ORACLE
PEOPLESOFT

PeopleTools 8.54: MultiChannel Framework

November 2016

ORACLE

PeopleTools 8.54: MultiChannel Framework
CDSKU
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

License Restrictions Warranty/Consequential Damages Disclaimer

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

Warranty Disclaimer

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

Restricted Rights Notice

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

Hazardous Applications Notice

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Trademark Notice

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

Third Party Content, Products, and Services Disclaimer

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Contents

Preface ..... **xxix**
Understanding the PeopleSoft Online Help and PeopleBooks xxix
PeopleSoft Hosted Documentation xxix
Locally Installed Help xxix
Downloadable PeopleBook PDF Files xxix
Common Help Documentation xxx
Field and Control Definitions xxx
Typographical Conventions xxx
ISO Country and Currency Codes xxxi
Region and Industry Identifiers xxxi
Using and Managing the PeopleSoft Online Help xxxii
Understanding PeopleSoft MultiChannel Framework xxxii
PeopleTools Related Links xxxii
Contact Us xxxiii
Follow Us xxxiii
Chapter 1: Getting Started with PeopleSoft MultiChannel Framework **35**
PeopleSoft MultiChannel Framework Overview 35
PeopleSoft MultiChannel Framework Implementation 35
Chapter 2: Understanding PeopleSoft MultiChannel Framework **39**
PeopleSoft MultiChannel Framework 39
PeopleSoft MultiChannel Framework Elements and Channels 39
PeopleSoft MultiChannel Framework Elements 39
PeopleSoft MultiChannel Framework Channels 40
PeopleSoft MultiChannel Framework Universal Queue 41
PeopleSoft MultiChannel Console 42
PeopleSoft MCF Architecture 42
PeopleSoft MCF Server Architecture 42
Chat Architecture 43
Email Architecture 45
Chapter 3: Configuring PeopleSoft Computer Telephony Integration **47**
Understanding the PeopleSoft CTI Console 47
PeopleSoft CTI 47
PeopleSoft CTI Components 48
Adapter-Based CTI Requirements 49
PeopleSoft MultiChannel API 50
Required Security for PSMCAPI 52
JavaScript MultiChannel API 52
Applet-Based CTI Requirements 52
Java Applet Console 52
Java Runtime Environment 53
Configuring PeopleSoft CTI 54
Installing PeopleSoft CTI 54
Enabling PeopleSoft CTI 54
Configuring CTI Console Type 55
Creating a List of Frequently Dialed Phone Numbers 56
Entering Default Screen Pop-Up URL 57
Using the Reason Code Page 58

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Configuring PeopleSoft CTI Using Adapters....59
Configuring the CTI Console....59
Configuring PeopleSoft CTI Using Applets....60
Configuring the CTI Applet Console....61
Using the CTI Genesys Page....62
Using the CTI Cisco Page....63
Configuring PeopleSoft CTI Queues and CTI Agents....67
Configuring CTI Queues....67
Configuring CTI Agents....68
Using the Phone Book Page....70
Personalizing the Agent Console....71
Viewing Information About the Agent Information Page....73
Using Other PeopleSoft CTI Options....74
Configuring Pop-Up Windows....74
Supporting Single Sign-In....77
Logging CTI Events....77
Implementing Free Seating....78
Using the PeopleSoft CTI Sample Pages....78
Using the Outbound Call Page....78
Chapter 4: Using PeopleSoft CTI....81
Understanding PeopleSoft CTI....81
PeopleSoft CTI....81
Cisco Switch Considerations for the Applet-Based Solution....82
Using PeopleSoft CTI....82
Getting Started....83
Using the CTI Console....85
Selecting Call Actions....86
Answering a Call....88
Transferring a Caller....88
Initiating Conference Calls....89
Working with the Hold Status....90
Disconnecting a Caller....91
Switching Agent Ready Status....91
Dialing an Outbound Call....92
Completing a Call....93
Using Hot Keys....93
Chapter 5: Configuring REN Servers....95
Understanding REN Servers....95
REN Server Failover, Scalability, and Security Configuration....96
REN Server Failover....97
REN Server Clusters....97
Understanding SSL-Enabled REN Servers....98
Installing Digital Certificates....98
Authenticating Server and Client....98
Performance and Scalability for SSL-Enabled REN Servers....99
Configuring REN Server Security....99
Understanding REN Server Security Configuration....99
Defining Permission Lists for REN Server Access....100
Configuring REN Servers....101
Understanding REN Server Configuration Options....101
Configuring REN Servers and SSL-Enabled REN Servers....103

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Defining REN Servers....107
Configuring REN Server and SSL-Enabled REN Server Clusters.... 111
Defining a REN Server Cluster.... 111
Specifying REN Server Ownership.... 114
Specifying REN Server Cluster Members.... 115
Configuring a Reverse Proxy Server with a REN Server....116
Understanding RPS Configuration....116
Example: Configuring a WebLogic RPS for a REN Server on Another Host Machine.... 116
Configuring Apache-based Reverse Proxy Servers for a REN Server.... 118
Chapter 6: Configuring PeopleSoft MCF Servers and Clusters....121
Understanding PeopleSoft MCF Server and Cluster Architecture....121
PeopleSoft MCF Server Configuration....121
PeopleSoft MCF Cluster Architecture.... 122
Queue Server and Queue Server Failover.... 123
Logical Queues and Physical Queues....124
PeopleSoft MCF Log Server and Log Server Failover.... 125
Queue Server Scalability....125
Recommended Configurations.... 126
Configuring PeopleSoft MCF Clusters....126
Understanding PeopleSoft MCF Cluster Configuration.... 126
Configuring PeopleSoft MCF Clusters.... 127
Chapter 7: Configuring PeopleSoft MCF Queues and Tasks.... 129
Defining Queues....129
Defining Queues....129
Defining Chat Responses.... 131
Defining Static Push URLs....131
Configuring Tasks.... 132
Configuring Tasks....133
Viewing the Cluster Summary....137
Viewing the Cluster Summary.... 137
Tuning Cluster Parameters....138
Tuning Cluster Parameters.... 138
Notifying Clusters of Changed Parameters.... 147
Notifying Clusters of Changed Parameters....147
Chapter 8: Configuring PeopleSoft MCF Agents.... 149
Defining Agents.... 149
Creating Agents.... 149
Specifying Languages That an Agent Supports....151
Personalizing an Agent's Presence....151
Defining Optional Agent Characteristics....152
Setting Up Buddy Lists....153
Configuring Windows....154
Personalizing Chat....155
Specifying Agent-Specific URLs....157
Specifying Miscellaneous Parameters....158
Chapter 9: Administering Queues, Logs, and Tasks....161
Administering Physical Queues....161
Moving Agents Between Physical Queues.... 161
Moving Queues....162
Balancing Queues.... 163
Viewing Queue Server, Queue, and Agent States....164

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Viewing the Queue Server State....164
Viewing the Queue State Summary....165
Viewing the Agent State Summary....165
Viewing Broadcast, Chat, and Event Logs....166
Viewing Broadcast Logs....167
Viewing Chat Logs....168
Viewing Event Logs....169
Viewing PeopleSoft MCF Logs....172
Administering Overflow and Escalated Tasks....172
Administering Overflow Tasks....172
Administering Escalated Tasks....173
Chapter 10: Managing Tasks and Using Chat in PeopleSoft MultiChannel Framework....175
Managing Tasks with the MultiChannel Console....175
Using the MultiChannel Console to Work with Tasks....175
Communicating with Customers and Agents Using Chat....177
Using the Agent Chat Window....177
Using the Customer Chat Window....180
Chapter 11: Using PeopleSoft MCF Broadcast and Working with Sample Pages....183
Using PeopleSoft MCF Broadcast....183
Understanding JSMCAPI Broadcast....184
Implementing MCF Broadcast....184
Using JSMCAPI Broadcast with MCF Supervisor Console....185
Using PeopleCode Broadcast....189
Viewing Broadcast Logs....193
Working with Sample Pages....193
Using the Customer Chat Sample Page....193
Using the URL Wizard....195
Using the Generic Event Sample Page....196
Using the Generic Event Window....198
Using the Email Sample Page....199
Using the Email Window....200
Using and Demonstrating JSMCAPI....201
Understanding JSMCAPI....201
Common Elements Used in This Section....206
Using the CTI Sample Console....207
Using the Agent Console Page....215
Using the Monitor Agents Page and Sample Monitor - Agent States Page....218
Using the Monitor Queues Page and Sample Monitor - Queue Statistics Page....221
Using PeopleCode Built-in Functions with PeopleSoft MultiChannel Framework....224
Using Universal Queue Classes....225
Chapter 12: Configuring PeopleSoft MCF for Third-Party Routing Systems....227
Understanding Third-Party Routing Systems....227
Defining Third-Party Routing System Requirements....230
Defining Third-Party Routing Rules....230
Configuring PeopleSoft MCF for a Third Party....231
Defining the Third-Party Flag....231
Defining the Third-Party Flag....232
Defining the PeopleSoft MCF Cluster Page for a Third Party....232
Defining the PeopleSoft MCF Cluster Page for a Third Party....232
Tuning PeopleSoft MCF Cluster Parameters for a Third Party....234
Tuning PeopleSoft MCF Cluster Parameters for a Third Party....234

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Notifying PeopleSoft MCF Clusters of Changed Parameters for a Third Party....237
Notifying Third-Party Clusters of Changed Parameters....237
Defining PeopleSoft MCF Queues for a Third Party....238
Defining PeopleSoft MCF Queues for a Third Party....238
Defining Canned Queue Messages....240
Defining Canned Queue URLs....240
Defining PeopleSoft MCF Agents for a Third Party....241
Creating PeopleSoft MCF Agents for a Third Party....242
Setting Up Buddy Lists....243
Customizing Windows....244
Defining Messages....245
Specifying Agent-Specific URLs....246
Defining Agent's Presence....247
Specifying the Media....248
Specifying Languages That an Agent Supports....249
Specifying Miscellaneous Parameters....250
Configuring PeopleSoft MCF Tasks for a Third Party....252
Configuring PeopleSoft MCF Task for a Third Party....252
Configuring CTI for a Third Party....252
Communicating with Customers and Agents Using Chat....252
Using the Third-Party Chat Window....252
Using the Customer Chat Window....255
Viewing Event Logs for a Third Party....256
Viewing Event Logs for a Third Party....257
Viewing Broadcast Logs for a Third Party....258
Viewing Broadcast Logs for a Third Party....258
Working with Third-Party Sample Pages....258
Using the Customer Chat Sample Page....259
Using the Generic Event Sample Page....259
Using the Email Sample Page....259
Using the Sample Console Page....259
Using the MCF Broadcast Page....268
Using the PCodeBroadcast Page....269
Chapter 13: Understanding JSMCAPI Classes....271
Understanding JSMCAPI....271
Understanding JSMCAPI Classes....271
PSMC....276
Server....277
RENServer....278
Session....278
_Address....278
Line....279
Connection....279
Group....279
Task....279
_User....280
MediaType....280
Reason....280
Statistics....280
Data....281
Globals....281

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

MCEvent....281
Caps....281
ForwardMode....282
_Address Class Hierarchy....282
_Address Class Constructor....283
_Address....284
_Address Class Fields....284
caps....284
id....284
_Address Class Callback Event Method....284
onError....284
_UQAddress Class Constructor....285
_UQAddress....285
_UQAddress Class Fields....285
Tasks....285
_UQAddress Methods....286
acceptTask....286
dequeueTask....286
_UQAddress Class Callback Event Methods....286
onAccepted....287
onAcceptingTask....287
onDequeueingTask....287
onNotify....287
onTaskAdded....287
onTaskRemoved....288
onUnassigned....288
_User Class Hierarchy....288
_User Class Constructor....289
_User....289
_User Class Fields....289
agentID....289
caps....289
id....289
name....290
presences....290
states....290
ST_LOGGEDIN....290
ST_LOGGEDOUT....290
ST_NOTREADY....290
ST_READY....291
ST_UNKNOWN....291
ST_WORKNOTREADY....291
ST_WORKREADY....291
statistics....291
statistics1....291
statistics2....292
A2AChat Class Constructor....292
A2AChat....292
A2AChat Class Fields....292
address....293
agentID....293

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

agentName 293
appData 293
chatType 293
customerName 294
id 294
isConference 294
jr 294
question 295
subject 295
type 295
uniqueueId 295
A2AChat Class Method 295
getURL 295
A2AChatAddress Class Constructor 296
A2AChatAddress 296
A2AChatAddress Class Fields 296
id 296
tasks 297
A2AChatAddress Class Method 297
initiateChat 297
A2AChatAddress Class Callback Event Methods 297
onChatEnded 297
onInitiatingChat 298
onNotify 298
AgentStatistics Class Constructor 298
AgentStatistics 298
AgentStatistics Class Fields 298
averageCallDuration 299
averageHoldDuration 299
callsHandled 299
data 299
percentIdleTime 299
percentTimeAvailable 299
percentTimeInCurrentState 300
percentTimeUnavailable 300
timeCurrentLogin 300
timeWorking 300
totalTaskAcceptedLogin 300
totalTaskDoneLogin 300
totalTaskUnassignedLogin 301
unavailableDuration 301
waitDuration 301
AppData Class Constructor 301
AppData 301
AppData Class Fields 302
data 302
groupId 302
jr 302
question 302
strData 302
subject 302

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

uniqueId.... 303
url.... 303
userId....303
username.... 303
wizUrl.... 303
AppData Class Method....304
addKeyValue.... 304
Buddy Class Constructor.... 304
Buddy....304
Buddy Class Fields.... 305
Buddy Class Callback Event Methods.... 305
onStat1.... 305
onStat2.... 305
onState....306
Call Class Constructor.... 306
Call....306
Call Class Fields.... 306
line....307
statistics....307
CallData Class Constructor....307
CallData.... 307
CallData Class Fields....308
ani....308
callId.... 308
callType....308
data....308
dnis....308
CallData Class Method.... 309
addKeyValue.... 309
CallStatistics Class Constructor....309
CallStatistics....309
CallStatistics Class Fields....310
data....310
holdTime.... 310
queueTime....310
talkTime.... 310
Chat Class Constructor.... 310
Chat.... 311
Chat Class Fields.... 311
address....312
agentId....312
appData.... 312
chatconnection.... 312
chatType.... 312
customerName.... 313
groupId....313
question....313
subject.... 313
statistics....313
userData.... 313
Chat Class Method....314

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

gettpUrl.... 314
getUrl.... 314
ChatAddress Class Constructor.... 314
ChatAddress.... 315
ChatAddress Class Fields.... 315
chatconnections.... 315
ChatAddress Class Methods.... 316
chat.... 316
getChatconnectionByConnectionId.... 316
getChatconnectionindexByConnectionId.... 317
getFreeChatconnection.... 317
getFreeChatconnectionIndex.... 317
ChatAddress Callback Event Methods.... 317
onCapabilitiesChanged.... 318
ChatConnection Class Constructor.... 318
ChatConnection.... 318
ChatConnection Class Fields.... 319
caps.... 319
chat.... 319
connectionId.... 319
id.... 319
state.... 319
ChatConnection Class Methods.... 320
answer.... 320
attachUserData.... 320
conference.... 321
forward.... 321
gethistory.... 322
getUrl.... 322
message.... 323
pushURL.... 323
reject.... 323
release.... 324
typing.... 324
wrapup.... 325
ChatConnection Class Callback Event Methods.... 325
onAccepted.... 325
onAnswering.... 325
onCapabilitiesChanged.... 326
onChatdataChanged.... 326
onConferencing.... 326
onDialing.... 326
onDropped.... 326
onError.... 327
onForwarded.... 327
onForwardError.... 327
onForwarding.... 327
onHistory.... 327
onIncomingChat.... 328
onMessage.... 328
onPartyAdded.... 328

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onPartyChanged....328
onPartyRemoved....328
onProperties....329
onPushURL....329
onRejected....329
onReleased....329
onReleasing....329
onRevoked....330
onTalking....330
onTyping....330
onUserdataChanged....330
ChatConnectionCaps Class Constructor....330
ChatConnectionCaps....330
ChatConnectionCaps Class Fields....331
canAnswer....331
canConference....331
canConferenceSingle....331
canForward....331
canIndicateTyping....332
canPushURL....332
canReject....332
canSendMessage....332
ChatData Class Constructor....332
ChatData....332
ChatData Class Fields....333
data....333
ChatData Class Methods....333
addKeyValue....333
Email Class Constructor....333
Email....333
Email Class Fields....334
address....334
agentId....335
appData....335
customerName....335
emailconnection....335
emailId....335
groupId....335
question....336
statistics....336
subject....336
userData....336
Email Class Method....336
gettpUrl....336
getUrl....337
EmailAddress Class Constructor....337
EmailAddress....337
EmailAddress Class Fields....338
agent....338
emailconnections....338
EmailAddress Class Methods....338

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

getEmailconnectionByConnectionId.... 338
getEmailconnectionindexByConnectionId.... 339
getFreeEmailconnection....339
EmailAddress Callback Event Methods.... 339
EmailConnection Class Constructor.... 340
EmailConnection....340
EmailConnection Class Fields.... 340
caps.... 340
connectionId....340
email....341
id.... 341
state.... 341
EmailConnection Class Methods....341
abandon....341
answer.... 342
attachUserData....342
complete....343
forward....343
reject....343
withdraw_RES.... 344
EmailConnection Class Callback Event Methods.... 344
onAnswering.... 344
onCapabilitiesChanged.... 345
onCompleted.... 345
onDropped....345
onEmaildataChanged.... 345
onError.... 345
onForwarded.... 346
onForwardError....346
onForwarding.... 346
onIncoming.... 346
onProcessing.... 346
onRejected....347
onRevoked.... 347
onUserdataChanged.... 347
onWithdraw_REQ....347
EmailConnectionCaps Class Constructor.... 347
EmailConnectionCaps....347
EmailConnectionCaps Class Fields.... 348
canAnswer....348
canComplete....348
canForward.... 348
canReject....348
EmailData Class Constructor....349
EmailData....349
EmailData Class Fields....349
data....349
EmailData Class Methods....349
addKeyValue.... 349
Extension Class Constructor....350
Extension....350

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Extension Class Fields....350
forwardMode....351
isDnd....351
lines....351
numOfLines....351
Extension Class Methods....351
cancelDnd....351
cancelForwardSet....352
forwardSet....352
getDialingLine....353
getFreeLine....353
getLineByConnectionId....353
getOffHookLine....354
setDnd....354
Extension Class Callback Event Methods....355
onCancelingDnd....355
onCancelingForward....355
onDnd....355
onDndCanceled....355
onForwardCanceled....356
onForwarded....356
onForwarding....356
onSettingDnd....356
ExtensionCaps Class Constructor....356
ExtensionCaps....357
ExtensionCaps Class Fields....357
canCancelDnd....357
canDial....357
canFwdBusy....357
canFwdBusyNoAnswer....358
canFwdCancelForward....358
canFwdDefault....358
canFwdNoAnswer....358
canFwdUnconditional....358
canRefreshState....358
canSetDnd....359
ForwardMode Class Constructor....359
ForwardMode....359
ForwardMode Class Field....359
mode....359
GenericAddress Class Constructor....360
GenericAddress....360
GenericAddress Class Fields....360
agent....361
genericconnections....361
GenericAddress Class Methods....361
getFreeGenericconnection....361
getGenericconnectionByConnectionId....362
getGenericconnectionindexByConnectionId....362
GenericAddress Class Callback Event Methods....362
GenericConnection Class Constructor....363

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

GenericConnection....363
GenericConnection Class Fields....363
caps....363
connectionId....363
generic....364
id....364
state....364
GenericConnection Class Methods....364
abandon....364
answer....365
attachUserData....365
complete....366
forward....366
reject....366
withdraw_RES....367
GenericConnection Class Callback Event Methods....367
onCapabilitiesChanged....367
onCompleted....368
onDropped....368
onError....368
onForwarded....368
onForwardError....368
onForwarding....369
onGenericdataChanged....369
onIncoming....369
onProcessing....369
onRejected....369
onRevoked....370
onUserdataChanged....370
onWithdraw_REQ....370
GenericConnectionCaps Class Constructor....370
GenericConnectionCaps....370
GenericConnectionCaps Class Fields....371
canAnswer....371
canComplete....371
canForward....371
canReject....371
GenericData Class Constructor....371
GenericData....372
GenericData Class Fields....372
data....372
GenericData Class Methods....372
addKeyValue....372
GenericTask Class Constructor....373
GenericTask....373
GenericTask Class Fields....373
address....374
agentId....374
appData....374
customerName....374
genericconnection....374

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

genericId....375
groupId....375
question....375
statistics....375
subject....375
userdata....375
GenericTask Class Method....376
gettpUrl....376
getUrl....376
GLOBALS Class Fields....376
A2AChat.PS_JR....377
A2AChat.TYPE_ANSWER....377
A2AChat.TYPE_CONSULT....377
Server.TYPE_CTI....377
Server.TYPE_UQ....377
Task.TYPE_A2ACHAT....378
Task.TYPE_CHAT....378
Task.TYPE_CTI....378
Task.TYPE_EMAIL....378
Task.TYPE_GENERIC....378
GLOBALS Class Methods....379
initJSMCAPI....379
isValid....379
MCFBroadcast....380
Group Class Constructor....380
Group....380
Group Class Fields....381
id....381
name....381
registered....381
statistics....381
statistics1....382
statistics2....382
Group Class Callback Event Methods....382
onStat....382
onStat1....382
onStat2....383
onTaskAdded....383
onTaskRemoved....383
GroupStatistics Constructor....383
GroupStatistics....383
GroupStatistics Fields....384
data....384
listOfTasksInTheQueueByTaskType....384
maxTaskCompletionTime....384
newestTask....384
newestTaskCompletionTime....384
numberOfAbandoned....384
numberOfLoggedIn....385
numberOfQueued....385
numUnassignedTasks....385

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

queuedWaitTime.... 385
queueUpTime.... 385
relativeQueueLoad.... 385
timeElapsedOldestTask.... 385
GroupStatistics1 Class Constructor.... 385
GroupStatistics1.... 386
GroupStatistics1 Class Fields.... 386
mostRecentTaskDone.... 386
mostRecentTaskDoneData.... 386
mostRecentTaskEnqueued.... 386
mostRecentTaskEnqueuedData.... 387
numAgentsAvailable.... 387
numAgentsLoggedIn.... 387
numEscalation.... 387
numOverflow.... 387
numTaskAccepted.... 387
numTaskDone.... 388
numTaskQueued.... 388
reasonFlag.... 388
taskTotalTimeInSystem.... 388
timeSinceStart.... 388
GroupStatistics2 Class Constructor.... 388
GroupStatistics2.... 388
GroupStatistics2 Class Fields.... 389
averageTaskDuration.... 389
averageWaitTime.... 389
oldestTask.... 389
recentTask.... 389
timeElapsedOldestTask.... 390
timeElapsedRecentTask.... 390
Line Class Constructor.... 390
Line.... 390
Line Class Fields.... 390
call.... 390
caps.... 391
connectionid.... 391
id.... 391
isMuted.... 391
state.... 391
Line Class Methods.... 392
alternate.... 392
answer.... 392
attachUserData.... 393
clear.... 393
complete.... 393
conference.... 394
conferenceSingle.... 394
dial.... 395
dropParty.... 395
getAni.... 396
getDescr.... 396

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

getDnis.... 396
getPadvalue.... 397
getReferenceId.... 397
getUrl.... 397
grabCall.... 398
hold.... 398
join.... 398
mute.... 399
park.... 399
reconnect.... 399
reject.... 400
release.... 400
retrieve.... 401
sendDTMF.... 401
setcallresult.... 401
setcallresultDNC.... 402
setcallresultReschedule.... 402
transfer.... 403
transferMute.... 403
unmute.... 404
updateCallData.... 404
Line Class Callback Event Methods.... 404
onAlternating.... 404
onAnswering.... 405
onAttachingUD.... 405
onCallDataChanged.... 405
onCapabilitiesChanged.... 405
onClearing.... 405
onCompleting.... 406
onConferencing.... 406
onDialing.... 406
onDropped.... 406
onError.... 406
onGrabbing.... 407
onHeld.... 407
onHolding.... 407
onJoining.... 407
onMuted.... 407
onOffHook.... 408
onOnHook.... 408
onParking.... 408
onPartyAdded.... 408
onPartyChanged.... 408
onPartyRemoved.... 409
onReconnecting.... 409
onRejected.... 409
onRejecting.... 409
onReleasing.... 409
onRetrieving.... 410
onRinging.... 410
onSetcallresult.... 410

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onSetcallresultDNC.... 410
onSetcallresultReschedule.... 410
onTalking.... 411
onTransfering.... 411
onUnmuted.... 411
onUpdatingCD.... 411
onUserDataChanged.... 411
LineCaps Class Constructor.... 412
LineCaps.... 412
LineCaps Class Fields.... 412
canAlternate.... 412
canAnswer.... 412
canAttachUserDate.... 413
canClear.... 413
canComplete.... 413
canConference.... 413
canConferenceSingle.... 413
canDropParty.... 413
canHold.... 414
canMute.... 414
canPark.... 414
canReconnect.... 414
canReject.... 414
canRelease.... 414
canRetrieve.... 415
canSendDTMF.... 415
canSetcallresult.... 415
canSetcallresultDNC.... 415
canSetcallresultReschedule.... 415
canTransfer.... 415
canTransferMute.... 416
canUnmute.... 416
canUpdateCallData.... 416
MCEvent Class Constructor.... 416
MCEvent.... 416
MCEvent Class Fields.... 417
extension.... 417
group.... 417
reason.... 417
user.... 417
MediaType Class Constructor.... 418
MediaType.... 418
MediaType Class Field.... 418
type.... 418
PSMC Class Constructor.... 419
PSMC.... 419
PSMC Class Fields.... 419
renserver.... 419
servers.... 419
sessions.... 420
PSMC Class Methods.... 420

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

closeSession.... 420
getCallById.... 420
getChatById.... 421
getEmailById.... 421
getGenericTaskById.... 421
getLineById.... 422
openSession.... 422
start.... 423
stop.... 423
Reason Class Constructor.... 423
Reason.... 423
Reason Class Fields.... 424
code.... 424
desc.... 424
reasonData1.... 424
reasonData2.... 425
reasonData3.... 425
RenServer Class Constructor.... 425
RenServer.... 425
RenServer Class Fields.... 425
isRunning.... 425
url.... 426
RenServer Class Callback Event Methods.... 426
onDown.... 426
onUp.... 426
Server Class Constructor.... 426
Server.... 426
Server Class Fields.... 427
id.... 427
info.... 427
state.... 427
type.... 427
Server Class Callback Event Methods.... 428
onBroadcast.... 428
onHbLost.... 428
onHbRecovered.... 429
onInService.... 429
onOutOfService.... 429
onRestart.... 429
Session Class Constructor.... 429
Session.... 429
Session Class Fields.... 430
addresses.... 430
buddies.... 430
groups.... 430
id.... 430
intervalBetweenReqs.... 431
numberRegsPerBulkReq.... 431
serverId.... 431
state.... 431
user.... 431

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Session Class Methods....432
broadcastSubscribe....432
broadcastUnsubscribe....432
close....432
open....433
registerAddress....433
registerBuddy....433
registerBuddiesBulk....434
registerGroup....434
registerGroupsBulk....435
registerUser....435
setAutoRecovery....435
statPublish....436
unregisterAddress....436
unregisterBuddy....437
unregisterGroup....437
unregisterUser....437
Session Class Callback Event Methods....438
onAddressRegistered....438
onAddressUnregistered....438
onBuddyRegistered....438
onBuddyUnregistered....438
onClosed....439
onError....439
onGroupRegistered....439
onGroupUnregistered....439
onInfo....439
onOpened....440
onUserRegistered....440
onUserUnregistered....440
Task Class Hierarchy....441
Task Class Constructor....441
Task....441
Task Class Fields....442
caseid....442
cost....442
customerid....442
group....442
id....442
onStat....443
priority....443
type....443
urlAbs....444
urlRel....444
TaskStatistics Class Constructor....444
TaskStatistics....444
TaskStatistics Class Fields....444
data....444
holdTime....445
queueTime....445
talkTime....445

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

User Class Constructor.... 445
User.... 445
User Class Fields.... 446
addresses.... 446
agentPassword.... 447
language.... 447
registeredAddresses.... 447
statesUq.... 447
User Class Methods.... 447
ctiBusyUq.... 447
disableMedia.... 448
enableMedia.... 448
isMediaEnabled.... 449
login.... 449
loginUq.... 450
logout.... 450
logoutUq.... 450
register.... 451
setNotReady.... 451
setPresence.... 452
setPresenceUq.... 452
setReady.... 452
setWorkNotReady.... 453
setWorkReady.... 453
unregister.... 454
User Class Callback Event Methods.... 454
onCapabilitiesChanged.... 454
onCtiBusy.... 455
onCTIBUSYUq.... 455
onCtiClear.... 455
onDropped.... 455
onError.... 455
onInfo.... 456
onLoggedIn.... 456
onLoggedOut.... 456
onLoggingIn.... 456
onLoggingOut.... 456
onMediaDisabled.... 457
onMediaEnabled.... 457
onNotReady.... 457
onPresenceChanged.... 457
onReady.... 457
onRegistered.... 458
onRegistering.... 458
onSettingNotReady.... 458
onSettingPresence.... 458
onSettingReady.... 458
onSettingWorkNotReady.... 459
onSettingWorkReady.... 459
onStat.... 459
onStat1.... 459

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onStat2 ... 459
onUnknown ... 460
onUnregistered ... 460
onUnregistering ... 460
onWorkNotReady ... 460
onWorkReady ... 460
UserCaps Class Constructor ... 461
UserCaps ... 461
UserCaps Class Fields ... 461
canHandleChat ... 461
canHandleEmail ... 461
canHandleGeneric ... 461
canHandleVoice ... 462
canLogin ... 462
canLogout ... 462
canRefreshState ... 462
canSetNotReady ... 462
canSetPresence ... 462
canSetReady ... 463
canSetWorkNotReady ... 463
canSetWorkReady ... 463
UserData Class Constructor ... 463
UserData ... 463
UserData Class Fields ... 464
UserData Class Field Constants ... 464
UserData Class Method ... 464
addKeyValue ... 464
UserStatistics1 Class Constructor ... 465
UserStatistics1 ... 465
UserStatistics1 Class Fields ... 465
availableCost ... 465
ctiBusy ... 465
currentQueue ... 466
mostRecentTaskData ... 466
mostRecentTaskId ... 466
numTaskAccepted ... 466
numTasksDone ... 466
numTasksUnassigned ... 466
presenceText ... 467
reasonFlag ... 467
state ... 467
timeInCurrentState ... 467
timeSinceLoggedIn ... 467
UserStatistics2 Class Constructor ... 467
UserStatistics2 ... 467
UserStatistics2 Class Fields ... 468
currentQueue ... 468
timeIdle ... 468
timeInCurrentState ... 468
timeNotReady ... 468
timeSinceLogin ... 469

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

totalTimeAvailable....469
totalTimeUnavailable....469
Chapter 14: Configuring the Email Channel.... 471
Understanding the Email Channel....471
Handling Email....472
Configuring Email Channel in PeopleSoft Integration Broker.... 474
Configuring GETMAILTARGET Connector Properties....475
Enabling Virus Scanning.... 483
Locating virusscan.xml File for Your Web Server.... 483
Configuring the virusscan.xml File....484
Using Virus Scan Error Logs....486
Demonstrating the Email Channel....487
Using the GetMail - Server Page....487
Using the MailStore - DB Page....490
Chapter 15: Configuring Instant Messaging in PeopleSoft MultiChannel Framework.... 493
Understanding Instant Messaging....493
Configuring Instant Messaging Server Details....493
Configuration for Yahoo....494
Configuring XMPP Domains.... 495
Configuring IM Users for XMPP Servers.... 495
Using the Instant Messaging Sample Pages....496
Pages Used to Test Instant Messaging....496
Using the PeopleCode Sample Page....497
Using the Button Sample Page.... 498
Developing an Application with Instant Messaging Action Button for Presence Detection....499
Using Single Button Presence.... 499
Understanding Presence Detection....499
Instant Messaging Servers Configuration for Single Presence Button.... 500
Configuring Screen Names.... 501
Determining your Screen Name....501
Creating a MSN Badge.... 501
Testing Single Presence Button....502
Developing a Sample Application with Single Presence Button....502
Installing Instant Messenger Clients....503
Working with Domain Offline and Online Buttons.... 504
Debugging IM.... 505
Viewing IMServlet Logs.... 505
Appendix A: JSMCAPI Quick Reference.... 507
JSMCAPI Classes.... 507
_Address.... 507
_UQAddress....507
_User.... 508
A2AChat.... 509
A2AChatAddress.... 510
AgentStatistics.... 511
AppData.... 512
Buddy....513
Call....514
CallData.... 515
CallStatistics....516
Chat.... 516

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

ChatAddress....518
ChatConnection....520
ChatConnectionCaps....522
ChatData....522
Connection....523
ConnectionListener....523
ConnectionRequest....524
Email....524
EmailAddress....525
EmailConnection....527
EmailConnectionCaps....528
EmailData....529
Extension....529
ExtensionCaps....531
ForwardMode....532
GenericAddress....532
GenericConnection....534
GenericConnectionCaps....535
GenericData....536
GenericTask....536
GLOBALS....538
Group....539
GroupStatistics....540
GroupStatistics1....540
GroupStatistics2....541
Line....542
LineCaps....545
MCEvent....547
MediaType....547
PSMC....548
Reason....549
RenServer....549
Server....550
Session....551
Task....553
TaskStatistics....554
User....555
UserCaps....558
UserData....559
UserStatistics1....560
UserStatistics2....560
Appendix B: Installing Digital Certificates for REN SSL....563
Installing Digital Certificates....563
Installing the CA Server Certificate....564
Installing the REN Server Certificate....564
Configuring Digital Certificates....565
Importing Certificates in Java Keystore for any REN Java Clients....566
Configuring the REN Server....567
Configuring REN Clusters....567
Installing Certificates for Local Node....567
Generating the Client Certificate....568

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Installing PSMCAPI Certificates..... 569
Configuring External Keystore in REN Server..... 570
Appendix C: PSRENCONFIG Quick Reference.....573
PSRENCONFIG Parameters.....573

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Preface

Understanding the PeopleSoft Online Help and PeopleBooks

The PeopleSoft Online Help is a website that enables you to view all help content for PeopleSoft Applications and PeopleTools. The help provides standard navigation and full-text searching, as well as context-sensitive online help for PeopleSoft users.

PeopleSoft Hosted Documentation

You access the PeopleSoft Online Help on Oracle's PeopleSoft Hosted Documentation website, which enables you to access the full help website and context-sensitive help directly from an Oracle hosted server. The hosted documentation is updated on a regular schedule, ensuring that you have access to the most current documentation. This reduces the need to view separate documentation posts for application maintenance on My Oracle Support, because that documentation is now incorporated into the hosted website content. The Hosted Documentation website is available in English only.

Note: Only the most current release of hosted documentation is updated regularly. After a new release is posted, previous releases remain available but are no longer updated.

Locally Installed Help

If your organization has firewall restrictions that prevent you from using the Hosted Documentation website, you can install the PeopleSoft Online Help locally. If you install the help locally, you have more control over which documents users can access and you can include links to your organization's custom documentation on help pages.

In addition, if you locally install the PeopleSoft Online Help, you can use any search engine for full-text searching. Your installation documentation includes instructions about how to set up Oracle Secure Enterprise Search for full-text searching.

See *PeopleTools Installation* for your database platform, "Installing PeopleSoft Online Help." If you do not use Secure Enterprise Search, see the documentation for your chosen search engine.

Note: Before users can access the search engine on a locally installed help website, you must enable the Search portlet and link. Click the Help link on any page in the PeopleSoft Online Help for instructions.

Downloadable PeopleBook PDF Files

You can access downloadable PDF versions of the help content in the traditional PeopleBook format. The content in the PeopleBook PDFs is the same as the content in the PeopleSoft Online Help, but it has a different structure and it does not include the interactive navigation features that are available in the online help.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Common Help Documentation

Common help documentation contains information that applies to multiple applications. The two main types of common help are:

- Application Fundamentals
- Using PeopleSoft Applications

Most product families provide a set of application fundamentals help topics that discuss essential information about the setup and design of your system. This information applies to many or all applications in the PeopleSoft product family. Whether you are implementing a single application, some combination of applications within the product family, or the entire product family, you should be familiar with the contents of the appropriate application fundamentals help. They provide the starting points for fundamental implementation tasks.

In addition, the *PeopleTools: Applications User's Guide* introduces you to the various elements of the PeopleSoft Pure Internet Architecture. It also explains how to use the navigational hierarchy, components, and pages to perform basic functions as you navigate through the system. While your application or implementation may differ, the topics in this user's guide provide general information about using PeopleSoft Applications.

Field and Control Definitions

PeopleSoft documentation includes definitions for most fields and controls that appear on application pages. These definitions describe how to use a field or control, where populated values come from, the effects of selecting certain values, and so on. If a field or control is not defined, then it either requires no additional explanation or is documented in a common elements section earlier in the documentation. For example, the Date field rarely requires additional explanation and may not be defined in the documentation for some pages.

Typographical Conventions

The following table describes the typographical conventions that are used in the online help.

Typographical Convention	***Description***
Key+Key	Indicates a key combination action. For example, a plus sign (+) between keys means that you must hold down the first key while you press the second key. For Alt+W, hold down the Alt key while you press the W key.
. . . (ellipses)	Indicate that the preceding item or series can be repeated any number of times in PeopleCode syntax.
{ } (curly braces)	Indicate a choice between two options in PeopleCode syntax. Options are separated by a pipe (|).
[ ] (square brackets)	Indicate optional items in PeopleCode syntax.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Typographical Convention	*Description*
& (ampersand)	When placed before a parameter in PeopleCode syntax, an ampersand indicates that the parameter is an already instantiated object. Ampersands also precede all PeopleCode variables.
⇒	This continuation character has been inserted at the end of a line of code that has been wrapped at the page margin. The code should be viewed or entered as a single, continuous line of code without the continuation character.

ISO Country and Currency Codes

PeopleSoft Online Help topics use International Organization for Standardization (ISO) country and currency codes to identify country-specific information and monetary amounts.

ISO country codes may appear as country identifiers, and ISO currency codes may appear as currency identifiers in your PeopleSoft documentation. Reference to an ISO country code in your documentation does not imply that your application includes every ISO country code. The following example is a country-specific heading: "(FRA) Hiring an Employee."

The PeopleSoft Currency Code table (CURRENCY_CD_TBL) contains sample currency code data. The Currency Code table is based on ISO Standard 4217, "Codes for the representation of currencies," and also relies on ISO country codes in the Country table (COUNTRY_TBL). The navigation to the pages where you maintain currency code and country information depends on which PeopleSoft applications you are using. To access the pages for maintaining the Currency Code and Country tables, consult the online help for your applications for more information.

Region and Industry Identifiers

Information that applies only to a specific region or industry is preceded by a standard identifier in parentheses. This identifier typically appears at the beginning of a section heading, but it may also appear at the beginning of a note or other text.

Example of a region-specific heading: "(Latin America) Setting Up Depreciation"

Region Identifiers

Regions are identified by the region name. The following region identifiers may appear in the PeopleSoft Online Help:

- Asia Pacific
- Europe
- Latin America
- North America

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Industry Identifiers

Industries are identified by the industry name or by an abbreviation for that industry. The following industry identifiers may appear in the PeopleSoft Online Help:

- USF (U.S. Federal)
- E&G (Education and Government)

Using and Managing the PeopleSoft Online Help

Click the Help link in the universal navigation header of any page in the PeopleSoft Online Help to see information on the following topics:

- What's new in the PeopleSoft Online Help.
- PeopleSoft Online Help acessibility.
- Accessing, navigating, and searching the PeopleSoft Online Help.
- Managing a locally installed PeopleSoft Online Help website.

Understanding PeopleSoft MultiChannel Framework

PeopleSoft MultiChannel Framework delivers an integrated infrastructure to support multiple interaction channels for call center agents or other PeopleSoft users who must respond to incoming requests and notifications on these channels.

PeopleSoft MultiChannel Framework supports voice, email, web-based chat, instant messaging, and generic event channels.

PeopleTools Related Links

Oracle's PeopleSoft PeopleTools 8.54 Documentation Home Page (Doc ID 1664613.1)

"PeopleTools Product/Feature PeopleBook Index" (PeopleTools 8.54: Getting Started with PeopleTools)

PeopleSoft Information Portal

My Oracle Support

PeopleSoft Training from Oracle University

PeopleSoft Video Feature Overviews on YouTube

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Contact Us

Send us your suggestions Please include release numbers for the PeopleTools and applications that you are using.

Follow Us

Get the latest PeopleSoft updates on Facebook.

Follow PeopleSoft on Twitter@PeopleSoft_Info.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 1

Getting Started with PeopleSoft MultiChannel Framework

PeopleSoft MultiChannel Framework Overview

PeopleSoft MultiChannel Framework (MCF) provides the tools that are required to support multiple channels of communication between customers (users) and agents. Some PeopleSoft applications, such as an email response management system (ERMS) from PeopleSoft CRM, use PeopleSoft MultiChannel Framework or you can develop your own applications on the framework that is provided.

PeopleSoft MultiChannel Framework delivers an integrated infrastructure to support multiple interaction channels for call center agents and other PeopleSoft users who must respond to incoming requests and notifications on these channels. PeopleSoft MultiChannel Framework supports email, web-based chat, voice, instant messaging, and generic event channels. It can be used from any PeopleSoft application.

PeopleSoft MultiChannel Framework also supports a broadcast function, which allows a user, such as a supervisor, to broadcast a notification message to a group of agents.

PeopleSoft MultiChannel Framework enables third-party routing systems that enable applications, such as PeopleSoft CRM, to provide embedded multichannel functionality. You may choose either the PeopleSoft queue server or the third-party routing server to route voice, email, chat, and generic events.

In the planning phase of your implementation, take advantage of all PeopleSoft sources of information, including the installation guides, PeopleTools documentation, and the PeopleBooks that are specific to your applications.

PeopleSoft MultiChannel Framework Implementation

This section describes the required steps for implementing PeopleSoft MCF.

Depending on your business use of PeopleSoft MCF, several activities are necessary for implementation:

- At a minimum, you must configure a real-time event notification (REN) server.
- If you are using a PeopleSoft-supplied application, such as ERMS, you must also configure MCF servers, clusters, queues, and agents.
- You can develop your own applications built on the PeopleSoft MCF.
- If you are using PeopleSoft CTI or other third-party MCF integrations, additional configuration is required.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Configuring REN Servers and Clusters

The REN server routes event notifications through the PeopleSoft MultiChannel Framework. Certain PeopleSoft applications, such as Reporting and Optimization, use the REN server to route notifications without using any of the rest of the PeopleSoft MultiChannel Framework. Therefore, a minimal configuration of PeopleSoft MultiChannel Framework includes at least one REN server and REN server cluster.

REN server-specific security setup is required, including configuration of Real-time Event Notification Permissions for each role using a REN server alone or as part of PeopleSoft MultiChannel Framework.

To provide a secure channel of communication between the clients and the REN servers, the REN servers may be SSL-enabled. SSL-enabled REN servers enable secure communication by providing client and server authentication.

Typically, a system administrator configures and manages REN servers and clusters.

See Understanding REN Servers.

Configuring MCF Servers, Clusters, Queues, and Tasks

MCF servers (queue servers and log servers) work with REN servers to queue and route task notifications to agents. An implementation of PeopleSoft MultiChannel Framework requires configuration of MCF servers, MCF clusters, queues, and tasks.

Typically, a system administrator configures and manages MCF servers and clusters, and either a system administrator or an agent supervisor configures and manages queues and tasks.

See Understanding PeopleSoft MCF Server and Cluster Architecture.

Creating and Defining Agents

Agents are PeopleSoft users who are further defined as PeopleSoft MultiChannel Framework agents. Agents manage tasks that are assigned to them. Agents can log on to and accept tasks from the MultiChannel Console.

Typically, an agent supervisor defines agents and assigns agents to appropriate queues.

See Defining Agents.

Using MCF Sample Pages

PeopleSoft MultiChannel Framework is delivered with sample pages that demonstrate the use of email, web-based chat, generic, and instant messaging channels. An application developer can use these sample page definitions as the basis for application pages. PeopleCode API Reference includes detailed descriptions of PeopleSoft's Mail Classes, MCFIMInfo Classes, and the Universal Queue Classes.

Additional sample pages demonstrate how to use the JavaScript MultiChannel Application Programming Interface (JSMCAPI) to customize the CTI console and to develop supervisor desktops using the available monitoring functions.

See Working with Sample Pages.

See "Understanding PeopleSoft MultiChannel Framework Mail Classes" (PeopleTools 8.54: PeopleCode API Reference) and"Understanding Universal Queue Classes" (PeopleTools 8.54: PeopleCode API Reference).

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Configuring PeopleSoft CTI

PeopleSoft MultiChannel Framework includes support for PeopleSoft CTI. PeopleSoft CTI requires supporting computer-telephony middleware and additional configuration separate from other channels of PeopleSoft MultiChannel Framework.

See Understanding the PeopleSoft CTI Console.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 2

Understanding PeopleSoft MultiChannel Framework

PeopleSoft MultiChannel Framework

PeopleSoft MultiChannel Framework delivers an integrated infrastructure to support multiple interaction channels for call center agents or other PeopleSoft users who must respond to incoming requests and notifications on these channels.

PeopleSoft MultiChannel Framework can be used from any PeopleSoft application.

In this context, the word *channel* refers to the technology used to communicate during an interaction. PeopleSoft MultiChannel Framework supports the following channels:

- Voice (telephone).
- Web collaboration (chat).
- Email.
- Generic tasks.
- Instant messaging.

The PeopleSoft MultiChannel Framework includes an HTML agent console, universal queueing, real-time task routing, customer-to-agent and collaborative chat, and centralized event logging.

PeopleSoft MultiChannel Framework Elements and Channels

This section discusses:

- PeopleSoft MultiChannel Framework elements.
- PeopleSoft MultiChannel Framework channels.
- PeopleSoft MultiChannel Framework universal queue.
- PeopleSoft MultiChannel Console.

PeopleSoft MultiChannel Framework Elements

PeopleSoft MultiChannel Framework comprises the following services and elements:

- Universal queue server, running on the Universal Queue server process (PSUQSRV).

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Real-time event notification (REN) server, running on the REN server process (PSRENSRV).
- MultiChannel Framework (MCF) log server, running on the MCFLOG server process (PSMCFLOG).
- MultiChannel console, the HTML interface through which users manage the channel interactions assigned to them.
- Chat windows, the HTML interfaces used for customer-to-agent and collaborative chat sessions.
- Agents, identified by their expertise and responsibilities.
- PeopleCode built-in functions and an email application package.
- GETMAILTARGET connector running under PeopleSoft Integration Broker.
- PeopleSoft MultiChannel Application Programming Interface (PSMCAPI) to enable server-side computer-telephony integration (CTI) integration.
- JavaScript MultiChannel Application Programming Interface (JSMCAPI) to enable a customizable CTI console and monitoring functions.

Each of these services and elements requires configuration.

In addition, each communication channel handled by PeopleSoft MultiChannel Framework requires supporting elements:

- CTI middleware to notify the system of telephone calls.
- An email server to store and serve email.
- Application pages to request customer-to-agent chat sessions and to provide context data and resolution logic for all interactions.
- Application pages or batch processes to enqueue generic events.

PeopleSoft MultiChannel Framework Channels

This section discusses support for communications channels offered by PeopleSoft MCF.

Voice

The agent console offers a softphone and full CTI support with Oracle-validated third-party CTI systems. Relevant application pages appear based on data attached to the call by the Interactive Voice Response (IVR) and CTI middleware.

Web Collaboration

PeopleSoft application pages can include Live Help buttons that initiate customer-to-agent chat sessions. The customer and agent chat windows are browser-based and do not require a client installation or applet download. The universal queue routes chat requests to the first available agent with the skills required to handle that request. The agent chat window displays relevant customer information and enables the agent to push web content to the customer. The agent can manage multiple chat sessions from the agent console.

Agents can also include peers and supervisors in chat conferences and transfer chat sessions to other agents or queues. Agents can also initiate collaborative chats with other agents on their buddy lists.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Email

PeopleSoft MCF enables applications to fetch Multipart Internet Mail Extensions emails from Post Office Protocol 3 (POP3) and Internet Message Access Protocol 4 (IMAP4) mail servers, store their parts in a database, and route the email to call center agents by either adding the email to worklists or enqueueing them on the universal queue. Email attachments can be stored in a database or stored in an attachment repository. Email attachments that are stored in the database or in the attachment repository are accessible by URLs from a browser. The repository and the database check user-based and role-based security before retrieving an attachment. The email framework is built on PeopleSoft Integration Broker technology.

PeopleSoft MCF does not provide a mechanism to move existing email attachments that are stored in the attachment repository to the database.

When you choose to store email attachments in the database, and if an attachment cannot be stored in the database due to the unavailability of a connection, an error is displayed. In such cases, you must attempt to save the attachment again.

Note: If you choose to store email attachments in the database, ensure that you configure the default local node in Integration Broker.

PeopleSoft MCF supports emails conforming to the Simple Mail Transfer Protocol (SMTP) specifications including both inbound and outbound HTML email.

Generic Channel

Channels that are not provided by PeopleSoft MCF can be integrated by means of the generic channel to enqueue tasks onto the universal queue.

Instant Messaging

PeopleSoft MCF enables you to use instant messaging called from an application page by using one of several instant messaging clients, such as AOL, Yahoo, or Sametime.

PeopleSoft MultiChannel Framework Universal Queue

The universal queue accepts, evaluates, and distributes incoming task requests from multiple communication channels: email, web chat, and generic notifications.

The universal queue handles email, chat, and generic tasks. It distributes workload across the call center, or any other pool of qualified users, based on the priority of the task and the availability of agents possessing the required skill level and language skills. Availability is based on agent presence and the cost of the new task (a measure of the task's impact on agent capacity) against the current workload of each agent.

Agents can forward tasks to other agents or to another queue. The task is removed from the transferring agent's workload and added to the accepting agent's workload.

Email and generic tasks that are not closed before the agent signs out persist in the database. Persisted tasks are reassigned to the same agent that accepted the tasks when the agent signs in again. A task that is not accepted, within configurable time limits, by the agent to whom it was assigned is reassigned to another qualified agent, if one is available. Tasks that are not resolved within configurable time limits

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

are automatically escalated. Tasks that cannot be assigned to or are not accepted by any agent within configurable time limits are moved to an overflow table.

Voice tasks (CTI) are not queued or routed by the universal queue. They take precedence over all other tasks. However, the queue server adds the cost of voice tasks to the agent workload calculations it uses to queue and assign incoming tasks.

PeopleSoft MultiChannel Console

The agent console is the web-browser-based desktop from which the user manages all tasks, irrespective of channel. The console combines CTI, chat, email, and generic notice response tools into one configurable window. Agents use the console to sign in, to select their current queue, to accept tasks, and to initiate and accept collaborative chat requests with buddy users. After the agent accepts a task, additional browser-based windows appear to enable the agent's response. These windows are task-dependent and include elements developed specifically for supporting applications, such as email response management systems.

Note: The Multichannel Console link is available in the universal navigation header on PeopleSoft Pure Internet Architecture (PIA) and the visibility of the link is controlled by the PTPT4700 permission list and the PeopleTools MCF Console role. Only users assigned with this permission list and role can view the Multichannel Console link on PIA. The exception to this rule are the PeopleTools Administrator and Portal Administrator users.
To access the link and use its features, you must have the security permissions as required for the respective functions.

A custom CTI console can be created by means of the JSMCAPI.

PeopleSoft MCF Architecture

This section discusses:

- PeopleSoft MCF server architecture.
- Chat architecture.
- Email architecture.

PeopleSoft MCF Server Architecture

The REN server (the PSRENSRV process) is essential to the framework architecture. MCF events are sent to REN servers, which then deliver them to recipients of those topics. The REN server is a modified web server using the HTTP 1.0 or HTTP 1.1 communications protocol. Communication with server processes and MCF browser windows is bidirectional, because the browser windows maintain persistent connections to the REN server. Events can be sent proactively to browser windows without polling or page refreshes. To provide a secure channel of communication to overcome concerns from PeopleSoft customers about sensitive data and its security, the REN server can be Secure Sockets Layer (SSL)-enabled. An SSL-enabled REN server provides secure communication that encrypts and provides client and server authentication.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Applications send interaction and action requests (tasks) received from the supported communication channels to logical queues. The REN server notifies the universal queue server (the PSUQSRV process) responsible for that queue that a new task has arrived. Tasks are queued in order of priority until they can be assigned to an available agent qualified to respond to the task, at which time the queue server sends an assignment notification to the user's MultiChannel Console through the REN server.

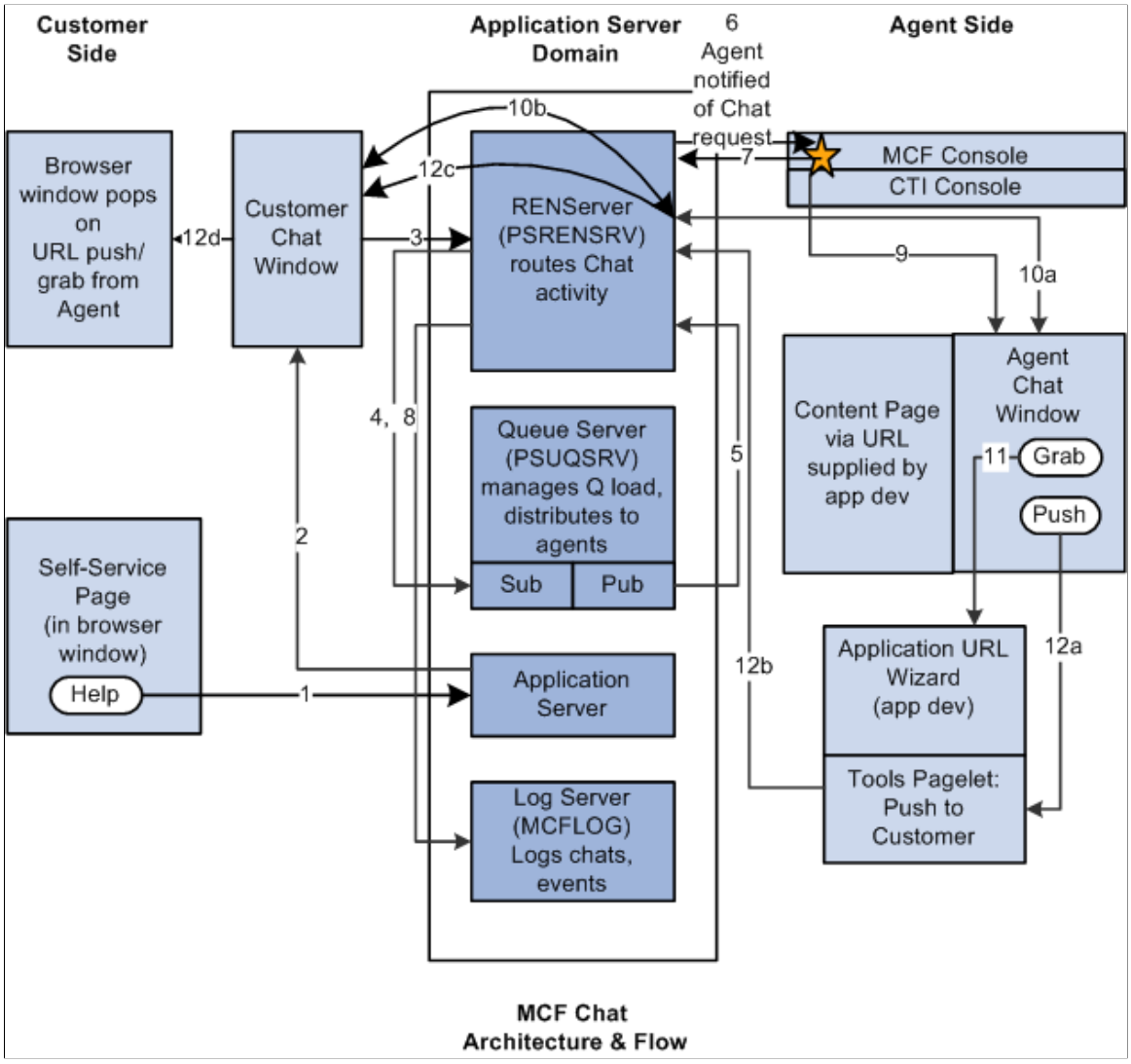
The queue server routes work requests (tasks) to users based upon a set of configurable policy properties that define which agents can handle what types of tasks and when the tasks can be assigned. The queue server manages state information about the current status of active agents and active tasks.

The MCF log server logs MCF events and chat content to the database. You configure logging levels on the MCF administration pages.

Chat Architecture

Image: MCF chat architecture and flow

The following diagram illustrates the architecture of MCF chat and the flow of a chat session.



When a customer clicks the Help button on an application page:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

1. InitChat() passes the following parameters to the application server:
   - Queue number.
   - Priority override.
   - Context page URL.
   - Query.

     See "InitChat" (PeopleTools 8.54: PeopleCode Language Reference).

2. The application server posts to the REN server to notify the queue server that a customer chat is waiting.

   The application server then returns the name and port of the REN server and the iScript to build the customer chat window.

3. The customer chat window appears and communicates with the REN server to receive all events on that chat topic.

4. The REN server notifies the queue server that a customer chat is waiting.

   The queue server determines the appropriate agent according to workload, cost, agent availability, skill level, and language.

5. The queue server tells the REN server to notify the agent that the agent has been assigned a chat.

6. The REN server notifies the agent from the MultiChannel Console that the agent has been assigned a chat.

7. The agent, from the MultiChannel Console, notifies the REN server to notify the queue server that the agent has accepted the task.

8. The REN server notifies the queue that the agent has accepted the task.

9. The MultiChannel Console displays an agent chat window.

   The agent responds to inquiry.

10. Agent to customer two-way communication is mediated by the REN server.

11. If the agent chooses to grab a URL, the Grab button invokes the application URL wizard.

12. If the agent chooses to push a URL, the Push button invokes a pagelet (labeled with an *A* in the diagram) to push a selected URL through the REN server (labeled with a *B* in the diagram) and customer chat window (labeled with a *C* in the diagram) to a new browser window.

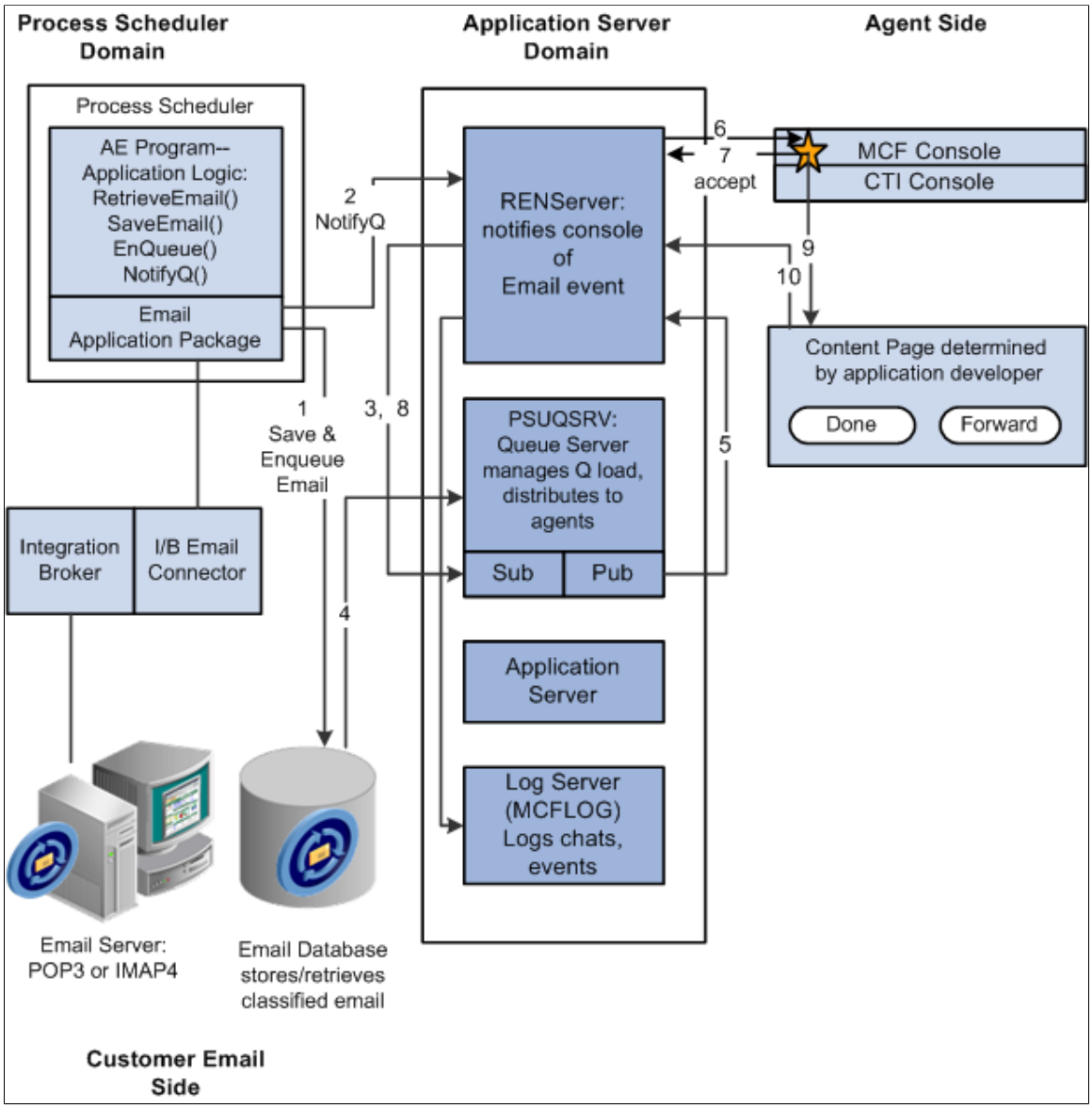
The entire chat session can be logged through the MCF log server (labeled with a dotted line in the preceding diagram).

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Email Architecture

Image: MCF email architecture and flow

This diagram illustrates the architecture of MCF email and the flow of email processing.



When a customer sends an email:

1. A PeopleSoft Application Engine program uses the MCF email application package classes and PeopleCode built-in functions to save and enqueue email in a database.
2. The PeopleSoft Application Engine program notifies the REN server that email has been enqueued.
3. The REN server notifies the queue server of the waiting email.
4. The queue server retrieves email information required to determine appropriate routing from the database.

   The queue server determines the appropriate agent to handle each email according to workload, cost, agent availability, skill level, and language.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

5. The queue server tells the REN server to notify the agent that the agent has been assigned an email.
6. The REN server notifies the agent from the MultiChannel Console that the agent has been assigned an email.
7. The agent, from the MultiChannel Console, notifies the REN server to notify the queue server that the agent has accepted the task.
8. The REN server notifies the queue that the agent has accepted the task.
9. The MultiChannel Console displays an agent email window, as determined by the application developer.

   The agent responds to the email.
10. The agent's resolution of the task is communicated back to the REN server by either the Done or Forward button.

Email events can be logged to a database by the MCF log server.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 3

Configuring PeopleSoft Computer Telephony Integration

Understanding the PeopleSoft CTI Console

This section discusses:

- PeopleSoft CTI.
- PeopleSoft CTI components.

Important! The CTI Console Java applet is no longer supported in PeopleTools 8.53. Refer to the Product Certifications link on My Oracle Support, https://support.oracle.com (sign-in required), for more information about supported products.

PeopleSoft CTI

PeopleSoft CTI enables you to integrate your PeopleSoft applications with your call center. PeopleSoft CTI offers the following benefits:

- Seamlessly integrates your PeopleSoft application with Oracle-validated third-party CTI systems to improve agent productivity.

  Agents refers to the individuals who interact with your customers using CTI.
- Requires only that you install a supported web browser on the agent's workstation.
- Enables agents to take advantage of browser-based call management and automatic population of PeopleSoft transaction pages with the relevant customer data associated with an incoming call.
- Transmits DTMF data.
- Handles outbound calls from automated systems.

PeopleSoft CTI is an optional component that you can integrate with the PeopleSoft MultiChannel Framework. This means that you can incorporate a CTI channel within the MultiChannel Framework. PeopleSoft CTI requires third-party middleware in the form of an Oracle-validated third-party CTI middleware.

In PeopleSoft CTI, the CTI middleware performs the call routing. The universal queue is not involved in routing calls. For an incoming call, the CTI middleware notifies the MultiChannel Console, which then notifies the queue server so that the agent's workload can be updated with the cost of a call.

For vendors using the adapter-based CTI solution, refer to Oracle Validated Application Integrations — Find a Partner Solution (http:www.oracle.com/partnerships/isv/integration/search.html)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

For vendors using the applet-based CTI solution, sign in to My Oracle Support and select Product Certifications tab.

PeopleSoft CTI Components

The PeopleSoft CTI Console works together with the IVR (Interactive Voice Response) system, the CTI middleware, an Automatic Call Distributor (ACD), and your PeopleSoft application.

When a customer calls, the caller enters his or her information (for example, an account number) using the IVR system. Using this information, the CTI middleware routes the call to an appropriate ACD queue. The ACD sends the call to the next available agent on that queue and notifies the CTI middleware that an incoming call is on that Directory Number (DN). The CTI middleware, in turn, notifies the CTI Console and passes the customer's information as attached data. The CTI Console uses the attached data to determine what PeopleSoft transaction page to open (pop-up) for the agent and what application data to retrieve from the database. The agent can manage the call using the CTI Console, which in turn communicates with the PBX (Private Branch Exchange) using the CTI middleware.

You can configure PeopleSoft CTI systems using either the adapter solution or the applet solution. These terms are defined in the following table.

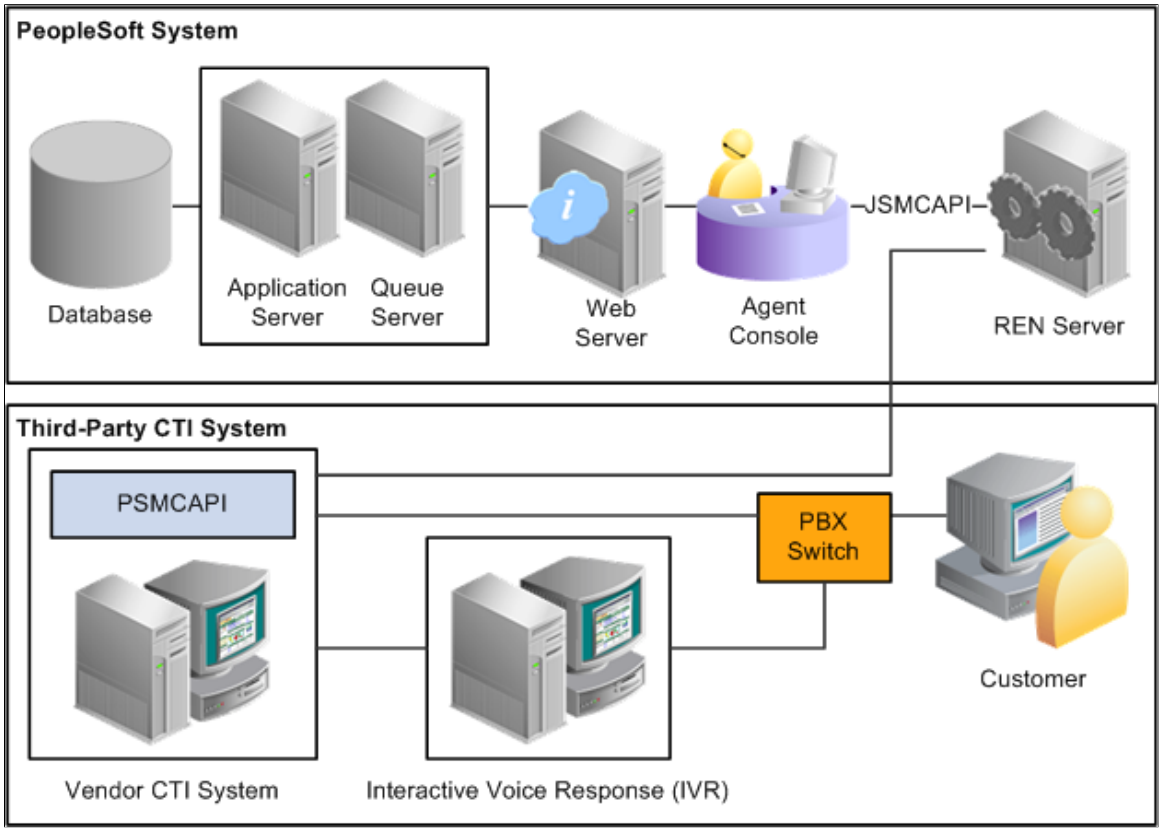
Adapter	For PeopleSoft CTI systems using PeopleTools 8.45.05 or later, the CTI Console uses a JavaScript MultiChannel Application Programming Interface (JSMCAPI) to communicate with CTI middleware that implements the PeopleSoft MultiChannel Application Programming Interface (PSMCAPI).
Applet	The CTI Console uses a Java applet that runs within the web browser to communicate directly with your CTI middleware. The CTI Console communicates to the Java applet using JavaScript. The applet is delivered in a file called pCti.cab, which is approximately 500 KB. The applet resides in the browser cache to reduce network traffic and improve response time.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Configure PeopleSoft CTI to use either the applet or the CTI console (JSMCAPI) on the CTI Type tab on the Configure CTI page.

Image: PeopleSoft CTI integration architecture

This diagram illustrates PeopleSoft CTI architecture that uses PSMCAPI and JSMCAPI.



Related Links

"Understanding Internet Script Classes" (PeopleTools 8.54: PeopleCode API Reference)

Adapter-Based CTI Requirements

This section describes components required for PeopleSoft CTI using the adapter-based solution. It discusses:

- PeopleSoft MultiChannel API (application program interface)
- Required Security for PSMCAPI
- JavaScript MultiChannel API

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleSoft MultiChannel API

The PeopleSoft MultiChannel API (PSMCAPI) is a Java API and software development kit (SDK) that provides server-side connectivity with the PeopleSoft CTI system. PSMCAPI enables third-party telephony vendors and system integrators to integrate with PeopleSoft applications.

PSMCAPI and JSMCAPI are installed during PeopleTools installation.

This table provides the installed locations of PSMCAPI and JSMCAPI components:

Component	*Location*
PSMCAPI Java archive	<PS_HOME>\sdk\psmcapi\dist\lib
PSMCAPI configuration files: • psmcapilog.properties • renclient.properties	<PS_HOME>\sdk\psmcapi\dist\config
JSMCAPI JavaScript file	<PIA_HOME>\webserv\<domain>\applications\peoplesoft\PORTAL\ps\pMCF

Working With Older Adapters

For some implementations using older versions of third party adapters, you may require a version of psmcapi.jar built with an older version of JDK. PeopleTools provides a previous version of the PSMCAPI for use with older adapters.

This table provides the installed locations of the PSMCAPI components provided for use with older adapters:

Component	*Location*
PSMCAPI Java Archive	PS_HOME\sdk\psmcapi\dist\archive\lib
PSMCAPI configuration files	PS_HOME\sdk\psmcapi\dist\archive\config

Configuring PSMCAPI

Two files, psmcapilog.properties and renclient.properties, include parameters to configure PSMCAPI. Configure the logging characteristics of PSMCAPI in psmcapi.properties. To have PSMCAPI generate full debug logs, set com.peoplesoft.pt.mcf.level to FINEST.

The following table lists all the parameters in renclient.properties:

Parameter	*Default Value*	*Description*
interval_heartbeat_server	30000	Heartbeat interval in milliseconds from PSMCAPI to console.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameter	***Default Value***	***Description***
interval_heartbeat_client	30000	Heartbeat interval in milliseconds from console to PSMCAPI.
maxsize_eventqueue	100000	Maximum number of events in the event queue.
interval_event_expired	300000	Expiration interval in milliseconds for an event in the event queue. An expired event is never published to the client/ console and is discarded from the event queue.
interval_topic_reaper	3600000	Topic reaper interval in milliseconds.
number_of_requests_in_requestschunk	1	The number of requests that JSMCAPI sends at a time during auto-recovery.
waitingtime_between_requestchunks	1	The time elapsed between two consecutive JSMCAPI request queues during auto-recovery.
mtu_size	0	The Maximum Transmission Unit size of your computer or network. Set mtu_size to 0 (zero) for no TCP packet padding, or to the maximum transmission unit (MTU) size for your network or computer to remove TCP acknowledgement delays.
heartbeats_to_miss	2	The number of heartbeat intervals to wait before removing a nonresponsive client
psmcapi_heartbeats_to_miss	5	The number of PSMCAPI heartbeat intervals to wait before removing a nonresponsive client.
tcp_nodelay	True	TCP no delay. Set to True to disable the TCP Nagle algorithm.
disable_session	False	This parameter is used when multiple PSMCAPI implementations subscribe to the same REN server. Set to True to disable the internal session and heartbeat listeners.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Required Security for PSMCAPI

CTI agents require only that the MCF Agent security object be enabled on the REN Permissions page of their associated permission list. The MCF Agent security object includes security for both CTI and other MCF channels.

To enable CTI configuration pages, authorize the CTI Type, CTI Applet, and CTI Console panel items on the PT_CTI page permissions for the appropriate permission list.

Also enable WEBLIB_MCF weblib permissions for the appropriate permission list.

Related Links

Configuring REN Servers

"Security Administration Overview" (PeopleTools 8.54: Security Administration)

JavaScript MultiChannel API

The JavaScript MultiChannel Application Programming Interface (JSMCAPI) is an interface that application developers can use to generate the CTI Console or to enable CTI functionality on a PeopleSoft Pure Internet Architecture page. The JSMCAPI builds on the real-time event notification (REN) JavaScript client. JSMCAPI uses standard JavaScript.

Related Links

Using and Demonstrating JSMCAPI

Applet-Based CTI Requirements

PeopleSoft CTI using the adapter based solution uses:

- Java Applet Console.
- Java Runtime Environment (JRE).

Important! The CTI Console Java applet is no longer supported in PeopleTools 8.53. Refer to the Product Certifications link on My Oracle Support, https://support.oracle.com (sign-in required), for more information about supported products.

Java Applet Console

This section lists some considerations for:

- Genesys components
- Cisco components

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Genesys Components

If you are using a Genesys CTI system with the PeopleSoft CTI Java applet console, the assumption is that you already have a functioning Genesys system configured at your site. Genesys Java API is shipped as part of the CTI applet; however, no other Genesys products are shipped with the PeopleSoft system.

To interact with the PeopleSoft CTI system, you need a Genesys T-Server installed and configured before you begin installing your PeopleSoft CTI system. Refer to your Genesys documentation for installation information.

In addition, your PeopleSoft application may require the following CTI components to attach data to incoming calls:

- Genesys Strategy Builder or Interaction Router or equivalent.
- An IVR supported by Genesys and capable of passing call data to Genesys.

For detailed information regarding specific versions that Oracle supports, refer to the Product Certifications on the My Oracle Support web-site.

Cisco Components

If you are using a Cisco CTI system with the PeopleSoft CTI Java applet console, the assumption is that you already have a functioning and configured Cisco system at your site. The Cisco Java API is shipped as part of the CTI applet; however, no other Cisco products are shipped with the PeopleSoft system.

You need a Cisco ICM Central Controller Server installed and configured before you begin installing your PeopleSoft CTI system.

In addition, your PeopleSoft application may require an IVR supported by Cisco and capable of passing call data to Cisco. This enables CTI components to attach data to incoming calls.

Note: For detailed information regarding specific versions that Oracle supports, refer to the Product Certifications on the My Oracle Support web-site.

Java Runtime Environment

Java applets, such as the PeopleSoft CTI Java applet console, run within a Java Virtual Machine (JVM) that uses a *sandbox* (a controlled environment in which remote programs run) to restrict access to what the applets can do. The JRE is required for those PeopleSoft CTI systems using the Java applet console. The JRE is not required if you are using the JSMCAPI console.

If you are using the Java applet console, you should install and use the Sun Java Plug-in 1.4.2 or later. You can download the plug-in from Sun Microsystems at www.java.com.

Applet Protection

The CTI applet has an attached digital signature, which enables the PeopleSoft CTI applet to open a network connection to your CTI system as opposed to being able to send back data only to the web server that downloaded it.

The digital signature protects the applet against tampering, and it requires that the agent grant permission to run the applet. When doing so, agents need to indicate whether such permission should be granted to

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

any PeopleSoft code for all subsequent sessions or for the current session only. Instruct your end users to select the appropriate option for your site.

For more information, see Supported web browsers for PeopleSoft PeopleTools Certifications on the My Oracle Support web site.

Configuring PeopleSoft CTI

This section describes how to install and configure the PeopleSoft CTI system. This information assumes that you already have a functioning CTI system installed and configured at your site. The information in this section is common to both the adapter and applet solutions.

This section discusses how to:

- Install PeopleSoft CTI.
- Enable PeopleSoft CTI.
- Create CTI configurations.
- Use the Shared Phone Book page.
- Use the Miscellaneous page.
- Use the Reason Code page.

Installing PeopleSoft CTI

When you run the PeopleSoft Pure Internet Architecture setup program, the PeopleSoft CTI files are installed automatically to your web server.

Note: You do not need to select any additional options from the install program dialogs. The CTI files are installed by default.

After you have run the PeopleSoft Pure Internet Architecture setup program, enable the PeopleSoft CTI Console and configure the system as discussed in the following sections.

See the product documentation for *PeopleTools Installation* for your database platform and *CTI vendor*.

Enabling PeopleSoft CTI

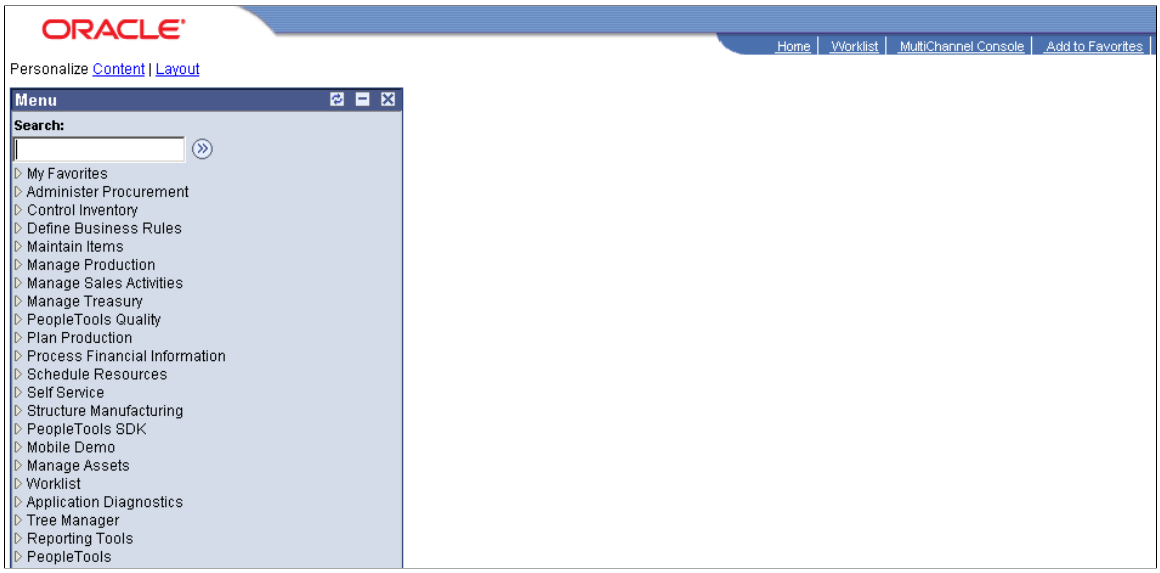
If a user is set up as a CTI agent or a MultiChannel Framework agent and if the user is assigned the PTPT4700 permission list and the PeopleTools MCF Console role, the Multichannel Console link appears in the universal navigation header (in the upper-right portion of the screen) when he or she accesses the

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleSoft Pure Internet Architecture. The following example shows the Multichannel Console link displayed in the PeopleSoft Pure Internet Architecture:

Image: MultiChannel Console link

This example illustrates the MultiChannel Console link displayed in PeopleSoft Pure Internet Architecture.



Users who do not have the PeopleTools MCF Console role do not see this link.

Note: Blended agents—agents who use both the CTI and MCF consoles—must sign in to queues supported by the same REN server cluster.

See Using the CTI Sample Console.

Related Links

Configuring CTI Agents

Configuring CTI Console Type

Access the CTI Type page using either of the following navigation paths, whichever is appropriate to you:

PeopleTools, MultiChannel Framework, CTI, CTI Type (if you are using the CTI server) or PeopleTools, MultiChannel Framework, Third-Party Configuration, CTI Configuration, CTI Type (if you are using a third-party routing server).

Image: CTI Type page

This example illustrates the fields and controls on the CTI Type page.

Select the Is Applet check box if you are using the CTI Java applet console.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Clear the Is Applet check box if you are using the CTI JavaScript (JSMCAPI) console.

A CTI configuration contains all the information required for a user to be able to connect to a CTI server.

Note: The Configuration ID is created when CTI is configured for the first time.

