount property returns a count of number of agents that qualify to receive a given call.

To use this property, do the following:

```
Dim lCount as Long
lCount = oContactRealtimeElement.QualifyingAgentsCount
```

Property:	QualifyingAgentsCount (of ContactRealtimeElement)
Description:	Returns number of agents eligible to receive a given call.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.10.1.8 QueueKey

The QueueKey property is the key for the queue entity associated with this contact. A single entry will be provided in this record set for every contact that is being processed by the system.

To use this property, do the following:

```
Dim lQueueKey as Long
lQueueKey = oContactRealtimeElement.QueueKey
```

Property:	QueueKey
Description:	The database table unique key for the queue entity associated with this contact.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.10.1.9 Source

To use this property, do the following:

```
Dim strSource as String
strSource = oContactRealtimeElement.Source
```

Property:	Source
Description:	The source where the contact was generated.
Type:	String
Potential Errors:	None
Notes:	Read-only

9.10.1.10 TimeInState

To use this property, do the following:

```
Dim lTimeInState as Long
lTimeInState = oContactRealtimeElement.TimeInState
```

Property:	TimeInState
Description:	The amount of time that the contact has been in its current state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.10.1.11 TotalWaitTime

To use this property, do the following:

```
Dim lTotalWaitTime as Long
lTotalWaitTime = oContactRealtimeElement.TotalWaitTime
```

Property:	TotalWaitTime
Description:	The amount of time that the contact has been waiting in the system.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.11 ContactRealtimeEvent

The ContactRealtimeEvent object represents a collection of one or more Contact Real-time Elements. Each Contact Real-time Element contains the statistics for one contact.

Interfaces supported:

- IContactRealtimeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.11.1 Properties

This section contains properties exposed through the ContactRealtimeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.11.1.1 _NewEnum

```
For Each oContactRealtimeElement in colContactRealtimeEvent
   ...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator.

For example, to use this method, do the following.

```
Dim oContactRealtimeElement as
HiPathProCenterLibrary.ContactRealtimeElement
For Each oContactRealtimeElement in colContactRealtimeEvent
   txtTextBox.Text = txtTextBox.Text & _
   "ContactID: " & _
   oContactRealtimeElement.ContactID & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the ContactIDs in the ContactRealtimeEvent.

Property:	_NewEnum
Description:	Provides access to elements of the ContactRealtimeEvent by means of operator FOREACH.
Returns:	OContactRealtimeElement- for each item in the ContactRealtimeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_UnableToAllocateMemory • Error_Generic_IterationGetNewEnum
Notes:	This method allows the ContactRealtimeEvent collection to support the 'For Each' iterator.

9.11.1.2 Count

The Count property returns the number of ContactRealtimeElement objects that exist in the ContactRealtimeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colContactRealtimeEvent.Count
```

Property:	Count
Description:	The number of ContactRealtimeElement objects in ContactRealtimeEvent.

Type:	Long
Potential Errors:	None
Notes:	Read-only

9.11.1.3 Item

```
oContactRealtimeElement = Item (ContactID as String)
```

The Item method finds a Contact Real-time Element object in the ContactRealtimeEvent collection that is indexed by the ContactID. If an item is found in the collection at the ContactID value, the Item method returns the ContactRealtimeElement object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

ContactID as String	This is the ContactID for the ContactRealtimeElement item you are requesting.

This method returns the following parameters:

oContactRealtimeElement as ContactRealtimeElement	This is the ContactRealtimeElement object specified by the ContactID value. This contains the statistics for the ContactRealtimeElement.

To use this method, do the following:

```
Dim oContactRealtimeElement as
HiPathProCenterLibary.ContactRealtimeElement

Set oContactRealtimeElement = colContactRealtimeEvent.Item
(ContactID)
```

This checks to see if the ContactRealtimeElement specified by the ContactID value exists. If it does, oContactRealtimeElement will contain the ContactRealtimeElement object. If it does not exist in the ContactRealtimeEvent, this will raise an error.

Property:	Item
Description:	Finds a ContactRealtimeElement in the ContactRealtimeEvent collection that is indexed ContactID.
Parameters:	ContactID — The ContactID for the ContactRealtimeElement that you are looking for.
Returns:	oContactRealtimeElement — The ContactRealtimeElement object specified by the ContactID value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the ContactRealtimeElement object does not exist in the ContactRealtimeEvent, It will raise an error.

9.11.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a ListenForEvents request. This QueryID is uniquely generated every time the client calls ListenForEvents method.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colContactRealtimeEvent.QueryID
```

Property:	QueryID
Description:	The QueryID associated with ContactRealtimeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.12 GroupCumulativeElement

The GroupCumulativeElement object provides the cumulative statistics for a group in a given time range.

Interfaces supported:

- IGroupCumulativeElement (default)
- IXMLAccess

9.12.1 Properties

This section contains properties exposed through the GroupCumulativeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.12.1.1 ConsultOut

To use this property, do the following:

```
Dim lConsultOut as Long
lConsultOut = oGroupCumulativeElement.ConsultOut
```

Property:	ConsultOut
Description:	The number of contacts associated with the group that were consulted out by the first answering user.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.12.1.2 GroupKey

The GroupKey property is the key of the group that you want to get the statistics for.

To use this property, do the following:

```
Dim lGroupKey as Long
lGroupKey = oGroupCumulativeElement.GroupKey
```

Property:	GroupKey
Description:	The database table unique key of the group that you want to get the statistics for.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.12.1.3 Offered

To use this property, do the following:

```
Dim lOffered as Long
lOffered = oGroupCumulativeElement.Offered
```

Property:	Offered
Description:	The number of contacts that were offered to the group.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.12.1.4 Received

To use this property, do the following:

```
Dim lReceived as Long
lReceived = oGroupCumulativeElement.Received
```

Property:	Received
Description:	The number of contacts that were received by the group.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.12.1.5 ReceivedHereOfferedElsewhere

To use this property, do the following:

```
Dim lReceivedHereOfferedElsewhere as Long
lReceivedHereOfferedElsewhere =
oGroupCumulativeElement.ReceivedHereOfferedElsewhere
```

Property:	ReceivedHereOfferedElsewhere
Description:	The number of routed contacts that were received in the group, which is the primary group configured to handle these contacts, but were offered to another group.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.12.1.6 TransferOut

To use this property, do the following:

```
Dim lTransferOut as Long
lTransferOut = oGroupCumulativeElement.TransferOut
```

Property:	TransferOut
Description:	The number of contacts associated with the group that were transferred by the first answering user.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.13 GroupCumulativeEvent

The GroupCumulativeEvent object represents a collection of zero (0) or more time ranges. Each time range contains a collection of zero (0) or more Group Cumulative Elements. Each Group Cumulative Element contains the statistics for the specified queue or aggregate within the given time range.

Interfaces supported:

- IGroupCumulativeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.13.1 Properties

This section contains properties exposed through the GroupCumulativeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.13.1.1 _NewEnum

```
For Each oTimeRange in colGroupCumulativeEvent
   ...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oTimeRange as HiPathProCenterLibrary.TimeRange
For Each oTimeRange in colGroupCumulativeEvent
      txtTextBox.Text = txtTextBox.Text & _
      "Time Range: " & _
      oTimeRange.TimeRange & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the Time Ranges associated with all the TimeRange collections in GroupCumulativeEvent.

Property:	_NewEnum
Description:	Provides access to elements of the GroupCumulativeEvent collection by means of operator FOREACH.
Returns:	OTimeRange — For each item in the GroupCumulativeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_UnableToAllocateMemory • Error_Generic_IterationGetNewEnum • Error_Generic_UnknownEventType
Notes:	This property allows the GroupCumulativeEvent collection to support the 'For Each' iterator.

9.13.1.2 Count

The Count property returns the number of GroupCumulativeElement collections of GroupCumulativeElement objects that exist in the GroupCumulativeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colGroupCumulativeEvent.Count
```

Property:	Count
Description:	The number of TimeRange collections of GroupCumulativeElement objects in GroupCumulativeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.13.1.3 Item

```
oTimeRange = Item (time as Date)
```

The Item method finds a TimeRange collection of Aggregate Cumulative Elements in the GroupCumulativeEvent collection that is indexed and sorted on time interval. If an item is found in the collection at the specified time interval, the Item method returns the TimeRange collection of GroupCumulativeElement objects. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

time as Date	This is the time interval associated with the TimeRange collection item you are requesting.

This method returns the following parameters:

oTimeRange as TimeRange	This is the TimeRange collection object for TimeRange specified by the time value. This contains a collection of GroupCumulativeElement objects.

To use this method, do the following:

```
Dim oTimeRange as HiPathProCenterLibary.TimeRange
Set oTimeRange = colGroupCumulativeEvent.Item (time)
```

This checks to see if the TimeRange specified by the time value exists. If it does, oTimeRange will contain the TimeRange object. If it does not exist in the GroupCumulativeEvent collection, this will raise an error.

Property:	Item
Description:	Finds a TimeRange collection in the GroupCumulativeEvent collection that is indexed and sorted on time interval.
Parameters:	Time — The time interval associated with the TimeRange collection item you are requesting.
Returns:	oTimeRange — The TimeRange collection object for TimeRange specified by the time value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the TimeRange collection does not exist in the GroupCumulativeEvent, It will raise an error.

9.13.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a ListenForEvents request. This QueryID is uniquely generated every time the client calls ListenForEvents method.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colGroupCumulativeEvent.QueryID
```

Property:	QueryID
Description:	The QueryID associated with GroupCumulativeEvent collection.

Type:	Long
Potential Errors:	None
Notes:	Read-only

9.14 GroupRealtimeElement

The GroupRealtimeElement object provides the statistics for a single group.

Interfaces supported:

- IGroupRealtimeElement2 (default)
- IXMLAccess

9.14.1 Properties

This section contains properties exposed through the GroupRealtimeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.14.1.1 AwayUsers

To use this property, do the following:

```
Dim sAwayUsers as Integer
sAwayUsers = oGroupRealtimeElement.AwayUsers
```

Property:	AwayUsers
Description:	The number of users in the group who are in Away presence state.
Type:	Short
Potential Errors:	None
Notes:	Read-only

9.14.1.2 BusyUsers

To use this property, do the following:

```
Dim sBusyUsers as Integer
sBusyUsers = oGroupRealtimeElement.BusyUsers
```

Property:	WorkUsers
Description:	The number of users in the group who are in Busy presence state.
Type:	Short
Potential Errors:	None
Notes:	Read-only

9.14.1.3 CallsWaiting

To use this property, do the following:

```
Dim sCallsWaiting as Integer
sCallsWaiting = oGroupRealtimeElement.CallsWaiting
```

Property:	CallsWaiting
Description:	The number of contacts waiting in queue for the group.
Type:	Short
Potential Errors:	None
Notes:	Read-only

9.14.1.4 GroupKey

The GroupKey property is the key of the group that you want to get the statistics for.

To use this property, do the following:

```
Dim lGroupKey as Long
lGroupKey = oGroupRealtimeElement.GroupKey
```

Property:	GroupKey
Description:	The database table unique key of the group you want statistics for.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.14.1.5 HandlingDirectUsers

To use this property, do the following:

```
Dim sHandlingDirectUsers as Integer
sHandlingDirectUsers = oGroupRealtimeElement.HandlingDirectUsers
```

Property:	HandlingDirectUsers
Description:	The number of users in the group who are handling direct contacts.
Type:	Short
Potential Errors:	None
Notes:	Read-only

9.14.1.6 HandlingRoutedUsers

To use this property, do the following:

```
Dim sHandlingRoutedUsers as Integer
sHandlingRoutedUsers = oGroupRealtimeElement.HandlingRoutedUsers
```

Property:	HandlingRoutedUsers
Description:	The number of users in the group who are handling routed contacts.
Type:	Short
Potential Errors:	None
Notes:	Read-only

9.14.1.7 IdleUsers

To use this property, do the following:

```
Dim sIdleUsers as Integer
sIdleUsers = oGroupRealtimeElement.IdleUsers
```

Property:	IdleUsers
Description:	The number of users in the group who are in Idle presence state.
Type:	Short
Potential Errors	None
Notes:	Read-only

9.14.1.8 LoggedOnUsers

To use this property, do the following:

```
Dim sLoggedOnUsers as Integer
sLoggedOnUsers = oGroupRealtimeElement.LoggedOnUsers
```

Property:	LoggedOnUsers
Description:	The number of users in the group who are logged on.
Type:	Short
Potential Errors:	None
Notes:	Read-only

9.15 GroupRealtimeEvent

The GroupRealtimeEvent object represents a collection of one or more GroupRealtimeElements. Each GroupRealtimeElement contains statistics for one group.

Interfaces supported:

- IGroupRealtimeEvent2 (default)
- IStatisticsEvent
- IXMLAccess

9.15.1 Properties

This section contains properties exposed through the GroupRealtimeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.15.1.1 _NewEnum

```
For Each oGroupRealtimeElement in colGroupRealtimeEvent
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oGroupRealtimeElement as
HiPathProCenterLibrary.GroupRealtimeElement
For Each oGroupRealtimeElement in colGroupRealtimeEvent
   txtTextBox.Text = txtTextBox.Text & _
   "Group Key: " & _
   oGroupRealtimeElement.GroupKey & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the Group Keys in the GroupRealtimeEvent.