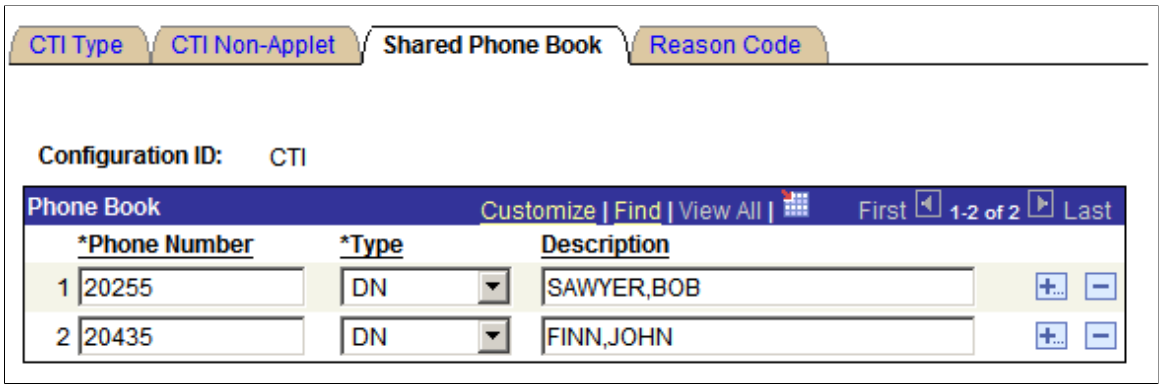
Creating a List of Frequently Dialed Phone Numbers

Access the Shared Phone Book page using either of the following navigation paths, whichever is appropriate to you:

PeopleTools, MultiChannel Framework, CTI, Shared Phone Book (if you are using the CTI server) or PeopleTools, MultiChannel Framework, Third-Party Configuration, CTI Configuration, Shared Phone Book (if you are using a third-party routing server).

Image: Shared Phone Book page

This example illustrates the fields and controls on the Shared Phone Book page. You can find definitions for the fields and controls later on this page.



On this page, you can manage a list of frequently dialed phone numbers for a specific CTI configuration. These numbers appear when an agent connected to that CTI configuration selects the drop-down list box when dialing a number from the CTI Console. This saves the agents from manually entering any dialed numbers present in that CTI configuration when making outbound calls.

Note: Phone lists are updated on the CTI Console only after the CTI Console launches. To refresh phone lists, refresh the browser and reactivate the console.

Phone Number Enter a frequently dialed phone number to associate with this configuration.

Type Select one of the following values:

- *DN* (directory number): The number that identifies a telephone set on a PBX or in the public network. The caller dials this number to establish a connection to the addressed party. The DN can be a local PBX extension (a local DN) or a public network telephone number.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- *Queue* The directory number that identifies an ACD queue or group. Calls to a group are distributed to agents belonging to the group, according to ACD algorithms.

Description Add a description for the telephone number.

Entering Default Screen Pop-Up URL

The default URL provides a way to enable CTI screenpops if data from the voice task should not be used to construct the base of the pop-up URL. If the default URL is populated, then it will be used for the screenpop, even if sufficient user or call data is provided in a voice task to construct a URL.

Note: User and Call data may be used as parameters to the URL.

Access the Miscellaneous page using the following navigation path:

PeopleTools, MultiChannel Framework, CTI, Miscellaneous

Note: This page is available through the CTI server only.

Image: Miscellaneous page

This example illustrates the fields and controls on the Miscellaneous page. You can find definitions for the fields and controls later on this page.

Set default screen pop-up URL parameters on the Miscellaneous page.

Use Default Screen Popup URL Select if you want to set a default URL for the pop-up screen.

The system uses the default method to determine what URL to use for the page launched for incoming calls. The system uses that default URL to examine the user data attached to that call. However, for customers who do not want to attach this user data to incoming calls, this option enables you to use the same URL for all pop-up screens.

After selecting this option, enter the URL in the Default Screen Popup URL edit box.

Default Screen Popup URL If the Use Default Screen Popup URL check box is selected, enter the value for the default URL.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: To ensure that the user does not have to sign in to the pop-up window, the domain of the default URL must exactly match the domain of the sign-in URL. If the sign-in URL has no domain, the machine name of the default URL should be the same as the machine name of the sign-in URL. If either the domain or the machine name do not match, the system prompts the user for the user ID and password for the first pop-up screen.

With Matching Domains:

Within the same domain, such as example.com, the machine names can be different. For example:

Signon URL:

http://ntserver1.example.com/peoplesoft8/signon.html

Default URL:

http://uxserver2.example.com/servlets/iclientservlet/peoplesoft8/?ICType=PANEL&Menu=UTILITIES&Market=GBL&Component=MESSAGE_CATALOG1&Target=Main1&LANGUAGE_CD=ENG&MESSAGE_SET_NBR=1

Without Matching Domains:

Without the matching domains, the machine name should be the same in both URLs. For example:

Signon URL:

http://ntserver1/peoplesoft8/signon.html

Default URL:

http://ntserver1/servlets/iclientservlet/peoplesoft8/?ICType=PANEL&Menu=UTILITIES&Market=GBL&Component=MESSAGE_CATALOG1&Target=Main1&LANGUAGE_CD=ENG&MESSAGE_SET_NBR=1

Related Links

"Understanding Single Signon" (PeopleTools 8.54: Security Administration)

Using the Reason Code Page

Access the Reason Code page using the following navigation path:

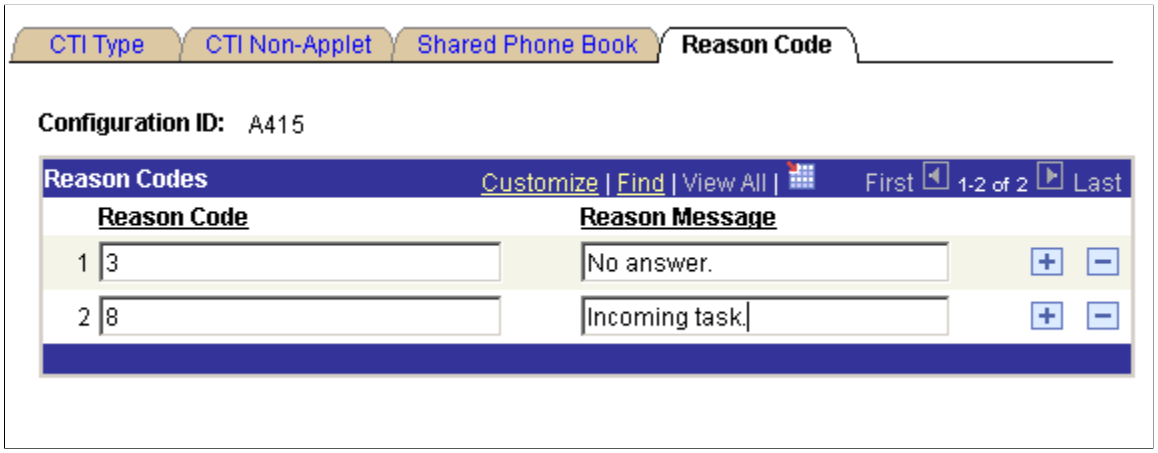
PeopleTools, MultiChannel Framework, Third-Party Configuration, CTI Configuration, Reason Code

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: This page is available only to third-party vendors.

Image: Reason Code page

This example illustrates the fields and controls on the Reason Code page.



This page is used by the third-party vendors to define their customized codes.

Related Links

Understanding JSMCAPI

Configuring PeopleSoft CTI Using Adapters

This section discusses how to configure the CTI console.

Configuring the CTI Console

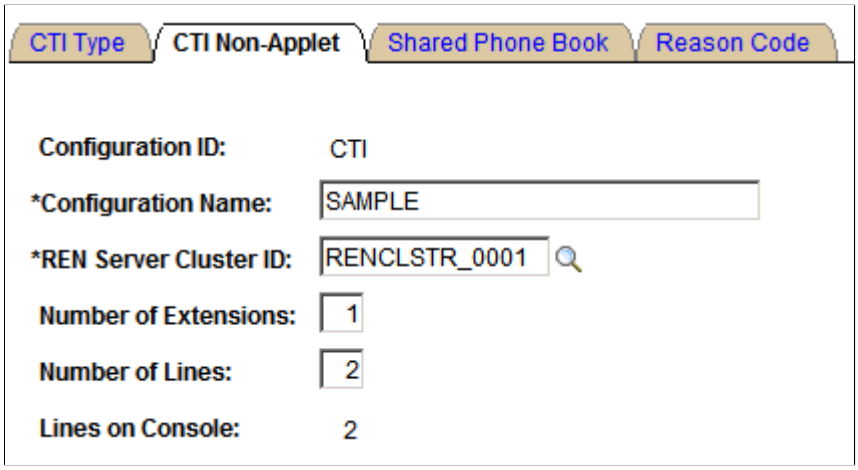
Access the CTI Configuration page using either of the following navigation paths, whichever is appropriate to you, to configure the CTI (JSMCAPI) console:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, CTI, Configuration (if you are using the CTI server) or PeopleTools, MultiChannel Framework, Third-Party Configuration, CTI Configuration, Configuration (if you are using a third-party routing server).

Image: CTI Non-Applet page

This example illustrates the fields and controls on the CTI Non-Applet page. You can find definitions for the fields and controls later on this page.



Configuration ID Displays the name of the CTI configuration. The name cannot be modified after it is created.

Configuration Name Add a descriptive name to help identify the configuration.

Number of Extensions Enter the number of extensions or directory numbers associated with the telephone.

Number of Lines Enter the number of lines associated with each extension. Depending on your configuration, you can enter up to two lines.

Lines on Console The CTI Console supports up to two lines. The only supported configurations are two extensions with one line each, one extension with two lines, and one extension with one line.

Configuring PeopleSoft CTI Using Applets

This section discusses how to:

- Configure the CTI applet console
- Use the CTI Genesys page
- Use the CTI Cisco page

Important! The CTI Console Java applet is no longer supported in PeopleTools 8.53. Refer to the Product Certifications link on My Oracle Support, https://support.oracle.com (sign-in required), for more information about supported products.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

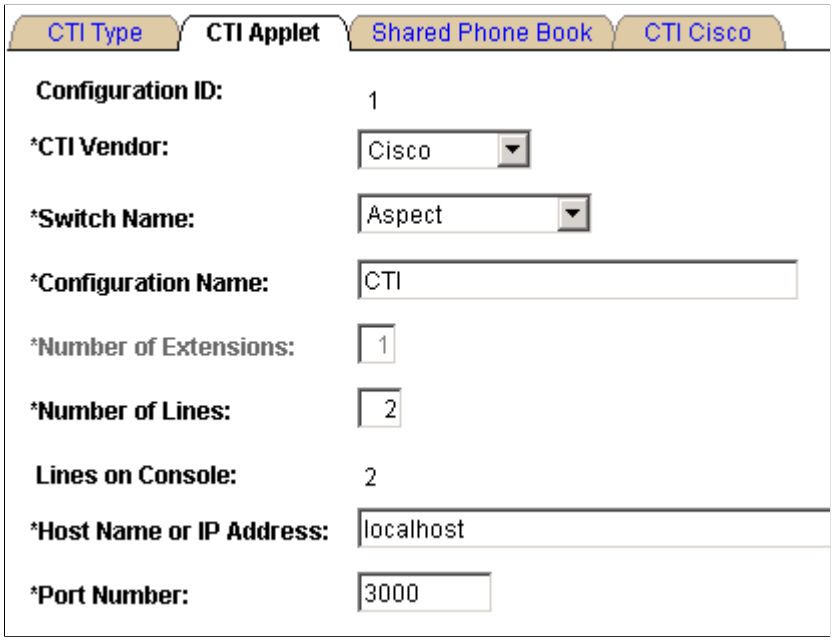
Configuring the CTI Applet Console

Access the CTI Applet page using either of the following navigation paths, whichever is appropriate to you, to configure PeopleSoft CTI for use with the CTI applet console.

PeopleTools, MultiChannel Framework, CTI, CTI Applet orPeopleTools, MultiChannel Framework, Third-Party Configuration, CTI Configuration, CTI Applet.

Image: CTI Applet page

This example illustrates the fields and controls on the CTI Applet page. You can find definitions for the fields and controls later on this page.



Configuration ID

Displays the name that you gave the configuration when you created it. The name cannot be modified after it is created.

CTI Vendor

Select the vendor of your CTI solution. The options are:

- *Genesys*
- *Cisco*

This value controls whether the CTI Cisco page or the CTI Genesys page appears in the component. For example, if you select *Cisco,* the CTI Cisco page appears.

Switch Name

Select from one of the supported switches.

Configuration Name

Add a descriptive name to help identify the configuration.

Number of Extensions

Enter the number of extensions or directory numbers associated with the telephone.

For Genesys, *1* or *2* can be used.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

For Cisco, only *1* can be used.

Number of Lines

Enter the number of lines associated with each extension. Depending on your configuration, you can specify up to two lines.

If Number of Extensions is *1,* enter *2.*

If Number of Extensions is *2,* enter *1.*

Lines on Console

The CTI Console supports up to two lines. The only supported configurations are two extensions with one line each, one extension with two lines, and one extension with one line.

Host Name or IP Address

Enter the host name or IP address for your CTI server.

Port Number

Enter the port number on which the T-Server, Configuration Server, or Cisco ICM listens.

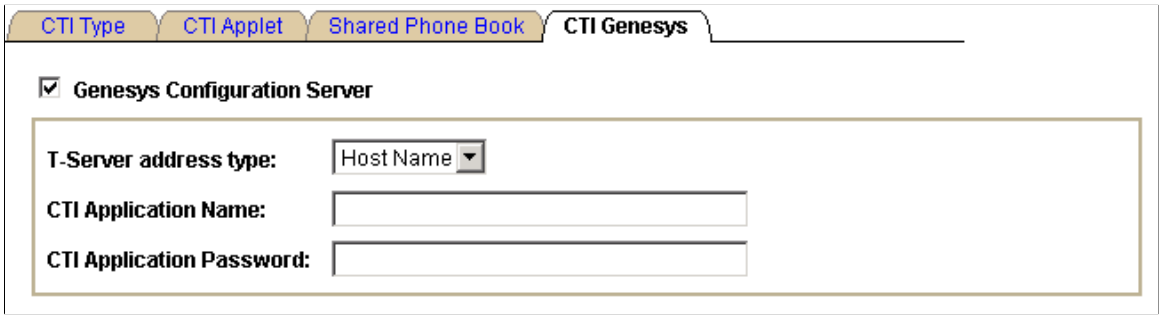
Using the CTI Genesys Page

Access the CTI Genesys page using either of the following navigation paths, whichever is appropriate to you:

PeopleTools, MultiChannel Framework, CTI, CTI Genesys or PeopleTools, MultiChannel Framework, Third-Party Configuration, CTI Configuration, CTI Genesys.

Image: CTI Genesys page

This example illustrates the fields and controls on the CTI Genesys page. You can find definitions for the fields and controls later on this page.



The CTI Genesys page provides additional options for Genesys implementations. This page appears only when Genesys is selected as the CTI vendor on the General Configuration page and the chosen CTI type is Applet.

Genesys Configuration Server

Select to direct the CTI Console to get data required for connecting to the T-Server from a configuration server instead of from the PeopleSoft database. Using a configuration server is transparent to the user or agent. When the agent activates the console, it requests a list of available T-Servers from the Configuration Server, and then the console sequentially attempts to connect to each T-Server in that list until it establishes a

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

connection. If it reaches the end of the list without connecting to a T-Server, an error message is returned to the agent.

Note: If you select this option, you enable the Application User Information group box on this page. Enter the appropriate application user name and password for the Genesys system.

T-Server address type

If you are using a configuration server to indicate what T-Server to connect to (instead of connecting directly to a T-Server), indicate whether the configuration server is returning the host name or the IP address of the T-server.

Enter the type address to use when sending information to the T-Server. The options are *Host Name* and *IP Address.* Your selection should correspond to the host name or IP address that you entered on the CTI Applet page.

CTI Application Name and CTI Application Password

Enter the Genesys application name and password used to sign in to the Genesys Configuration Server. Using this option, each user connects to the configuration server with the same application name.

Using the CTI Cisco Page

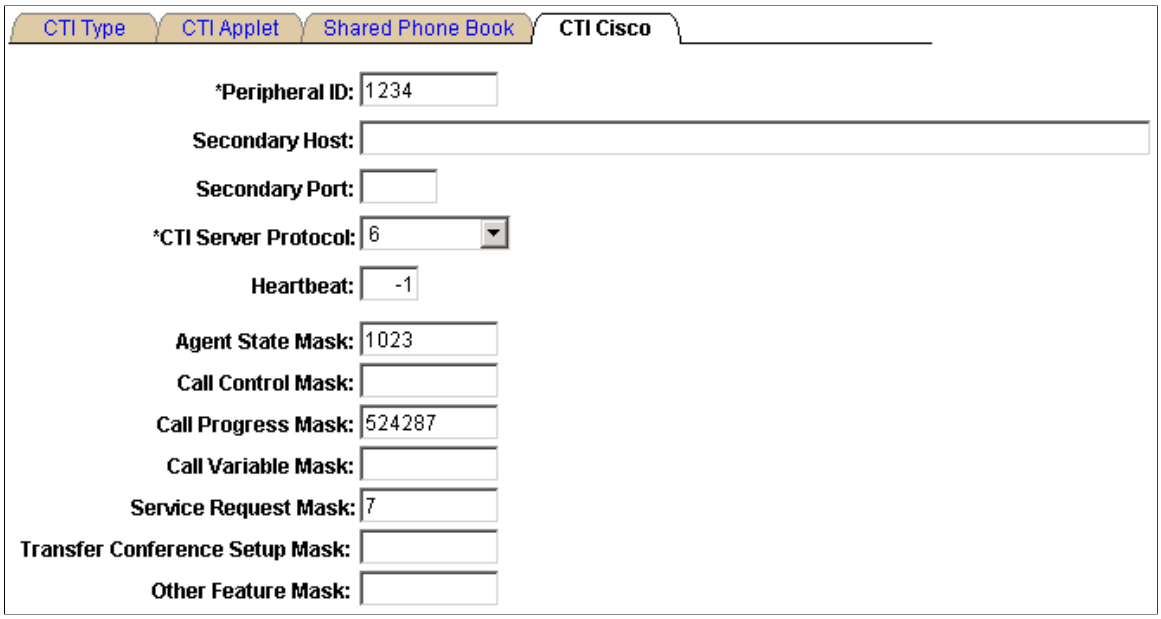
Access the CTI Cisco page using either of the following navigation paths, whichever is appropriate to you:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, CTI, CTI Cisco or PeopleTools, MultiChannel Framework, Third-Party Configuration, CTI Configuration, CTI Cisco.

Image: CTI Cisco page

This example illustrates the fields and controls on the CTI Cisco page. You can find definitions for the fields and controls later on this page.



The CTI Cisco page provides additional options for Cisco implementations. This page appears only when Cisco is selected as the CTI Vendor on the General Configuration page and the chosen CTI type is Applet.

Peripheral ID

Enter the switch (ACD) that the current configuration uses. The peripheral ID is the alias used to identify a switch within the system.

Secondary Host and Secondary Port

Enter a second ICM server. In most cases, the secondary server is used for fail over and load balancing.

CTI Server Protocol

Enables you to set the server protocol for a configuration. Currently, only protocol 6 is supported.

Heartbeat

Enter an interval in seconds for the system to send a message to the CTI Server to make sure it is running. To disable the feature, enter *−1*. To enable it, enter a value in seconds. The minimum value is 5 seconds. Refer to the Cisco documentation for any recommendations for this option.

Agent State Mask

Enter a combination of agent state masks that the CTI agent wants to receive. For instance, you may set the AGENT_AVAILABLE_MASK when you want the application to receive available AGENT_STATE_EVENT messages. Oracle supports the following masks:

- AGENT_LOGIN_MASK
- AGENT_LOGOUT_MASK

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- AGENT_NOT_READY_MASK
- AGENT_AVAILABLE_MASK
- AGENT_TALKING_MASK
- AGENT_WORK_NOT_READY_MASK
- AGENT_WORK_READY_MASK
- AGENT_BUSY_OTHER_MASK
- AGENT_HOLD_MASK

Note: This value can be overridden by a Cisco system-level mask.

Refer to your Cisco documentation for more information.

Call Control Mask

The PeopleSoft system does not currently use this mask in the CTI application. This option is reserved for future use.

Refer to your Cisco documentation for information about this mask.

Note: This value can be overridden by a Cisco system-level mask.

Call Progress Mask

Displays any unsolicited call event messages that you want your application to receive. For instance, you can enter the events that a particular application depends on, such as Begin_Call, CALL _DELIVERED_EVENT, and so on. You can opt to skip events that the application does not depend on, and this can reduce network traffic.

The PeopleSoft system uses the following masks:

- BEGIN_CALL_MASK
- END_CALL_MASK
- CALL_DATA_UPDATE_MASK
- CALL_FAILED_MASK
- CALL_DELIVERED_MASK
- CALL_ESTABLISHED_MASK
- CALL_HELD_MASK
- CALL_RETRIEVED_MASK
- CALL_CLEARED_MASK
- CALL_CONNECTION_CLEARED_MASK

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- CALL_ORIGINATED_MASK
- CALL_CONFERENCED_MASK
- CALL_TRANSFERRED_MASK
- CALL_DIVERTED_MASK
- CALL_SERVICE_INITIATED_MASK

Note: Modify this value with caution. If you mistakenly elect to skip an event on which an application depends, the application can fail.
This value can be overridden by a Cisco system-level mask.

Call Variable Mask

This option is intended for future use. It is not currently implemented for use with PeopleSoft applications.

Refer to your Cisco documentation for information about this mask.

Note: This value can be overridden by a Cisco system-level mask.

Service Request Mask

Enables you to adjust CTI service masks. The PeopleSoft system uses the following masks:

- CTI_SERVICE_CLIENT_EVENTS
- CTI_SERVICE_CALL_DATA_UPDATE
- CTI_SERVICE_CLIENT_CONTROL

If you modify this option, do not enter a value lower than *7*.

Refer to your Cisco documentation for more information about this option.

Note: This value can be overridden by a Cisco system-level mask.

Transfer Conference Setup Mask

Enter the valid ways the application can be configured for a transfer or conference call.

Refer to your Cisco documentation for more information about this option.

Other Feature Mask

Enables you to select the other features supported by an application.

Refer to your Cisco documentation for more information about this option.

Two additional parameters are available to configure the Cisco expanded call context variable. Configure these expanded call context variables in the Cisco middleware:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- user.PS.referenceID: CTI internal use only.

  The console uses this parameter to identify a call.

- user.PS.OutboundContext: This variable, a string type, is attached to an outbound call.

  This variable can be used to associate an outbound call with the desired context.

Configuring PeopleSoft CTI Queues and CTI Agents

To configure CTI queues and agents, use the Queue Configuration (PT_CTI_QUEUE) and CTI Agent Configuration (PT_CTI_AGENT) components.

This section discusses how to:

- Configure CTI queues.
- Configure CTI agents.
- Use the Phone Book page.
- Personalize he agent console.
- View information about the Agent Info page.

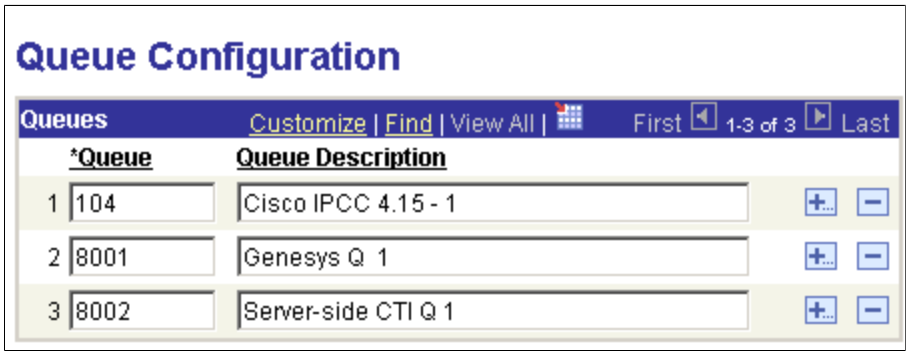
Configuring CTI Queues

Access the Queue Configuration page using the following navigation path:

PeopleTools, MultiChannel Framework, CTI Configuration, Queue

Image: Queue Configuration page

This example illustrates the fields and controls on the Queue Configuration page. You can find definitions for the fields and controls later on this page.



Use this page to add queues for agents.

Queue Enter the directory number identifying an ACD group. Calls to a group are distributed to ACD agents belonging to that group, according to ACD algorithms.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: Queue names can be alphanumeric.

Queue Description Enter a brief description of the queue.

See Defining PeopleSoft MCF Queues for a Third Party.

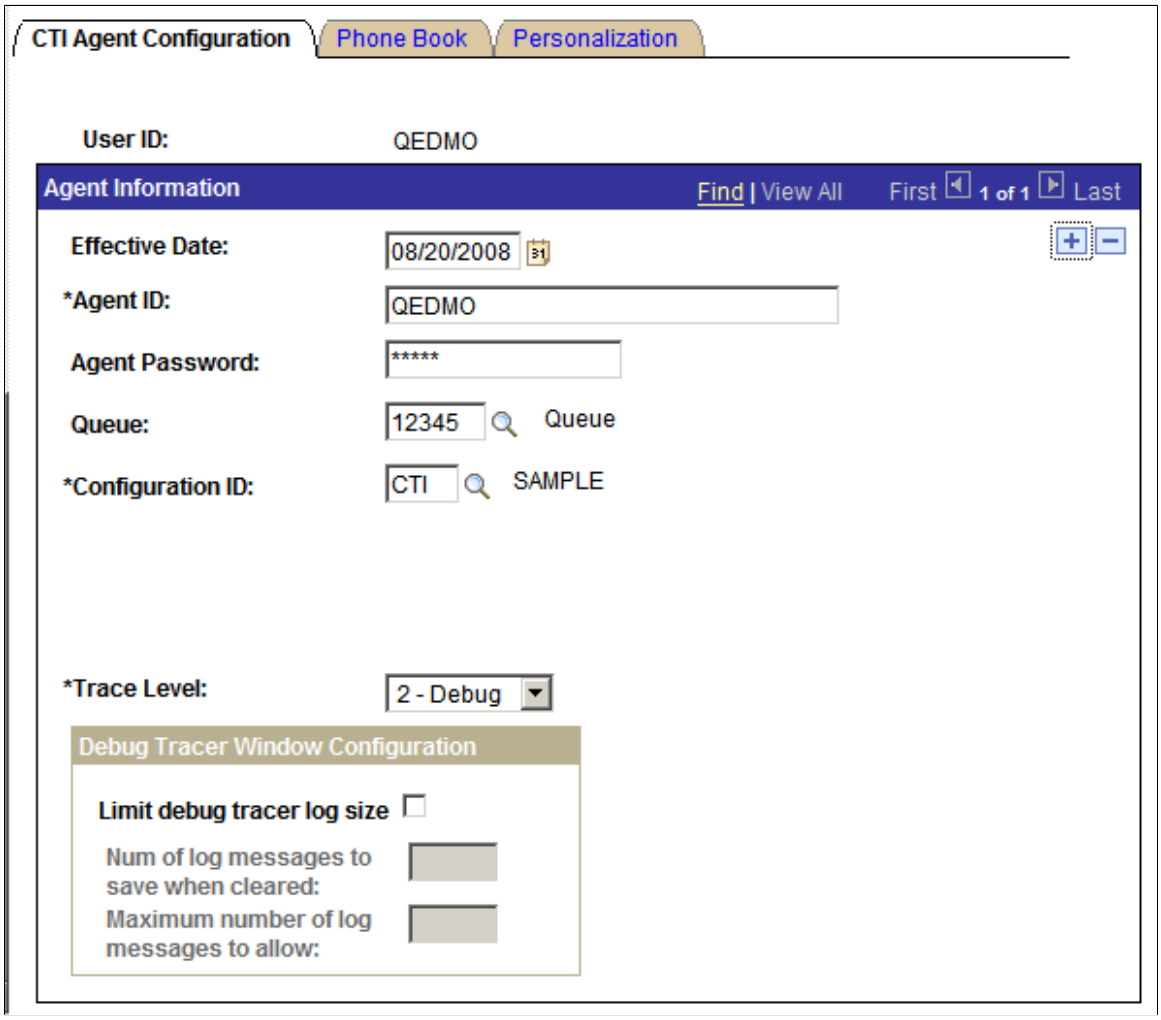
Configuring CTI Agents

Access the CTI Agent Configuration page using the following navigation path:

PeopleTools, MultiChannel Framework, CTI Configuration, Agent, CTI Agent Configuration

Image: CTI Agent Configuration page

This example illustrates the fields and controls on the CTI Agent Configuration page. You can find definitions for the fields and controls later on this page.



Configuring CTI agents involves the CTI Agent Configuration page, the Phone Book page, and the Personalization page.

User ID Displays the PeopleSoft user ID of the agent.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Agent Information

Effective Date

Enter the date on which the current configuration should become active.

Agent ID

Enter a user ID for the agent within the switch.

Agent Password

Enter the password that the agent uses to sign in to the phone, if any.

Queue

Enter the name of the queue that you want to assign to an agent. Use the Queue Configuration page to associate a queue with a directory number identifying an ACD group.

Configuration ID

Select the name of the configuration that you want to associate with the agent. The configuration ID is the name of the configuration that you created using the CTI Applet or CTI Configuration page.

Application User Name/Password

These fields are applet-specfic and appear only in the applet-based solution. Enter the CTI user name and password of the agent.

CTI Client Signature

Note: This option appears only for Cisco configurations.

This control shows the signature of a particular agent. The signature uniquely identifies an agent if you have implemented call monitoring. Typically, this value appears as an email address, such as john.doe@example.com.

Trace Level

Options are:

- *0 - None:* Disables tracing.
- *1 - Info* (informational): Traces agent actions, such as dialing out, transfers, and so on.
- *2 - Debug:* Used to troubleshoot crashes and other major errors.

  If you ever need to open a PeopleSoft GSC case about a CTI issue, include a level 2 - Debug trace.

If a value other than *0* is selected, a tracer window appears to display activities and events on the chat or MultiChannel Console for debugging purposes.

If you are using the JSMCAPI console, two tracers (a JSMCAPI tracer and a console tracer) appear if trace level 2, Debug, is selected.

If you are using the Java applet console, trace information is written to the browser's Java Console, which must be enabled. In Microsoft Internet Explorer, enable the Java Console by selecting Tools, Internet Options, Advanced, Java, Java Console.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

View the Java Console by selecting View, Java Console. To clear the console, press C on your keyboard.

In Sun Java, enable Java Console by selecting Tools, Sun Java.

Note: Setting the trace level to *2 - Debug* can degrade performance. Unless you are troubleshooting the system, set the trace level to *None.*

Limit debug tracer log size

This check box is enabled when the value entered for Trace Level is not *0.* Select this check box to enable the agent to clear the tracer log based on Number of log messages to save when cleared and Maximum number of log messages to allow.

If the check box is cleared, the tracer log will not get cleared and the Number of log messages to save when cleared and Maximum number of log messages to allowfields will be disabled.

Number of log messages to save when cleared

Specify the minimum number of recent tracer log messages that should be maintained in the tracer window.

Maximum number of log messages to allow

Specify the maximum number of tracer log messages that will be maintained in the tracer window.

See Creating PeopleSoft MCF Agents for a Third Party.

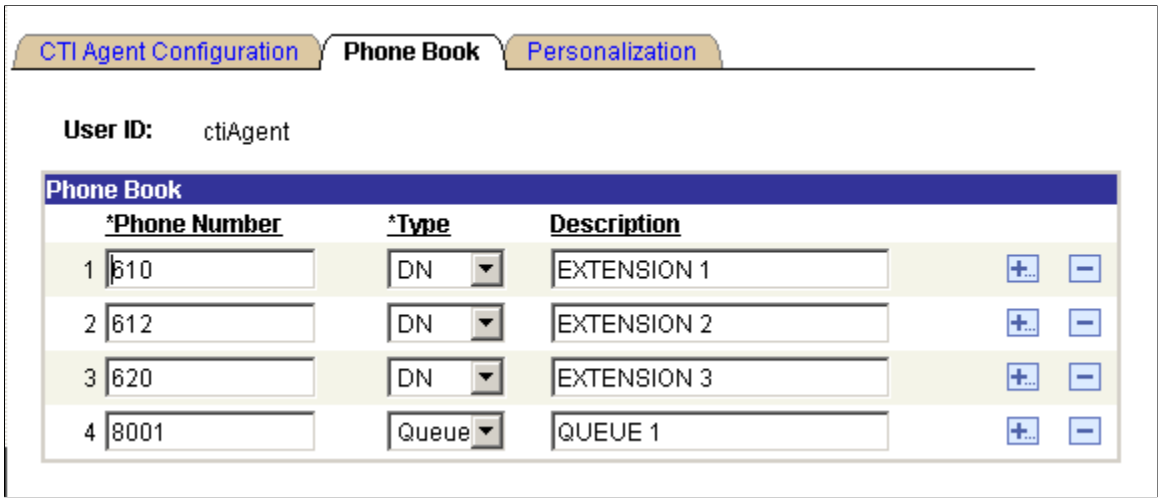
Using the Phone Book Page

Access the Phone Book page using the following navigation path:

PeopleTools, MultiChannel Framework, CTI Configuration, Agent, Phone Book

Image: Phone Book page

This example illustrates the fields and controls on the Phone Book page. You can find definitions for the fields and controls later on this page.



 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

On this page, you can manage a list of frequently dialed telephone numbers for a *specific* CTI agent. These numbers appear when that agent selects the drop-down list box for a number to dial in the CTI Console. This listing saves the agents from having to manually enter frequently dialed numbers when making outbound calls.

Note: Phone lists are updated on the CTI Console only after the CTI Console launches. To refresh phone lists, refresh the browser and reactivate the console.

Phone Number

Enter frequently dialed telephone numbers associated with a particular agent.

Type

Select one of the following options:

- *DN* (directory number): This number identifies a telephone set on a PBX or in the public network. The caller dials this number to establish a connection to the addressed party. The DN can be a local PBX extension (a local DN) or a public network telephone number.
- *Queue* Directory number identifying an ACD queue or group. Calls to a group are distributed to agents belonging to the group, according to ACD algorithms.

Description

Add a description for the telephone number.

Personalizing the Agent Console

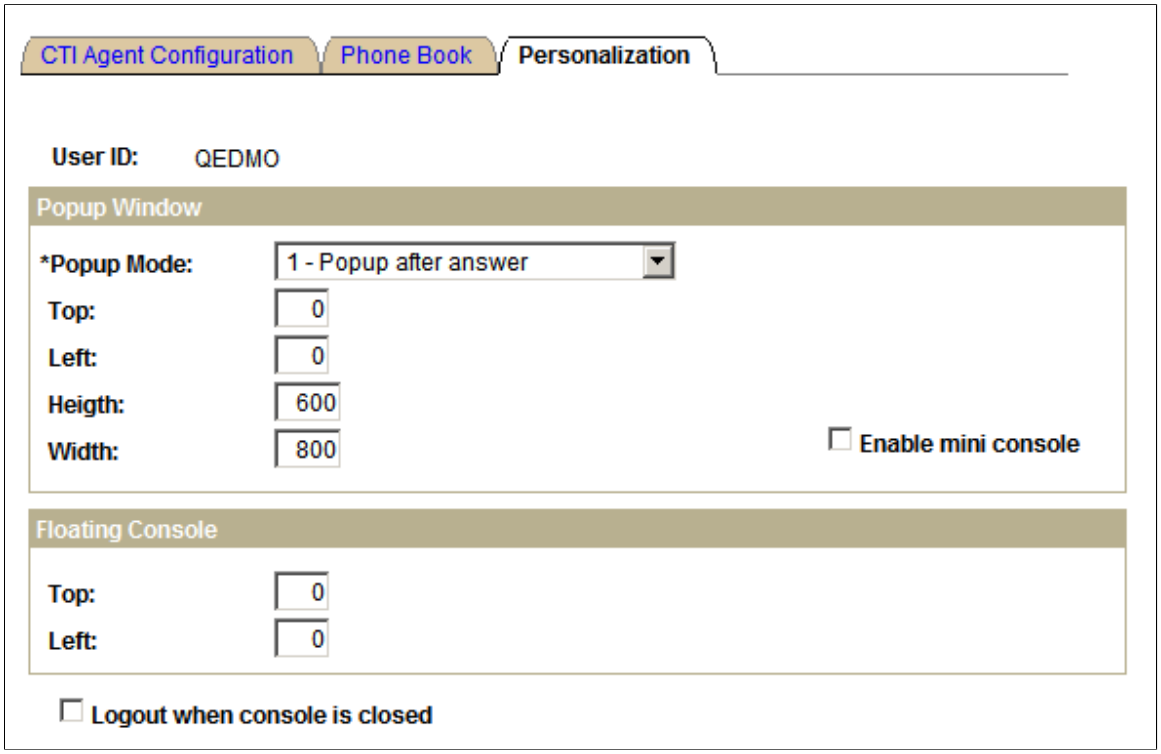
Access the Personalization page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, CTI Configuration, Agent, Personalization

Image: Personalization page

This example illustrates the fields and controls on the Personalization page. You can find definitions for the fields and controls later on this page.



Use this page to personalize timing, size, and position of the pop-up window on the desktop as well as the position of the floating console.

Popup Window

Popup Mode

Enables you to configure when the pop-up window appears. You can have it appear after the call is answered, or you can have it appear when a call comes in. If it appears before the call is answered, the agent can determine whether she or he wants to answer the call based on the information that appears in the pop-up window.

The default pop-up mode is *1- Popup after answer.*

In some cases, for example accepting a transfer, although you set the value as *0- Popup when incoming,* you still get the pop-up window after you answer. This is because no call data was attached by the CTI vendor to the incoming event, and consequently the PeopleSoft CTI Console couldn't build the URL for the pop-up window.

For example, assume party A is transferring a call to party B. Party B gets the incoming event, but party B doesn't get a pop-up window. This is because the console did not receive the user data to generate the URL despite the fact that the mode is set

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

to *0.* But after B answers this call, A completes the transfer. Then, B gets the event *partychanged,* receives the user data, and generates the URL and pop-up window.

Top Enter the top position, in pixels. This value is relative to the top of the screen.

Left Enter the left position, in pixels. This value is relative to the left side of the screen.

Height Enter the height of the window, in pixels. The minimum value is 100.

Width Enter the width of the window, in pixels. The minimum value is 100.

Logout when console is closed This setting depends on CTI vendor support in applet-based consoles, and may not be supported by all vendors. For JSMCAPI consoles and applet consoles with vendor support, select this setting to automatically sign out the agent when the CTI Console is closed.

Enable mini console Select to enable the administrator or user to configure the presence of the CTI mini console in the pop-up window. If this check box is not selected, the mini console does not appear in the popup window.

Note: The mini console has performance overhead.

Floating Console

Top Enter the top position in pixels. This value is relative to the top of the screen.

Left Enter the left position in pixels. This value is relative to the left side of the screen.

Viewing Information About the Agent Information Page

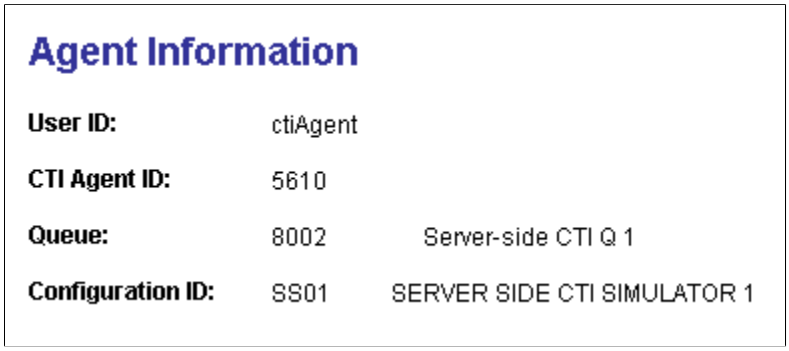
Access the Agent Information page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, CTI Configuration, Agent Information

Image: Agent Information page

This example illustrates the fields and controls on the Agent Information page. You can find definitions for the fields and controls later on this page.



The Agent Information page is a read-only page that displays the following information about a CTI agent.

User ID	Displays the agent's PeopleSoft user ID.
Agent ID	Displays the agent's CTI middleware ID.
Queue	Displays the queue to which an agent is assigned.
Configuration ID	Displays the configuration ID associated with the agent.

Using Other PeopleSoft CTI Options

You can configure other optional CTI parameters.

This section discusses how to:

- Configure pop-up windows.
- Support single sign-in.
- Log CTI events.
- Implement free seating.

Configuring Pop-Up Windows

PeopleSoft CTI can launch and populate transaction pages in the following ways:

- Default URL.

  The same URL is used for all calls. The Automatic Number Identification (ANI) is passed in as a parameter to that URL. The ANI identifies the telephone number from which the incoming call originated, and the number may be useful in determining what application data to retrieve. For example, it may be the home phone number of a customer.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Build the URL from attached Call Data.

  PeopleSoft CTI formats a URL for the browser with a specific target PeopleSoft menu, market, and component. However, this method cannot be used if you are accessing multiple databases through the PeopleSoft Portal.

- iScripts.

  PeopleSoft CTI opens the transaction page using an iScript. This method must be used if you are accessing multiple databases through the PeopleSoft Portal. The iScript communicates with the targeted database to populate the appropriate transaction page with the caller's data.

Note: The call ID is passed in the URL string as a variable named *callID.*

Note: Browser settings and add-on tools that block pop-up windows can prevent CTI inbound call pop-up windows from appearing. If possible, configure such tools to allow pop-up windows from your site or domain.

Setting Up Genesys for Pop-Up Windows Using Applet Solution

Unless you choose to use the default URL for your pop-up windows, you must create certain user-defined variables that are attached to each incoming call. These variables are then sent to the PeopleSoft CTI application providing instructions on which PeopleSoft Pure Internet Architecture page appears for the agent.

Note: The following variables are case-sensitive. Match the case of the variables as shown in this section.

ICType

Represents the type of PeopleSoft service being called, either a panel (page) or a script.

If the ICTYPE is set to *Panel,* then the following three attached data variables must be set:

- Menu: The name of the PeopleSoft menu containing the destination component.
- Market: The market property of the target component.
- Component: The target component name in the PeopleSoft Application.

If the ICTYPE is set to *Script,* then the following attached data variable must be set:

ICScriptProgramName. This represents the location of the iScript, which is run through the pop-up screen.

Descr (description)

Optional data for descriptive purposes only. You can add any descriptive information that might be useful to the agent. This information appears in the CTI Console. For example, you may want the agent to be aware of the customer's priority status, as in Gold Customer.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Application Specific Data

Any other attached data keys that are sent to the CTI Console by the Genesys telephony server, such as customer number, are passed to the target application page as parameters.

The parameters are separated from the PeopleSoft URL by a question mark (?). Parameters are separated from other parameters by an ampersand (&). If you are using iScripts, you can use the GetParameter methods to read the parameters in your PeopleCode. If you choose to use ICPanels instead, the parameters are used as the key list.

Refer to your PeopleSoft application documentation for specific instructions on any other attached data keys that must be passed to the CTI Console by Genesys.

Setting Up Cisco for Pop-Up Windows Using Applet Solution

Unless you choose to use the default URL for your pop-ups windows, you must create certain expanded call context (ECC) variables. ECC variables are variables that you define and enable in the Cisco ICM Configuration Manager to store values associated with the call. These variables are sent to the PeopleSoft CTI Console when ICM notifies it of an incoming call, and the console uses the variables to determine which PeopleSoft page to launch for the agent.

Every call data key used in your implementation must be registered in the Cisco ICM, before the PeopleSoft system can receive the attached call data from the IVR.

The PeopleSoft system recognizes only data keys with *user.PS.* in front of all the PeopleSoft key names. For example,

```
user.PS.Descr
```

The following table contains the call data variables that must be created and registered in the ICM.

Note: The following variables are case-sensitive. Match the case of the variables as shown in this section.

ICType

user.PS.ICType. 6 characters. Valid values are *Panel* (page) or *Script*. It represents the type of PeopleSoft service being called, either a panel (page) or a script.

If ICType is set to *Panel*, the following variables apply:

- user.PS.Component. 20 characters. This variable is required to specify the name of the destination component.
- user.PS.Menu. 10 characters. This variable is required to specify the name of the menu within PeopleSoft containing the destination component.
- user.PS.Market. 3 characters. This variable is required to specify the name of the market of the destination component.

If the ICType is set to *Script*, then the following variable applies:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

	user.PS.ICScriptProgramName. 70 characters. If the ICTYPE is set to Script, then this variable is required to specify the location of the iScript, which is run through the pop-up screen.
Descr	user.PS.Descr. 30 characters. You can add any descriptive information that might be useful to the agent. This information appears in the control bar. For example, you may want the agent to be aware of the customer's priority status, as in Gold Customer.
Application Specific Data	Any other variables that the application requires must be prefixed with *user.PS.* PeopleSoft truncates *user.PS.* before passing the parameter to the application. For example, user.PS.App1 (3 characterss) user.PS.App2 (1 characterss)

Note: Typical character sizes for these variables appear in the previous table. The actual size of these variables depends on the components specified by the PeopleSoft application you are using. Refer to your PeopleSoft application documentation for correct size information.

Related Links

"Understanding Internet Script Classes" (PeopleTools 8.54: PeopleCode API Reference)

Supporting Single Sign-In

In the applet-based solution, PeopleSoft CTI offers single sign-in. The console connects to the CTI middleware using the CTI user ID and password retrieved from the PeopleSoft database.

Related Links

"Understanding Single Signon" (PeopleTools 8.54: Security Administration)

Logging CTI Events

CTI events can be logged in the CTI Event Log. You can view the logged CTI events by navigating to PeopleTools, MultiChannel Framework, Universal Queue, Administration, CTI Event Log.

To implement CTI event logging, navigate to PeopleTools, MultiChannel Framework, Universal Queue, Configuration, Cluster Tuning and set the key *log_cti* to *yes.* Use the Notify Cluster page to notify the affected queue to refresh logging parameters.

The CTI event log logs all events and requests related to a session, user, or call connection. It also logs all events related to call and group information.

Note: Applet-based CTI events are not logged.

Related Links

Tuning Cluster Parameters

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Implementing Free Seating

When users sign in, they do not need to reenter telephone extensions and other user information if they have used the workstation before and the relevant information for the telephone associated with that workstation has not changed.

The PeopleSoft system enables free seating by maintaining a cookie on the workstation.

Using the PeopleSoft CTI Sample Pages

To demonstrate an outbound call, use the Sample Pages component (PT_CTI_DEMOOUTB). The CTI sample pages are intended for demonstration purposes only and should not be used in production.

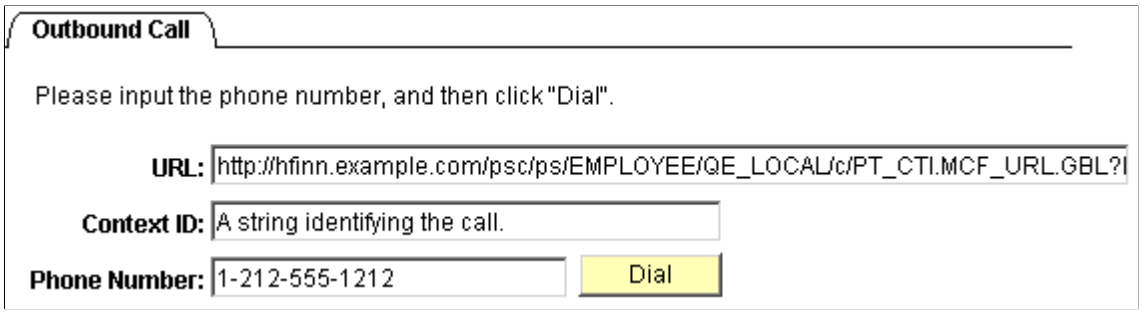
Using the Outbound Call Page

Access the Outbound Call page using the following navigation path:

PeopleTools, MultiChannel Framework, CTI Configuration, Sample Pages, Outbound Call

Image: Outbound Call page

This example illustrates the fields and controls on the Outbound Call page. You can find definitions for the fields and controls later on this page.



The Outbound Call page is an example of how you can customize an application page to enable a user to direct the CTI Console to dial a telephone number displayed on that page. The outbound calling demonstration works only when the CTI Console is enabled and the user has registered with the CTI vendor.

URL	Enter the URL of a page to display when dialing out.
Context ID	Enter a string to attach to the call as outbound context.
Phone Number	Enter the telephone number that you want to dial. This field accepts numeric digits only. Do not enter special characters, such as - (hyphen), . (period), or other separators.
Dial	Click to dial the telephone number that you entered.

Related Links

Using the CTI Sample Console

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 4

Using PeopleSoft CTI

Understanding PeopleSoft CTI

This section discusses:

- PeopleSoft CTI.
- Cisco switch considerations for the applet-based solution.

Important! The CTI Console Java applet is no longer supported in PeopleTools 8.53. Refer to the Product Certifications link on My Oracle Support, https://support.oracle.com (sign-in required), for more information about supported products.

PeopleSoft CTI

PeopleSoft CTI is a browser-based call-management system that helps call agents work more efficiently with customers. PeopleSoft CTI integrates Oracle-validated third-party CTI systems and your PeopleSoft applications. It exchanges data between the CTI system and your PeopleSoft applications so that the system automatically fills PeopleSoft transaction pages with the appropriate customer information—the information related to the caller.

With PeopleSoft CTI, you can perform the following tasks:

- Operate two lines or two extensions.
- Answer incoming calls.
- Release calls.
- Put a caller on hold.
- Monitor call status.
- Access PeopleSoft applications.
- Transfer callers.
- Initiate conference calls.
- Place an outbound call.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: PeopleSoft CTI supports Oracle-validated third-party CTI systems. In the following sections, when the phrase *your CTI middleware* or *CTI vendor* appears, assume that it refers to Oracle-validated third-party CTI systems, as appropriate for your installation.
For a list of partners that offer CTI middleware integrations, refer to Oracle Validated Application Integrations – Find a Partner Solution, http://www.oracle.com/partnerships/isv/integration/search.html
Because a CTI console developed with the JavaScript MultiChannel Application Programming Interface (JSMCAPI) can be customized, features described in PeopleSoft CTI may not appear or may act differently.

The CTI Interface

The PeopleSoft CTI interface is incorporated *within* the MultiChannel Console.

Related Links

Understanding the PeopleSoft CTI Console
Managing Tasks with the MultiChannel Console
Communicating with Customers and Agents Using Chat

Cisco Switch Considerations for the Applet-Based Solution

The following list presents the limitations of the Cisco Systems Internet Protocol (IP) Contact Center switch:

- MAKE_CALL is supported only when the agent is in the Not Ready state.
- Consultative and blind transfers are supported.

  Placing a call on hold and making a new call and completing the transfer is not supported due to limitations in the Call Manager.
- Hold and retrieve are supported, but not during a consultative call because it disrupts the consult relationship in Call Manager.
- Only the conference initiator can add parties to the conference.
- Cisco does not support two applications managing the same call.

  Therefore, you cannot run a Cisco softphone and the PeopleSoft CTI console on the same workstation.

Note: Refer to your CTI vendor documentation for further information about any limitations imposed by a particular switch. In some cases, the switch limitations may actually be due to the CTI server, so be sure to refer to any server limitations that may also impose limitations on the switch.

Using PeopleSoft CTI

This section discusses how to:

- Get started.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Use the CTI console.
- Select call actions.
- Answer a call.
- Transfer a caller.
- Initiate conference calls.
- Work with the hold status.
- Disconnect a caller.
- Switch agent ready status.
- Dial an outbound call.
- Complete a call.
- Use hot keys.

Getting Started

Getting started with PeopleSoft CTI is a three-step process that involves:

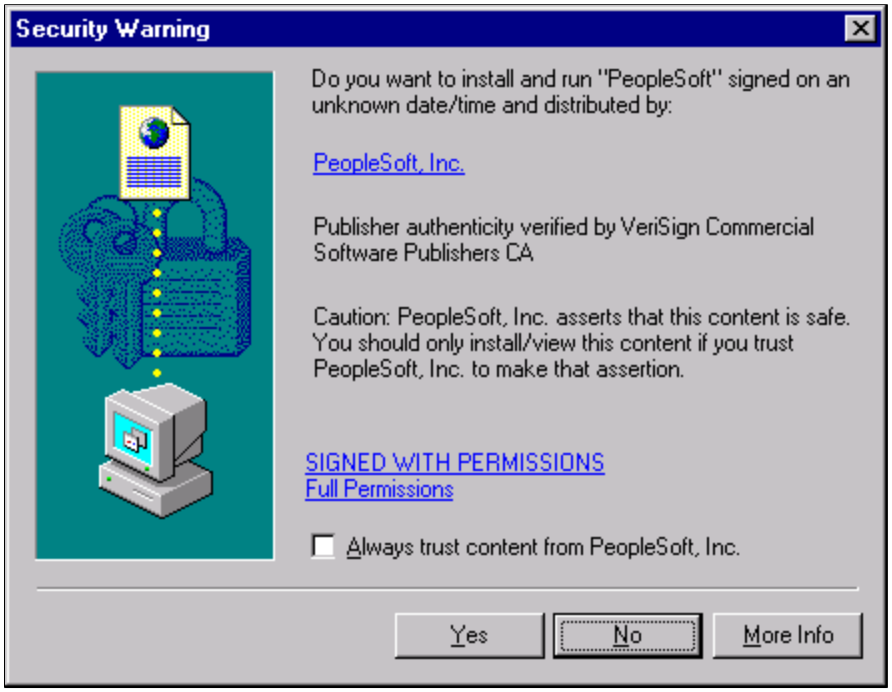
- Signing in to the PeopleSoft system.
- Specifying your extension information.
- Connecting to your CTI middleware.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The first time that you access PeopleSoft CTI using the applet-based solution, a Security Warning dialog box *may* appear prompting you to trust information from PeopleSoft. Click the Yes button.

Image: Security Warning dialog box

This example illustrates the Security Warning dialog box.



The Security Warning dialog box may not appear if:

- You have previously accepted PeopleSoft applets as trusted.
- The system is not correctly installed.

Note: If you have any questions or concerns about this warning, contact your system administrator.

To sign in to PeopleSoft CTI:

1. On the PeopleSoft sign-in page, enter your PeopleSoft user ID and password as you normally do to sign in to the PeopleSoft system.
2. Click the MultiChannel Console link in the universal navigation header.
3. On the CTI control bar, set your configuration by performing the following actions:
   - Ensure that your agent ID appears beneath the CTI Agent ID label.
   - Ensure that the appropriate queue name appears beneath the Queue label.

     Queues are discussed in a subsequent section.
   - If you are signing in from a workstation with a different extension, enter the current extensions in the Extension field.
   - The Extension Type field indicates whether an agent can receive calls by way of a queue or just from a directory number (DN).

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

If this value is set to *Queue,* the console offers options that are queue-specific, such as being able to log on to a queue.

- Click the Activate button to sign in to your CTI middleware.

  The CTI console appears.

Using the CTI Console

Change any registration parameters if necessary and click Set or Activate (depending on your version). After you do so, the CTI interface appears, which looks similar to the following:

Note: Note that the PeopleSoft CTI interface operates *within* the MultiChannel Framework Console. CTI users will see the MultiChannel Console only if their user ID belongs to a role that has MCF_AGENT or MCF_SUPR real-time event notification (REN) permissions, and WEBLIB_MCFLINK web libraries in their role's permission list.

Image: The CTI Console

This example illustrates the fields and controls on the CTI Console. You can find definitions for the fields and controls later on this page.

Note: Do not open two consoles for the same user on a single machine. Doing so can result in an error.

(register)

Select to register and unregister with the CTI middleware. The button acts as a toggle switch. When the green check mark appears, you are registered; when the red *X* appears, you are not registered.

Lines

Select the option to the left of the telephone icon to activate the associated line. The icon that you select determines the active line. All call actions that you choose apply only to the active line. The color of the icon reveals the activity on the line. The colors are:

- Blue: Inactive.
- Red: Ringing.
- Green: On call.
- Yellow: On hold.

Select Call Action

Select actions related to calls, including *Answer, Hold,* and *Transfer.* Which call action options appear depends on your current status. For instance, the Hold call action is valid only while you are on the line with a caller. Call action options are discussed in detail in the following section.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Status Displays the agent status, as in *Ready* or *Not Ready* (to receive calls). It can also show *Ready/DND* (do not disturb) if the agent is not using a queue.

Q (queue) If the agent belongs to an automatic call distributor (ACD) group, the group appears here.

Messages Informational messages appear in the right corner of the CTI Console. Examples of such messages are *Dialing, Connected,* and *Released.* These messages do not persist.

Note: Error messages appear in separate windows.

Call Duration The system tracks the amount of time spent on calls for each line.

Incoming Call Information Displays the string associated with the Descr call variable. For example, *Gold Customer.*

Note: In the applet-based solution, some CTI systems will automatically sign out the CTI agent when the CTI console is closed. For CTI systems in which this is not automatic, the option can be set on the agent Personalization page (select PeopleTools, MultiChannel Framework, CTI Configuration, Agent, Personalization.)

Selecting Call Actions

The Select Call Action drop-down list box contains all of the options that you have for handling calls. Depending on the status of the agent or the telephone line, certain selections from the drop-down list box are not available.

After you select a call action, two buttons, the Go performs the selected call action.

The following list describes the call actions:

Dial When you are in Agent Ready or Agent Not Ready mode, you can call another party.

Answer Call This button appears and flashes when an incoming call is waiting to be answered.

Transfer Mute Transfers the caller to the desired number without speaking to the intended recipient. A dialog box appears enabling you to enter the extension of the person to whom you want to transfer the caller.

Note: In the applet-based solution, this action is not supported on Aspect switches.

Transfer Transfers the caller to the desired number. You have the opportunity to speak with the recipient before transferring the

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

caller. A dialog box appears enabling you to enter the extension of the person to whom you want to transfer the caller.

Conference Enables you to add one or more individuals to your call.

Hold Places the caller on hold.

Retrieve Takes the caller out of *Hold* status. This option appears only when the caller is on hold.

Release Disconnects the caller. This option is available as a selection only after a call is answered.

Change When an agent status is unknown, the system sets the status to *Change* so that the user can set the status manually to match the status for the telephone.

Note: You can transfer to or initiate a conference call with individuals who are not enabled to access PeopleSoft CTI. Their phone rings, but remember that the pop-up window showing customer data does not appear.

In the applet-based solution, the following call actions are available depending on whether you are using a DN or Queue configuration:

Queue Select from:

Log on: Enables the agent to log an extension on to a queue.

Ready: Indicates that the agent is ready to receive incoming calls. This option is available when the extension is associated with a queue and the status bar reads Agent Not Ready.

Not Ready: Stops incoming calls. This option is available when the extension is associated with a queue and the status bar reads Agent Ready.

Log off: Enables the agent to log an extension off a queue. When logged off, the agent is no longer participating in the queue.

DN This option is available when the extension is not associated with a queue.

Ready: Indicates that the agent is ready to receive incoming calls.

In the adapter solution, the availability and meaning of the call actions are determined by the third-party CTI vendor. Consult your CTI vendor for details.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Answering a Call

After you have signed in to PeopleSoft CTI, you can receive calls. For each incoming ACD call to your extension, the telephone extension icon turns red. After you have accepted a call, the system does not send you more incoming calls until you have completed the current call.

To answer a call:

1. Select the radio button to the left of the telephone icon that has turned red.

   The *Answer* option is automatically selected as the current option in the drop-down list box when an incoming call arrives.

2. Click Go.

   The pop-up browser launches with the appropriate PeopleSoft transaction page displayed. The system determines which page to display based on caller information sent by the CTI middleware.

   After you have answered a call, you enter the not available status.

If an agent erroneously cancels a call instead of clicking Go when accepting an incoming call, the agent can recover the call by:

1. Selecting a call option.
2. Selecting the line that is receiving the incoming call.

Note: If CTI screen pop-ups are configured, the CTI console attaches a CTI miniconsole to the screen pop-up page. This miniconsole takes a few seconds to initialize before it can be used.
In the Javascript console, you can configure whether the miniconsole appears.

Transferring a Caller

Occasionally, you need to transfer callers to other agents. PeopleSoft CTI supports two types of transfers:

- Transfer Mute: This option enables you to transfer a call without speaking to the target agent before transferring the call.
- Transfer: This option is also known as a consultative transfer, which means that you consult with the target agent before transferring the call.

Note: You can always transfer or invite users to a conference call even if the called party is not CTI-enabled. The non-CTI-enabled users do not get pop-up windows, but their phones still ring.
When you initiate a transfer or conference call on the Cisco system using the applet-based solution and the contacted party does not answer, wait until the system notifies you of the unsuccessful connection before attempting another action. This wait can be 20 to 30 seconds.
In the applet-based solution, Cisco ICM does not notify client applications about some call events during two-step transfers or conferences. As a result, the state of the PeopleSoft CTI Console may not stay synchronized with that of the teleset, especially if the teleset is used to initiate or complete these call actions. The CTI console should automatically resynchronize with the teleset when the call is completed.

To perform a transfer mute:

1. Select the appropriate telephone line.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The selected line must be green.

2. From the Select Call Action drop-down list box, select *TransferMute.*

3. In the Phone No. edit box, select the number that you want to dial.

   The drop-down list box contains all of the numbers from the shared phone book and agent phone book. If the number that you want to dial does not appear, click Dial other number and manually enter the number.

4. Click Go.

   This connects the caller to the new agent and releases your line.

   The system prompts the recipient of the transfer that it is transferring a call from your extension. When the recipient accepts the transfer, the PeopleSoft page connected to the caller's case opens as it did when you first received the call.

To perform a transfer (consultative):

1. Select the appropriate telephone line.

   The selected line must be green.

2. From the Select Call Action drop-down list box, select *Transfer.*

3. In the Phone No. edit box, select the number that you want to dial.

   The drop-down list box contains all of the numbers from the shared phone book and agent phone book. If the number that you want to dial does not appear, click Dial other number and manually enter the number.

4. Click Go.

   When you use one extension with two lines, the outbound call you make to the agent to whom you are transferring the incoming call gets initiated on your second line and the incoming call gets placed on hold. When the outbound call is established, you can consult with the recipient and place that call on hold. To complete the transfer, you need to return to the first line and select *Complete* and click Go. This action releases the call on the second line and transfers the call on the first line to the recipient of the transfer.

   When you use two extensions, each with one line, the CTI Console does not have access to the outbound call to the intended recipient of the transfer. When the outbound call is established, you can consult with the recipient. You do not have to toggle between the two lines, and you cannot put the recipient on hold.

5. To complete the transfer, select *Complete* and click Go.

Initiating Conference Calls

If you need the assistance of other agents to answer a caller's questions, you can use the conference feature to include the appropriate agents on a call.

To initiate a conference call:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

1. Select the appropriate telephone line.

   The selected line must be green.

2. From the Select Call Action drop-down list box, select *Conference.*

3. From the drop-down list box of all numbers from the shared and agent phone books, select a number to be dialed.

   If the number is not there, select *Dial other number* to access an edit box and enter the number to be dialed.

4. Click Go.

   The system notifies the target agent of the incoming call (conference). The PeopleSoft page associated with the caller's case opens for the target agent as it did for you when you first received the call.

   This feature depends upon two parameters set up by the administrator:

   - Default Screen Pop-up URL.
   - Personalization: Screen Pop-up Mode.

     If the default URL for screen pop-up is set and the screen pop-up mode is 0 (pop up when incoming) for the second agent, the agent gets the screen pop-up as soon as the call is transferred.

     If the screen pop-up is set to 1 (pop up after answer), the screen pops up only after the first agent completes the transfer or conference call. However, if the default URL for the screen pop-up is not set, then whether the mode is 0 or 1 doesn't matter. In that case, the second agent gets the screen pop-up only after the first agent completes the transfer or conference call.

     When you use one extension with two lines, the outbound call that you make to the agent to whom you are inviting to the conference gets initiated on your second line and the incoming call gets placed on hold. After the outbound call is established, you can consult with the third party, and place that call on hold. To complete the conference, you need to return to the first line and select *Complete* and click Go. This action releases the call on the second line and starts the conference on the first line.

     When you use two extensions, each with one line, the CTI Console cannot access the outbound call to the third party. When the outbound call is established, you consult with the target agent. You do not have to toggle between the two lines, and you do cannot put the recipient on hold. To start the conference, select *Complete* and click Go.

5. After consulting with the target agent, select *Complete* from the Select Call Action drop-down list box, and click Go.

Working with the Hold Status

Putting calls on hold and retrieving calls on hold is likely to be the call action that you perform most.

To place a call on hold:

1. Select the appropriate telephone line.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The selected line must be green.

2. From the Select Call Action drop-down list box, select *Hold.*
3. Click Go.

To retrieve a call on hold:

1. Select the appropriate telephone line.

   The selected line must be green.

   The retrieve option is automatically selected as the current option in the drop-down list box when a call is on hold.

2. Click Go.

Disconnecting a Caller

After you have finished a call, you need to release the call.

To release a call:

1. Select the appropriate telephone line.

   The selected line must be green.

2. On the console control bar, select *Release* from the Lines drop-down list box.
3. Click Go.

   If you are using the applet-based solution, the system automatically places you in wrap-up mode, which enables you to complete any remaining work before accepting more incoming calls. Technically, when you are in wrap-up mode, your status is *Agent Not Ready.*

   If you are using the adapter-based solution and you have configured wrap-up mode in your CTI middleware, the console can automatically place you in a *Not Ready* or *Work Not Ready* state, which enables you to complete any remaining work before accepting more incoming calls.

   When you are ready to accept incoming calls, select *Agent Ready.*

Switching Agent Ready Status

Your agent status determines whether you can receive incoming calls.

In the applet-based solution, agent statuses include:

- *Agent Ready*

  To activate *Agent Ready* status:

  From the Select Call Action drop-down list box, select *Agent Ready.*

  When you are ready, the system routes incoming calls to your extensions.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Do Not Disturb

  To activate Do Not Disturb status:

  From the Select Call Action drop-down list box, select *DND*.

  With Do Not Disturb, your extensions do not accept incoming calls.

  Note: This status is not available to agents associated with an ACD queue.

- *Agent Not Ready*

  To activate *Agent Not Ready* status:

  From the Select Call Action drop-down list box, select *Agent Not Ready.*

  This status is typically used when agents are at their desks, but temporarily unable to receive calls. While you are not ready, the system routes calls to other available agents.

  Note: This status applies only to agents associated with an ACD queue.

Note: In the adapter-based solution, the adapter provider is responsible for determining what states and statuses are available in the drop-down list boxes, as well as the meaning of those states.

Dialing an Outbound Call

You can use PeopleSoft CTI to place a call while the agent status is either *Agent Ready* or *Agent Not Ready.*

Note: If you have multiple extensions assigned to you, *do not* call one of your extensions from the other.

To place an outbound call:

1. Select your status from *Agent Ready* or *Agent Not Ready.*
2. Select the option next to the telephone icon representing a free line.

   For example, if you had a customer on hold on one line, you would select the icon for the second line.
3. From the Select Call Action drop-down list box, select *Dial.*
4. From the drop-down list box showing all numbers from the shared and agent phone books, select a number to be dialed.

   If the number is not there, select Dial other number to get an edit box and enter the number to be dialed.
5. Click Go.

   As with any other call you receive, you can access all the call actions for calls that you initiate. You can transfer the person that you've called, place the line on hold, or initiate a conference with another party.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: If you are using the applet-based solution, the system places you in *Agent Not Ready* status until you release the call. If you are using the adapter-based solution, you must configure your CTI middleware call actions.

When you make an outbound call:

- You can specify a URL to display a page.

  When you display the page, the outbound call data is also attached to the URL so that an application can collect information such as the automatic number identification (ANI), dialed number identification service (DNIS), and so on.

- A context ID (such as a customer case number or an invoice number) is attached to the call data.

  This allows PeopleSoft application context to be passed on to CTI middleware, where it can be stored. This context can be used to establish a relationship between the outbound call and the application context when the call was made.

Related Links

Creating a List of Frequently Dialed Phone Numbers

Completing a Call

When you disconnect or release a call from the miniconsole, the system disconnects you from the CTI middleware, and the miniconsole becomes disabled (unavailable for entry). However, the PeopleSoft page remains active so that you can finish updating information if needed.

The availability of an agent to accept incoming calls depends on how the system administrators of the agent have set up the agent's CTI middleware.

Using Hot Keys

Hot keys are combinations of keyboard buttons that you can press instead of using a mouse. To help you easily select options, PeopleSoft CTI offers the following hot keys:

Hot Key	*Description*
Alt + R	Registers your phone extensions with the CTI middleware.
Alt + 1	Makes Extension 1 the active line.
Alt + 2	Makes Extension 2 the active line.
Alt + S	Presents a list of applicable call actions for you to select.
Alt + P	Presents a list of frequently called telephone numbers for you to select.
Alt + G	Enables you to perform a call action.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Hot Key	*Description*
Alt + C	Enables you to cancel a call action.
Alt + Z	Enables you to check agent status.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 5

Configuring REN Servers

Understanding REN Servers

This section discusses:

- REN server failover, scalability, and security configuration.
- REN server failover.
- REN server clusters.

The REN server, an application server domain process, is essential to PeopleSoft MultiChannel Framework (MCF) architecture. MCF events are sent to REN servers, which deliver them to recipients of those topics.

REN servers are also used by other PeopleSoft applications to push event notifications to users, such as the Reporting Window output option and the Optimization Progress Window.

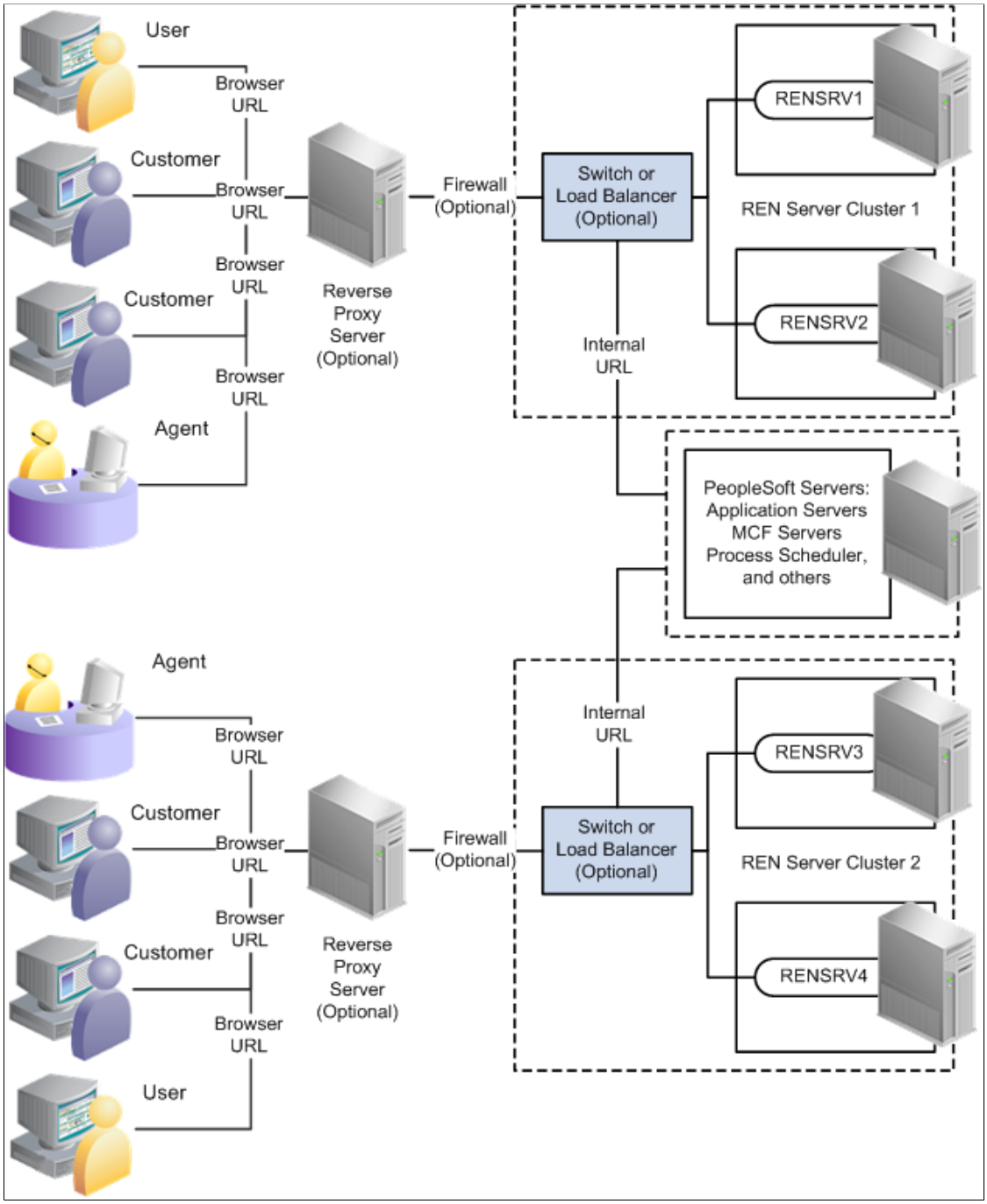
The REN server is a modified web server using the HTTP 1.0 or 1.1 communications protocol. Communication with MCF server processes and browser windows is bidirectional because they maintain persistent connections to the REN server. Events can be sent proactively to browser windows without polling or page refreshes.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

REN Server Failover, Scalability, and Security Configuration

Image: REN server configuration example

REN servers can be configured to support both failover and scalability, and should be protected with firewalls and appropriate security measures, as illustrated in the following diagram.



 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

REN Server Failover

Although the REN server is integrated into an application server domain, it is not a standard PeopleTools server process (it has no database connection) and therefore has a separate failover mechanism. Two scenarios exist for failure recovery:

- For a standalone REN server, Oracle Tuxedo restarts the server if it fails.

  MCF servers and consoles reconnect to the REN server. However, any active browser sessions (such as MCF chat) are interrupted until a connection can be reestablished between the chat console and the restarted REN server.

- For clustered REN servers, each REN server in the cluster is a peer that mirrors the current state.

  This configuration has two advantages over a standalone REN server:

  - Clustered REN servers guard against hardware failure (provided that the clustered REN servers are on different host machines).
  - Active browser sessions are not lost.

REN Server Clusters

You can configure a REN server cluster with only one REN server member. However, a REN server cluster that is configured with two or more REN servers provides failover.

All REN servers in a cluster mirror each other and appear to external processes as a single URL. The REN server cluster must have an HTTP load balancer or switch as its front end. All connections with browsers and application server processes address the front end's URL. The load balancer should use an active standby content-switching rule to route all traffic to a designated REN server in the cluster. The front end selects an alternate member of the cluster only when the designated REN server fails to respond.

The REN server cluster maintains mirrored state in all members by relaying events with HTTP messages. Increasing the number of servers within a REN server cluster therefore does not address scalability issues. Clustering REN servers does not improve performance and may increase processing overhead and internal network traffic. The internal HTTP connections between cluster members should be high speed for best performance. Because of the overhead involved in synchronous cluster members, each member of a cluster can handle less load than a REN server in a cluster with only one REN server.

Note: In an environment in which multiple REN servers exist within a single cluster, the primary REN server sends synchronization data to the other members of the cluster. If any of these synchronization messages fail, then the primary REN server retries up to cluster_retry_count times. The minimum value of this parameter, cluster_retry_count, in psrenconfig.txt is 0, which means that the REN server does not retry.

If a REN server crashes, it does not rejoin the cluster because it would not be synchronized with the other clustered REN servers. The entire cluster must be shut down and rebooted to restore all members back to full participation.

Incoming cluster requests must eventually route to the front end's HTTP address. Queue servers and application servers use the cluster URL, which is typically set to be the URL of the front end. Browser clients make requests using the browser URL, which may be set to the front end, or to a server that

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

proxies to the load balancer. If browser transactions are encrypted with SSL, then the browser URL is an HTTPS address to a reverse proxy server or SSL accelerator.

Note: If you use SSL between the browser and REN server, then you must use a reverse proxy server or SSL accelerator, unless you have configured an SSL-enabled REN server.

Note: When clustering multiple REN servers, typically there is some performance degradation.

Note: Periodically, Javsacript-based clients to the REN server, such as our MCF and CTI consoles, will perform a clean up cycle. This clean up cycle, called a "tunnel refresh" will cause the client to disconnect temporarily from the REN server. Depending on the implementation of the client, this may cause a message to be presented to the user, and reconnection may or may not be automatic.

Understanding SSL-Enabled REN Servers

You can enable a secure channel of communication between the clients and the REN server by enabling SSL on the REN server using openssl. The SSL protocol runs above Transmission Control Protocol/ Internet Protocol (TCP/IP) and below higher-level protocols, such as HTTP and IMAP4. By using TCP/ IP on behalf of higher-level protocols, openssl allows an SSL-enabled server to authenticate itself to an SSL-enabled client, a client to authenticate itself to a server, and both machines to establish an encrypted connection.

This section discusses:

- Installing digital certificates.
- Authenticating server and client.
- Performance and scalability for SSL-enabled REN servers.

Installing Digital Certificates

REN servers require digital certificates to work in SSL mode. The servers pick up the certificates from the keystore in the PeopleTools database or an external keystore. The certificates must be imported into PeopleTools database from PeopleTools, Security, SecurityObjects, Digital Certificates. Certificates that are installed in the database will have a unique combination of certificate type and alias.

The certificate type that is used for the server should be of the type CERT, and the alias is <machine name>.<domain name>. When the certificate is configured with a unique alias name, it should be associated with the REN server that is SSL-enabled. The REN server loads its server certificate from the database at the start-up.

See Installing Digital Certificates.

Authenticating Server and Client

For server authentication, the server sends its certificate to the client as a part of the SSL handshake and the client authenticates by verifying the Certificate Authority (CA) of the certificate against its trusted keystore. When the REN server is configured for SSL, all clients must trust the CA of the server

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

certificate to participate in a successful communication. The keystore can either be in the database or an external keystore.

Client authentication verifies the clients's authenticity to participate in a communication with the server. When the REN server is configured for client authentication, all clients must supply a valid client certificate to participate in a successful communication.

All clients must use the REN cluster's HTTPS URL to communicate in the SSL mode. If the REN server is SSL only, access is denied to any client trying to communicate with a HTTP URL port. The browser-based clients, the application server client, and the REN Java clients should be configured appropriately to communicate with an SSL-enabled REN server.

Performance and Scalability for SSL-Enabled REN Servers

During an SSL transaction, the handshake is an added overhead that occurs. However, for every transaction, the handshake is done once to authenticate the server and the client. After authentication, the data is digitally signed, encrypted, and exchanged on an established session. For each console, authentication establishes a session only once, and no subsequent transactions inherit any overhead of authentication.

Configuring REN Server Security

This section provides an overview of REN server security configuration and discusses how to define permission lists for REN server access.

Understanding REN Server Security Configuration

Protect the REN server behind firewalls. A reverse proxy server can be used between browser clients and the REN server. Browser sessions can be SSL-encrypted by means of a reverse proxy server or hardware SSL accelerator.

Note: The security of your PeopleSoft system and configuration of load balancers, switches, and reverse proxy servers is beyond the scope of this document. Refer to your PeopleBooks for more information.

REN server access from browser clients is restricted to users who are currently signed in to PeopleSoft software with appropriate REN server permissions. You must enable single sign-in security to obtain REN server access. Permission to access REN server applications is granted on permission lists, which are in turn associated with security roles and user IDs. Clients lacking access permission receive a 403 Forbidden page from the REN server.

Note: REN server access requires that single sign-in be enabled.

Related Links

"Security Administration Overview" (PeopleTools 8.54: Security Administration)

"System and Server Administration Overview" (PeopleTools 8.54: System and Server Administration)

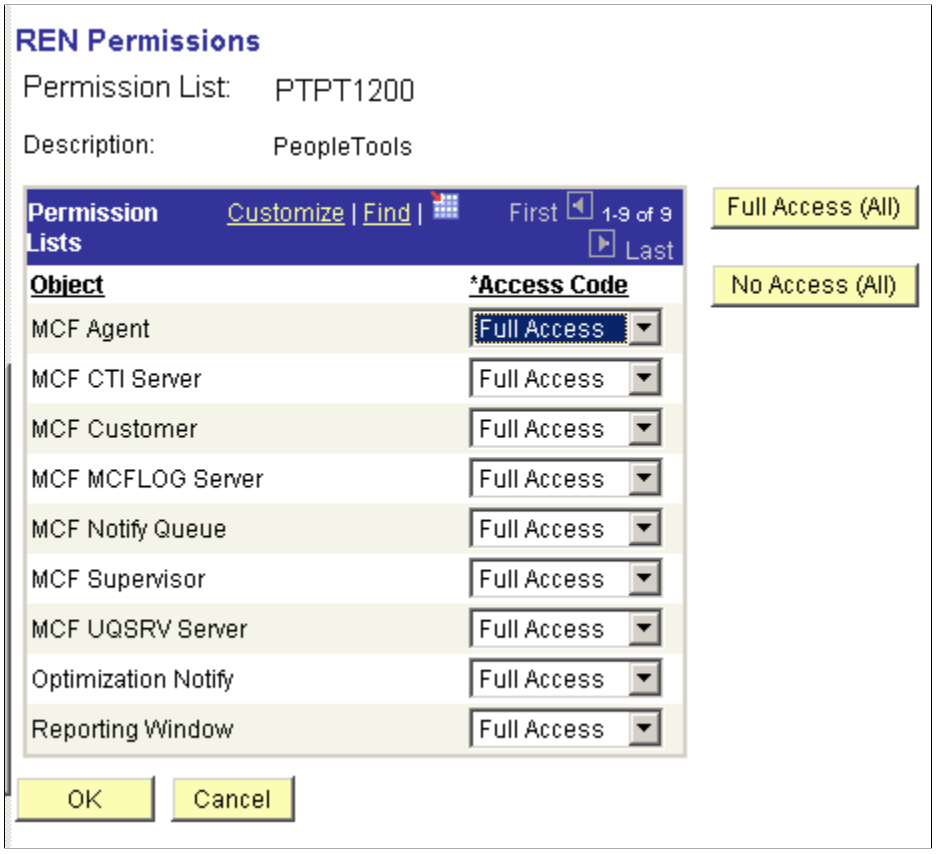
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Defining Permission Lists for REN Server Access

The following REN Permissions page shows the objects and permissions that are defined for permission list PTPT1200. You can create custom permission lists and define access to REN servers.