Property:	_NewEnum
Description:	Provides access to elements of the GroupRealtimeEvent by means of operator FOREACH.
Returns:	OGroupRealtimeElement — For each item in the GroupRealtimeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This property allows the GroupRealtimeEvent collection to support the 'For Each' iterator.

9.15.1.2 Count

The Count property returns the number of GroupRealtimeElement objects that exist in the GroupRealtimeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colGroupRealtimeEvent.Count
```

Property:	Count
Description:	The number of GroupRealtimeElement objects in GroupRealtimeEvent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.15.1.3 Item

```
oGroupRealtimeElement = Item (Key as long)
```

The Item method finds a GroupRealtimeElement object in the GroupRealtimeEvent collection that is indexed by the GroupKey. If an item is found in the collection at the key value, the Item method returns the GroupRealtimeElement object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

Key as Long	This is the GroupKey for the GroupRealtimeElement item you are requesting.

This method returns the following parameters:

oGroupRealtimeElement as GroupRealtimeElement	This is the GroupRealtimeElement object for GroupRealtimeElement specified by the Key value. This contains the statistics for the GroupRealtimeElement.

To use this method, do the following:

```
Dim oGroupRealtimeElement as
HiPathProCenterLibary.GroupRealtimeElement

Set oGroupRealtimeElement = colGroupRealtimeEvent.Item (Key)
```

This checks to see if the GroupRealtimeElement specified by the Key value exists. If it does, oGroupRealtimeElement will contain the GroupRealtimeElement object. If it does not exist in the GroupRealtimeEvent, this will raise an error.

Method:	Item
Description:	Finds a GroupRealtimeElement in the GroupRealtimeEvent collection that is indexed by the Group Key.
Parameters:	Key — The key for the GroupRealtimeElement that you are looking for.
Returns:	oGroupRealtimeElement — The GroupRealtimeElement object specified by the key value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the GroupRealtimeElement object does not exist in the GroupRealtimeEvent, It will raise an error.

9.15.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a ListenForEvents request. This QueryID is uniquely generated every time the client calls ListenForEvents method.

To use this property, do the following:

```
Dim lQueryID as Long

lQueryID = colGroupRealtimeEvent.QueryID
```

Property:	QueryID
Description:	The QueryID associated with GroupRealtimeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.16 QueueCumulativeElement

The QueueCumulativeElement object provides the cumulative statistics for a single queue in a given time range.

Interfaces supported:

- IQueueCumulativeElement (default)
- IXMLAccess

9.16.1 Properties

This section contains properties exposed through the QueueCumulativeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.16.1.1 Abandoned

To use this property, do the following:

```
Dim lAbandoned as Long
lAbandoned = oQueueCumulativeElement.Abandoned
```

Property:	Abandoned (of QueueCumulativeElement)
Description:	The number of contacts that were abandoned while waiting in the queue or while being offered to the user.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.16.1.2 AbandonedRate

To use this property, do the following:

```
Dim fAbandonedRate as Single
fAbandonedRate = oQueueCumulativeElement.AbandonedRate
```

Property:	AbandonedRate (of QueueCumulativeElement)
Description:	The percentage of contacts routed to the queue that were abandoned before being answered.
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.16.1.3 Answered

To use this property, do the following:

```
Dim lAnswered as Long
lAnswered = oQueueCumulativeElement.Answered
```

Property:	Answered (of QueueCumulativeElement)
Description:	The number of contacts routed to the queue that were answered.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.16.1.4 AverageWaitTime

To use this property, do the following:

```
Dim fAverageWaitTime as Single
fAverageWaitTime = oQueueCumulativeElement.AverageWaitTime
```

Property:	AverageWaitTime (of QueueCumulativeElement)
Description:	The average amount of time that contacts waited in the queue before being answered, abandoned, or redirected.
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.16.1.5 MaximumWaitTime

To use this property, do the following:

```
Dim lMaximumWaitTime as Long
lMaximumWaitTime = oQueueCumulativeElement.MaximumWaitTime
```

Property:	MaximumWaitTime (of QueueCumulativeElement)
Description:	The maximum amount of time that a contact waited in the queue before being answered, abandoned, or redirected.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.16.1.6 QueueKey

The QueueKey property is the key of the queue that you want to get the statistics for.

To use this property, do the following:

```
Dim lQueueKey as Long
lQueueKey = oQueueCumulativeElement.QueueKey
```

Property:	QueueKey (of QueueCumulativeElement)
Description:	The database table unique key of the queue that you want to get the statistics for.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.16.1.7 Received

To use this property, do the following:

```
Dim lReceived as Long
lReceived = oQueueCumulativeElement.Received
```

Property:	Received (of QueueCumulativeElement)
Description:	The number of contacts that were received in the queue.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.16.1.8 Redirected

To use this property, do the following:

```
Dim lRedirected as Long
lRedirected = oQueueCumulativeElement.Redirected
```

Property:	Redirected (of QueueCumulativeElement)
Description:	The number of contacts received in the queue that were redirected.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.16.1.9 ServiceLevel

To use this property, do the following:

```
Dim fServiceLevel as Single
fServiceLevel = oQueueCumulativeElement.ServiceLevel
```

Property:	ServiceLevel (of QueueCumulativeElement)
Description:	The percentage of contacts routed to the queue that were answered within the site-defined or queue-defined service level.
Type:	Float
Potential Errors:	None
Notes:	Read-Only

9.17 QueueCumulativeEvent

The QueueCumulativeEvent object represents a collection of zero (0) or more time ranges. Each time range contains a collection of zero (0) or more QueueCumulativeElements. Each QueueCumulativeElement contains the statistics for the specified queue within the given time range.

Interfaces supported:

- IQueueCumulativeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.17.1 Properties

This section contains properties exposed through the QueueCumulativeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.17.1.1 _NewEnum

```
For Each oTimeRange in colQueueCumulativeEvent
   ...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator.

For example, to use this method, do the following.

```
Dim oTimeRange as HiPathProCenterLibrary.TimeRange
For Each oTimeRange in colQueueCumulativeEvent
   txtTextBox.Text = txtTextBox.Text & _
   "Time Range: " & _
   oTimeRange.TimeRange & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the Time Ranges associated with all the TimeRange collections in QueueCumulativeEvent.

Property:	_NewEnum
Description:	Provides access to elements of the QueueCumulativeEvent collection by means of operator FOREACH.
Returns:	OTimeRange — For each item in the QueueCumulativeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownEventType
Notes:	This property allows the QueueCumulativeEvent collection to support the 'For Each' iterator.

9.17.1.2 Count

The Count property returns the number of TimeRange collections of QueueCumulativeElement objects that exist in the QueueCumulativeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colQueueCumulativeEvent.Count
```

Property:	Count
Description:	The number of TimeRange collections of QueueCumulativeElement objects in QueueCumulativeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.17.1.3 Item

```
oTimeRange = Item (time as Date)
```

The Item method finds a TimeRange collection of AggregateCumulativeElements in the QueueCumulativeEvent collection that is indexed and sorted on time interval. If an item is found in the collection at the specified time interval, the Item method returns the TimeRange collection of QueueCumulativeElement objects. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

time as Date	This is the time interval associated with the TimeRange collection item you are requesting.

This method returns the following parameters:

oTimeRange as TimeRange	This is the TimeRange collection object for TimeRange specified by the time value. This contains a collection of QueueCumulativeElement objects.

To use this method, do the following:

```
Dim oTimeRange as HiPathProCenterLibary.TimeRange
Set oTimeRange = colQueueCumulativeEvent.Item (time)
```

This checks to see if the TimeRange specified by the time value exists. If it does, oTimeRange will contain the TimeRange object. If it does not exist in the QueueCumulativeEvent collection, this will raise an error.

Property:	Item
Description:	Finds a TimeRange collection in the QueueCumulativeEvent collection that is indexed and sorted on time interval.
Parameters:	Time — The time interval associated with the TimeRange collection item you are requesting.
Returns:	oTimeRange — The TimeRange collection object for TimeRange specified by the time value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the TimeRange collection does not exist in the QueueCumulativeEvent, It will raise an error.

9.17.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a ListenForEvents request. This QueryID is uniquely generated every time the client calls ListenForEvents method.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colQueueCumulativeEvent.QueryID
```

Property:	QueryID
Description:	The QueryID associated with QueueCumulativeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.18 QueueRealtimeElement

The QueueRealtimeElement object provides the statistics for a single queue.

Interfaces supported:

- IQueueRealtimeElement (default)
- IXMLAccess

9.18.1 Properties

This section contains properties exposed through the QueueRealtimeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.18.1.1 AbandonedRate

To use this property, do the following:

```
Dim fAbandonedRate as Single
fAbandonedRate = oQueueRealtimeElement.AbandonedRate
```

Property:	AbandonedRate
Description:	The weighted percentage of the last 24 contacts routed to the queue that were abandoned..
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.18.1.2 AverageAbandonedWaitTime

To use this property, do the following:

```
Dim fAverageAbandonedWaitTime as Single
fAverageAbandonedWaitTime =
oQueueRealtimeElement.AverageAbandonedWaitTime
```

Property:	AverageAbandonedWaitTime
Description:	The average amount of time that contacts waited in the queue before being abandoned, based on the last 10 contacts that were abandoned..

Type:	Float
Potential Errors:	None
Notes:	Read-only

9.18.1.3 AverageAnsweredWaitTime

To use this property, do the following:

```
Dim fAverageAnsweredWaitTime as Single
fAverageAnsweredWaitTime =
oQueueRealtimeElement.AverageAnsweredWaitTime
```

Property:	AverageAnsweredWaitTime
Description:	The average amount of time that contacts waited in the queue before being answered, based on the last 10 contacts that were answered..
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.18.1.4 Contacts

To use this property, do the following:

```
Dim sContacts as Integer
sContacts = oQueueRealtimeElement.Contacts
```

Property:	Contacts
Description:	The number of contacts that are waiting in the queue.
Type:	Short
Potential Errors:	None
Notes:	Read-only

9.18.1.5 EstimatedAnsweredWaitTime

To use this property, do the following:

```
Dim lEstimatedAnsweredWaitTime as Long
```

```
lEstimatedAnsweredWaitTime =
oQueueRealtimeElement.EstimatedAnsweredWaitTime
```

Property:	EstimatedAnsweredWaitTime
Description:	The estimated amount of time that contacts will spend in the queue before being routed to a user.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.18.1.6 EstimatedServiceLevel

To use this property, do the following:

```
Dim fEstimatedServiceLevel as Single

fEstimatedServiceLevel =
oQueueRealtimeElement.EstimatedServiceLevel
```

Property:	EstimatedServiceLevel
Description:	The estimated service level of the queue.
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.18.1.7 OldestContactWaitTime

To use this property, do the following:

```
Dim lOldestContactWaitTime as Long

lOldestContactWaitTime =
oQueueRealtimeElement.OldestContactWaitTime
```

Property:	OldestContactWaitTime
Description:	The amount of time that the oldest contact has been waiting in the queue.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.18.1.8 OverflowedContacts

To use this property, do the following:

```
Dim sOverflowedContacts as Integer
```

```
sOverflowedContacts = oQueueRealtimeElement.OverflowedContacts
```

Property:	OverflowedContacts
Description:	The number of contacts waiting in the queue that have overflowed.
Type:	Short
Potential Errors:	None
Notes:	Read-only

9.18.1.9 QueueKey

The QueueKey property is the key of the queue that you want to get the statistics for.

To use this property, do the following:

```
Dim lQueueKey as Long
lQueueKey = oQueueRealtimeElement.QueueKey
```

Property:	QueueKey
Description:	The database table unique key of the queue that you want to get the statistics for.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.18.1.10 ServiceLevel

To use this property, do the following:

```
Dim fServiceLevel as Single
fServiceLevel = oQueueRealtimeElement.ServiceLevel
```

Property:	ServiceLevel
Description:	The current service level, calculated based on the last 24 contacts routed to the queue.
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.19 QueueRealtimeEvent

The QueueRealtimeEvent object represents a collection of one or more QueueRealtimeElements. Each QueueRealtimeElement contains the statistics for one queue.

Interfaces supported:

- IQueueRealtimeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.19.1 Properties

This section contains properties exposed through the QueueRealtimeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.19.1.1 _NewEnum

```
For Each oQueueRealtimeElement in colQueueRealtimeEvent
   ...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oQueueRealtimeElement as
HiPathProCenterLibrary.QueueRealtimeElement
For Each oQueueRealtimeElement in colQueueRealtimeEvent
   txtTextBox.Text = txtTextBox.Text & _
   "Queue Key: " & _
   oQueueRealtimeElement.QueueKey & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the queue Keys in the QueueRealtimeEvent.

Property:	_NewEnum
Description:	Provides access to elements of the QueueRealtimeEvent by means of operator FOREACH.
Returns:	OQueueRealtimeElement — For each item in the QueueRealtimeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This property allows the QueueRealtimeEvent collection to support the 'For Each' iterator.

9.19.1.2 Count

The Count property returns the number of QueueRealtimeElement objects that exist in the QueueRealtimeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colQueueRealtimeEvent.Count
```

Property:	Count
Description:	The number of QueueRealtimeElement objects in QueueRealtimeEvent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.19.1.3 Item

```
oQueueRealtimeElement = Item (Key as long)
```

The Item method finds a QueueRealtimeElement object in the QueueRealtimeEvent collection that is indexed by the QueueKey. If an item is found in the collection at the key value, the Item method returns the QueueRealtimeElement object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

Key as Long	This is the QueueKey for the QueueRealtimeElement item you are requesting.

This method returns the following parameters:

oQueueRealtime Element as QueueRealtime Element	This is the QueueRealtimeElement object for QueueRealtimeElement specified by the Key value. This contains the statistics for the QueueRealtimeElement.

To use this method, do the following:

```
Dim oQueueRealtimeElement as
HiPathProCenterLibary.QueueRealtimeElement
Set oQueueRealtimeElement = colQueueRealtimeEvent.Item (Key)
```

This checks to see if the QueueRealtimeElement specified by the Key value exists. If it does, oQueueRealtimeElement will contain the QueueRealtimeElement object. If it does not exist in the QueueRealtimeEvent, this will raise an error.

Property:	Item
Description:	Finds a QueueRealtimeElement in the QueueRealtimeEvent collection that is indexed by the queue Key.
Parameters:	Key — The key for the QueueRealtimeElement that you are looking for.
Returns:	oQueueRealtimeElement the QueueRealtimeElement object specified by the key value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the QueueRealtimeElement object does not exist in the QueueRealtimeEvent, it will raise an error.

9.19.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a ListenForEvents request. This QueryID is uniquely generated every time the client calls ListenForEvents method.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colQueueRealtimeEvent.QueryID
```

Property:	QueryID
Description:	The QueryID associated with QueueRealtimeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.20 QueueWrapupReasonCumulativeElement

The QueueWrapupReasonCumulativeElement object represents an individual record in the QueueWrapupReasonCumulativeEvent collection.

nterfaces Supported:

- IQueueWrapupReasonCumulativeElement (default)
- IXMLAccess

9.20.1 Properties

This section contains properties exposed through the QueueWrapupReasonCumulativeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.20.1.1 Count

The Count property returns the wrapup reason statistic value.

To use this property, do the following:

```
Dim lCount as Long
lCount = colQueueWrapupReasonCumulativeElement.Count
```

Property:	Count (of QueueWrapupReasonCumulativeElement)
Description:	The number of TimeRange objects QueueWrapupReasonCumulativeElement.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.20.1.2 QueueKey

The QueueKey property returns the user key value.

To use this property, do the following:

```
Dim lKey as Long
lKey = colQueueWrapupReasonCumulativeElement.QueueKey
```

Property:	QueueKey (of QueueWrapupReasonCumulativeElement)
Description:	Returns a queue key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.20.1.3 WrapupReasonKey

The WrapupReasonKey property returns the wrapup reason key value.

To use this property, do the following:

```
Dim lKey as Long
lKey = colQueueWrapupReasonCumulativeElement.WrapupReasonKey
```

Property:	WrapupReasonKey (of QueueWrapupReasonCumulativeElement)
Description:	Returns a wrapup reason key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.21 QueueWrapupReasonCumulativeEvent

The QueueWrapupReasonCumulativeEvent object represents a collection of one or more of TimeRange objects each containing one or more QueueWrapupReasonCumulativeElement objects.

Interfaces supported:

- IQueueWrapupReasonCumulativeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.21.1 Properties

This section contains properties exposed through the QueueWrapupReasonCumulativeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.21.1.1 _NewEnum

```
For Each oTimeRange in colQueueWrapupReasonCumulativeEvent
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oTimeRange as HiPathProCenterLibrary.TimeRange
For Each oTimeRange in colQueueWrapupReasonCumulativeEvent
...
Next
```

Property:	_NewEnum (of QueueWrapupReasonCumulativeEvent)
Description:	Provides access to elements of the QueueWrapupReasonCumulativeEvent by means of operator FOREACH.
Returns:	oTimeRange — For each item in the QueueWrapupReasonCumulativeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the QueueWrapupReasonCumulativeEvent collection to support the 'For Each' iterator.

9.21.1.2 Count

The Count property returns the number of TimeRange objects that exist in the QueueWrapupReasonCumulativeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colQueueWrapupReasonCumulativeEvent.Count
```

Property:	Count (of QueueWrapupReasonCumulativeEvent)
Description:	The number of TimeRange objects QueueWrapupReasonCumulativeEvent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.21.1.3 Item

```
oTimeRange = Item (range as Date)
```

The Item method finds a Time Range object in the QueueWrapupReasonCumulativeEvent collection that is indexed by the range parameter. If an item is found in the collection at the range value, the Item method returns the TimeRange object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

range as Date	This is the time range for the item you are requesting.

This method returns the following parameters:

oTimeRange as ITimeRange	This is the TimeRange object for the specified "range" value. This contains the collection of one or more of QueueWrapupReasonCumulativeElement objects.

To use this method, do the following:

```
Dim oTimeRange as HiPathProCenterLibary.TimeRange
Set oTimeRange = colQueueWrapupReasonCumulativeElement.Item
(range)
```

This checks to see if the TimeRange specified by the range value exists. If it does, oTimeRange will contain the TimeRange object. If it does not exist in the QueueWrapupReasonCumulativeEvent, this will raise an error.

Method:	Item (of QueueWrapupReasonCumulativeEvent)
Description:	Finds a TimeRange in the QueueWrapupReasonCumulativeEvent collection that is indexed by the range parameter.
Parameters:	range - the time range for the TimeRange that you are looking for.
Returns:	oTimeRange — The TimeRange object specified by the range value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the TimeRange object does not exist in the QueueWrapupReasonCumulativeEvent, the call will raise an error.

9.21.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a QueueWrapupReasonCumulative requests. This QueryID is uniquely generated every time the client starts to listen for QueueWrapupReasonCumulative events.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colQueueWrapupReasonCumulativeEvent.QueryID
```

Property:	QueryID (of QueueWrapupReasonCumulativeEvent)
Description:	The QueryID associated with QueueWrapupReasonCumulativeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.22 StatisticsEvent

The StatisticsEvent is an interface that contains a summary of the event that has occurred. All events sent back from the StatisticsManager will support the StatisticsEvent interface. This non-creatable object also provides a common method of passing all events that occur in the StatisticsManager to the applications using the SDK. This enables various pieces of information to be returned from the StatisticsManager and simple handling to be performed before determining how (and if) you want to handle this event.

Applications can get specific information about the event by querying for other interfaces that are supported by the event interface. To determine what detailed object type the event being returned, you can use the ObjectType property, and then ask for the appropriate interface.

Interfaces supported:

- IStatisticsEvent

For example:

```
Private Sub g_oStatisticsManager_EventOccurred
   (ByVal StatisticsEvent as
   HiPathProCenterLibrary.StatisticsEvent)
Select Case StatisticsEvent.ObjectType
   Case StatisticsEventObjectType_ManagerStateChanged
      ' Here since you have a ManagerStateChanged event,
      ' you query for the ManagerStateChangedEvent interface.
      Dim oManagerStateChangedEvent as
      HiPathProCenterLibrary.ManagerStateChangedEvent
      Set oManagerStateChangedEvent = StatisticsEvent
      MsgBox "Statistics Manager is now up?: "
      oManagerStateChangedEvent.State
   Default
      MsgBox "Unknown ObjectType"
End Select
End Sub
```

For the StatisticsManager, the only event that can occur at this time is the ManagerStateChangedEvent, but in the future, this may change to provide a wider variety of events.

9.22.1 Properties

This section contains properties exposed through the StatisticsEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.22.1.1 Code

The Code property returns the event code of the event that has occurred in the StatisticsManager. In most instances this will be directly related to the ObjectType that is also contained in the StatisticsEvent interface. This property determines what has happened. Since the ObjectType refers to the object that is returned, they will be related.

This property returns a value from the enStatisticsEventCodes enumerations described in Section 3.46, "enStatisticsEventCodes", on page 111. This property is read-only, and is set internally in the StatisticsManager before it fires the event back through the EventOccurred event handler. For more information, see Section 5.1.3.1, "EventOccurred", on page 166.

To use this property, do the following:

```
Dim enCode as HiPathProCenterLibrary.enStatisticsEventCodes
enCode = oStatisticsEvent.Code
```

Property:	Code
Description:	The code of what has occurred in the StatisticsManager.
Type:	enStatisticsEventCodes
Potential Errors:	None
Notes:	Read-only

9.22.1.2 EventType

The EventType property returns the event type that this event corresponds with. This event type is the event that was listened for this event to be sent. If the event is always going to be sent (Error or ManagerStateChanged event), then this property will be StatisticsEventType_NotSet.

This property returns a value from the enStatisticsEventTypes enumerations described in Section 3.48, "enStatisticsEventTypes", on page 113. This property is read-only, and is set internally in the Statistics Manager before it fires the event back through the EventOccurred event handler. For more information, see Section 5.1.3.1, "EventOccurred", on page 166.

To use this property, do the following:

```
Dim enEventType as HiPathProCenterLibrary.StatisticsEventTypes
enEventType = oStatisticsEvent.EventType
```

Property:	EventType
Description:	The event type of the StatisticsEvent event that is sent back.
Type:	enStatisticsEventTypes
Potential Errors:	None
Notes:	This determines what event group the object belongs to.

9.22.1.3 ObjectType

The ObjectType property returns the object interface type of the detailed event information contained in the StatisticsEvent. In most instances this will be directly related to the Code that is also contained in the StatisticsEvent interface. This property determines what the detailed interface is that is contained in the StatisticsEvent.

This property returns a value from the enStatisticsEventObjectTypes enumerations described in Section 3.47, "enStatisticsEventObjectTypes", on page 112. This property is read-only, and is set internally in the StatisticsManager before it fires the event back through the EventOccurred event handler. For more information, see Section 5.1.3.1, "EventOccurred", on page 166.

To use this property, do the following:

```
Dim enObjectType as
HiPathProCenterLibrary.enStatisticsEventObjectTypes
```

```
enObjectType = oStatisticsEvent.ObjectType
```

Property:	ObjectType
Description:	The detailed object type of the StatisticsManager event that is sent back.
Type:	enStatisticsEventObjectTypes
Potential Errors:	None
Notes:	This determines what object type you can query for more detailed event information.

9.22.1.4 Resource

The Resource property returns the resource that this event corresponds with. The resource is a string value that is specific for an event. This is the resource that is used when registering for an event through the ListenForEvent method.

To use this property, do the following:

```
Dim strResource as String
strResource = oStatisticsEvent.Resource
```

Property:	Resource
Description:	The resource that corresponds to the StatisticsManager event that is sent back.
Type:	String
Potential Errors:	None
Notes:	This determines the resource the event is associated with.

9.23 TimeRange

The TimeRange object represents a general standard read-only collection of any of the following (indexed by key):

- UserCumulativeElements
- GroupCumulativeElements
- QueueCumulativeElements
- AggregateCumulativeElements.

Interfaces supported:

- ITimeRange (default)

9.23.1 Properties

This section contains properties exposed through the TimeRange interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.23.1.1 _NewEnum

```
For Each oAggregateCumulativeElement in colTimeRange
...
Next
For Each oGroupCumulativeElement in colTimeRange
...
Next
For Each oQueueCumulativeElement in colTimeRange
...
Next
For Each oUserCumulativeElement in colTimeRange
...
Next
```

This property provides access to elements of the collection of cumulative elements by means of operator FOREACH. Depending on the event type that you are trying to access, it will provide access to elements of user, group, queue or aggregate cumulative elements.

The result of each of the following examples is that the property sets the Text property of the txtTextBox to contain all the Queue Keys in the TimeRange.