Image: REN Permissions page

This example illustrates the objects and permissions that are defined for a permission list on the REN Permissions page.



To define permission lists for REN server access:

1. Select PeopleTools, Security, Permissions & Roles, Permission Lists.
2. On the search page, search for and select your permission list.
3. On the Permission List page, select the PeopleTools tab.
4. Click Realtime Event Notification Permissions.
5. On the REN Permissions page, select your permissions.

   To enable REN server access for roles that are defined with the current permission list, select *Full Access* for each object that is required by the role. For example, users who require access to the MultiChannel Console must have Full Access defined for the MCF Agent object.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: To enable access to the Report-to-Window functionality, add WEBLIB_RPT to the Web Libraries page of the permission list, and set Reporting Window to Full Access on the REN Permissions page. Grant full access to the MCF CTI Server object only on the permission list that is assigned to the CTI server role. No other users should have MCF CTI Server access.
The user ID that is configured to start the Process Scheduler must have full access to the Reporting Window REN permission on at least one permission list for that user ID. If the user ID does not have full access to the Reporting Window, then the pop-up window stays in a status of queued.

Related Links

Required Security for PSMCAPI

Configuring REN Servers

To configure REN servers, use the REN Server (REN_SERVER_CMP) component.

This section provides an overview of REN server configuration options and discusses how to:

- Configure REN servers and SSL-enabled REN servers.
- Define REN servers.

Understanding REN Server Configuration Options

Depending on your requirements, choose one of two REN server creation and configuration options:

- To create a single REN server in a particular database using default configuration parameters, create an application server domain using PSADMIN.

  Event Notification may be enabled in the quick-configure menu. An associated REN server cluster is automatically created when Event Notification is enabled.

- To create additional REN servers in a particular database, configure each REN server as required on the REN Server Definition and REN Server Cluster pages.

  Then create the associated application server domains. Event Notification must also be enabled in the quick-configure menu.

When a REN server starts, it looks for configuration information in the database, using the application server domain name and host name as keys. If the associated configuration information exists in the database, the REN server uses it. If no such configuration information exists, the REN server is configured by defaults, which also configure a REN server cluster for each REN server. You can change the default REN server configuration by using the REN Server Configuration pages, but such changes do not take effect until the REN server starts up again.

Note: You can create only one REN server per application server domain.

This section discusses some possible REN server configurations that depend on domain server topology.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Simple Configuration: Mycompany.com

In this configuration, the REN server is on the host machine MachA, the REN server uses the default port number 7180, the domain name server (DNS) addresses the host machine as MachA.mycompany.com, and no SSL or reverse proxy server is involved:

Parameter	*Value*
PeopleSoft Pure Internet Architecture Authentication Token Domain	mycompany.com
Authentication Domain in REN Server Cluster Configuration	mycompany.com
REN Server Cluster Root Path	/psren
REN Server Cluster URL	http://MachA:7180
REN Server Browser URL	http://MachA.mycompany.com:7180

Simple Configuration with SSL-Enabled REN Server: Mycompany.com

In this configuration, the REN server is on the host machine MachA, the REN server uses the default port number 7143, and DNS addresses the host machine as MachA.mycompany.com. The REN server is SSL-enabled.

Parameter	*Value*
PeopleSoft Pure Internet Architecture Authentication Token Domain	mycompany.com
Authentication Domain in REN Server Cluster Configuration	mycompany.com
REN Server Cluster Root Path	/psren
REN Server Cluster URL	https://MachA:7143
REN Server Browser URL	https://MachA.mycompany.com:7143

Reverse Proxy Server with Non-SSL Configuration

This configuration includes a single REN server and a reverse proxy server. The reverse proxy server could be either a dedicated reverse proxy server or a web server with a proxy plug-in configured to redirect both PeopleSoft Pure Internet Architecture and REN server requests. The application server host machine is MachA, and the REN server uses its default port 7180. The reverse proxy server is on MachRPS using port 8080 for HTTP. The DNS server must recognize MachRPS.mycompany.com.

Parameter	*Value*
PeopleSoft Pure Internet Architecture Authentication Token Domain	mycompany.com

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameter	*Value*
Authentication Domain in REN Server Cluster Configuration	mycompany.com
REN Server Cluster Root Path	/psren
REN Server Cluster URL	http://MachA:7180
REN Server Cluster Browser URL	http://MachRPS.mycompany.com:8080

Reverse Proxy Server with SSL Configuration and Secure HTTP

For SSL, install certificates on the reverse proxy server, set the server to encrypt all communications, and use HTTPS URLs from the browser. In this example, the reverse proxy server uses port 8443 for SSL:

Parameter	*Value*
PeopleSoft Pure Internet Architecture Authentication Token Domain	mycompany.com
Authentication Domain in REN Server Cluster Configuration	mycompany.com
REN Server Cluster Root Path	/psren
REN Server Cluster URL	http://MachA:7180 **Note:** The cluster URL should not be a secure HTTP address if SSL is handled through a reverse proxy server.
REN Server Browser URL	https://MachRPS.mycompany.com:8443 **Note:** This is a secure HTTP address (HTTPS).

Note: If you use SSL between the browser and REN server, you must use a reverse proxy server or SSL accelerator.

Related Links

"Security Administration Overview" (PeopleTools 8.54: Security Administration)

Configuring REN Servers and SSL-Enabled REN Servers

Specify REN server configuration parameters based on your network topology and server arrangement.

Define the parameters for REN server configuration in three locations:

- Authentication token domain, set during PeopleSoft Pure Internet Architecture installation or in web profile configuration.
- REN server configuration parameters, specified in an application server domain using PSADMIN.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- REN server parameters, including cluster and browser URLs, set in the PeopleTools REN Server and REN Cluster components.

  Configuration parameters that are set in the REN Server and REN Cluster components override any defaults in PSADMIN.

Authentication Domain

The authentication domain tells PeopleSoft Pure Internet Architecture the internet domain name that browser clients use when accessing PeopleSoft applications across the internet. The token is required to comply with the same-origin security policy that is enforced by most browsers. The domain name that is specified in the REN Server Configuration page must be identical to the domain name that is specified as the authentication token domain during PeopleSoft Pure Internet Architecture installation.

If authentication domain is not set during PeopleSoft Pure Internet Architecture installation, define the authentication domain in web profile configuration to match the REN server configuration.

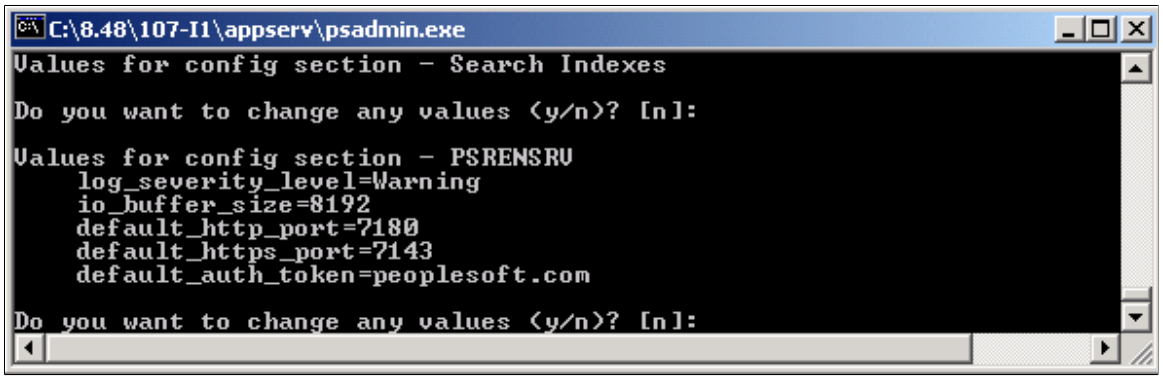
Note: You must specify the authentication token domain if you access the REN server and the PeopleSoft Pure Internet Architecture web server using different DNS names from the browser client (for example, if they are on different machines).

Configuring a REN Server and SSL-Enabled REN Server with PSADMIN

If necessary, you can specify parameters in the PSRENSRV section of the PSADMIN application server domain configuration, as illustrated in the following example:

Image: Configuring PSRENSRV in PSADMIN

This example illustrates the parameters in the PSRENSRV section of the PSADMIN application server domain configuration.



Specify parameters as described in the following table:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameter	***Default***	***Description***
log_severity_level	Warning	This is the logging level for the REN server. Select from one of the following log severity levels, from less to more logged data: Error, Warning, Notice, Debug. **Note:** Do not use Debug in a production environment.
io_buffer_size	8192	This is the TCP buffer size in bytes that is used for serving content. Do not exceed a value of 65536. If the REN server is running on Microsoft Windows, change io_buffer_size to a minimum value of 56000.
default_http_port	7180	This is the REN server's HTTP port. The default value is *7180.* The default_http_port parameter takes effect only when a REN server starts up for the first time and the database does not already contain configuration information for the REN server. **Note:** After the HTTP port number that is assigned to the REN server has been established in the database, the only way to change it is on the REN Server Definition page. Editing the port number in the psappsrv.cfg file does not overwrite the value that is stored in the database.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameter	*Default*	*Description*
default_https_port	7143	This is the REN server's HTTPS port for SSL-enabled REN server. The default value is *7143*. **Note:** The https port is used only when the REN server is SSL-enabled. The default_https_port parameter is configured in psappsrv.cfg and is used when a SSL-enabled REN server starts up for the first time. **Note:** To change the default https port, use the REN Server Definition page. Changing the SSL Port requires the reboot of the REN server.
default_auth_token	example.com	This is the fully qualified domain name of the application server. This value should match the value of the web server's authentication domain. The default_auth_token parameter takes effect only when a REN server starts up for the first time and the database does not already contain configuration information for the REN server. When configuring the REN server parameters through PSADMIN, do not place a period or dot (.) before the default_auth_token value. For example, the parameter should read default_auth_token=example.com

Access logging is no longer enabled by default. In order to enable access logging, remove the leading #(hash) from this line in the “Modules to load” section of the psrenconfig.txt file.

```
#ns_param  nslog   ${bindir}/${prefix}rennslog${ext}
```

After specifying REN server configuration parameters, be sure to specify Y (Yes) when asked if you want event notification configured and MCF server configured. Boot this domain from the Domain Administration menu.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: Use PeopleSoft Pure Internet Architecture REN server definition and configuration pages to modify configuration parameters whenever possible. REN server configuration parameters that you make using PSADMIN are written to the psappsrv.cfg file in the application server directory. REN server configuration values that are found in the database override any values that are found in psappsrv.cfg.
Use static IP addresses for your web servers. If you use dynamic IP addresses (DHCP), ensure that the domain name server (DNS) can map fully qualified domain names to the dynamic IP addresses.
If you are using Microsoft Internet Explorer internet security zones, include both the web server and REN server addresses in the same security zone; alternatively, exclude both addresses from security zones.

Socket Binding

The REN server listens on the port that is defined in the REN Server Definition page, which is by default *7180*. However, the host name to which the REN server binds is determined by information in the psrenconfig.txt file for each application server domain. If the host machine contains multiple network interface cards (NICs), then the REN server binds by default to only one NIC, which is given by `uname()` on Unix, or `GetComputerName()` on Microsoft Windows.

To bind a REN server to a specific NIC, manually edit psrenconfig.txt for the appropriate application server domain, changing both set address and set hostname to the IP address and locally-known host name of the NIC. For example:

```
set address   192.168.10.1
set hostname  hostsrv.example.com
```

Note: If you enter an invalid IP address in the psreconfig.txt file, the REN server may not start correctly. Check the REN server log for error messages that identify the issue.

Note: To configure a REN server with a virtual machine name, manually enable the #ns_param javascript_kn_server parameter in the psrenconfig.txt by removing # (hash) from the parameter.

Configuring TCP_NODELAY

The parameter, TCP_NODELAY in psrenconfig.txt controls whether to disable the TCP Nagle algorithm on the TCP packets sent by the REN server. Two instances of TCP_NODELAY are available in psrenconfig.txt. TCP_NODELAY in the nssock section is used by non-SSL REN servers, and the instance in the nsopenssl section is used by SSL-enabled REN servers. The TCP Nagle algorithm is generally enabled by default and inserts a short delay before sending small TCP packets. This helps prevent network overload.

If TCP_NODELAY is set to 0, the TCP algorithm acts normally. This is the recommended configuration for most applications. However, for certain CTI applications, this parameter must be set to 1 to improve performance. If TCP_NODELAY is set to 1, the TCP Nagle algorithm is disabled on operating system platforms that support disabling this feature.

Defining REN Servers

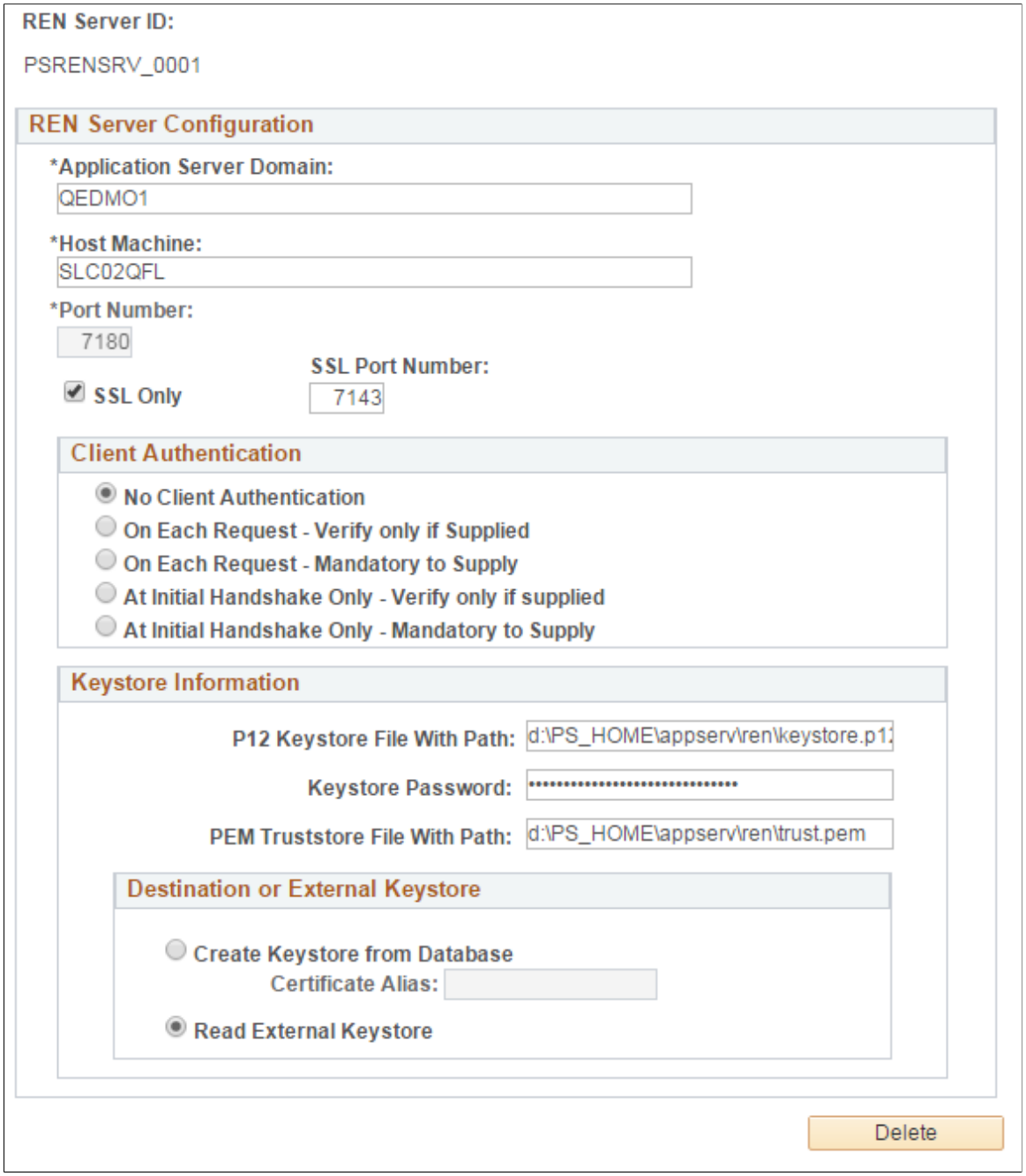
Access the REN Server Configuration page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, REN Server Configuration, REN Server Definition

Image: REN Server Configuration page

This example illustrates the fields and controls on the REN Server Configuration page. You can find definitions for the fields and controls later on this page.



Application Server Domain Enter the application server domain that is serving this REN server.

Host Machine Enter the name of the host machine on which the specified application server domain runs.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

This entry requires the host machine name, not its DNS name. However, the host machine name may need to be fully qualified, for example, machineA.example.com. On a Unix machine, determine the host name by running `uname -a`. On a Microsoft Windows machine, determine the host name by running `hostname` at a command prompt.

Port Number

Enter the HTTP port number on which this REN server is addressed.

Change the HTTP port value if multiple REN servers are running on the same host machine to avoid port conflicts.

SSL Only

Select to enable SSL on REN server.

Note: If this option is selected, you must enter the SSL port. If you want to enable SSL on the REN server, ensure that you create and configure a Renserver certificate before defining a REN server.

SSL Port Number

Enter the HTTPS port number on which this SSL-enabled REN server is addressed.

Client Authentication

Select an option to determine the level of client authentication.

Note: If the browser is configured for client authentication pop-up or the browser has more than one certificate configured, the SSL session ends if the user fails to provide the certificate within three heartbeats. To avoid such a session time-out, the user must either accept the client certificate within a heartbeat or increase the session time-out value in psrenconfig.txt.

The Client Authentication values are described in a subsequent table.

P12 Keystore File With Path

Enter the absolute path where the PKCS #12 keystore will be created or is located.

Keystore Password

Enter a password for the internal or the external PKCS #12 keystore. The password is internally encrypted.

PEM TrustStore File With Path

Enter the absolute path where the TrustStore will be created or is located.

Note: The trusted certificates are the root certificates, which are different from the certificate chain.

Create Keystore from Database

Select to use stored certificates from the Digital Certificates page.

Certificate Alias

Select a certificate alias to be used as a server certificate by the SSL-enabled REN server.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: The certificate alias is stored in the PSKEYDB, PSCERTDB, and PSREN records.

Read External Keystore

Select to use an external PKCS #12 keystore to read the Private Key, the Certificate Chain, and the Trusted Certificates. If you want to use an external Java keystore, you must configure the external Java keystore before you define the REN server on this page.

See Configuring External Keystore in REN Server

Note: By default, the Read External Keystore radio button is not selected, that is, the application server, at boot up, reads the Private Key, the Certificate Chain, and the Trusted Certificates from the database and creates a PKCS #12 keystore at the location entered in the P12 Keystore File With Path field.

Note: When you select the Read External Keystore radio button, the application server, at boot up, streams the path and password of the external PKCS #12 keystore to the REN server. The path of the external keystore is entered in the P12 Keystore File With Pathfield and the password is entered in the Keystore Password field. Subsequently, the REN server, at boot up, loads the external PKCS #12 keystore by using the password.

The following table shows the client authentication values:

Parameter	*Flag Value*	*Description*
No Client Authentication	0	Client authentication is disabled.
On each Request-Verify only if Supplied	1	Client authentication is enabled The server sends a client certificate request to the client. Verification happens only if the certificate is provided. If the verification process fails, the TLS/SSL handshake is immediately terminated. If the client does not return any certificate, SSL communication still continues
On Each Request-Mandatory to Supply	3	Client authentication is enabled and mandates that the client provide the certificate. If the client does not return a certificate, the TLS/SSL handshake is immediately terminated with a handshake failure alert. If the client returns a certificate, it is verified. The communication fails if the verification fails.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameter	*Flag Value*	*Description*
At Initial handshake Only-Verify only if Supplied	5	Client authentication is enabled and requests a client certificate on the initial TLS/SSL handshake only. Verification happens only if the certificate is provided. If the client does not provide any certificate, SSL communication still continues. If verification fails, the TLS/SSL handshake is immediately terminated.
At Initial handshake Only-Mandatory to Supply	7	Client authentication is enabled and mandates that the client provide the certificate only in initial TLS/SSL handshake.

Configuring REN Server and SSL-Enabled REN Server Clusters

A cluster is typically a collection of REN servers among which the session information is replicated. You cannot add both SSL and non-SSL servers in a single cluster.

To configure REN server clusters, use the REN Cluster (REN_CLUSTER_CMP) component.

This section discusses how to:

- Cluster REN servers and SSL-enabled REN servers.
- Specify REN server ownership.
- Specify REN server cluster members.

REN server clusters address failover and scalability.

Defining a REN Server Cluster

A REN server serves requests only if it is a part of the cluster. If the REN server is SSL-enabled:

- All the member REN servers must be SSL-enabled REN servers.
- All the member REN servers must use the same server certificate.
- Both REN Server Cluster URL and REN Server Browser URL must start with HTTPS and use the HTTPS port.

Note: When the administrator changes the REN server to be in SSL mode, he or she must also ensure that the REN server is a member of SSL clusters only. In any given REN cluster, all REN servers that are members must be either SSL-only servers or non–SSL servers. For SSL-enabled REN servers, use SSL-enabled PeopleSoft Pure Internet Architecture.

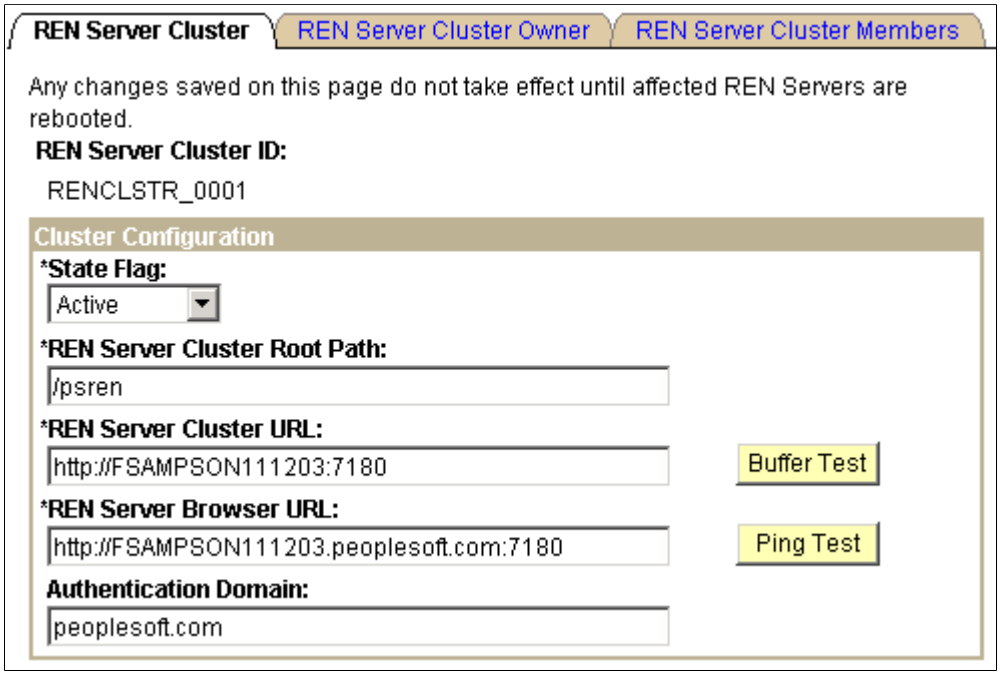
Access the REN Server Cluster page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, REN Server Configuration, REN Server Cluster, REN Server Cluster

Image: REN Server Cluster page

This example illustrates the fields and controls on the REN Server Cluster page. You can find definitions for the fields and controls later on this page.



By default, if you start a REN server from PSADMIN without configuring a REN server cluster, a cluster is created with a cluster ID RENCLSTR_000n

State Flag

Select *Active* or *Inactive.*

This field determines whether the cluster can receive new client requests. For scalability, configure multiple REN server clusters with the same ownership and set them to active status. Then the reporting window and customer chat applications will direct new client requests to a randomly chosen active REN server cluster. If all clusters are inactive, the client receives an error message.

If the cluster supports MCF servers, current chat sessions continue even after a cluster is inactive. But the MCF system does not route any additional requests to an inactive cluster.

Inactivate a cluster before deleting the cluster, or before removing a member REN server from the cluster. You can inactivate a REN server cluster without deleting the cluster.

REN Server Cluster Root Path

The default REN server cluster root path is /psren. Change this as required so that multiple REN server clusters are addressable through a single reverse proxy server.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Changes to the root path should also be reflected in the URL mapping of any reverse proxy server.

REN Server Cluster URL

The REN server cluster URL is the address that is used to reach the REN server cluster internally.

This is the URL that is used by internal processes. If the MCF cluster is served by a REN server cluster, the cluster URL is that of the switch or load balancer in front of the clustered REN servers. The cluster URL must be unique for each cluster. No two clusters can address the same cluster URL. Specify the cluster URL in the form <http://<DNS_machine_name>:<port>, where:

- <DNS_machine_name> is the server machine name that is recognized by your DNS.
- <port> is the REN server port number; the default value is 7180.

  This port number is the REN server port number or the port number of a proxy server, load balancer, or other front end.
- The protocol must be HTTP if the REN server is non-SSL; for an SSL-enabled REN server, the protocol must be HTTPS.

Note: Use the limit_http_responses parameter in the psrenconfig.txt configuration file to control the number of HTTP responses that will be sent to the browser following an HTTP POST. The default value of this parameter is *1*, meaning that each POST receives only one response. If limit _http_responses is set to *0*, the number of responses will not be limited, and each POST may receive more than 1 HTTP response from the REN server. Multiple responses can cause unpredictable browser, load balancer, and ping test behavior.

Buffer Test

Click Buffer Test to initiate a test of the REN servers' ability to break up and send a large file using multiple internal buffers.

The buffer test bypasses REN server security, and does not depend on specified domain names (authentication domain), so you can use it to verify that the REN server is running on the network.

REN Server Browser URL

The REN server browser URL is the address that is used by external clients and by agent chat to reach the application that is served by this REN server cluster.

The browser URL may be different from the cluster URL, which should not have to go through any firewall, reverse proxy server, or other outward-facing security barrier. If the REN server is reached through a load balancer, switch, or reverse proxy server,

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

specify the fully qualified URL of that device as accessed from the user's browser. The URL must be the address of the gateway machine (proxy server, load balancer, or SSL accelerator). Specify the address in the form http: or https://<DNS_machine_name>.<domain_name>:<port>, where:

- <DNS_machine_name> is the server machine name that is recognized by your DNS.
- <domain_name> is the fully qualified domain name that is recognized by your DNS.
- <port> is the REN server port number; the default value is 7180.

  This port number is the REN server port number or the port number of a proxy server, load balancer, or other front end.

Note: If the REN server is SSL-enabled, the browser URL must be HTTPS. Also, if you use an HTTPS address for PIA, you must use an HTTPS address for the browser URL.

Ping Test

Click to initiate a test of the REN server that is specified in the browser URL fields. Failure may indicate that a URL or authentication domain is incorrectly specified, the REN server is not running, or single sign-in is not implemented.

Authentication Domain

Enter the authentication domain. This must be the same as the authentication domain that is specified in the PeopleSoft Pure Internet Architecture installation or in the web profile configuration.

Specifying REN Server Ownership

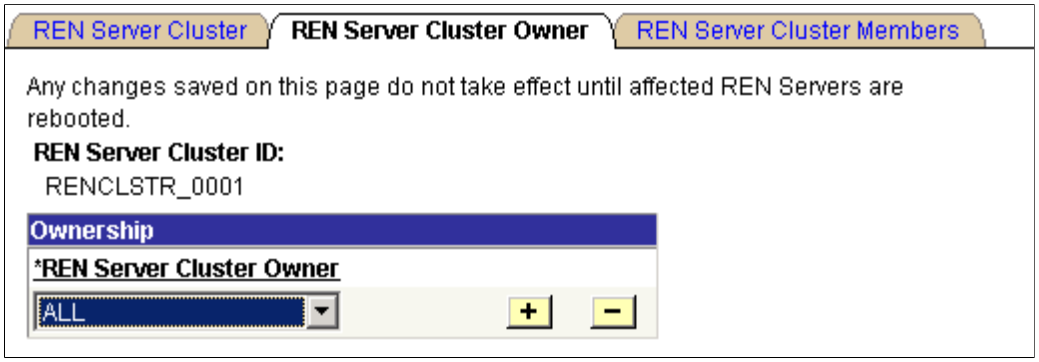
Access the REN Server Cluster Owner page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, REN Server Configuration, REN Server Cluster, REN Server Cluster Owner

Image: REN Server Cluster Owner page

This example illustrates the fields and controls on the REN Server Cluster Owner page. You can find definitions for the fields and controls later on this page.



REN Server Cluster Owner

Select the owner of this REN server cluster from the drop-down list box. Select from the following values:

- *All*
- *MCF*
- *Optimization*
- *Reporting*

Specifying an owner for a REN server cluster limits client access to that cluster. This is useful to ensure performance under load.

Specifying an owner for a REN server cluster also supports security. For example, an MCF cluster can be created only on a REN server cluster that is owned by MCF or ALL.

Specifying REN Server Cluster Members

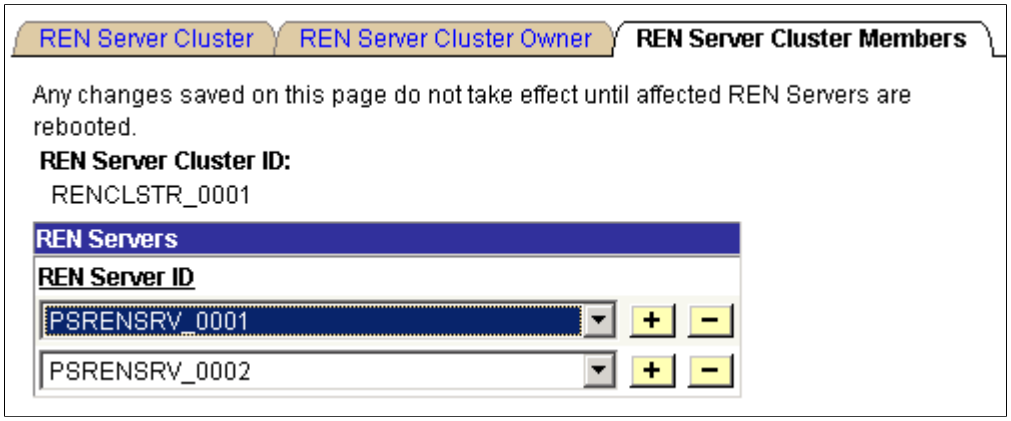
Access the REN Server Cluster Members page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, REN Server Configuration, REN Server Cluster, REN Server Cluster Members

Image: REN Server Cluster Members page

This example illustrates the fields and controls on the REN Server Cluster Members page. You can find definitions for the fields and controls later on this page.



REN Server ID Select a REN server from the drop-down list box.

Each REN server can belong to only one REN server cluster.

Configuring a Reverse Proxy Server with a REN Server

This section provides an overview of reverse proxy server (RPS) configuration and provides examples.

Understanding RPS Configuration

Production PeopleSoft installations may configure the REN server behind an RPS. The RPS isolates the REN server and other web servers from the open internet, provides SSL session handling, and presents a single-server origin to outside clients. PeopleSoft customers may put REN servers and PeopleSoft Pure Internet Architecture web servers behind one RPS, or just REN servers.

These examples assume that:

- You have installed the current PeopleTools release on both host machines.
- You have configured a web server using the default parameters on the first host machine.
- You have configured a REN server using the default parameters on the first host machine.

See Understanding REN Server Configuration Options.

Example: Configuring a WebLogic RPS for a REN Server on Another Host Machine

This example presents one possible configuration for a REN server running on one host machine and installing an RPS to run on a second host machine, using Oracle WebLogic . The RPS redirects clients to both a REN server and to the PeopleSoft Pure Internet Architecture web server.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

To configure an RPS for a REN server on another host machine:

1. Install a new web server domain on the second machine.

   Name the domain *rps*.

   Configure the following values:

   - AppServer Name: <*application_server_machine_name*>
   - JSL Port: *9999*

     The RPS will not make Jolt connections.
   - HTTP Port: *8080*
   - HTTPS Port: *8443*

2. Start the new web server.

   Navigate to <PIA_HOME>\webserv\rps, and run `startPIA.cmd`.

3. Sign in to the WebLogic Server Administrative Console for the *rps* web server.

   Access the WebLogic Server Administrative Console at http://<webserver>:<port>/console (for example, http://localhost:8080/console).

   When prompted for a user name and password, specify the WebLogic system ID and password. If you've followed the default WebLogic Server install, the ID and password are *system* and *password*.

4. Using the console's hierarchical navigation, navigate to rps, Deployments, Applications, PeopleSoft. Select the Targets tab.

   Clear the PIA option.

   Click Apply.

5. Using the console's hierarchical navigation, navigate to rps, Deployments, Web Application Modules, HttpProxyServlet. Select the Targets tab. Select the PIA option. Click Apply.

6. For better web server performance, navigate to rps, Servers, PIA. Select the Protocols tab, select the HTTP tab, and set both Duration and HTTPS Durationto *120* secs.

7. Stop the *rps* web server.

   Navigate to <PIA_HOME>\webserv\rps and run `stopPIA.cmd`.

8. Configure RPS parameters for the *rps* server.

   Locate the file web.xml at PIA_HOME/webserv/rps/applications/HttpProxyServlet/WEB-INF.

   Edit web.xml in a text editor, changing the WebLogic port and WebLogic host from *8080* to *80* (the value *8080* is a default value that is derived during installation of the domain *rps*). For example:

```
<init-param>
 <param-name>WebLogicPort</param-name>
 <param-value>80</param-value>
```

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

```
 <description>HTTP listen port of WebLogic PIA/PORTAL server.</description>
</init-param>
```

To specify the associated REN server, (which is on another machine), edit web.xml, changing the REN server host machine, port, and root URL from their default RPS values. For example:

```
<init-param>
 <param-name>WebLogicHost</param-name>
 <param-value>MACHINE_2</param-value>
 <description>Hostname of REN server.</description>
</init-param>
<init-param>
 <param-name>WebLogicPort</param-name>
 <param-value>7180</param-value>
 <description>Listen port of REN server.</description>
</init-param>
```

Another example is:

```
<servlet-mapping>
 <servlet-name>RENHttpProxyServlet</servlet-name>
 <url-pattern>/psren/*</url-pattern>
</servlet-mapping>
```

9. Reboot the RPS web server.

   Navigate to <PIA_HOME>\webserv\rps, and run `startPIA.cmd`.

10. (Optional) Configure and enable SSL on the RPS machine.

Note: When using Apache 1.3.x or 2.0.x RPS, you must configure the kn_response_flush_override and the flush_rps_buffer_size_for_knjs parameters in the psrenconfig.txt file. If you are using Apache 1.3.x, set both of these parameters to *4096*. If you are using Apache 2.0.x, set both parameters to *8192*. Apache needs both parameters present with the same buffer size. The kn_response_flush_override parameter flushes a message, while the flush_rps_buffer_size_for_knjs parameter flushes the stay-alive.

Note: Using WebLogic as a reverse proxy server is not recommended for a production system.

Note: The WebSphere plug-in for Microsoft IIS does not work as a RPS for the REN server.

Configuring Apache-based Reverse Proxy Servers for a REN Server

Apache-based proxy servers vary widely in configurations, here we present an example configuration. The configuration files for your environment may be quite different.

To proxy for RenServer, find and edit the httpd.conf configuration file. Make the following modifications to the file:

1. Move the line LoadModule proxy_module modules/ApacheProxyModule.dll to the bottom of the file.

2. Comment out the line AddModule mod_proxy.c.

3. Add the following five lines after LoadModule proxy_module:

```
<IfModule mod_proxy.c>
  ProxyRequests Off
  ProxyPass /psren http://machine:7180/psren
  ProxyPassReverse /psren http://machine:7180/psren
</IfModule>
```

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

4. Reboot your webserver and reverse proxy server.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 6

Configuring PeopleSoft MCF Servers and Clusters

Understanding PeopleSoft MCF Server and Cluster Architecture

This section discusses:

- PeopleSoft MCF server configuration.
- PeopleSoft MCF cluster architecture.
- Queue server and queue server failover.
- Logical queues and physical queues.
- MCF log server and log server failover.
- Queue server scalability.
- Recommended configurations.

PeopleSoft MCF Server Configuration

PeopleSoft MCF depends on processes that are configured and booted as part of an application server domain. Configure MCF processes (servers) through PSADMIN, along with other processes in each application server domain.

Note: The REN server process can be used by applications that are separate from the queue server and MCF log processes. In this case, you can configure the application server domain for event notification without creating the MCF servers.

After considering performance and failover issues, the MCF system administrator provides configuration information that describes the arrangement of queues, domains, queue server processes, REN server processes, MCF processes, and URL addresses.

REN server processes are configured on PeopleSoft Pure Internet Architecture pages before or after being initiated in an application server domain. Each queue server process is uniquely identified in the system by the combination of the machine name, domain subdirectory name, and process identifier. MCF log processes use the same queue-server identification scheme.

Both the queue server and the MCF log server are REN Java clients. If the REN server is configured to accept only SSL connections, then you must configure SSL certificates for the queue server and MCF log server.

See Installing Digital Certificates

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Related Links

Understanding REN Servers

PeopleSoft MCF Cluster Architecture

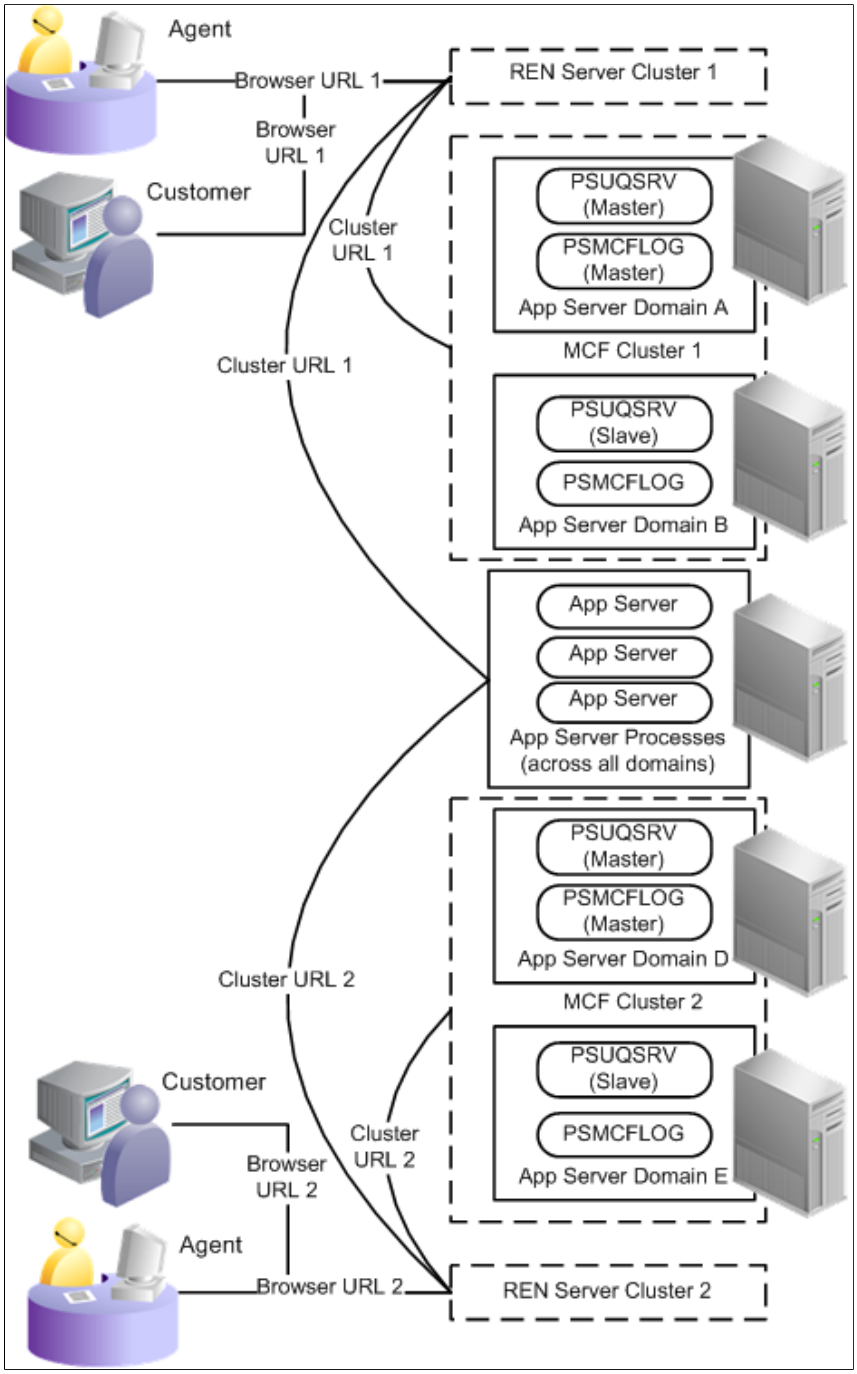
In a PeopleSoft system, all application server processes, including MCF servers and REN servers, belong to an application server domain. Each domain can have only one REN server process (PSRENSRV), one queue server process (PSUQSRV), and one MCF log server process (PSMCFLOG). Domains can

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

be redundantly clustered to provide failover. Logical queues can be serviced by multiple clusters for scalability. Support for scalability and failover is integrated into the configuration process.

Image: PeopleSoft MultiChannel Framework cluster architecture

The following diagram illustrates MCF cluster architecture.



Queue Server and Queue Server Failover

The queue server is a server process in the PeopleSoft application domain that routes email, chat, and generic tasks to the agent based on the agent properties, such as state and skill set, and task routing properties, such as priority, language, and cost.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The queue server process (PSUQSRV) is a Tuxedo-managed server with a standard PeopleSoft database connection. Each queue server process is the central routing point for one or more physical queues. The queue server maintains state information for work requests, work in progress, agent availability, and agent workload. Queue server state is written to database records, except for the assignment of chat to agents.

The queue server can recover from a crash because most of its state is written to the database. When a queue server reboots, it checks the database and loads state information for open work tasks. The queue server connects to its REN server and issues restart queries to each console so that it can rebuild agent assignment information, which may have changed while the queue server was down.

Although a single queue server can recover state after recovering from software failure, this does not guard against hardware failure. Multiple queue server processes running on multiple host machines and configured in a cluster provide failover for hardware failure. Unlike the REN server cluster, the clustered queue servers operate as one master and many slaves. The master handles all routing decisions, while slave processes monitor the master and step in only if it fails. Any rebooted queue server rejoins the cluster in a slave role. Any slave that is promoted to master loads state from the database and issues queries to consoles as if it were the only process in the cluster.

Each queue server process follows a fixed procedure to ensure that the cluster has at most one functioning master. Database locks eliminate possible race conditions, and the master periodically writes a timestamp to indicate its health. The masterinterval parameter controls the frequency at which the master process must update the timestamp in the cluster table. The masterinterval parameter corresponds to the maximum time after a master queue server fails before another queue server process takes over. Minimizing this value provides rapid failover response time but also requires frequent database updates.

See Tuning Cluster Parameters.

Each queue server must be part of an MCF cluster, and each MCF cluster must include at least one queue server. An MCF cluster of only one queue server provides no redundancy against hardware failure.

Create a queue server that starts when an application domain is started by selecting MCF servers from the quick-configure menu during application domain configuration.

In summary, configure queue servers to provide hardware failover. Each queue server is part of an MCF cluster. To support hardware failover, distribute master and slave queue servers over multiple hosts. Every queue server in an MCF cluster communicates with the same REN server ID. Therefore, REN server failover is also crucial.

The first queue server that places a valid master entry for itself in the cluster table becomes the master queue server. In most cases, the master queue server is the first queue server started. No configuration parameter exists to designate master or slave queue server within a cluster.

After an MCF cluster's master queue server is established, all other cluster members become slave queue servers. If the slave queue servers within a cluster detect a failure of the master queue server, the remaining slave servers compete to become the master queue server. If the master queue server reboots before a slave takes over, the master queue server also competes. No configuration parameter exists to designate priority among slave servers.

See Understanding REN Servers.

Logical Queues and Physical Queues

PeopleSoft MultiChannel Framework enables the configuration of both logical and physical queues.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

A logical queue is an application-level queue that receives work requests (tasks) relating to an application area, such as chat requests regarding sales information, and routes them to agents that are capable of handling the work. For example, you might configure a logical queue called SALES for sales inquiries and another called SUPPORT for support issues.

Logical queues can be partitioned into physical queues for scalability. A physical queue is managed by a single MCF cluster. For scalability, the tasks that are enqueued on a logical queue are distributed by the framework among all available physical queues. For example, the SALES queue could be serviced by four MCF clusters across four physical queues: SALES1, SALES2, SALES3, and SALES4.

Each agent can be assigned to only one physical queue within each logical queue. Each agent can be assigned to multiple logical queues.

PeopleSoft MCF Log Server and Log Server Failover

The MCF log server (PSMCFLOG) is a Tuxedo-managed server that is similar to the queue server. Each MCF log server receives events that are sent by a REN server and is responsible for writing MCF events to the database.

The MCF log server logs events to PS_MCFUQEVENTLOG. By default, the log server does not log periodic state information broadcasts from the queue server to the MultiChannel Console. If you need to log these events, configure logging on the Cluster Tuning page. You can also configure the log server to log the contents of chat sessions. Chat session logging is deactivated by default. Logged chats are stored in PS_MCFCHATLOG.

If the MCF log server crashes, it resumes functioning immediately after restarting. When the first slave log server detects a failed master, it takes over as the master log server for the cluster. The new master log server again receives all base topics, but it does not log chat sessions that started or continued during the time that the original master log server failed. The new master log server does not log per-agent events for agents that were signed in at, or during, the time of the failure.

An MCF log server is created along with a queue server when you enable MCF servers during application server domain configuration. No specific log server configuration is available during domain configuration.

Queue Server Scalability

PeopleSoft MultiChannel Framework is scalable to support large-capacity call centers or other large organizations. The basic strategy is to divide the workload by spreading it over several MCF clusters. This is accomplished by creating multiple physical queues for each logical queue and spreading the management responsibility for each physical queue to separate queue server processes, preferably on multiple host machines. This technique should not be confused with failover protection, which also adds processes and machines. In failover, the added processes are clustered together and do not provide performance improvement.

Organize applications using PeopleSoft MultiChannel Framework around logical queues (for example, SALES queue and SUPPORT queue). Incoming work tasks are sent to a logical queue. PeopleSoft MultiChannel Framework then assigns the task to one of the corresponding physical queues. This assignment is random across the queues. The load across the servers is balanced by servicing only one physical queue per logical queue by single MCF cluster.

For example, a logical SUPPORT queue might be split into physical SUPPORT1 and SUPPORT2 queues such that work requests are randomly distributed between the two physical queues. Half the agents

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

receive from one queue and half from the other. This splits the workload evenly between the two queue server processes, while still presenting one logical SUPPORT queue to the application.

Recommended Configurations

Consider the following configuration options to ensure maximum reliability and scalability of your PeopleSoft MultiChannel Framework installation:

- Configure multiple MCF servers in a cluster across multiple host machines.

  This provides protection against single-point failures.

  Each MCF cluster requires a REN server cluster. Configuring multiple REN server clusters is functionally the same as configuring multiple MCF clusters for scalability. Inside a REN server cluster, configuring multiple REN servers is functionally the same as multiple queue servers for failover.

  See Configuring PeopleSoft MCF Clusters.

- Use REN server clusters only for failover.

  REN server clusters do not enhance performance.

- Split logical queues into more than one physical queue if more work is required on that queue than a single process or machine can handle.

- If an application server domain is likely to be restarted regularly for reasons that are not related to PeopleSoft MultiChannel Framework, configure PeopleSoft MultiChannel Framework in a separate domain.

  Regular restarting of MCF servers affects performance because the MCF servers must recover state when they are recycled or when a slave takes over from a master server.

Configuring PeopleSoft MCF Clusters

An MCF cluster is a group of multiple MCF-enabled application server domains in which all queue servers and log servers communicate with the same REN server cluster. Only one queue server and one log server in an MCF cluster are active at any one time. These are called the masters. The rest are dormant and redundant, and are called the slaves. If a master drops out of the cluster for any reason, the slaves elect a new master to take its place.

This section provides an overview of MCF cluster configuration and discusses how to configure MCF clusters.

Understanding PeopleSoft MCF Cluster Configuration

Each MCF cluster includes a minimum of one queue server and one log server communicating with one REN server. An MCF cluster is typically identified by the ID of the REN server cluster serving it. No configuration limit is placed on the maximum number of queue servers in an MCF cluster.

In MCF architecture, a chat client initiates contact with an agent through the designated external (browser) URL for a REN server. This URL can point to a load balancer, switch, reverse proxy server,

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

or other hardware or software directing requests through a firewall. The external URL is also known as the browser URL because it supports the MCF browser windows (agent chat, customer chat, and MultiChannel Console).

For communication of queue servers, log servers, and application servers with REN servers, for example, handling email requests behind a firewall, you can use an internal URL. Specify both internal (cluster) and external (browser) REN server URLs during MCF cluster configuration.

Note: If no security is implemented, the browser and cluster URLs may be the same.

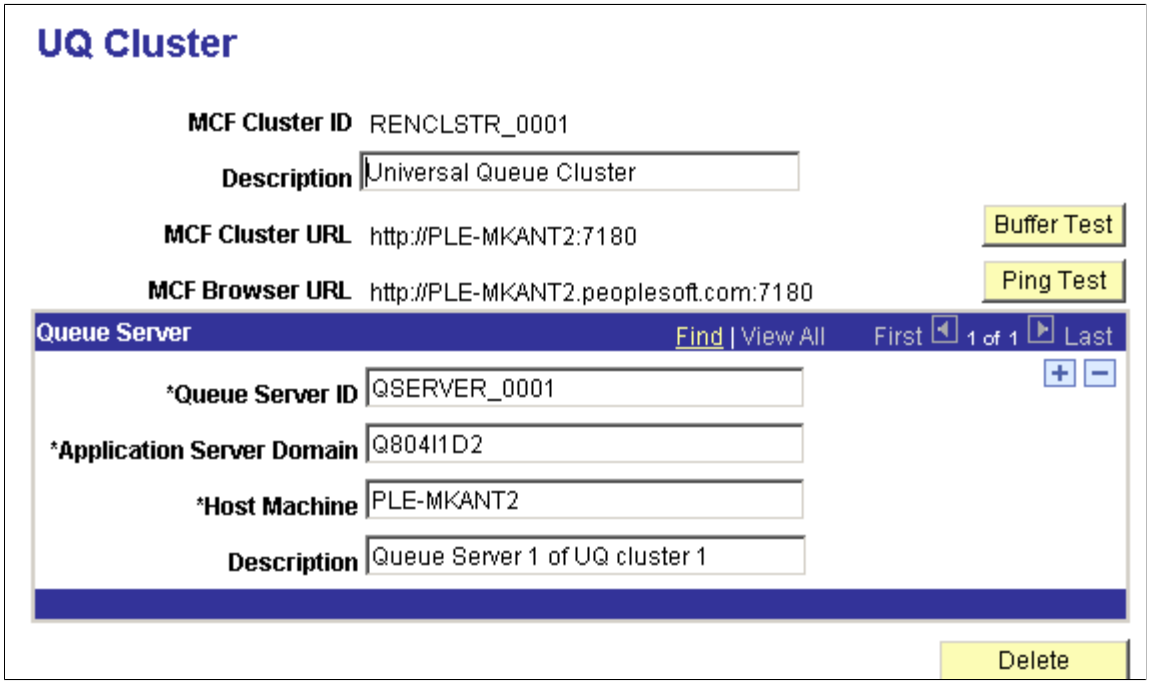
Configuring PeopleSoft MCF Clusters

Access the UQ Cluster page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Configuration, Cluster (MCF_UQCLUSTER)

Image: UQ Cluster page

This example illustrates the fields and controls on the UQ Cluster page. You can find definitions for the fields and controls later on this page.



MCF Cluster ID Displays the MCF cluster ID.

MCF Cluster URL Displays the URL for the REN server that serves this cluster. This is the URL that is used by internal processes. If the MCF cluster is served by a REN server cluster, the cluster URL is that of the switch or load balancer in front of the clustered REN servers.

Buffer Test Click to initiate a test of the REN server's ability to break up and send a large file using multiple internal buffers.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

	The buffer test does not depend on specified domain names, so you can use it to verify that RENSRV is running on the network.
MCF Browser URL	Displays the URL for a REN server cluster that serves this MCF cluster for external clients and for agent chat. The browser URL may be different from the cluster URL, which should not have to go through any firewall, reverse proxy server, or other outward-facing security barrier.
Ping Test	Click to initiate a test of the REN server that is specified in the URL fields. Failure may indicate that a URL is incorrectly specified.
Delete	Click to delete the entire MCF cluster. No active agents or tasks should be on the cluster.

Note: If the cluster's queue server configuration is changed, changes to the actual application server domains must be made manually using PSADMIN. For example, if a cluster member (queue server) is removed, the affected application server domain must be shut down and reconfigured (set the MCF Servers field to *No*) using PSADMIN. If the cluster URL is changed, all associated queue server domains must be shut down and rebooted.

Queue Server

An MCF cluster can consist of a primary queue server and any number of backup servers.

Each cluster requires a minimum of one queue server. The primary queue server is the first queue server started, and the remaining queue servers are backups. If the primary queue server fails, the system determines the subsequent primary queue server among the backups.

You can add a queue server to a cluster by adding a new row. Before removing a queue server, ensure that it is not the master, and then shut down its domain. Then click Delete (the minus sign).

If a domain is started with a queue server that does not belong to a cluster, the universal queue server and MCF log server poll the MCF configuration tables indefinitely until the queue server is assigned to a cluster.

Queue Server ID	Enter a unique identifier for each queue server to identify its entries in the database control tables. The log server process that is paired with this queue server uses this same ID to identify its entry in the log cluster table.
Application Server Domain	Enter the application server domain of which this queue server is a member.
Host Machine	Enter the name of the application server host machine.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 7

Configuring PeopleSoft MCF Queues and Tasks

Defining Queues

To define queues, use the MCF Queue (MCF_Q_CONFIG_CMP) component.

This section discusses how to:

- Define queues.
- Define chat responses.
- Define static push URLs.

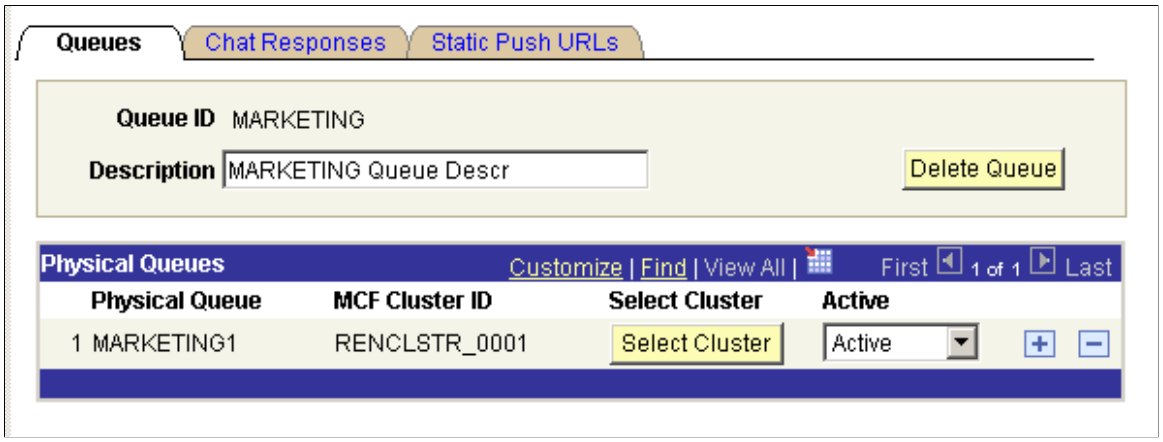
Defining Queues

Access the Queues page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Configuration, Queue, Queues

Image: Queues page

This example illustrates the fields and controls on the Queues page. You can find definitions for the fields and controls later on this page.



Queue ID Queue IDs must be alphanumeric, cannot end in a numeral, but can include underscore characters.

Delete Queue Click to remove this queue.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Deleting a logical queue means that no work or agents can be assigned to the queue, and the queue is removed from all agents' available queues.

You can delete a logical queue only if all of its constituent physical queues are inactive and have no tasks. Verify that no application code assigns tasks to a queue before deleting the queue. All agents that are assigned to the child physical queue will receive a message notifying them to sign out of their MultiChannel Consoles when the logical queue is deleted.

Physical Queue

Identifies that part of a logical queue that is serviced by the selected MCF cluster.

Physical queue IDs always end in a number, for example, SALES2. The physical queue identifier automatically increments by one for each physical queue that is added.

Physical queue IDs are automatically generated. The maximum number of physical queues is nine.

MCF Cluster ID

Identifies the MCF cluster that services this physical queue. Click Select Cluster to select from configured MCF clusters.

Each MCF cluster can service only one physical queue per logical queue. For example, an MCF cluster could service physical queue SALES1 or SALES2, but not both.

An MCF cluster can service multiple physical queues belonging to different logical queues. For example, an MCF cluster could service physical queues SALES1, MARKETING2, and COBOL1.

Select Cluster

Click to select a cluster from a list of available MCF clusters.

The selected MCF cluster services this physical queue. The primary queue server in this cluster manages tasks and agents that are assigned to this physical queue.

Active

Select *Active* or *Inactive* from the drop-down list box.

The queue server does not send new tasks to an inactive physical queue. Agents and existing tasks remain on an inactive physical queue. The physical queue must be active to receive new queued tasks.

Only inactive physical queues can be deleted from a logical queue. Inactivate a physical queue and complete or transfer all assigned tasks before deleting the physical queue.

Active and inactive status support *follow-the-sun* practices. For example, an organization could support SALES1 in the London office, and SALES2 in the San Francisco office when the London office is closed by activating the appropriate queues.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

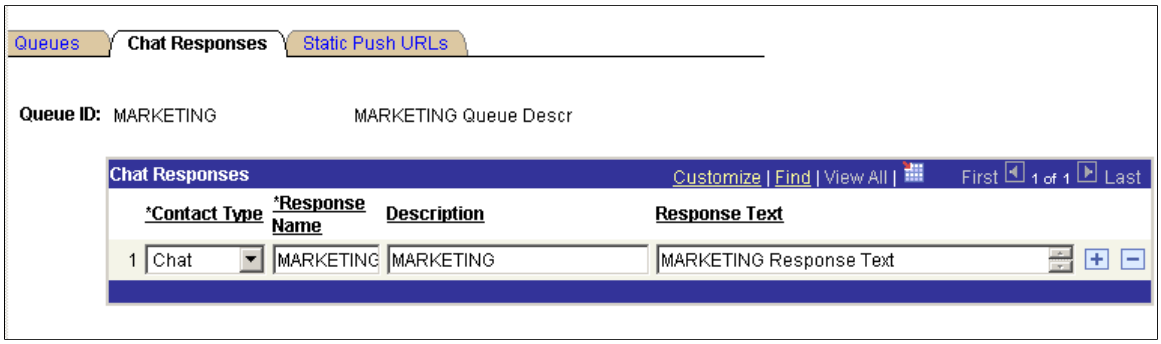
Defining Chat Responses

Access the Chat Responses page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Configuration, Queue, Chat Responses

Image: Chat Responses page

This example illustrates the fields and controls on the Chat Responses page. You can find definitions for the fields and controls later on this page.



Chat responses that are specified on this page are available to all agents who are signed in to this queue.

Contact Type

Select one of the following contact types:

- *Chat*
- *Email* (not currently supported)
- *Generic* (not currently supported)

Response Name

The response name appears in the agent's Template Messages drop-down list box for all agents who belong to a physical queue on this logical queue

All chat responses are downloaded when the agent launches the agent chat console by accepting a customer chat. Template messages are not available in collaborative chat.

Response Text

The specified message appears in the client chat window when selected by the agent.

Defining Static Push URLs

Access the Static Push URLs page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Universal Queue, Configuration, Queue, Static Push URLs

Image: Static Push URLs page

This example illustrates the fields and controls on the Static Push URLs page. You can find definitions for the fields and controls later on this page.

Static URLs enable the agent to push a predefined URL to the customer, which appears in a pop-up window for the customer. The agent can edit these URLs in the chat window after selecting them.

URL Name

Enter a name to identify the URL.

The URL name appears in the agent's Static URL drop-down list box for all agents who belong to this logical queue

URL Description

Enter a description of the URL.

This description appears only on this page to further describe this URL or, for example, its reason for inclusion.

URL

Enter the queue push URL.

The URL must include the opening http:// and any required parameters.

All static URLs that are defined for the queue are downloaded when the agent launches the agent chat console by accepting a customer chat. Static URLs are not available in collaborative chat.

If you send a static push URL that is a PeopleSoft Pure Internet Architecture URL, ensure that the recipient has permissions to access that portal, node, or page.

Configuring Tasks

To configure tasks, use the MCF Task (MCF_TASKCFG_CMP) component.

This section discusses how to configure tasks.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

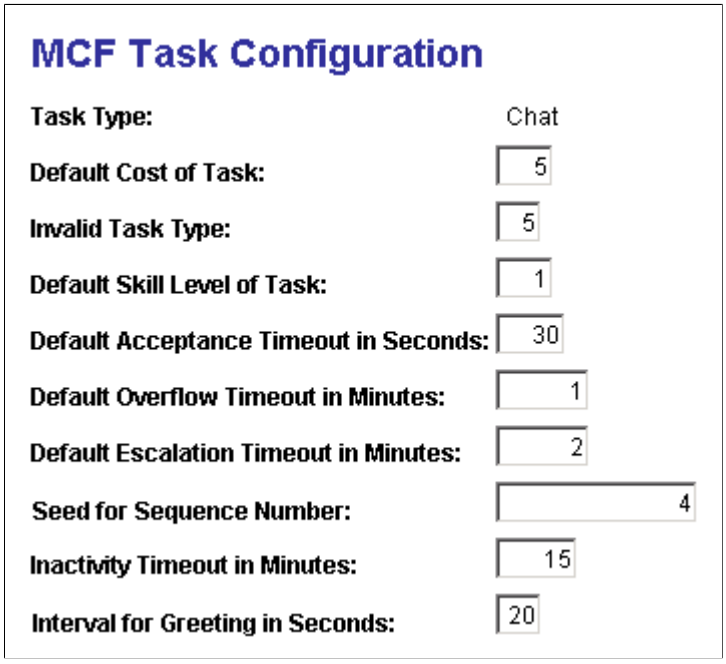
Configuring Tasks

Access the MCF Task Configuration page using either of the following navigation paths whichever is appropriate to you:

PeopleTools, MultiChannel Framework, Universal Queue, Configuration, Tasks (if you are using the CTI server) or PeopleTools, Multichannel Framework, Third-Party Configuration, Tasks (if you are using a third-party routing server).

Image: MCF Task Configuration page

This example illustrates the fields and controls on the MCF Task Configuration page. You can find definitions for the fields and controls later on this page.



Define values for different types of tasks that the queue server uses to assign tasks to appropriate agents and to manage tasks that are not accepted or closed within configurable time limits.

After you define or change values for a task, you must use the Refresh Task Properties button on the Cluster Notify page to propagate the changes to clusters. If you are working with MCF tasks, use the Cluster Notify page (MCF_CL_NOTIFY_PG) by navigating to PeopleTools, Multichannel Framework, Universal Queue, Configuration, Notify Cluster. If you are configuring third-party tasks, use the third-party Cluster Notify page (MCFTP_CL_NOTIFY_PG) by navigating to PeopleTools, Multichannel Framework, Third-Party Configuration, Cluster Notify.

Default Cost of Task

Enter the cost of the task.

Cost is a measure of the workload that each task places on an agent. The cost of a task is an estimate of the task's expected complexity and of the time that is required to resolve the task. The minimum value is *0,* and no maximum value exists.

The costs of tasks that are assigned to an agent are added up and evaluated against the maximum workload for each agent to determine whether the agent can receive additional tasks.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

For example, if an agent has a maximum workload of 100, and the default cost of a chat is 20, the agent can manage five concurrent chat sessions, assuming that the default cost is not overridden in the InitChat() built-in function call and that no other task types have been assigned.

Note: Although priority has no effect on voice tasks (which are not queued), voice task cost is included in calculating an agent's workload.

Default costs are:

- Chat: *5*
- Email: *2*
- Generic: *1*
- Voice: *10*

Default Priority of Task

Enter the priority of this task. A higher value means a higher priority. Tasks are ordered on a physical queue based on their assigned priority.

The minimum value is *0,* and no maximum value exists.

A queue server gives precedence to a task of higher-priority value over a task of lower-priority value when looking for an agent to assign the task to. This means that the queue server always assigns a task of priority 100 to a qualified available agent before it looks for an agent for a task of priority 10. If two tasks have the same priority, they are assigned in the order of their enqueue time.

The value that is specified here can be overridden in the EnQueue() or InitChat() built-in function call.

Note: Priority has no effect on voice tasks, which are not queued; however, voice task cost is included in calculating an agent's workload.

Default priorities are:

- Chat: *5*
- Email: *2*
- Generic: *1*
- Voice: *10*

Default Skill Level of Task

Enter the minimum agent skill that is required to handle this task.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The queue server assigns this task type to an available agent with the lowest skill level on that queue that is greater than or equal to the skill level that is required by the task.

The minimum value is *0,* and no maximum value exists.

The value that is specified here can be overridden in the EnQueue() or InitChat() built-in function call.

Default skill levels are:

- Chat: *1*
- Email: *1*
- Generic:*1*
- Voice : *2*

Note: Only the third-party routing server supports voice channel. The queue server does not route voice tasks.

Default Acceptance Timeout in Seconds

Specify the period of time that an agent has to accept an assigned task (to click the flashing icon on the MultiChannel Console). If the task is not accepted within this time, the task is reenqueued for assignment to another agent.

The queue server uses an algorithm to minimize reassignment of tasks that previously timed out to the same agent.

The value that is specified here can be overridden in the EnQueue() or InitChat() built-in function call.

Default acceptance timeouts are:

- Chat: *30*
- Email: *30*
- Generic: *30*
- Voice : *10*

Note: Only the third-party routing server supports voice channel. The queue server does not route voice tasks.

Default Overflow Timeout in Minutes

Specify the overflow time-out.

The overflow time-out is the time period that a queue server has to find an agent who accepts a task (click the flashing icon on the MultiChannel console). If the task is not accepted within this time, the task is removed from the queue and placed in the overflow table. This table can be managed from the Overflow Administration page.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The value that is specified here can be overridden in the EnQueue() or InitChat() built-in function call.

Default overflow time-outs are:

- Chat: *2*
- Email: *120*
- Generic: *120*
- Voice: *1*

Note: Only the third-party routing server supports voice channel. The queue server uses CTI to route voice tasks.

Default Escalation Timeout in Minutes

Specify the default escalation time-out.

The escalation time-out is the time period within which a task must be closed. If the task is not closed within this time, the task is removed from the queue and from the agent's accepted task list (that is, the task is unassigned) and the task is placed in the escalation table. This table can be managed from the Escalation Administration page.

Escalation time-out is valid on a chat session only after it is accepted by an agent, and has no effect on voice tasks (CTI).

The value that is specified here can be overridden in the EnQueue() or InitChat() built-in function call.

Default escalation time-outs are:

- Chat: *10*
- Email: *480*
- Generic: *480*
- Voice: *2*

Note: Only the third-party routing server supports voice channel. The queue server uses CTI to route voice tasks.

Seed for Sequence Number

Displays the current cumulative count of tasks of this type that are enqueued. Do not modify this value until it has reached its upper limit of 2,147,483,647.

This value does not apply for the voice task type.

Inactivity Timeout in Minutes

Specify the inactivity time-out

Inactivity time-out applies to chat only. The inactivity timeout is the time period within which an agent or customer must

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

participate in a chat. If the chat session is dormant for more than this time, the chat is terminated.

The default value is *15* minutes.

Interval for Greeting in Seconds Specify the interval, in seconds, over which the initial greeting appears in a customer chat window while the system searches for an available agent.

Note: All task parameters are delivered with sample values. Determine a range for these values that is appropriate to your business requirements. For example, task cost could vary over a range of 1 to 100 instead of 1 to 10.

Related Links

Notifying Clusters of Changed Parameters
Notifying PeopleSoft MCF Clusters of Changed Parameters for a Third Party
Creating Agents

Viewing the Cluster Summary

To view the Cluster Summary page, use the Cluster Summary (MCF_RSERV_CFG_CMP) component.

This section discusses how to view the Cluster Summary page.

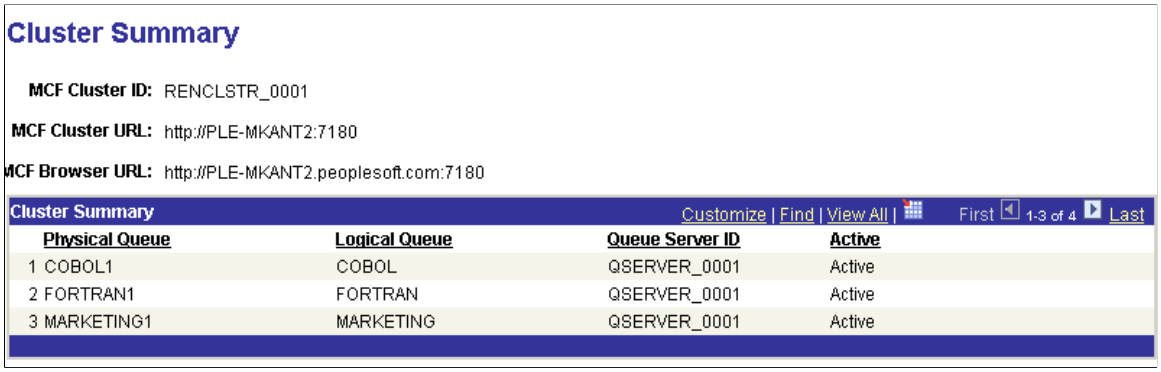
Viewing the Cluster Summary

Access the Cluster Summary page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Configuration, Cluster Summary

Image: Cluster Summary page

The Cluster Summary page displays details of the selected MCF cluster.



The Cluster Summary page displays the associated MCF cluster URLs and queue details for the selected MCF cluster. The information cannot be changed from this page.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Tuning Cluster Parameters

To tune cluster parameters, use the Cluster Tuning (MCF_SYSTEM_NV_CMP) component.

This section discusses how to tune cluster parameters.

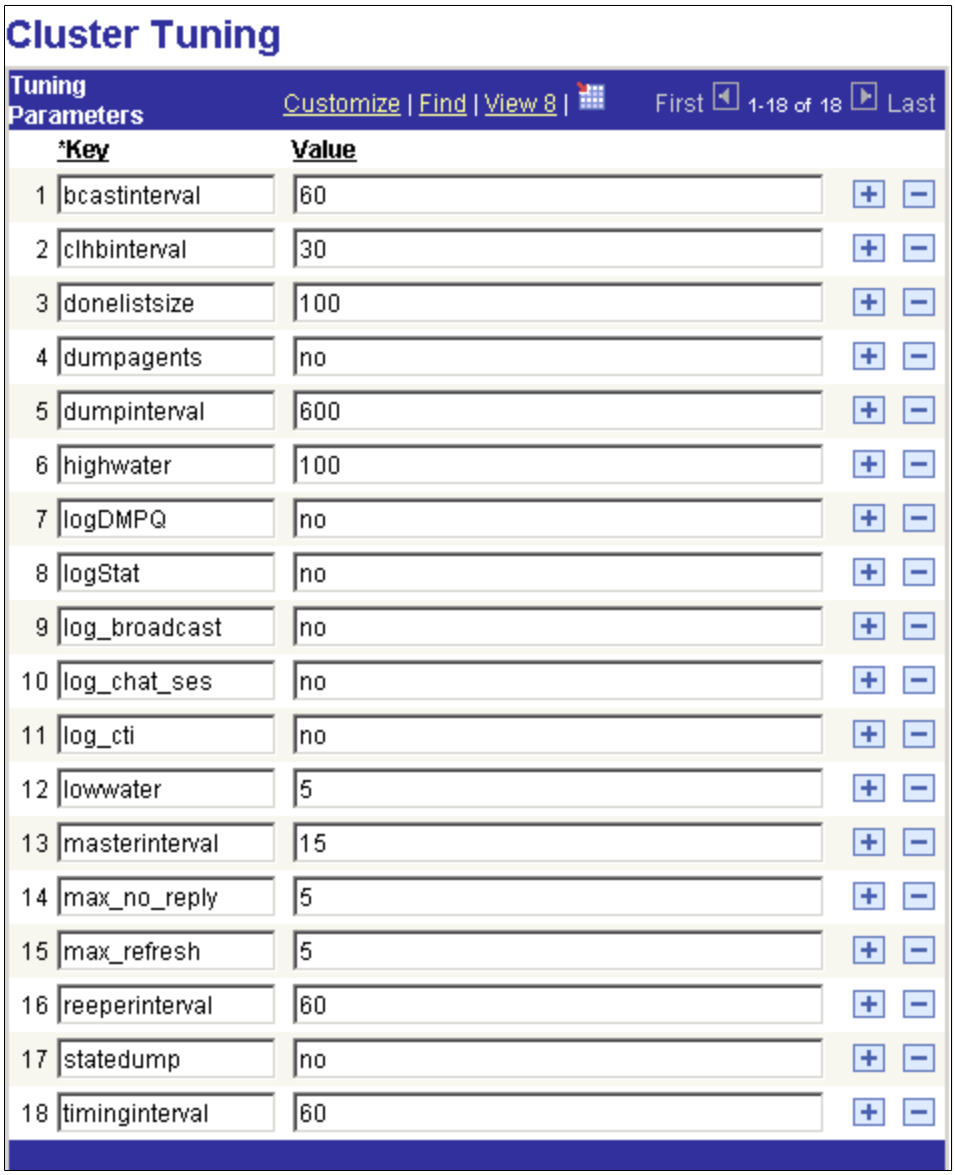
Tuning Cluster Parameters

Access the Cluster Tuning page using the following navigation path:

PeopleTools , MultiChannel Framework, Universal Queue, Configuration, Cluster Tuning

Image: Cluster Tuning page

This example illustrates the tuning parameters on the Cluster Tuning page.



Use the Cluster Tuning page to set MCF cluster parameters to optimize performance or enable logging.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

If you make changes to a cluster parameter, you must use the Notify Cluster page to propagate the changes.

See Notifying Clusters of Changed Parameters.

The following table lists the cluster tuning parameters you can modify and describes the default values and usage of each.

Key	***Default value***	***Usage***
bcastinterval	*60*	The interval, in seconds, after which the number of unassigned tasks per physical queue and the number of agents that are logged into each physical queue are broadcast to the MultiChannel Consoles for display next to the queue names. A smaller value provides more accurate queue statistics, but increases the load on the queue server and REN server. A larger value decreases queue server and REN server load, but also decreases statistical accuracy. The bcastinterval value also determines how frequently onStat2 event statistics are calculated. A smaller value provides updated statistics more frequently. This is a timing parameter. If you change the value of this parameter, you must use the Refresh Threshold/Timing Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters, Using the Monitor Queues Page and Sample Monitor - Queue Statistics Page, Using the Monitor Agents Page and Sample Monitor - Agent States Page.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Key	*Default value*	*Usage*
clhbinterval	*30*	The interval, in seconds, in which the queue server expects to receive a heartbeat from a connected JSMCAPI client. If the queue server receives no heartbeat during this interval, the queue server stops the client session. For a queue server, to avoid a session time-out for the client when CPU usage of the machine on which the client is running is high, increase the heartbeat interval of the client. **Note:** For third-party server, increase the heartbeat interval on the third-party server side to avoid a session time-out. This is a timing parameter. If you change the value of this parameter, you must use the Refresh Threshold/Timing Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters, Using and Demonstrating JSMCAPI.
donelistsize	*100*	The number of completed tasks that are stored in the list that is used to calculate average task duration. Configure the donelistsize value depending on the task volume that is encountered and the interval over which you want to monitor tasks. This is a timing parameter. If you change the value of this parameter you must use the Refresh Threshold/Timing Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters, Using and Demonstrating JSMCAPI.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Key	***Default value***	***Usage***
dumpagents	*No*	Enter *Yes* if the status of agent activity should be written to the database during the periodic state dumps. Logging agent status increases queue server load, but provides information about agent performance. This is a task parameter. If you change the value of this parameter, you must use the Refresh Task Properties button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters, Viewing the Agent State Summary.
dumpinterval	*600*	The interval, in seconds, after which the queue state is written to the database. A smaller value increases load on the queue server, but provides more frequent statistics. A larger value decreases load on the queue server, but provides less frequent statistics. A value of less than one minute will significantly reduce queue server performance. This is a timing parameter. If you change the value of this parameter, you must use the Refresh Threshold/Timing Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters, Viewing the Queue Server State, Viewing the Queue State Summary.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Key	***Default value***	***Usage***
highwater	*100*	The maximum number of persistent tasks that are retrieved from the database and cached in memory in the queue server. The highwater and lowwater mark values determine how often, and how many, persistent tasks should be read into memory. A higher value causes the queue server to retrieve more persistent tasks at one time, which results in less frequent access to the database, but also more tasks for the queue server to manage, slowing performance. A lower value speeds performance, but requires more frequent access to the database. If a large number of enqueued tasks cannot be routed by the queue server (for example, a call center handling a specific physical queue is offline), increase the highwater mark so that tasks that can be routed can fit into memory cache. This is a threshold parameter. If you change the value of this parameter, you must use the Refresh Threshold/Timing Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters.
logDMPQ	*No*	Enter *Yes* if you want PSMCFLOG to log REN server event notifications resulting from bcastinterval broadcasts. Only the event is logged, not its contents. This is a logging parameter. If you change the value of this parameter, you must use the Refresh Logging Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters, Viewing Event Logs.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Key	***Default value***	***Usage***
logStat	*No*	Enter *Yes* to log the statistics that are returned by the queue server for the onStat1 user and group events to the database. This is a logging parameter. If you change the value of this parameter, you must use the Refresh Logging Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters, Using and Demonstrating JSMCAPI.
log_broadcast	*No*	Enter *Yes* to activate logging of the broadcast messages that are sent. This is a logging parameter. If you change the value of this parameter, you must use the Refresh Logging Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters, Viewing Broadcast Logs.
log_chat_ses	*No*	Enter *Yes* to activate logging of the contents of chat sessions. This is a logging parameter. If you change the value of this parameter, you must use the Refresh Logging Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters, Viewing Chat Logs.
log_cti	*No*	Select *Yes* to activate logging of CTI events. This is a logging parameter. If you change the value of this parameter, you must use the Refresh Logging Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters, Logging CTI Events.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Key	***Default value***	***Usage***
lowwater	*5*	The minimum number of persistent tasks that are cached in memory in the queue server. When the lowwater value is reached, the queue server retrieves another batch of persistent tasks, up to the highwater value. The highwater and lowwater mark values determine when, and how many, persistent tasks should be read into memory. A higher value requires the queue server to access the database more frequently. A lower value can cause the queue server to run out of persistent tasks before refreshing its queue. The lowwater value should be greater than or equal to the maximum number of agents that are logged onto any physical queue at one time. This is a threshold parameter. If you change the value of this parameter, you must use the Refresh Threshold/Timing Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Key	***Default value***	***Usage***
masterinterval	*15*	The interval, in seconds, after which a cluster master updates its timestamp in its cluster tables. Slave clusters check the timestamp to determine whether the master cluster is still running. A lower value enables rapid discovery of a failed master server, but increases queue server overhead. A higher value reduces queue server overhead, but delays discovery of a failed master server. If only one queue server is configured for an MCF cluster, this value can be large. The masterinterval value also acts as a heartbeat interval for the master queue server connection to user consoles. This is a timing parameter. If you change the value of this parameter, you must use the Refresh Threshold/Timing Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters.
max_no_reply	*5*	Sets the maximum number of consecutive agent timeouts before the queue server automatically signs out the agent and sets the agent's console status as Assumed Unavailable. This is a timing parameter. If you change the value of this parameter, you must use the Refresh Threshold/Timing Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Key	***Default value***	***Usage***
max_refresh	*5*	Sets the maximum number of consecutive times that results are discarded when task queue is refreshed from the database if an intervening notification of new persistent tasks exists. This is a timing parameter. If you change the value of this parameter, you must use the Refresh Threshold/Timing Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters.
reeperinterval	*60*	The interval, in seconds, after which deleted tasks are cleared from memory in the queue server. A lower value increases queue server load but clears memory more frequently. A higher value decreases queue server load but clears memory less frequently. This is a timing parameter. If you change the value of this parameter, you must use the Refresh Threshold/Timing Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters.
statdump	*No*	Specify *Yes* to write queue server state to the database during the periodic state dumps. The state dump interval is set by the dumpinterval parameter. This is a task parameter. If you change the value of this parameter, you must use the Refresh Task Properties button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters, Using and Demonstrating JSMCAPI.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Key	*Default value*	*Usage*
timinginterval	*60*	The interval, in seconds, after which the database is checked for expired or overflowed persistent tasks. This parameter does not affect real-time tasks. A lower value increases queue server load but detects timed-out tasks more quickly. A higher value decreases queue server load but detects timed-out tasks less frequently. This is a timing parameter. If you change the value of this parameter, you must use the Refresh Threshold/Timing Parameters button on the Cluster Notify page to notify clusters of the changed parameter. See Notifying Clusters of Changed Parameters.

Notifying Clusters of Changed Parameters

To notify clusters of changed parameters, use the Cluster Notify (MCF_AD_NOTIFY_CMP) component.

This section discusses how to notify clusters of changed parameters.

Notifying Clusters of Changed Parameters

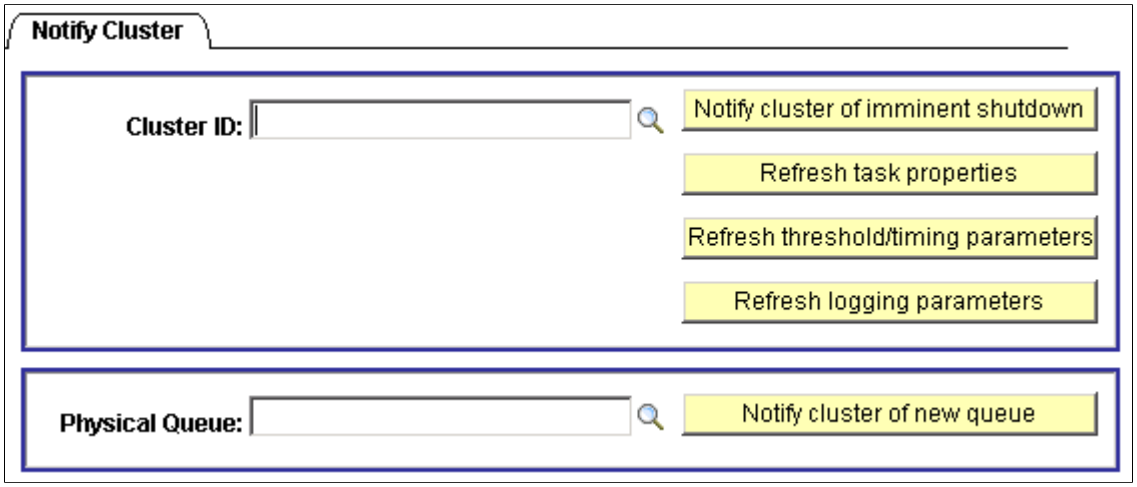
Access the Notify Cluster page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Universal Queue, Configuration, Cluster Notify

Image: Notify Cluster page

This example illustrates the fields and controls on the Notify Cluster page. You can find definitions for the fields and controls later on this page.



Use the Notify Cluster page to notify an MCF cluster of certain changes to its parameters or constituent queues, or that its application servers are being shut down.

For example, after changing MCF cluster parameters on the Cluster Tuning page, use the Notify Cluster page to refresh the tuning parameters.

Notify cluster of imminent shutdown Click to send a message to all agents who are signed in to the selected MCF cluster that they have been signed out.

Send this notification if the cluster's application servers are being shut down.

Refresh task properties Click to load task properties that have been changed on the Tasks page for the selected MCF cluster.

Refresh threshold/timing parameters Click to reload threshold and timing parameters that have changed on the Cluster Tuning page for the selected MCF cluster.

Refresh logging parameters Click to reload logging parameters that have changed on the Cluster Tuning page for the selected MCF cluster.

Notify cluster of new queue Click to notify the selected MCF cluster that the selected physical queue has been added.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 8

Configuring PeopleSoft MCF Agents

Defining Agents

To define agents, use the MCF Agent (MCF_AGENT_CMP) component.

This section discusses how to:

- Create agents.
- Specify languages that an agent supports.
- Personalize an agent's presence.

The previous three agent definition pages form the basis of agent configuration. Other parameters are optional.

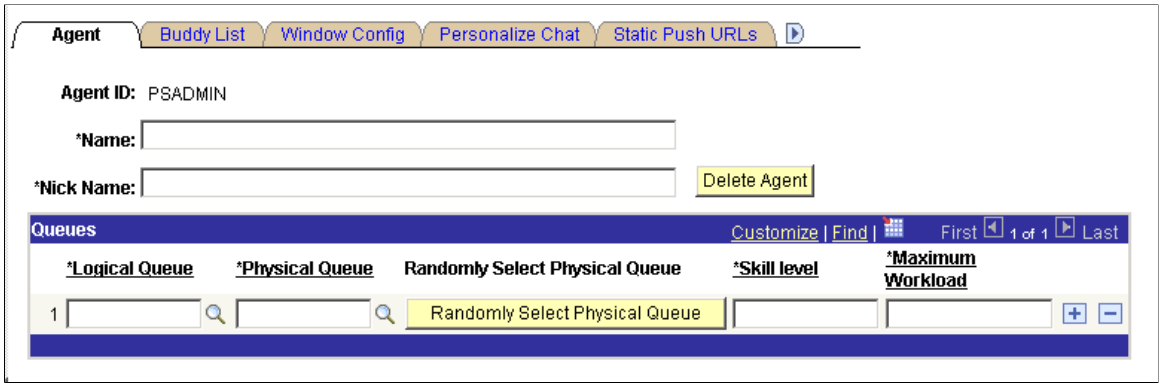
Creating Agents

Access the Agent page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Agent, Agent

Image: Agent page

This example illustrates the fields and controls on the Agent page. You can find definitions for the fields and controls later on this page.



Name

Enter the full name, in (lastname,firstname) format, of this agent.

The agent name appears in other agents' buddy lists.

Note: There is no space in between lastname, firstname.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Nick Name

Enter a short name for this agent.

The agent nickname identifies this agent in chat sessions and logs.

Delete Agent

An agent cannot be deleted if the agent still has accepted tasks on any queues to which the agent belongs. Before deleting an agent, ensure that the agent is logged off from all queues to which the agent is assigned.

Logical Queue ID

Enter the ID of a logical queue to which this agent is assigned.

Each agent can be assigned to more than one logical queue. An agent can log on to only one queue at a time from the MultiChannel Console.

Note: Do not overwrite the logical queue except when first creating an agent, as the agent's tasks may lose their assignments.

Physical Queue ID

Agents are randomly assigned to a physical queue when the logical queue is associated with the agent. An agent who services a logical queue logs on to a physical queue that is managed by a specific MCF cluster.

While an agent can service multiple logical queues, the agent can belong to only one physical queue per logical queue.

Note: Do not overwrite the physical queue except when first creating an agent, as this may orphan tasks. Use the Physical Queues Move Agent page.

Randomly Select Physical Queue

Click to assign another physical queue (within this logical queue) randomly. This selection will help to spread multiple agents evenly over available physical queues.

Skill Level

Select the skill level of this agent for the tasks that are assigned for this queue. This field is required.

The agent is assigned only tasks requiring a skill level that is less than or equal to the skill level specified here. If more than one qualified agent is available to accept the task, the queue server gives preference to the agent with the lowest skill level.

Each agent can have a different skill level for each queue to which the agent is assigned.

Maximum Workload

Select the maximum load that this agent can be assigned before tasks are held or assigned to other agents. This field is required.

The cost of each accepted task is added to the agent's current workload. A task is not assigned to an agent if its cost pushes the agent's current workload over the maximum.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: Do not delete a queue from an agent's list unless that agent has no open accepted tasks in that queue.

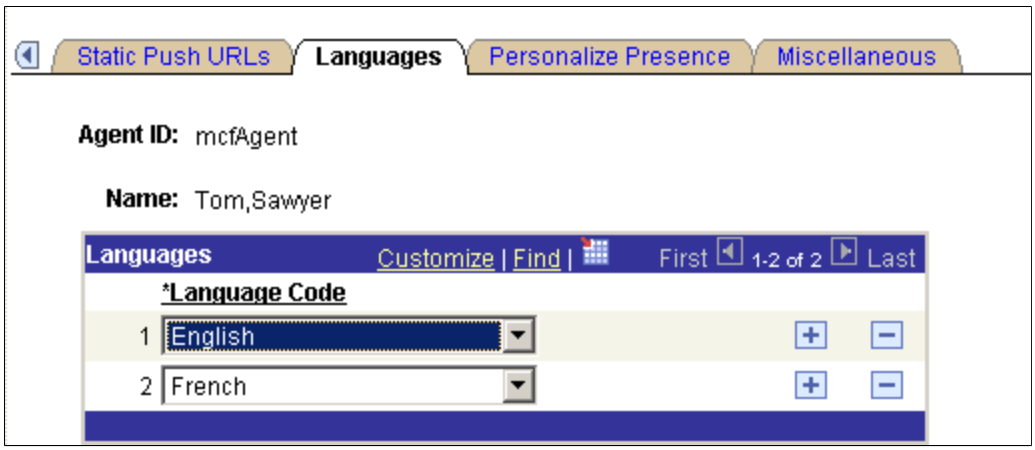
Specifying Languages That an Agent Supports

Access the Languages page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Agent, Languages

Image: Languages page

This example illustrates the fields and controls on the Languages page.



Specify the languages that each agent is qualified to support.

The agent is assigned only tasks that have been enqueued with a language code in the agent's language list. For the EnQueue() built-in function, the language code is specified as a parameter. For the InitChat() built-in function, the language code is determined by the user profile of the initiator.

If you do not specify a language code for a new agent, the default value is *English*.

Personalizing an Agent's Presence

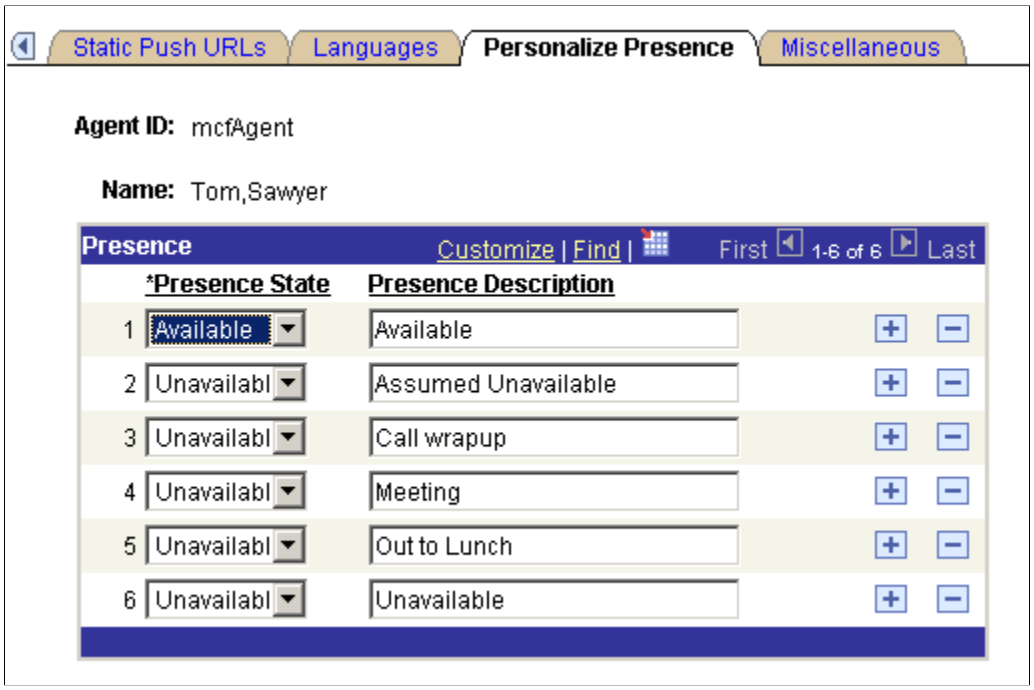
Access the Personalize Presence page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Agent, Personalize Presence

Image: Personalize Presence page

This example illustrates the fields and controls on the Personalize Presence page. You can find definitions for the fields and controls later on this page.



Each agent can configure the presence description that is displayed when the agent is available or unavailable. The queue server understands only the presence state, available or unavailable, but you can specify more specific presence descriptions when displaying or logging an agent's presence. For example, Lunch, Meeting, or Indisposed are unavailable states that can be used for tracking agent time and activity.

If you do not specify presence descriptions, default values are used.

Presence State Select *Available* or *Unavailable.*

Presence Description Enter a description for each agent state.

The description appears in logs of agent activity and when agent presence is displayed.

Note: The *Available* state has only one description. For the *Unavailable* state, you can enter any presence description, such as tea break, coffee break, lunch, and so on.

Defining Optional Agent Characteristics

To define optional agent characteristics, use the MCF Agent (MCF_AGENT_CMP) component.

This section discusses how to:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Set up buddy lists.
- Configure windows.
- Personalize chat.
- Specify agent-specific URLs.
- Specify miscellaneous parameters.

The agent configuration pages are considered optional because most do not have default values and can remain unconfigured without affecting an agent's ability to log on to a queue and accept tasks.

Setting Up Buddy Lists

Access the Buddy List page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Agent, Buddy List

Image: Buddy List page

This example illustrates the fields and controls on the Buddy List page. You can find definitions for the fields and controls later on this page.

The agent's buddy list facilitates collaborative chat and chat conferencing.

Agent Buddy

Select another agent with whom this agent can have a chat session or can ask to conference into another chat.

Each agent buddy must be logged in a physical queue on the same cluster to be able to chat. If two agents must be able to chat but they do not share a cluster, use the Physical Queue Move Agent page to move the agents into physical queues on the same MCF cluster.

Agent buddies are listed with their login status in the buddy list on the multichannel console.

Name

Displays the buddy agent's nickname.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

An agent's presence, as shown in the buddy list on the MultiChannel Console or on the Invite Agent list on the chat console, indicates the agent's availability for chat or conference.

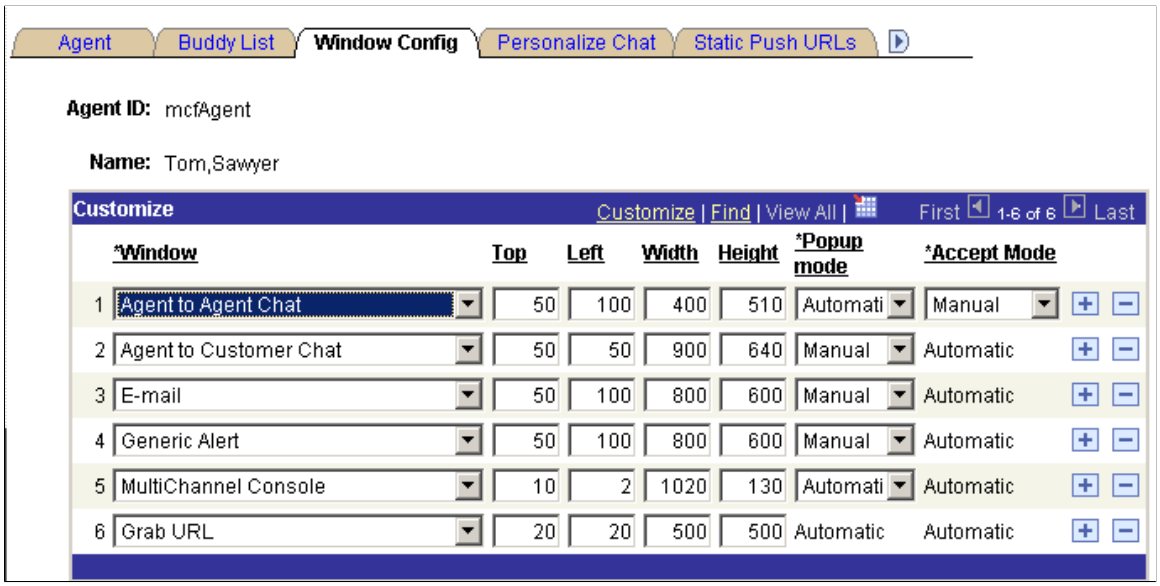
Configuring Windows

Access the Window Config page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Agent, Window Config

Image: Window Config page

This example illustrates the fields and controls on the Window Config page. You can find definitions for the fields and controls later on this page.



Set the initial agent window placement and size by specifying parameters on this page. An agent can resize and move the windows.

Window

Select the window to which the specified configuration applies.

Select from:

- *Agent to Agent Chat*
- *Agent to Customer Chat*
- *E-mail*
- *Generic Alert*
- *Grab URL*
- *MultiChannel Console*

Top and Left

Enter the distance in pixels from the top and left edges of the screen when the window first appears.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Width and Height

Enter the width and height, in pixels, of the window when it first appears.

Popup Mode

Select from:

Automatic: The window appears automatically. For customer-initiated chat or tasks that are initiated from the EnQueue() built-in function, the task is automatically accepted as well. For agent-initiated chat, the agent can elect not to accept the task; in effect, the agent can preview the task. If this is the desired behavior, select *Manual* from the Accept Mode drop-down list box. If you want agent-to-agent tasks to function like customer-initiated tasks, select *Automatic* from the Accept Mode drop-down list box.

Manual: The window does not appear until the agent clicks the task on the agent MultiChannel Console. For customer-initiated chat or tasks that are initiated from the EnQueue() built-in function, clicking the task means that the task is automatically accepted as well. For agent-initiated chat, the behavior depends on the setting for the Accept Mode field.

Accept Mode

Select from:

Automatic: Agent-to-agent chats are automatically accepted without requiring the agent to click the icon.

Manual: Agent-to-agent chats require the agent to click the icon.

Accept mode affects only collaborative chat.

Personalizing Chat

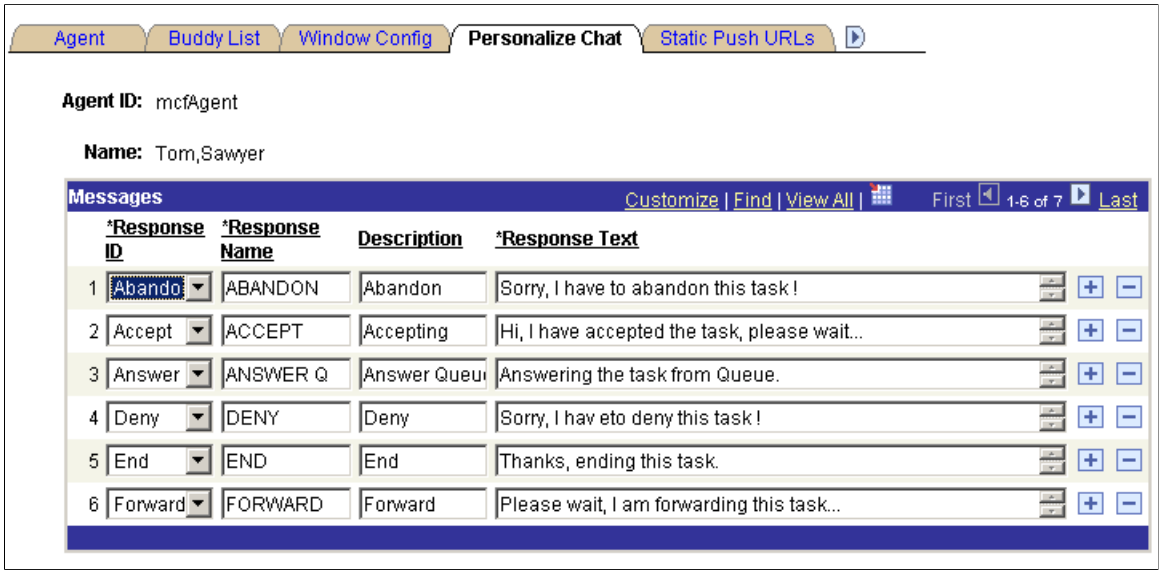
Access the Personalize Chat page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Agent, Agent Messages

Image: Personalize Chat page

This example illustrates the fields and controls on the Personalize Chat page. You can find definitions for the fields and controls later on this page.



An agent can create personalized responses in addition to the system responses that are defined for each queue.

Response ID

Responses, except those that are identified by *Other,* are linked to specific events. These responses are always sent on these events from this agent. If an agent does not have a customized response for a specific event, the response is read from a default value that is set in the Message Catalog. The response text that is set here overrides the default text that is set in the Message Catalog.

Select from:

- *Abandon:* A chat is abandoned when a chat initiator closes the chat window before the chat is accepted by an agent. This message appears when the agent accepts the abandoned chat.
- *Accept:* This response is automatically sent to the chat initiator when an agent accepts a chat in response to a chat request that does not include a question.

  If the chat request includes a question, the agent's *Answer Question* text is sent in response instead of the *Accept* response.
- *Answer Question:* This response is automatically sent to the chat initiator when an agent accepts a chat in response to a chat request that includes a question.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- *Deny:* This response applies only to collaborative chat. If an agent elects not to accept a chat, this message is automatically sent to the chat initiator.
- *End:* If either party quits a chat after the chat is accepted, this message is displayed from the agent.
- *Forward:* If the agent forwards a chat session to another queue, this message is sent to the customer.
- *Other:* These messages are never automatically sent in a chat session. Their message names appear in the Template Messages drop-down list box on the agent chat page. These messages are appended to the template messages (chat responses) that are defined for the queue.

Response Name This name appears in the agent's template response drop-down list box.

Response Text Enter the response text to appear in the chat window.

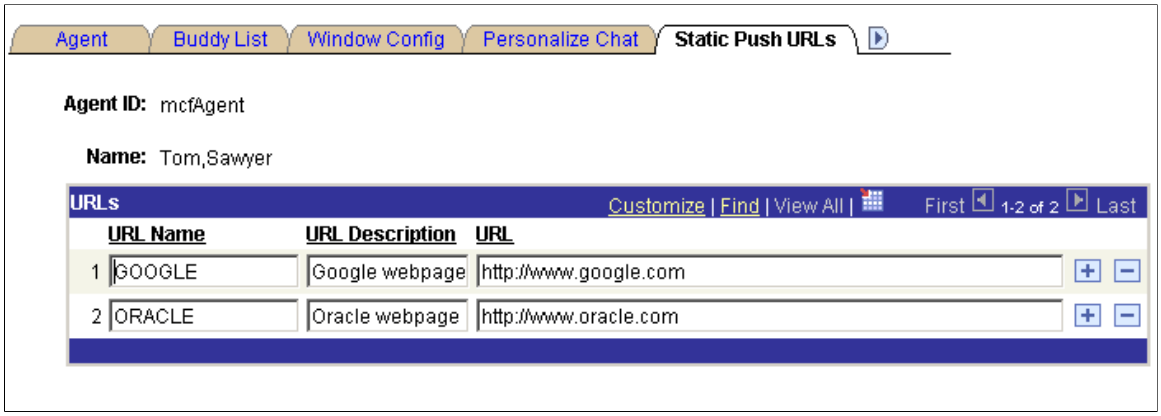
Specifying Agent-Specific URLs

Access the Static Push URLs page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Agent, Agent Push URLs

Image: Static Push URLs page

This example illustrates the fields and controls on the Static Push URLs page. You can find definitions for the fields and controls later on this page.



This page defines URLs that this agent can send to a client browser. These URLs are in addition to the URLs that are defined in the queue configuration page.

URL Name The URL name appears in the agent's static URL drop-down list box.

URL Description This description appears only on this page, to further describe this URL or, for example, its reason for inclusion.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

URL

Enter the queue push URL.

The URL must include the opening http:// and any required parameters.

All static URLs that are defined for the agent are downloaded when the agent launches the agent chat console by accepting a customer chat. Static URLs are not available in collaborative chat.

If you send a PeopleSoft Pure Internet Architecture URL, be sure that the recipient has permissions to access that portal , node, or page.

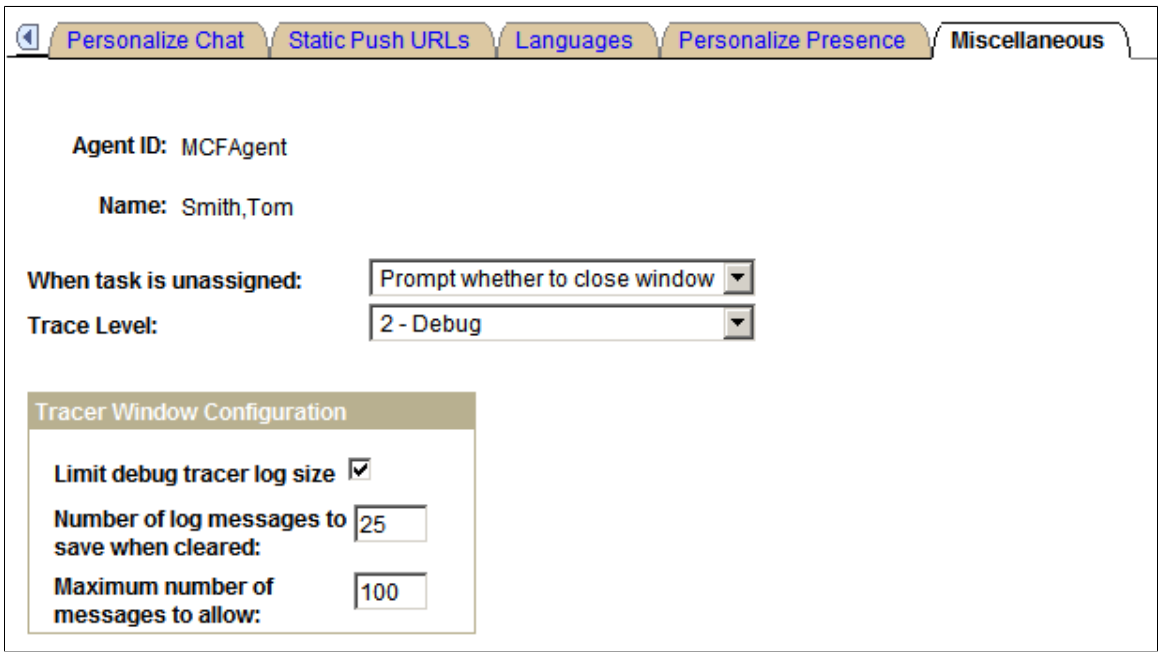
Specifying Miscellaneous Parameters

Access the Miscellaneous page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Agent, Miscellaneous

Image: Miscellaneous page

This example illustrates the fields and controls on the Miscellaneous page. You can find definitions for the fields and controls later on this page.



When task is unassigned

Select from the following values the action that occurs when a task that is assigned to an agent is unassigned:

- *Prompt whether to close window* (default).
- *Close the task window.*
- *Do not close the task window.*

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Trace Level

Select from the following log trace levels:

- *0 - None*
- *1 - Information*
- *2 - Debug*

If a value other than *0* is selected, a tracer window appears to display activities and events on the chat or MultiChannel Console for debugging purposes.

Limit debug tracer log size

This check box is enabled when the value entered for Trace Level is not 0. Select this check box to enable the agent to clear the tracer log based on Number of log messages to save when cleared and Maximum number of log messages to allow.

If the check box is deselected, the tracer log will not be cleared and the Number of log messages to save when cleared and Maximum number of log messages to allow will be disabled.

Number of log messages to save when cleared

Specify the minimum number of recent tracer log messages that should be maintained in the tracer window.

Maximum number of log messages to allow

Specify the maximum number of tracer log messages that will be maintained in the tracer window.

Note: Number of log messages to save when cleared and Maximum number of log messages to allow fields are required if the Limit debug tracer log size check box is selected.

Limit Debug Tracer Log Size Example

This table lists the values entered on the Miscellaneous page:

Field	***Value***
Trace Level	*2 - Debug*
Limit debug tracer log size	Selected
Number of log messages to save when cleared	*25*
Maximum number of log messages to allow	*100*

Based on these values, the first 75 messages will be cleared from the tracer window after 100 messages are logged. It will retain the most recent 25 messages for the agents reference. This process will repeat for every 100 messages that are logged in the tracer. The maximum number of messages is the tracer window at any one point in time is 100.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 9

Administering Queues, Logs, and Tasks

Administering Physical Queues

To administer physical queues, use the Physical Queues (MCF_ACCPT_TASK_CMP) component.

This section discusses how to:

- Move agents between physical queues.
- Move queues.
- Balance queues.

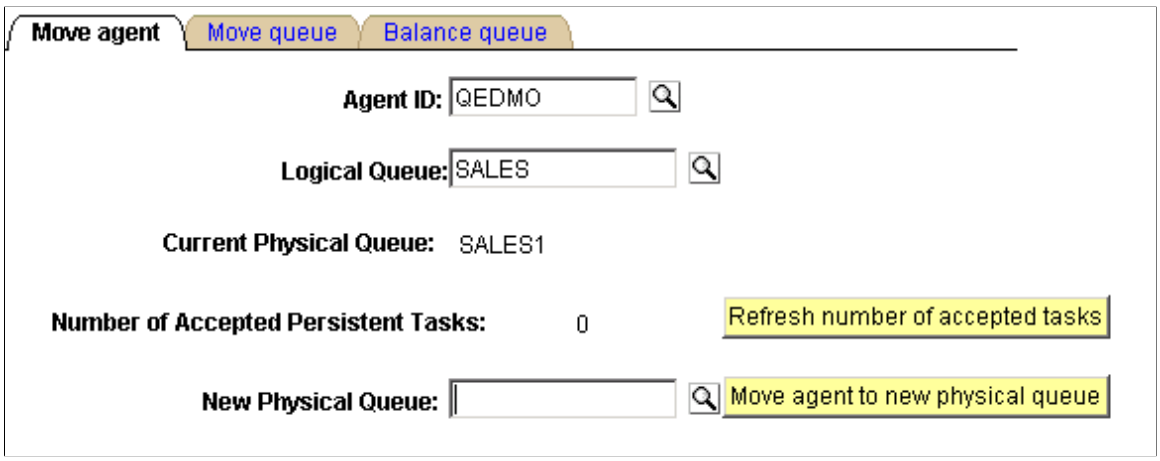
Moving Agents Between Physical Queues

Access the Move agent page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Physical Queues, Move agent

Image: Move agent page

This example illustrates the fields and controls on the Move agent page. You can find definitions for the fields and controls later on this page.



You can move an agent, and any open persistent tasks that are associated with that agent, from one physical queue in one cluster to another physical queue in another cluster on the same logical queue.

Logical Queue Select the logical queue within which the agent is to be moved.

Number of Accepted Persistent Tasks Displays the number of persistent tasks this agent has accepted on this physical queue.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

	This number is updated and displayed when you select the logical queue.
Refresh number of accepted tasks	Click to update the number of persistent tasks accepted by this agent on the current physical queue.
New Physical Queue	Select the new physical queue to which this agent and the persistent tasks accepted by this agent will be assigned.
Move agent to new physical queue	Click to perform the action.

Note: Ongoing chat sessions, which are not persistent tasks, are not affected by the move agent action.

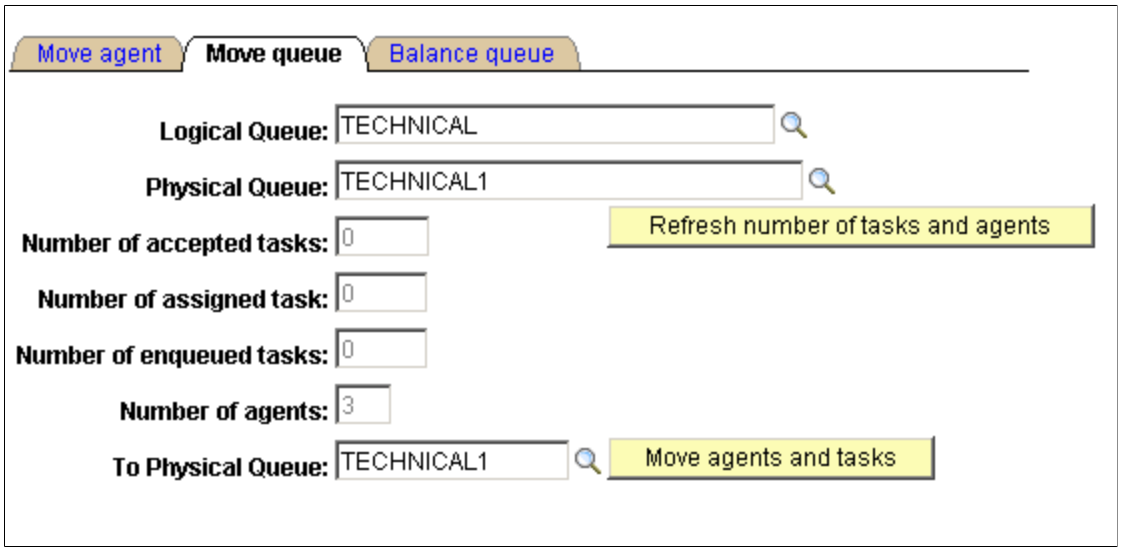
Moving Queues

Access the Move queue page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Physical Queues, Move queue

Image: Move queue page

This example illustrates the fields and controls on the Move queue page. You can find definitions for the fields and controls later on this page.



You can move all agents and their open persistent tasks from one physical queue to another physical queue within the same logical queue. For example, to delete a physical queue, move its agents and persistent tasks to another queue before deleting the first queue. Or, if the cluster serving this physical queue is overloaded, you can create another physical queue on another cluster, mark the first physical queue as inactive, create another physical queue on another cluster and move agents and persistent tasks from the inactive physical queue to the new physical queue.

Logical Queue	Select the logical queue within which the selected physical queue's agents and persistent tasks will be moved.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Physical Queue	Select the physical queue from which agents and persistent tasks will be moved.
Number of accepted tasks and Number of assigned tasks	Displays the number of accepted and assigned tasks open on this physical queue.
Refresh number of tasks and agents	Click to update the number of agents and persistent tasks assigned to this queue.
Number of enqueued tasks and Number of agents	Displays the number of enqueued tasks and assigned agents on this physical queue.
To Physical Queue	Select the physical queue to which the currently assigned agents and tasks will be moved.
Move agents and tasks	Click to move the assigned agents and tasks to the specified physical queue. **Note:** The physical queue must be inactive before moving agents and tasks. Inactivate the physical queue on the Queues page. Agents on the inactive physical queue are automatically logged off before being moved.

Related Links

Defining Queues

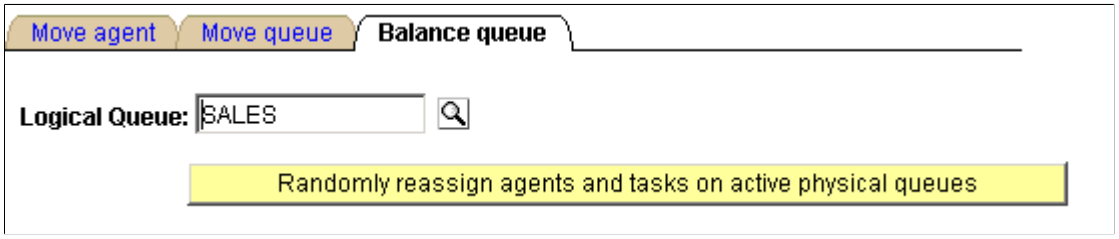
Balancing Queues

Access the Balance queue page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Physical Queues, Balance Queue

Image: Balance queue page

This example illustrates the fields and controls on the Balance queue page. You can find definitions for the fields and controls later on this page.



Over time, the distribution of agents and their associated skill levels, languages, and so on, across the physical queues belonging to a logical queue may change as agents are added or deleted. Rather than manually rebalancing the queue by moving individual agents, you can use the Balance queue page to randomly reassign agents and their open persistent tasks across all the active physical queues belonging to the selected logical queue.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Logical Queue Select the logical queue across which agents and persistent tasks will be balanced.

Randomly reassign agents and tasks on active physical queues Click to balance agents and tasks across the active physical queues for the selected logical queue.

This action redistributes agents and tasks assigned to this logical queue across all active physical queues without regard to previous assignments. Ensure that agents assigned to the affected physical queues have shut down their MultiChannel Consoles. They are *not* logged off automatically.

Viewing Queue Server, Queue, and Agent States

To view queue server state, queue state, and agent state, use the Queue Server State (MCF_QSERVSTATE_CMP), Queue State Summary (MCF_QUEUESTATE_CMP), and Agent State Summary (MCF_AGENTSTATE_CMP) components.

This section discusses how to:

- View the queue server state.
- View the queue state summary.
- View the agent state summary.

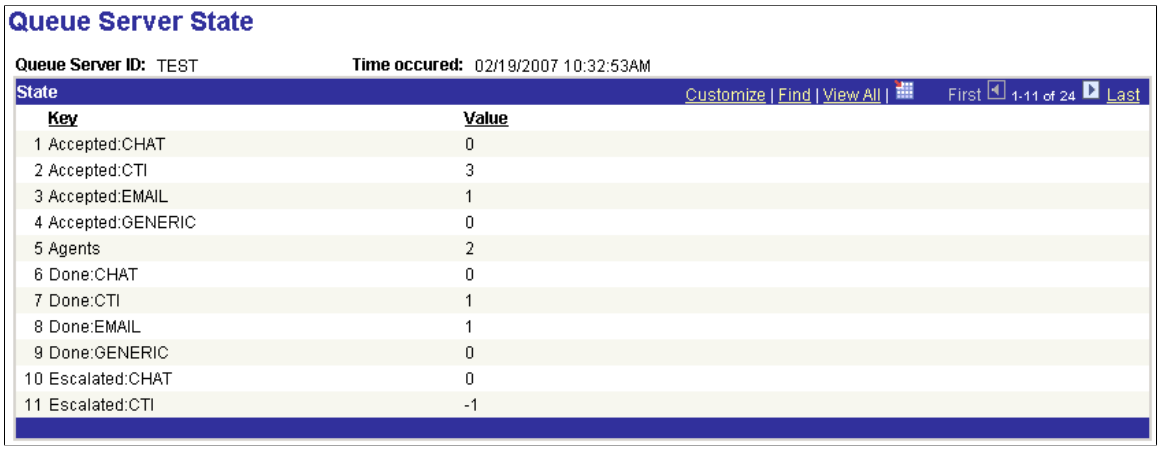
Viewing the Queue Server State

Access the Queue Server State page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Queue Server State

Image: Queue Server State page

This example illustrates the cumulative diagnostic totals for the selected queue server on the Queue Server State page.



	Key	Value
1	Accepted:CHAT	0
2	Accepted:CTI	3
3	Accepted:EMAIL	1
4	Accepted:GENERIC	0
5	Agents	2
6	Done:CHAT	0
7	Done:CTI	1
8	Done:EMAIL	1
9	Done:GENERIC	0
10	Escalated:CHAT	0
11	Escalated:CTI	-1

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The Queue Server State page displays cumulative diagnostic totals for the selected queue server, at the selected time, across all physical queues serviced by this queue server. Cumulative totals are reset after each state dump.

This state information comprises statistics to measure load and throughput that can be analyzed and used for tuning the system. For example, if counts are high, consider adding another physical queue to balance the load.

The state stamps are inserted for every primary queue server by cluster at configurable intervals, irrespective of user activity. You configure the intervals on the Cluster Tuning page.

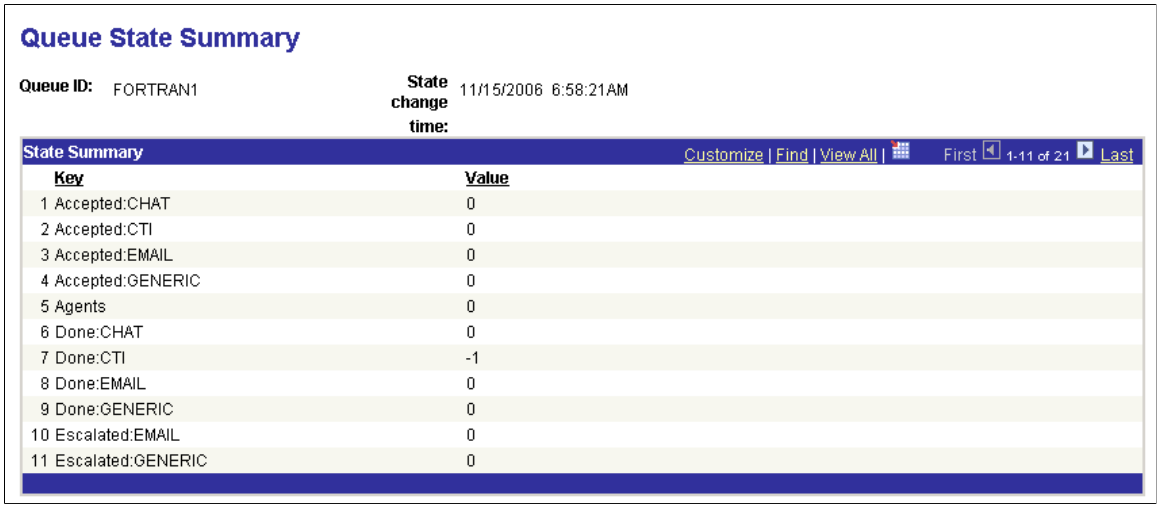
Viewing the Queue State Summary

Access the Queue State Summary page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Queue State Summary

Image: Queue State Summary page

This example illustrates the cumulative diagnostic totals for the selected physical queue on the Queue State Summary page.



Queue State Summary

Queue ID: FORTRAN1 State change time: 11/15/2006 6:58:21AM

State Summary Customize | Find | View All | First 1-11 of 21 Last

Key	Value
1 Accepted:CHAT	0
2 Accepted:CTI	0
3 Accepted:EMAIL	0
4 Accepted:GENERIC	0
5 Agents	0
6 Done:CHAT	0
7 Done:CTI	-1
8 Done:EMAIL	0
9 Done:GENERIC	0
10 Escalated:EMAIL	0
11 Escalated:GENERIC	0

The Queue State Summary page displays cumulative diagnostic totals for the selected physical queue.

This state information comprises statistics to measure load and throughput that can be analyzed and used for tuning the system. For example, if counts are high for this queue, consider adding another physical queue to balance the load.

The state stamps are inserted for every physical queue at configurable intervals, irrespective of user activity. The intervals are configured on the Cluster Tuning page.

Viewing the Agent State Summary

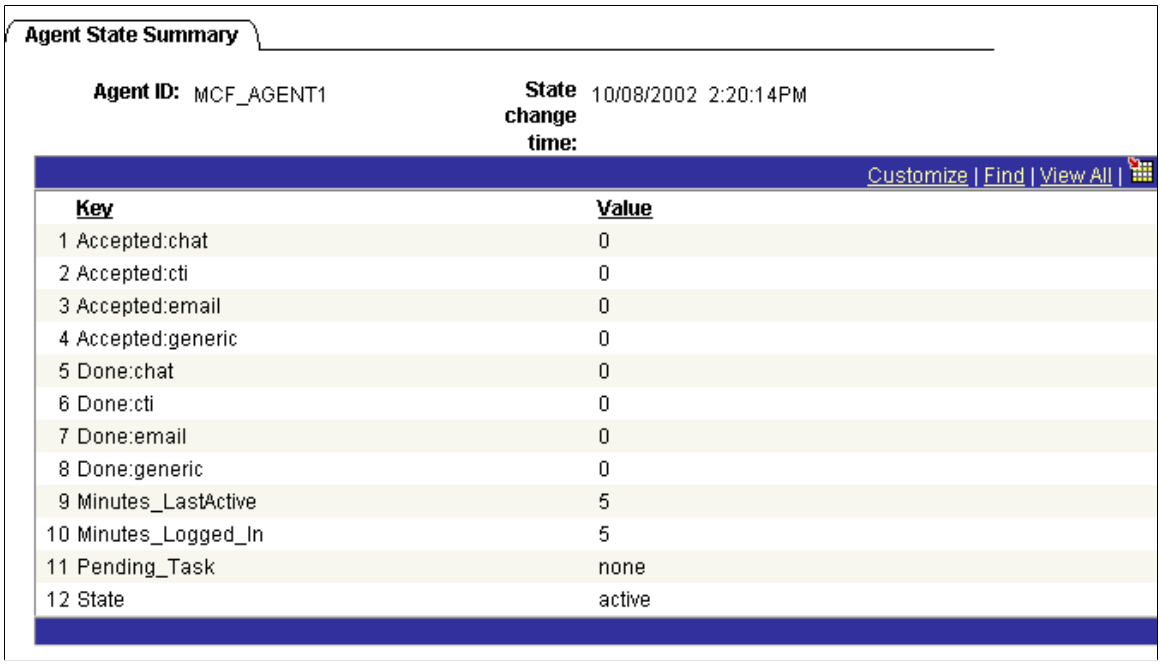
Access the Agent State Summary page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Agent State Summary

Image: Agent State Summary page

This example illustrates the cumulative totals for the selected agent on the Agent State Summary page.



Agent State Summary

Agent ID: MCF_AGENT1 State change time: 10/08/2002 2:20:14PM

Customize | Find | View All |

	Key	Value
1	Accepted:chat	0
2	Accepted:cti	0
3	Accepted:email	0
4	Accepted:generic	0
5	Done:chat	0
6	Done:cti	0
7	Done:email	0
8	Done:generic	0
9	Minutes_LastActive	5
10	Minutes_Logged_In	5
11	Pending_Task	none
12	State	active

The Agent State Summary page displays cumulative totals for the selected agent.

This state information comprises statistics to measure agent performance and status that can be analyzed and used for performance evaluation.

The state stamps are inserted for every agent at configurable intervals, irrespective of user activity. State is only recorded for agents currently logged on. The intervals are configured on the Cluster Tuning page by setting the dumpinterval. Enable agent logging by setting dumpagent to *yes*.

An agent state of *Active* indicates the agent is available; *Inactive* indicates the agent is not available.

Note: On a busy system, recording state information frequently may slow performance.

Related Links

Tuning Cluster Parameters

Viewing Broadcast, Chat, and Event Logs

To view broadcast, chat, and event logs, use the Broadcast Log (MCF_BCAST_LOG_CMP), Chat Log (MCF_CHAT_LOG_CMP), and Event Log (MCF_EVENTLOGL_CMP) components.

This section discusses how to:

- View broadcast logs.
- View chat logs.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- View event logs.
- View PeopleSoft MCF logs.

Viewing Broadcast Logs

The broadcast log page displays detailed information about any broadcast messages that were sent.

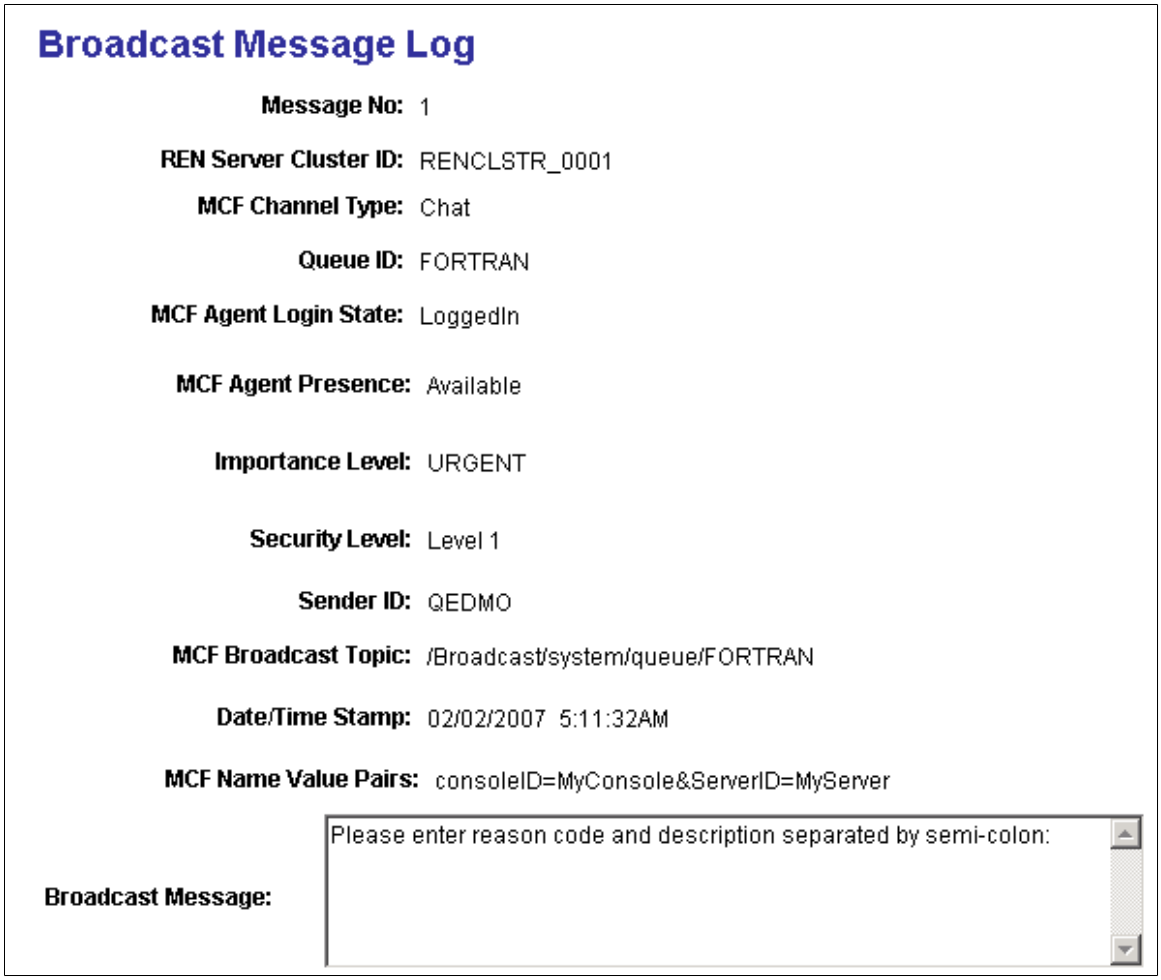
Access the Broadcast Message Log page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Broadcast Log

You can also access the Broadcast Message Log page by searching by message number, queue ID, or REN cluster ID.

Image: Broadcast Message Log page

This example illustrates the details of a broadcast message on the Broadcast Message Log page.



Note: To view the broadcast logs, set *log_broadcast* to *Yes* on the Cluster Tuning page. For this parameter to take effect, click Refresh logging parameters on the Cluster Notify page.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Related Links

Using PeopleSoft MCF Broadcast

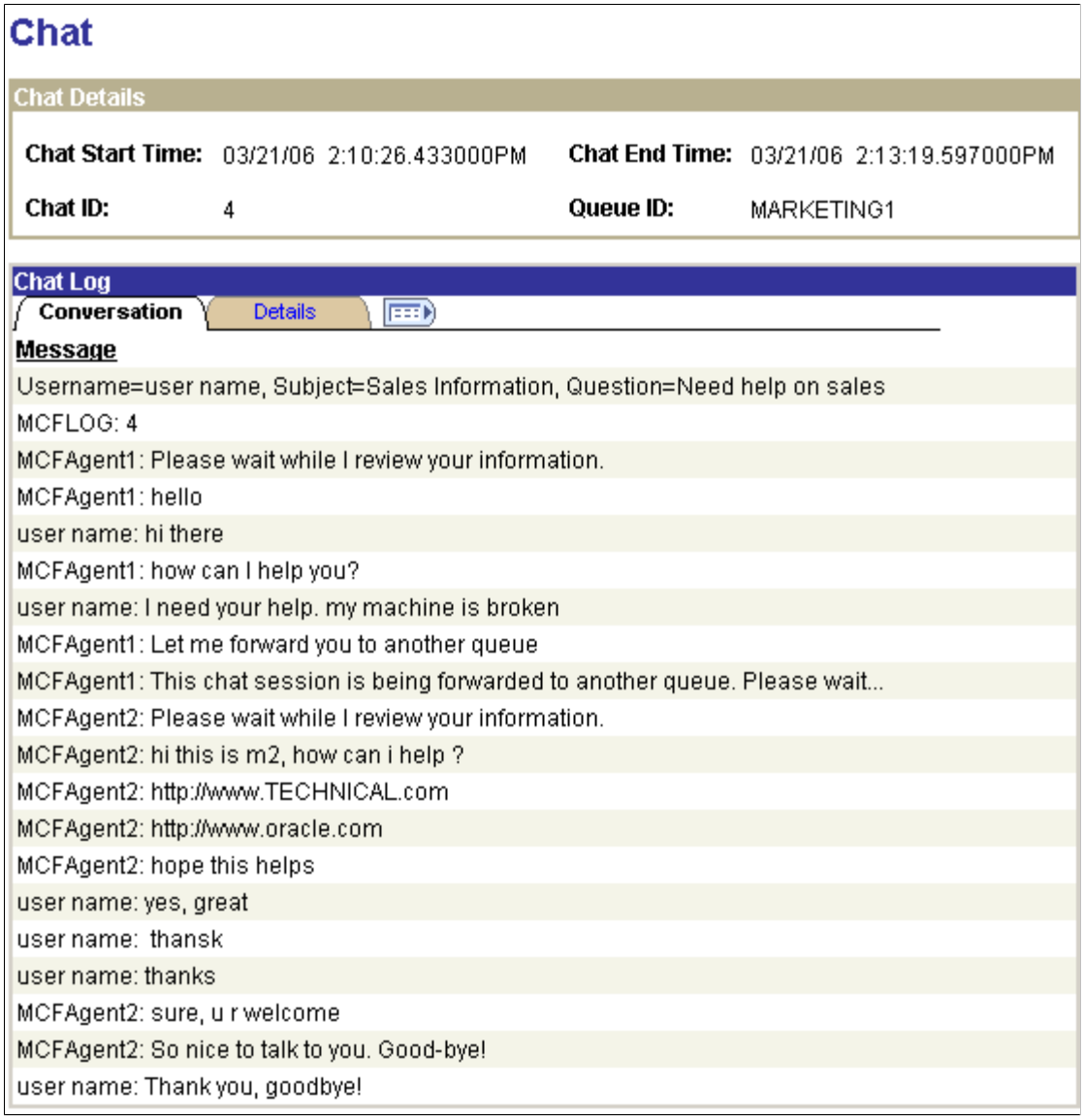
Viewing Chat Logs

Access the Chat page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Chat Log

Image: Conversation tab of the Chat page

This example illustrates a chat log on the Chat page.



If enabled, this log records the contents and events of every chat session.

To enable chat logging, set log_chat_ses to *yes* on the Cluster Tuning page. For this parameter to take effect, click Refresh logging parameters on the Cluster Notify page.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Details Tab

Access the Details tab of the Chat Page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Chat Log, Details

Image: Details tab of the Chat page

This example illustrates details of a chat log on the Chat page.

Chat

Chat Details

Chat Start Time:	03/21/06 2:10:26.433000PM	**Chat End Time:**	03/21/06 2:13:19.597000PM
Chat ID:	4	**Queue ID:**	MARKETING1

Chat Log

Conversation | **Details**

User ID	Date/Time Stamp	MCF Chat action	Queue ID	Task identifier
m1	03/21/06 2:10:26PM	InitChat	MARKETING1	UNDEFINED
MCFLOG	03/21/06 2:10:26PM	LogID		UNDEFINED
m1	03/21/06 2:10:34PM	Accept	UNDEFINED	QSERVER_0001_CHAT_0
m1	03/21/06 2:10:41PM	Msg		UNDEFINED
m1	03/21/06 2:10:45PM	Msg		UNDEFINED
m1	03/21/06 2:10:57PM	Msg		UNDEFINED
m1	03/21/06 2:11:25PM	Msg		UNDEFINED
m1	03/21/06 2:11:41PM	Msg		UNDEFINED
m1	03/21/06 2:11:47PM	Forward	TECHNICAL1	UNDEFINED
m2	03/21/06 2:12:17PM	Accept	UNDEFINED	QSERVER_0001_CHAT_0
m2	03/21/06 2:12:33PM	Msg		UNDEFINED
m2	03/21/06 2:12:39PM	Unknown		UNDEFINED
m2	03/21/06 2:12:54PM	Unknown		UNDEFINED
m2	03/21/06 2:13:01PM	Msg		UNDEFINED
m1	03/21/06 2:13:06PM	Msg		UNDEFINED
m1	03/21/06 2:13:07PM	Msg		UNDEFINED
m1	03/21/06 2:13:12PM	Msg		UNDEFINED
m2	03/21/06 2:13:17PM	Msg		UNDEFINED
m2	03/21/06 2:13:19PM	End		UNDEFINED
m1	03/21/06 2:13:19PM	End		UNDEFINED

The Details tab displays detailed information about the selected chat conversation.

Viewing Event Logs

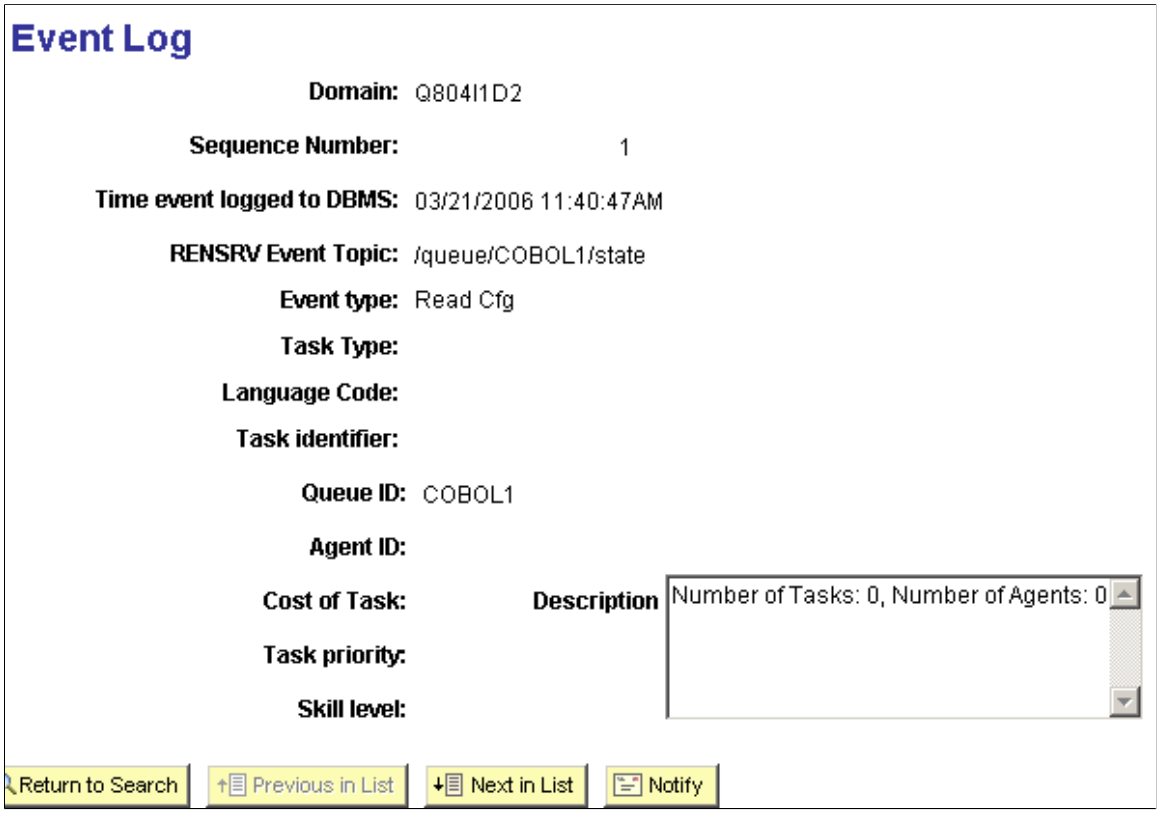
Access the Event Log page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Event Log

Image: Event Log page

This example illustrates the fields and controls on the Event Log page. You can find definitions for the fields and controls later on this page.



Domain	The application server domain on which this event occurred.
Time event logged to DBMS	The time that this event was recorded in the database.
Event Type	The type of event, as described in the table that follows.
Task Type	The type of task for this event: chat, email, or generic. CTI events are logged in the CTI event log.

The event log records PeopleSoft MultiChannel Framework events sent to the real-time event notification (REN) server, excluding chat content (which is logged in the chat log). The event log can be used for debugging as well as for system monitoring. For example, you can determine when agents log in and log out, or when the queue server was first notified of newly enqueued events.

Data displayed in the event log depends on the event type. Not all fields apply to every event.

To enable logging of state broadcast events, set *logDMPQ* to *yes* on the Cluster Tuning page. For this parameter to take effect, click Refresh logging parameters on the Cluster Notify page.

Time is displayed and searched on in the format MM/DD/YYYY HH:MM:SSA/PM.

The following table lists possible event types:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Name	***Translate Table Value***	***REN MultiChannel Framework Topic***	***Description***
Accepted	ACPD	/agent/<agentID>/accepted Agent 's list of accepted tasks	Agent's list of accepted tasks
Accept	ACPT	/queue/agents/accept	Agent accepts an assigned task.
Bcst Admin	BCST	/queue/admin/statedump	Broadcast universal queue information
Contact	CNCT	/queue/contact	Real-time contact (for example, chat)
DB Cntct	DBCT	/queue/dbcontact	Database management system contact (such as email or generic)
Dump Q	DMPQ	/queue/<queueID>/state	Dump queue state information to log
Done	DONE	/queue/agents/dequeue	Done (dequeue)
Forward	FWD	/queue/agents/forward	Forward
Notify	NTFY	/agent/<agentID>/notify	Notify agent of assigned task.
Presence	PRES	/queue/agents/presence	Agent's presence change
Restrt Ack	RACK	/queue/agents/restartack	Restart acknowledgement
Read Cfg	READ	/uqsrv/reread/defaults	Reread defaults
Restart	RSRT	/queue/<queueID>/restart	Restart
Unknown	UNKN	UNKNOWN	Unknown REN server event
Unassign	USGN	/agent/<agentID>/unassign	Unassign

The following table lists event logs:

Name	***Translate Table Value***	***REN MultiChannel Framework Topic***	***Required Argument Value***	***Meaning***
Abandon	ABAN	/chat/<userID>/ <chatID>	ps_type=abandon	Abandoned chat session
End	END	/chat/<userID>/ <chatID>	ps_type=end	End chat session
Login	LGIN	/queue/agents/loginstate	ps_state=login	Log on
Logout	LGOT	/queue/agents/loginstate	ps_state=logout	Log out

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Name	*Translate Table Value*	*REN MultiChannel Framework Topic*	*Required Argument Value*	*Meaning*
Message	MSG	/chat/<userID>/<chatID>	ps_type=msg	Message
Push URL	PUSH	/chat/<userID>/<chatID>	ps_type=pushurl	Push URL
Timeout	TOUT	/chat/<userID>/<chatID>	ps_type=timeout	Timeout chat session

Viewing PeopleSoft MCF Logs

Diagnostic PSUQSRV and PSMCFLOG traces are written to the log directory of each application server domain. The trace level is determined by the LogFence setting in the domain configuration. You can set the LogFence parameter with PSADMIN configuration of the application server domain. The following table lists LogFence setting values:

LogFence Setting	*Tracing Level*
1	Fatal errors
2	Errors
3	Warnings
4	Level 1 Diagnostic: Logs the queues that the universal queue is servicing, logs new queues that are added, and logs agent logon and logout.
5	Level 2 Diagnostic: Logs most debugging information, except periodic events (such as timer check and heartbeat).
6	Level 3 Diagnostic: Logs everything, including periodic events.

Administering Overflow and Escalated Tasks

To administer overflow and escalated tasks, use the Overflow Administration (MCF_OVERFLOWL_CMP) and Escalation Administration (MCF_ESCAL_CMP) components.

This section discusses how to:

- Administer overflow tasks.
- Administer escalated tasks.

Administering Overflow Tasks

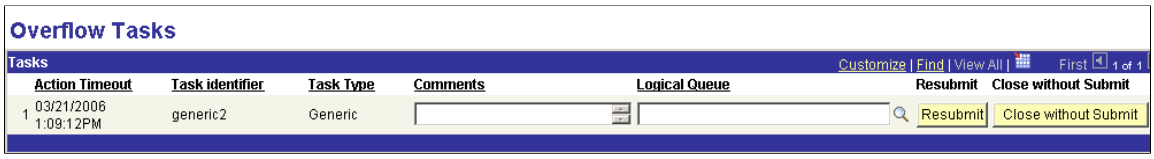
Access the Overflow Tasks page using the following navigation path:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Overflow Admin (overflow administration)

Image: Overflow Tasks page

This example illustrates the fields and controls on the Overflow Tasks page. You can find definitions for the fields and controls later on this page.



Use this page to manage tasks that could not be assigned to an agent within the specified overflow timeout.

Action Timeout	Displays the time at which the overflow occurred.
Task Identifier	Displays the task identifier.
Task Type	Types include chat, email, generic, and voice (not supported).
Comments	Enter optional text commentary about the resolution of the task. For example, note that the customer sent follow-up email that was answered by an agent.
Logical Queue	Select a logical queue to which to resubmit this task.
Resubmit	Click to send the task to the specified logical queue to retry assignment. Only persistent tasks can be resubmitted; chat cannot be resubmitted.
Close without Submit	Click to close this task without sending it to retry assignment.
Detail	Click to display additional information about the task.

Administering Escalated Tasks

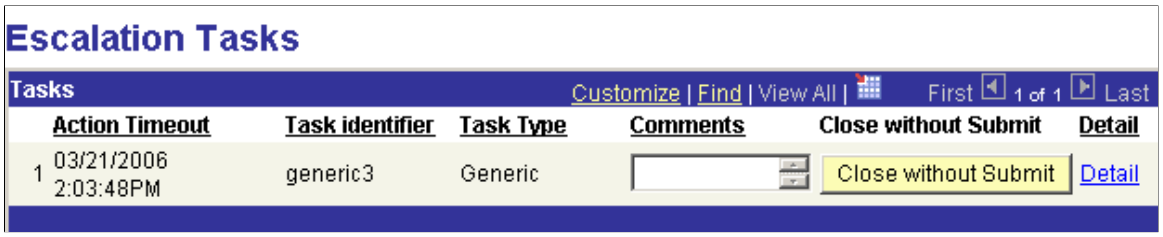
Access the Escalation Tasks page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Universal Queue, Administration, Escalation Admin (escalation administration)

Image: Escalation Tasks page

This example illustrates the fields and controls on the Escalation Tasks page. You can find definitions for the fields and controls later on this page.



Use this page to manage accepted persistent tasks that were automatically unassigned from agents because they were not closed within the specified escalation timeout. Tasks can be closed without submitting them.

Action Timeout	Displays when the escalation occurred.
Task Identifier	Displays the task identifier.
Task Type	Types include chat, email, generic, and voice (not currently supported).
Comments	Enter optional text commentary about the task's resolution. For example, note that the customer sent follow-up email that was answered by an agent.
Close without Submit	Click to close this task without sending it back to retry assignment.
Detail	Click to view additional information about this escalated task.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 10

Managing Tasks and Using Chat in PeopleSoft MultiChannel Framework

Managing Tasks with the MultiChannel Console

Use the MultiChannel Console to accept, respond to, transfer, and complete tasks and to initiate or join chat sessions.

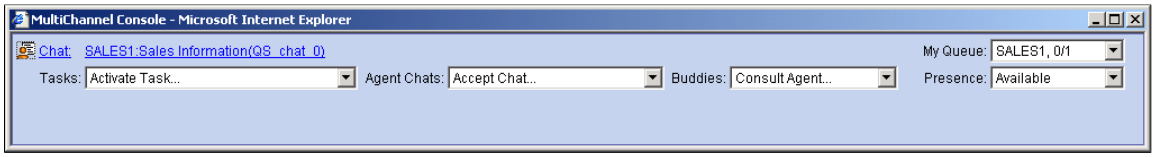
This section discusses how to use the MultiChannel Console to work with tasks.

Using the MultiChannel Console to Work with Tasks

If your user ID includes security permissions for PeopleTools MCF Console, the MultiChannel Console navigation tool appears in the universal navigation header and if your user ID includes security permissions for PeopleSoft MultiChannel Framework, you can access the Multichannel Console. To access the console, click the tool.

Image: MultiChannel Console navigation tool

This example illustrates the fields and controls on the MultiChannel Console. You can find definitions for the fields and controls later on this page.



You cannot launch two MultiChannel Consoles on the same workstation.

You can launch consoles on different workstations using the same user ID, but you cannot log on to physical queues served by the same MCF cluster.

When the MultiChannel Console is launched, it attempts to connect to the real-time event notification (REN) server associated with the last physical queue listed in the My Queue drop-down list box. If that REN server is not running, the agent receives an error message. If one or more of the agent's physical queues are associated with a running REN server, the agent can log on by selecting another queue.

Newly assigned tasks appear as a link above the Tasks drop-down list box, and display a flashing icon. Click the link to accept the task. Accepted tasks appear in the Tasks drop-down list box.

Collaborative chat requests (agent-to-agent chats) may appear as a link above the Agent Chats drop-down list box, but all collaborative chat requests (accepted or not) appear in the Agent Chats drop-down list box.

Configure the size and initial location of the MultiChannel Console on the agents Window Config (window configuration) page.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: If you use web content zones in Microsoft Internet Explorer security options, your PeopleSoft Pure Internet Architecture web server address and your REN server address must be in the same security zone. If you use Secure Sockets Layer (SSL) security, you may receive a security warning message when your console first opens. You can accept the warning message without compromising SSL security.

Tasks

After you accept an assigned task, the task appears here.

Activate a task to work on by selecting it. Activating a task brings the window associated with that task to the foreground. If the associated task window is not running, it is launched by the console.

A task is removed from the task list when you mark the task as done using one of the following methods:

- For email and generic tasks, the application page of the task may include a Done button; refer to the application's documentation.
- For chat, the task is complete when the chat dialog ends.

Agent Chats

Displays a list of your collaborative chats, both accepted and requested, including a label indicating if the chat is inbound or outbound. Select a chat to activate it.

If you are invited to join in a chat session (conference), the conference chat is added to the list. After you accept the conference, the conference is added to the Tasks drop-down list box and removed from the Agent Chats drop-down list box. The chat is removed from the list when the chat dialog ends.

Buddies

Includes agents that are identified as buddies in your agent configuration. Add buddies on the Buddy List page of the Agent component.

To initiate a collaborative chat, select an agent. A chat dialog box displays.

Note: You can only initiate a collaborative chat with an agent who appears on your buddy list. However, if you appear on another agent's buddy list, that agent can initiate a collaborative chat with you, even if that agent is not one of your buddies.

My Queue

Select a queue to log on to that queue.

The drop-down list box includes the physical queues that you can access.

To change queues, select a new queue. You are logged off from the old queue and logged on to the selected queue.

Add or delete queues on the Queues page of the Agents component.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Presence

Displays your current presence and enables selection of a new presence. Select from:

- *Active*
- *Inactive*

Customize presence labels to more closely track agent activity. Edit your presence options on the Presence page of the Agent component.

If you select an inactive presence, an icon appears in the upper-left corner of the MultiChannel Console. Changes in agent presence are logged and include the customized presence description.

Presence status persists for eight hours from the time of its last change.

You can use hot key combinations, described in the following table, to navigate between MultiChannel Console fields:

Hot Key Combination	***Opens***
ALT + A	Agent chats
ALT + B	Buddies
ALT + P	Presence
ALT + Q	My Queues
ALT + T	Tasks

Communicating with Customers and Agents Using Chat

This section discusses how to:

- Use the agent chat window.
- Use the customer chat window.

Using the Agent Chat Window

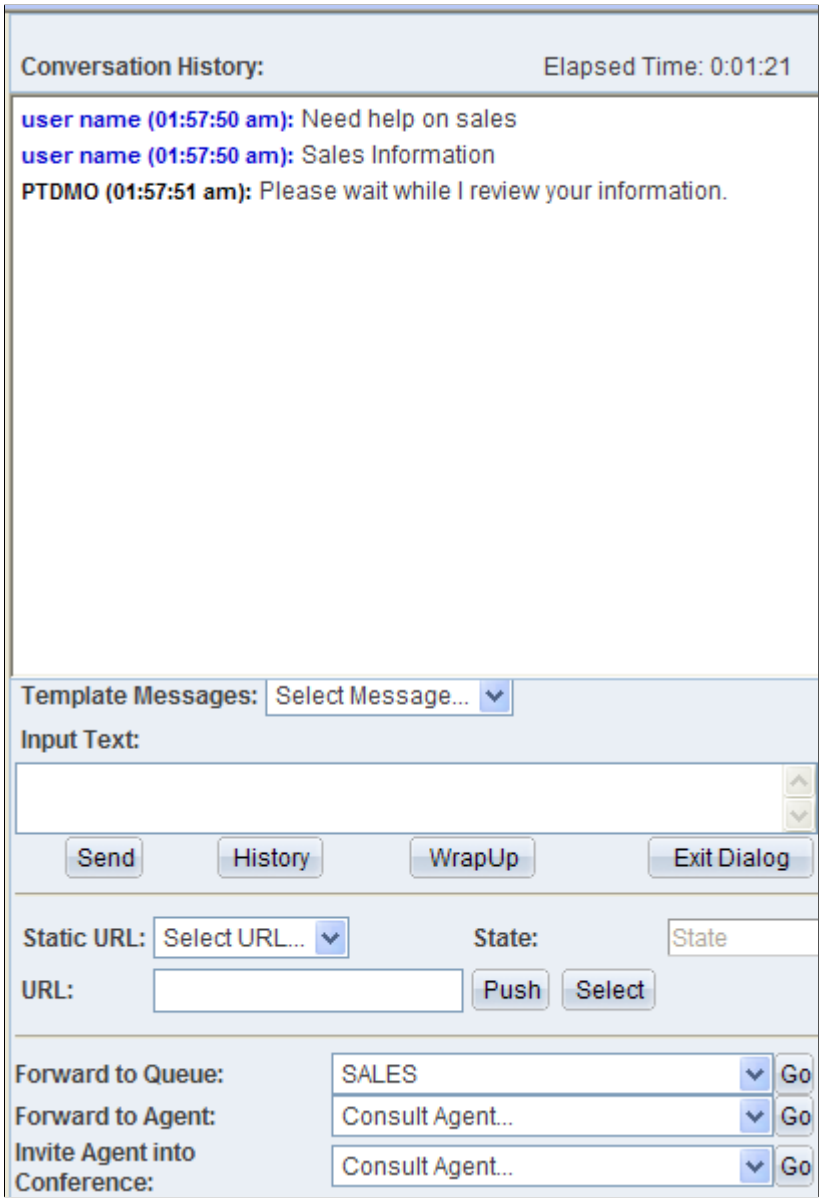
When you accept a chat task, a chat window opens.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The format of the agent chat window and the contents of its right pane are delivered as part of PeopleSoft MultiChannel Framework. The content of the left pane is determined by application developers, and is passed as a parameter to the InitChat() built-in function.

Image: Agent Chat window

This example illustrates the fields and controls on the Agent Chat window. You can find definitions for the fields and controls later on this page.



Configure the size and initial location of the agent chat window on the agent's Window Config (window configuration) page.

Conversation History

Lists progress of the chat, line by line.

If chat logging is enabled, the conversation is recorded in the database chat log by the MCF log server. View the chat log on the Chat Log page.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

If accessibility features are not turned off in the My Personalizations component, an additional text box appears below the conversation history. The most recent customer input appears in the secondary text box, which can be read by screen reading software.

Template Messages

Send the customer a standard message by selecting one from the list.

The message text appears in the Input Text text box. Click Send to send the message.

Edit template messages for each queue on the queue Chat Responses page.

Input Text

To respond to the customer, enter text, and then click Send or press Enter.

The maximum size of the text that can be sent at one time is 4096 bytes (4 kilobytes).

Send

Click to send the contents of the Input Text text box.

Exit Dialog

Click to end the chat.

The chat window remains open, enabling you to return to the page for follow-up work after the customer exits the dialog.

Static URL and Push

To send a static URL to the customer, select a URL name, then click Push.

When a URL is pushed to a customer, a browser window for that URL is launched on the customer's workstation.

URL

Displays static and grabbed URLs.

Select

Click to launch an application page, from which a URL can be returned to populate the URL field.

The format of the URL wizard page that appears when you click Grab is defined by application developers. PeopleSoft supplies a sample URL wizard page.

See Using the URL Wizard.

Forward to Queue

To forward the current chat to another queue, select the queue.

You can only forward to another physical queue, and that physical queue must be served by the same MCF cluster as the physical queue that assigned the chat.

Invite Buddy

To request that another agent join the chat, select the agent and click Go.

The buddy must be logged on to a physical queue that is served by the same MCF cluster as the physical queue that sent you this

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

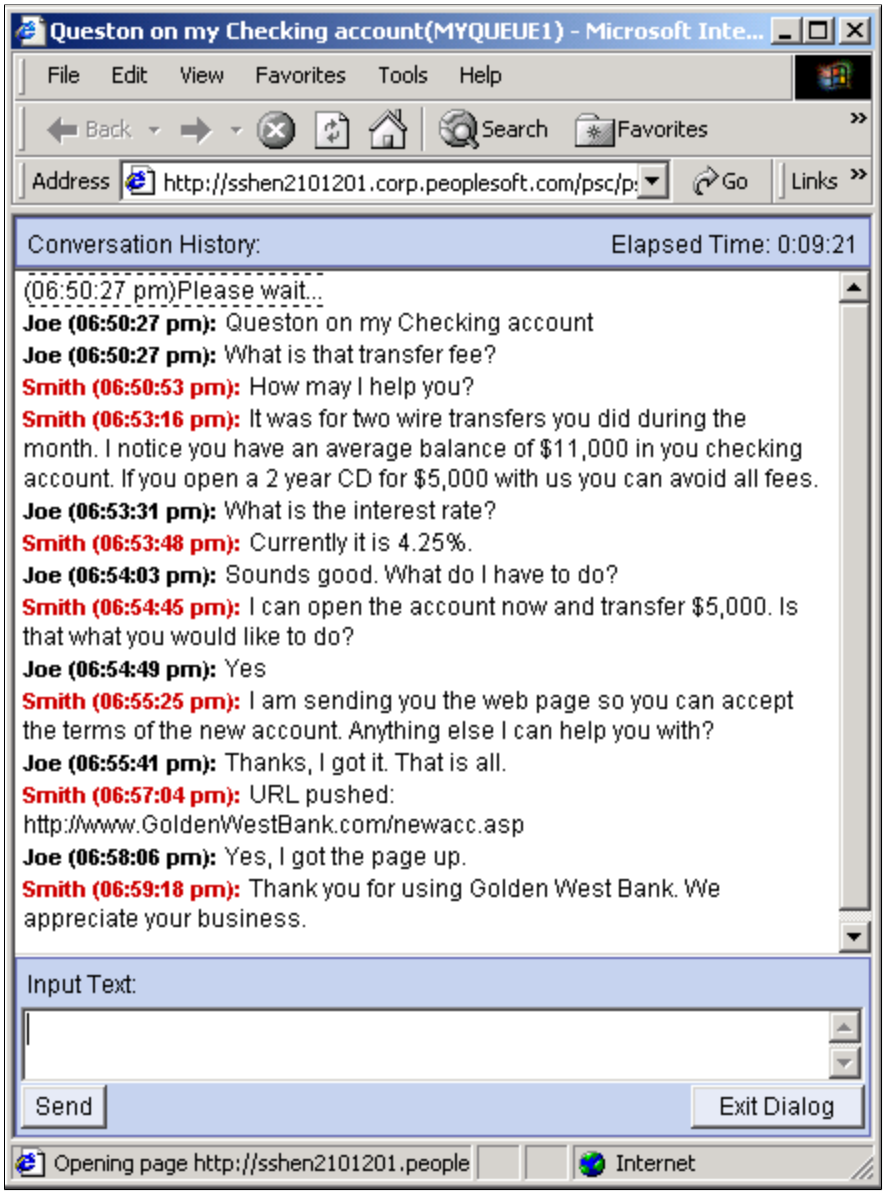
chat. Chat conferencing is only supported on customer-initiated chat sessions.

Using the Customer Chat Window

When a customer initiates a chat, a chat window appears:

Image: Customer chat window

This example illustrates the fields and controls on customer chat window. You can find definitions for the fields and controls later on this page.



The format and content of the customer chat window is delivered as part of PeopleSoft MultiChannel Framework. The customer chat window includes a conversation history text box, input text box, and Send and Exit Dialog buttons.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

If accessibility features are not turned off in the My Personalizations component, an additional text box appears below the conversation history. The most recent agent input appears in the secondary text box, which can be read by screen reading software.

Input Text	The customer enters text here. The maximum size of the text that can be sent at one time is 4096 bytes (4 kilobytes).
Send	Click to send the input text to the agent.
Exit Dialog	Click to end the chat and close the chat window.

Note: The agent collaborative chat window is substantially the same as the customer chat window.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 11

Using PeopleSoft MCF Broadcast and Working with Sample Pages

Using PeopleSoft MCF Broadcast

Use the broadcast function to broadcast a notification message. This function is typically used by a supervisor to send a notification message to specific recipients based on the parameters that are provided by the sender. Broadcast notifications can be sent system-wide, cluster-wide, queue-based, task-based, or activity-based.

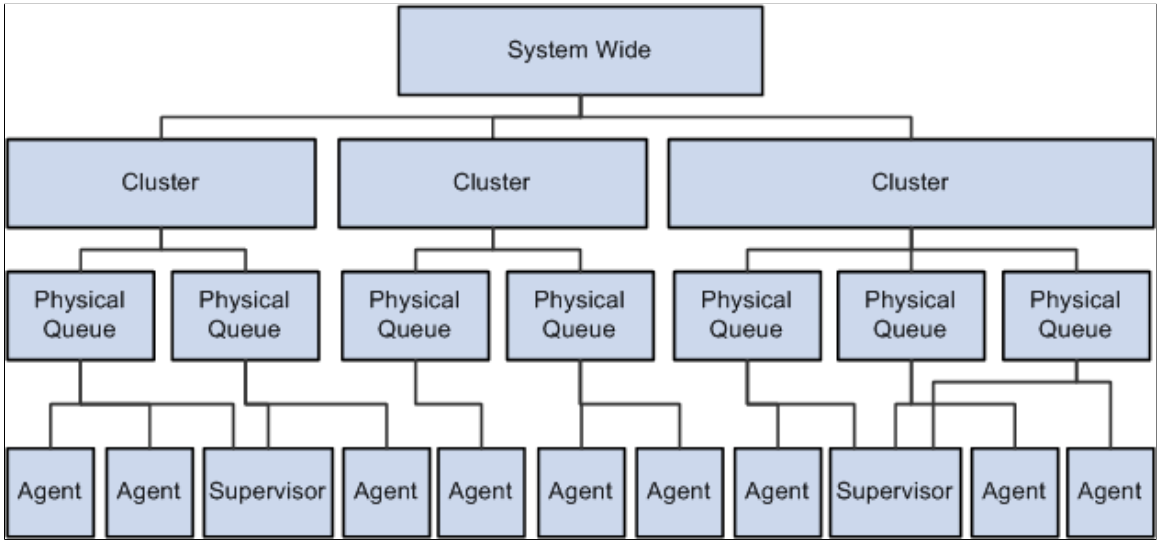
System-wide broadcast notifications can be sent only using the PeopleCode API to all logical queues on the system. The same notification using JSMCAPI is sent to all the physical queues on the cluster.