Example 1: oAggregateCumulativeElement

```
Dim oAggregateCumulativeElement as
HiPathProCenterLibrary.AggregateCumulativeElement
For Each oAggregateCumulativeElement in colTimeRange
txtTextBox.Text = txtTextBox.Text & _
"Queue Key: " & _
oAggregateCumulativeElement.QueueKey & vbCrLf
Next
```

Example 2: oGroupCumulativeElement

```
Dim oGroupCumulativeElement as
HiPathProCenterLibrary.GroupCumulativeElement
For Each oGroupCumulativeElement in colTimeRange
```

```
txtTextBox.Text = txtTextBox.Text & _
"Queue Key: " & _
oGroupCumulativeElement.QueueKey & vbCrLf
Next
```

Example 3: oQueueCumulativeElement

```
Dim oQueueCumulativeElement as
HiPathProCenterLibrary.QueueCumulativeElement
For Each oQueueCumulativeElement in colTimeRange
txtTextBox.Text = txtTextBox.Text & _
"Queue Key: " & _
oQueueCumulativeElement.QueueKey & vbCrLf
Next
```

Example 4: oUserCumulativeElement

```
Dim oUserCumulativeElement as
HiPathProCenterLibrary.UserCumulativeElement
For Each oUserCumulativeElement in colTimeRange
txtTextBox.Text = txtTextBox.Text & _
"Queue Key: " & _
oUserCumulativeElement.QueueKey & vbCrLf
Next
```

Property:	_NewEnum (of TimeRange)
Description:	Provides access to elements of the TimeRange by means of operator FOREACH.
Returns:	Cumulative Event — For each item in the TimeRange collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the TimeRange collection to support the 'For Each' iterator.

9.23.1.2 Count

The Count property returns the number of cumulative elements that exist in the TimeRange collection. Depending on the event type, it will provide the count to elements of user, group, queue, or aggregate cumulative elements.

To use this property, do the following:

```
Dim lCount as Long
```

```
lCount = colTimeRange.Count
```

Property:	Count (of TimeRange)
Description:	The number of Cumulative Elements in TimeRange.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.23.1.3 Item

```
oAggregateCumulativeElement = Item (Key as long)
oGroupCumulativeElement = Item (Key as long)
oQueueCumulativeElement = Item (Key as long)
oUserCumulativeElement = Item (Key as long)
```

Depending on the event type, the Item method finds a user, group, queue or aggregate cumulative element in the TimeRange collection that is indexed by the Key value. If an item is found in the collection at the Key value, the Item method returns the corresponding cumulative element. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

Key as Long	This is the Key for the Cumulative Element item you are requesting.

This method returns the following parameters:

val as LPUNKNOWN	It returns LPUNKNOWN as this interface is used by all the cumulative events. It can be resolved to the appropriate cumulative element.

To use this method, do the following, selecting from the applicable examples:

Example 1: oAggregateCumulativeElement

```
Dim oAggregateCumulativeElement as HiPathProCenterLibary.
AggregateCumulativeElement
Set oAggregateCumulativeElement = colTimeRange.Item (Key)
```

Example 2: oGroupCumulativeElement

```
Dim oGroupCumulativeElement as HiPathProCenterLibary.
GroupCumulativeElement
Set oGroupCumulativeElement = colTimeRange.Item (Key)
```

Example 3: oQueueCumulativeElement

```
Dim oQueueCumulativeElement as HiPathProCenterLibary.
QueueCumulativeElement

Set oQueueCumulativeElement = colTimeRange.Item (Key)
```

Example 4: oUserCumulativeElement

```
Dim oUserCumulativeElement as HiPathProCenterLibary.
UserCumulativeElement

Set oUserCumulativeElement = colTimeRange.Item (Key)
```

The result of each of these examples is to check if the Cumulative Element specified by the Key value exists. If it does, the object returned will contain the Cumulative Element. If it does not exist in the TimeRange, this will raise an error.

Method:	Item (of TimeRange)
Description:	Finds a Cumulative Element in the TimeRange collection that is indexed by the Key.
Parameters:	Key — The key for the Cumulative Element that you are looking for.
Returns:	The Cumulative Element specified by the key value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound • Error_Generic_UnknownEventType
Notes:	If the Cumulative Element does not exist in the TimeRange, It will raise an error.

9.23.1.4 TimeRange

The TimeRange property is the time range which is provided as a timestamp. It specifies the start of the 15–minute period to which it refers.

To use this property, do the following:

```
Dim dteTimeRange as Date

dteTimeRange = colTimeRange.TimeRange
```

Property:	TimeRange (of TimeRange)
Description:	The time range that specifies the start of the 15–minute period to which it refers.
Type:	Date
Potential Errors:	None
Notes:	Read-only

9.24 UserCumulativeElement

The UserCumulativeElement object provides the cumulative statistics for a single user in a given time range.

Interfaces supported:

- IUserCumulativeElement3 (default)
- IXMLAccess

9.24.1 Properties

This section contains properties exposed through the UserCumulativeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.24.1.1 AbandonedWhileRinging

To use this property, do the following:

```
Dim lAbandonedWhileRinging as Long

lAbandonedWhileRinging =
oUserCumulativeElement.AbandonedWhileRinging
```

Property:	AbandonedWhileRinging (of UserCumulativeElement)
Description:	The number of routed contacts that were abandoned while being offered to the user.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.2 ConsultedOut

To use this property, do the following:

```
Dim lConsultedOut as Long

lConsultedOut = oUserCumulativeElement.ConsultedOut
```

Property:	ConsultedOut (of UserCumulativeElement)
Description:	The number of routed contacts that were consulted out by the user. If the user consulted out more than once during a single contact, the contact is counted only once.

Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.3 Disconnected

To use this property, do the following:

```
Dim l Disconnected as Long
l Disconnected = oUserCumulativeElement. Disconnected
```

Property:	Disconnected (of UserCumulativeElement)
Description:	The number of routed contacts that were handled by the user and completed without being transferred or requeued.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.4 Handled

To use this property, do the following:

```
Dim lHandled as Long
lHandled = oUserCumulativeElement.Handled
```

Property:	Handled (of UserCumulativeElement)
Description:	The number of routed and direct contacts that were handled by the user.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.5 MaximumRoutedHandlingTime

To use this property, do the following:

```
Dim lMaximumRoutedHandlingTime as Long
```

```
lMaximumRoutedHandlingTime =
oUserCumulativeElement.MaximumRoutedHandlingTime
```

Property:	MaximumRoutedHandlingTime (of UserCumulativeElement)
Description:	The maximum amount of time spent handling a routed contact.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.6 Offered

To use this property, do the following:

```
Dim lOffered as Long
lOffered = oUserCumulativeElement.Offered
```

Property:	Offered (of UserCumulativeElement)
Description:	The number of routed and direct contacts that were offered to the user.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.7 Requeued

To use this property, do the following:

```
Dim lRequeued as Long
lRequeued = oUserCumulativeElement.Requeued
```

Property:	Requeued (of UserCumulativeElement)
Description:	The number of routed contacts that the user requeued.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.8 RoutedHeld

To use this property, do the following:

```
Dim lRoutedHeld as Long
```

```
lRoutedHeld = oUserCumulativeElement.RoutedHeld
```

Property:	RoutedHeld (of UserCumulativeElement)
Description:	The number of routed calls and callbacks that the user placed on hold.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.9 TotalAwayTime

To use this property, do the following:

```
Dim lTotalAwayTime as Long
lTotalAwayTime = oUserCumulativeElement.TotalAwayTime
```

Property:	TotalAwayTime (of UserCumulativeElement)
Description:	The amount of time that the user spent in Away presence state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.10 TotalBusyTime

To use this property, do the following:

```
Dim lTotalBusyTime as Long
lTotalBusyTime = oUserCumulativeElement.TotalBusyTime
```

Property:	TotalBusyTime (of UserCumulativeElement)
Description:	The amount of time that the user spent in Busy presence state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.11 TotalHandledTime

To use this property, do the following:

```
Dim lTotalHandledTime as Long
lTotalHandledTime = oUserCumulativeElement.TotalHandledTime
```

Property:	TotalHandledTime (of UserCumulativeElement)
Description:	The amount of time that the user spent handling routed and direct contacts.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.12 TotalHoldTime

To use this property, do the following:

```
Dim lTotalHoldTime as Long
lTotalHoldTime = oUserCumulativeElement.TotalHoldTime
```

Property:	TotalHoldTime (of UserCumulativeElement)
Description:	The amount of time that the user spent in Other handling state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.13 TotalIdleTime

To use this property, do the following:

```
Dim lTotalIdleTime as Long
lTotalIdleTime = oUserCumulativeElement.TotalIdleTime
```

Property:	TotalIdleTime (of UserCumulativeElement)
Description:	The amount of time that the user spent in Idle presence state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.14 TotalLoggedOnTime

To use this property, do the following:

```
Dim lTotalLoggedOnTime as Long
lTotalLoggedOnTime = oUserCumulativeElement.TotalLoggedOnTime
```

Property:	TotalLoggedOnTime (of UserCumulativeElement)
Description:	The amount of time that the user spent logged on.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.15 TotalPendingTime

To use this property, do the following:

```
Dim lTotalPendingTime as Long
lTotalPendingTime = oUserCumulativeElement.TotalPendingTime
```

Property:	TotalPendingTime (of UserCumulativeElement)
Description:	The amount of time the user spent in Pending handling state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.16 TotalPostProcessingTime

To use this property, do the following:

```
Dim lTotalPostProcessingTime as Long
lTotalPostProcessingTime =
oUserCumulativeElement.TotalPostProcessingTime
```

Property:	TotalPostProcessingTime (of UserCumulativeElement)
Description:	The amount of time that the user spent in Post-processing handling state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.17 TotalRingTime

To use this property, do the following:

```
Dim lTotalRingTime as Long
lTotalRingTime = oUserCumulativeElement.TotalRingTime
```

Property:	TotalRingTime (of UserCumulativeElement)
Description:	The total amount of time that the user's extension was ringing for routed or direct contacts (incoming) or listening to the ringing of a dialed extension (outgoing).
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.18 TotalTalkTime

To use this property, do the following:

```
Dim lTotalTalkTime as Long
lTime = oUserCumulativeElement.TotalTalkTime
```

Property:	Time (of UserCumulativeElement)
Description:	The amount of time that the user spent in Talking handling state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.19 TotalTimeOther

To use this property, do the following:

```
Dim lTotalTimeOther as Long
lTotalTimeOther = oUserCumulativeElement.TotalTimeOther
```

Property:	TotalTimeOther (of UserCumulativeElement)
Description:	The amount of time that the user spent in Other handling state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.20 TransferredOut

To use this property, do the following:

```
Dim lTransferredOut as Long
lTransferredOut = oUserCumulativeElement.TransferredOut
```

Property:	TransferredOut (of UserCumulativeElement)
Description:	The number of routed contacts that the user transferred directly without requeuing.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.21 Undelivered

To use this property, do the following:

```
Dim lUndelivered as Long
lUndelivered = oUserCumulativeElement.Undelivered
```

Property:	Undelivered (of UserCumulativeElement)
Description:	The number of routed calls that were assigned to the user but were not delivered to the user's extension.
Type:	Long
Potential Errors:	None
Notes:	Read-Only

9.24.1.22 Unhandled

To use this property, do the following:

```
Dim lUnhandled as Long
lUnhandled = oUserCumulativeElement.Unhandled
```

Property:	Unhandled (of UserCumulativeElement)
Description:	The number of routed contacts that the user transferred directly without requeuing.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.23 UserKey

The UserKey property returns the key for the user in the database.

To use this property, do the following:

```
Dim lUserKey as Long
lUserKey = oUserCumulativeElement.UserKey
```

Property:	UserKey (of UserCumulativeElement)
Description:	The database table unique key for the User object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.24.1.24 Utilization

To use this property, do the following:

```
Dim fUtilization as Single
fUtilization = oUserCumulativeElement.Utilization
```

Property:	Utilization (of UserCumulativeElement)
Description:	The percentage of the user's logged-on time that was spent handling contacts.
Type:	Float
Potential Errors:	None
Notes:	Read-only

9.25 UserCumulativeEvent

The UserCumulativeEvent object represents a collection of zero (0) or more time ranges. Each time range contains a collection of zero (0) or more UserCumulativeElement objects. Each UserCumulativeElement contains the statistics for the specified queue or aggregate within the given time range.

Interfaces supported:

- IUserCumulativeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.25.1 Properties

This section contains properties exposed through the UserCumulativeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.25.1.1 _NewEnum

```
For Each oTimeRange in colUserCumulativeEvent
   ...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oTimeRange as HiPathProCenterLibrary.TimeRange
For Each oTimeRange in colUserCumulativeEvent
   txtTextBox.Text = txtTextBox.Text & _
   "Time Range: " & _
   oTimeRange.TimeRange & vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain all the Time Ranges associated with all the TimeRange collections in UserCumulativeEvent.

Property:	_NewEnum
Description:	Provides access to elements of the UserCumulativeEvent collection by means of operator FOREACH.
Returns:	OTimeRange — For each item in the UserCumulativeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory • Error_Generic_UnknownEventType
Notes:	This property allows the UserCumulativeEvent collection to support the 'For Each' iterator.

9.25.1.2 Count

The Count property returns the number of TimeRange collections of UserCumulativeElement objects that exist in the UserCumulativeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colUserCumulativeEvent.Count
```

Property:	Count
Description:	The number of TimeRange collections of UserCumulativeElement objects in UserCumulativeEvent collection.

Type:	Long
Potential Errors:	None
Notes:	Read-only

9.25.1.3 Item

```
oTimeRange = Item (time as Date)
```

The Item method finds a TimeRange collection of Aggregate Cumulative Elements in the UserCumulativeEvent collection that is indexed and sorted on time interval. If an item is found in the collection at the specified time interval, the Item method returns the TimeRange collection of UserCumulativeElement objects. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

time as Date	This is the time interval associated with the TimeRange collection item you are requesting.

This method returns the following parameters:

oTimeRange as TimeRange	This is the TimeRange collection object for TimeRange specified by the time value. This contains a collection of UserCumulativeElement objects.

To use this method, do the following:

```
Dim oTimeRange as HiPathProCenterLibary.TimeRange
Set oTimeRange = colUserCumulativeEvent.Item (time)
```

This checks to see if the TimeRange specified by the time value exists. If it does, oTimeRange will contain the TimeRange object. If it does not exist in the UserCumulativeEvent collection, this will raise an error.

Property:	Item
Description:	Finds a TimeRange collection in the UserCumulativeEvent collection that is indexed and sorted on time interval.
Parameters:	Time — The interval associated with the TimeRange collection item you are requesting.
Returns:	oTimeRange — The TimeRange collection object for TimeRange specified by the time value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the TimeRange collection does not exist in the UserCumulativeEvent, It will raise an error.

9.25.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a ListenForEvents request. This QueryID is uniquely generated every time the client calls the ListenForEvents method.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colUserCumulativeEvent.QueryID
```

Property:	QueryID
Description:	The QueryID associated with UserCumulativeEvent collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.26 UserRealtimeElement

The UserRealtimeElement object provides the statistics for a single user.

Interfaces supported:

- IUserRealtimeElement3 (default)
- IXMLAccess

9.26.1 Properties

This section contains properties exposed through the UserRealtimeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.26.1.1 Extension

To use this property, do the following:

```
Dim strExtension as String
strExtension = oUserRealtimeElement.Extension
```

Property:	Extension (of UserRealtimeElement)
Description:	The telephone extension that the user is logged on to..
Type:	String
Potential Errors:	None
Notes:	Read-only

9.26.1.2 HandlingStates

The HandlingStates property queries the UserRTHandlingStates interface and returns a read-only collection of HandlingState objects that contains all the handling states defined in the system. This property does not accept any parameters.

To use this property, do the following:

```
Dim oHandlingStates as
HiPathProCenterLibrary.UserRTHandlingStates
set oHandlingStates = oUserRealtimeElement.HandlingStates
```

Property:	HandlingStates (of UserRealtimeElement)
Description:	The state of the user while handling a contact..
Type:	UserRTHandlingStates
Potential Errors:	None
Notes:	Read-only

9.26.1.3 MaximumUnusedCapacity

To use this property, do the following:

```
Dim lMaximumUnusedCapacity as Long

lMaximumUnusedCapacity =
oUserRealtimeElement.MaximumUnusedCapacity
```

Property:	MaximumUnusedCapacity
Description:	The maximum number of contacts (including direct contacts) that the user can accept before reaching their configured capacity.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.26.1.4 MinimumUnusedCapacity

To use this property, do the following:

```
Dim lMinimumUnusedCapacity as Long

lMinimumUnusedCapacity =
oUserRealtimeElement.MinimumUnusedCapacity
```

Property:	MinimumUnusedCapacity
Description:	The minimum number of contacts (including direct contacts) that the user can accept before reaching their configured capacity.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.26.1.5 PresenceState

To use this property, do the following:

```
Dim enPresenceState as HiPathProCenterLibrary.enPresenceStates

enPresenceState = oUserRealtimeElement.PresenceState
```

Property:	PresenceState
Description:	The presence state of the user.
Type:	enPresenceStates
Potential Errors:	None
Notes:	Read-only

9.26.1.6 RoutingState

To use this property, do the following:

```
Dim enRoutingStates as HiPathProCenterLibrary.enRoutingStates
enRoutingState = oUserRealtimeElement.RoutingState
```

Property:	RoutingState
Description:	The routing state of the user.
Type:	enRoutingStates
Potential Errors:	None
Notes:	Read-only

9.26.1.7 RoutingStateReasonKey

To use this property, do the following:

```
Dim lRoutingStateReasonKey as Long
lRoutingStateReasonKey =
oUserRealtimeElement.RoutingStateReasonKey
```

Property:	RoutingStateReasonKey
Description:	The Unavailable reason or Work reason that the user selected.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.26.1.8 TimeInPresenceState

To use this property, do the following:

```
Dim lTimeInPresenceState as Long
lTimeInPresenceState = oUserRealtimeElement.TimeInPresenceState
```

Property:	TimeInPresenceState
Description:	The amount of time that the user has been in the current presence state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.26.1.9 TimeInRoutingState

To use this property, do the following:

```
Dim lTimeInRoutingState as Long
lTimeInRoutingState = oUserRealtimeElement.TimeInRoutingState
```

Property:	TimeInRoutingState
Description:	.The amount of time that the user has been in the current routing state
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.26.1.10 UserKey

The UserKey property returns the key for the user in the database.

To use this property, do the following:

```
Dim lUserKey as Long
lUserKey = oUserRealtimeElement.UserKey
```

Property:	UserKey
Description:	The database unique key of the User object.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.27 UserRealtimeEvent

The UserRealtimeEvent object represents a collection of one or more UserRealtimeElement. Each UserRealtimeElement contains the statistics for one user.

Interfaces supported:

- IUserRealtimeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.27.1 Properties

This section contains properties exposed through the UserRealtimeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.27.1.1 _NewEnum

```
For Each oUserRealtimeElement in colUserRealtimeEvent
   ...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oUserRealtimeElement as
HiPathProCenterLibrary.UserRealtimeElement
   For Each oUserRealtimeElement in colUserRealtimeEvent
      txtTextBox.Text = txtTextBox.Text & _
      "Aggregate Key: " & _
      oUserRealtimeElement.AggregateKey & vbCrLf
   Next
```

This sets the Text property of the txtTextBox to contain all the Aggregate Keys in the UserRealtimeEvent.

Property:	_NewEnum
Description:	Provides access to elements of the UserRealtimeEvent by means of operator FOREACH.
Returns:	oUserRealtimeElement — For each item in the UserRealtimeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the UserRealtimeEvent collection to support the 'For Each' iterator.

9.27.1.2 Count

The Count property returns the number of UserRealtimeElement objects that exist in the UserRealtimeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colUserRealtimeEvent.Count
```

Property:	Count
Description:	The number of UserRealtimeElement objects in UserRealtimeEvent.
Type:	Long
Potential Errors	None
Notes:	Read-only

9.27.1.3 Item

```
oUserRealtimeElement = Item (Key as long)
```

The Item method finds a UserRealtimeElement object in the UserRealtimeEvent collection that is indexed by the UserKey. If an item is found in the collection at the key value, the Item method returns the UserRealtimeElement object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

Key as Long	This is the UserKey for the UserRealtimeElement item you are requesting.

This method returns the following parameters:

UserRealtime Element as UserRealtime Element	This is the UserRealtimeElement object for UserRealtimeElement specified by the Key value. This contains the statistics for the UserRealtimeElement.

To use this method, do the following:

```
Dim oUserRealtimeElement as
HiPathProCenterLibary.UserRealtimeElement
Set oUserRealtimeElement = colUserRealtimeEvent.Item (Key)
```

This checks to see if the UserRealtimeElement specified by the Key value exists. If it does, UserRealtimeElement will contain the UserRealtimeElement object. If it does not exist in the UserRealtimeEvent, this will raise an error.

Method:	Item
Description:	Finds a UserRealtimeElement in the UserRealtimeEvent collection that is indexed by the Aggregate Key.
Parameters:	Key — The key for the UserRealtimeElement that you are looking for.
Returns:	oUserRealtimeElement- the UserRealtimeElement object specified by the key value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the UserRealtimeElement object does not exist in the UserRealtimeEvent, It will raise an error.

9.27.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a ListenForEvents request. This QueryID is uniquely generated every time the client calls ListenForEvents method.

To use this property, do the following:

```
Dim lQueryID as Long
lQueryID = colUserRealtimeEvent.QueryID
```

Property:	QueryID
Description:	The QueryID associated with UserRealtimeEvent collection.
Type:	Long
Potential Error:	None
Notes:	Read-only

9.28 UserRTHandlingState

The UseRTHandlingState object stores a collection of user real-time handling states.

Interfaces supported:

- IUserRTHandlingState2 (default)
- IXMLAccess

9.28.1 Properties

This section contains properties exposed through the UserRTHandlingState interface. For more information on the potential errors listed in the following sections, see Section 3.18, "enErrors", on page 59.

9.28.1.1 AssociatedQueueKey

To use this property, do the following:

```
Dim lAssociatedQueueKey as Long
lAssociatedQueueKey = oUserRTHandlingState.AssociatedQueueKey
```

Property:	AssociatedQueueKey (of UserRTHandlingState)
Description:	The name of the queue that is associated with the contact that the user is handling.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.28.1.2 CallID

To use this property, do the following:

```
Dim strCallID as String
strCallID = oUserRTHandlingState.CallID
```

Property:	CallID (of UserRTHandlingState)
Description:	The unique key for the contact in the system..
Type:	String
Potential Errors:	None
Notes:	Read-only

9.28.1.3 ContactDescription

To use this property, do the following:

```
Dim strContactDescription as String
strContactDescription =
oUserUserRTHandlingState.ContactDescription
```

Property:	ContactDescription (of UserRTHandlingState)
Description:	The description of the contact that the user is handling.
Type:	String
Potential Errors:	None
Notes:	Read-only

9.28.1.4 ContactType

To use this property, do the following:

```
Dim enContactType as HiPathProCenterLibrary.enContactTypes
enContactType = oUserRTHandlingState.ContactType
```

Property:	ContactType (of UserRTHandlingState)
Description:	The type of contact that the user is handling.
Type:	enContactTypes
Potential Errors:	None
Notes:	Read-only

9.28.1.5 HandlingState

To use this property, do the following:

```
Dim enHandlingStates as HiPathProCenterLibrary.enHandlingStates
enHandlingStates = oUserRTHandlingState.HandlingState
```

Property:	HandlingState (of UserRTHandlingState)
Description:	The state of a user while handling a contact.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.28.1.6 HandlingStateReason

To use this property, do the following:

```
Dim lHandlingStateReason as Long
lHandlingStateReason = oUserRTHandlingState.HandlingStateReason
```

Property:	HandlingStateReason (of UserRTHandlingState)
Description:	The user's current handling state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.28.1.7 IsPrimary

The IsPrimary property specifies whether the contact being handled is the primary contact.

To use this property, do the following:

```
Dim bIsPrimary as Boolean
bIsPrimary = oUserRTHandlingState.IsPrimary
```

Property:	IsPrimary (of UserRTHandlingState)
Description:	The user's current handling state.
Type:	Boolean
Potential Errors:	None
Notes:	Read-only

9.28.1.8 SequenceNumber

An internal number used to enable contacts to be uniquely identified.

To use this property, do the following:

```
Dim lSequenceNumber as Long
lSequenceNumber = oUserHandlingState.SequenceNumber
```

Property:	SequenceNumber (of UserHandlingState)
Description:	
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.28.1.9 TimeInHandlingState

To use this property, do the following:

```
Dim lTimeInHandlingState as Long
lTimeInHandlingState = oUserRTHandlingState.TimeInHandlingState
```

Property:	TimeInHandlingState (of UserRTHandlingState)
Description:	The amount of time that the user has been in the current handling state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.28.1.10 TotalContactActiveTime

To use this property, do the following:

```
Dim lTotalContactActiveTime as Long
lTotalContactActiveTime =
oUserRTHandlingState.TotalContactActiveTime
```

Property:	TotalContactActiveTime (of UserRTHandlingState)
Description:	The amount of time that a contact has been active.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.28.1.11 TotalContactHandlingTime

To use this property, do the following:

```
Dim lTotalContactHandlingTime as Long
lTotalContactHandlingTime =
oUserRTHandlingState.TotalContactHandlingTime
```

Property:	TotalContactHandlingTime (of UserRTHandlingState)
Description:	The amount of time that the user has been in the current routing state.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.29 UserRTHandlingStates

The UserRTHandlingStates object is a collection of the UserRTHandlingState objects.

Interfaces supported:

- IUserRTHandlingStates2
- IXMLAccess

9.29.1 Properties

This section contains properties exposed through the UserRTHandlingStates interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.29.1.1 _NewEnum

```
For Each oUserRTHandlingState in colUserRTHandlingStates
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this method, do the following.