Cluster-wide broadcast enables a user to broadcast to an entire audience that is logged on or subscribed to this broadcast topic on a particular cluster. A cluster broadcast can be configured to target a specific audience by specifying a particular queue, a channel, or an agent log state in the broadcast call.

A queue-based broadcast can send a notification to all agents on a particular queue. A channel-type broadcast is received by agents serving a particular channel. An agent working on multiple channels can receive broadcast that is meant for all those channels. An agent login state or activity-based broadcast allows a broadcast notification to be sent to all agents in a particular state.

Image: Cluster-wide broadcast

The following diagram illustrates a cluster-wide broadcast of notifications.



This section provides an overview of JSMCAPI broadcast and discusses how to:

- Implement MCF broadcast.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Configure JSMCAPI Broadcast using MCF Supervisor console.
- Configure PeopleCode broadcast.
- View broadcast logs.

Understanding JSMCAPI Broadcast

A JSMCAPI console primarily operates on physical queues. Whenever an agent or a supervisor uses a JSMCAPI console, the agent chooses a physical queue and logs onto it. To broadcast a message, a console typically uses this physical queue as a parameter to determine the target cluster. JSMCAPI broadcast works on the following restrictions:

- JSMCAPI broadcast function does not provide any routing logic because it operates within a limited set of physical queues that are assigned to the agent or supervisor.

  The agent can choose any physical queue from the assigned queue list without logging onto any particular physical queue.
- The supervisors or the users of JSMCAPI broadcast can use only the physical queues on their list to send a broadcast notification.
- JSMCAPI works within the confine of a single cluster and does not have any knowledge or access to other clusters in the system.

Implementing MCF Broadcast

To enable you to implement broadcast, MultiChannel Framework provides a PeopleCode built-in function and a JSMCAPI interface for JSMCAPI users (agents and supervisors) to send broadcast notifications.

Note: The broadcast notifications are displayed only in the third-party sample pages.

Subscribing and Publishing Broadcast

The JSMCAPI broadcast function is a specialized publish call to the REN server. Both JSMCAPI and PeopleCode publish to physical-queue broadcast topic (such as /Broadcast/SALES1), and the subscribers always subscribe to broadcast topic for the physical queues to which they were assigned.

To subscribe, use the following function:

```
BroadcastSubscribe(cluster,queue,task,state,presence,method)
```

To publish, use the following function:

```
MCFBroadcast(cluster,queue,task,state,presence,message,securitylevel,
importancelevel, senderid, NameValuePairString
```

To unsubscribe, use the following function:

```
void broadcastUnsubscribe(type)
```

where <type> determines the type of broadcast, system-wide, queue-wide, or agent-wide.

Related Links

broadcastSubscribe

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

broadcastUnsubscribe

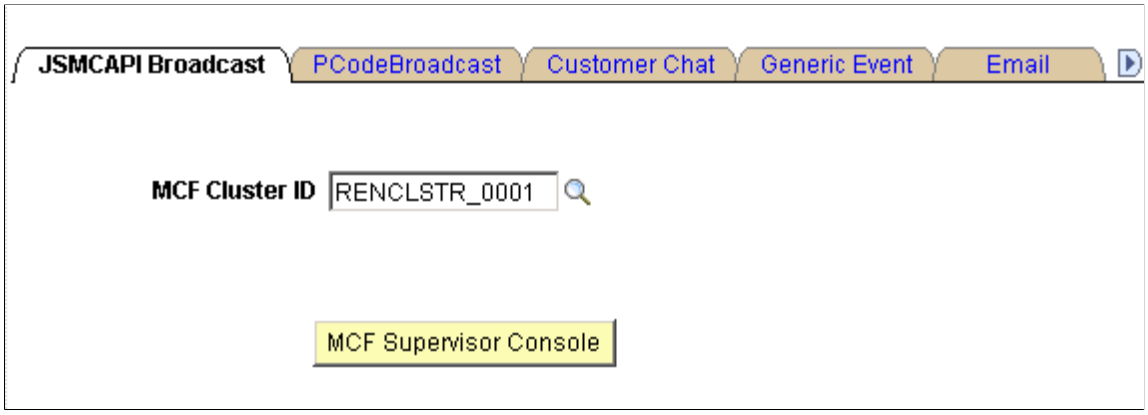
Using JSMCAPI Broadcast with MCF Supervisor Console

Access the JSMCAPI Broadcast page using the following navigation path:

PeopleTools , MultiChannel Framework, Universal Queue, Sample Pages, JSMCAPI Broadcast

Image: JSMCAPI Broadcast page

This example illustrates the fields and controls on the JSMCAPI Broadcast page. You can find definitions for the fields and controls later on this page.



REN Server Cluster ID Select the REN server cluster on which to test the MCF Supervisor console.

Note: The MCF Supervisor console is specific to the REN cluster Id selected.

MCF Supervisor Console Click to initiate the supervisor console.

Note: To demonstrate the MCF Supervisor console, you must have MCF servers, both queue and log servers, running and communicating with the specified REN server cluster.

After you click MCF Supervisor console, a new browser window appears that displays the sample supervisor console.

Note: To enable the new browser window to appear, disable any pop-up blocking software for your browser.

MCF Supervisor Console

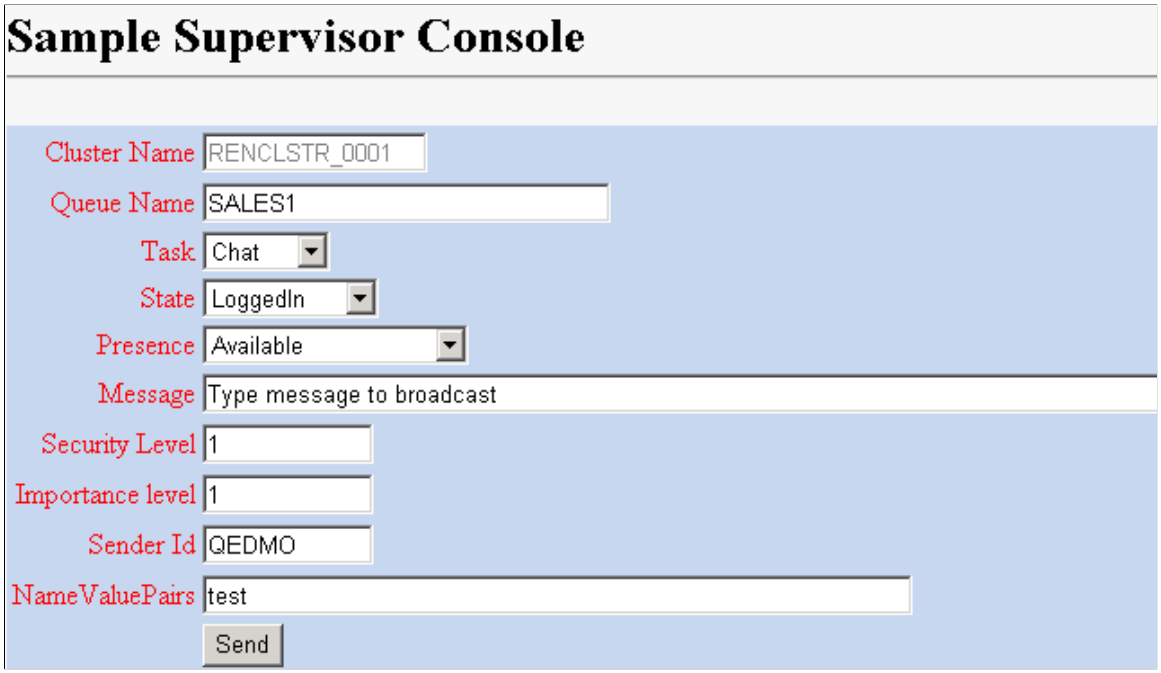
Access the MCF Supervisor console using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Universal Queue, Sample Pages. Choose an MCF Cluster ID and Click on the MCF Supervisor Console to open the MCF Supervisor Console.

Image: Sample Supervisor Console

This example illustrates the fields and controls on Sample Supervisor Console. You can find definitions for the fields and controls later on this page.



Cluster Name Displays the REN server cluster on which to test broadcast.

Queue Name Enter the name of the queue that is in the specified cluster.

Note: Enter the queue name only when you want to broadcast the message to a particular queue.

Task Select the task. Values are *email, chat, voice, generic,* and *none.*

State Select the state of the agent.

Note: If the state of the agent is *LoggedIn* when the broadcast message is being sent by the supervisor, the agent receives the broadcast message using Agent console.

Presence Select the presence.

Note: If the agent is logged out, the presence should be inActive. The broadcast message is received only when the agent is currently logged in to the queue on which broadcast is sent.

Message Enter the message that you want to broadcast.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: To view the broadcast message that is sent by the supervisor, use the Agent console sample page. Select the same REN cluster that is used by the supervisor to broadcast, and the broadcast message is displayed on the Agent console.

See Using the Agent Console Page.

Security Level (Optional) Enter the security level.

Importance Level (Optional) Enter the importance level.

Sender Id (Optional) Enter the sender ID. Use this option only when you want the sender ID to appear as something other than the ID that actually sent the message.

NameValue Pairs Enter the name and value pairs to configure your data.

Note: These name-value pairs are concatenated as a string. You can define any name-value parameters for your application and send them in the name-value pair string. The JSMCAPI passes the same as a string to the console. Your particular application needs to process that string accordingly.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: For a system-wide or a cluster-wide broadcast, the broadcast message from the supervisor is sent to all agents, as well as the supervisors who are logged in to the cluster that is specified by the supervisor. The system-wide broadcast is same as cluster-wide broadcast because a JSMCAPI supervisor knows only the cluster it is operating on.
For a queue-wide broadcast, the broadcast message from the supervisor is sent to all agents who are currently logged in to the queue that is used by the supervisor to broadcast the message. This assumes that the queue is in the same cluster as the supervisor who is sending the broadcast message.
For agent-wide broadcast, the broadcast message will be sent to all agents who are currently logged to any queue in the same cluster as the supervisor who is sending the message.

Image: Queue-wide broadcast

The following diagram illustrates queue-wide broadcast of notifications.

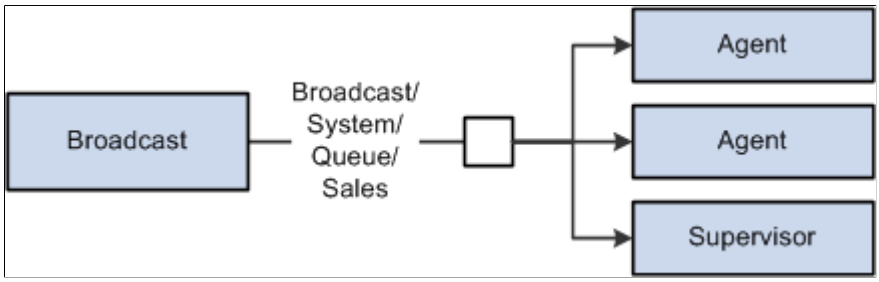
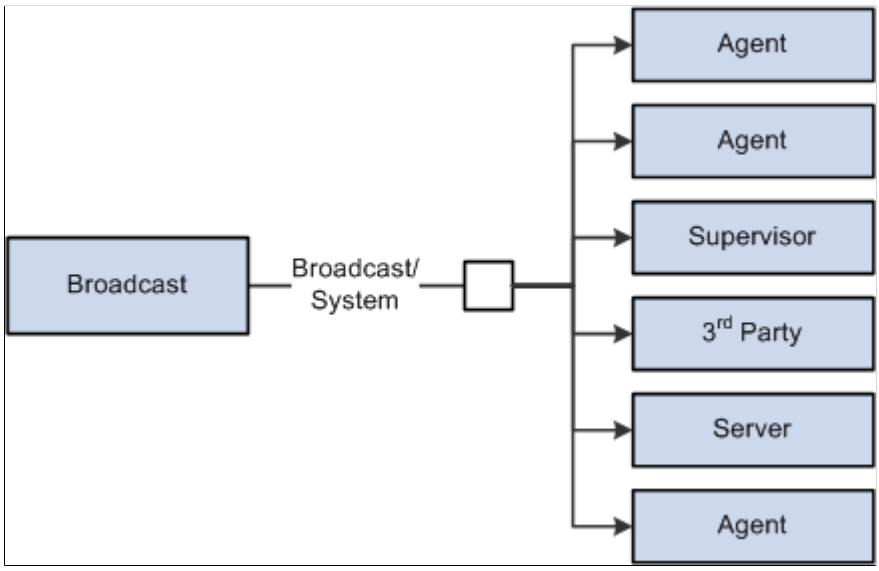


Image: Cluster-wide broadcast

The following diagram illustrates cluster-wide broadcast of notifications.



The following table describes the combinations:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Type of Broadcast	*Cluster*	*Queue*	*Agent Login State*	*Agent Activity State*	*Broadcast Audience and Description*
JSMCAPI	Not specified	Not specified	Not specified	Not specified	This is a cluster-wide broadcast. All agents on Cluster _1 will receive the broadcast.
JSMCAPI	Cluster_1	Not specified	Not specified	Not specified	Same as the preceding. All agents on Cluster _1 will receive the broadcast.
JSMCAPI	Cluster_1	SALES	Not specified	Not specified	All agents subscribing to SALES will receive the broadcast. This assumes that SALES is on Cluster_1.
JSMCAPI	Cluster_1	SALES	Logged	Not specified	All agents who are logged (active and inactive) to sales will receive the broadcast. This assumes that SALES is on Cluster_1.

Using PeopleCode Broadcast

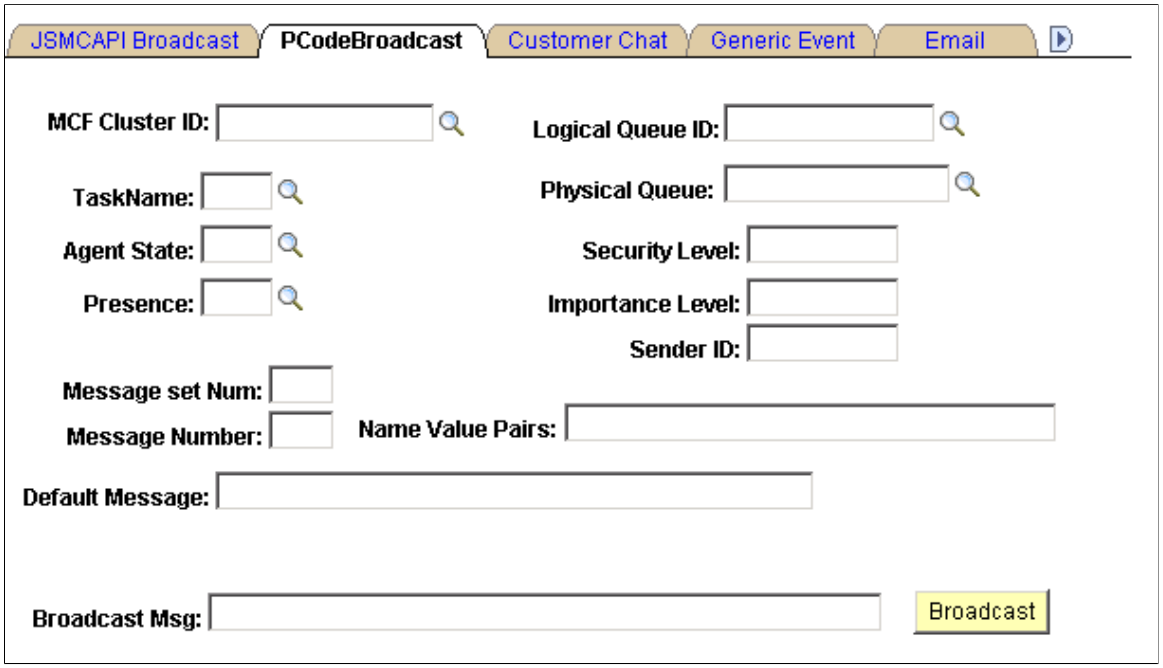
Access the PCodeBroadcast page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Universal Queue, Sample Pages, PCodeBroadcast

Image: PCodeBroadcast page

This example illustrates the fields and controls on the PCodeBroadcast page. You can find definitions for the fields and controls later on this page.



MCF Cluster ID Select the target cluster receiving the broadcast message.

Logical Queue ID Select the name of the target logical queue.

Physical Queue ID Select the name of the target physical queue.

TaskName Select the task. Select from *email, chat*, *voice*, *generic* or *none.*

Agent State Select the state of the agent.

Note: The agent receives the broadcast message only if the agent is logged in.

Presence Select the presence.

Note: If the agent is logged out, the presence should be *inActive.* All agents who are currently logged in receive the broadcast message.

Message Set Number Select the message set number if you want to broadcast a message from the Message Catalog.

Message Number Select the message number in the message set.

Default Message Enter the default message.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

If no message is found in Message Catalog with the preceding message number, the text of the default message is used for the broadcast instead.

Security Level (Optional) Enter the security level.

Importance Level (Optional) Enter the importance level.

Sender Id Enter the sender ID if you want the sender that is displayed with the notification to be an ID other than the one that sent the broadcast.

Message Enter the message that you want to broadcast.

Note: To view the broadcast message that is sent by the supervisor, use the Agent console sample page. Select the same REN cluster as used by the supervisor to broadcast, and the broadcast message is displayed on the Agent console.

See Using the Agent Console Page.

NameValue Pairs Enter the name and value pairs to configure your data.

Note: These name-value pairs are concatenated as a string. You can define any name-value parameters for your application and send them in the name-value pair string. The JSMCAPI passes the same as a string to the console. Your particular application needs to process that string accordingly

---

Note: For a system-wide PeopleCode Broadcast, the broadcast message from the supervisor is sent to all agents, as well as the supervisors on all active REN clusters.
For a cluster-wide PeopleCode Broadcast, the broadcast message from the supervisor is sent to all agents, as well as the supervisors on the specified REN clusters.
For a queue-wide broadcast, the broadcast message from the supervisor is sent to all agents who are currently logged in to the queue that is specified by the queue ID in all clusters if the MCF cluster ID is not specified. If the cluster ID is specified, all agents that are logged into the queue that is specified by the queue ID in the cluster that is specified by the cluster ID receive the broadcast message.
For agent-wide broadcast, if both the cluster ID and queue ID are specified, the broadcast message is sent to all agents that are currently logged into the queue that is specified by the queue ID. This assumes that the queue is in the cluster that is specified by the cluster ID. However, if only the cluster ID is mentioned, all agents that are logged and active on any queue on that cluster receive the broadcast message.

The following table describes the combinations:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Type of Broadcast	*Cluster*	*Queue*	*Agent Login State*	*Agent Activity State*	*Broadcast Audience and Description*
PeopleCode	Not specified	Not specified	Not specified	Not specified	This is a true system-wide broadcast. All agents on all active REN clusters receive the broadcast.
PeopleCode	Cluster_1	Not specified	Not specified	Not specified	All agents on Cluster_1 receive the broadcast.
PeopleCode	Cluster_1	SALES	Not specified	Not specified	All agents subscribing to SALES receive the broadcast. This assumes that SALES is on Cluster_1.
PeopleCode	Not specified	SALES	Logged	Not specified	All agents that are logged (active and inactive) to SALES (all clusters with physical queues that are associated with SALES) receive the broadcast.
PeopleCode	Cluster_1	SALES	Logged	Active	All agents that are logged into SALES and are in active state receive the broadcast. This assumes that SALES is on Cluster_1.
PeopleCode	Cluster_1	Not specified	Logged	Active	All agents that are logged and active on any queue on Cluster_1 receive the broadcast.
PeopleCode	Cluster_1	Not specified	Not specified	Active	All agents that are logged and active on any queue on Cluster_1 receive the broadcast. This is the same as the preceding because only agents that are logged in can be active.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Related Links

"MCFBroadcast" (PeopleTools 8.54: PeopleCode Language Reference)

Viewing Broadcast Logs

To view broadcast log, use the Broadcast log page.

See Viewing Broadcast Logs.

Working with Sample Pages

To demonstrate MCF tools and functionality, use the MCF Sample Pages (MCF_DEMO_CMP) component.

This section discusses how to:

- Use the Customer Chat sample page.
- Use the URL wizard.
- Use the Generic Event sample page.
- Use the Generic Event window.
- Use the Email sample page.
- Use the Email window.

Using the Customer Chat Sample Page

Access the Customer Chat page using the following navigation path:

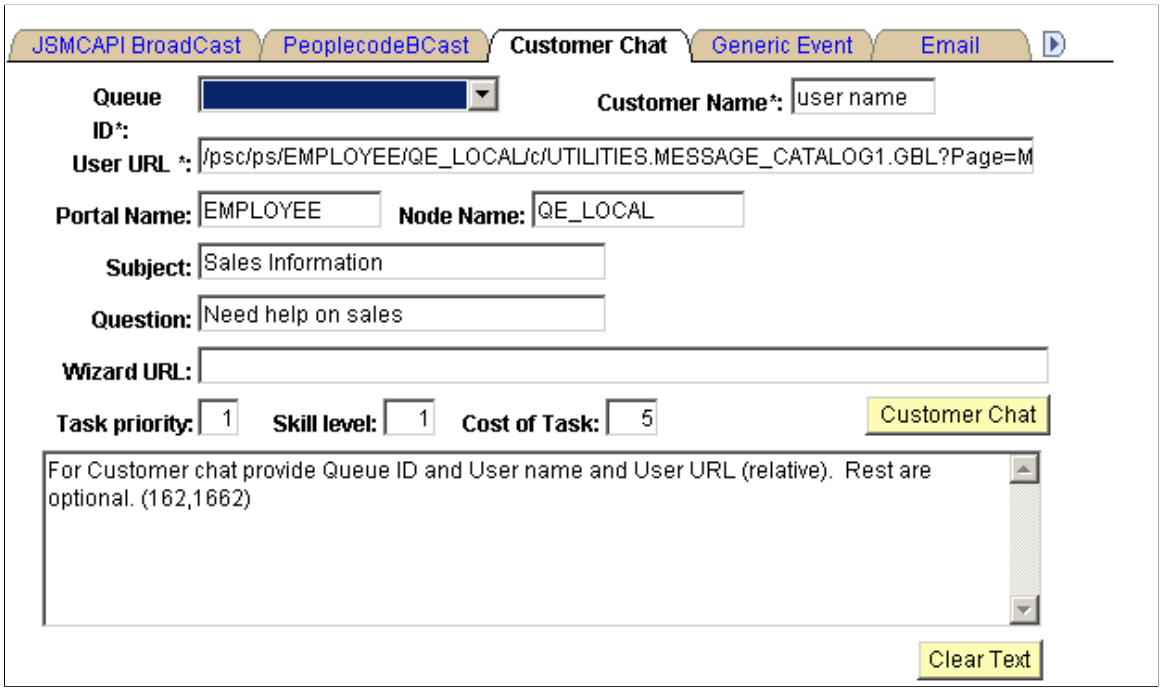
PeopleTools, MultiChannel Framework, Universal Queue, Sample Pages, Customer Chat

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: This page is available using both queue server and third-party routing server.

Image: Customer Chat page

This example illustrates the fields and controls on the Customer Chat page.



This page demonstrates the InitChat() built-in function, including required and optional parameters.

This page is not intended for production use. On a typical application page, the developer includes a Live Help or Customer Chat button that calls the InitChat() built-in function and passes required parameters, including the context of the chat request. The sample customer chat page demonstrates values that can be included in the InitChat() parameters. The user may not be prompted for any information.

To run a sample chat:

1. Access the MultiChannel Console.
2. Open the Customer Chat page.
3. Open the Queue ID drop-down list box and select the queue that you are logged on to.

   The other fields are automatically populated with values that generate a sample chat session. You can change the values as long as valid required values are included.

   The User URL is a required field that contains the application page URL to send to the agent chat browser.

   The optional Wizard URL field represents a grab URL; the actual page that appears is determined by InitChat() parameters.

   The text box displays error messages and additional information for using the sample page.
4. Click Customer Chat.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The customer chat window appears.

5. On the MultiChannel Console, click the flashing chat notification icon to accept the chat session.

   The agent chat window appears.

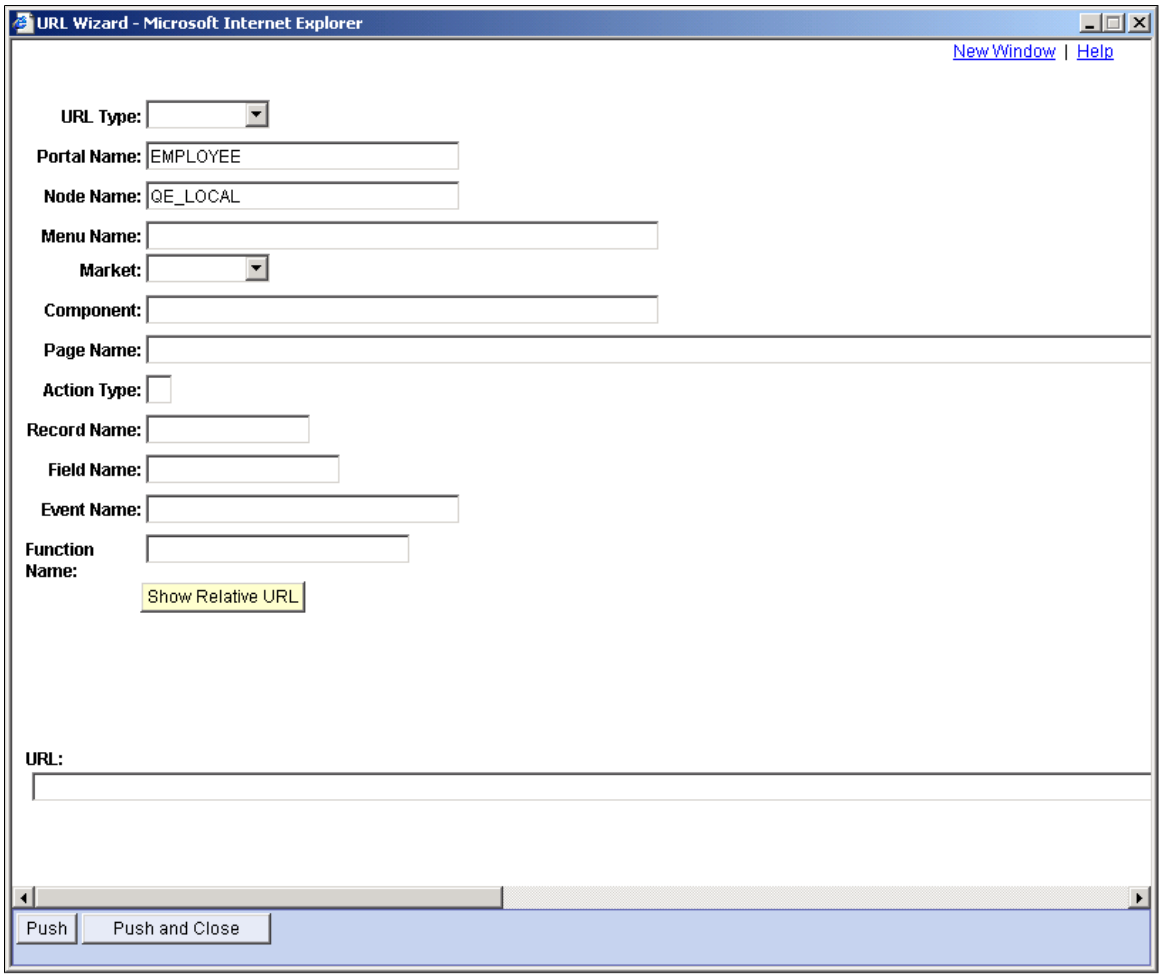
   You can enter text as both agent and customer to demonstrate chat functionality.

Using the URL Wizard

Click Select from the agent chat window during a chat session to open the URL wizard:

Image: URL Wizard page

This example illustrates the fields and controls on the URL Wizard page. You can find definitions for the fields and controls later on this page.



Use the wizard to form a URL to send to the customer. The URL Wizard page is a template that demonstrates constructing a URL and sending it to a customer. The URL wizard can be used as is or as the basis for developing URL generation for your application.

URL Type Select the type of URL that is being accessed. Values are:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- *Component:* Generates a PeopleSoft Pure Internet Architecture component URL.

  Component URL generation uses the PeopleCode built-in function **GenerateComponentContentRelURL**.

  See "GenerateComponentContentRelURL" (PeopleTools 8.54: PeopleCode Language Reference).

- *External:* Not currently used.

- *Gen URL:* Not currently used.

- *iScript:* Generates a PeopleSoft Pure Internet Architecture iScript URL.

  iScript URL generation uses the PeopleCode built-in function **GenerateScriptContentRelURL.**.

  See "GenerateScriptContentRelURL" (PeopleTools 8.54: PeopleCode Language Reference).

Fields that are not required by the selected URL type are not available.

Show Relative URL Click to display the relative URL that is generated by the values that you have entered into the page's fields.

URL Displays the generated relative URL.

Push Click to send the generated relative URL to the customer's chat window.

The Push button demonstrates functionality that must be included in an application-specific URL wizard.

Push and Close Click to send the generated relative URL to the customer's chat window and close the URL wizard.

The Push and Close button demonstrates functionality that must be included in an application-specific URL wizard.

Using the Generic Event Sample Page

Access the Generic Event page using the following navigation path:

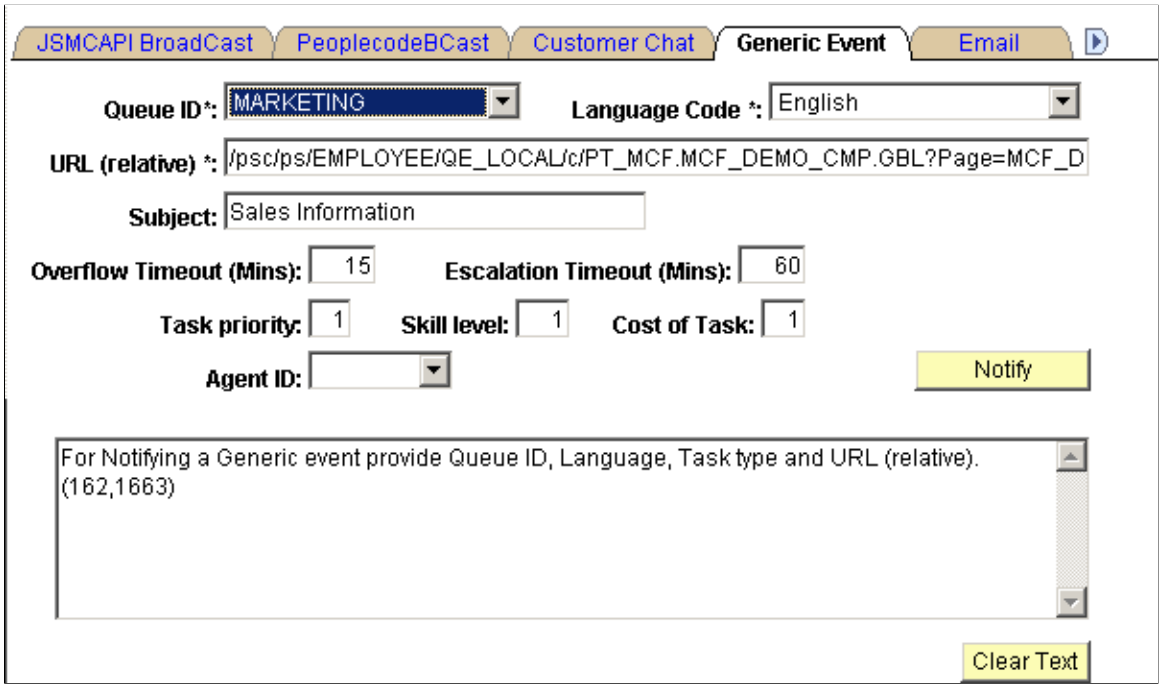
PeopleTools, MultiChannel Framework, Universal Queue, Sample Pages, Generic Event

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: This page is available using both queue server and third-party routing server.

Image: Generic Event page

This example illustrates the fields and controls on the Generic Event page.



This page demonstrates a generic persistent event using the EnQueue() and NotifyQ() built-in functions, including required and optional parameters.

This page is not intended for production use. On a typical application page, the developer includes an Enqueue or similar button that calls the EnQueue() built-in function and passes required parameters, including the context of the request. The sample customer chat page demonstrates values that can be included in the EnQueue() parameters. The user may not be prompted for any information.

To run a sample generic event:

1. Access the MultiChannel Console.
2. Open the Generic Event page.
3. Open the Queue ID drop-down list box, and select the queue that you are logged on to.
4. Open the Agent ID drop-down list box, and select the agent ID that you are logged on with.
5. Select *Generic* for the task type.
6. Click Notify.
7. On the MultiChannel Console, click the flashing event notification icon to accept the event.

   The Generic Event window appears.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

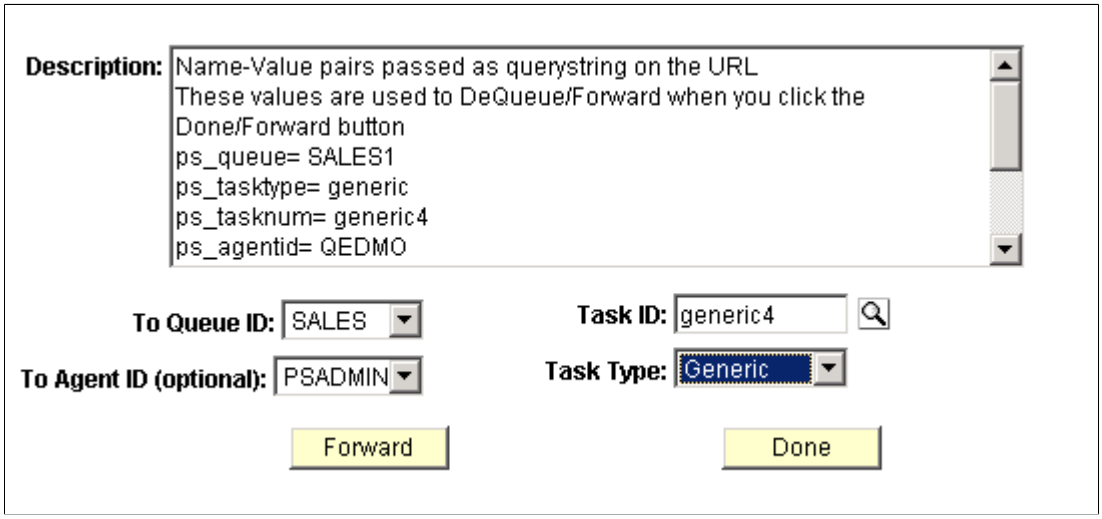
Using the Generic Event Window

The format of the generic event window is determined by application developers. PeopleSoft supplies a sample generic event window to demonstrate the available functionality, including the DeQueue() built-in.

Configure the size and initial location of the generic event window on the Agent Window Configuration page:

Image: Generic Event window

This example illustrates the fields and controls on the Generic Event window. You can find definitions for the fields and controls later on this page.



Description	Displays text from the generic event.
To Queue ID	Select a queue to which the generic event will be forwarded.
Task ID	Select the task ID of the generic event to forward.
To Agent ID	Select the agent ID to whom the generic event will be forwarded.
Task Type	Select the task type of the generic event to be forwarded.
Forward	Click to send the generic event to the selected agent or queue. This button demonstrates the functionality of the Forward() built-in function.
Done	Click to notify the real-time event notification (REN) server that you are done with this task. A message asks if you want to close the window. The Done button demonstrates the functionality of the DeQueue() built-in function.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Related Links

"DeQueue" (PeopleTools 8.54: PeopleCode Language Reference)

Using the Email Sample Page

Access the Email page using the following navigation path:

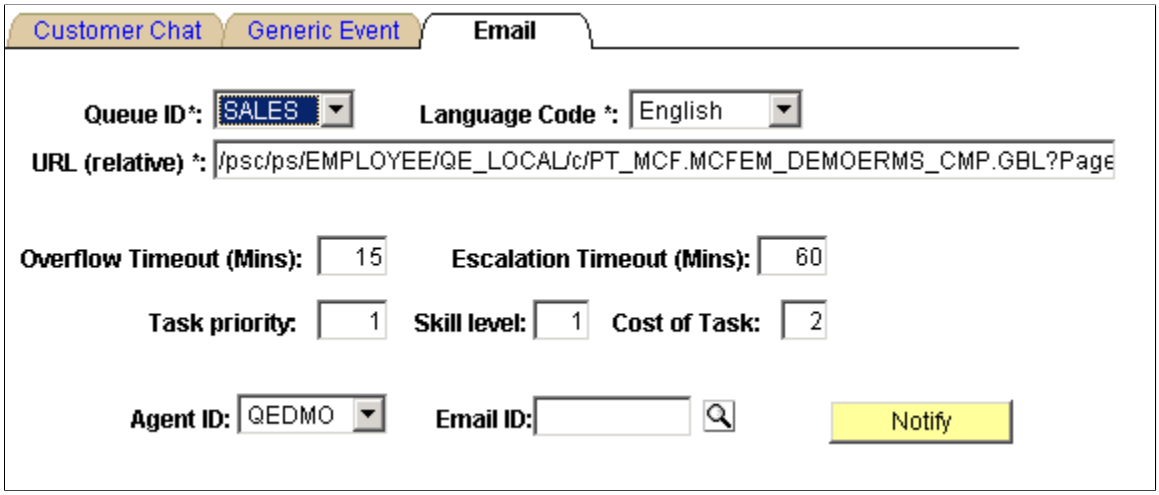
PeopleTools, MultiChannel Framework, Universal Queue, Sample Pages, Email

---

Note: This page is available using both queue server and third-party routing server.

Image: Email page

This example illustrates the fields and controls on the Email page.



The email sample pages are intended for demonstration purposes and should not be used in production.

EnQueue() and NotifyQ() can also be called from PeopleSoft Application Engine batch programs.

See "EnQueue" (PeopleTools 8.54: PeopleCode Language Reference) and "NotifyQ" (PeopleTools 8.54: PeopleCode Language Reference).

Note: For you to fully demonstrate the functionality of the Email sample page, an email must be read and written to the email database using the GetMail - Server sample page.

To process a sample email:

1. Open the Queue ID drop-down list box and select the queue that you are logged on to.
2. Open the Agent ID drop-down list box and select the agent ID that you are logged on with.
3. Click Notify.
4. On the MultiChannel Console, click the flashing email notification icon to accept the email.

   The email window appears.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

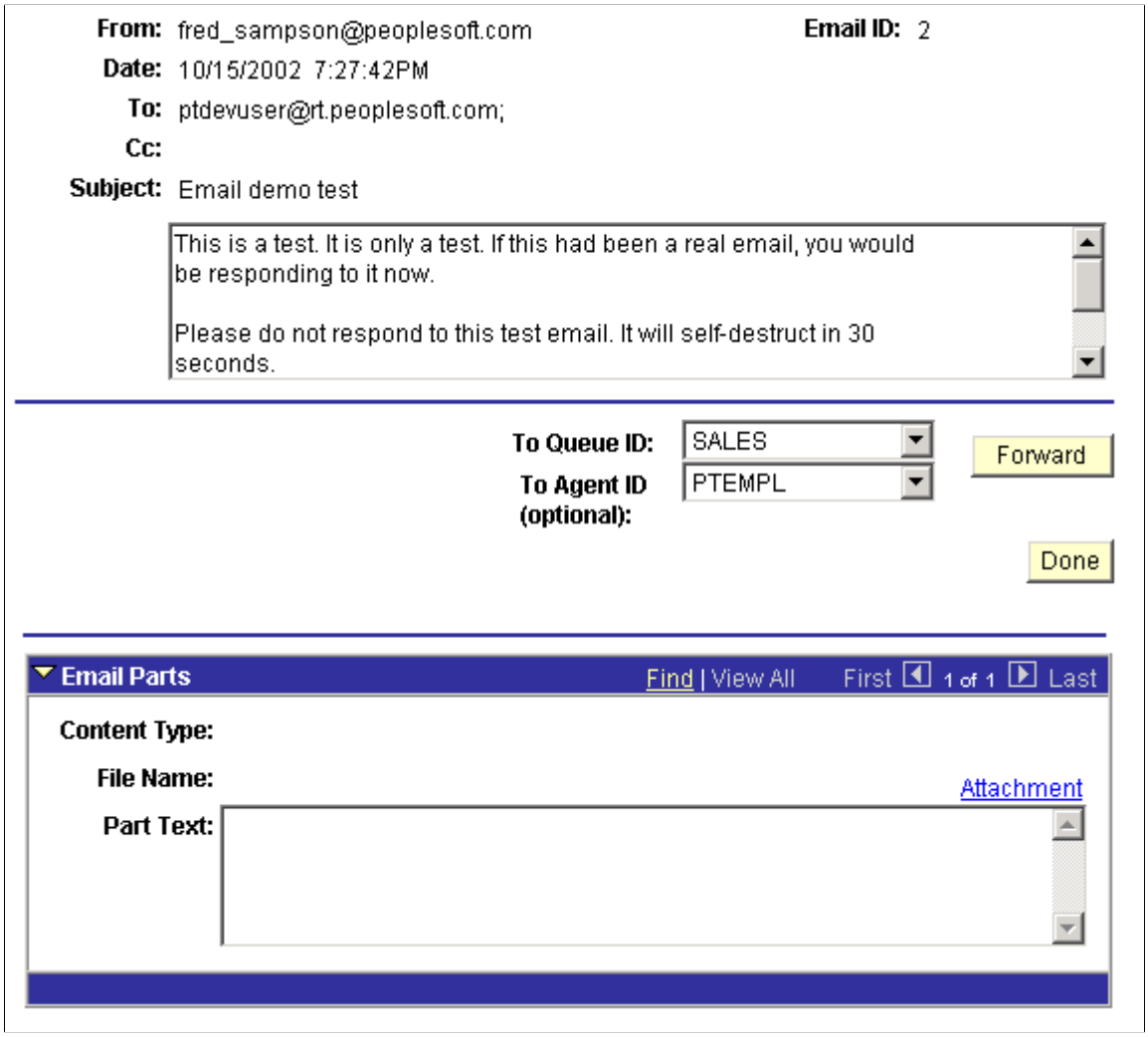
Using the Email Window

The format of the email window is determined by application developers. PeopleSoft supplies a sample email event window to demonstrate available functionality.

Configure the size and initial location of the email event window on the agent's Window Configuration page.

Image: Email window

This example illustrates the fields and controls on the Email window. You can find definitions for the fields and controls later on this page.



To Queue ID — To forward the email to another queue, select a queue ID from the drop-down list box and click Forward.

To Agent ID — To forward the email to another agent, select an agent ID from the drop-down list box and click Forward.

Forward — Click to forward the email to the specified queue or agent.

This button demonstrates the functionality of the Forward() built-in function.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Done Click to quit the email and remove it from the queue.

The Done button demonstrates the functionality of the DeQueue() built-in.

Email Parts

If an email has been divided into parts stored in the email database, or has an attachment, they can be accessed here.

Related Links

Demonstrating the Email Channel

"Understanding PeopleSoft MultiChannel Framework Mail Classes" (PeopleTools 8.54: PeopleCode API Reference)

Using and Demonstrating JSMCAPI

To demonstrate MCF consoles and tools that use JSMCAPI, use the CTI Sample Pages (PT_CTI_DEMOOUTB) and MCF Sample Pages (MCF_DEMO_CMP) components.

This section provides an overview of JSMCAPI and discusses how to:

- Use the CTI Sample Console page and Sample CTI Console.
- Use the Agent Console page and Sample UQ Agent Console.
- Use the Monitor Agents page and Sample Monitor - Agent States page.
- Use the Monitor Queues page Sample Monitor - Queue Statistics page.

Understanding JSMCAPI

The JSMCAPI enables custom configuration of MCF consoles (including the CTI console), MCF functionality on a PeopleSoft Pure Internet Architecture page, and queue and agent monitoring. For example, developers can create supervisor dashboards with which to monitor activity on their channels of interest, or developers can modify consoles according to their company's business requirements. The JSMCAPI builds on the REN JavaScript client. JSMCAPI uses standard JavaScript.

PeopleSoft provides sample pages demonstrating the functionality that is enabled by JSMCAPI. These pages are for demonstration purposes only, and should not be used in a production environment.

JSMCAPI is delivered with PeopleTools; you do not need a separate installation. The jsmcapi.js file is located in the <PIA_HOME>\webserv\<domain>\applications\peoplesoft\PORTAL\ps\pMCF folder.

Understanding Reason Objects and Reason Codes

PeopleSoft MCF provides reason codes to explain any error or an event that has happened. The third party vendors uses the same reason codes as provided by PeopleSoft MCF to create reason objects that they send with the events. PSMCAPI internally parses the event, gets the reason code, and sends it to JSMCAPI with the event. The CTI sample console or third-party multichannel console looks up a mapper

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

table that maintains the mapping of the reason codes to the Message Catalog entries and displays the reason code messages associated with the error or event that was sent by PSMCAPI.

To display the reason code messages, the sample console picks up all the reason codes and reason messages from the Message Catalogue. Whenever an error occurs, the JavaScript function will extract the reasoncode from the reason object in event and look for the reason message. Then, the JavaScript code prepares the reason message from the fields of reason object and displays the reason message in a new small window.

Reason codes can also be send with requests from JSMCAPI side. The JavaScript function will prompt for the reason codes and any extra data as part of reason.

The following table lists the reason codes and their corresponding reason messages:

Reason Code	***Description/Message Entry***
0	System or general error has occurred.
1	See provider for details.
2	Extension or agent is busy.
3	No answer.
4	Task is in transfer.
5	Task is in conference.
6	Task is abandoned.
7	Searching for an agent.
8	Incoming task.
9	Task is assigned.
10	Connection is established.
11	User or Task data is updated.
12	Request is completed successfully.
13	State of the agent, group, or extension changed.
14	New task is initiated.
15	Conference is not successful.
16	Transfer is not successful.
17	Retrieve is not successful.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Reason Code	***Description/Message Entry***
18	Forward is not successful.
19	Reject is not successful.
20	Revoke is not Successful.
21	Task is revoked.
22	Task is withdrawn.
23	Do Not Disturb is turned on.
24	Do Not Disturb is turned off.
25	Forwarding of calls is set.
26	Forwarding of calls is canceled.
27	Phone is on hook.
28	Phone is off hook.
29	New call is initiated.
30	Call is put on hold.
31	Call is parked.
32	Call is cleared.
33	Call is alternated between hold and active states.
34	Call is taken off hold.
35	Hold is not successful.
36	Invalid session.
37	Invalid State.
38	Incorrect Data.
39	Unsupported Feature.
40	Configuration Error.
41	Server is on.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Reason Code	***Description/Message Entry***
42	Server is off.
43	Agent is Busy.
44	Unexpected Error.

The following table maps the reason codes to the entries in the Message Catalog:

Message Number ***MCFMSGNUM***	***Message Set Number*** ***MCFMSGSETNUM***	***Reason Code Number*** ***REASONCODENO***
2790	162	0
2791	162	1
2792	162	2
2793	162	3
2794	162	4
2795	162	5
2796	162	6
2797	162	7
2798	162	8
2799	162	9
2800	162	10
2801	162	11
2802	162	12
2803	162	13
2804	162	14
2805	162	15
2806	162	16
2807	162	17

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Message Number ***MCFMSGNUM***	***Message Set Number*** ***MCFMSGSETNUM***	***Reason Code Number*** ***REASONCODENO***
2808	162	18
2809	162	19
2810	162	20
2811	162	21
2812	162	22
2813	162	23
2814	162	24
2815	162	25
2816	162	26
2817	162	27
2818	162	28
2819	162	29
2820	162	30
2821	162	31
2822	162	32
2823	162	33
2824	162	34
2825	162	35
2826	162	36
2827	162	37
2828	162	38
2829	162	39
2830	162	40

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Message Number *MCFMSGNUM*	*Message Set Number* *MCFMSGSETNUM*	*Reason Code Number* *REASONCODENO*
2831	162	41
2832	162	42
2833	162	43
2834	162	44

Related Links

Understanding JSMCAPI Classes
JSMCAPI Classes

Common Elements Used in This Section

This section discusses common elements used on the CTI sample pages.

Common Elements Used in CTI Sample Pages

The following common elements are used in CTI sample pages.

Agent Id Select the CTI or MCF agent's agent ID.

In the case of a CTI agent, the agent ID may be different from the agent's user ID.

g/e/c (generic, email, chat) Represents three task types: generic, email, chat.

Click to initiate the action that is selected in the associated drop-down list box.

MCF Cluster ID Select the MCF cluster to be monitored or on which to test the sample console.

Physical Queue Select the physical queue to be monitored.

Each MCF logical queue can comprise one or more physical queues. Because each physical queue is associated with an MCF cluster, only physical queues can be monitored.

State Displays a value indicating the state of the associated element.

Statistics Display statistics that are generated by the CTI server.

User Id Displays the CTI agent's PeopleSoft user ID.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

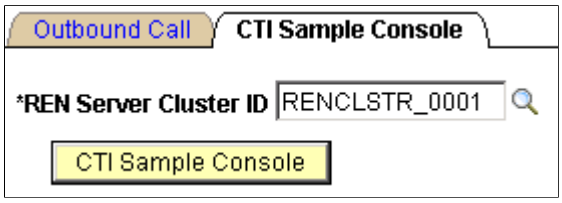
Using the CTI Sample Console

Access the CTI Sample Console page using the following navigation path:

PeopleTools, MultiChannel Framework, CTI Configuration, Sample Pages, CTI Sample Console

Image: CTI Sample Console page

This example illustrates the fields and controls on the CTI Sample Console page. You can find definitions for the fields and controls later on this page.



REN Server Cluster ID Select the REN server cluster on which to test the sample console.

CTI Sample Console Click to initiate the sample console.

Note: To demonstrate the CTI sample console, you must have a CTI server running and communicating with the specified REN server cluster.

After clicking CTI Sample Console, a new browser window appears that displays the sample console along with a tracer window. The Sample CTI Console contains the following group boxes which are explained in detail in the following sections:

- Open/Close Session
- Register User to Session
- Register Group, Login User to Group, and User-Group States
- Register Extension to Session, and Extension Operations
- Buddies
- Error Message/Information
- Server State and Broadcasts

Note: To enable the new browser window to appear, disable any pop-up blocking software for your browser.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Open/Close Session

Access the Open/Close Session group box.

Image: Open/Close Session group box

This example illustrates the fields and controls on the Open/Close Session group box. You can find definitions for the fields and controls later on this page.

The Open/Close Session group box displays information about the session.

CTI Session Id This field displays a value identifying the session.

If no session ID is displayed, check to ensure that the CTI server and REN server are both running.

You can close an active session by clicking the Go button. You cannot make a selection from the drop-down list box.

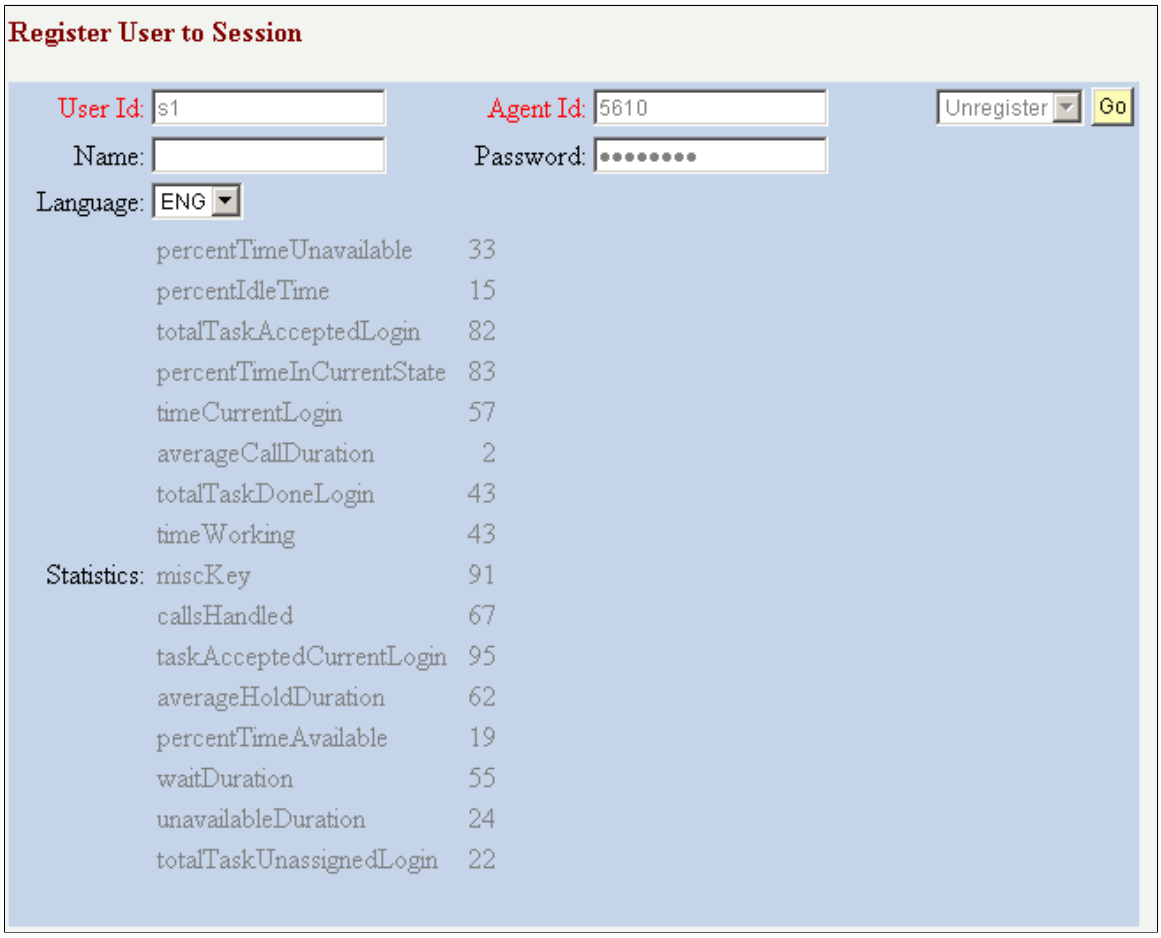
 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Register User to Session

Access the Register User to Session group box.

Image: Register User to Session group box

This example illustrates the fields and controls on the Register User to Session group box. You can find definitions for the fields and controls later on this page.



Register a user to the session in the Register User to Session group box.

Because a CTI agent can have an agent ID different from the agent's user ID, both values are required. If the agent has a password, the password is also required.

Language Reserved for future use.

The default language is English.

You can register or unregister an agent by clicking the Go button. You cannot make a selection from the drop-down list box.

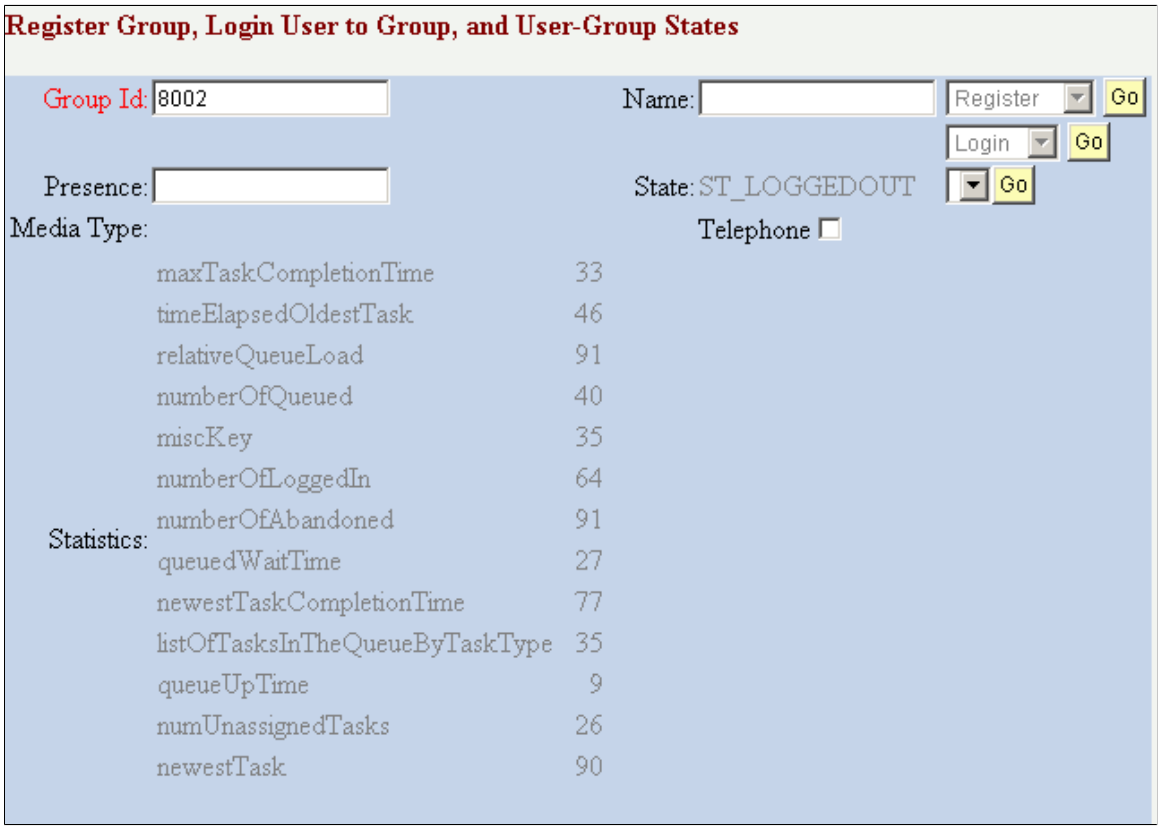
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Register Group, Login User to Group, and User-Group States

Access the Register Group, Login User to Group, and User-Group States group box.

Image: Register Group, Login User to Group, and User Group States group box

This example illustrates the fields and controls on the Register Group, Login User to Group, and User Group States group box. You can find definitions for the fields and controls later on this page.



Group Id

Enter the group ID to which the agent will be registered.

To register the agent with the specified group, click the Go button. You cannot make a selection from the drop-down list box.

Login

Click the Go button to log in if *Login* appears in the drop-down list box.

Click the Go button to log out if *Logout* appears in the drop-down list box.

Note: The agent cannot receive calls until the agent is in the Ready state.

Note: After you login to the group, a new pop-up window opens prompting for reason code data. If you register to an invalid group, the system displays an error message.

See Understanding JSMCAPI.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Presence

Enter a presence description to associate with the state that is selected in the drop-down list box.

This presence value is different from any presence values that are predefined in the agent configuration. The agent cannot select from the pre-defined values.

State

Select an agent state.

State options are:

- *Ready*
- *Not Ready*
- *Work Ready*
- *Work Not Ready*

Each state value has an associated state code that appears in the State field.

To enable incoming calls, select *Ready* from the drop-down list box and click the Go button.

Media Type

Reserved for future use.

The sample console supports only the telephone (voice) channel. However, JSMCAPI can support other channels.

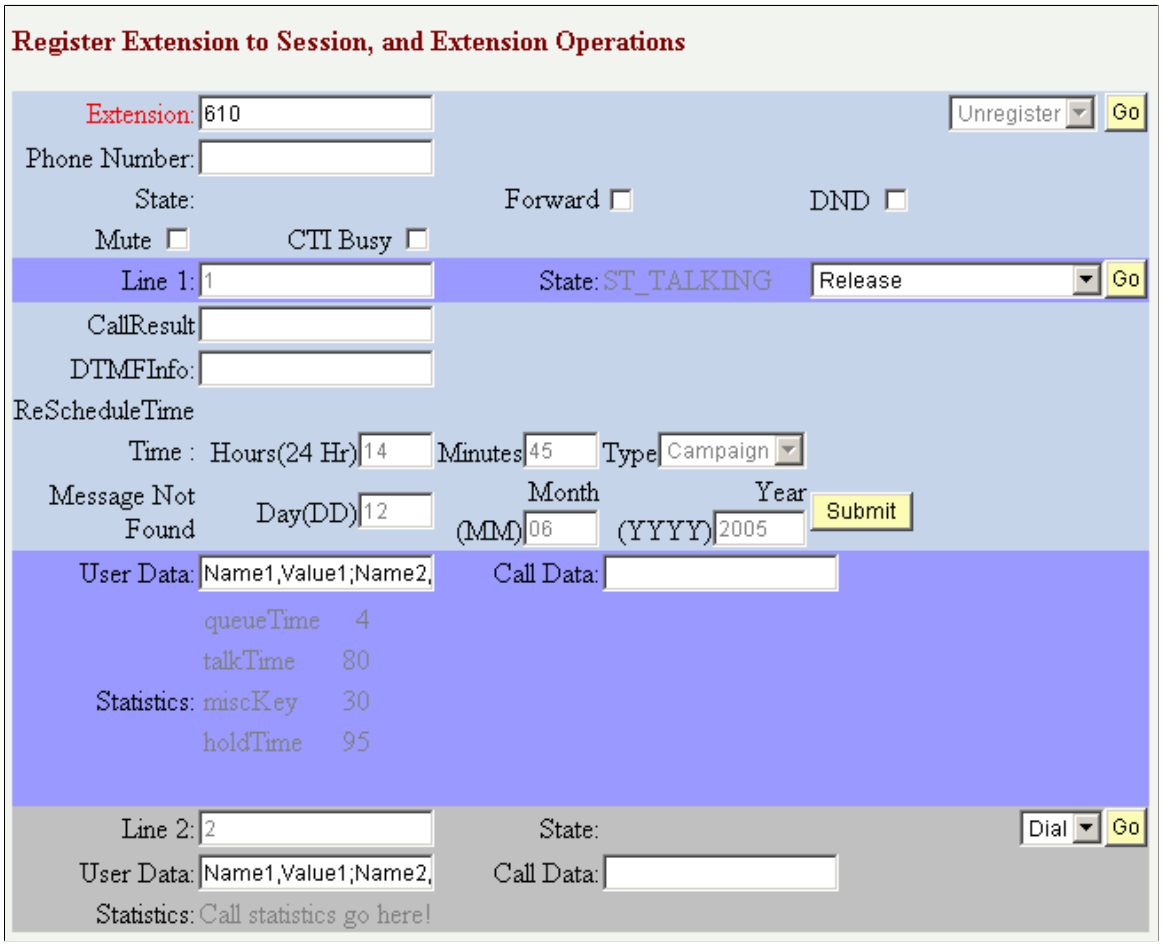
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Register Extension to Session and Extension Operations

Access the Register Extension to Session, and Extension Operations group box.

Image: Register Extension to Session and Extension Operations group box

This example illustrates the fields and controls on the Register Extension to Session and Extension Operations group box. You can find definitions for the fields and controls later on this page.



Extension Enter a valid telephone extension number.

Phone Number Enter a valid telephone number for use in subsequent operations, such as dial, forward, or do-not-disturb (DND).

Mute Select to mute a call.

Note: This is enabled only when the agent is on a call.

CTI Busy This flag is checked by the queue server for an agent handling a voice task.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: If the flag is checked, the queue server will not assign new tasks to such an agent. The CTI Busy flag allows the application to implement a request which conveys the status of the agent to the queue server. Because the queue server stores the task list and the status in its temporary cache, a recovered or a rebooted queue server will not have the information of a agent's previous CTI status. The agent can send its CTI status to the queue server to indicate that it is busy with a CTI task.

Forward	Select to forward incoming calls to the specified number.
DND (do not disturb)	Select to put the selected extension in do-not-disturb status.
Line 1 or Line 2	Displays a name or value for each line. On the sample console, the value will either be *1* or *2*.
DTMFInfo (Dual-tone Multi frequency information)	Select to send DTMF digits on the line
CallResult	Select to set call result on the line
ReScheduleTime	Select to set reschedule time on the line
Type	Select to set reschedule type on the line.
User Data	Enter name-value pairs representing user data to be attached to a call.
Call Data	Enter any call data to be attached to the call. Call data includes: • *ANI* • *DNIS* • *THISDN* • *OTHERDN*

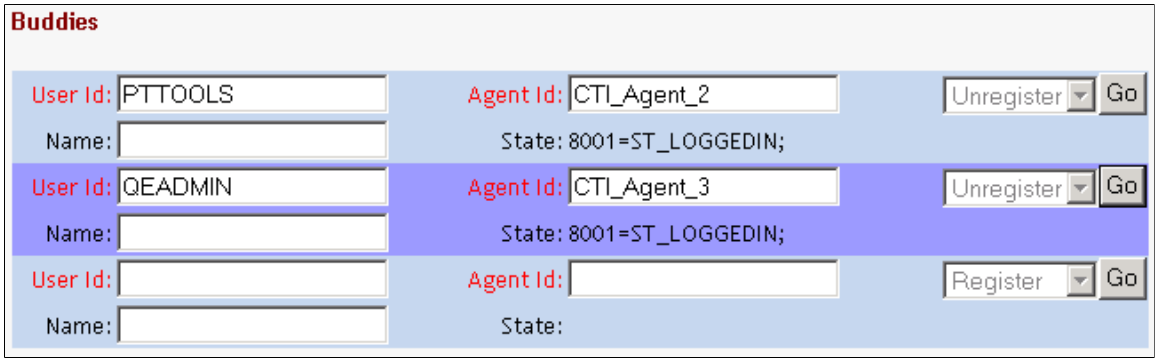
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Buddies

Access the Buddies group box.

Image: Buddies group box

This example illustrates the fields and controls on the Buddies group box.



Before an agent can chat with a buddy, the buddy must be registered in the Buddies group box.

Because a CTI agent can have an agent ID that is different from the agent's user ID, both values are required.

Note: If one agent is logged on the multichannel console and another agent is logged on the sample pages, the agents may see an incorrect buddy state for each other.

Error Messages/Information

Access the Error Messages/Information group box.

Image: Error Messages/Information group box

This example illustrates the fields and controls on the Error Messages/Information group box. You can find definitions for the fields and controls later on this page.

Any error messages associated with the process display in this section.

Clear Click the button to clear a displayed error message.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Server State and Broadcasts

Access the Server State and Broadcasts group box.

Image: Server State and Broadcasts group box

This example illustrates the fields and controls on the Server State and Broadcasts group box. You can find definitions for the fields and controls later on this page.

CTI State Displays the status of the CTI server.

REN State Displays the status of the REN server.

Info (information) Displays the text of a broadcast message from the CTI server.

Tracer

The tracer displays all requests between the CTI server and the console, which can be useful for debugging.

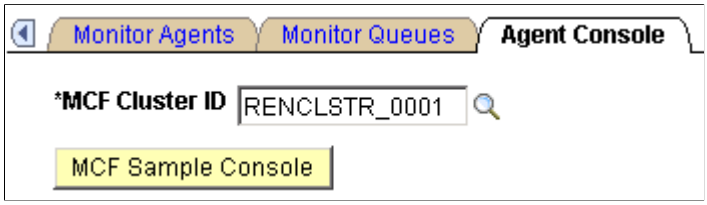
Using the Agent Console Page

Access the Agent Console page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Sample Pages, Agent Console tab

Image: Agent Console page

This example illustrates the fields and controls on the Agent Console page. You can find definitions for the fields and controls later on this page.



MCF Sample Console Click to initiate the sample console.

Note: To demonstrate the MCF sample console, you must have an MCF cluster running and communicating with the specified REN server cluster.

After you click MCF Sample Console, a new browser window displaying the sample console appears along with a Tracer Window. The MCF Sample Page contains the following group boxes which are explained in detail in the following sections:

- Open/Close Session
- Register User to Session

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Register Group, Login User to Group, and User-Group States
- Address Operations
- Buddies
- Error Message/Information
- Server State and Broadcasts

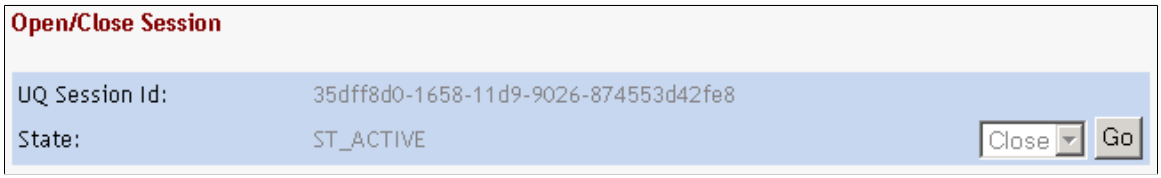
Note: To enable the new browser window to appear, disable any pop-up blocking software for your browser.

Open/Close Session

Access the Open/Close Session group box.

Image: Open/Close Session group box

This example illustrates the fields and controls on the Open/Close Session group box. You can find definitions for the fields and controls later on this page.



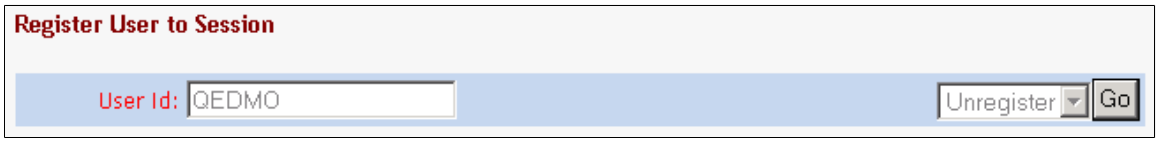
UQ Session Id The universal queue session ID.

Register User to Session

Access the Register User to Session group box.

Image: Register User to Session group box

This example illustrates the fields and controls on the Register User to Session group box.



Register Group, Login User to Group, and User-Group States

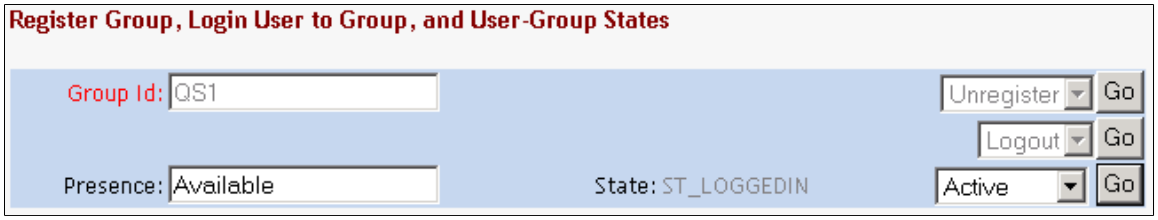
Access the Register Group, Login User to Group, and User-Group States group box.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: The presence change happens only when you change from active to inactive or from inactive to active. When you log in, the default value is active and therefore no presence change occurs.

Image: Register Group, Login User to Group, and User-Group States group box

This example illustrates the fields and controls on the Register Group, Login User to Group, and User-Group States group box.

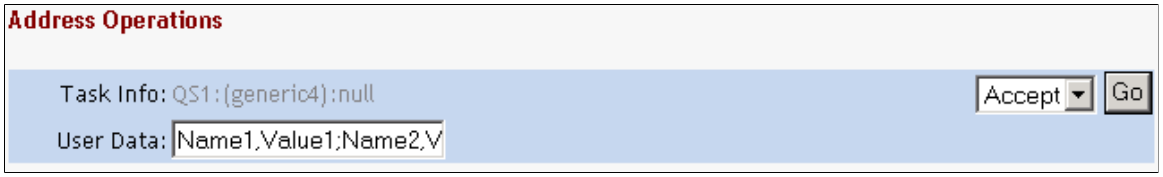


Address Operations

Access the Address Operations group box.

Image: Address Operations group box

This example illustrates the fields and controls on the Address Operations group box.

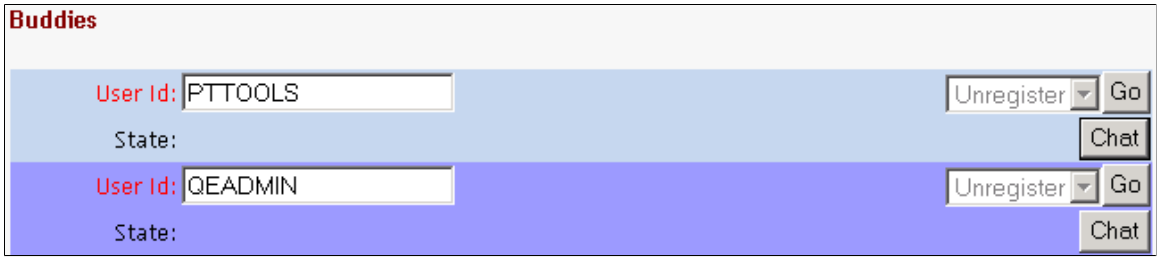


Buddies

Access the Buddies group box.

Image: Buddies group box

This example illustrates the fields and controls on the Buddies group box.



Note: If one agent is logged on the multichannel console and another agent is logged on the sample pages, the agents may see an incorrect buddy state for each other.

Note: The buddy states on this page are not automatically displayed. You must monitor an agent first using the Monitor Agents page before the buddy states are displayed.

See Using the Monitor Agents Page and Sample Monitor - Agent States Page.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

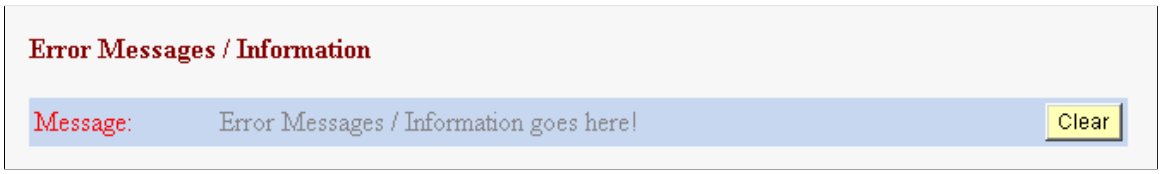
Click to initiate an agent-to-agent chat session.

Error Messages/Information

Access the Error Messages/Information group box.

Image: Error Messages/Information group box

This example illustrates the fields and controls on the Error Messages/Information group box.



Any error messages associated with the process display here.

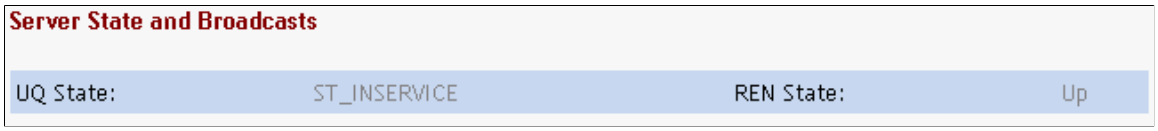
Clear Click the button to clear a displayed error message.

Server State and Broadcasts

Access the Server State and Broadcasts group box.

Image: Server State and Broadcasts group box

This example illustrates the fields and controls on the Server State and Broadcasts group box. You can find definitions for the fields and controls later on this page.



UQ State Displays the state of the queue server.

REN State Displays the state of the REN server.

Using the Monitor Agents Page and Sample Monitor - Agent States Page

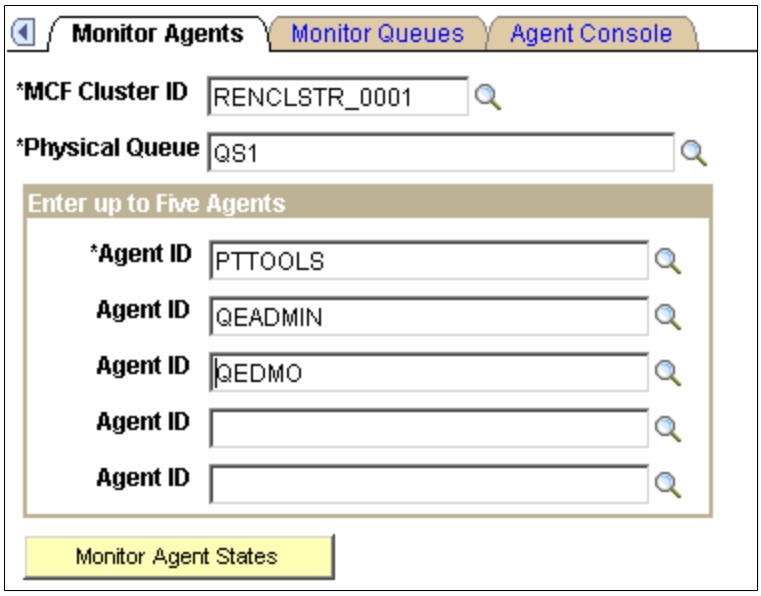
Access the Monitor Agents page using the following navigation path:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Universal Queue, Sample Pages, Monitor Agents tab

Image: Monitor Agents page

This example illustrates the fields and controls on the Monitor Agents page.



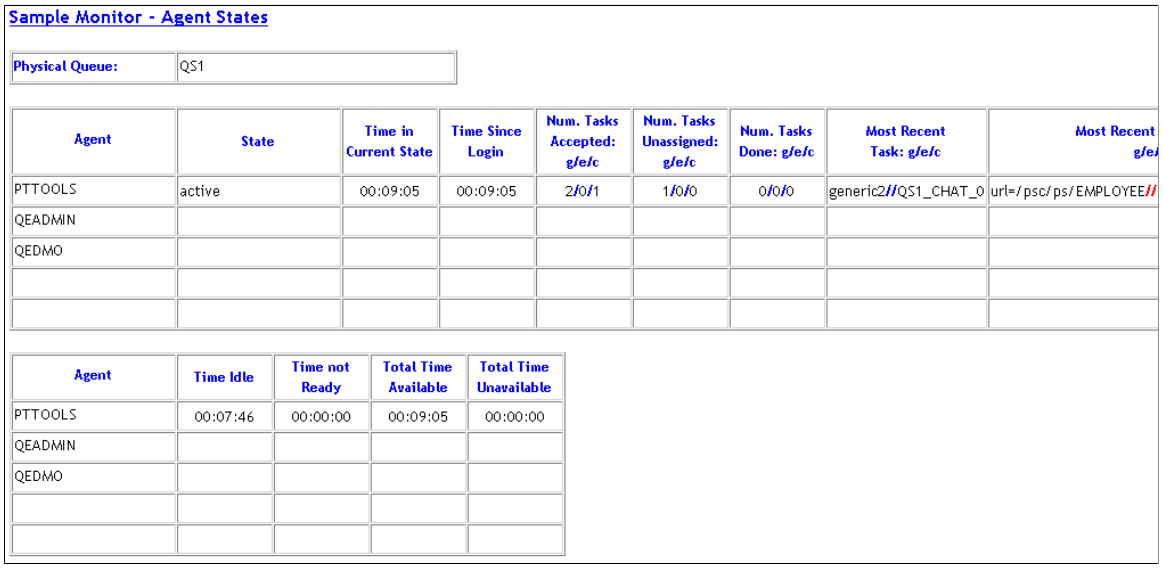
Select the MCF cluster, physical queue on that cluster, and up to five agents on that queue to be monitored.

After you click Monitor Agent States, a new browser window appears.

Note: To enable the new browser window to appear, disable any pop-up blocking software for your browser.

Image: Sample Monitor - Agent States page

This example illustrates the fields and controls on the Sample Monitor - Agent States page. You can find definitions for the fields and controls later on this page.



Sample Monitor - Agent States

Physical Queue:	QS1

Agent	State	Time in Current State	Time Since Login	Num. Tasks Accepted: g/e/c	Num. Tasks Unassigned: g/e/c	Num. Tasks Done: g/e/c	Most Recent Task: g/e/c	Most Recent g/e/
PTTOOLS	active	00:09:05	00:09:05	2/0/1	1/0/0	0/0/0	generic2//QS1_CHAT_0	url=/psc/ps/EMPLOYEE//
QEADMIN								
QEDMO								

Agent	Time Idle	Time not Ready	Total Time Available	Total Time Unavailable
PTTOOLS	00:07:46	00:00:00	00:09:05	00:00:00
QEADMIN				
QEDMO				

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Physical Queue

The ID of the physical queue that is being monitored appears in the first table.

Agent Statistics Table 1

The first agent statistics table displays information that is published by the JSMCAPI user.onStat1 event.

Time in Current State	Displays the duration of the agent's current state. The value returned is `timeInCurrentState`.
Time Since Login	Denotes the duration since the agent logged in. The value returned is `timeLogin`.
Num. Tasks Accepted: g/e/c (number of tasks accepted)	Displays the number of tasks in each task type that the agent has accepted since login. The value returned is `numTaskAccepted`.
Num. Tasks Unassigned: g/e/c (number of tasks unassigned)	Displays the number of tasks in each task type that have been unassigned from the agent. Returned by `numTasksUnassigned`.
Num. Tasks Done: g/e/c (number of tasks done)	Displays the number of tasks in each task type that the agent has completed. Returned by `numTasksDone`.
Most Recent Task: g/e/c	Identifies the most recent task that was accepted since login for each task type. Returned by `mostRecentTaskId`.
Most Recent Task Data: g/e/c	Displays data on the most recent task that was accepted since login for each task type. Returned by `mostRecentTaskInfo`

Agent Statistics Table 2

The second agent statistics table displays information that is published by the JSMCAPI user.onStat2 event.

Time Idle	Displays the duration of time during which the agent has been idle. An agent is considered to be idle from the time the agent logs on until the agent accepts a task, and during the time between completing the last accepted task and accepting another task. The value returned is `timeIdle`.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Time not Ready

Displays the duration of time during which the queue could not assign tasks to the agent because the agent's maximum workload has been reached.

The value returned is `timeNotReady`.

Total Time Available

Displays the total amount of time that the agent has been in an available status.

The value returned is `totalTimeAvailable`.

Total Time Unavailable

Displays the total amount of time that the agent has been in an unavailable status.

The value returned is `totalTimeUnavailable`.