```
Dim oUserRTHandlingState as
HiPathProCenterLibrary.UserRTHandlingState
For Each oUserRTHandlingState in m_colUserRTHandlingStates
   txtTextBox.Text = txtTextBox.Text & _
      " CallID: " & _
      oUserRTHandlingState.CallID & _
      " HandlingState: " & _
      oUserRTHandlingState.HandlingState &
      vbCrLf
Next
```

This sets the Text property of the txtTextBox to contain a collection of handling states for the user while they are handling a contact.

Property:	_NewEnum (of UserRTHandlingStates)
Description:	Allows for the 'For Each' operator on the UserRTHandlingStates collection.
Returns:	oUserRTHandlingState — For each item in the UserRTHandlingStates collection until all items have been returned.
Potential Errors:	None
Notes:	This method allows the UserRTHandlingStates collection to support the 'For Each' iterator.

9.29.1.2 Count

The Count property returns the number of UserRTHandlingState objects that exist in the UserRTHandlingStates collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colUserRTHandlingStates.Count
```

Property:	Count (of UserRTHandlingStates)
Description:	The number of UserRTHandlingState objects that are in the UserRTHandlingStates collection.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.29.1.3 Item

```
oUserRTHandlingState = Item (ContactID as String, SequenceNumber as Long)
```

The Item method finds an UserRTHandlingState object in the UserRTHandlingStates collection that is indexed by the ContactID and SequenceNumber. If an item is found in the collection, the Item method returns the UserRTHandlingState object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameter:

ContactID as String	This is the ContactID for the period you are requesting.
SequenceNumber as Long	This is the SequenceNumber for the handling state you are requesting.

This method returns the following parameter:

oUserRTHandlingState as UserRTHandlingState	This is the UserRTHandlingState object that is indexed by the ContactID and SequenceNumber.

To use this method, do the following:

```
Dim oUserRTHandlingState as
HiPathProCenterLibary.UserRTHandlingState

Set oUserRTHandlingState = colUserRTHandlingStates.Item
(strContactID, lSequenceNumber)
```

This checks to see if the UserRTHandlingState exists. If it does, oUserRTHandlingState will contain the UserRTHandlingState object. If it does not exist in the UserRTHandlingStates collection, this will raise an error.

Method:	Item (of UserRTHandlingStates)
Description:	Finds an UserRTHandlingState object in the UserRTHandlingStates collection that is indexed by the ContactID and SequenceNumber.
Parameters:	ContactID — The ContactID for the UserRTHandlingState that you are requesting. SequenceNumber — The SequenceNumber for the UserRTHandlingState that you are requesting.
Returns:	oUserRTHandlingState — The user status for the user specified.
Potential Errors:	None
Notes:	If the UserRTHandlingState object for the user does not exist in our collection, you raise an error.

9.30 UserWrapupReasonCumulativeElement

The UserWrapupReasonCumulativeElement object represents an individual record in the UserWrapupReasonCumulativeEvent collection.

Interfaces supported:

- IUserWrapupReasonCumulativeElement (default)
- IXMLAccess

9.30.1 Properties

This section contains properties exposed through the UserWrapupReasonCumulativeElement interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.30.1.1 Count

The Count property returns the wrapup reason statistic value.

To use this property, do the following:

```
Dim lCount as Long
lCount = colUserWrapupReasonCumulativeElement.Count
```

Property:	Count (of UserWrapupReasonCumulativeElement)
Description:	The number of TimeRange objects UserWrapupReasonCumulativeElement.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.30.1.2 UserKey

The UserKey property returns the user key value.

To use this property, do the following:

```
Dim lKey as Long
lKey = colUserWrapupReasonCumulativeElement.UserKey
```

Property:	UserKey (of UserWrapupReasonCumulativeElement)
Description:	Returns a user key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.30.1.3 WrapupReasonKey

The WrapupReasonKey property returns the wrapup reason key value.

To use this property, do the following:

```
Dim lKey as Long
lKey = colUserWrapupReasonCumulativeElement.WrapupReasonKey
```

Property:	WrapupReasonKey (of UserWrapupReasonCumulativeElement)
Description:	Returns a wrapup reason key.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.31 UserWrapupReasonCumulativeEvent

The UserWrapupReasonCumulativeEvent object represents a collection of one or more of TimeRange objects each containing one or more UserWrapupReasonCumulativeElement objects.

Interfaces supported:

- IUserWrapupReasonCumulativeEvent (default)
- IStatisticsEvent
- IXMLAccess

9.31.1 Properties

This section contains properties exposed through the UserWrapupReasonCumulativeEvent interface. For more information on the potential errors listed in the following sections, see Section 3.18, “enErrors”, on page 59.

9.31.1.1 _NewEnum

For Each oTimeRange in colUserWrapupReasonCumulativeEvent

```
...
Next
```

The _NewEnum property is not seen directly in the SDK. Instead, it is implicitly exposed through the For Each iterator that Visual Basic exposes. Since C++ handles collections differently, _NewEnum may be seen by Visual C++ developers. Please refer to the sample code for examples on how to iterate through a collection in C++.

This property allows for the iteration of collections in Visual Basic through the 'For Each' iterator. For example, to make use of this property, do the following.

```
Dim oTimeRange as HiPathProCenterLibrary.TimeRange
For Each oTimeRange in colUserWrapupReasonCumulativeEvent
…
```

```
Next
```

Property:	_NewEnum (of UserWrapupReasonCumulativeEvent)
Description:	Provides access to elements of the UserWrapupReasonCumulativeEvent by means of operator FOREACH.
Returns:	oTimeRange — For each item in the UserWrapupReasonCumulativeEvent collection until all items have been returned.
Potential Errors:	• Error_Generic_IterationGetNewEnum • Error_Generic_UnableToAllocateMemory
Notes:	This method allows the UserWrapupReasonCumulativeEvent collection to support the 'For Each' iterator.

9.31.1.2 Count

The Count property returns the number of TimeRange objects that exist in the UserWrapupReasonCumulativeEvent collection.

To use this property, do the following:

```
Dim lCount as Long
lCount = colUserWrapupReasonCumulativeEvent.Count
```

Property:	Count (of UserWrapupReasonCumulativeEvent)
Description:	The number of TimeRange objects UserWrapupReasonCumulativeEvent.
Type:	Long
Potential Errors:	None
Notes:	Read-only

9.31.1.3 Item

```
oTimeRange = Item (range as Date)
```

The Item method finds a Time Range object in the UserWrapupReasonCumulativeEvent collection that is indexed by the range parameter. If an item is found in the collection at the range value, the Item method returns the TimeRange object. If the item does not exist in the collection, then the Item method will raise an error.

This method accepts the following parameters:

range as Date	This is the time range for the item you are requesting.

This method returns the following parameters:

oTimeRange as ITimeRange	This is the TimeRange object for the specified "range" value. This contains the collection of one or more of UserWrapupReasonCumulativeElement objects.

To use this method, do the following:

```
Dim oTimeRange as HiPathProCenterLibary.TimeRange

Set oTimeRange = colUserWrapupReasonCumulativeElement.Item
(range)
```

This checks to see if the TimeRange specified by the range value exists. If it does, oTimeRange will contain the TimeRange object. If it does not exist in the UserWrapupReasonCumulativeEvent, this will raise an error.

Property:	Item (of UserWrapupReasonCumulativeEvent)
Description:	Finds a TimeRange in the UserWrapupReasonCumulativeEvent collection that is indexed by the range parameter.
Parameters:	range — The time range for the TimeRange that you are looking for.
Returns:	oTimeRange — The TimeRange object specified by the range value.
Potential Errors:	• Error_Generic_CreatingObject • Error_Generic_ItemNotFound
Notes:	If the TimeRange object does not exist in the UserWrapupReasonCumulativeEvent, the call will raise an error.

9.31.1.4 QueryID

The QueryID property returns QueryID associated with the event which helps the client to correlate the received events with a UserWrapupReasonCumulative requests. This QueryID is uniquely generated every time the client starts to listen for UserWrapupReasonCumulative events.

To use this property, do the following:

```
Dim lQueryID as Long

lQueryID = colUserWrapupReasonCumulativeEvent.QueryID
```

Property:	QueryID (of UserWrapupReasonCumulativeEvent)
Description:	The QueryID associated with UserWrapupReasonCumulativeEvent collection.

Type:	Long
Potential Errors:	None
Notes:	Read-only

Index

A

Abandoned
- AggregateCumulativeElement 575
- QueueCumulativeElement 619

AbandonedRate
- AggregateCumulativeElement 575
- AggregateRealtimeElement 581
- QueueCumulativeElement 619
- QueueRealtimeElement 626

AbandonedWhileRinging 647
Accept
- CallbackCall 308
- EmailCall 389
- WebCollaborationCall 525

ACDGroupNumber 242
Add
- ContactData 119, 141
- CriterionList 345
- EmailAddresses 367
- EmailAttachments 374
- EnumList 125
- KeyList 129

AddCallbackSchedule 309
Administration Manager 175
AdministrationEvent 191
AdministrationManager object 175
Agent 279
AgentExtension 542
AgentID 294
AgentKey
- AgentStatusEvent 295
- AssignedEvent 543
- CallbackCall 302
- EmailCall 376
- Party 446
- UnassignedEvent 570
- VoiceCall 503
- WebCollaborationCall 519

AgentStatusEvent 294
Aggregate 195
AggregateCumulativeElement 574
AggregateCumulativeEvent 578
AggregateKey
- AggregateCumulativeElement 575
- AggregateRealtimeElement 582
- AggregateWrapupReasonCumulativeElement 588

AggregateRealtimeElement 581
AggregateRealtimeEvent 585
Aggregates 197
AggregateWrapupReasonCumulativeElement 588
AggregateWrapupReasonCumulativeEvent 589
ANI 503
Answer
- ServicesEnabled 465
- ServicesEnabledEvent 473
- VoiceCall 509

Answered
- AggregateCumulativeEvent 576
- QueueCumulativeEvent 620

AssignAgentKey 552
AssignedEvent 542
AssociatedQueueKey 667
AsXML 145
Attachments 376
AttemptCount 593
AutoAcknowledgedEvent 298
Available
- ServicesEnabled 465
- ServicesEnabledEvent 473

AverageAbandonedWaitTime
- AggregateRealtimeElement 582
- QueueRealtimeElement 626

AverageAnsweredWaitTime
- AggregateRealtimeElement 583
- QueueRealtimeElement 627

AverageWaitTime
- AggregateCumulativeElement 576
- QueueCumulativeElement 620

AwayUsers 613

B

BodyUTF8 377
BusinessUnitName 148
BusyUsers 613

C

CallbackCall 301
CallbackRealtimeElement 593
CallbackRealtimeEvent 595
CallbackReason 318
CallbackReasons 320
CallbackSchedule 323
CallbackSchedules 302, 324

CallID
AssignedEvent 543
AutoAcknowledgedEvent 299
CallbackCall 303
CallbackRealtimeElement 593
CallInformationUpdatedEvent 328
CallMeEvent 331
ConferencedEvent 334
ContactRealtimeElement 599
ContentAddedEvent 338
DeliveredEvent 348
DequeuedEvent 544
DisconnectedEvent 354
DivertedEvent 359
EmailCall 377
EmailSummary 407
EnqueuedEvent 548
EstablishedEvent 411
FailedEvent 416
HeldEvent 421
NetworkReachedEvent 432
OffHookEvent 435
OriginatedEvent 439
QueuedEvent 447
RetrievedEvent 452
RoutingCall 552
RoutingInformationUpdatedEvent 564
ScheduleCallback 456
TimedOutEvent 568
TransferredEvent 481
UnassignedEvent 570
UserEventHandlingState 486
UserHandlingState 493
UserRTHandlingState 667
VoiceCall 504
WebCollaborationCall 520
CallInformationUpdatedEvent 327
CallMeEvent 330
CallsWaiting 614
CancelMandatoryWrappingUp 496
Code
AdministrationEvent 192
HiPathProCenterEvent 168
MediaEvent 428
RoutingEvent 561
StatisticsEvent 640
Complete 310
Conference
ServicesEnabled 466
ServicesEnabledEvent 474
VoiceCall 510
ConferencedEvent 334
ConferenceRequest 526
connectivity issues, handling 40
Consult
ServicesEnabled 466
ServicesEnabledEvent 474
VoiceCall 511
ConsultedOut 647
ConsultOut 607
ContactData 117
CallbackCall 303
CallInformationUpdatedEvent 328
EmailCall 378
RoutingCall 553
RoutingInformationUpdatedEvent 566
VoiceCall 504
WebCollaborationCall 520
ContactDescription 668
ContactMediaType 493
ContactRealtimeElement 599
ContactRealtimeEvent 603
Contacts
AggregateRealtimeElement 583
QueueRealtimeElement 627
ContactType 668
VoiceCall 505
Content
EmailTemplate 212
WebCollaborationTemplate 250
ContentAddedEvent 337
ConversationID
AutoAcknowledgedEvent 299
CallInformationUpdatedEvent 329
DeliveredEvent 349
DisconnectedEvent 354
DivertedEvent 359
EmailCall 378
EstablishedEvent 411
HeldEvent 422
QueuedEvent 448
RetrievedEvent 452
TransferredEvent 481
Count
AggregateCumulativeEvent 579
AggregateRealtimeEvent 586
Aggregates 198
AggregateWrapupReasonCumulativeElement 589
AggregateWrapupReasonCumulativeEvent 591
CallbackRealtimeEvent 597
CallbackReasons 322
CallbackSchedules 326
ContactData 118, 140
ContactRealtimeEvent 604

CriterionList 344
DeleteList 201
Departments 206
EmailAddresses 366
EmailAttachments 373
EmailDiscardReasons 210
EmailsHistoryQueryResultEvent 400
EmailSummaries 405
EmailTemplates 215
EnumList 124
GroupCumulativeEvent 610
GroupRealtimeEvent 617
Groups 220
KeyList 128
Languages 224
MediaTypes 138
Parties 444
PostProcessingReasons 228
QueueCumulativeEvent 624
QueueRealtimeEvent 632
Queues 234
QueueWrapupReasonCumulativeElement 634
QueueWrapupReasonCumulativeEvent 637
RoutingStateReasons 240
ScheduledCallbacksQueryResultEvent 462
TimeRange 644
UserCumulativeEvent 657
UserEventHandlingStates 491
UserHandlingStates 499
UserRealtimeEvent 665
UserRTHandlingStates 672
Users 249
UserWrapReasonCumulativeElement 674
UserWrapupReasonCumulativeEvent 676
WebCollaborationTemplates 255
WrapupReasons 259
Create 313
CreatedByKey 456
Creating a Manager object without events 35
CreatorAgentKey 304
Criterion 341
CriterionList 342
CurrentPriority 599
CurrentState 600
CustomerName
CallbackCall 304
CallbackRealtimeElement 594
ScheduledCallbackElement 457
WebCollaborationCall 521

D

DefaultRetryInterval 318
Defer 390
DeferTimeStamp 407
Delete 314
DeleteList 200
DeliveredEvent 348
DeliveredToParty 349
Department 203, 242
DepartmentKey 243
Departments 205
Dequeue 557
DequeuedEvent 543
DequeueReason 544
Description
Aggregate 195
CallbackCall 305
CallbackRealtimeElement 594
CallbackReason 319
CallInformationUpdatedEvent 329
ContactRealtimeElement 600
Department 204
DequeuedEvent 546
EmailCall 379
EmailDiscardReason 207
EmailTemplate 212
EnqueuedEvent 548
Group 217
Language 221
PostProcessingReason 226
Queue 231
RoutingCall 553
RoutingInformationUpdatedEvent 566
RoutingStateReason 236
ScheduledCallbackElement 457
VoiceCall 505
WebCollaborationCall 521
WebCollaborationTemplate 251
WrapupReason 257
Destination
ContactRealtimeElement 600
WebCollaborationCall 522
DestinationParty 360
Destroying the Manager Objects 36
Device 446
DiagnosticFileName 148
DiagnosticsFilterEnabled 149
Dial
ServicesEnabled 467
ServicesEnabledEvent 475
VoiceCall 512
DisableDiagnosticFilter 151
Discard 391
DiscardReasonKey 355

Disconnect
ServicesEnabled 467
ServicesEnabledEvent 475
VoiceCall 513
WebCollaborationCall 527
Disconnected 648
DisconnectedEvent 354
DisconnectedParty 356
DisplayName 364
Divert
ServicesEnabled 468
ServicesEnabledEvent 476
VoiceCall 514
DivertedEvent 358
DivertedParty 360
DivertingParty 361
DNIS 506

E

EmailAddress 363, 364
EmailAddresses 364
EmailAttachment 368
EmailAttachments 371
EmailCall 375
EmailDiscardReason 207
EmailDiscardReasons 209
EmailsHistoryQueryResultEvent 399
EmailSummaries 404
EmailSummary 406
EmailTemplate 211
EmailTemplates 214
EnableDiagnosticFilter 152
EndTime 323
Enqueue 558
EnqueuedEvent 547
enumerations
enAdministrationBaseObjectTypes 43
enAdministrationEventCodes 44
enAdministrationEventObjectTypes 45
enAdministrationEventTypes 47
enAdministrationFunctionalities 48
enCallbackOrigins 48
enCallStates 49
enCapability 52
enContactTypes 52
enDeliveredReasons 53
enDequeueReasons 53
enDiagnosticFilters 54
enDisconnectReasons 55
enDivertReasons 56
enEmailsHistoryQueryCriteria 57
enEmailTemplateTypes 57
enEmailTypes 57
enErrors 59
enEventModes 81
enHandlingStates 82
enHiPathProCenterEventCodes 83
enHiPathProCenterEventObjectTypes 84
enHiPathProCenterEventTypes 84
enHiPathProCenterFunctionalities 85
enHoldReasons 85
enKeyValuePairTypes 86
enLocaleIDs 87
enManagerStates 87
enMediaBaseObjectTypes 87
enMediaEventCodes 89
enMediaEventObjectTypes 95
enMediaEventTypes 98
enMediaFunctionalities 100
enMediaTypes 100
enPartyTypes 101
enPresenceStates 101
enResults 102
enRoutingBaseObjectTypes 102
enRoutingEventCodes 103
enRoutingEventObjectTypes 105
enRoutingEventTypes 106
enRoutingFunctionalities 107
enRoutingStates 107
enServers 108
enServerStates 109
enStatisticsEventCodes 111
enStatisticsEventObjectTypes 112
enStatisticsEventTypes 113
enStatisticsFunctionalities 113
enUserTypes 114
enWebCollaborationTemplateTypes 115
EnumList 122
EstablishedEvent 410
EstimatedAnsweredWaitTime
AggregateRealtimeElement 583
QueueRealtimeElement 627
EstimatedServiceLevel
AggregateRealtimeElement 584
QueueRealtimeElement 628
EventOccurred
AdministrationManager 189
HiPathProCenterManager 166
MediaManager 275
RoutingManager 541
events
about 37
AdministrationManager 189
creating a Manager object with 34

examples of 38
HiPathProCenterManager 166
listening for 37
MediaManager 275
RoutingManager 541
stop listening for 37
EventType
AdministrationEvent 193
HiPathProCenterEvent 169
MediaEvent 428
RoutingEvent 562
StatisticsEvent 641
ExitPostProcessing 496
Extension
Agent 280
AgentStatusEvent 295
ForwardingStateEvent 419
ServicesEnabled 468
UserRealtimeElement 660
ExternalConsult 392
ExternalConsultCallID 379
ExternalTransfer 393

F

FailedEvent 415
FirstName 243
ForConsult 436
ForwardingState
Agent 280
ForwardingStateEvent 420
ForwardingStateEvent 419
FromEmailAddress
EmailCall 380
EmailSummary 407
FromParty
DeliveredEvent 350
EstablishedEvent 412
FailedEvent 416
NetworkReachedEvent 432
OffHookEvent 436
OriginatedEvent 439
QueuedEvent 448
TransferredEvent 482
FromXML 146

G

GetFunctionalityState
AdministrationManager 176
HiPathProCenterManager 153
MediaManager 262
RoutingManager 534
StatisticsManager 572
GetVoicePreferredDevice
Agent 287
Group 217
GroupCumulativeElement 606
GroupCumulativeEvent 609
GroupKey
GroupCumulativeElement 607
GroupRealtimeElement 614
GroupRealtimeEvent 616
Groups 218

H

Handled 648
HandlingDirectUsers 614
HandlingRoutedUsers 615
HandlingState
UserEventHandlingState 486
UserHandlingState 493
UserRTHandlingState 668
HandlingStateReason 669
HandlingStateReasonKey
UserEventHandlingState 487
UserHandlingState 494
HandlingStates
Agent 281
AgentStatusEvent 296
UserRealtimeElement 660
HeldEvent 421
HeldParty
DeliveredEvent 350
HeldEvent 422
RetrievedEvent 453
HiPathProCenterEvent 167
HiPathProCenterManager 147
HireAdministrationManager 154
HireMediaManager 155
HireRoutingManager 156
HireStatisticsManager 158
Hold
ServicesEnabled 468
ServicesEnabledEvent 476
VoiceCall 515
HoldingParty 423

I

ID
Agent 282
User 244
IdleUsers 615
Initialize 159
installing the SDK
for all users 31

for the logged on user 30
InvitedParty 335
InvitingParty 335
IsMandatoryWrapUp 494
IsMandatoryWrapupRequired 282
IsOverflow 400
IsPrimary
UserEventHandlingState 487
UserHandlingState 495
UserRTHandlingState 669
Item
AggregateCumulativeEvent 579
AggregateRealtimeEvent 586
Aggregates 199
AggregateWrapupReasonCumulativeEvent 591
CallbackRealtimeEvent 597
CallbackReasons 322
CallbackSchedules 326
ContactData 120, 141
ContactRealtimeEvent 605
CriterionList 344
DeleteList 201
Departments 206
EmailAddresses 366
EmailAttachments 373
EmailDiscardReasons 210
EmailsHistoryQueryResultEvent 402
EmailSummaries 405
EmailTemplates 216
EnumList 124
GroupCumulativeEvent 610
GroupRealtimeEvent 617
Groups 220
KeyList 129
Languages 224
MediaTypes 138
Parties 444
PostProcessingReasons 229
QueueCumulativeEvent 624
QueueRealtimeEvent 632
Queues 235
QueueWrapupReasonCumulativeEvent 637
RoutingStateReasons 240
ScheduleCallbacksQueryResultEvent 462
TimeRange 645
UserCumulativeEvent 658
UserEventHandlingStates 491
UserHandlingStates 499
UserRealtimeEvent 665
UserRTHandlingStates 672
Users 249
UserWrapupReasonCumulativeEvent 676
WebCollaborationTemplates 255
WrapupReasons 260

K

Key
Agent 283
Aggregate 196
CallbackReason 319
Department 204
EmailDiscardReason 208
EmailTemplate 213
Group 218
KeyValuePair 132
Language 222
Queue 231
User 244
WebCollaborationTemplate 251
WrapupReason 257
KeyList 127
KeyValuePair 132

L

Language 221
LanguageKey 251
Languages 223
LastDeferTimeStamp 380
LastName 245
ListenForEvents
AdministrationManager 177
HiPathProCenterManager 161
MediaManager 277
RoutingManager 535
StatisticsManager 573
LocaleID 149
LoggedOnForMediaType 283
LoggedOnMediaTypes 296
LoggedOnUserKey 150
LoggedOnUsers 615
Logoff
Agent 288
ServicesEnabled 469
ServicesEnabledEvent 477
Logon
Agent 289
HiPathProcenterManager 163
ServicesEnabled 469
ServicesEnabledEvent 477

M

Manager
Administration 175
Media 261

Routing 533
Statistics 571
Manager objects
creating 34
destroying 36
Managers in the SDK, overview 33
ManagerStateChangedEvent 135
MaximumCapacity 245
MaximumRoutedHandlingTime 648
MaximumUnusedCapacity 661
MaximumWaitTime
AggregateCumulativeElement 576
QueueCumulativeElement 620
Media Manager 261
MediaBase 425
MediaEvent 426
MediaManager object 261
MediaType
Aggregate 196
ContactRealtimeElement 601
MediaEvent 429
Queue 232
RoutingCall 554
RoutingEvent 562
UserEventHandlingState 488
MediaTypes 136, 245
MiddleInitial 246
MinimumUnusedCapacity 661
MultipleContactHandlingCapable 246
multitenant environment
business unit name required 32
writing applications in 31