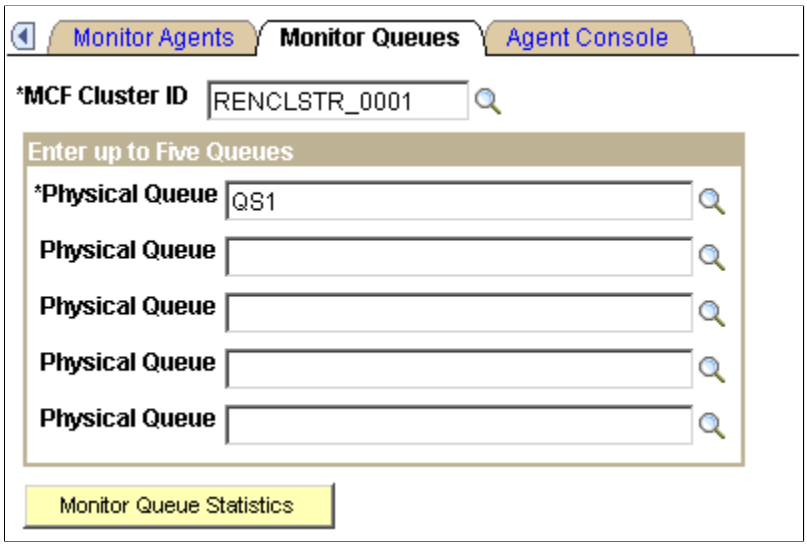
Using the Monitor Queues Page and Sample Monitor - Queue Statistics Page

Access the Monitor Queues page using the following navigation path:

PeopleTools, MultiChannel Framework, Universal Queue, Sample Pages, Monitor Queues tab

Image: Monitor Queues page

This example illustrates the fields and controls on the Monitor Queues page.



Select the MCF cluster and up to five physical queues on that cluster to be monitored.

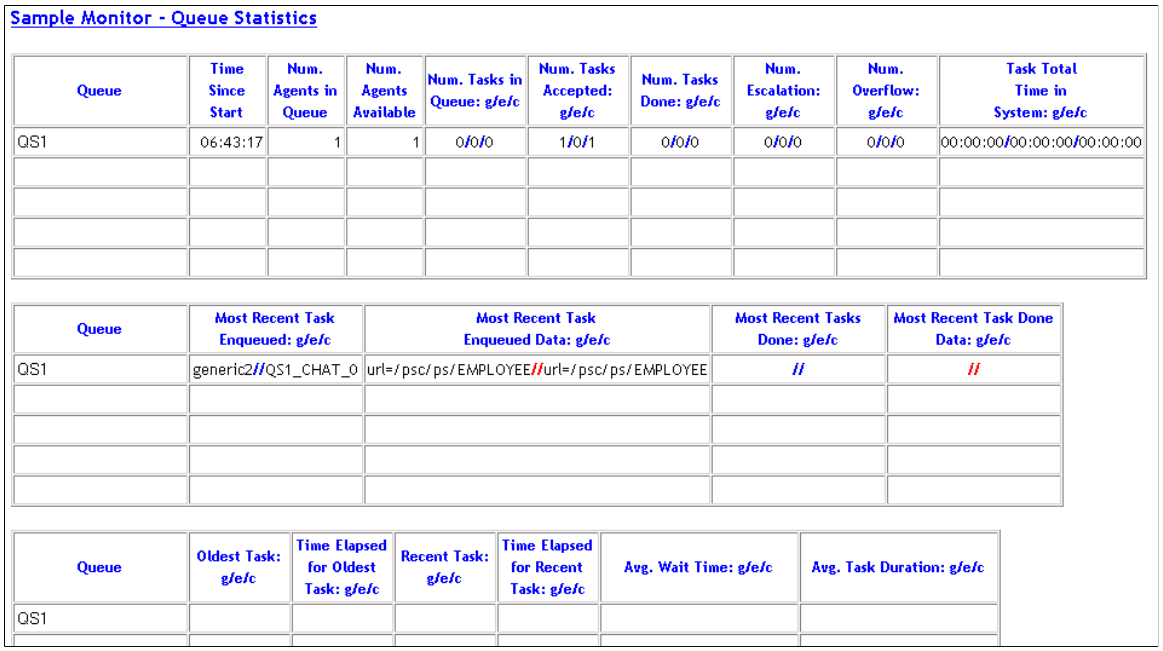
After you click Monitor Queue Statistics, a new browser window appears.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: To enable the new browser window to appear, disable any pop-up blocking software for your browser.

Image: Sample Monitor - Queue Statistics page

This example illustrates the fields and controls on the Sample Monitor - Queue Statistics page. You can find definitions for the fields and controls later on this page.



Sample Monitor - Queue Statistics

Queue	Time Since Start	Num. Agents in Queue	Num. Agents Available	Num. Tasks in Queue: g/e/c	Num. Tasks Accepted: g/e/c	Num. Tasks Done: g/e/c	Num. Escalation: g/e/c	Num. Overflow: g/e/c	Task Total Time in System: g/e/c
QS1	06:43:17	1	1	0/0/0	1/0/1	0/0/0	0/0/0	0/0/0	00:00:00/00:00:00/00:00:00

Queue	Most Recent Task Enqueued: g/e/c	Most Recent Task Enqueued Data: g/e/c	Most Recent Tasks Done: g/e/c	Most Recent Task Done Data: g/e/c
QS1	generic2//QS1_CHAT_0	url=/psc/ps/EMPLOYEE//url=/psc/ps/EMPLOYEE	//	//

Queue	Oldest Task: g/e/c	Time Elapsed for Oldest Task: g/e/c	Recent Task: g/e/c	Time Elapsed for Recent Task: g/e/c	Avg. Wait Time: g/e/c	Avg. Task Duration: g/e/c
QS1						

Queue Statistics 1

The first queue statistics table displays data that does not change frequently. The table displays information that is published by the JSMCAPI group.onStat1 event.

Queue

Displays an identifier for this queue.

The value returned is `Queue_Id`.

Time Since Start

The duration since this queue was started (application server domain start).

The value returned is `timeSinceStart`.

Num. Agents in Queue (number of agents in queue)

The number of agents that are currently logged in to this queue.

The value returned is `numAgentsLoggedIn`.

Num. Agents Available (number of agents available)

The number of agents that are currently logged in to this queue with an available status.

The value returned is `numAgentsAvail`.

Num. Tasks in Queue: g/e/c (number of tasks in queue)

The number of tasks that are currently on this queue, by task type.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

	The value returned is `numTasksQueued`.
Num. Tasks Accepted: g/e/c (number of tasks accepted)	The number of accepted tasks that are currently on this queue, by task type. The value returned is `numTaskAccepted`.
Num. Tasks Done: g/e/c (number of tasks done)	The number of tasks on this queue that have been completed, by task type. The value returned is `numTaskDone`.
Num. Escalation: g/e/c (number of escalation)	The number of escalated tasks on this queue, by task type. The value returned is `numEscalation`.
Num. Overflow: g/e/c (number of overflow)	The number of overflow tasks on this queue, by task type. The value returned is `numOverflow`.
Task Total Time in System: g/e/c	Displays the total duration of the most recently completed task in the system. The total time is the difference between the time that the task was enqueued and the time at which it was marked done. The value returned is `taskTotalTimeInSystem`.

Queue Statistics 2

The second queue statistics table displays data that changes frequently. The table displays information that is published by the JSMCAPI group.onStat1 event.

Most Recent Task Enqueued: g/e/c	Identifies the task that was most recently enqueued on this queue, by task type. The value returned is `mostRecentTaskEnqueued`.
Most Recent Task Enqueued Data: g/e/c	Displays data for the most recently enqueued task on this queue, by task type. The value returned is `mostRecentTasksEnqueuedData`.
Most Recent Tasks Done: g/e/c	Identifies the task that was most recently completed, by task type. The value returned is `mostRecentTaskDone`.
Most Recent Task Done Data: g/e/c	Displays data for the most recently completed tasks, by task type. The value returned is `mostRecentTaskDoneData`.

Queue Statistics 3

The table displays information that is published by the JSMCAPI group.onStat2 event.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Oldest Task: g/e/c	Identifies the oldest enqueued and accepted task on this queue, by task type. The value returned is `oldestTask`.
Time Elapsed for Oldest Task: g/e/c	Displays the duration that the oldest task on this queue has been enqueued, by task type. The value returned is `timeElapsedOldestTask`.
Recent Task: g/e/c	Identifies the most recent task that was enqueued and accepted on this queue, by task type. The value returned is `recentTask`.
Time Elapsed for Recent Task: g/e/c	Displays the duration that the most recent task on this queue has been enqueued, by task type. The value returned is `timeElapsedRecentTask`.
Avg. Wait Time: g/e/c (average wait time)	Displays the average time between a task being enqueued and being accepted, by task type. The value returned is `averageWaitTime`.
Avg. Task Duration: g/e/c (average task duration)	Displays the average duration before a task is completed. The value returned is `averageTaskDuration`. The average task duration is calculated by referencing a list of completed tasks. The size of that list is determined by the donelistsize parameter that is set on the Cluster Tuning page. See Tuning Cluster Parameters.

Using PeopleCode Built-in Functions with PeopleSoft MultiChannel Framework

PeopleSoft MultiChannel Framework uses several PeopleCode built-in functions. Application developers use these functions to communicate task requests and parameters to the queue server. For example, use InitChat code behind a button on an application page to initiate a chat session with an agent.

The built-in functions are:

- DeQueue

  Use DeQueue to notify the queue server that an enqueued task has been completed and to remove the task from the queue.

- EnQueue

  Use EnQueue to add a task to an active physical queue belonging to the specified logical queue.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Forward

  Use Forward to transfer a task from one agent to another agent or another logical queue.

- InitChat

  Use InitChat to place a customer chat request on a queue.

- NotifyQ

  Use NotifyQ to notify the queue server of a task enqueued by EnQueue.

See PeopleCode Language Reference.

Using Universal Queue Classes

The universal queue classes provide the means by which applications can inspect objects that are processed by the queue server, such as tasks, agents, and queues. The classes also enable tasks to reenter the queue with their original task ID, as well as keep task times based on their original entry into the system.

See "Understanding Universal Queue Classes" (PeopleTools 8.54: PeopleCode API Reference).

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 12

Configuring PeopleSoft MCF for Third-Party Routing Systems

Understanding Third-Party Routing Systems

In previous releases, PeopleSoft CTI required a third-party middleware in the form of the Genesys CTI Framework, a Cisco ICM system, or CTI middleware that implements PSMCAPI.

Beginning with PeopleTools 8.48, PeopleSoft MultiChannel Framework (MCF) uses third-party multichannel routing systems to offer a wide range of communication channels to empower PeopleSoft applications like CRM. In the third-party system, the queue is the logical storage unit for representing a work item, and no difference exists between a logical and physical queue. The third party develops the universal routing system to route email, voice, chat, and generic tasks.

PeopleSoft customers may choose either the pre-8.48 features of queue server and CTI or the new third-party routing system to use email, chat, voice, and generic channels.

PSMCAPI provides an interface to the third-party routing system that enables communication between various PeopleSoft components and the third party. An event, like an incoming task from the third-party routing system, is received by PSMCAPI and pushed to JSMCAPI (MCF console) using the REN server. Similarly, a request from MCF console or JSMCAPI is sent to PSMCAPI using the REN server and

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

is eventually passed on to the third-party routing system. The REN server routes requests and events internally to the PeopleSoft system, including an agent desktop or browser.

Image: PSMCAPI interface with PeopleSoft MCF

The following diagram illustrates how PSMCAPI acts as an interface to the third-party routing system and communicates with JSMCAPI through the REN server.

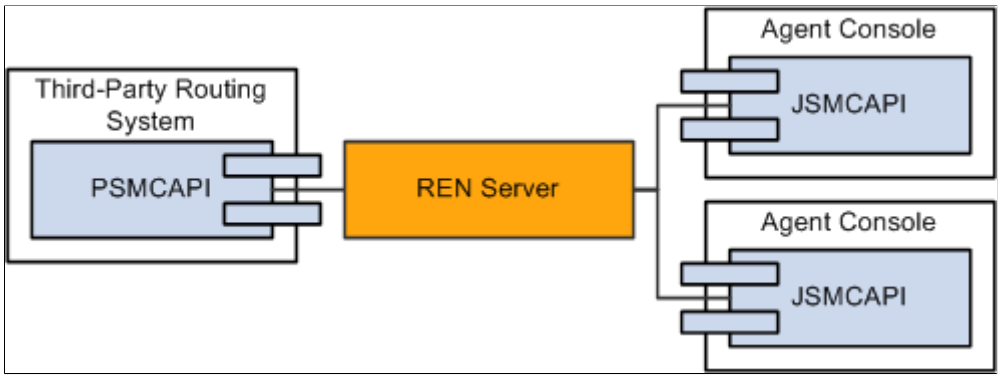
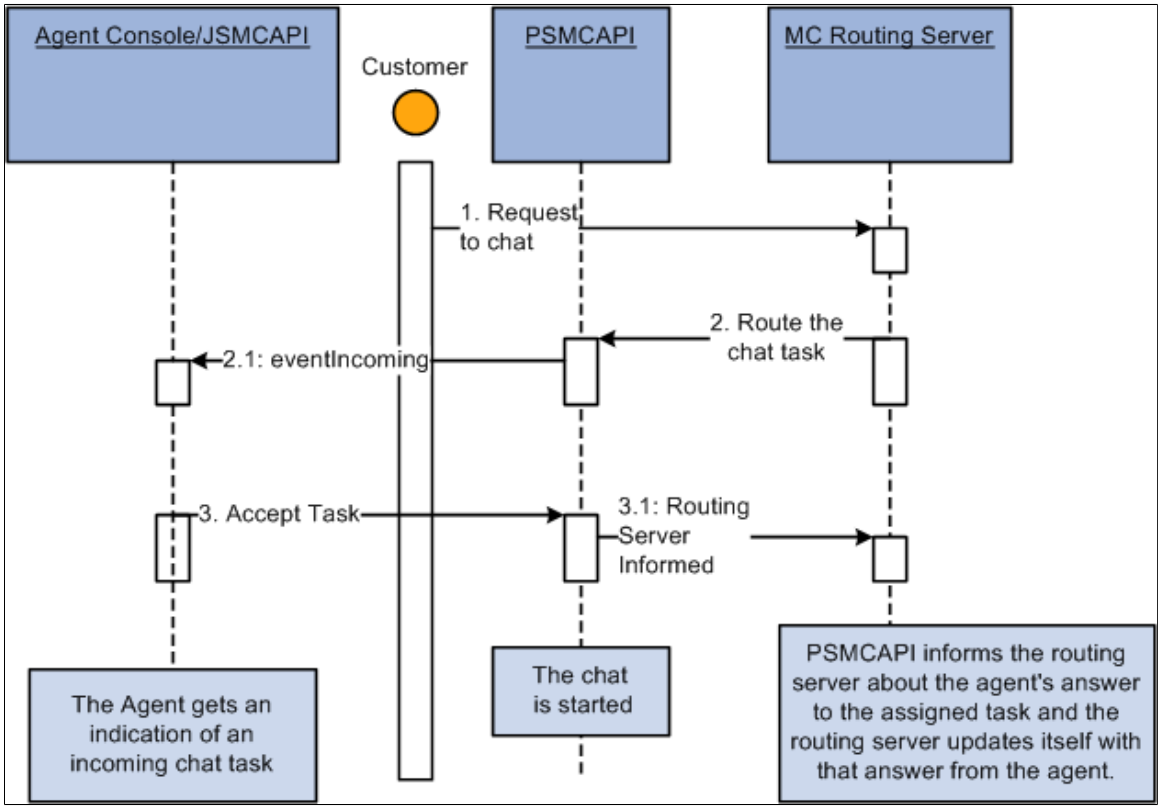


Image: Interaction of PSMCAPI with third-party and JSMCAPI

The following diagram illustrates the interaction of PSMCAPI with third party and JSMCAPI.

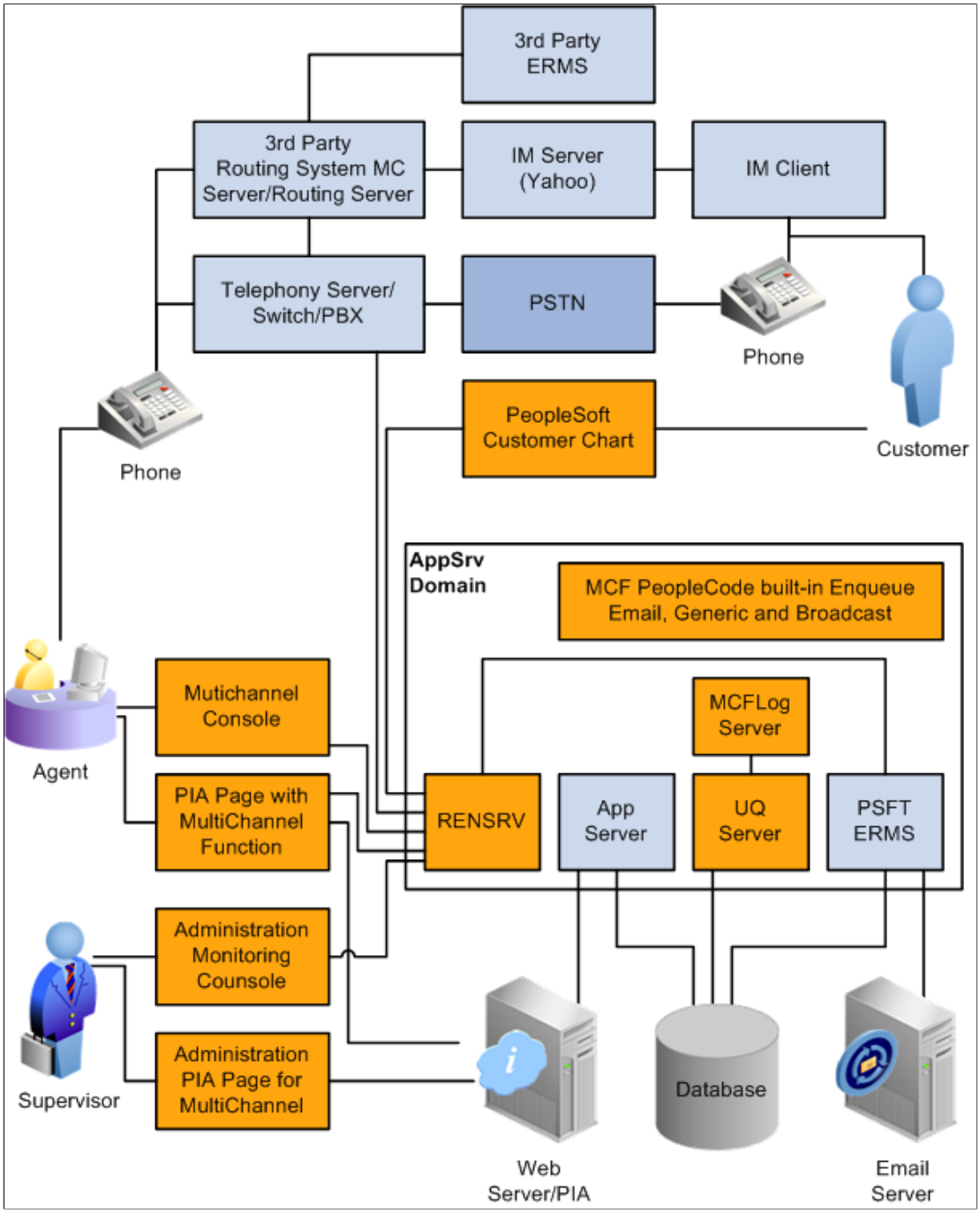


 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PSMCAPI and JSMCAPI support multiple channels of communication, queue, and agent statistics information that enable the leverage of third-party routing systems.

Image: Third-party routing system with PeopleSoft MCF

The following diagram illustrates how the third-party routing system interacts with PeopleSoft MCF.



The third-party routing system connects to all kinds of media servers, and routes the task to the agent based on routing rules. The third-party routing system connects to the REN server through HTTP or HTTPS using PSMCAPI. All requests sent by the agents to third-party system such as login, logout, accept incoming call, and forward email use the above connection. All media requests sent by PeopleCode built-in functions to the REN server such as InitChat or Enqueue also use the above

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

connection. All events from the routing system such as assignment, broadcast, and statistics, go to the agent or the supervisor through this connection.

In the third-party routing system, the queue server acts as an overflow or escalation adapter that writes the overflow or escalated tasks from the third-party routing system to the PeopleSoft Database. When using a third-party routing system for escalated and overflowed task, the queue server is used for task insertions to the database. The third-party routing system processes and maintains task properties and timers. On a task time-out, the routing system sends an event to the queue server to insert the task in the database. The routing system creates the events to remove the task from the agents and recalculates their workload.

This section discusses how to:

- Define third-party routing system requirements.
- Define third-party routing rules.

Defining Third-Party Routing System Requirements

In general, the third-party router:

- Allows business users to define routing strategies.
- Defines escalation and timeout routing paths.
- Allows agents to receive any type of work from multiple channels and queues simultaneously or in parallel.
- Allows pushed and pulled events simultaneously or in parallel.
- Defines intra- and extra-channel escalations.

Defining Third-Party Routing Rules

Third-party routing rules are based on:

- Task type, for example, email, chat, voice, scheduled callback, fax, and so on.
- Time against SLA (Service Level Agreement) in form of overflow and escalation timers.
- Priority of the event expressed as a positive single-digit number.
- Skill within organization or task type.
- Primary language required to resolve any issue.
- Cost of each event type assigned by the administrator to avoid overloading.

The agents or supervisors must have the capabilities to:

- Change the skills, language, priority, or cost of an item to affect its subsequent routing.
- Assign an item to a specific queue or agent.
- Accept, reject, forward, mark as resolved, or mark as awaiting additional information.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Configuring PeopleSoft MCF for a Third Party

This section lists the basic steps to configure PeopleSoft MCF for a third party. Subsequent sections of this PeopleBook describe each step in detail.

You complete these steps to configure PeopleSoft MCF for a third party:

1. Define the third-party flag.

   See Defining the Third-Party Flag.

2. Define PeopleSoft MCF clusters for a third party.

   See Defining the PeopleSoft MCF Cluster Page for a Third Party.

3. Tune PeopleSoft MCF clusters for a third party.

   See Tuning PeopleSoft MCF Cluster Parameters for a Third Party.

4. Notify PeopleSoft MCF of changed parameters for a third party.

   See Notifying PeopleSoft MCF Clusters of Changed Parameters for a Third Party.

5. Define PeopleSoft MCF queues for a third party.

   See Defining PeopleSoft MCF Queues for a Third Party.

6. Define PeopleSoft MCF agents for a third party.

   See Defining PeopleSoft MCF Agents for a Third Party.

7. Configure PeopleSoft MCF tasks for a third party.

   See Configuring PeopleSoft MCF Tasks for a Third Party.

8. Configure CTI.

   See Configuring CTI for a Third Party.

9. View event logs.

   See Viewing Event Logs for a Third Party.

10. View broadcast logs.

    See Viewing Broadcast Logs for a Third Party.

11. Work with third-party sample pages.

    See Working with Third-Party Sample Pages.

Defining the Third-Party Flag

This section discusses how to define the third-party flag.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

To define the third-party flag, use the Third-party Flag (MCF_TP_FLAG_CMP) component.

Defining the Third-Party Flag

Access the Third Party Flag page using the following navigation path:

PeopleTools, MultiChannel Framework, Third-Party Configuration, Third-Party Flag

Image: Third-Party Flag page

This example illustrates the fields and controls on the Third-Party Flag page.

The default value of the third-party flag is *False.* If the flag is set to *False,* the queue server routes chat, generic, and email and uses PeopleSoft CTI to route voice. If the flag is set to *True*, all events, chat, generic, email, and voice are routed by the third-party routing system, and you can configure MCF using third-party pages only. The queue server:

- Stops listening to all routing requests.
- Starts up new listeners for overflow and escalation.

Separate database tables are created for queue server and third-party routing server that combine all fields from MCF and CTI. When the third-party flag is set to *True*, all pages using queue server are disabled and grayed out. When the third-party flag is set to *False*, all pages using third-party routing system are disabled and grayed out.

Note: Whenever you change the value of this flag, you must reboot the queue server and MCF Log server.

Defining the PeopleSoft MCF Cluster Page for a Third Party

This section discusses how to define the PeopleSoft MCF Cluster page for the third party. To view the Cluster page, use the Cluster (MCF_TP_RENCFG_CMP) component.

Defining the PeopleSoft MCF Cluster Page for a Third Party

The Cluster page displays the associated PeopleSoft MCF cluster URLs and log server details for the selected cluster.

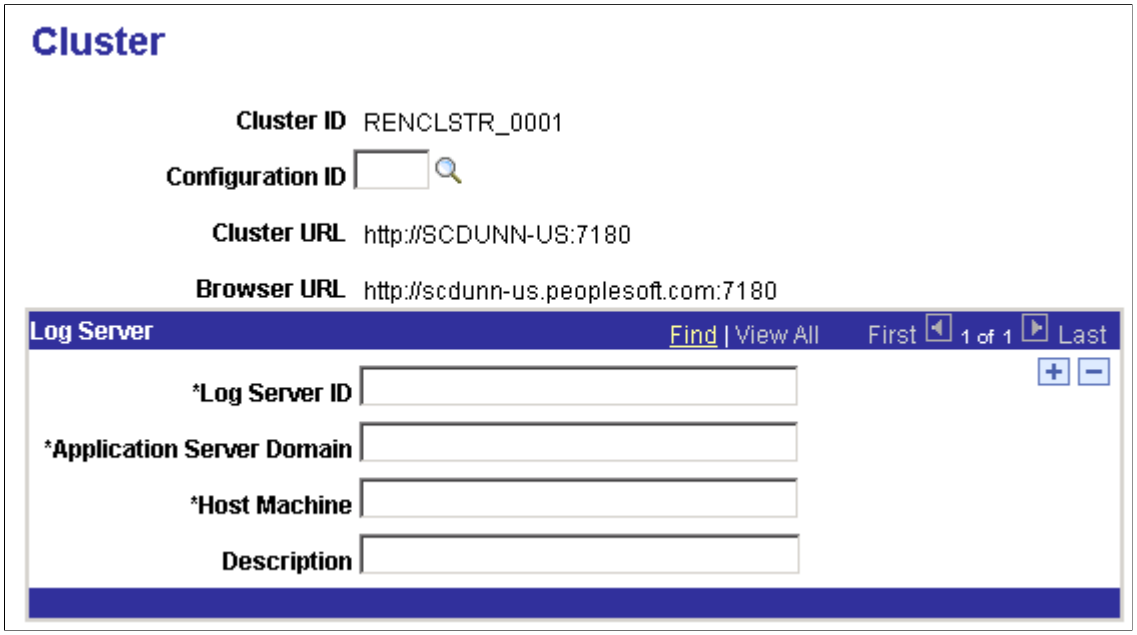
Access the Cluster page using the following navigation path:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Third-Party Configuration, Cluster

Image: Cluster page

This example illustrates the fields and controls on the Cluster page. You can find definitions for the fields and controls later on this page.



Cluster ID

Displays the cluster ID of the cluster containing the queue server and log server.

Configuration ID

Displays the configuration ID.

Note: This value is required only for CTI configuration.
To select the configuration ID from lookup, define the configuration ID when you are configuring CTI.

See Configuring CTI Console Type.

Cluster URL

Displays the URL for the REN server that serves this cluster.

Note: The Cluster URL is defined on the REN Server Definitions page.

See Defining REN Servers.

Browser URL

Displays the browser URL.

Note: The browser URL may be different from the cluster URL, which should not have to go through any firewall, reverse proxy server, or other outward-facing security barrier.
The browser URL is same as defined on the REN Server Definitions page.

See Defining REN Servers.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Log Server

A cluster consists of a primary log server and any number of backup servers. Each cluster requires a minimum of one log server and one queue server. The primary log server is the first log server started and the remaining log servers are backups. If the primary log server fails, the system determines the subsequent primary log server among the backups.

Note: You can save a cluster without creating a log server or a queue server. But, for MCF to run, you need at least one log server and one queue server.

You can add a log server to a cluster by adding a new row. Before removing a log server, ensure that it is not the master, then shut down its domain. Then click Delete (the minus sign).

Log Server ID	Enter a unique identifier for each log server to identify its entries in the log cluster table. The queue server process paired with this log server uses this same ID to identify its entry in the database table.
Application Server Domain	Enter the application server domain of which this log server is a member.
Host Machine	Enter the name of the application server host machine.
Description	Enter the description of the host machine.

Tuning PeopleSoft MCF Cluster Parameters for a Third Party

This section discusses how to tune PeopleSoft MCF cluster parameters for a third party. Tuning cluster parameters may give you better performance. To tune cluster parameters, use the Cluster Tuning (MCF_TP_SYS_NV_CMP) component.

Tuning PeopleSoft MCF Cluster Parameters for a Third Party

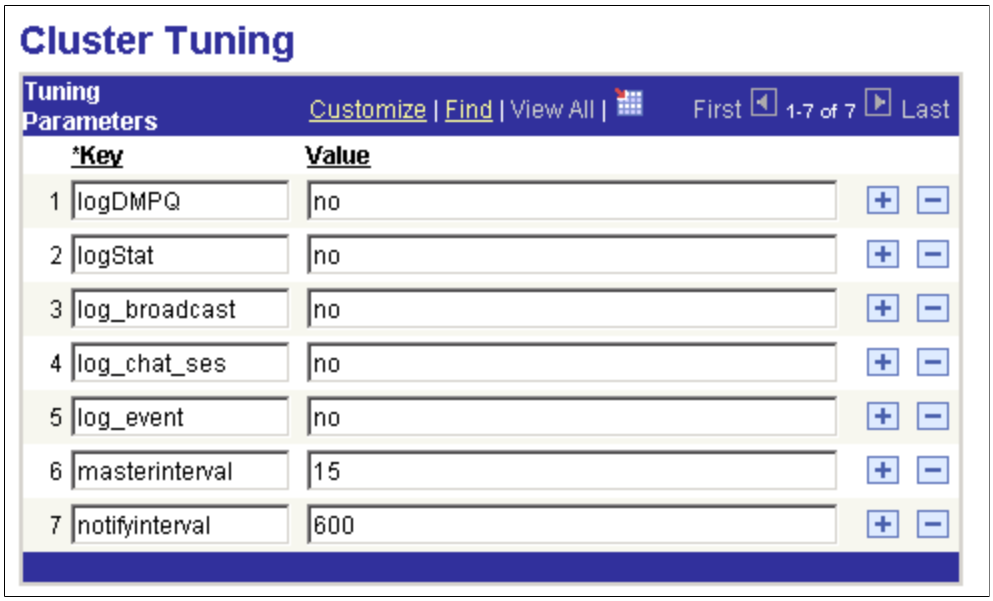
Access the Cluster Tuning page using the following navigation path:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Third-Party Configuration, Cluster Tuning

Image: Cluster Tuning page

This example illustrates the cluster tuning parameters on the Cluster Tuning page.



Use the Cluster Tuning page to set MCF cluster parameters to optimize performance or enable logging for a cluster.

If you make changes to a cluster parameter, you must use the third-party Notify Cluster page to propagate the changes.

See Notifying Third-Party Clusters of Changed Parameters.

The following table lists the cluster tuning parameters you can modify and describes the default values and usage of each:

Key	***Default value***	***Usage***
logDMPQ	*No*	Select *Yes* if you want PSMCFLOG to log REN server event notifications resulting from bcastinterval broadcasts. Only the event is logged, not its contents. This is a logging parameter. If you change the value of this parameter you must use the Refresh Logging Parameters button on the third-party Cluster Notify page to notify clusters of the changed parameter See Notifying Third-Party Clusters of Changed Parameters, Viewing Event Logs.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Key	***Default value***	***Usage***
logStat	*No*	Select *Yes* to log the statistics returned by the queue server for the onStat1 user and group events to the database. This is a logging parameter. If you change the value of this parameter you must use the Refresh Logging Parameters button on the third-party Cluster Notify page to notify clusters of the changed parameter See Notifying Third-Party Clusters of Changed Parameters, Using and Demonstrating JSMCAPI.
log_broadcast	*No*	Select *Yes* to turn on logging of the broadcast messages that are sent. This is a logging parameter. If you change the value of this parameter you must use the Refresh Logging Parameters button on the third-party Cluster Notify page to notify clusters of the changed parameter See Notifying Third-Party Clusters of Changed Parameters, Viewing Broadcast Logs.
log_chat_ses	*No*	Select *Yes* to turn on logging of the contents of chat sessions. This is a logging parameter. If you change the value of this parameter you must use the Refresh Logging Parameters button on the third-party Cluster Notify page to notify clusters of the changed parameter See Notifying Third-Party Clusters of Changed Parameters, Viewing Chat Logs.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Key	*Default value*	*Usage*
masterinterval	*15*	Interval, in seconds, after which a cluster master updates its timestamp in its cluster tables. Slave clusters check the timestamp to determine that the master cluster is still running. A lower value enables rapid discovery of a failed master server, but increases log server overhead. A higher value reduces log server overhead, but delays discovery of a failed master server. If only one log server is configured for an MCF cluster, this value can be large. The masterinterval value also acts as a heartbeat interval for the master log server connection to user consoles. This is a timing parameter. If you change the value of this parameter you must use the Refresh Timing Parameters button on the third-party Cluster Notify page to notify clusters of the changed parameter. See Notifying Third-Party Clusters of Changed Parameters.
notifyinterval	600	Interval, in seconds, after which the database is checked for any pending enqueued tasks. The third party will be notified if any enqueued task is pending in the database. This is a timing parameter. If you change the value of this parameter you must use the Refresh Timing Parameters button on the third-party Cluster Notify page to notify clusters of the changed parameter. See Notifying Third-Party Clusters of Changed Parameters.

Notifying PeopleSoft MCF Clusters of Changed Parameters for a Third Party

This section describes how to notify PeopleSoft MCF clusters of changed parameters for a third party. To notify clusters of changed parameters, use the Cluster Notify (MCF_TP_NOTIFY_CMP) component.

Notifying Third-Party Clusters of Changed Parameters

Use the Notify Cluster page to notify a cluster of certain changes to its parameters, constituent queues, or that its application servers are being shut down.

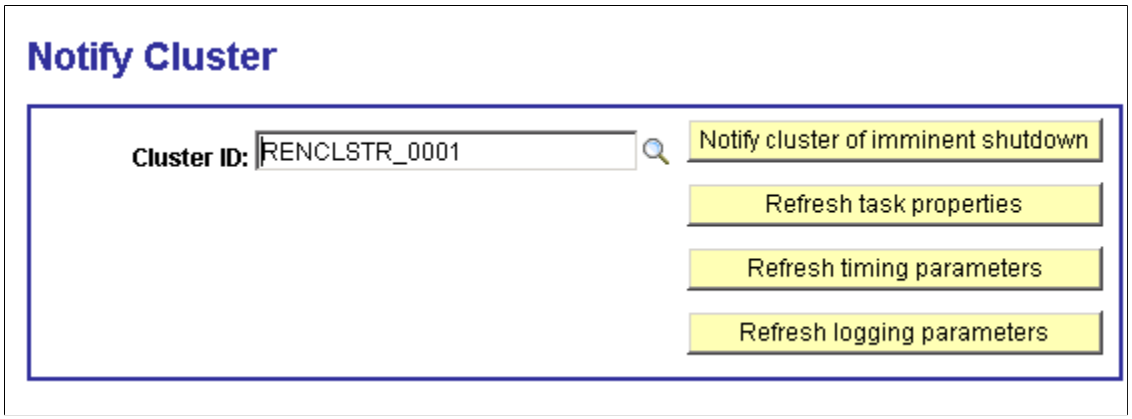
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Access the Notify Cluster page using the following navigation path:

PeopleTools, MultiChannel Framework, Third-Party Configuration, Cluster Notify

Image: Notify Cluster page

This example illustrates the fields and controls on the Notify Cluster page. You can find definitions for the fields and controls later on this page.



Notify cluster of imminent shutdown Click to send a message to every agent logged on to the selected cluster that they have been logged off.

Send this notification if the cluster's application servers are being shut down.

Refresh task properties Click to load task properties that have been changed on the Tasks page for the selected cluster.

Refresh timing parameters Click to reload parameters, other than logging parameters, changed on the Cluster Tuning page for the selected cluster.

Refresh logging parameters Click to reload logging parameters changed on the Cluster Tuning page for the selected cluster.

Defining PeopleSoft MCF Queues for a Third Party

To define PeopleSoft MCF queues for the third-party routing system, use the third-party queue (MCF_TP_Q_CFG_CMP) component. This section discusses how to:

- Define queues.
- Define canned queue messages.
- Define canned queue URLs.

Defining PeopleSoft MCF Queues for a Third Party

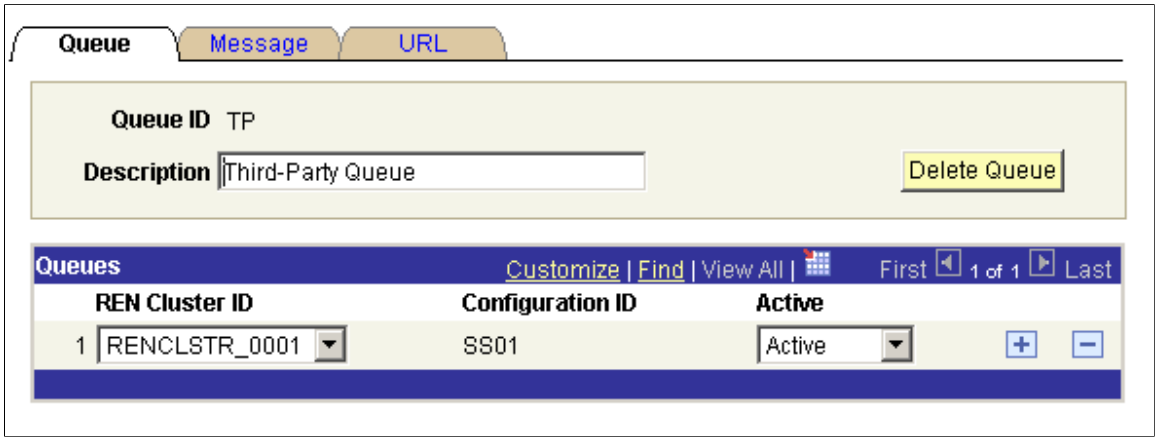
Access the Queue page using the following navigation path:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Third-Party Configuration, Queue

Image: Queue page

This example illustrates the fields and controls on the Queue page. You can find definitions for the fields and controls later on this page.



Queue ID

Queue IDs must be alphanumeric, but they may include underscore characters.

Configuration ID

Displays the configuration ID.

Note: This value is required only for CTI configuration. To select the configuration ID from lookup, define the configuration ID when you are configuring CTI.

See Configuring CTI Console Type.

Delete Queue

Click to remove this queue.

Deleting a queue means that no work or agents can be assigned to the queue and the queue is removed from all agents' available queues.

Cluster ID

The Cluster ID field identifies the cluster that services this queue.

Active

Select *Active* or *Inactive* from the drop-down list box.

The queue server does not send new tasks to an inactive queue. Agents and existing tasks remain on an inactive queue. To receive new queued tasks, the queue must be active.

Active and inactive statuses support “follow-the-sun” practices. For example, SALES1 could be supported in the London office, and SALES2 could be supported in the San Francisco office when the London office is closed by activating and inactivating the appropriate queues.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

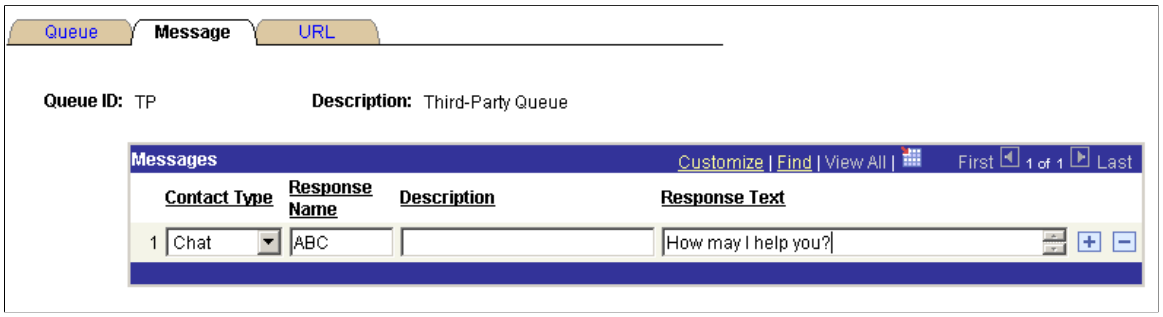
Defining Canned Queue Messages

Access the Message page using the following navigation path:

PeopleTools, MultiChannel Framework, Third-Party Configuration, Queue, Message

Image: Message page

This example illustrates the fields and controls on the Message page. You can find definitions for the fields and controls later on this page.



Contact Type

Select one of the following contact types:

- *Chat*
- *Email*

Response Name

The response name appears in the agent's Template Messages drop-down list box for all agents who belong to this queue.

All messages are downloaded when the agent launches the agent chat console by accepting a customer chat. Template messages are not available in collaborative chat.

Response Text

The specified message appears in the client chat window when selected by the agent.

Defining Canned Queue URLs

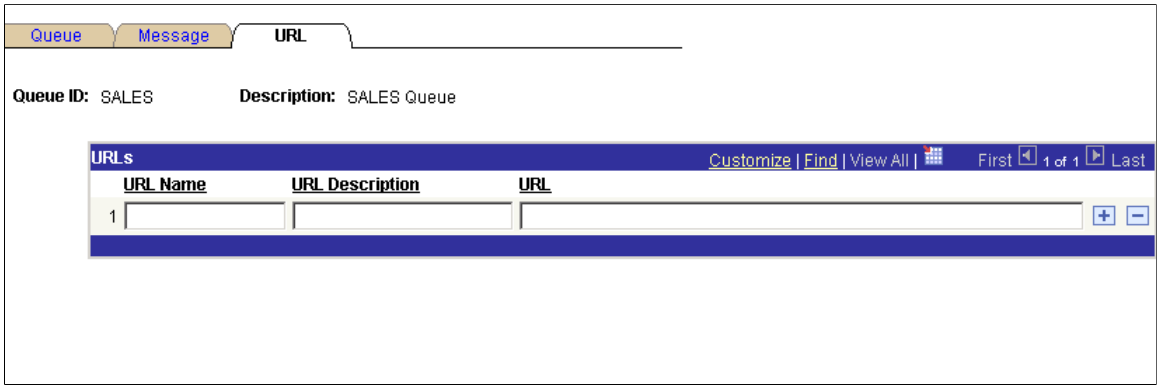
Access the URL page using the following navigation path:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Third-Party Configuration, Queue, URL

Image: URL page

This example illustrates the fields and controls on the URL page. You can find definitions for the fields and controls later on this page.



Canned queue URLs enable the agent to push a predefined URL to the customer, which appears in a pop-up window for the customer. The agent can edit these URLs in the chat window after selecting them.

URL Name

Enter the name of the URL.

The URL name appears in the agent's URL drop-down list box for all agents that belong to this queue.

URL Description

Enter a description of the URL.

This description appears only on this page to further describe this URL or, for example, its reason for inclusion.

URL

Enter the URL. The URL must include the protocol (http://) and any required parameters.

All canned queue URLs defined for the queue are downloaded when the agent launches the agent chat console by accepting a customer chat. These URLs are not available in collaborative chat.

If you send a canned queue URL that is a PeopleSoft Pure Internet Architecture URL, be sure that the recipient has permissions to access that portal, node, or page.

Defining PeopleSoft MCF Agents for a Third Party

To define PeopleSoft MCF agents for a third party, use the (MCF_TP_AGENT_CMP) component. This section discusses how to:

- Create agents.
- Set up buddy lists.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Customize windows.
- Define messages.
- Specify agent-specific URLs.
- Define agent's presence.
- Specify media.
- Specify languages that an agent supports.
- Specify miscellaneous parameters.

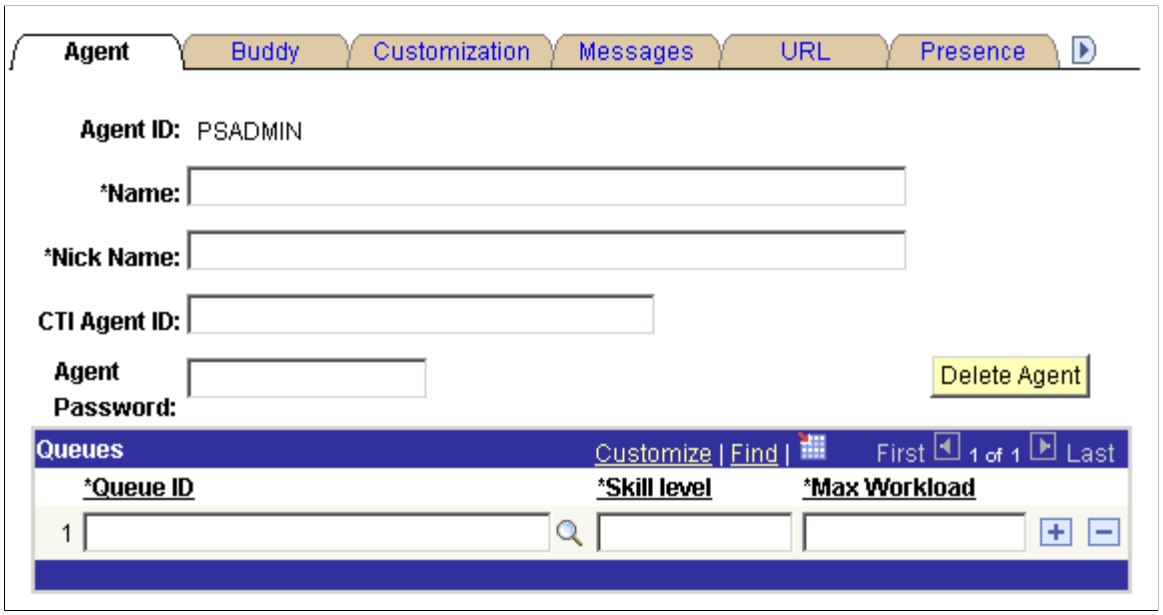
Creating PeopleSoft MCF Agents for a Third Party

Access the Agent page using the following navigation path:

PeopleTools, MultiChannel Framework, Third Party Configuration, Agent

Image: Agent page

This example illustrates the fields and controls on the Agent page. You can find definitions for the fields and controls later on this page.



Agent ID	Specifies the agent ID.
Name	Enter the full name of this agent in (lastname,firstname) format. The agent name appears in other agents' buddy lists.
Nick Name	Enter a short name for this agent. The agent nickname identifies this agent in chat sessions and logs.
CTI Agent ID	Enter a CTI agent ID for a CTI agent.
Agent Password	Enter a password for the agent.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Queue ID

Enter the ID of a queue to which this agent is assigned. Each agent may be assigned to more than one queue. An agent can log on to only one queue at a time from the MultiChannel Console.

Skill

Select the skill level of this agent for the tasks assigned for this queue. This field is required.

This option is used by third parties for task assignment. Third parties define skill levels based on business needs, requirements, policy, and so on.

Maximum Workload

Select the maximum load that this agent can be assigned before tasks are held or assigned to other agents. This field is required.

This option is used by third parties for task assignment. Third parties define maximum workload based on business needs, requirements, policy, and so on.

Note: Do not delete a queue from an agent's list unless that agent has no open accepted tasks on that queue.

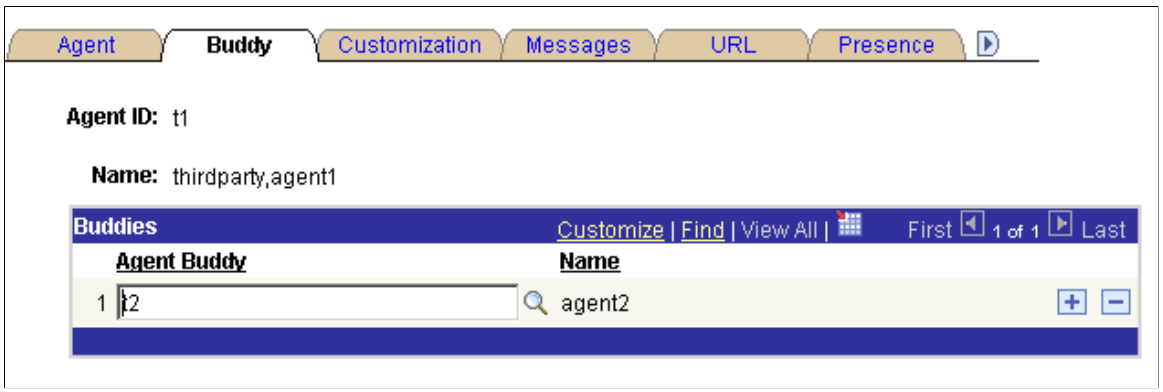
Setting Up Buddy Lists

Access the Buddy page using the following navigation path:

PeopleTools, MultiChannel Framework, Third Party Configuration, Agent, Buddy

Image: Buddy page

This example illustrates the fields and controls on the Buddy page. You can find definitions for the fields and controls later on this page.



The agent's buddy list facilitates collaborative chat and chat conferencing.

Agent Buddy

Select another agent with whom this agent can have a chat session or can ask to conference onto another chat.

Name

Displays the buddy agent's nickname. The name appears in the buddy list on the MultiChannel Console or agent chat window.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

An agent's presence as shown in the buddy list on the MultiChannel Console or on the Invite Agent list on the chat console indicates the agent's availability for chat or conference.

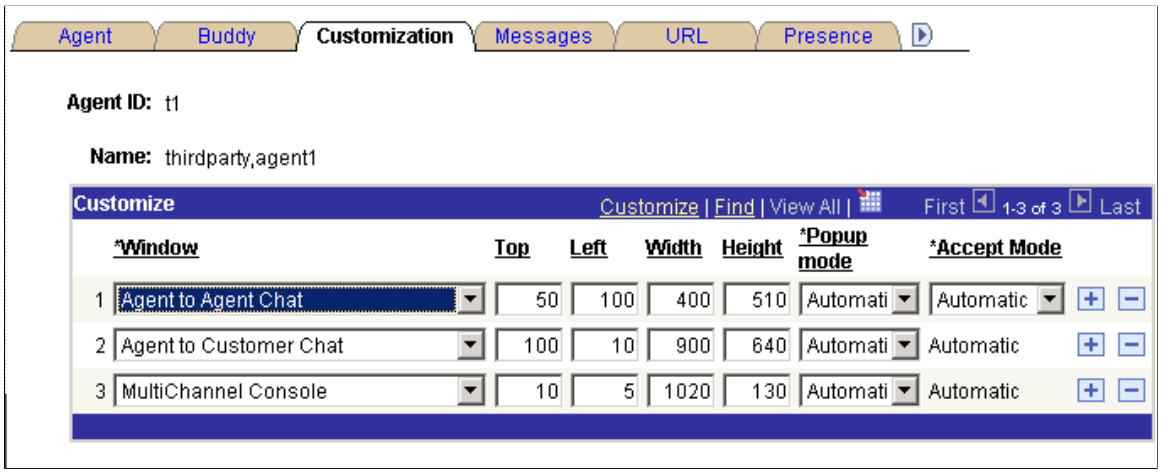
Customizing Windows

Access the Customization page using the following navigation path:

PeopleTools, MultiChannel Framework, Third Party Configuration, Agent, Customization

Image: Customization page

This example illustrates the fields and controls on the Customization page. You can find definitions for the fields and controls later on this page.



Set the initial agent window placement and size by specifying parameters on this page. An agent may resize and move the windows.

Window

Select the window to which the specified configuration applies.

Select from:

- *Agent to Agent Chat*
- *Agent to Customer Chat*
- *E-mail*
- *Generic Alert*
- *Grab URL*
- *MultiChannel Console*

Top and Left

Enter the distance in pixels from the top and left edge of the screen when the window first appears.

Width and Height

Enter the width and height, in pixels, of the window when it first appears.

Popup Mode

Select from:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Automatic: The window appears automatically. For customer-initiated chat or tasks initiated from the EnQueue built-in function, the task is automatically accepted as well. For agent-initiated chat, the agent can elect not to accept the task; in effect, the agent can preview the task. If this is the desired behavior, select *Manual* from the Accept Mode drop-down list box. If you want agent-to-agent tasks to function like customer-initiated tasks, select *Automatic* from the Accept Mode drop-down list box.

Manual: The window does not appear until the agent clicks the task on the agent MultiChannel Console. For customer-initiated chat or tasks initiated from the EnQueue() built-in function, clicking the task means that the task is automatically accepted as well. For agent-initiated chat, the behavior depends on the setting for the Accept Mode field.

Accept Mode

Select from:

Automatic: Agent-to-agent chats are automatically accepted without requiring the agent to click the icon.

Manual: Agent-to-agent chats require the agent to click the icon.

Accept mode affects only collaborative chat.

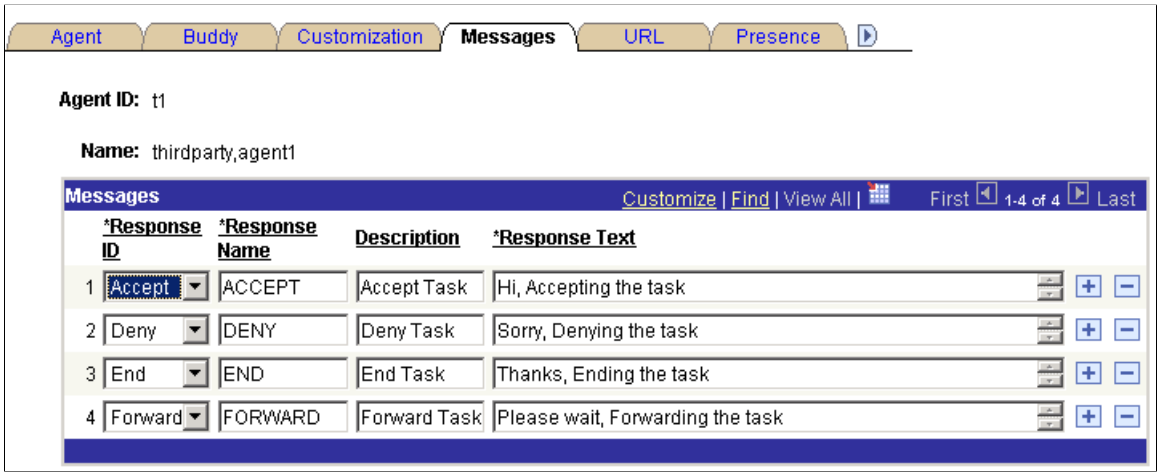
Defining Messages

Access the Messages page using the following navigation path:

PeopleTools, MultiChannel Framework, Third Party Configuration, Agent, Messages

Image: Messages page

This example illustrates the fields and controls on the Messages page. You can find definitions for the fields and controls later on this page.



An agent can create personalized responses in addition to the system responses defined for each queue.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Response ID

Responses, except those identified as Other, are linked to specific events. These responses, except for those identified Other, are always sent on these events from this agent. If an agent does not have a customized response for a specific event, it is read from a default response set in the Message Catalog. The response text set here overrides the default text set in the Message Catalog.

Select from:

- *Abandon:* A chat is abandoned when a chat initiator closes the chat window before the chat is accepted by an agent. This response appears when the agent accepts the abandoned chat.
- *Accept:* This response is automatically sent to the chat initiator when an agent accepts a chat in response to a chat request that does not include a question. If the chat request includes a question, the agent's Answer Question text is sent in response instead of the Accept response.
- *Answer Question:* This response is automatically sent to the chat initiator when an agent accepts a chat in response to a chat request that includes a question.
- *Deny:* Only applies to collaborative chat. If an agent elects not to accept a chat, this response is automatically sent to the chat initiator.
- *End:* If either party exits a chat after the chat is accepted, this response is displayed from the agent.
- *Forward:* If the agent forwards a chat session to another queue, this response is sent to the customer.
- *Other:* These responses are never automatically sent in a chat session. Their message names appear in the Template Messages drop-down list box on the agent chat page. These messages are appended to the template messages (chat responses) defined for the queue.

Response Name

This name appears in the agent's template response drop-down list box.

Response Text

Enter the response text to appear in the chat window.

Specifying Agent-Specific URLs

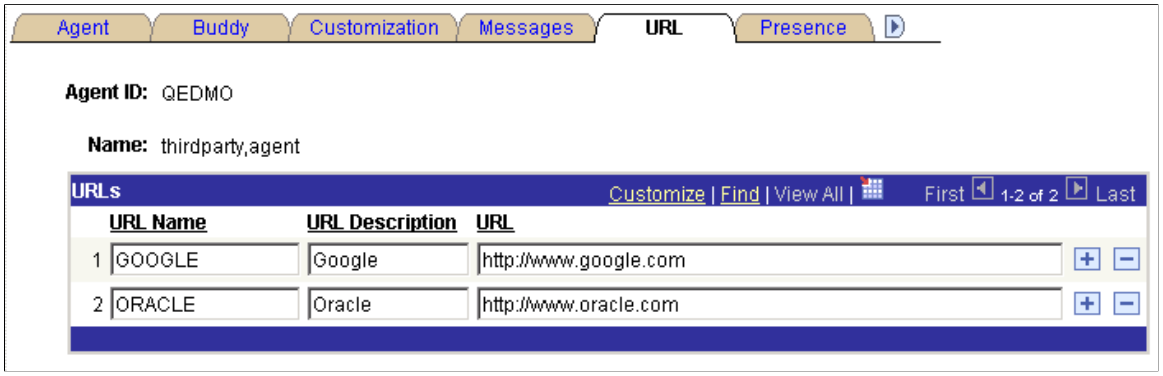
This page defines URLs that this agent can send to a client browser. Access the URL page using the following navigation path:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Third Party Configuration, Agent, URL

Image: URL page

This example illustrates the fields and controls on the URL page. You can find definitions for the fields and controls later on this page.



URL Name The URL name appears in the agent's URL drop-down list box.

URL Description This description appears only on this page, to further describe this URL or, for example, its reason for inclusion.

URL Enter the URL. The URL must include the protocol (http://) and any required parameters.

All URLs defined for the agent are downloaded when the agent launches the agent chat console by accepting a customer chat. These URLs are not available in collaborative chat.

If you send a PeopleSoft Pure Internet Architecture URL, be sure that the recipient has permissions to access that portal, node, or page.

Defining Agent's Presence

Each agent can customize the presence description displayed when the agent is available or unavailable. The queue server only understands the presence state, available or unavailable, but you can enter more specific presence descriptions when displaying or logging an agent's presence. If you do not enter presence descriptions, default values are used.

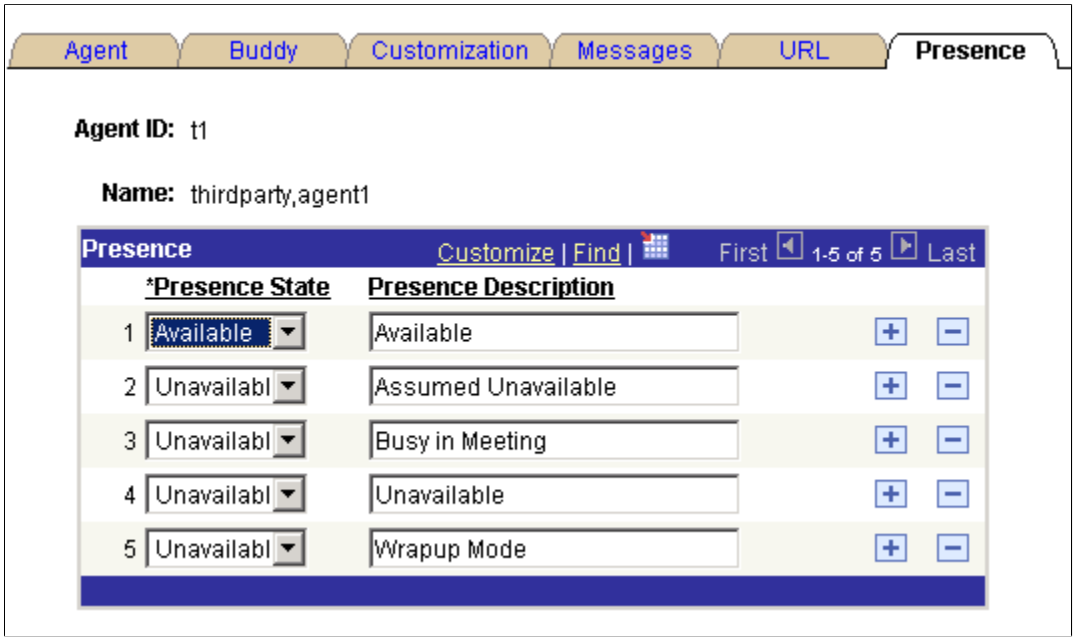
Access the Presence page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Third Party Configuration, Agent, Presence

Image: Presence page

This example illustrates the fields and controls on the Presence page. You can find definitions for the fields and controls later on this page.



Presence State Select *Available* or *Unavailable.*

Presence Description Enter a description for each agent state. The description appears in logs of agent activity and when agent presence is displayed.

Note: An available state has only one presence description, *Available.* For an unavailable state, you can enter any presence description, such as tea break, coffee break, lunch, and so on.

Specifying the Media

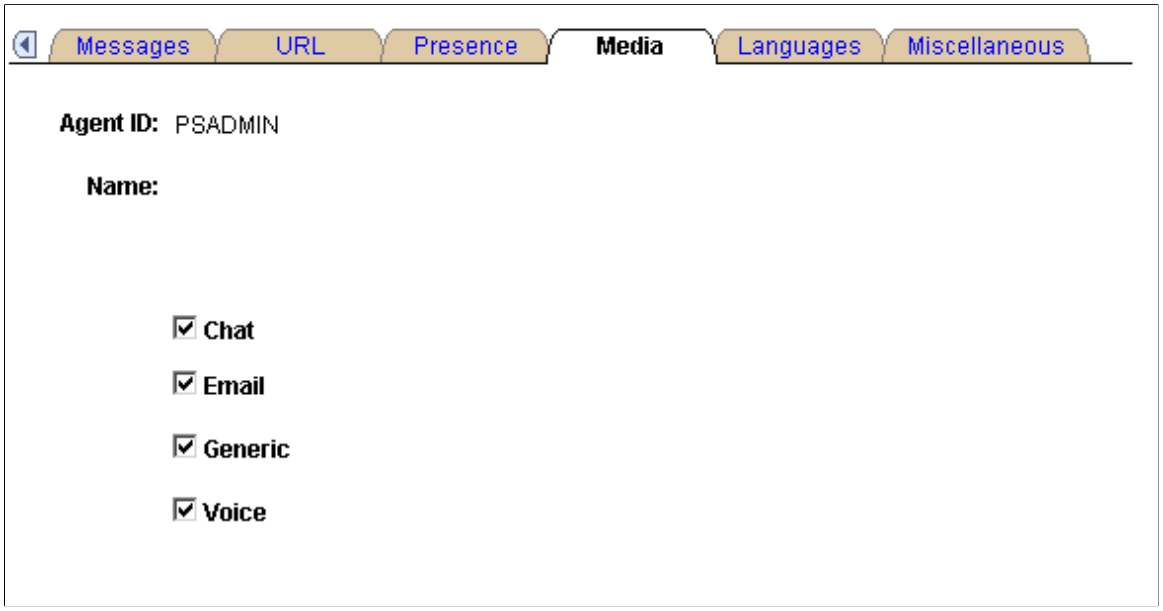
The media page defines the capability of each agent to perform a task. Access the Media page using the following navigation path:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Third Party Configuration, Agent, Media

Image: Media page

This example illustrates the fields and controls on the Media page. You can find definitions for the fields and controls later on this page.



Chat	Select this check box if the agent has a capability to chat.
Email	Select this check box if the agent has a capability to email.
Generic	Select this check box if the agent has a capability to perform a generic task.
Voice	Select this check box if the agent has a capability to perform a voice task.

Specifying Languages That an Agent Supports

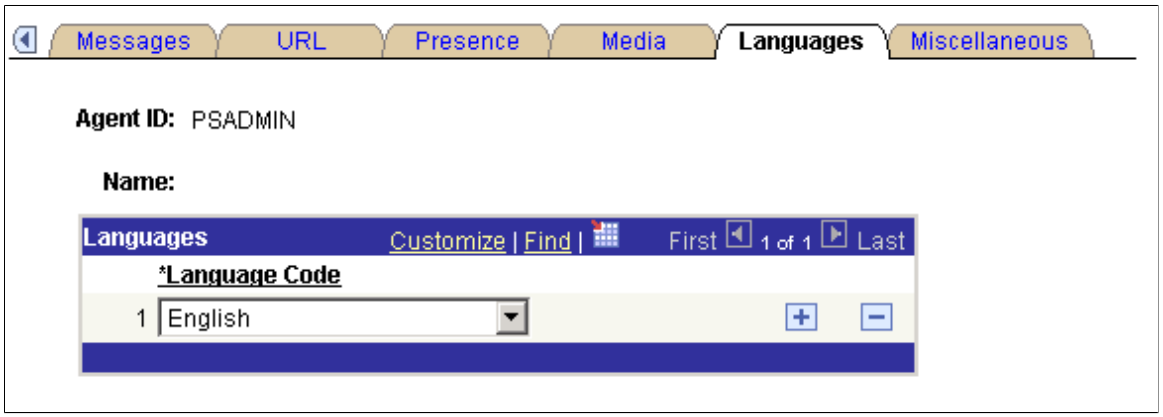
Access the Languages page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Third Party Configuration, Agent, Languages

Image: Languages page

This example illustrates the fields and controls on the Languages page.



Enter the languages that each agent is qualified to support.

The agent is assigned only tasks that have been enqueued with a language code in the agent's language list. For the EnQueue built-in function, the language code is specified as a parameter. For the InitChat built-in function, the language code is determined by the user profile of the initiator.

If you do not enter a language code for a new agent, the default value is English.

Specifying Miscellaneous Parameters

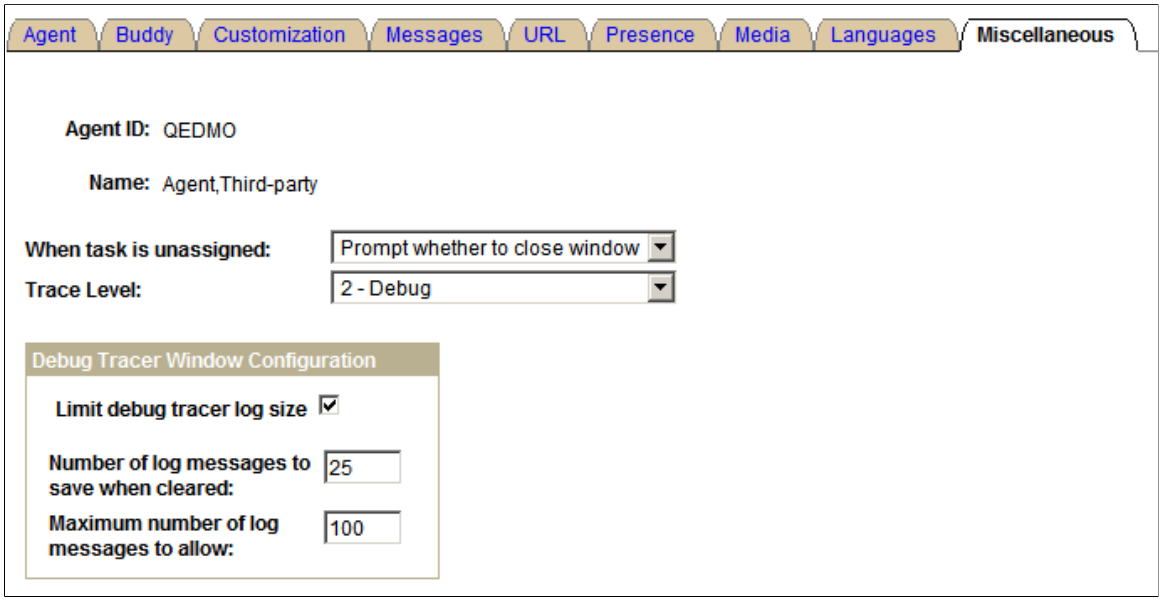
Access the Miscellaneous page using the following navigation path:

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Third Party Configuration, Agent, Miscellaneous

Image: Miscellaneous page

This example illustrates the fields and controls on the Miscellaneous page. You can find definitions for the fields and controls later on this page.



When task is unassigned

Select from the following options the action that occurs when a task assigned to an agent is unassigned:

- *Prompt whether to close window (default).*
- *Close the task window.*
- *Do not close the task window.*

Trace Level

Select from the following log trace levels:

- *0 - None*
- *1 - Information*
- *2 - Debug*

Note: If a value other than *0* is selected, a tracer window appears to display activities and events on the chat or MultiChannel Console for debugging purposes.

Limit debug tracer log size

This check box is enabled when the value entered for Trace Level is not 0. Select this check box to enable the agent to clear the tracer log based on Number of log messages to save when cleared and Maximum number of log messages to allow.

If the check box is cleared, the tracer log will not be cleared and the Number of log messages to save when cleared and

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

	Maximum number of log messages to allow fields will be disabled.
Number of log messages to save when cleared	Specify the minimum number of recent tracer log messages that should be maintained in the tracer window.
Maximum number of log messages to allow	Specify the maximum number of tracer log messages that will be maintained in the tracer window.

Note: Number of log messages to save when cleared and Maximum number of log messages to allow fields are required if the Limit debug tracer log size check box is selected.

Configuring PeopleSoft MCF Tasks for a Third Party

This section discusses how to configure tasks. To configure tasks, use the MCF Task (MCF_TP_TASKCFG_CMP) component.

Configuring PeopleSoft MCF Task for a Third Party

See Configuring Tasks.

Configuring CTI for a Third Party

This section discusses how to configure CTI using third-party routing system. To configure CTI, use the MCF_TP_CTI_CONFIG component.

See Configuring PeopleSoft CTI.

Communicating with Customers and Agents Using Chat

This section discusses how to use the Agent Chat window.

- Use the third-party agent chat window.
- Use the customer chat window.

Using the Third-Party Chat Window

When you accept a chat task, a chat window opens.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The format of the agent chat window and the contents of its right pane are delivered as part of PeopleSoft MultiChannel Framework. The content of the left pane is determined by application developers, and is passed as a parameter to the InitChat() built-in function.

Image: Third-party agent chat window

This example illustrates the fields and controls on the third-party agent chat window. You can find definitions for the fields and controls later on this page.

Conversation History

Lists progress of the chat, line by line.

If chat logging is enabled, the conversation is recorded in the database chat log by the MCF log server. View the chat log on the Chat Log page.

If accessibility features are not turned off in the My Personalizations component, an additional text box appears below the conversation history. The most recent customer input

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

	appears in the secondary text box, which can be read by screen reading software.
Template Messages	Send the customer a standard message by selecting one from the list. The message text appears in the Input Text text box. Click Send to send the message. Edit template messages for each queue on the queue Chat Responses page.
Input Text	To respond to the customer, enter text, and then click Send or press Enter. The maximum size of the text that can be sent at one time is 4096 bytes (4 kilobytes).
Send	Click to send the contents of the Input Text text box.
Exit Dialog	Click to end the chat. The chat window remains open, enabling you to return to the page for follow-up work after the customer exits the dialog.
Static URL and Push	To send a static URL to the customer, select a URL name, then click Push. When a URL is pushed to a customer, a browser window for that URL is launched on the customer's workstation.
URL	Displays static and grabbed URLs.
Select	Click to launch an application page, from which a URL can be returned to populate the URL field. The format of the URL wizard page that appears when you click Grab is defined by application developers. PeopleSoft supplies a sample URL wizard page. See Using the URL Wizard.
Forward to Queue	To forward the current chat to another queue, select the queue.
History	Click to get the history for the chat conversation.
Wrapup	Click to write wrapup comments for the chat. Wrapup comments are stored in PS_MCFCHATLOG.
Forward to Agent	To forward the current chat to another agent, select the queue and click Go.
Invite Agent into Conference	To request that another agent join the chat, select the agent and click Go.

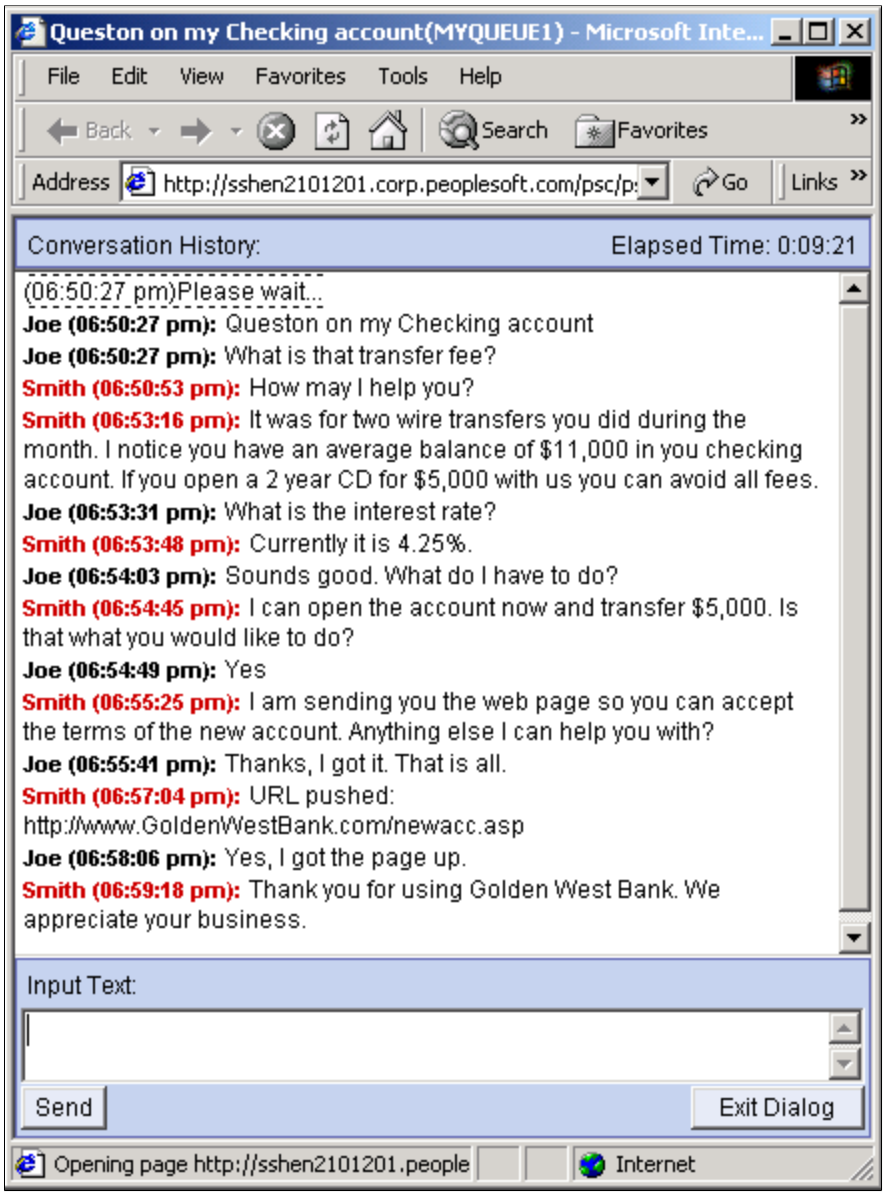
 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Using the Customer Chat Window

When a customer initiates a chat, a chat window appears:

Image: Customer chat window

This example illustrates the fields and controls on a customer chat window. You can find definitions for the fields and controls later on this page.



The format and content of the customer chat window is delivered as part of PeopleSoft MultiChannel Framework. The customer chat window includes a conversation history text box, input text box, and Send and Exit Dialog buttons.

If accessibility features are not turned off in the My Personalizations component, an additional text box appears below the conversation history. The most recent agent input appears in the secondary text box, which can be read by screen reading software.

Input Text	The customer enters text here.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The maximum size of the text that can be sent at one time is 4096 bytes (4 kilobytes).

Send Click to send the input text to the agent.

Exit Dialog Click to end the chat and close the chat window.

Note: The agent collaborative chat window is substantially the same as the customer chat window.

Viewing Event Logs for a Third Party

This section discusses how to view event logs. To view event logs, use the (MCF_TP_EVNTLOG_CMP) component.

Image: Event Log page

This example illustrates an event log.

Domain	Time event logged to DBMS	Event type	Task Type	Queue ID	Agent ID	CTI Agent ID	ANI	DNIS	This DN	Other DN	Call ID
MCF1	12/07/2005 5:02:30PM	Restrt Ack	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/12/2005 12:09:40PM	User RQ	Voice	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/12/2005 12:09:40PM	User Event	Voice	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/15/2005 2:01:29PM	Q Shutdown	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/15/2005 2:01:29PM	Q Shutdown	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/15/2005 2:01:34PM	Restrt Ack	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/19/2005 1:33:14PM	Q Shutdown	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/19/2005 1:33:14PM	Q Shutdown	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/19/2005 1:34:02PM	Restrt Ack	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/19/2005 2:23:06PM	Accept	Generic	MARKETING	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/19/2005 2:23:06PM	DB Cntct	Generic	MARKETING	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/21/2005 11:05:00AM	Q Shutdown	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/21/2005 11:05:00AM	Q Shutdown	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)
MCF1	12/28/2005 11:28:28AM	Q Shutdown	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)	(blank)

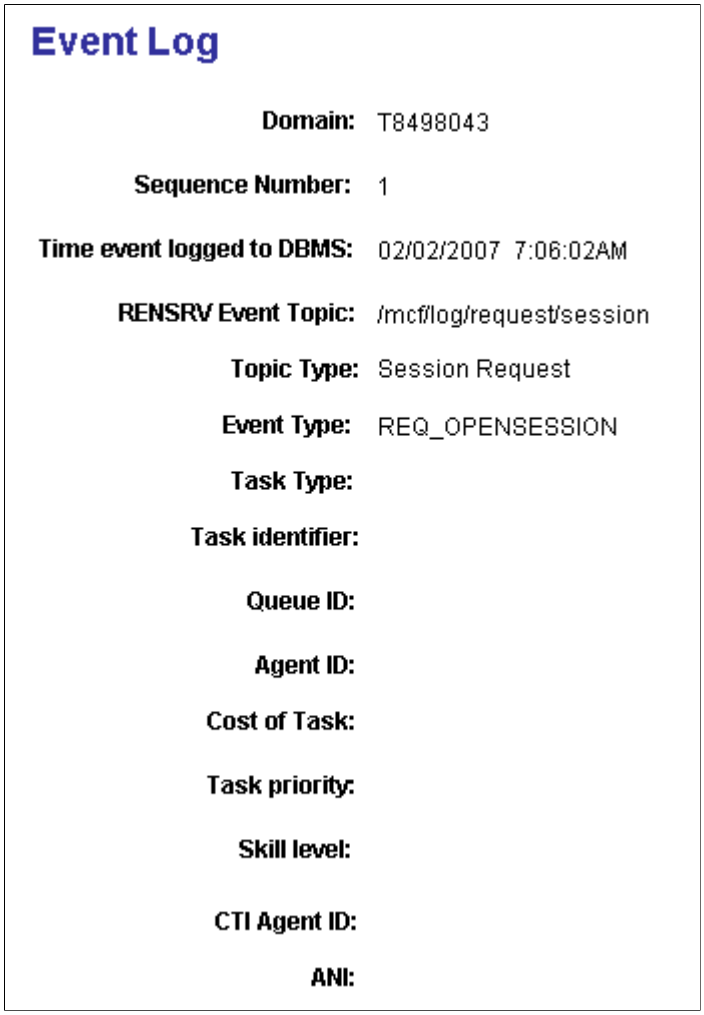
 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Viewing Event Logs for a Third Party

Select any event from the Event Log page to display the details of any event.

Image: Event Log page (page 1)

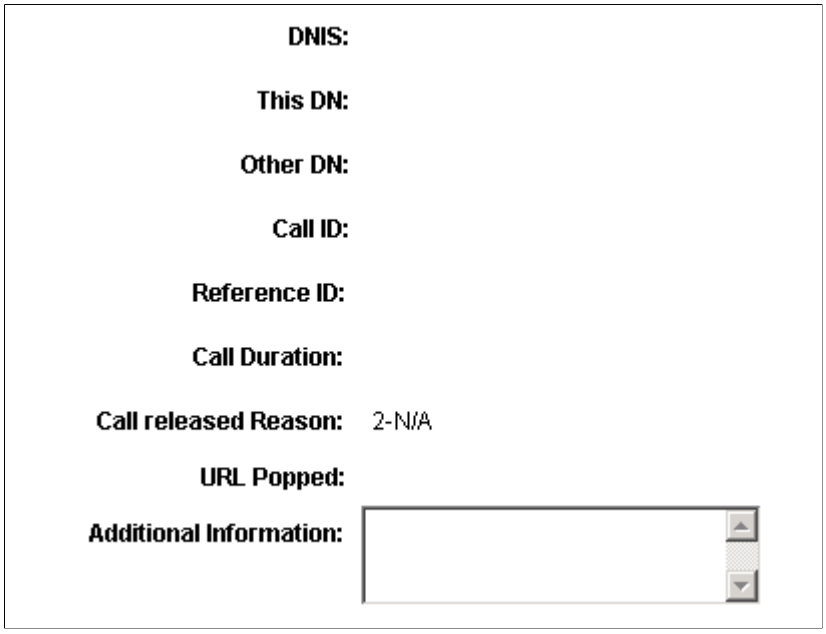
This example illustrates the fields and controls on the Event Log page (page 1 of 2).



Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Image: Event Log page (page 2)

This example illustrates the fields and controls on the Event Log page (page 2 of 2).



Viewing Broadcast Logs for a Third Party

To view the broadcast log, access the broadcast log (MCF_TP_BCASTLG_CMP) component.