N

Name
Aggregate 196
CallbackReason 320
Department 204
EmailAttachment 369
EmailDiscardReason 208
EmailTemplate 213
Group 218
Language 222
PostProcessingReason 226
Queue 232
RoutingStateReason 237
WebCollaborationTemplate 252
WrapupReason 258
NetworkReachedEvent 430
NewAgent 263
NewCallbackCall 263
NewEmailCall 264
_NewEnum
AggregateCumulativeEvent 578
Aggregates 197
AggregateWrapupReasonCumulativeEvent 590
CallbackRealtimeEvent 596
CallbackReasons 321
CallbackSchedules 325
ContactData 117, 139
ContactRealtimeEvent 603
CriterionList 343
DeleteList 200
Departments 205
EmailAddresses 365
EmailAttachments 372
EmailDiscardReasons 209
EmailsHistoryQueryResultEvent 399
EmailSummaries 404
EmailTemplates 215
EnumList 123
GroupCumulativeEvent 609
GroupRealtimeEvent 616
Groups 219
KeyList 127
Languages 223
MediaTypes 137
Parties 443
PostProcessingReasons 227
QueueCumulativeEvent 623
QueueRealtimeEvent 630
Queues 233
QueueWrapupReasonCumulativeEvent 636
RoutingStateReasons 239
ScheduledCallbacksQueryResultEvent 461
TimeRange 643
UserCumulativeEvent 656
UserEventHandlingStates 489
UserHandlingStates 498
UserRealtimeEvent 664
UserRTHandlingStates 671
Users 248
UserWrapupReasonCumulativeEvent 675
WebCollaborationTemplates 254
WrapupReasons 259
NewRoutingCall 536
NewServicesEnabled 264
NewVoiceCall 265
NewWebCollaborationCall 265
NextScheduledPhoneNumber 458
NextScheduledStart 458

O

ObjectType

AdministrationEvent 193
DeleteList 202
HiPathProCenterEvent 171
MediaEvent 429
RoutingEvent 563
StatisticsEvent 641
Offered
GroupCumulativeElement 607
UserCumulativeElement 649
OffHookEvent 435
OldestContactWaitTime
AggregateRealtimeElement 584
QueueRealtimeElement 628
OpenScape Contact Center Manager 147
Origin
CallbackRealtimeElement 594
ScheduledCallbackElement 459
OriginalFromParty
AutoAcknowledgedEvent 300
CallMeEvent 331
ConferencedEvent 336
ContentAddedEvent 338
DeliveredEvent 351
DisconnectedEvent 357
DivertedEvent 361
EstablishedEvent 412
FailedEvent 417
HeldEvent 423
NetworkReachedEvent 433
OffHookEvent 437
OriginatedEvent 440
QueuedEvent 449
RetrievedEvent 453
TransferredEvent 482
OriginalToParty
AutoAcknowledgedEvent 300
CallMeEvent 332
ConferencedEvent 336
ContentAddedEvent 339
DeliveredEvent 351
DisconnectedEvent 357
DivertedEvent 362
EstablishedEvent 413
FailedEvent 417
HeldEvent 424
NetworkReachedEvent 433
OffHookEvent 437
OriginatedEvent 440
QueuedEvent 449
RetrievedEvent 453
TransferredEvent 482
OriginatedEvent 438
OverflowedContacts
AggregateRealtimeElement 584
QueueRealtimeElement 628

P

Parameters 139
Parties
object 442
VoiceCall 506
Party 445
CallMeEvent 332
ContentAddedEvent 339
PartyType 446
patching the SDK 31
Path 369
PostProcessingReason 225
PostProcessingReasons 227
PreferredDevice
Agent 284
ForwardingStateEvent 420
PresenceState
Agent 285
AgentStatusEvent 297
UserRealtimeElement 661
Priority
CallbackCall 305
DequeuedEvent 546
EnqueuedEvent 549
RoutingCall 554
RoutingInformationUpdatedEvent 567
ScheduleCallback 459
programming languages supported 29
purpose of the SDK 29
PushCallMeForm 528

Q

QualifyingAgentsCount 601
Query
MediaManager 266
RoutingManager 537
QueryAggregates 179
QueryAsyncClientEmailsHistory 267
QueryAsyncConversationEmailsHistory 269
QueryAsyncEmailsHistory 270
QueryAsyncScheduledCallbacks 272
QueryCallbackDeleteReasons 180
QueryCallbackRetryReasons 180
QueryDepartments 181
QueryEmailDiscardReasons 181
QueryEmailSummaries 273
QueryEmailTemplates 182
QueryGroups 183

QueryID
AggregateCumulativeEvent 580
AggregateRealtimeEvent 587
AggregateWrapupReasonCumulativeEvent 592
CallbackRealtimeEvent 598
ContactRealtimeEvent 606
EmailsHistoryQueryResultEvent 403
GroupCumulativeEvent 611
GroupRealtimeEvent 618
QueueCumulativeEvent 625
QueueRealtimeEvent 633
QueueWrapupReasonCumulativeEvent 638
ScheduledCallbacksQueryResultEvent 463
UserCumulativeEvent 659
UserRealtimeEvent 666
UserWrapupReasonCumulativeEvent 677
QueryPostProcessingReasons 183
QueryQueues 184
QueryUnavailableReasons 184
QueryUsers 185
QueryWebCollaborationLanguages 185
QueryWebCollaborationTemplates 186
QueryWorkReasons 187
QueryWrapupReasons 187
Queue 230
QueueCumulativeElement 619
QueueCumulativeEvent 622
QueuedEvent 447
QueuedToParty 450
QueueKey
CallbackCall 306
ContactRealtimeElement 602
DequeuedEvent 547
EmailCall 381
EmailSummary 409
EnqueuedEvent 549
QueueCumulativeElement 621
QueueRealtimeElement 629
QueueWrapupReasonCumulativeElement 635
RoutingCall 555
RoutingInformationUpdatedEvent 567
ScheduledCallbackElement 460
VoiceCall 507
WebCollaborationCall 522
QueueKeys
EmailTemplate 213
WebCollaborationTemplate 252
QueueRealtimeElement 626
QueueRealtimeEvent 630
Queues 233
QueueWrapupReasonCumulativeElement 634
QueueWrapupReasonCumulativeEvent 635

R

Reason
DeliveredEvent 352
DisconnectedEvent 358
DivertedEvent 362
HeldEvent 424
ReasonKey
PostProcessingReason 226
RoutingStateReason 237
Received
AggregateCumulativeElement 577
GroupCumulativeElement 608
QueueCumulativeElement 621
ReceivedHereOfferedElsewhere 608
Reconnect
ServicesEnabled 470
ServicesEnabledEvent 478
VoiceCall 516
Redirected
AggregateCumulativeElement 577
QueueCumulativeElement 621
RedirectParty
DeliveredEvent 352
EstablishedEvent 413
TransferredEvent 483
redistributing the SDK runtime 31
ReferenceCallID 441
Reject 529
Remove
ContactData 121, 143
CriterionList 346
EmailAddresses 368
EmailAttachments 375
EnumList 126
KeyList 130
RemoveAll
ContactData 122, 144
CriterionList 347
EnumList 126
KeyList 130
RemoveAt 131
RemoveCallbackSchedule 316
Reply 395
ReplyAttachments 381
ReplyBodyUTF8 382
ReplyFromEmailAddress 382
ReplySubjectUTF8 383
ReplyToBCCEmailAddresses 383
ReplyToCCEmailAddresses 384
ReplyToEmailAddress 384
RequestPostProcessing 497

Requeue
EmailCall 396
WebCollaborationCall 530
Requeued 649
ReserveAgentKey 555
ReserveAgentMaximumTime 556
Resource
AdministrationEvent 194
HiPathProCenterEvent 171
MediaEvent 430
RoutingEvent 563
StatisticsEvent 642
Result
CallbackCall 306
EmailsHistoryQueryResultEvent 403
ScheduledCallbacksQueryResultEvent 464
Resume 397
Retrieve
EmailAttachment 371
ServicesEnabled 470
ServicesEnabledEvent 478
VoiceCall 517
RetrievedEvent 451
RetrievingParty 454
RetryDelay 306
RoutedHeld 649
Routing Manager 533
RoutingBase 550
RoutingCall 551
RoutingEvent 560
RoutingInformationUpdatedEvent 564
RoutingManager object 533
RoutingState
Agent 285
AgentStatusEvent 297
UserRealtimeElement 662
RoutingStateReason 236
RoutingStateReasonKey
Agent 286
AgentStatusEvent 298
UserRealtimeElement 662
RoutingStateReasons 238
runtime product, overview 30

S

ScheduledCallbackElement 455
ScheduledCallbacksQueryResultEvent 460
SDK Managers, overview 33
SendContent 531
SequenceNumber
CallMeEvent 333
ConferencedEvent 337
ContentAddedEvent 340
DeliveredEvent 353
DisconnectedEvent 358
EmailCall 385
EstablishedEvent 414
FailedEvent 418
NetworkReachedEvent 434
OffHookEvent 438
OriginatedEvent 441
QueuedEvent 450
RetrievedEvent 455
TransferredEvent 484
UserEventHandlingState 488
UserHandlingState 495
UserRTHandlingState 669
VoiceCall 507
WebCollaborationCall 523
Server 174
ServerState 174
ServerStateChangedEvent 173
ServiceLevel
AggregateCumulativeElement 577
AggregateRealtimeElement 585
QueueCumulativeElememt 622
QueueRealtimeElement 629
ServicesEnabled 464
ServicesEnabledEvent 472
SetPrimary 497
SetRoutingState
Agent 293
SetVoicePreferredDevice
Agent 292
Source
ContactRealtimeElement 602
WebCollaborationCall 523
StartTime 324
State
AdministrationManager 175
CallbackCall 307
CallbackRealtimeElement 595
EmailCall 385
EmailSummary 409
HiPathProCenterManager 150
ManagerStateChangedEvent 135
MediaManager 261
RoutingCall 556
RoutingManager 533
StatisticsManager 571
VoiceCall 508
WebCollaborationCall 524
Statistics Manager 571
StatisticsEvent 639

StatisticsManager object 571
StopListeningForEvents
AdministrationManager 188
HiPathProCenterManager 164
MediaManager 278
RoutingManager 538
StatisticsManager 574
Subject
EmailSummary 410
SubjectUTF8 386

T

TelephoneNumber
CallbackSchedule 324
CallMeEvent 333
TimedOutEvent 569
Text 340
TimedOutEvent 568
TimeInHandlingState 670
TimeInPrensenceState 662
TimeInRoutingState 663
TimeInState
CallbackRealTimElement 595
ContactRealtimeElement 602
TimeRange 642, 646
ToBCCEmailAddresses 386
ToCCEmailAddresses 387
ToEmailAddress 387
ToEmailAddresses 388
ToParty
DeliveredEvent 353
EstablishedEvent 414
FailedEvent 418
NetworkReachedEvent 434
OriginatedEvent 442
QueuedEvent 451
TransferredEvent 484
TotalAwayTime 651
TotalBusyTime 651
TotalContactActiveTime 670
TotalContactHandlingTime 670
TotalHandledTime 651
TotalHoldTime 652
TotalIdleTime 652
TotalLoggedOnTime 652
TotalPendingTime 653
TotalPostProcessingTime 653
TotalRingTime 653
TotalTalkTime 654
TotalTimeOther 654
TotalWaitTime 603
Transcript 524
Transfer
EmailCall 398
ServicesEnabled 471
ServicesEnabledEvent 479
VoiceCall 518
TransferOut 608
TransferredEvent 480
TransferredOut 654
TransferredParty 485
TrunkID 415
Type
Criterion 341
EmailTemplate 214
KeyValuePair 133
MediaBase 426
RoutingBase 551
User 247
WebCollaborationTemplate 252

U

UnassignedEvent 569
Unavailable
ServicesEnabled 471
ServicesEnabledEvent 479
Undelivered 655
Unhandled 655
Update
MediaManager 274
RoutingManager 540
UpdateCallbackSchedule 317
User 241
UserCumulativeElement 647
UserCumulativeEvent 656
UserEventHandlingState 485
UserEventHandlingStates 488
UserHandlingState 492
UserHandlingStates 497
UserKey
UserCumulativeElement 655
UserRealtimeElement 663
UserWrapupReasonCumulativeElement 674
UserMediaNotReadyEvent 500
UserMediaReadyEvent 501
UserMediaType
UserMediaNotReadyEvent 501
UserMediaReadyEvent 502
UserName 247
UserRealtimeElement 659
UserRealtimeEvent 663
UserRTHandlingState 666
UserRTHandlingStates 671
Users 247

UserWrapupReasonCumulativeElement 673
UserWrapupReasonCumulativeEvent 675
Utilization 656

V

Value
    Criterion 342
    KeyValuePair 133
Visible 134
VoiceCall 502

W

WaitTime
    CallbackCall 308
    EmailCall 388
    VoiceCall 508
    WebCollaborationCall 525
WebCollaborationCall 519
WebCollaborationTemplate 250
WebCollaborationTemplates 254
Work
    ServicesEnabled 472
    ServicesEnabledEvent 480
WrapupReason 256
WrapupReasonKey
    AggregateWrapupReasonCumulativeElement 589
    QueueWrapupReasonCumulativeElement 635
    UserWrapupReasonCumulativeElement 674
WrapupReasons 258
WriteToDiagnosticFile 165

X

XMLAccess 144

© 2024 Mitel Networks Corporation. All Rights Reserved. Mitel and the Mitel logo are trademark(s) of Mitel Networks Corporation. Unify and associated marks are trademarks of Unify Software and Solutions GmbH & Co. KG. All other trademarks herein are the property of their respective owners.