Viewing Broadcast Logs for a Third Party

See Viewing Broadcast Logs.

Working with Third-Party Sample Pages

To demonstrate tools and functionality, use the Sample Pages (MCF_TP_DEMO_CMP) component. These sample pages demonstrate how JSMCAPI can be used with the PeopleCode behind it. The third party may develop their own interface, which may look significantly different from the sample pages included in the section.

This section discusses how to:

- Use the Customer Chat sample page.
- Use the Generic Event sample page.
- Use the Email sample page.
- Use the Sample Console page.
- Use the JSMCAPI Broadcast page.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Use the PeopleCode Broadcast page.

Using the Customer Chat Sample Page

See Using the Customer Chat Sample Page.

Using the Generic Event Sample Page

See Using the Generic Event Sample Page.

Using the Email Sample Page

See Using the Email Sample Page.

Using the Sample Console Page

Access Sample Console Page using the navigation path: PeopleTools, MultiChannel Framework, Third-Party Configuration, Sample Pages, Sample Console Page. Click the lookup button and select the REN cluster.

Image: Sample Console Page

This example illustrates the fields and controls on Sample Console Page. You can find definitions for the fields and controls later on this page.

MCF Sample Console Click to initiate the sample console.

Note: To demonstrate the MCF sample console, you must have an MCF cluster running and communicating with the specified REN server cluster.

After clicking MCF Sample Console, a new browser window displaying the sample console appears along with a tracer window. The Sample Console contains the following group boxes, which are detailed in the following topics:

- Open/Close Session
- Register User to Session
- Register Group, Login User to Group, and User-Group States
- Address Operations
- Register Extension to Session, and Extension Operations

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Buddies
- Buddies Bulk Registration
- Groups Bulk Registration
- Error Messages / Information
- Server State and Broadcasts

Note: To enable the new browser window to appear, disable any pop-up blocking software for your browser.

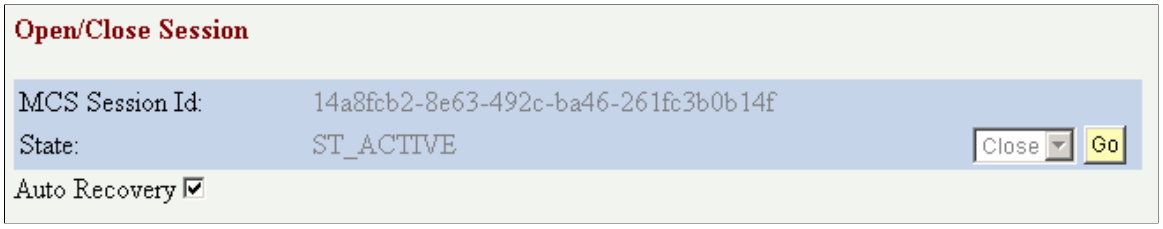
Before accessing the Open/Close Session group box, enable the third-party routing server.

Open/Close Session

Access the Open/Close Session group box. (Select PeopleTools, MultiChannel framework, Third-party Configuration, Sample Pages. Enter the MCF Cluster ID and click the Sample Console button).

Image: Open/Close Session group box

This example illustrates the fields and controls on the Open/Close Session group box. You can find definitions for the fields and controls later on this page.



MCS Session Id Displays the session ID of the MultiChannel server.

State Displays the state of the session.

Note: If the state is *ST_FAILED*, check the third-party routing server. It must be up and running.

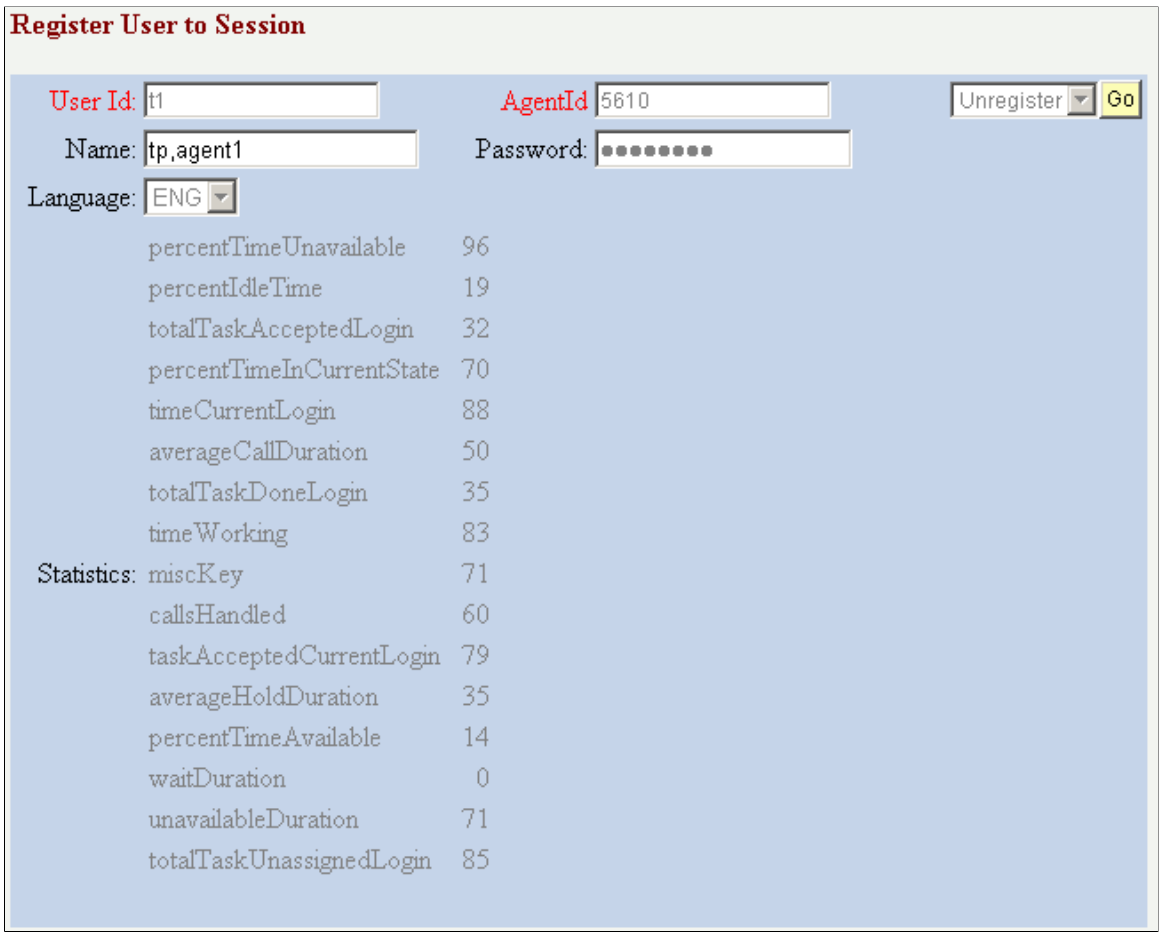
 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Register User to Session

Access the Register User to Session group box. (Select PeopleTools, MultiChannel framework, Third-party Configuration, Sample Pages. Enter the MCF Cluster ID and click the Sample Console button).

Image: Register User to Session group box

This example illustrates the fields and controls on the Register User to Session group box. You can find definitions for the fields and controls later on this page.



Register a user to the session in the Register User to Session group box.

User Id Enter user ID of the agent.

Agent Id Enter agent's ID.

> **Note:** Because a CTI agent can have an agent ID different from the agent's user ID, both values are required. If the agent has a password, the password is also required.

Name Enter the name of the agent.

Password Enter the password if the agent has a password.

Language Select the language.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The language field is reserved for future use. The default language is English.

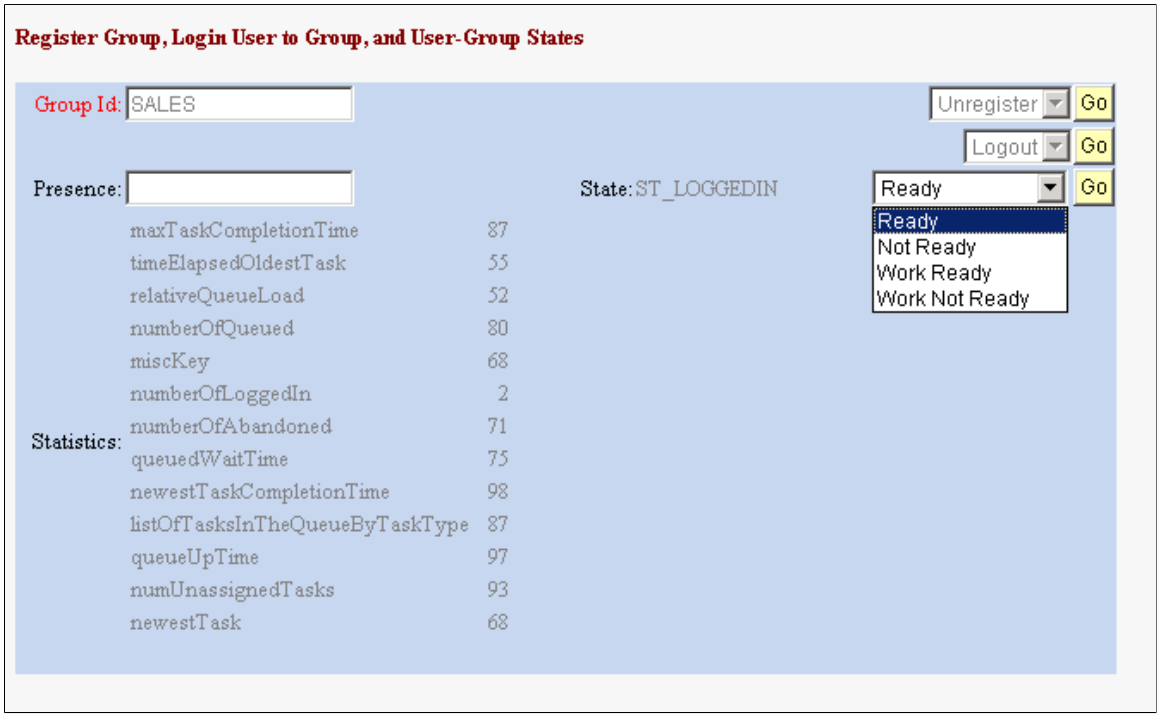
Statistics Displays the agent statistics.

Register Group, Login User to Group, and User-Group States

Access the Register Group, Login User to Group, and User-Group States group box. (Select PeopleTools, MultiChannel framework, Third-party Configuration, Sample Pages. Enter the MCF Cluster ID and click the Sample Console button).

Image: Register Group, Login User to Group, and User-Group States group box

This example illustrates the fields and controls on the Register Group, Login User to Group, and User-Group States group box. You can find definitions for the fields and controls later on this page.



Group Id Enter the name of the group to which the agent will be registered. The group ID is the same as the name of the queue to which the agent is logged in.

To register the agent with the specified group, click the Go button next to the Register field.

Note: After you click Go, the field value changes to *Unregister*.

Login Click the Go button to log in if *Login* appears in the drop-down list box.

Click the Go button to log out if *Logout* appears in the drop-down list box.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: After you log in to the group, a new pop-up window opens, prompting you for reason code data. If you register to an invalid group, a pop-up window opens showing the error messages.

See Understanding JSMCAPI.

Presence

Enter a presence description to associate with the state selected in the drop-down list box.

This presence value is different from any presence values predefined in the agent configuration. The agent cannot select from the predefined values.

State

Select an agent state from the following options:

- *Ready*
- *Not Ready*
- *Work Ready*
- *Work Not Ready*

Each state value has an associated state code that appears in the State field. To enable incoming calls, select *Ready* from the drop-down list box and click the Go button.

Statistics

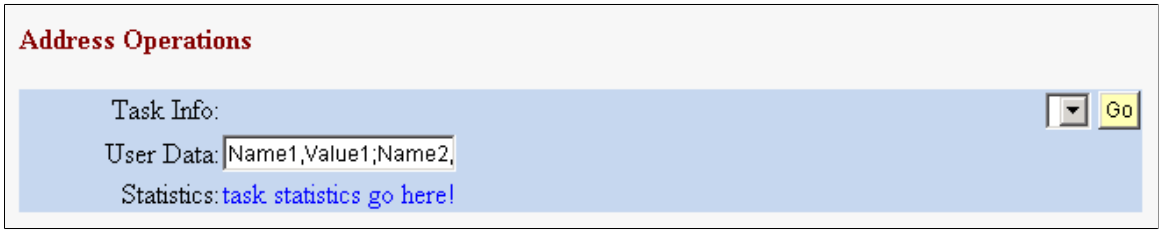
Displays the group statistics.

Address Operations

Access the Address Operations group box. (Select PeopleTools, MultiChannel framework, Third-party Configuration, Sample Pages. Enter the MCF Cluster ID and click the Sample Console button).

Image: Address Operations group box

This example illustrates the fields and controls on the Address Operations group box. You can find definitions for the fields and controls later on this page.



Task Info (task information)

Displays task information.

User Data

Displays user data.

Statistics

Displays task statistics.

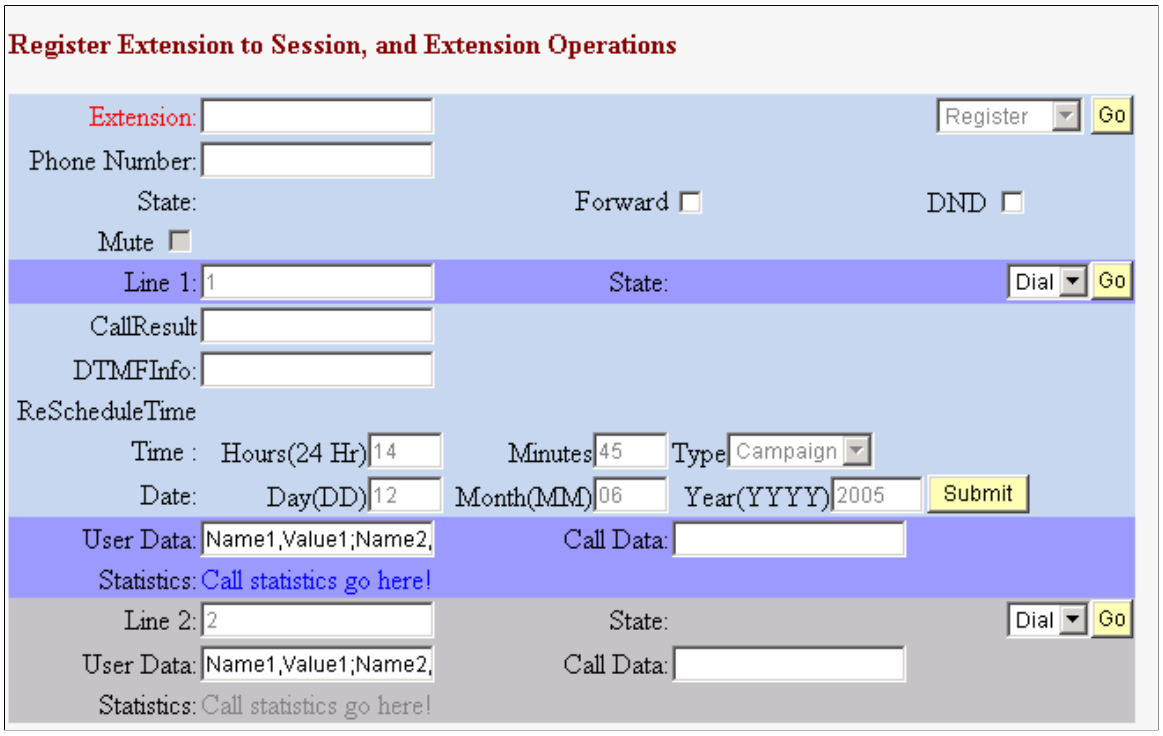
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Register Extension to Session and Extension Operations

Access the Register Extension to Session, and Extension Operations group box. (Select PeopleTools, MultiChannel framework, Third-party Configuration, Sample Pages. Enter the MCF Cluster ID and click the Sample Console button).

Image: Register Extension to Session and Extension Operations group box

This example illustrates the fields and controls on the Register Extension to Session and Extension Operations group box . You can find definitions for the fields and controls later on this page.



Extension	Enter a valid telephone extension number.
Phone Number	Enter a valid telephone number for use in subsequent operations, such as dial, forward, or do-not-disturb (DND).
Mute	Select to mute the call.
	Note: Mute is enabled only when the agent is on the call.
Forward	Select to forward incoming calls to the specified number.
DND (do not disturb)	Select to put the selected extension in do-not-disturb status.
Line 1 or Line 2	Displays a name or value for each line. On the sample console, the value will either be *1* or *2*.
DTMF Info (Dual Tone Multi-Frequency information)	Select to send DTMF digits on the line.
CallResult	Select to set call result on the line.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

ReScheduleTime Select to set reschedule time on the line.

Type Select to set reschedule type on the line.

User Data Enter name-value pairs representing user data to be attached to a call.

Call Data Enter any call data to be attached to the call.

Call data includes:

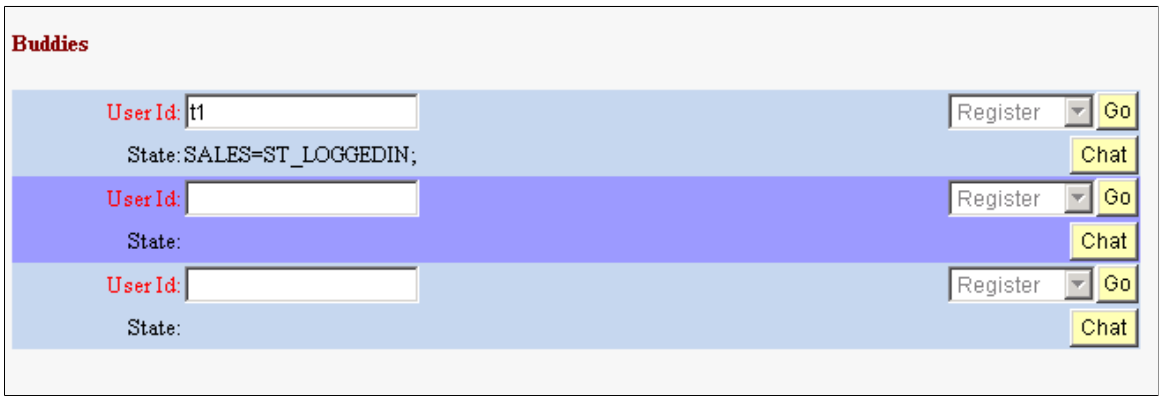
- *ANI*
- *DNIS*
- *THISDN*
- *OTHERDN*

Buddies

Access the Buddies group box. (Select PeopleTools, MultiChannel framework, Third-party Configuration, Sample Pages. Enter the MCF Cluster ID and click the Sample Console button).

Image: Buddies group box

This example illustrates the fields and controls on the Buddies group box. You can find definitions for the fields and controls later on this page.



User Id Enter the user ID of the buddy with whom you want to chat.

State Displays the state of the buddy.

Chat Click to initiate an agent-to-buddy chat session.

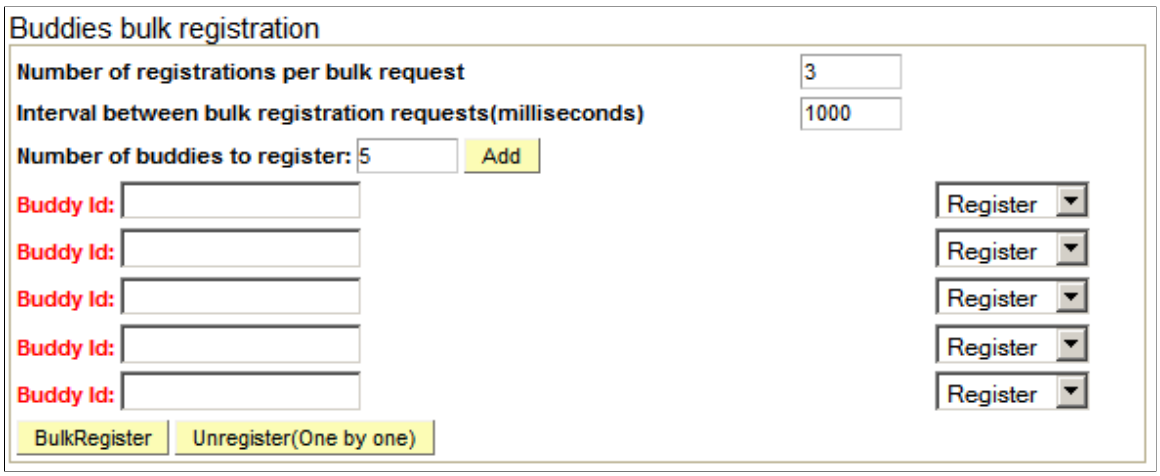
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Buddies Bulk Registration

Access the Buddies bulk registration group box. (Select PeopleTools, MultiChannel framework, Third-party Configuration, Sample Pages. Enter the MCF Cluster ID and click the Sample Console button).

Image: Buddies bulk registration group box

This example illustrates the fields and controls on the Buddies bulk registration group box. You can find definitions for the fields and controls later on this page.



Number of registrations per bulk request	Enter the number of registrations per bulk request.
Interval between bulk registration requests(milliseconds)	Enter the interval between bulk registration requests in milliseconds.
Number of buddies to register	Enter the number of buddies to add and click the Add button. Edit boxes for Buddy Id will appear.
BulkRegister	Click to register the buddies. An Unregister button will appear next to each Group Id. You can use this button to unregister a specific group.
Unregister(One by One)	Click to unregister groups one by one.

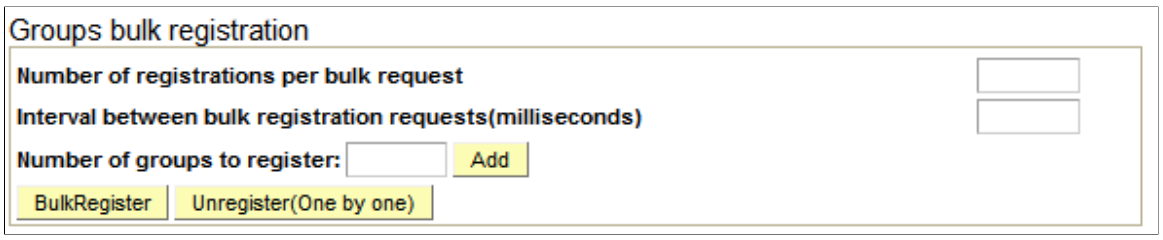
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Groups Bulk Registration

Access the Groups bulk registration group box. (Select PeopleTools, MultiChannel framework, Third-party Configuration, Sample Pages. Enter the MCF Cluster ID and click the Sample Console button).

Image: Groups bulk registration group box

This example illustrates the fields and controls on the Groups bulk registration group box. You can find definitions for the fields and controls later on this page.



Number of registrations per bulk request	Enter the number of registrations per bulk request.
Interval between bulk registration requests(milliseconds)	Enter the interval between bulk registration requests in milliseconds.
Number of groups to register	Enter the number of groups to add and click the Add button. Edit boxes for Group Id will appear.
BulkRegister	Click to register the buddies. An Unregister button will appear next to each Buddy ID. You can use this button to unregister a specific buddy.
Unregister(One by One)	Click to unregister buddies one by one.

Error Messages/Information

Access the Error Messages/Information group box (Select PeopleTools, MultiChannel framework, Third-party Configuration, Sample Pages. Enter the MCF Cluster ID and click the Sample Console button).

Image: Error Messages/Information group box

This example illustrates the fields and controls on the Error Messages/Information group box. You can find definitions for the fields and controls later on this page.

Any error messages associated with the process appear here.

Clear	Click the button to clear a displayed error message.

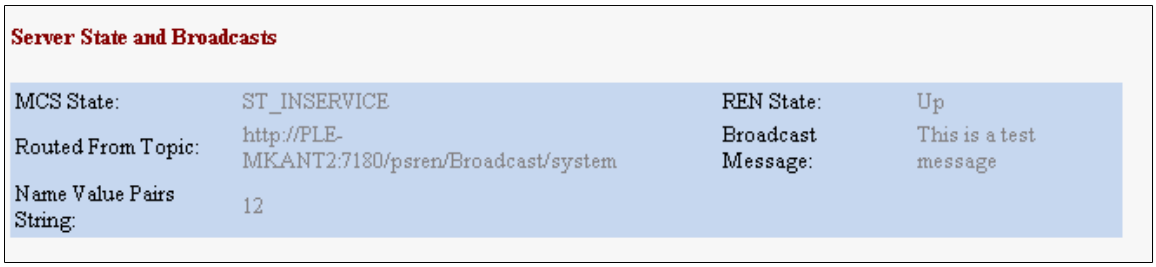
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Server State and Broadcasts

Access the Server State and Broadcasts group box. (Select PeopleTools, MultiChannel framework, Third-party Configuration, Sample Pages. Enter the MCF Cluster ID and click the Sample Console button).

Image: Server State and Broadcasts group box

This example illustrates the fields and controls on the Server State and Broadcasts group box. You can find definitions for the fields and controls later on this page.



MCS State Displays the MultiChannel server's state.

Routed From Topic Displays the machine name from where the broadcast message was sent.

Name Value Pairs Displays the name value pair string.

REN State Displays the status of the REN server.

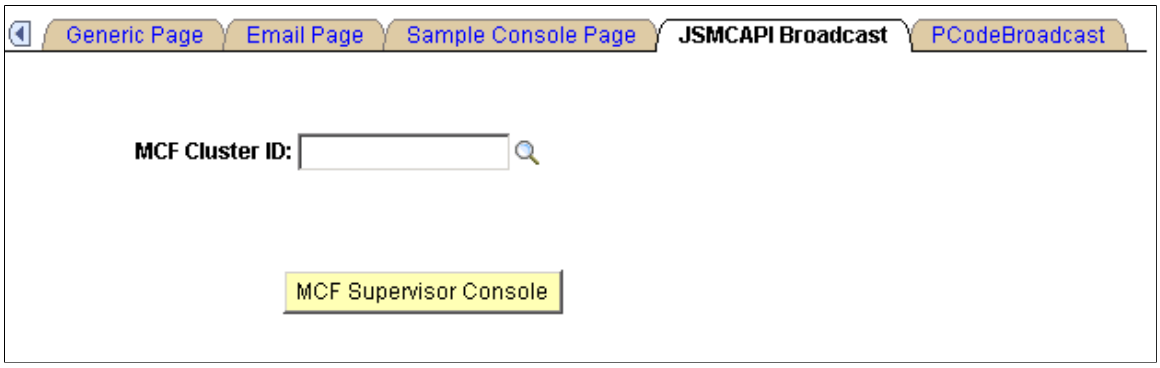
Broadcast Message Displays the supervisor's broadcast message.

Using the MCF Broadcast Page

This section discusses how to use the JSMCAPI Broadcast page for the third party.

Image: JSMCAPI Broadcast page

This example illustrates the fields and controls on the JSMCAPI Broadcast page. You can find definitions for the fields and controls later on this page.



MCF Cluster ID Select the REN server cluster on which to test the MCF Supervisor console.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: The MCF Supervisor console is specific to the REN cluster Id selected.

MCF Supervisor Console Click to initiate the supervisor console.

Note: To demonstrate the MCF Supervisor console, you must have MCF servers, both queue and log servers, running and communicating with the specified REN server cluster.

After you click MCF Supervisor Console, a new browser window appears that displays the sample supervisor console.

Note: To enable the new browser window to appear, disable any pop-up blocking software for your browser.

Related Links

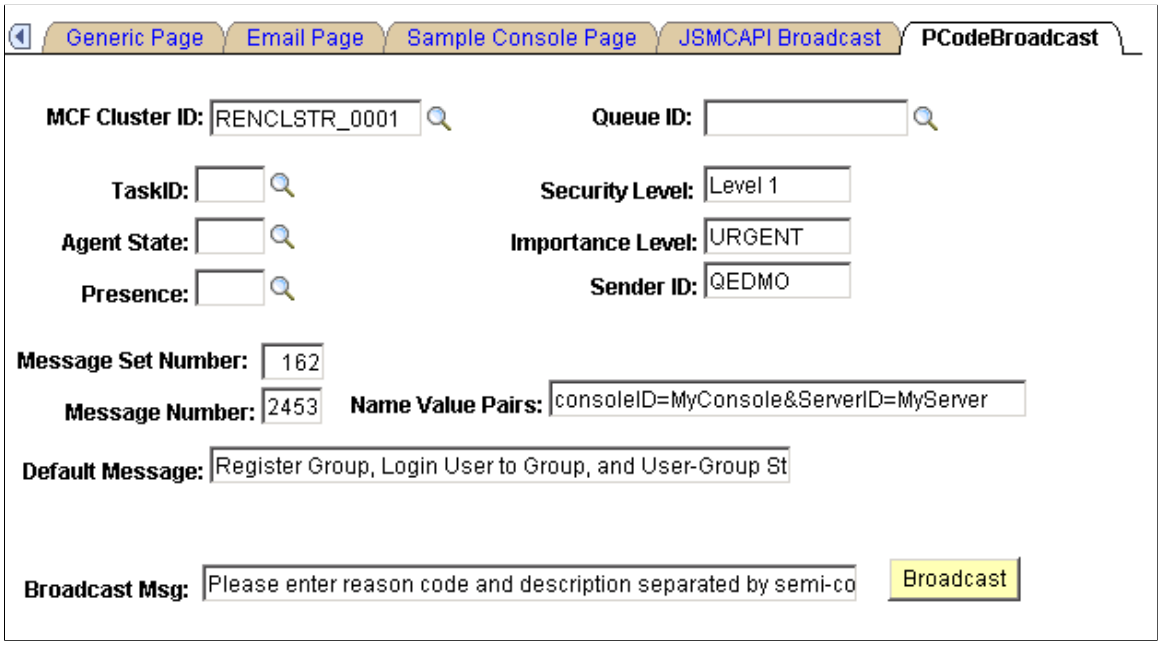
Using JSMCAPI Broadcast with MCF Supervisor Console

Using the PCodeBroadcast Page

This section discusses how to use the PCodeBroadcast page.

Image: PCodeBroadcast page

This example illustrates the fields and controls on the PCodeBroadcast page.



See Using PeopleCode Broadcast.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 13

Understanding JSMCAPI Classes

Understanding JSMCAPI

This section discusses JSMCAPI, a JavaScript-based application programming interface (API) used to build custom applications, such as MultiChannel consoles, or to enable MultiChannel functionality on the PeopleSoft Pure Internet Architecture page. The API is built on the REN server JavaScript client and is a pure JavaScript API.

JSMCAPI is the interface through which the application developer accesses the JSMCAPI functionality. JSMCAPI provides a set of objects, such as User, Address, Group, and so on. The PSMC, a global object, provides all those methods called to send agent requests to the server; it also provides an event handler callback method for the PeopleSoft MultiChannel Application Programming Interface (PSMCAPI) call backs for events coming from the CTI server.

The JSMCAPI communicates with the PSMCAPI through the REN server using MCP (Multi-Channel Protocol). MCP includes topic and event definition. In general, JSMCAPI:

- Sends agent requests to the MultiChannel Server.

  The requests can be agent state requests such as login, logout, set ready, and so on. Requests can also be call requests, such as dial-out, answer the call, transfer/conference, and so on.

- Receives PBX/Switch events from the CTI server.

  Events can be the agent state change, such as event ready, not ready, and so on. Events can also be call events, such as incoming call, call released, call transferred, and so on.

Understanding JSMCAPI Classes

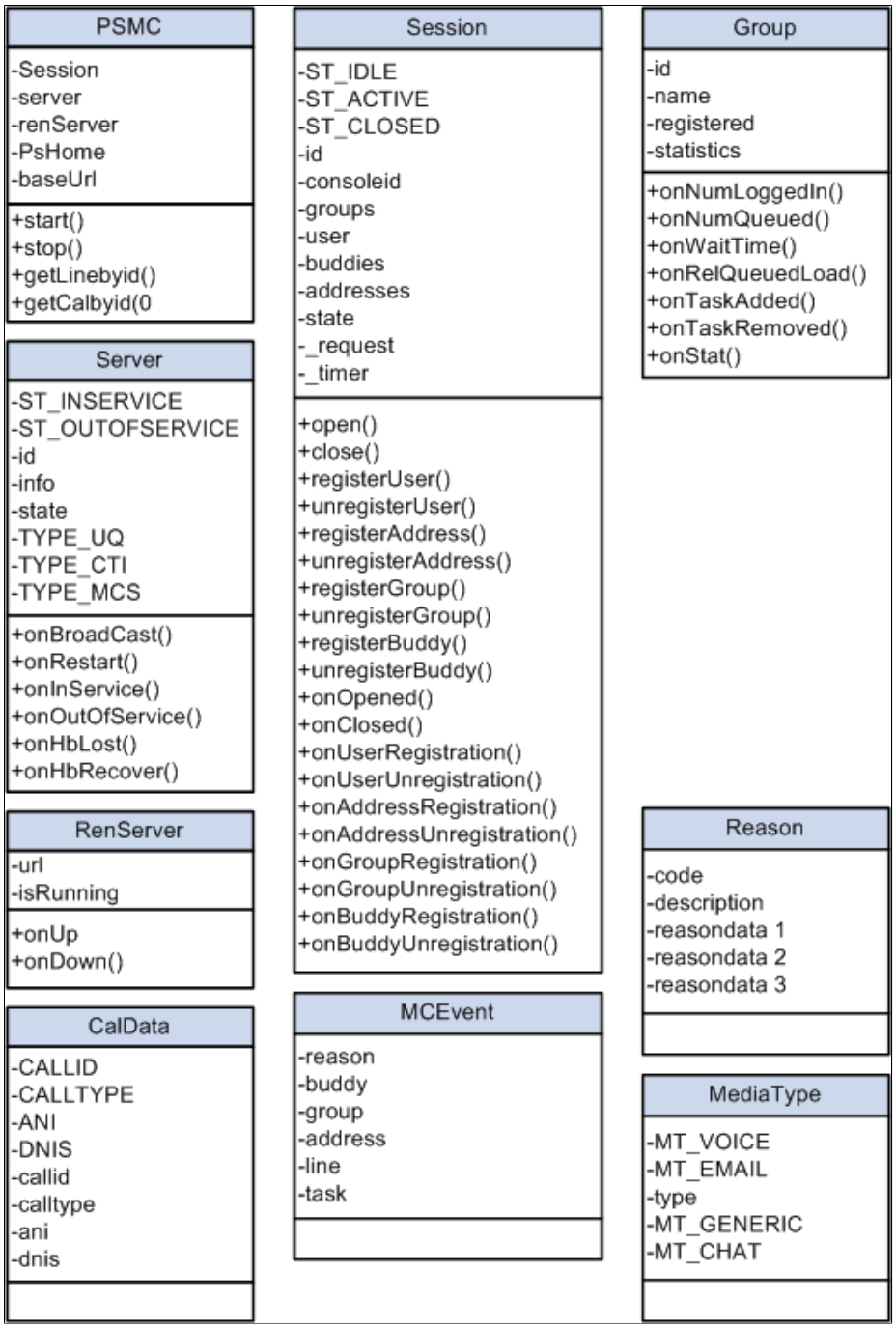
This sections gives an overview of all the JSMCAPI classes.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The following class diagrams explain all the classes, fields, and the methods.

Image: Class diagram part 1

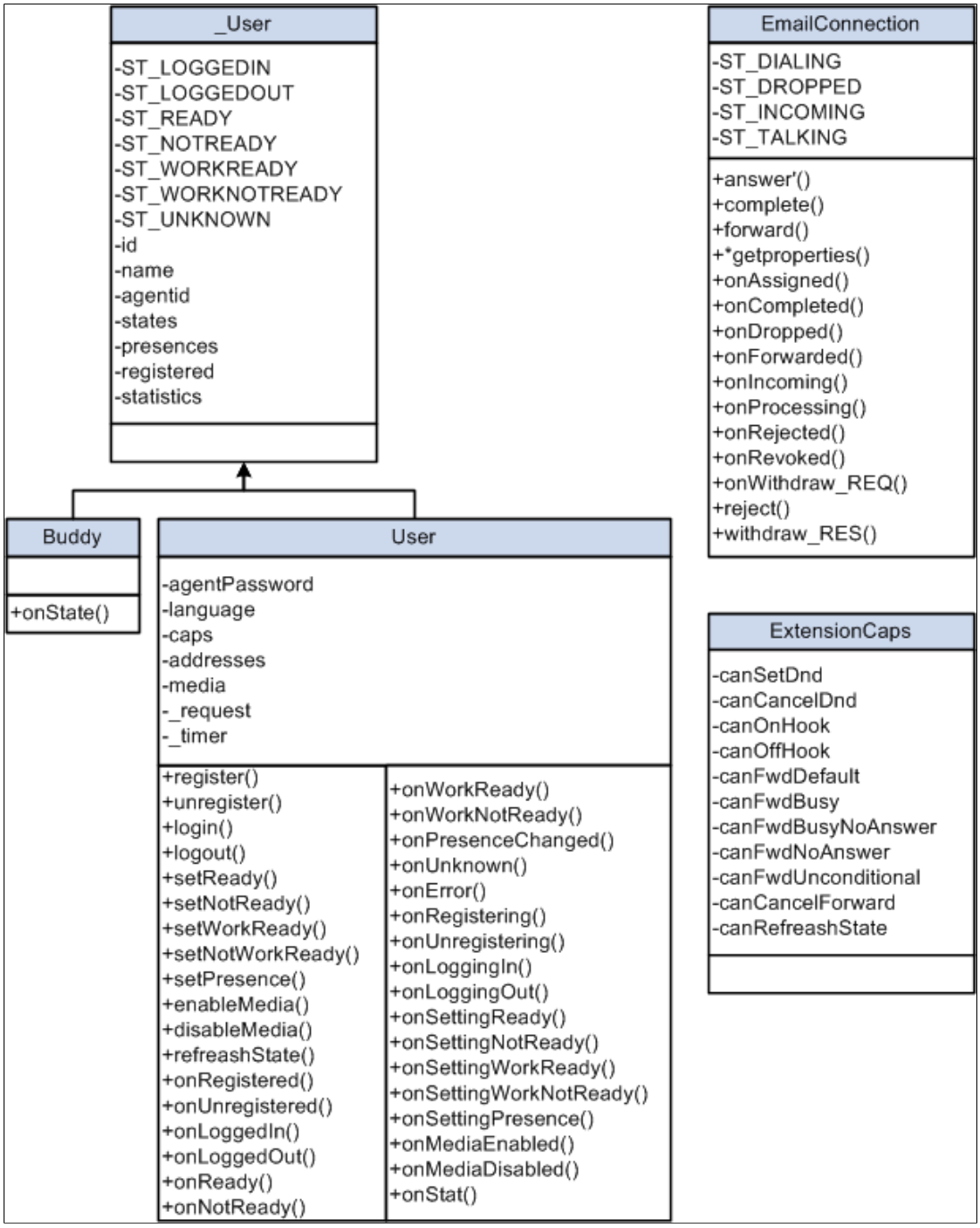
The following class diagram explains the classes, fields, and methods of JSMCAPI (diagram 1 of 4).



 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Image: Class diagram part 2

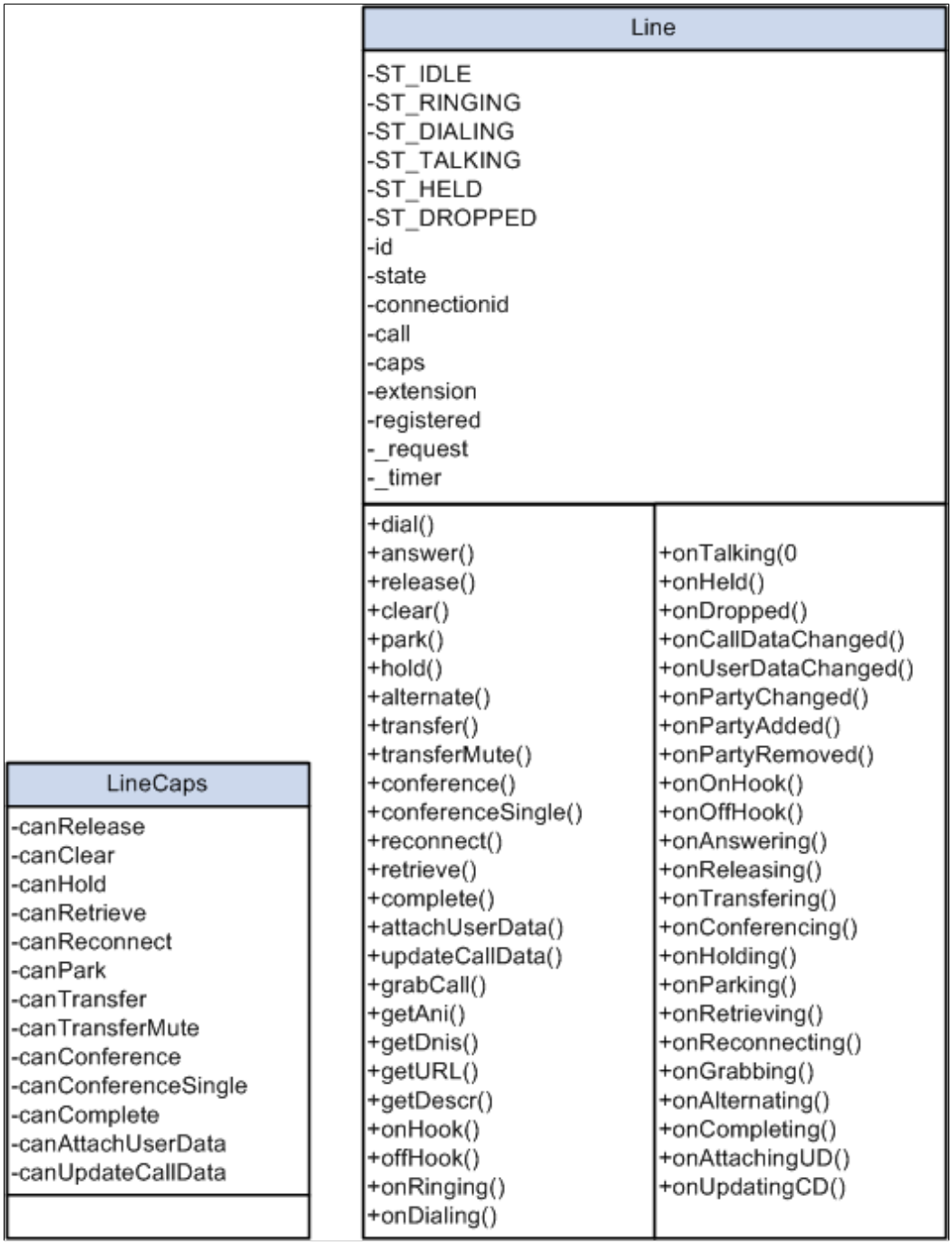
The following class diagram explains the classes, fields, and methods of JSMCAPI (diagram 2of 4).



Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Image: Class diagram part 3

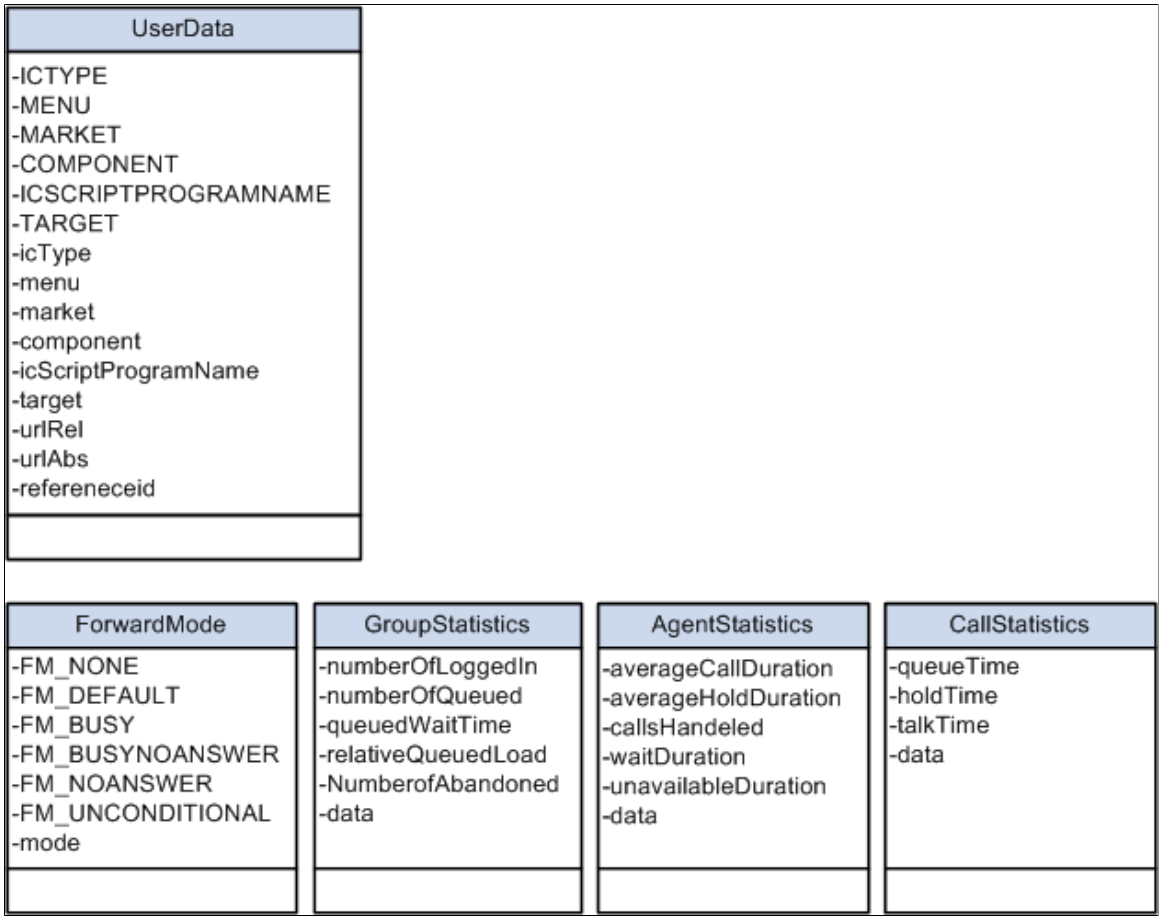
The following class diagram explains the classes, fields, and methods of JSMCAPI (diagram 3 of 4).



 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Image: Class diagram part 4

The following class diagram explains the classes, fields, and methods of JSMCAPI (diagram 4 of 4).



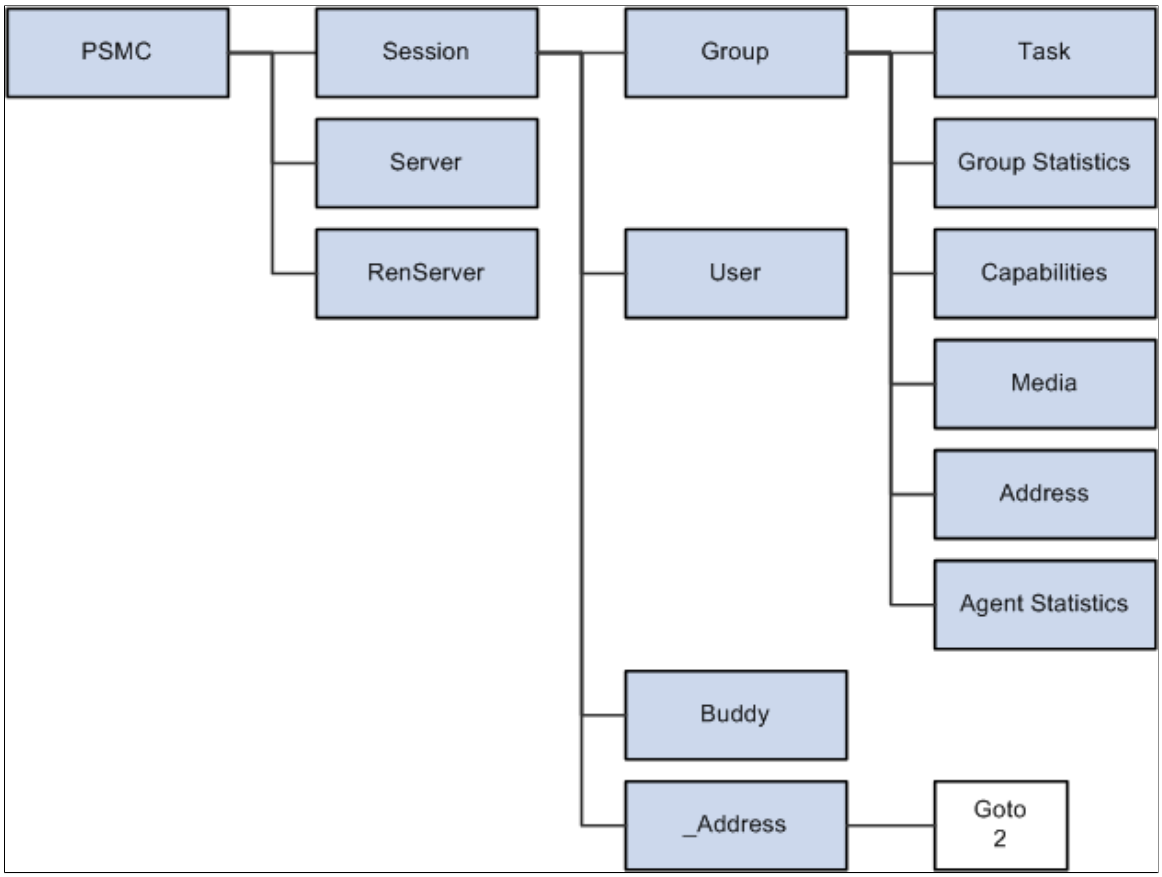
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PSMC

PSMC is the base class that an application accesses to start an instance of JSMCAPI. The application can access all JSMCAPI functionality through this object. A session object is created from this class.

Image: PSMC base class part 1

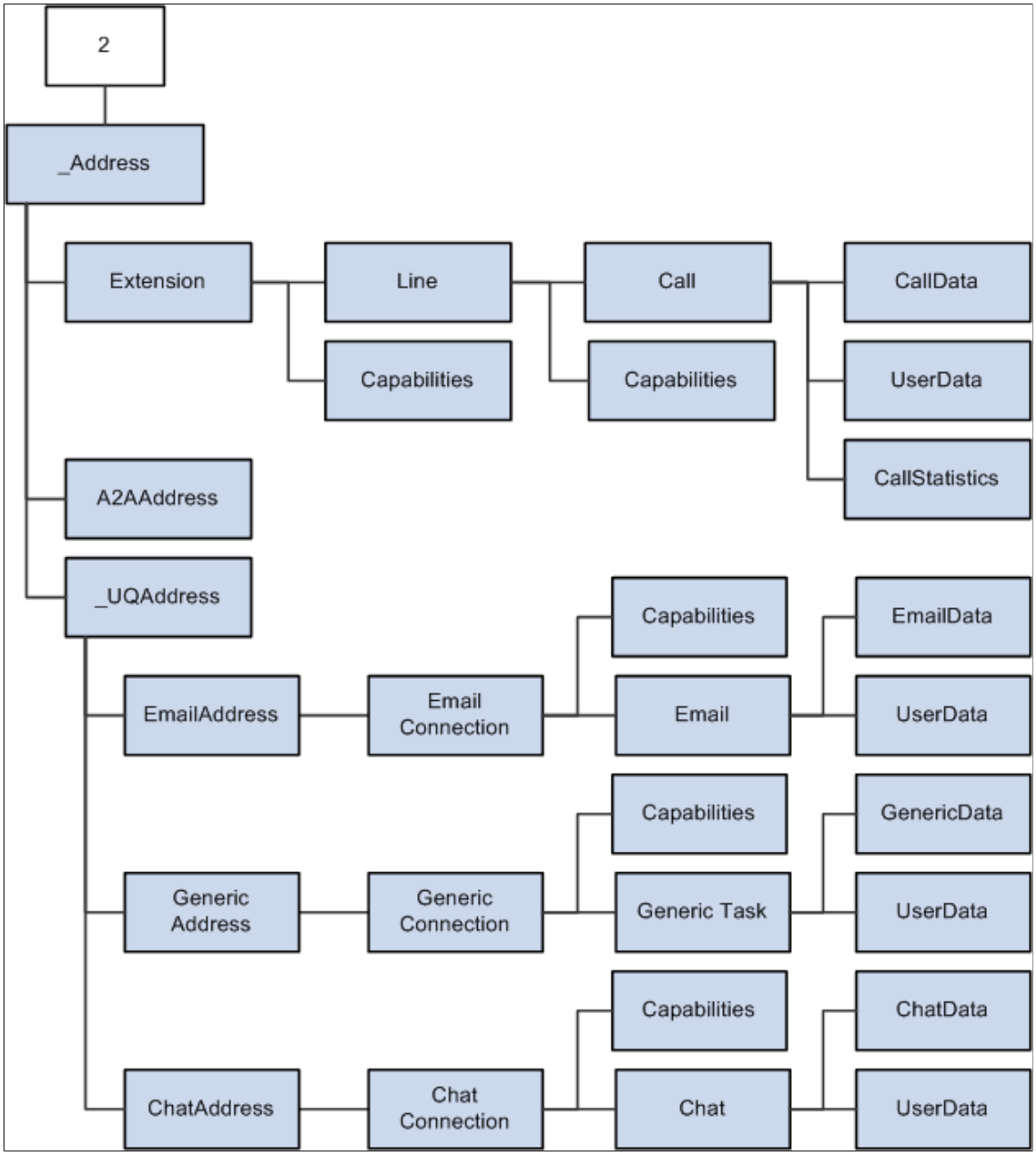
The following diagram explains the relationship of PSMC with all other classes (diagram 1 of 2).



 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Image: PSMC base class part 2

The following diagram explains the relationship of PSMC with all other classes (diagram 2 of 2).



Related Links

PSMC Class Constructor

Server

The Server class refers to the routing server and can execute events for specific server states. Three types of servers are available: CTI, queue server, and MultiChannel server. The constants are CTI, UQ and MCS, respectively.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Related Links

Server Class Constructor

RENServer

The RENServer cluster is represented in the RENServer class. It provides the URL and the server status (active or shutdown).

Related Links

RenServer Class Constructor

Session

Sessions are set for users registering with the server. Addresses, buddies, and groups are registered for the user with the session. The connection between the server and JSMCAPI is a session. There is only one session object per PSMC object.

Related Links

Session Class Constructor

_Address

The _Address class identifies the user with a unique ID. The ID identifies the user to the routing system and is unique for each channel or media type.

The subclasses of _Address class are:

- Extension
- _UQAddress
- A2AchatAddress

Further delineation of address by media type is provided by:

- ChatAddress
- EmailAddress
- GenericAddress

ChatAddress, EmailAddress, and GenericAddress classes extend _UQAddress.

Related Links

Extension Class Constructor
_UQAddress Class Constructor
A2AChatAddress Class Constructor
ChatAddress Class Constructor
EmailAddress Class Constructor
GenericAddress Class Constructor

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Line

Line class describes the line of the extension for a call task. This version supports one extension with two lines and two extensions with one line in each.

Related Links

Line Class Constructor

Connection

Tasks are routed to agents through a connection. Email, chat and generic have a dedicated connection class, like EmailConnection, ChatConnection, and GenericConnection. These connections provide task-specific manipulation functions.

Related Links

ChatConnection Class Constructor
EmailConnection Class Constructor

Group

The group class defines the group or queue information. Each session can have one or more group objects.

Related Links

Group Class Constructor

Task

This abstract base class defines a unit of work. The classes that extend task per media type are:

- Call
- Email
- Chat
- A2AChat
- GenericTask

Related Links

Call Class Constructor
Email Class Constructor
Chat Class Constructor
A2AChat Class Constructor
GenericTask Class Constructor

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

_User

_User is a base class. The subclasses are User and Buddy. These classes define the properties that pertain to the user such as the addresses, languages, or presence. _User is a virtual class and should not be instantiated.

Related Links

_User Class Constructor

MediaType

This class defines the media that an agent can handle.

Related Links

MediaType Class Constructor

Reason

The Reason class defines the message or error message that accompanies an event or request. Globalization of the messages is implemented. Furthermore, extra data can be passed in this object for providing a detailed message.

Related Links

Reason Class Constructor

Statistics

Statistics are provided by the routing server for agent, call, task, group, and user. The following classes describe the statistics for each component:

- AgentStatistics
- CallStatistics
- TaskStatistics
- GroupStatistics
- GroupStatistics1
- GroupStatistics2
- UserStatistics1
- UserStatistics2

Related Links

AgentStatistics Class Constructor
CallStatistics Class Constructor
TaskStatistics Class Constructor

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

GroupStatistics Constructor
GroupStatistics1 Class Constructor
GroupStatistics2 Class Constructor
UserStatistics1 Class Constructor
UserStatistics2 Class Constructor

Data

When a task is introduced to the system, data pertaining to the task is defined in the following classes:

- CallData
- EmailData
- ChatData
- GenericData

Application-specific data is provided for tasks via the AppData class. Similarly, user data is defined in UserData.

Related Links

CallData Class Constructor
EmailData Class Constructor
ChatData Class Constructor
GenericData Class Constructor

Globals

This class defines functions that are used universally.

Related Links

GLOBALS Class Fields

MCEvent

Events passed to the application handler are defined by MCEvent.

Related Links

MCEvent Class Constructor

Caps

Capabilities (Caps) define the ability or allowed actions for a component. The following classes define specific capabilities:

- ChatConnectionCaps
- EmailConnectionCaps

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- GenericConnectionCaps
- ExtensionCaps
- LineCaps
- ChatConnectionCaps
- EmaiConnectionCaps
- GenericConnectionCaps
- UserCaps

Related Links

ChatConnectionCaps Class Constructor
EmailConnectionCaps Class Constructor
GenericConnectionCaps Class Constructor
ExtensionCaps Class Constructor
LineCaps Class Constructor

ForwardMode

ForwardMode describes various forward modes that can be used while setting the forwarding mode for an Address.

Related Links

ForwardMode Class Constructor

_Address Class Hierarchy

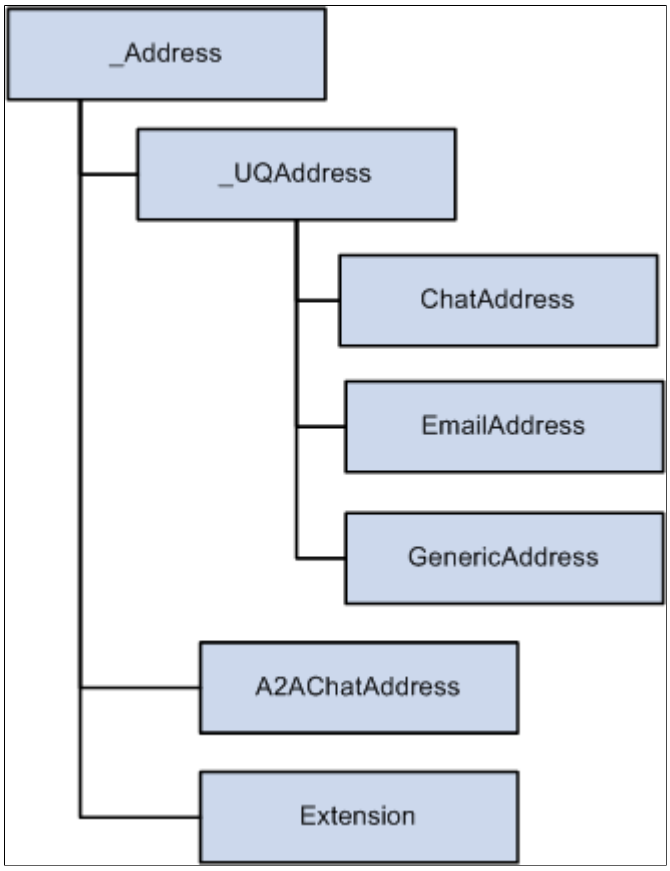
The _Address class can be extended by other subclasses.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Do not instantiate _Address or _UQAddress class objects. Instead, use the child classes.

Image: _Address class hierarchy

The following flow chart shows the different subclasses of _Address class and how they interrelate.



Related Links

_Address Class Constructor
_UQAddress Class Constructor
A2AChatAddress Class Constructor
Extension Class Constructor
ChatAddress Class Constructor
EmailAddress Class Constructor
GenericAddress Class Constructor

_Address Class Constructor

The following is the _Address class constructor.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

_Address

Syntax

_Address()

Description

Creates an _Address object that describes the address.

Parameters

None.

Returns

An _Address object.

_Address Class Fields

This section discusses the JSMCAPI _Address class fields, which are described in alphabetical order.

caps

Description

The capacities of the address.

Type: object.

id

Description

The ID of the address.

Type: string.

_Address Class Callback Event Method

The following is the callback event method used with a JSMCAPI _Address object.

onError

Syntax

onError(*event*)

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Fires on the event of an address error.

_UQAddress Class Constructor

The following is the UQAddress class constructor.

_UQAddress

Syntax

_UQAddress()

Description

Creates a _UQAddress object, which is the base for the various addresses associated with the universal queue server.

Parameters

None.

Returns

A universal queue address object.

_UQAddress Class Fields

The _UQAddress class inherits the following fields from the _Address class:

- caps
- id

See _Address Class Fields.

The following are the _UQAddress class fields.

Tasks

Description

The associative array of the tasks on the queue managed by the address.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

_UQAddress Methods

The following are the _UQAddress methods.

acceptTask

Syntax

acceptTask(*task, reason*)

Description

Sends a request to the universal queue server to signal that the client has accepted the task.

Parameters

task	The task ID.
reason	The associated reason code.

Returns

Request number.

dequeueTask

Syntax

dequeueTask(*task*)

Description

Sends a request to the universal queue server to remove the task from its queue.

Parameters

task	The task ID.

Returns

Request number.

_UQAddress Class Callback Event Methods

The _UQAddress class inherits the onError callback event method from the _Address class.

See _Address Class Callback Event Method.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The following are the callback event methods used with a JSMCAPI universal queue address object. The callback event methods are described in alphabetical order.

onAccepted

Syntax

onAccepted(*event*)

Description

Fires when the task is accepted.

onAcceptingTask

Syntax

onAcceptingTask(*event*)

Description

Fires as the task is being accepted.

onDequeueingTask

Syntax

onDequeueingTask(*event*)

Description

Fires as the task is being dequeued.

onNotify

Syntax

onNotify(*event*)

Description

Fires on task notification.

onTaskAdded

Syntax

onTaskAdded(*event*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Fires when the task is added to the addressed queue.

onTaskRemoved

Syntax

onTaskRemoved(*event*)

Description

Fired when the task is removed

onUnassigned

Syntax

onUnassigned(*event*)

Description

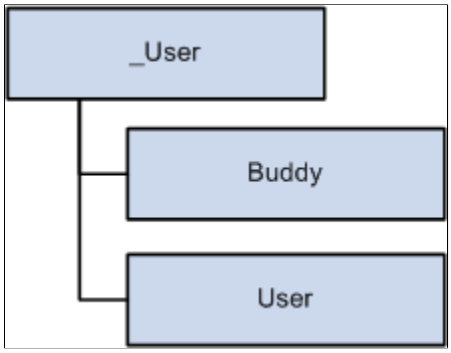
Fired when the task is unassigned.

_User Class Hierarchy

The _User class can be extended by other subclasses.

Image: _User class hierarchy

The following flow chart shows the different subclasses of _User class and how they interrelate.



Related Links

Buddy Class Constructor
User Class Constructor

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

_User Class Constructor

The following is the _User class constructor.

_User

Syntax

```
_User()
```

Description

Describes the base class of user/agent.

Parameters

None.

Returns

A _User object.

_User Class Fields

The following are the fields associated with the JSMCAPI _User class. These fields are discussed in alphabetical order.

agentID

Description

The agent's agent ID.

Type: string.

caps

Description

The agent capabilities on each group.

Type: associative array.

id

Description

The agent's PeopleSoft user ID.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Type: string.

name

Description

The agent's user name.

Type: string.

presences

Description

The agent's presence on each group.

Type: associative array.

states

Description

Agent state on each group.

Type: associative array, including the constants beginning with ST_*.

These constants must be accessed from an instantiated object.

ST_LOGGEDIN

Description

The agent is logged in.

ST_LOGGEDOUT

Description

The agent is logged out.

ST_NOTREADY

Description

The agent is not ready.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

ST_READY

Description

The agent is ready.

ST_UNKNOWN

Description

The agent's state is unknown.

ST_WORKNOTREADY

Description

The agent is in the work not ready state.

ST_WORKREADY

Description

The agent is in the work ready state.

statistics

Description

Agent statistics for CTI.

Type: AgentStatistics object.

Related Links

AgentStatistics Class Constructor

statistics1

Description

Agent statistics for the universal queue server.

Type: UserStatistics1 object.

Related Links

UserStatistics1 Class Constructor

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

statistics2

Description

Agent statistics for the universal queue server.

Type: UserStatistcs2 object.

Related Links

UserStatistics2 Class Constructor

A2AChat Class Constructor

The following is the A2AChat class constructor.

A2AChat

Syntax

A2AChat(*event*,*address*,*chatType*)

Description

Creates an agent-to-agent chat object.

The chat is related to the A2AChatAddress. It does not extend Task as it is not generated or tracked by the universal queue server. Construction occurs inside the A2AChatAddress.

Parameters

event	Enter the event associated with the agent-to-agent chat.
address	Enter the address from A2AChatAddress.
chatType	Enter the chat type, for example consult or answer.

Returns

An agent-to-agent chat object.

A2AChat Class Fields

The following are the A2AChat fields.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

address

Description

The address containing the agent-to-agent chat.

Type: A2AChatAddress object.

Related Links

A2AChatAddress Class Constructor

agentID

Description

The associated agent's ID.

Type: string.

agentName

Description

The associated agent's name.

Type: string.

appData

Description

The application data that is provided to the client with the notification event.

Type: AppData object.

Related Links

AppData Class Constructor

chatType

Description

The type of chat, either A2AChat.TYPE_CONSULT or A2AChat.TYPE_ANSWER.

Type: string, with one of the following constants:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Value	*Description*
TYPE_ANSWER	The A2AChat type answer. This type is generated when another agent wants this agent to answer an A2AChat.
TYPE_CONSULT	The A2AChat type consult. This type is generated when this agent wants to consult another agent.

Example

The A2AChat.TYPE_* can only be accessed as in the following example:

```
var value = A2AChat.TYPE_CONSULT;
```

Related Links

GLOBALS Class Fields

customerName

Description

The name of the customer in the ongoing chat.

Type: object.

id

Description

The ID of this chat.

Type: string.

isConference

Description

A flag to check whether it is conference or not.

Type: object.

jr

Description

JournalRoute

Type: string.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

question

Description

Question that is asked when customer initiates chat.

Type: object.

subject

Description

Subject of the chat. Customer will provide this subject when he initiates chat.

Type: object.

type

Description

The task type, chat.

Type: string.

uniqueueId

Description

The unique ID.

Type: string.

A2AChat Class Method

The following is the A2AChat class method.

getURL

Syntax

getURL(*defaultURL*)

Description

Returns the URL for the given task.

Parameters

defaultURL	If not null, this value will override the generated base URL.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Returns

The URL associated with this task.

A2AChatAddress Class Constructor

The following is the A2AChatAddress constructor.

A2AChatAddress

Syntax

A2AChatAddress()

Description

Agent-to-agent chat address. Handles the creation of agent-to-agent chats for incoming and outgoing chat communication.

Parameters

None.

Returns

An A2AChatAddress object.

A2AChatAddress Class Fields

The A2AChatAddress class inherits the following fields from the _Address class:

- id
- caps

See _Address Class Fields.

The following are the fields associated with the A2AChatAddress class.

id

Description

The address ID.

Type: string.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

tasks

Description

The array of A2AChats associated with this address.

Type: associative array.

Related Links

GLOBALS Class Fields

A2AChatAddress Class Method

The following is the A2AChatAddress class method.

initiateChat

Syntax

initiateChat(*agentId*)

Description

Initializes the A2AChatAddress by creating an A2AChat task.

Parameters

agentId	The ID of the agent with whom you want to chat.

Returns

Request number.

A2AChatAddress Class Callback Event Methods

The A2AChatAddress class inherits the onError callback event method from the _Address class.

See _Address Class Callback Event Method.

The following are the A2AChatAddress class callback methods:

onChatEnded

Syntax

onChatEnded(*event*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

This event gets called when an agent to agent chat is closed.

onInitiatingChat

Syntax

onInitiatingChat(*event*)

Description

This event gets called when chat is getting initiated.

onNotify

Syntax

onNotify(*event*)

Description

This event gets called when there is a new A2Achat notification.

AgentStatistics Class Constructor

The following is the AgentStatistics class constructor.

AgentStatistics

Syntax

AgentStatistics()

Description

The agent statistics information.

Parameters

None.

Returns

An AgentStatistics object.

AgentStatistics Class Fields

The following are the AgentStatistics class fields.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

averageCallDuration

Description

The average call duration, in seconds, for an agent.

Type: number

averageHoldDuration

Description

The average hold duration, in seconds, for an agent.

Type: number.

callsHandled

Description

The total number of calls handled by an agent.

Type: number.

data

Description

An associative array of key-value pairs that includes all agent statistics.

Type: associative array.

percentIdleTime

Description

The percentage of time for which the agent is idle.

Type: number

percentTimeAvailable

Description

The percentage of time for which the agent is in ready state.

Type: number

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

percentTimeInCurrentState

Description

The percentage of time in current state.

Type: number

percentTimeUnavailable

Description

The percentage of time for which the agent is in not ready state.

Type: number

timeCurrentLogin

Description

The time since login for the agent.

Type: number.

timeWorking

Description

The time spent working on tasks.

Type: number.

totalTaskAcceptedLogin

Description

The number of tasks accepted since login time

Type: number

totalTaskDoneLogin

Description

Number of tasks done by the agent since login time i.e. current login.

Type: number.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

totalTaskUnassignedLogin

Description

Number of tasks unassigned since login time

Type: number.

unavailableDuration

Description

The total time for which an agent is unavailable.

Type: string.

waitDuration

Description

The total time an agent has to wait for a call.

Type: string.

AppData Class Constructor

The following is the AppData class constructor.

AppData

Syntax

```
AppData()
```

Description

The AppData object describes the key-value pairs of the data received on the agent-to-agent chat event.

Parameters

None.

Returns

Type: AppData object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

AppData Class Fields

The following are the AppData class fields.

data

Description

An associative array of the data in name value pairs.

Type: associative array.

groupId

Description

The group ID passed in by the notify event.

Type: string.

jr

Description

The JournalRoute.

Type: string.

question

Description

The initial question of the task.

Type: string.

strData

Description

The raw application data as a string.

Type: string.

subject

Description

The subject of the task.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Type: string.

uniqueId

Description

The unique ID.

Type: string.

url

Description

The URL passed in by the AppData.

Type: string with the following constant:

Value	*Description*
URL	The associated URL.

userId

Description

The agent's user ID.

Type: string.

username

Description

The agent's user name.

Type: string.

wizUrl

Description

The wizard URL that is used for popping up new task window.

Type: string.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

AppData Class Method

The following is the AppData class method.

addKeyValue

Syntax

addKeyValue(*key, value*)

Description

Add the key-value pair to the AppData object.

Parameters

key	The key.
value	The value.

Returns

None.

Buddy Class Constructor

The Buddy class extends the _User class.

See _User Class Constructor.

The following is the Buddy class constructor.

Buddy

Syntax

Buddy()

Description

Describes the buddy. An agent can register buddies and is notified of state changes.

Parameters

None.

Returns

A Buddy object.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Buddy Class Fields

The Buddy class inherits the following fields from the _User class:

- agentID
- caps
- id
- name
- presences
- states
- statistics
- statistics1
- statistics2

See _User Class Constructor.

Buddy Class Callback Event Methods

The following are the Buddy class callback event methods.

onStat1

Syntax

onStat1(*event*)

Description

Fires when statistics1 is received.

onStat2

Syntax

onStat2(*event*)

Description

Fires when statistics2 is received.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onState

Syntax

onState(*event*)

Description

Fires when the state event is received.

Call Class Constructor

The Call class extends the Task class.

See Task Class Constructor.

The following is the Call class constructor.

Call

Syntax

Call(*strCall*)

Description

The Call object describes the call task information associated with the line.

Parameters

strCall	Enter the call to use with this object.

Returns

A Call object.

Call Class Fields

The Call class inherits the following fields from the Task class:

- caseid
- cost
- customerid
- group
- id

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- onStat
- priority
- type
- urlAbs
- urlRel

See Task Class Fields.

The following are additional Call class fields.

line

Description

The line object that is associated with this call.

Type: line object.

Related Links

Line Class Constructor

statistics

Description

Call statistics object associated with the call.

Type: CallStatistics object.

Related Links

CallStatistics Class Constructor

CallData Class Constructor

The following is the CallData class constructor.

CallData

Syntax

```
CallData()
```

Description

The CallData object describes the key-value pairs of the call data with the call object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

None.

Returns

A CallData object.

CallData Class Fields

The following are the CallData class fields:

ani

Description

The ANI caller id.

Type: string.

callId

Description

The call ID.

Type: string.

callType

Description

The call type.

Type: string.

data

Description

The key-value pairs of the call data.

Type: string.

dnis

Description

The DNIS callee ID.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Type: string.

CallData Class Method

The following is the CallData class method.

addKeyValue

Syntax

addKeyValue(*key*), *value*

Description

Add key value to the CallData object.

Parameters

key	The key.
value	The value.

Returns

None.

CallStatistics Class Constructor

The following is the CallStatistics class constructor.

CallStatistics

Syntax

CallStatistics()

Description

The call statistics information.

Parameters

None.

Returns

A CallStatistics object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

CallStatistics Class Fields

The following are the CallStatistics class fields.

data

Description

An associative array of key-value pairs that includes all statistics.

Type: associative array.

holdTime

Description

Duration that the call is on hold.

Type: string.

queueTime

Description

Duration that this call has been in the queue.

Type: string.

talkTime

Description

Duration that the call is established.

Type: string.

Chat Class Constructor

The Chat class extends the Task class.

See Task Class Constructor.

The following is the Chat class constructor.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chat

Syntax

Chat(*event, address*)

Description

The Chat object describes the task information for tasks associated with the ChatAddress.

Parameters

event	The onNotify event.
address	The chat address.

Returns

A Chat object.

Chat Class Fields

The Chat class inherits the following fields from the Task class:

- caseid
- cost
- customerid
- group
- id
- onStat
- priority
- type

  The task type for chat that is Task.TYPE_CHAT.
- urlAbs
- urlRel

See Task Class Fields.

The following are the additional Chat class fields.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

address

Description

The address containing the agent-to-agent chat.

Type: A2AChatAddress object.

Related Links

A2AChatAddress Class Constructor

agentId

Description

The agent's ID.

Type: string.

appData

Description

The application data that is provided to the client with the notification event.

Type: AppData object.

Related Links

AppData Class Constructor

chatconnection

Description

The chat connection object associated with this call.

Type: line object

chatType

Description

Type of chat, for example consulting chat or answering chat

Type: string

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

customerName

Description

The customer username.

Type: string.

groupId

Description

The group ID.

Type: string.

question

Description

The question for this chat.

Type: string.

subject

Description

The subject of the chat.

Type: string.

statistics

Description

The chat statistics object that associated with this chat.

Type: string.

userData

Description

The user data associated with the chat.

Type: object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chat Class Method

The Chat class extends the Task class.

See Task Class Constructor.

The following is the Chat class method.

gettpUrl

Syntax

gettpUrl(*defaultUrl*)

Description

Returns the URL for the given task for a third-party routing server.

Parameters

defaultUrl	If not null, this value will override the generated base URL.

Returns

Returns the URL associated with this task.

getUrl

Syntax

getUrl(*defaultUrl*)

Description

Returns the URL for the given task.

Parameters

defaultUrl	The default URL for this chat. If not null this value will override the generated base URL.

Returns

Returns the URL for the chat.

ChatAddress Class Constructor

The ChatAddress class extends the _UQAddress class.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

See _UQAddress Class Constructor.

The following is the ChatAddress class constructor.

ChatAddress

Syntax

```
ChatAddress()
```

Description

Handles the creation of customer chat tasks.

Parameters

None.

Returns

A ChatAddress object.

ChatAddress Class Fields

The ChatAddress class inherits the following fields from the _Address class:

- id

  The address's ID is equal to Task.TYPE_CHAT.

- caps

See _Address Class Fields.

The ChatAddress class inherits the following field from the _UQAddress class:

tasks

See _UQAddress Class Fields.

The following are ChatAddress class fields.

chatconnections

Description

List of ChatConnection objects that are associated with chat address.

Type: chatconnection object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

ChatAddress Class Methods

The ChatAddress class inherits the following methods from the _UQAddress class:

- acceptTask
- dequeueTask

See _UQAddress Methods.

The following are ChatAddress class methods.

chat

Syntax

chat(*agentId*,*buddyId*,*reason*)

Description

Sends a request to initiate a new chat.

Parameters

agentId	The source agent ID.
buddyId	The Id of the buddy.
reason	The reason code for chat request.

Returns

Request number.

getChatconnectionByConnectionId

Syntax

getChatconnectionByConnectionId(*connectionId*)

Description

Get the chatconnection object with the chatconnection Id.

Parameters

connectionId	The connection id associated with a chat.

Returns

Line object, null if there is no connection with that chatconnection id.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

getChatconnectionindexByConnectionId

Syntax

```
getChatconnectionindexByConnectionId
(connectionId)
```

Description

Get the chatconnection index with the chatconnection Id.

Parameters

connectionId	The connection id associated with a chat.

Returns

index, −1 if there is no connection with that chatconnection id.

getFreeChatconnection

Syntax

```
getFreeChatconnection()
```

Description

Get the free chatconnection object. A connection is free if there is no activity on this line.

Returns

Returns chatconnection object, null if there is no free line.

getFreeChatconnectionIndex

Syntax

```
getFreeChatconnectionIndex()
```

Description

Get the free chatconnection index. A connection is free if there is no activity on this line.

Returns

Returns chatconnection index, null if there is no free line.

ChatAddress Callback Event Methods

The ChatAddress class inherits the onError callback event method from the _Address class.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

See _Address Class Callback Event Method.

The ChatAddress class inherits the following callback event methods from the _UQAddress class:

- onAccepted
- onAcceptingTask
- onDequeueingTask
- onNotify
- onTaskAdded
- onTaskRemoved
- onUnassigned

See _UQAddress Class Fields.

The following are ChatAddress callback event methods:

onCapabilitiesChanged

Syntax

onCapabilitiesChanged*event*

Description

Fires when ChatAddress capabilities change

ChatConnection Class Constructor

The following is the chatconnection class constructor.

ChatConnection

Syntax

ChatConnection()

Description

Chat connection object.

Parameters

None

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Returns

Returns a chatconnection object.

ChatConnection Class Fields

The following are chatconnection class fields:

caps

Description

The capabilities of the ChatConnection.

chat

Description

Chat task associated with the connection.

Type: string

connectionId

Description

The connection Id.

Type: string

id

Description

The constant representing ChatConnection Id.

Type: string

state

Description

The connection state.

Type: string

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Value	*Description*
ST_DIALING	Dialing state.
ST_DROPPED	Dropped state.
ST_IDLE	Idle state.
ST_INCOMING	Incoming state.
ST_TALKING	Talking state.
ST_WRAPUP	Wrap up state.

ChatConnection Class Methods

The following are chatconnection class methods:

answer

Syntax

answer(*agentId*,*reason*)

Description

Answer a chat request.

Parameters

agent Id The Id of the agent answering the call.

reason The reason code for the answer request.

Returns

Request number.

attachUserData

Syntax

attachUserData(*userData*,*reason*)

Description

Attach user data for the chat connection.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

userData	User data of the chat connection.
reason	The reason code for the answer request.

Returns

Request number.

conference

Syntax

```
conference(agentId,reason)
```

Description

Conference a chat with another agent.

Parameters

agent Id	The Id of the agent invited to the conference.
reason	The reason code for the conference request.

Returns

Request number.

forward

Syntax

```
forward(fromagentId,toagentId, qid, userdata, chatdata, reason)
```

Description

Forward chat to another agent or another queue.

Parameters

fromagentid	The agent id from whom task is forwarded.
toagentid	The target agentid to whom task is forwarded.
qid	The target queue to forward the task.
userdata	The user data.
chatdata	The chat data.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Reason	The reason code to forward the task.

Returns

Request number.

gethistory

Syntax

```
gethistory(agentId,noofLines, reason)
```

Description

Request chat history by specifying number of lines of chat history needed.

Parameters

agent Id	The source agent ID.
noofLines	Number of lines of history required.
reason	The reason code for the history request.

Returns

Request number.

getUrl

Syntax

```
getUrl(defaultURL)
```

Description

Get the URL for screen popup.

Parameters

defaultUrl	The default popup URL.

Returns

Returns a URL string.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

message

Syntax

message(*agentId*,*message*, *reason*)

Description

Send a chat message while chatting.

Parameters

agent Id	The source agent ID.
message	The message to send to other party.
reason	The reason code for the message request.

Returns

Request number.

pushURL

Syntax

pushURL(*URL*)

Description

Request to push a URL to the customer.

Parameters

URL	URL to push.

Returns

Request number.

reject

Syntax

reject(*agentId*,*reason*)

Description

Reject a chat.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

agent Id	The source agent ID.
reason	The reason code for reject.

Returns

Request number.

release

Syntax

```
release(agentId,reason)
```

Description

Release a chat.

Parameters

agent Id	The source agent ID.
reason	The reason code for release request.

Returns

Request number.

typing

Syntax

```
typing(agentId,reason)
```

Description

Indicates typing in a chat conversation by one party.

Parameters

agent Id	The source agent ID.
reason	The reason code for typing.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

wrapup

Syntax

wrapup(*agentid*,*message*,*tasknum*,*reason*)

Description

Stores chat wrap up comments.

Parameters

agentid	The agent id who enters wrap up comments.
message	Chat message.
tasknum	Task number.
reason	The reason code.

Returns

Request number.

ChatConnection Class Callback Event Methods

The following are ChatConnection class callback event methods.

onAccepted

Syntax

onAccepted(*event*)

Description

Fires when chat is accepted by a agent.

onAnswering

Syntax

onAnswering(*event*)

Description

Fires when answering the chat.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onCapabilitiesChanged

Syntax

onCapabilitiesChanged(*event*)

Description

Fires when chat capabilities change.

onChatdataChanged

Syntax

onChatdataChanged(*event*)

Description

Fires when there is a change in chat data.

onConferencing

Syntax

onConferencing(*event*)

Description

Fires when conferencing the chat.

onDialing

Syntax

onDialing(*event*)

Description

Fires when chat is in the process of connecting.

onDropped

Syntax

onDropped(*event*)

Description

Fires when chat is dropped.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onError

Syntax

onError(*event*)

Description

Fires when a there is a chat connection error.

onForwarded

Syntax

onForwarded(*event*)

Description

Fires when chat is forwarded.

onForwardError

Syntax

onForwardError(*event*)

Description

Fires when there is error in forwarding the task.

onForwarding

Syntax

onForwarding(*event*)

Description

Fires when forwarding the chat.

onHistory

Syntax

onHistory(*event*)

Description

Fires when agent receives chat history.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onIncomingChat

Syntax

onIncomingChat(*event*)

Description

Fires when ChatConnection state is changed to INCOMING.

onMessage

Syntax

onMessage(*event*)

Description

Fires when there is incoming message.

onPartyAdded

Syntax

onPartyAdded(*event*)

Description

Fires when a new chat party is added.

onPartyChanged

Syntax

onPartyChanged(*event*)

Description

Fires when the chat party changes.

onPartyRemoved

Syntax

onPartyRemoved(*event*)

Description

Fires when a chat party is removed.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onProperties

Syntax

onProperties(*event*)

Description

Fires when a agent receives chat properties.

onPushURL

Syntax

onPushURL(*event*)

Description

Fires when a routing system push URL.

onRejected

Syntax

onRejected(*event*)

Description

Fires when a chat is rejected.

onReleased

Syntax

onReleased(*event*)

Description

Fires when a chat is released.

onReleasing

Syntax

onReleasing(*event*)

Description

Fires when conferencing the chat.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onRevoked

Syntax

onRevoked(*event*)

Description

Fires when a chat is revoked.

onTalking

Syntax

onTalking(*event*)

Description

Fires when an agent/customer is in the state of talking.

onTyping

Syntax

onTyping(*event*)

Description

Fires when typing in a chat conversation.

onUserdataChanged

Syntax

onUserdataChanged(*event*)

Description

Fires when there is a change in user data.

ChatConnectionCaps Class Constructor

The following is the chatconnectioncaps class constructor.

ChatConnectionCaps

Syntax

ChatConnectionCaps(*strCaps*)

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Describes the ChatConnection capabilities.

Parameters

strCaps	Chat connection capability.

Returns

Returns a chatconnection object.

ChatConnectionCaps Class Fields

The following are the ChatConnectionCaps class fields.

canAnswer

Description

Answer capability.

Type: boolean

canConference

Description

Conference capability.

Type: boolean

canConferenceSingle

Description

Conference single capability.

Type: boolean

canForward

Description

Forward capability.

Type: boolean

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

canIndicateTyping

Description

Indicate Typing capability.

Type: boolean

canPushURL

Description

PushURL capability.

Type: boolean

canReject

Description

Reject capability.

Type: boolean

canSendMessage

Description

SendMessage capability.

Type: boolean

ChatData Class Constructor

The following is ChatData class constructor.

ChatData

Syntax

```
ChatData()
```

Description

The ChatData object describes the key-value pairs of the chat data with the chat object.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

ChatData Class Fields

The following is ChatData class field.

data

Description

Key value pairs that includes data.

Type: object.

ChatData Class Methods

The following is ChatData class method.

addKeyValue

Syntax

addKeyValue(*key, value*)

Description

Add key value to the chatdata object.

Parameters

key	The variable name.
value	The value of the variable.

Email Class Constructor

The Email class extends the Task class.

See Task Class Constructor.

The following is the email class constructor.

Email

Syntax

Email(*event*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

The Email object describes the task information for tasks associated with the EmailAddress.

Parameters

event	The notification event.

Returns

Returns an Email object.

Email Class Fields

The Chat class inherits the following fields from Task class:

- caseid
- cost
- customerid
- group
- id
- onStat
- priority
- type

  The task type is equal to Task.TYPE_EMAIL.
- urlAbs
- urlRel

See Task Class Fields.

The following are the additional Email class fields.

address

Description

The address containing the A2AChat.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

agentId

Description

The agent's ID.

appData

Description

The application data that is provided to the client with the notification event.

Type: AppData object.

Related Links

Task Class Fields

customerName

Description

The customer username.

Type: string.

emailconnection

Description

The emailconnection object that associated with this email.

Type: line object.

emailId

Description

The unique email id generated from an Enqueue. It identifies the email stored in the database ie. it serves as a key to the data stored in the database.

Type: string.

groupId

Description

The group id.

Type: string.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

question

Description

The question for this email.

Type: string.

statistics

Description

The statistics object that is associated with this email.

Type: object.

subject

Description

The subject of this email.

Type: string.

userData

Description

The user data that is associated with this email.

Type: object.

Email Class Method

The following are the Email class methods:

gettpUrl

Syntax

gettpUrl(*defaultUrl*)

Description

Returns the URL for the given task for third-party routing.

Parameters

defaultUrl	If not null, this value will override the generated base URL.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Returns

The URL associated with this task.

getUrl

Syntax

```
getURL(defaultUrl)
```

Description

Returns the URL for the given task.

Parameters

defaultUrl	If not null, this value will override the generated base URL.

Returns

The URL associated with this task.

EmailAddress Class Constructor

The EmailAddress class extends the _UQAddress class.

See _UQAddress Class Constructor.

The following is the EmailAddress class constructor.

EmailAddress

Syntax

```
EmailAddress()
```

Description

Handles the creation of email tasks.

Parameters

None.

Returns

An EmailAddress object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

EmailAddress Class Fields

The EmailAddress class inherits the following fields from the _Address class:

- id

  The address's ID is equal to Task.TYPE_EMAIL.

- caps

See _Address Class Fields.

The EmailAddress class inherits the tasks field from the _UQAddress class.

See _UQAddress Class Fields.

The following are EmailAddress class fields.

agent

Description

The agent id of this email address.

emailconnections

Description

Describes the EmailConnection object.

EmailAddress Class Methods

The EmailAddress class inherits the following fields from the _UQAddress class:

- acceptTask
- dequeueTask

See _UQAddress Methods.

getEmailconnectionByConnectionId

Syntax

getEmailconnectionByConnectionId(*connectionId*)

Description

Get the emailconnection object with the emailconnection id.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

connectionId	The connection id associated with a email.

Returns

Returns line object, null if there is no connection with that emailconnection id.

getEmailconnectionindexByConnectionId

Syntax

getEmailconnectionindexByConnectionId(*connectionId*)

Description

Get the emailconnection index with the emailconnection id.

Parameters

connectionId	The connection id associated with a email.

Returns

Returns index, -1 if there is no connection with that emailconnection id.

getFreeEmailconnection

Syntax

getFreeEmailconnection()

Description

Get the free email connection object. A connection is free if there is no activity on this line.

Returns

Returns emailconnection object, null if there is no free line.

EmailAddress Callback Event Methods

The EmailAddress class inherits the onError callback event method from the _Address class.

See _Address Class Callback Event Method.

The EmailAddress class inherits the following callback event methods from the _UQAddress class:

- onAccepted

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- onAcceptingTask
- onDequeueingTask
- onNotify
- onTaskAdded
- onTaskRemoved
- onUnassigned

See _UQAddress Class Callback Event Methods.

EmailConnection Class Constructor

The following is EmailConnection class constructor.

EmailConnection

Syntax

```
EmailConnection()
```

Description

Handles email connection.

EmailConnection Class Fields

The following are EmailConnection class fields.

caps

Description

The capabilities of the EmailConnection.

connectionId

Description

The email connection Id.

Type: string

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

email

Description

Email task id associated with this email connection.

Type: string with following constants:

id

Description

The id of the EmailConnection.

Type: string

state

Description

The connection state.

Type: string

Value	*Description*
ST_DIALING	Dialing state.
ST_DROPPED	Dropped state.
ST_IDLE	Idle state.
ST_INCOMING	Incoming state.
ST_TALKING	Talking state.

EmailConnection Class Methods

The following are the EmailConnection class methods:

abandon

Syntax

abandon(*reason*)

Description

Abandon email task without completing the task .

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

Reason	The reason code to abandon.

Returns

Request number.

answer

Syntax

answer(*reason*)

Description

Answer/Accept the email assignment.

Parameters

Reason	The reason code to answer.

Returns

Request number.

attachUserData

Syntax

attachUserData(*userData*,*reason*)

Description

Attach user data for the email connection.

Parameters

userData	User data of the email connection.
reason	The reason code for the answer request.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

complete

Syntax

complete(*reason*)

Description

Send task completion notification.

Parameters

Reason	The reason code to complete.

Returns

Request number.

forward

Syntax

forward(*fromagentId*,*toagentId*, *qid*, *userdata*, *emaildata*, *reason*)

Description

Forward email to another agent or to another queue.

Parameters

fromagentid	The agent id from whom task is forwarded.
toagentid	The target agent id to whom task is forwarded.
qid	The target queue to forward the task.
userdata	The user data.
emaildata	The email data.
Reason	The reason code to forward the task.

Returns

Request number.

reject

Syntax

reject(*reason*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Reject email assignment.

Parameters

Reason	The reason code to reject.

Returns

Request number.

withdraw_RES

Syntax

```
withdraw_RES
(reason)
```

Description

Send response for the request of (Withdraw of Email) by routing server. This is the response from the agent to the routing system.

Parameters

Reason	The reason code to withdraw.

Returns

Request number.

EmailConnection Class Callback Event Methods

The following are EmailConnection class callback event methods:

onAnswering

Syntax

```
onAnswering(event)
```

Description

Fires when email task is getting answered.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onCapabilitiesChanged

Syntax

onCapabilitiesChanged(*event*)

Description

Fires when there is a change in connection capabilities.

onCompleted

Syntax

onCompleted(*event*)

Description

Fires when EmailConnection state is changed to COMPLETED.

onDropped

Syntax

onDropped(*event*)

Description

Fires when EmailConnection state is changed to DROPPED.

onEmaildataChanged

Syntax

onEmaildataChanged(*event*)

Description

Fires when there is a change in email data.

onError

Syntax

onError(*event*)

Description

Fires when a there is a email connection error.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onForwarded

Syntax

onForwarded(*event*)

Description

Fires when email is forwarded to another queue or to another agent.

onForwardError

Syntax

onForwardError(*event*)

Description

Fires when there is error in forwarding the task.

onForwarding

Syntax

onForwarding(*event*)

Description

Fires when forwarding the email.

onIncoming

Syntax

onIncoming(*event*)

Description

Fires when EmailConnection state is changed to INCOMING.

onProcessing

Syntax

onProcessing(*event*)

Description

Fires when EmailConnection state is changed to PROCESSING.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onRejected

Syntax

onRejected(*event*)

Description

Fires when email task is rejected by agent and the routing system acknowledges the same.

onRevoked

Syntax

onRevoked(*event*)

Description

Fires when email is revoked by routing system before agent accepts that.

onUserdataChanged

Syntax

onUserdataChanged(*event*)

Description

Fires when there is a change in user data.

onWithdraw_REQ

Syntax

onWithdraw_REQ(*event*)

Description

Fires when there is a request for email withdraw from routing server. This is a request from routing server to the agent.

EmailConnectionCaps Class Constructor

The following is EmailConnectionCaps class constructor:

EmailConnectionCaps

Syntax

EmailConnectionCaps(*strCaps*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Describes the EmailConnection capabilities.

Parameters

strCaps	The capability of the connection

Returns

Returns EmailConnection object.

EmailConnectionCaps Class Fields

The following are the EmailConnectionCaps class fields.

canAnswer

Description

Answer/Accept an email capability.

Type: boolean

canComplete

Description

Capability to complete an assigned email.

Type: boolean

canForward

Description

Capability to forward an email.

Type: boolean

canReject

Description

Capability to reject an email.

Type: boolean

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

EmailData Class Constructor

The following is EmailConnectionCaps class constructor.

EmailData

Syntax

EmailData()

Description

The EmailData object describes the key-value pairs of the email data with the Email object.

Returns

Returns EmailData object.

EmailData Class Fields

The following is EmailData class field.

data

Description

Key value pairs that includes email data.

Type: object.

EmailData Class Methods

The following is EmailData class method.

addKeyValue

Syntax

addKeyValue(*key, value*)

Description

Add key value to the emaildata object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

key	The variable name.
value	The value of the variable.

Extension Class Constructor

The Extension class extends the _Address class.

See _Address Class Constructor.

The following is the Extension class constructor.

Extension

Syntax

```
Extension(numOfLines)
```

Description

The Extension object describes the CTI address.

Parameters

numOfLines	Enter the number of lines for this extension.

Returns

An Extension object.

Extension Class Fields

The Extension class inherits the following fields from the _Address class:

- caps
- id

See _Address Class Fields.

The following are the Extension class fields.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

forwardMode

Description

The forward mode of the extension.

Type: forwardMode object.

Related Links

ForwardMode Class Constructor

isDnd

Description

Flag for Do Not Disturb.

Type: boolean.

lines

Description

List of line objects associated with this extension.

Type: list.

numOfLines

Description

The number of lines for this extension.

Type: number.

Extension Class Methods

The Extension class inherits the onError method from the Address class.

The following are the Extension class methods.

cancelDnd

Syntax

cancelDnd(*reason*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Cancel the DND (do not disturb).

Parameters

reason	The reason for DND cancellation.

Returns

Request number.

cancelForwardSet

Syntax

`cancelForwardSet``(`*`reason`*`)`

Description

Cancel the forward.

Parameters

reason	The reason for cancelling the forward.

Returns

Request number.

forwardSet

Syntax

`forwardSet``(`*`number`*`, `*`mode`*`, `*`reason`*`)`

Description

Forward the call to another number/extension.

Parameters

mode	The forward mode.
number	The number to which all calls will be forwarded.
reason	The reason to forward.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

getDialingLine

Syntax

getDialingLine()

Description

Get the dialing line object.

Parameters

None

Returns

Returns a line object. Null if there is no dialing line.

Related Links

Line Class Constructor

getFreeLine

Syntax

getFreeLine()

Description

Get the free line object. A line is free if there is no activity on this line.

Parameters

None

Returns

Returns a line object. It returns null if there is no free line.

getLineByConnectionId

Syntax

getLineByConnectionId(*connectionID*)

Description

Get the line object with the call.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

connectionId	The connection ID associated with a call.

Returns

Returns a Line object. Returns null if there is no line with that call.

Related Links

Line Class Constructor

getOffHookLine

Syntax

getOffHookLine()

Description

Get the offhook line object.

Parameters

None.

Returns

Returns a Line object. Returns null if there is no offhook line.

Related Links

Line Class Constructor

setDnd

Syntax

setDnd(*reason*)

Description

Fires when setting DND (do not disturb).

Parameters

reason	The reason to set the DND.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Extension Class Callback Event Methods

The following are the Extension class callback event methods.

onCancelingDnd

Syntax

onCancelingDnd(*event*)

Description

Fires when canceling DND (do not disturb).

onCancelingForward

Syntax

onCancelingForward(*event*)

Description

Fires when canceling forward.

onDnd

Syntax

onDnd(*event*)

Description

Fires when DND (Do Not Disturb) is requested and processed.

onDndCanceled

Syntax

onDndCanceled(*event*)

Description

Fires when DND is cancelled.

Parameters

event	The event object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Returns

None.

onForwardCanceled

Syntax

onForwardCanceled(*event*)

Description

Fires when forward is canceled.

onForwarded

Syntax

onForwarded(*event*)

Description

Fires when call is forwarded.

onForwarding

Syntax

onForwarding(*event*)

Description

Fires when forwarding the call.

onSettingDnd

Syntax

onSettingDnd(*event*)

Description

Fires when setting DND (do not disturb).

ExtensionCaps Class Constructor

The following is the Extension class constructor.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

ExtensionCaps

Syntax

ExtensionCaps(*strCaps*)

Description

Describes the extension's capabilities.

Parameters

strCaps	A string comprising the extension's capabilities.

Returns

An ExtensionCaps object.

ExtensionCaps Class Fields

The following are the Extension class fields.

canCancelDnd

Description

This extension can cancel DND (do not disturb).

Type: boolean.

canDial

Description

This extension can dial out.

Type: boolean.

canFwdBusy

Description

This extension can forward calls if busy.

Type: boolean.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

canFwdBusyNoAnswer

Description

This extension can forward calls if busy/no answer.

Type: boolean.

canFwdCancelForward

Description

This extension can cancel forward.

Type: boolean.

canFwdDefault

Description

This extension can forward.

Type: boolean.

canFwdNoAnswer

Description

This extension can forward if no answer.

Type: boolean.

canFwdUnconditional

Description

This extension can forward unconditionally.

Type: boolean.

canRefreshState

Description

This extension can refresh state.

Type: boolean.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

canSetDnd

Description

This extension can set DND (do not disturb).

Type: boolean.

ForwardMode Class Constructor

The following is the ForwardMode class constructor.

ForwardMode

Syntax

```
ForwardMode()
```

Description

Describes various forward modes that can be used while setting the forwarding mode for an Address.

Parameters

None.

Returns

A ForwardMode object.

ForwardMode Class Field

The following is the ForwardMode class field.

mode

Description

The current forwarding mode.

Type: string of the following constants.

Value	*Description*
BUSY	The BUSY forwarding mode.
BUSYNOANSWER	The BUSYNOANSWER forwarding mode.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Value	*Description*
DEFAULT	The DEFAULT forwarding mode.
NOANSWER	The NOANSWER forwarding mode.
NONE	No forwarding mode.
UNCONDITIONAL	The UNCONDITIONAL forwarding mode.

GenericAddress Class Constructor

The GenericAddress class extends the _UQAddress class.

The following is the GenericAddress class constructor.

GenericAddress

Syntax

```
GenericAddress()
```

Description

Handles the creation of generic tasks.

Parameters

None.

Returns

A GenericAddress object.

GenericAddress Class Fields

The GenericAddress class inherits the following fields from the _Address class.

- caps
- id

  The address's ID is equal to Task.TYPE_GENERIC.

See _Address Class Fields.

The GenericAddress class inherits the tasks field from the _UQAddress class.

See _UQAddress Class Fields.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The following are GenericAddress class fields:

agent

Description

The agent id of this generic address owner.

Type: object

genericconnections

Description

List of GenericConnections.

Type: genericconnectionobject

GenericAddress Class Methods

The GenericAddress class inherits the following methods from the _UQAddress class:

- acceptTask
- dequeueTask

See _UQAddress Methods.

The following are GenericAddress own class methods.

getFreeGenericconnection

Syntax

```
getFreeGenericconnectio()
```

Description

Get the free generic connection object. A connection is free if there is no activity on this line.

Parameters

None.

Returns

Returns genericconnection object, null if there is no free line.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

getGenericconnectionByConnectionId

Syntax

getFreeGenericconnectio(*connectionId*)

Description

Get the genericconnection object with the genericconnection id.

Parameters

connectionId	The connection id associated with a generic.

Returns

Returns line object, null if there is no connection with that genericconnection id.

getGenericconnectionindexByConnectionId

Syntax

getGenericconnectionindexByConnectionId(*connectionId*)

Description

Get the genericconnection index with the genericconnection id.

Parameters

connectionId	The connection id associated with a generic.

Returns

Returns index, -1 if there is no connection with that genericconnection id.

GenericAddress Class Callback Event Methods

The GenericAddress class inherits the onError callback event method from the _Address class.

See _Address Class Callback Event Method.

The GenericAddress class inherits the following callback event methods from the _UQAddress class:

- onAcceptingTask
- onDequeueingTask
- onTaskAdded
- onTaskRemoved

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- onNotify
- onAccepted
- onUnassigned

See _UQAddress Class Callback Event Methods.

GenericConnection Class Constructor

The following is the GenericConnection class constructor.

GenericConnection

Syntax

```
GenericConnection()
```

Description

Handles Generic connection.

Parameters

None.

Returns

A genericconnection object.

GenericConnection Class Fields

The following are GenericConnection class fields:

caps

Description

The capabilities of the GenericConnection.

connectionId

Description

The generic connection Id.

Type: string

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

generic

Description

The generic task id associated with this generic connection.

Type: string

id

Description

The id of the GenericConnection.

Type: string

state

Description

The connection state.

Type: string with following constants:

Value	*Description*
ST_DIALING	Dialing state.
ST_DROPPED	Dropped state.
ST_IDLE	Idle state.
ST_INCOMING	Incoming state.
ST_TALKING	Talking state.

GenericConnection Class Methods

The following are the GenericConnection class methods:

abandon

Syntax

abandon(*reason*)

Description

Abandon generic task.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

Reason	The reason code to abandon.

Returns

Request number.

answer

Syntax

answer(*reason*)

Description

Answer/Accept the generic task assignment.

Parameters

Reason	The reason code to answer.

Returns

Request number.

attachUserData

Syntax

attachUserData(*userData*,*reason*)

Description

Attach user data for the generic connection.

Parameters

userData	User data of the generic connection.
reason	The reason code for the answer request.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

complete

Syntax

complete(*reason*)

Description

Send task completion notification.

Parameters

Reason	The reason code to complete.

Returns

Request number.

forward

Syntax

forward(*fromagentId*,*toagentId*, *qid*, *userdata*, *genericdata*, *reason*)

Description

Forward generic task to another Agent or to another queue.

Parameters

fromagentid	The agent id from whom task is forwarded.
toagentid	The target agent id to whom task is forwarded.
qid	The target queue to forward the task.
userdata	The user data.
emaildata	The email data.
Reason	The reason code to forward the task.

Returns

Request number.

reject

Syntax

reject(*reason*)

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Reject generic task assignment.

Parameters

Reason	The reason code to reject.

Returns

Request number.

withdraw_RES

Syntax

```
withdraw_RES
(reason)
```

Description

Send response for the request of (Withdraw of Generic task) by routing server. This is the response from the agent to the routing system.

Parameters

Reason	The reason code to withdraw.

Returns

Request number.

GenericConnection Class Callback Event Methods

The following are ChatConnection class callback event methods:

onCapabilitiesChanged

Syntax

```
onCapabilitiesChanged(event)
```

Description

Fires when there is a change in GenericData.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onCompleted

Syntax

onCompleted(*event*)

Description

Fires when GenericConnection state is changed to COMPLETED.

onDropped

Syntax

onDropped(*event*)

Description

Fires when GenericConnection state is changed to DROPPED.

onError

Syntax

onError(*event*)

Description

Fires when a there is a generic connection error.

onForwarded

Syntax

onForwarded(*event*)

Description

Fires when generic task is forwarded.

onForwardError

Syntax

onForwardError(*event*)

Description

Fires when there is error in forwarding the generic task.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onForwarding

Syntax

onForwarding(*event*)

Description

Fires when forwarding the generic task.

onGenericdataChanged

Syntax

onGenericdataChanged(*event*)

Description

Fires when there is a change in GenericData.

onIncoming

Syntax

onIncoming(*event*)

Description

Fires when GenericConnection state is changed to INCOMING.

onProcessing

Syntax

onProcessing(*event*)

Description

Fires when GenericConnection state is changed to PROCESSING.

onRejected

Syntax

onRejected(*event*)

Description

Fires when generic task is rejected by agent and the routing system acknowledges the same.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onRevoked

Syntax

onRevoked(*event*)

Description

Fires when generic task is revoked by routing system before agent accepts that.

onUserdataChanged

Syntax

onUserdataChanged(*event*)

Description

Fires when there is a change in user data.

onWithdraw_REQ

Syntax

onWithdraw_REQ(*event*)

Description

Fires when there is a request for generic task withdraw from routing server. This is a request from routing server to the agent.

GenericConnectionCaps Class Constructor

The following is GenericConnectionCaps class constructor.

GenericConnectionCaps

Syntax

GenericConnectionCaps(*strCaps*)

Description

Describes the GenericConnection capabilities.

Parameters

strCaps	The capability of the connection.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Returns

Returns GenericConnection object.

GenericConnectionCaps Class Fields

The following are the GenericConnectionCaps class fields

canAnswer

Description

Answer/Accept a generic task capability.

Type: boolean

canComplete

Description

Complete an assigned generic task capability.

Type: boolean

canForward

Description

Forward a generic task capability.

Type: boolean

canReject

Description

Reject a generic task capability.

Type: boolean

GenericData Class Constructor

The following is GenericConnectionCaps class constructor:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

GenericData

Syntax

```
GenericData()
```

Description

The GenericData object describes the key-value pairs of the generic data with the generic object

Returns

Returns GenericData object.

GenericData Class Fields

The following is GenericData class field:

data

Description

Key value pairs that includes data.

Type: object.

GenericData Class Methods

The following is GenericData class method:

addKeyValue

Syntax

```
addKeyValue(key, value )
```

Description

Add key value to the genericdata object.

Parameters

key	The variable name.
value	The value of the variable.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

GenericTask Class Constructor

The GenericTask class extends the Task class.

The following is the GenericTask class constructor

GenericTask

Syntax

GenericTask(*event*)

Description

The GenericTask object describes the generic task information for tasks associated with the GenericAddress object.

Parameters

event	The triggering event.

Returns

A GenericTask object.

GenericTask Class Fields

The GenericTask class inherits the following fields from the Task class:

- caseid
- cost
- customerid
- group
- id
- onStat
- priority
- type

  The task type for generic task that is Task.TYPE_GENERIC.
- urlAbs
- urlRel

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

See Task Class Fields.

The following are the additional GenericTask class fields.

address

Description

The address containing the A2AChat.

agentId

Description

The agent id to whom this task is assigned.

Returns a string.

appData

Description

The application data that is provided to the client with the notification event.

Type: AppData object.

Related Links

AppData Class Constructor

customerName

Description

The customer username.

Type: string.

genericconnection

Description

The generic connection object that associated with this Generic task.

Type: line

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

genericId

Description

The unique generic id generated from an Enqueue. It identifies the data in the database associated with the generic task.

Type: string.

groupId

Description

The group id for which task is initiated.

Type: string.

question

Description

The question of this generic task which is raised by customer.

Type: string.

statistics

Description

Task statistics of the generic task.

Type: object.

subject

Description

The subject of the generic task.

Type: string.

userdata

Description

User data object that is associated with this generic task.

Type: object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

GenericTask Class Method

The following is the GenericTask class method.

gettpUrl

Syntax

gettpUrl(*defaultUrl*)

Description

Returns the URL for the given task for the third-party routing.

Parameters

defaultUrl	If not null, this value will override the generated base URL.

Returns

Returns the URL for the given task.

getUrl

Syntax

getUrl(*defaultUrl*)

Description

Returns the URL for the given task.

Parameters

defaultUrl	If not null, this value will override the generated base URL.

Returns

Returns the URL for the given task.

GLOBALS Class Fields

The following are global fields, which may be accessed without instantiating an object. They are accessed as shown.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

A2AChat.PS_JR

Description

The Journal Routing constant.

Example

A2AChat.PS_JR = "ps_jr";

A2AChat.TYPE_ANSWER

Description

The constant representing an A2AChat type of answer.

This type is generated when a different user wants this user to answer an A2AChat.

Example

A2AChat.TYPE_ANSWER = "answer";

A2AChat.TYPE_CONSULT

Description

The constant representing an A2AChat type of consult.

This type is generated when this user wants to consult a different user.

Example

A2AChat.TYPE_CONSULT = "consult";

Server.TYPE_CTI

Description

The constant representing a CTI server.

Example

Server.TYPE_CTI = "CTI";

Server.TYPE_UQ

Description

The UQ server type.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Example

Server.TYPE_UQ = "UQ";

Task.TYPE_A2ACHAT

Description

The constant representing an A2AChat.

Example

Task.TYPE_A2ACHAT = "A2ACHAT";

Task.TYPE_CHAT

Description

The constant representing a chat task.

Example

Task.TYPE_CHAT = "CHAT";

Task.TYPE_CTI

Description

The constant representing a CTI task.

Example

Task.TYPE_CTI = "CTI";

Task.TYPE_EMAIL

Description

The constant representing an email task.

Example

Task.TYPE_EMAIL = "EMAIL";

Task.TYPE_GENERIC

Description

The constant representing a generic task.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Example

Task.TYPE_GENERIC = "GENERIC";

GLOBALS Class Methods

The following are the GLOBALS class methods.

initJSMCAPI

Syntax

initJSMCAPI()

Description

The initialization function of the JSMCAPI. This is the first function should be called by application.

Parameters

None.

Returns

None. Initializes JSMCAPI.

isValid

Syntax

isValid(*obj*)

Description

Returns true if an object is not undefined and not null. *Obj* can be any JavaScript object or literal. This method is a convenience method to check that an object is both not null and not undefined.

Parameters

obj	*Obj* can be any JavaScript object or literal.

Returns

Returns a boolean.

Returns true if an object is not undefined and not null.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

MCFBroadcast

Syntax

```
MCFBroadcast(cluster, queue, task, state, presence, message, securitylevel,
importancelevel, senderid, NameValuePairString)
```

Description

Broadcast a message to any queue, cluster, or only agents.

Parameters

cluster	Cluster Id to which we need to send broadcast message.
queue	Denotes the queue Id in the cluster.
task	Denotes the task, such as email, chat, voice, or generic.
state	Denotes the state, such as LoggedIn and NotLoggedIn.
presence	Denotes whether the agent is ready or not ready.
Message	Enter the message to broadcast.
security level	Security level defined by application developers.
importance level	Importance level defined by application developers.
sender Id	The sender's user id
namevaluepairs	Any extra data formed as name-value pair string.

Returns

None

Group Class Constructor

The following is the Group class constructor.

Group

Syntax

```
Group()
```

Description

The Group object describes the group information.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

None.

Returns

A Group object.

Group Class Fields

The following are the Group class fields.

id

Description

The group ID.

Type: string.

name

Description

The group name.

Type: string.

registered

Description

True if the Group is registered on the server.

Type: boolean.

statistics

Description

The group statistics for CTI.

Type: GroupStatistics object

Related Links

GroupStatistics1 Class Constructor

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

statistics1

Description

The group statistics for the queue server.

Type: GroupStatistics1 object.

Related Links

GroupStatistics1 Class Constructor

statistics2

Description

The group statistics for the queue server.

Type: GroupStatistics2 object.

Related Links

GroupStatistics2 Class Constructor

Group Class Callback Event Methods

The following are the Group class callback event methods.

onStat

Syntax

onStat(*event*)

Description

Fires when there is new statistics going to this group.

onStat1

Syntax

onStat1(*event*)

Description

Fires when statistics1 is received.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onStat2

Syntax

onStat2(*event*)

Description

Fires when statistics2 is received.

onTaskAdded

Syntax

onTaskAdded(*event*)

Description

Fires when a task added to this group.

onTaskRemoved

Syntax

onTaskRemoved(*event*)

Description

Fires when a task is removed from this group.

GroupStatistics Constructor

The following is the GroupStatistics class constructor.

GroupStatistics

Syntax

GroupStatistics()

Description

The group statistics information.

Parameters

None.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Returns

A GroupStatistics object.

GroupStatistics Fields

The following are the GroupStatistics class fields.

data

Description

Key value pairs that include all statistics.

listOfTasksInTheQueueByTaskType

Description

Denotes the task type, task state and the time for which the task is in the system.

maxTaskCompletionTime

Description

Longest wait time for a task in the queue.

newestTask

Description

Time elapsed for the most recent task.

newestTaskCompletionTime

Description

Difference between queue time and the time when task is done.

numberOfAbandoned

Description

Number of tasks that are abandoned.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

numberOfLoggedIn

Description

Number of agents that are logged in the queue.

numberOfQueued

Description

Number of tasks that are queued.

numUnassignedTasks

Description

Number of unassigned tasks.

queuedWaitTime

Description

The average wait time, in seconds, of a queued task.

queueUpTime

Description

Time since the queue is available on the system (startup/boot time).

relativeQueueLoad

Description

Relative queue load.

timeElapsedOldestTask

Description

Time elapsed for the oldest task (difference between current time and en-queue time).

GroupStatistics1 Class Constructor

The following is the GroupStatistics1 class constructor.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

GroupStatistics1

Syntax

GroupStatistics1(*data*)

Description

UQ Group statistics sent on group refresh 1.

Parameters

data	Statistical data from the UQ server.

Returns

A GroupStatistics1 object.

GroupStatistics1 Class Fields

The following are the GroupStatistics1 class fields.

mostRecentTaskDone

Description

The most recently done task in the group.

Type: string.

mostRecentTaskDoneData

Description

The data for the most recent task done.

Type: string.

mostRecentTaskEnqueued

Description

The most recently enqueued task in the group.

Type: string.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

mostRecentTaskEnqueuedData

Description

The data for the most recently enqueued task.

Type: string.

numAgentsAvailable

Description

Number of agents available in the group.

Type: string.

numAgentsLoggedIn

Description

Number of agents logged in on the group.

Type: string.

numEscalation

Description

Number of escalated tasks in the group.

Type: string.

numOverflow

Description

Number of overflowed tasks in the group.

Type: string.

numTaskAccepted

Description

Number of tasks accepted in the group.

Type: string.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

numTaskDone

Description

Number of tasks done in the group.

Type: string.

numTaskQueued

Description

Number of tasks queued in the group.

Type: string.

reasonFlag

Description

The reason flag for this GroupStatistics event.

Type: string.

taskTotalTimeInSystem

Description

The total time in the system.

Type: string.

timeSinceStart

Description

The time since start.

Type: string.

GroupStatistics2 Class Constructor

The following is the GroupStatistics2 class constructor.

GroupStatistics2

Syntax

`GroupStatistics2``(`*`data`*`)`

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

UQ Group statistics sent on group refresh 2.

Parameters

data	UQ server statistical data.

Returns

A GroupStatistics2 object.

GroupStatistics2 Class Fields

The following are the GroupStatistics2 class fields.

averageTaskDuration

Description

The average task duration in the group.

Type: string.

averageWaitTime

Description

The average wait time in the group.

Type: string.

oldestTask

Description

The oldest task in the group.

Type: string.

recentTask

Description

The most recent task in the group.

Type: string.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

timeElapsedOldestTask

Description

The time elapsed for the oldest task in the group.

Type: string.

timeElapsedRecentTask

Description

The time elapsed for the most recent task in the group.

Line Class Constructor

The following is the Line class constructor.

Line

Syntax

```
Line()
```

Description

The Line object describes the line of the extension. JSMCAPI only supports one extension with two lines and two extensions with one line in each.

Parameters

None.

Returns

A Line object.

Line Class Fields

The following are the Line class fields.

call

Description

The Call object on the line.

Type: Call object.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Related Links

Call Class Constructor

caps

Description

The capabilities of the line.

Type: LineCaps object.

Related Links

LineCaps Class Constructor

connectionid

Description

The call connection ID on the line.

Type: string.

id

Description

The line ID.

Type: string.

isMuted

Description

Flag to see whether extension is muted or not.

Type: string.

state

Description

The line state.

Type: string with the following constants:

Value	*Description*
ST_DIALING	The dialing state.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Value	*Description*
ST_DROPPED	The dropped state.
ST_HELD	The held state.
ST_IDLE	The idle state.
ST_OFFHOOK	The offhook state.
ST_RINGING	The ringing state.
ST_TALKING	The talking state.

Line Class Methods

The following are the Line class methods.

alternate

Syntax

alternate(*reason*)

Description

Alternate a call.

Parameters

reason The reason to alternate a call.

Returns

Request number.

answer

Syntax

answer(*reason*)

Description

Answer a call.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

reason	The reason to answer a call.

Returns

Request number.

attachUserData

Syntax

attachUserDate(*userdata, reason*)

Description

Attach user data to a call.

Parameters

userdata	The user data to attach to the call.
reason	The reason to attach the user data.

Returns

Request number.

clear

Syntax

clear(*reason*)

Description

Clear a conference call.

Parameters

reason	The reason to clear the conference.

Returns

Request number.

complete

Syntax

complete(*reason*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Complete the two-step transfer/conference.

Parameters

reason	The reason to complete the two-step transfer/conference.

Returns

Request number.

conference

Syntax

```
conference(destination, reason, userdata, calldata)
```

Description

Conference a call.

Parameters

destination	The destination being invited into the conference.
reason	The reason to conference.
userdata	The user data to be attached.
calldata	The call data to be modified.

Returns

Request number.

conferenceSingle

Syntax

```
conferenceSingle(destination, reason, userdata, calldata)
```

Description

Single-step conference a call.

Parameters

destination	The destination being invited into the conference.
reason	The reason to conference.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

userdata	The user data to be attached.
calldata	The call data to be modified.

Returns

Request number.

dial

Syntax

```
dial(number, reason, userdata)
```

Description

Dial out.

Parameters

destination	The destination being invited into the conference.
reason	The reason to conference.
userdata	The user data to be attached.

Returns

Request number.

dropParty

Syntax

```
dropParty(destination, reason)
```

Description

Drop a party in conference.

Parameters

destination	The destination being dropped from the conference.
reason	The reason to drop the party.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

getAni

Syntax

```
getAni()
```

Description

Get the ANI.

Parameters

None.

Returns

Returns a string representing the ANI.

getDescr

Syntax

```
getDescr()
```

Description

Get the description attached to the call on the line.

Parameters

None.

Returns

Returns a string.

getDnis

Syntax

```
getDnis()
```

Description

Get the DNIS.

Parameters

None.

Returns

Returns a string.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

getPadvalue

Syntax

getPadvalue(*key*)

Description

Get the PAD value.

Returns

A string representing PAD value.

getReferenceId

Syntax

getReferenceId()

Description

Get the reference ID attached to the call on the line.

Parameters

None.

Returns

Returns a string.

getUrl

Syntax

getUrl(*defaultUrl*)

Description

Get the URL for screen pop-up.

Parameters

defaultUrl	The default pop-up URL.

Returns

Returns a string.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

grabCall

Syntax

grabCall(*destination*, *reason*)

Description

Grab a call from the queue.

Parameters

destination	The destination to be grabbed.
reason	The reason for the grab.

Returns

Request number.

hold

Syntax

hold(*reason*)

Description

Hold a call.

Parameters

reason	The reason to hold the call.

Returns

Request number.

join

Syntax

join(*reason*, *conferenceId*)

Description

Join an existing call/conference.

Parameters

reason	The reason to join the call.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

conferenceId	Call id.

Returns

Request number.

mute

Syntax

mute(*reason*)

Description

Mute an extension.

Parameters

reason	The reason to mute the call.

Returns

Request number.

park

Syntax

park(*destination, reason*)

Description

Park a call.

Parameters

destination	The destination on which the call will be parked.
reason	The reason to park the call.

Returns

Request number.

reconnect

Syntax

reconnect(*reason*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Reconnect to the original party during two-step transfer/conference.

Parameters

reason	The reason to reconnect.

Returns

Request number.

reject

Syntax

reject(*reason*)

Description

Reject an incoming call.

Parameters

reason	The reason to reject.

Returns

Request number.

release

Syntax

release(*reason*)

Description

Release a call.

Parameters

reason	The reason to release the call.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

retrieve

Syntax

retrieve(*reason*)

Description

Retrieve a held call.

Parameters

reason	The reason to retrieve the call.

Returns

Request number.

sendDTMF

Syntax

Syntax

sendDTMF(*reason*,*stringDTMF*)

Description

Send DTMF tones to the switch.

Parameters

reason	The reason for requesting DTMF.
stringDTMF	DTMF tones string (0-9,*,#).

Returns

Request number.

setcallresult

Syntax

setcallresult(*result*)

Description

Set call result.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

result	Result of the call.

Returns

Request number.

setcallresultDNC

Syntax

setcallresultDNC(*number*, *reason*)

Description

Do not call.

Parameters

number	The number not to be disturbed.
reason	The reason not to disturb.

Returns

Request number.

setcallresultReschedule

Syntax

setcallresultReschedule(*hour*, *minutes*, *day*, *month*,*year*)

Description

Reschedule a call.

Parameters

hour	Hours
Minutes	Minutes
Day	Day
Month	Month
Year	Year

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Returns

Request number.

transfer

Syntax

```
transfer(destination, reason, userdata, calldata)
```

Description

Transfer a call.

Parameters

destination	The destination to which the call will be transferred.
reason	The reason to transfer.
userdata	The user data to be attached.
calldata	The call data to be modified.

Returns

Request number.

transferMute

Syntax

```
transferMute(destination, reason, userdata, calldata)
```

Description

Transfer mute a call.

Parameters

destination	The destination to which the call will be transferred.
reason	The reason to transfer.
userdata	The user data to be attached.
calldata	The call data to be modified.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

unmute

Syntax

unmute(*reason*)

Description

Unmute a call.

Parameters

reason	The reason to unmute a call.

Returns

Request number.

updateCallData

Syntax

updateCallData(*calldata, reason*)

Description

Update call data to a call.

Parameters

calldata	The call data to update.
reason	The reason to update.

Returns

Request number.

Line Class Callback Event Methods

The following are the Line class callback event methods.

onAlternating

Syntax

onAlternating(*event*)

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Fires when alternating the call.

onAnswering

Syntax

onAnswering(*event*)

Description

Fires when answering the call.

onAttachingUD

Syntax

onAttachingUD(*event*)

Description

Fires when attaching user data.

onCallDataChanged

Syntax

onCallDataChanged(*event*)

Description

Fires when the call data on the call is changed.

onCapabilitiesChanged

Syntax

onCapabilitiesChanged(*event*)

Description

Fires when the capabilities changed.

onClearing

Syntax

onClearing(*event*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Fires when clearing the conference.

onCompleting

Syntax

onCompleting(*event*)

Description

Fires when completing the two-step transfer/conference.

onConferencing

Syntax

onConferencing(*event*)

Description

Fires when conferencing the call.

onDialing

Syntax

onDialing(*event*)

Description

Fires when there is an outgoing call.

onDropped

Syntax

onDropped(*event*)

Description

Fires when the call is released.

onError

Syntax

onError(*event*)

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Fires when there is error.

onGrabbing

Syntax

onGrabbing(*event*)

Description

Fires when grabbing a call from a queue.

onHeld

Syntax

onHeld(*event*)

Description

Fires when the call is on hold.

onHolding

Syntax

onHolding(*event*)

Description

Fires when holding the call.

onJoining

Syntax

onJoining(*event*)

Description

Fires when joining a call or a conference.

onMuted

Syntax

onMuted(*event*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Fires when a call is muted.

onOffHook

Syntax

onOffHook(*event*)

Description

Fires when the line is off hook.

onOnHook

Syntax

onOnHook(*event*)

Description

Fires when the line is on hook.

onParking

Syntax

onParking(*event*)

Description

Fires when parking the call.

onPartyAdded

Syntax

onPartyAdded(*event*)

Description

Fires when a new call party coming.

onPartyChanged

Syntax

onPartyChanged(*event*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Fires when the call party has changed.

onPartyRemoved

Syntax

onPartyRemoved(*event*)

Description

Fires when a call party is removed.

onReconnecting

Syntax

onReconnecting(*event*)

Description

Fires when reconnecting the call.

onRejected

Syntax

onRejected(*event*)

Description

Fires when call is rejected.

onRejecting

Syntax

onRejecting(*event*)

Description

Fires agent rejects an incoming call.

onReleasing

Syntax

onReleasing(*event*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Fires when releasing the call.

onRetrieving

Syntax

onRetrieving(*event*)

Description

Fires when retrieving the call.

onRinging

Syntax

onRinging(*event*)

Description

Fires when there is an incoming call.

onSetcallresult

Syntax

onSetcallresult(*event*)

Description

Fires when there is a request for setting call result.

onSetcallresultDNC

Syntax

onSetcallresultDNC(*event*)

Description

Fires when there is a request for do not call.

onSetcallresultReschedule

Syntax

onSetcallresultReschedule(*event*)

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Fires when there is a request to reschedule a call.

onTalking

Syntax

onTalking(*event*)

Description

Fires when the call is established.

onTransfering

Syntax

onTransfering(*event*)

Description

Fires when transferring the call.

onUnmuted

Syntax

onUnmuted(*event*)

Description

Fires when call is unmuted.

onUpdatingCD

Syntax

onUpdatingCD(*event*)

Description

Fires when updating the call data.

onUserDataChanged

Syntax

onUserDataChanged(*event*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Fires when the user data attached on the call is changed.

LineCaps Class Constructor

The following is the LineCaps class constructor.

LineCaps

Syntax

`LineCaps``(`*`strCaps`*`)`

Description

Describes the line's capabilities.

Parameters

strCaps	The line's capabilities.

Returns

A LineCaps object.

LineCaps Class Fields

The following are the LineCaps class fields.

canAlternate

Description

The line has alternate capability.

Type: boolean.

canAnswer

Description

The line has answer capability.

Type: boolean.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

canAttachUserDate

Description

The line has attach user data capability.

Type: boolean.

canClear

Description

The line has clear capability.

Type: boolean.

canComplete

Description

The line has complete capability.

Type: boolean.

canConference

Description

The line has conference capability.

Type: boolean.

canConferenceSingle

Description

The line has conference single capability.

Type: boolean.

canDropParty

Description

The line has drop party capability.

Type: boolean.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

canHold

Description

The line has hold capability.

Type: boolean.

canMute

Description

The line has mute capability.

Type: boolean.

canPark

Description

The line has park capability.

Type: boolean.

canReconnect

Description

The line has reconnect capability.

Type: boolean.

canReject

Description

The line has reject capability.

Type: boolean.

canRelease

Description

The line has release capability.

Type: boolean.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

canRetrieve

Description

The line has retrieve capability.

Type: boolean.

canSendDTMF

Description

The line has DTMF capability.

Type: boolean.

canSetcallresult

Description

The line has set call result capability.

Type: boolean.

canSetcallresultDNC

Description

The line has do not call capability.

Type: boolean.

canSetcallresultReschedule

Description

The line has reschedule capability.

Type: boolean.

canTransfer

Description

The line has transfer capability.

Type: boolean.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

canTransferMute

Description

The line has transfer mute capability.

Type: boolean.

canUnmute

Description

The line has unmute capability.

Type: boolean.

canUpdateCallData

Description

The line has update call data capability.

Type: boolean.

MCEvent Class Constructor

The following is the MCEvent class constructor.

MCEvent

Syntax

```
MCEvent()
```

Description

The MCEvent will be passed to the application event handler.

Parameters

None.

Returns

An MCEvent object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

MCEvent Class Fields

The following are the MCEvent class fields.

extension

Description

The extension associated with the event.

Type: Extension object.

Related Links

Extension Class Constructor

group

Description

The group associated with the event.

Type: Group object.

Related Links

Group Class Constructor

reason

Description

The reason associated with the event.

Type: Reason object.

Related Links

Reason Class Constructor

user

Description

The user associated with the event.

Type: User object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Related Links

User Class Constructor

MediaType Class Constructor

The following is the MediaType class constructor.

MediaType

Syntax

```
MediaType()
```

Description

Distinguishes between email and CTI media types.

Parameters

None.

Returns

A MediaType object.

MediaType Class Field

The following is the MediaType field.

type

Description

The media type.

Type: string with the following constants:

Value	*Description*
MT_CHAT	The chat media type.
MT_EMAIL	The email media type.
MT_GENERIC	The generic media type.
MT_VOICE	The voice media type (CTI).

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PSMC Class Constructor

The following is the PSMC class constructor.

PSMC

Syntax

PSMC ( )

Description

The global object that the application can access.

Parameters

None.

Returns

A PSMC object.

PSMC Class Fields

The following are the PSMC class fields.

renserver

Description

The PSMC instance of the RenServer object.

Type: RenServer object.

Related Links

RenServer Class Constructor

servers

Description

An associative array the different server objects, indexed by server ID.

Type: associative array.

Related Links

Server Class Constructor

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

sessions

Description

An associative array that holds the different session objects, indexed by server ID.

Type: associative array.

Related Links

Session Class Constructor

PSMC Class Methods

The following are the PSMC class methods.

closeSession

Syntax

`closeSession``(`*`serverId`*`)`

Description

Closes the specified session and server object and deletes them.

Parameters

serverId	The ID of the REN server cluster for a UQ server, or the server ID for a CTI server.

Returns

None.

getCallById

Syntax

`getCallById``(`*`id`*`)`

Description

Retrieve the call object by the call ID.

Parameters

id	The call ID.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Returns

Type: Call object.

Related Links

Call Class Constructor

getChatById

Syntax

getChatById(*id*)

Description

Retrieve the chat object by the task id.

Parameters

id	The task ID.

Returns

Type: chat object.

getEmailById

Syntax

getEmailById(*id*)

Description

Retrieve the email object by the task id.

Parameters

id	The task ID.

Returns

Type: email object.

getGenericTaskById

Syntax

getGenericTaskById(*id*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Retrieve the Generic object by the task id.

Parameters

id	The task ID.

Returns

Type: generic object.

getLineById

Syntax

getLineById(*id*)

Description

Retrieve the line object by the line ID.

Parameters

id	The line ID.

Returns

Type: Line object.

Related Links

Line Class Constructor

openSession

Syntax

openSession(*id*)

Description

Creates the session and server objects for the PSMC.

Parameters

id	The ID of the REN server cluster for a UQ server, or the server ID for a CTI server.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Returns

None.

start

Syntax

```
start()
```

Description

Start the JSMCAPI.

Parameters

None.

Returns

None.

stop

Syntax

```
stop()
```

Description

Stop the JSMCAPI.

Parameters

None.

Returns

None.

Reason Class Constructor

The following is the Reason class constructor.

Reason

Syntax

```
reason(code, desc, reasondata1, reasondata2, reasondata3)
```

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

The Reason object carries the reason code and reason description.

Parameters

code	The reason code.
desc	The reason description.
reasondata1	The reason data1.
reasondata2	The reason data2.
reasondata3	The reason data3

Returns

Type: Reason object.

Reason Class Fields

The following are the Reason class fields.

code

Description

The reason code.

Type: string.

desc

Description

The reason description.

Type: string.

reasonData1

Description

Place holder for extra data.

Type: string.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

reasonData2

Description

Place holder for extra data.

Type: string.

reasonData3

Description

Place holder for extra data.

Type: string.

RenServer Class Constructor

The following is the RenServer class constructor.

RenServer

Syntax

```
RenServer()
```

Description

The RenServer object describes the REN server cluster information.

Parameters

None.

Returns

A RenServer object.

RenServer Class Fields

The following are the RenServer class fields.

isRunning

Description

Flag containing the state of the REN server connection.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Type: boolean.

url

Description

The REN server URL.

Type: string.

RenServer Class Callback Event Methods

The following are the RenServer class callback event methods.

onDown

Syntax

onDown(*event*)

Description

Fires when the REN server is down.

onUp

Syntax

onUp(*event*)

Description

Fires when the REN server is up.

Server Class Constructor

The following is the Server class constructor.

Server

Syntax

Server(*serverId*)

Description

The Server object describes the server information.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

serverId	The ID of the REN server cluster for a UQ server, or the server ID for a CTI server.

Returns

A Server object.

Server Class Fields

The following are the Server class fields.

id

Description

The server ID.

Type: string.

info

Description

The server information string.

Type: string.

state

Description

The server state.

Type: string with the following constants.

Value	***Description***
ST_INSERVICE	The server is in service.
ST_OUTOFSERVICE	The server is out of service.

type

Description

The type of server, either CTI or UQ.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Type: string with the following constants.

Value	*Description*
TYPE_CTI	The server is a CTI server.
TYPE_MCS	The server is a third-party MultiChannel server.
TYPE_UQ	The server is a queue server.

Example

The TYPE_* fields are global fields, which may be accessed without instantiating an object. They are accessed as shown in this example:

```
Server.TYPE_UQ = "UQ";
```

Related Links

GLOBALS Class Fields

Server Class Callback Event Methods

The following are the Server class callback event methods.

onBroadcast

Syntax

onBroadcast(*event*)

Description

Fires when there is broadcast message.

onHbLost

Syntax

onHbLost(*event*)

Description

Fires when the server's heartbeat is lost.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onHbRecovered

Syntax

onHbRecovered(*event*)

Description

Fires when the server's heartbeat is recovered.

onInService

Syntax

onInService(*event*)

Description

Fires when the server changes state to in service.

onOutOfService

Syntax

onOutOfService(*event*)

Description

Fires when the server changes state to out of service.

onRestart

Syntax

onRestart(*event*)

Description

Fires when server restarts.

Session Class Constructor

The following is the Session class constructor.

Session

Syntax

Session(*serverId*)

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

The Session object describes the session with the server.

Parameters

serverId	The ID of the REN server cluster for a UQ server, or the server ID for a CTI server.

Returns

Returns a Session object.

Session Class Fields

The following are the Session class fields.

addresses

Description

The associative array of addresses that the session register.

Type: associative array.

buddies

Description

The buddy hash table in this session.

Type: associative array.

groups

Description

The group hash table in this session.

Type: associative array.

id

Description

The session ID generated by the server.

Type: string.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

intervalBetweenReqs

Description

Interval in milliseconds between requests.

Type: number

numberRegsPerBulkReq

Description

Number of users or groups to register for one bulk register request.

Type: number

serverId

Description

The serverId will be unique for every session. It is used to lookup the protocol objects.

Type: string.

state

Description

The session state.

Type: string with the following constants.

Value	*Description*
ST_ACTIVE	The session is in the active state.
ST_CLOSED	The session is in the closed state.
ST_CLOSING	The session is in the closing state.
ST_IDLE	The session is in the idle state.

user

Description

The current user for the session.

Type: User object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Related Links

User Class Constructor

Session Class Methods

The following are the Session class methods.

broadcastSubscribe

Syntax

broadcastSubscribe(*cluster*, *queue*, *task*, *state*, *presence*, *method*)

Description

Subscribe to broadcast messages.

Returns

Object.

broadcastUnsubscribe

Syntax

broadcastunsubscribe(*type*)

Description

Unsubscribe to broadcast messages.

Returns

None.

close

Syntax

close()

Description

Close the session with the server.

Parameters

None.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Returns

Request number.

open

Syntax

open()

Description

Open the session with the server.

Parameters

None.

Returns

Request number.

registerAddress

Syntax

registerAddress(*address*)

Description

Register the address to the server so that the server will report all events that occur on this address.

Parameters

address	The address to be registered.

Returns

Request number.

registerBuddy

Syntax

registerBuddy(*buddy*)

Description

Register the buddy to the JSMCAPI so that the JSMCAPI will report all state events that occur on this buddy.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

buddy	The buddy to be registered.

Returns

Request number.

registerBuddiesBulk

Syntax

registerBuddiesBulk(*buddies*, *numRegsPerBulkReq*, *intervalBetweenReqs*)

Description

Register multiple buddies to the JSMCAPI so that the JSMCAPI will report state events that occur for the buddies.

Parameters

buddies	An array of buddies
numRegPerBulkReq	Number of buddies to register in one bulk registration request.
intervalBetweenReqs	Interval in milliseconds between requests.

Returns

Request number.

registerGroup

Syntax

registerGroup(*group*)

Description

Register the group to the server so that the server will report all events that occur on this group.

Parameters

group	The group to be registered.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

registerGroupsBulk

Syntax

registerGroupsBulk(*groups*, *numRegsPerBulkReq*, *intervalBetweenReqs*)

Description

Register multiple groups to the server so that the server will report all events that occur on the groups.

Parameters

groups	An array of groups.
numRegPerBulkReq	Number of groups to register in one bulk registration request.
intervalBetweenReqs	Interval in milliseconds between requests.

Returns

Request number.

registerUser

Syntax

registerUser(*user*)

Description

Register the user to the server so that the server will report all those events that happen on this user.

Parameters

user	The user to be registered.

Returns

Request number.

setAutoRecovery

Syntax

setAutoRecovery(*autorecover*)

Description

Sets the auto recovery feature of the queue server connection to off.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

autoRecover	Recover.

statPublish

Syntax

```
statPublish(userStat, groupStat)
```

Description

Sets the statistics publishing levels for the queue server.

Parameters

userStat	Either 0, 1, or 2 for UserStatistics publishing. Enter 0 for no statistics publishing, 1 for Statistics1 publishing, and 2 for Statistics2 and Statistic1 publishing.
groupStat	Either 0, 1, or 2 for GroupStatistics publishing. Enter 0 for no statistics publishing, 1 for Statistics1 publishing, and 2 for Statistics2 and Statistic1 publishing.

Returns

Request number.

unregisterAddress

Syntax

```
unRegisterAddress(address)
```

Description

Unregister the address from the server so that the server will not report any events about this address.

Parameters

address	The address to be unregistered.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

unregisterBuddy

Syntax

unregisterBuddy(*buddy*)

Description

Unregister the buddy from the JSMCAPI so that the JSMCAPI will not report any events about the buddy state change.

Parameters

buddy	The buddy to be unregistered.

Returns

Request number.

unregisterGroup

Syntax

unregisterGroup(*group*)

Description

Unregister the group from the server so that the server will not report any events about this group.

Parameters

group	The group to be unregistered.

Returns

Request number.

unregisterUser

Syntax

unregisterUser(*user*)

Description

Unregister the user from the server so that the server will not report any events to this user.

Parameters

user	The user to be unregistered.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Returns

Request number.

Session Class Callback Event Methods

The following are the Session class callback event methods.

onAddressRegistered

Syntax

onAddressRegistered(*event*)

Description

Fires when the address is registered by the session.

onAddressUnregistered

Syntax

onAddressUnregistered(*event*)

Description

Fires when the address is unregistered by the session.

onBuddyRegistered

Syntax

onBuddyRegistered(*event*)

Description

Fires when the buddy is registered by the session.

onBuddyUnregistered

Syntax

onBuddyUnregistered(*event*)

Description

Fires when the buddy is unregistered by the session.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onClosed

Syntax

onClosed(*event*)

Description

Fires when the session is closed.

onError

Syntax

onError(*event*)

Description

Fires when there is a session error.

onGroupRegistered

Syntax

onGroupRegistered(*event*)

Description

Fires when a group is registered by the session.

onGroupUnregistered

Syntax

onGroupUnregistered(*event*)

Description

Fires when a group is unregistered by the session.

onInfo

Syntax

onInfo(*event*)

Description

Fires when there is a session information event, such as the user is already logged in.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onOpened

Syntax

onOpened(*event*)

Description

Fires when the session is opened.

onUserRegistered

Syntax

onUserRegistered(*event*)

Description

Fires when the user is registered by the session.

onUserUnregistered

Syntax

onUserUnregistered(*event*)

Description

Fires when the user gets unregistered by the session.

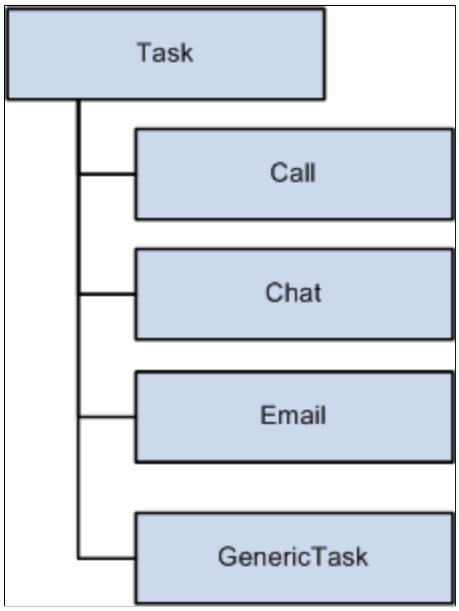
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Task Class Hierarchy

The Task class can be extended by other subclasses.

Image: Task class hierarchy

The following flow chart shows the different subclasses of the Task class and how they interrelate.



Related Links

Call Class Constructor
Chat Class Constructor
Email Class Constructor
GenericAddress Class Constructor

Task Class Constructor

The following is the Task class constructor.

Task

Syntax

```
Task()
```

Description

Task is an abstract base class.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

None.

Returns

Type: Task object.

Task Class Fields

The following are the Task class fields.

caseid

Description

The associated case ID.

Type: string.

cost

Description

The associated cost.

Type: string.

customerid

Description

The associated customer ID.

Type: string.

group

Description

The group for which task is assigned.

Type: string.

id

Description

The task ID.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Type: string.

onStat

Description

Method that will be triggered when there is statistics published regarding this task.

Type: string.

priority

Description

The priority of the task.

Type: string.

type

Description

The type of the task.

Type: string, one of the following:

Value	*Description*
TYPE_A2ACHAT	The A2AChat type.
TYPE_CHAT	The chat type.
TYPE_CTI	The CTI type.
TYPE_EMAIL	The email type.
TYPE_GENERIC	The generic type.

Example

The TYPE_* fields are global fields, which may be accessed without instantiating an object. They are accessed as shown in this example:

```
Task.TYPE_GENERIC = "GENERIC";
```

Related Links

GLOBALS Class Fields

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

urlAbs

Description

The absolute URL that is useful in constructing URL for new window.

Type: urlAbs object.

urlRel

Description

The relative URL that is useful in constructing URL for new window.

Type: urlRel object.

TaskStatistics Class Constructor

The following is the TaskStatistics class constructor.

TaskStatistics

Syntax

```
TaskStatistics()
```

Description

Describes the task statistics information.

Returns

Task statistics object.

TaskStatistics Class Fields

The following are the TaskStatistics class fields.

data

Description

Key value pairs that includes all statistics.

Type: collection.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

holdTime

Description

Time duration that the task is on hold.

Type: number

queueTime

Description

Time duration in the queue for this task.

Type: number

talkTime

Description

Time duration that the task is established.

Type: number

User Class Constructor

The User class extends the _User class.

See _User Class Constructor.

The following is the User class constructor.

User

Syntax

```
User(id, name, agentId, agentPassword)
```

Description

Describes the user/agent.

Parameters

id	The user ID.
name	The user name.
agentId	The agent ID.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

agentPassword The agent's password.

Returns

A User object.

User Class Fields

The User class inherits the following fields from the _User class:

- agentId
- caps
- id
- name
- presences
- ST_LOGGEDIN
- ST_LOGGEDOUT
- ST_NOTREADY
- ST_READY
- ST_UNKNOWN
- ST_WORKNOTREADY
- ST_WORKREADY
- states
- statistics
- statistics1
- statistics2

The following are the User class fields.

addresses

Description

The addresses associated with this agent.

Type: associative array.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

agentPassword

Description

The password for this agent.

Type: string.

language

Description

The language for this agent.

Type: string.

registeredAddresses

Description

The registered addresses for this user.

Type: associative array.

statesUq

Description

States of the user for the queue server indexed by group number

Type: object.

User Class Methods

The following are the User class methods.

ctiBusyUq

Syntax

ctiBusyUq(*busy*, *subtractcost*, *task*)

Description

Flip ctiBusy flag, reduce cost of a CTI task from an agent's workload, or both. If the task is the same as the agent's assigned task, then the task's cost will be used. In case they do not match, the task type is checked and the default cost of the task is used to recalculate the workload.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

busy	The ctibusy flag that can have a value of 0 or 1.
subtractcost	Flag to reduce cost of the cti task from the agent's workload. It can be set to 0 or 1.
task	The task object.

Returns

Request number.

disableMedia

Syntax

```
disableMedia(group, mediaType, extension, reason)
```

Description

Disable a media channel for an agent on a group/queue.

Parameters

group	The group (queue) on which to disable the specified media type.
mediaType	The media type to disable.
extension	The extension on which to disable the specified media type.
reason	The reason for disabling this media type.

Returns

Request number.

enableMedia

Syntax

```
enableMedia(group, mediaType, extension, reason)
```

Description

Enable a media channel for an agent on a group/queue.

Parameters

group	The group (queue) on which to enable the specified media type.
mediaType	The media type to enable.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

extension	The extension on which to enable the specified media type.
reason	The reason for enabling this media type.

Returns

Request number.

isMediaEnabled

Syntax

isMediaEnabled(*mediaType*, *group*)

Description

Determines if the specified media type is enabled.

Parameters

mediaType	Enter the media type.
group	Enter the group.

Returns

Returns true if the media type is enabled.

login

Syntax

login(*group*, *extension*, *reason*)

Description

Log the user in to a queue.

Parameters

group	The group to which the user will log in.
extension	The extension.
reason	The reason for the login.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

loginUq

Syntax

loginUq(*group*)

Description

Log a user in to a UQ server queue.

Parameters

group	The group (queue) to which the user will login.

Returns

Request number.

logout

Syntax

logout(*group*, *extension*, *reason*)

Description

Log the user out of the specified group (queue).

Parameters

group	The group from which the user will log out.
extension	The extension.
reason	The reason for the log out.

Returns

Request number.

logoutUq

Syntax

logoutUq(*group*)

Description

Log the user out from the UQ server queue.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

group	The group (queue) from which to log the user out.

Returns

Request number.

register

Syntax

```
register(extension, reason)
```

Description

Register an extension for the user.

Parameters

extension	The extension to be registered.
reason	The reason to register the extension.

Returns

Request number.

setNotReady

Syntax

```
setNotReady(group, presence, extension, reason)
```

Description

Set the user state as not ready.

Parameters

group	The group on which the user is changing state.
presence	The presence in the new state.
extension	The extension with which the user is changing state.
reason	The reason to set not ready.

Returns

Request number.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

setPresence

Syntax

setPresence(*group*, *presence*, *extension*, *reason*)

Description

Set the presence for the current state.

Parameters

group	The group on which the user is changing presence.
presence	The presence for the state.
extension	The extension with which the user is changing presence.
reason	The reason to set the presence.

Returns

Request number.

setPresenceUq

Syntax

setPresenceUq(*group*, *presenceText*, *reason*)

Description

Set the Presence for UQ servers.

Parameters

group	The group (queue) on which the user is changing his or her presence.
presenceText	The new presence.
reason	The reason to set the new presence.

Returns

Request number.

setReady

Syntax

setReady(*group*, *presence*, *extension*, *reason*)

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

Set the user state as ready.

Parameters

group	The group on which the user is changing state.
presence	The presence for the state.
extension	The extension with which the user is changing state.
reason	The reason to set the state.

Returns

Request number.

setWorkNotReady

Syntax

```
setWorkNotReady(group, presence, extension, reason)
```

Description

Set the user state as work not ready.

Parameters

group	The group on which the user is changing state.
presence	The presence for the state.
extension	The extension with which the user is changing state.
reason	The reason to set work not ready.

Returns

Request number.

setWorkReady

Syntax

```
setWorkReady(group, presence, extension, reason)
```

Description

Set the user state as work ready.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

group	The group on which the user is changing state.
presence	The presence for the state.
extension	The extension with which the user is changing state.
reason	The reason to set work ready.

Returns

Request number.

unregister

Syntax

`unregister``(`*`extension, reason`*`)`

Description

Unregister an extension for the user.

Parameters

extension	The extension to be unregistered.
reason	The reason to unregister the extension.

Returns

Request number.

User Class Callback Event Methods

The following are the User class callback event methods.

onCapabilitiesChanged

Syntax

`onCapabilitiesChanged``(`*`event`*`)`

Description

Fires when capabilities changed.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onCtiBusy

Syntax

onCtiBusy(*event*)

Description

Fires when the UQ server is notified that the CTI is busy.

onCTIBUSYUq

Syntax

onCTIBUSYUq(*event*)

Description

Fires when the UQ server is notified that the CTI is busy.

onCtiClear

Syntax

onCtiClear(*event*)

Description

Fires when the UQ server is notified that the CTI is clear.

onDropped

Syntax

onDropped(*event*)

Description

Fires when a user is dropped.

onError

Syntax

onError(*event*)

Description

Fires when an error occurs.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onInfo

Syntax

onInfo(*event*)

Description

Fires when an information event occurs. Currently only used on UQ server.

onLoggedIn

Syntax

onLoggedIn(*event*)

Description

Fires when the user is logged in.

onLoggedOut

Syntax

onLoggedOut(*event*)

Description

Fires when the user is logged out.

onLoggingIn

Syntax

onLoggingIn(*event*)

Description

Fires when the user is logging in.

onLoggingOut

Syntax

onLoggingOut(*event*)

Description

Fires when the user is logging out.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onMediaDisabled

Syntax

onMediaDisabled(*event*)

Description

Fires when a media type is disabled.

onMediaEnabled

Syntax

onMediaEnabled(*event*)

Description

Fires when a media type is enabled.

onNotReady

Syntax

onNotReady(*event*)

Description

Fires when a user state changes to not ready.

onPresenceChanged

Syntax

onPresenceChanged(*event*)

Description

Fires when the user's presence is changed.

onReady

Syntax

onReady(*event*)

Description

Fires when the user's state changes to ready.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onRegistered

Syntax

onRegistered(*event*)

Description

Fires when the address is registered.

onRegistering

Syntax

onRegistering(*event*)

Description

Fires when registering the address.

onSettingNotReady

Syntax

onSettingNotReady(*event*)

Description

Fires when setting the user's state to not ready.

onSettingPresence

Syntax

(*event*)

onSettingPresence

Description

Fires when setting presence for the current state.

onSettingReady

Syntax

onSettingReady(*event*)

Description

Fires when setting the user's state to ready.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onSettingWorkNotReady

Syntax

onSettingWorkNotReady(*event*)

Description

Fires when setting the user's state to work not ready.

onSettingWorkReady

Syntax

onSettingWorkReady(*event*)

Description

Fires when setting the user's state to work ready.

onStat

Syntax

onStat(*event*)

Description

Fires when statistics are received.

onStat1

Syntax

onStat1(*event*)

Description

Fires when statistics1 is received.

onStat2

Syntax

onStat2(*event*)

Description

Fires when statistics2 received.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

onUnknown

Syntax

onUnknown(*event*)

Description

Fires when the user's state changes to unknown.

onUnregistered

Syntax

onUnregistered(*event*)

Description

Fires when the address gets unregistered.

onUnregistering

Syntax

onUnregistering(*event*)

Description

Fires when unregistering the address.

onWorkNotReady

Syntax

onWorkNotReady(*event*)

Description

Fires when the user's state changes to work not ready.

onWorkReady

Syntax

onWorkReady(*event*)

Description

Fires when the user's state changes to work ready.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

UserCaps Class Constructor

The following is the UserCaps class constructor.

UserCaps

Syntax

UserCaps(*strCaps*)

Description

The UserCaps object describes the user capabilities.

Parameters

strCaps	The user capabilities.

Returns

A UserCaps object.

UserCaps Class Fields

The following are the UserCaps class fields

canHandleChat

Description

Chat task handling capability of the user.

Type: boolean.

canHandleEmail

Description

Email task handling capability of the user.

Type: boolean.

canHandleGeneric

Description

Generic task handling capability of the user.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Type: boolean.

canHandleVoice

Description

Voice task handling capability of the user.

Type: boolean.

canLogin

Description

Login capability of the user.

Type: boolean.

canLogout

Description

Logout capability of the user.

Type: boolean.

canRefreshState

Description

The refreshState capability of the user.

Type: boolean.

canSetNotReady

Description

The setNotReady capability of the user.

Type: boolean.

canSetPresence

Description

The setPresence capability of the user.

Type: boolean.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

canSetReady

Description

The setReady capability of the user.

Type: boolean.

canSetWorkNotReady

Description

The setWorkNotReady capability of the user.

Type: boolean.

canSetWorkReady

Description

The setWorkReady capability of the user.

Type: boolean.

UserData Class Constructor

The following is the UserData class constructor.

UserData

Syntax

```
UserData()
```

Description

The UserData object describes the key-value pairs of attached user data.

Parameters

None.

Returns

A UserData object.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

UserData Class Fields

The following are the UserData class fields constants.

UserData Class Field Constants

Description

UserData class uses the following constants.

Value	*Description*
COMPONENT	The component.
DESCR	The description
ICSCRIPTPROGRAMNAME	The IC script program name.
ICTYPE	The IC type.
MARKET	The market.
MENU	The menu.
REFERENCEID	The reference ID
TARGET	The target.
URLABS	The absolute URL.
URLREL	The relative URL.

UserData Class Method

The following is the UserData class method.

addKeyValue

Syntax

addKeyValue(*key*, *value*)

Description

Add the key-value pair to the UserData object.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameters

key	The key.
value	The value.

Returns

None.

UserStatistics1 Class Constructor

The following is the UseStatistics1 class constructor.

UserStatistics1

Syntax

```
UserStatistics1()
```

Description

UQ User statistics sent on user refresh 1.

Parameters

None.

Returns

A UserStatistics1 object.

UserStatistics1 Class Fields

The following are the UserStatistics1 class fields.

availableCost

Description

The amount of unused work.

ctiBusy

Description

Flag indicating that a user is engaged on CTI task.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

currentQueue

Description

Current queue/group the User last logged into.

Type: string.

mostRecentTaskData

Description

Most recent task's data.

Type: string.

mostRecentTaskId

Description

Most recent task's ID.

Type: string.

numTaskAccepted

Description

The number of tasks accepted by the user.

Type: string.

numTasksDone

Description

The number of tasks done by the user.

numTasksUnassigned

Description

The number of tasks unassigned by the user.

Type: string.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

presenceText

Description

The presence text for the user.

Type: string.

reasonFlag

Description

The reason flag for this event.

Type: string.

state

Description

Current state of the user.

Type: string.

timeInCurrentState

Description

Time in current state for the user.

Type: string.

timeSinceLoggedIn

Description

The time since the user logged in.

Type: string.

UserStatistics2 Class Constructor

The following is the UserStatistics2 class constructor.

UserStatistics2

Syntax

`UserStatistics2``()`

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Description

UQ User statistics sent on user refresh 2 event.

Parameters

None.

Returns

A UserStatistics2 object.

UserStatistics2 Class Fields

The following are the UserStatistics2 class fields.

currentQueue

Description

The current queue or group for the user.

Type: string.

timeIdle

Description

Idle time for the user.

Type: string.

timeInCurrentState

Description

The time in the current state for the user.

Type: string.

timeNotReady

Description

The time during which the user was in the not ready state.

Type: string.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

timeSinceLogin

Description

The time since the user logged in.

Type: string.

totalTimeAvailable

Description

The total time during which the user has been in the available state.

Type: string.

totalTimeUnavailable

Description

The total time during which the user has been in the unavailable state.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 14

Configuring the Email Channel

Understanding the Email Channel

Because the email channel uses a PeopleSoft Integration Broker gateway, the email channel requires additional configuration outside of PeopleSoft MultiChannel Framework configuration pages. The configuration of email channel in the PeopleSoft Integration Broker is described later.

See Configuring Email Channel in PeopleSoft Integration Broker.

PeopleSoft MultiChannel Framework (MCF) provides a framework for:

- Fetching emails from a mail server.
- Storing and retrieving email parts in a database.
- Managing attachments.
- Queueing of emails by the queue server.

Application developers use the PeopleCode application package PT_MCF_MAIL to develop PeopleSoft Application Engine programs and PeopleSoft Pure Internet Architecture components for email processing.

See "Understanding PeopleSoft MultiChannel Framework Mail Classes" (PeopleTools 8.54: PeopleCode API Reference).

PeopleSoft MultiChannel Framework email channel features include:

- The GETMAILTARGET target connector.
- A PeopleCode application package, PT_MCF_MAIL, to retrieve, store, and delete email.
- Support for both Post Office Protocol 3 (POP3) and Internet Message Access Protocol 4 (IMAP4) email protocols.
- Support for SSL connections.
- Support for NT LAN Manager (NTLM) and Simple Authentication and Security Layer (SASL) authentication mechanisms.
- Support for email attachments, including:
    - An attachments repository running on the same web server as PeopleSoft Integration Broker.
    - URL access to email attachments.
    - Storage and retrieval of email attachments from database.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

  - A relative addressing scheme to enable flexibility of repository location.
  - User- and role-based security to control access to attachments.

- The option to access email headers, attachments, and body from the mail server.
- Support for multiple message size thresholds to control distribution of email between the database, the attachment repository, and the mail server.
- Support for UTF8 Unicode as supported by the Sun Java Runtime Environment (JRE).

  See http://java.sun.com.

- Time zone conversions to support global service-level agreements.
- Sample email application pages demonstrating the functionality of the PT_MCF_MAIL application package.
- Ability to enable logging for MCFOutbound email by setting SMTP trace in psappsrv.cfg. The log file is created in the log directory defined in psappsrv.cfg.

  See "SMTPTrace" (PeopleTools 8.54: System and Server Administration).

Handling Email

This section describes some factors to consider when handling email using PeopleSoft MultiChannel Framework.

POP3 Versus IMAP4

Most mail servers support simultaneous client connections through both POP3 and IMAP4. Using IMAP4 for your MCF email connection provides significant benefit over using POP3. IMAP4 allows for the use of email folders, which allows malformed emails to be set aside for separate processing. Quarantining malformed emails allows system administrators to remove or correct invalid emails without interrupting normal processing.

Note: Using IMAP4 for your MCF email connection does not preclude the use of POP3 for any other client connections to your mail server.

Note: POP3 mail servers are typically read-only, which can create issues if emails are required to be deleted after being read.

Time Zone Offsets

Set the values of the email sent time zone offset (in minutes) and the receive time zone offset (in minutes) whenever possible. The default value is 800, which indicates time zone information is not available. When available, the values range from +720 to −720.

Connector Determination of Email Size

The connector sometimes cannot determine the size of the message. In such cases the size is set to 0 and an error message is written to the gateway error log.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Malformed Emails

Emails with syntax that does not comply to the latest standard definitions may not be processed successfully by MCF. MCF will return as much as possible of noncompliant emails, however, at times, invalid parts will be ignored. For example, email addresses with invalid characters will not be displayed, such as two @ symbols. Also, if the header information is not valid, such as a malformed content type, the associated email part may not appear.

A common example of a malformed email is one that contains 8-bit encoded information in a header, such as the To, Cc or Subject field, or in an attachment file name. According to the standard definitions, this information must be encoded using 7-bit encoding, as described in RFC 2045 and others. This encoding is usually done by the email sender.

Note: The PeopleSoft email reader only supports email messages that follow the SMTP specification. Many email clients and mail servers have their own proprietary formats; therefore, be aware that PeopleSoft MCF will interpret these emails according to the RFC specification. For example, some mail servers and clients allow 8-bit encoded data in email headers, which the PeopleSoft email reader does not allow.

Domain Validation

Application developers can use the method IsDomainNameValid to validate email address domains.

Note: The number of retries for domain validation is set by the parameter SMTPDNSTimeoutRetries in psappsrv.cfg.

See "SMTP Settings" (PeopleTools 8.54: System and Server Administration), "IsDomainNameValid" (PeopleTools 8.54: PeopleCode API Reference).

Email Address Syntax Validation

The ValidateAddress method checks the email address syntax against RFC standards.

See "ValidateAddress" (PeopleTools 8.54: PeopleCode API Reference).

Character Sets

If you wish to attach files which are named using characters outside of the character set recognized by your application server or process scheduler operating system, you should update the locale or regional settings of your operating system to allow those characters to be recognized.

SSL Connections

Certificates are an important component in SSL communication for authentication of the parties involved. In this case, one party is the Email Server and the other party is PeopleSoft Applications.

For the communication to work, you must import the trusted CA root certificates of the CA which signed the Email Server certificate into the Integration Broker Gateway truststore. If a CA issues a chain of certificates, you must add the entire certificate chain from the signer of the Email Server certificate to the Root CA certificate.

By default, the gateway truststore is the pskey peoplesoft keystore stored on the web server. The path to the gateway truststore is defined by the SSL_KEY_STORE_PATH variable in the setenv.cmd file.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The Email Server certificates must be imported into the Integration Broker keystore. The location of this keystore is defined by the secureFileKeystorePath property in Gateway Properties. By default, this keystore is the pskey peoplesoft keystore.

See "Importing Keys and Certificates Into the Keystore" (PeopleTools 8.54: System and Server Administration).

Connect data for outgoing emails can be configured in psappsrv.cfg or provided by the application using application package PT_MCF_MAIL.

See "Understanding PeopleSoft MultiChannel Framework Mail Classes" (PeopleTools 8.54: PeopleCode API Reference).

NTLM and SASL Authentication Mechanisms

MCF supports the NT LAN Manager (NTLM) and the Simple Authentication and Security Layer (SASL) authentication mechanisms for outgoing emails. To enable these authentication mechanisms, you must add and configure the NTLM and the SASL parameters in psappsrv.cfg and psprcs.cfg for a primary server and if required, for a backup server. MCF does not set default values for the NTLM and the SASL parameters in the configuration files.

Note: NTLM and SASL log will be available in the SMTP log because NTLM and SASL are part of SMTP sessions. That is, separate log files are not created for NTLM and SASL.

See "SMTP Settings" (PeopleTools 8.54: System and Server Administration).

Virus Scanning

You can enable virus scanning for inbound MCF Email attachments and outbound file attachments.

Configuring Email Channel in PeopleSoft Integration Broker

Before you configure the email channel for PeopleSoft Multichannel Framework, you must successfully install and configure the PeopleSoft Integration Broker.

See "Installing PeopleSoft Integration Broker" (PeopleTools 8.54: Integration Broker Administration)

For the email channel to work, you must configure the following in PeopleSoft Integration Broker:

- Define a gateway that will be used with the MCF_GetMail connector node.

  See "Defining Integration Gateways and Loading Connectors" (PeopleTools 8.54: Integration Broker Administration)

- Define MCF_GetMail node.

  See "Adding Node Definitions" (PeopleTools 8.54: Integration Broker Administration)

- Define a connector for the MCF_GetMail node and specify the gateway.

  See "Specifying Gateways and Connectors" (PeopleTools 8.54: Integration Broker Administration)

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- Configure GETMAILTARGET connector properties.

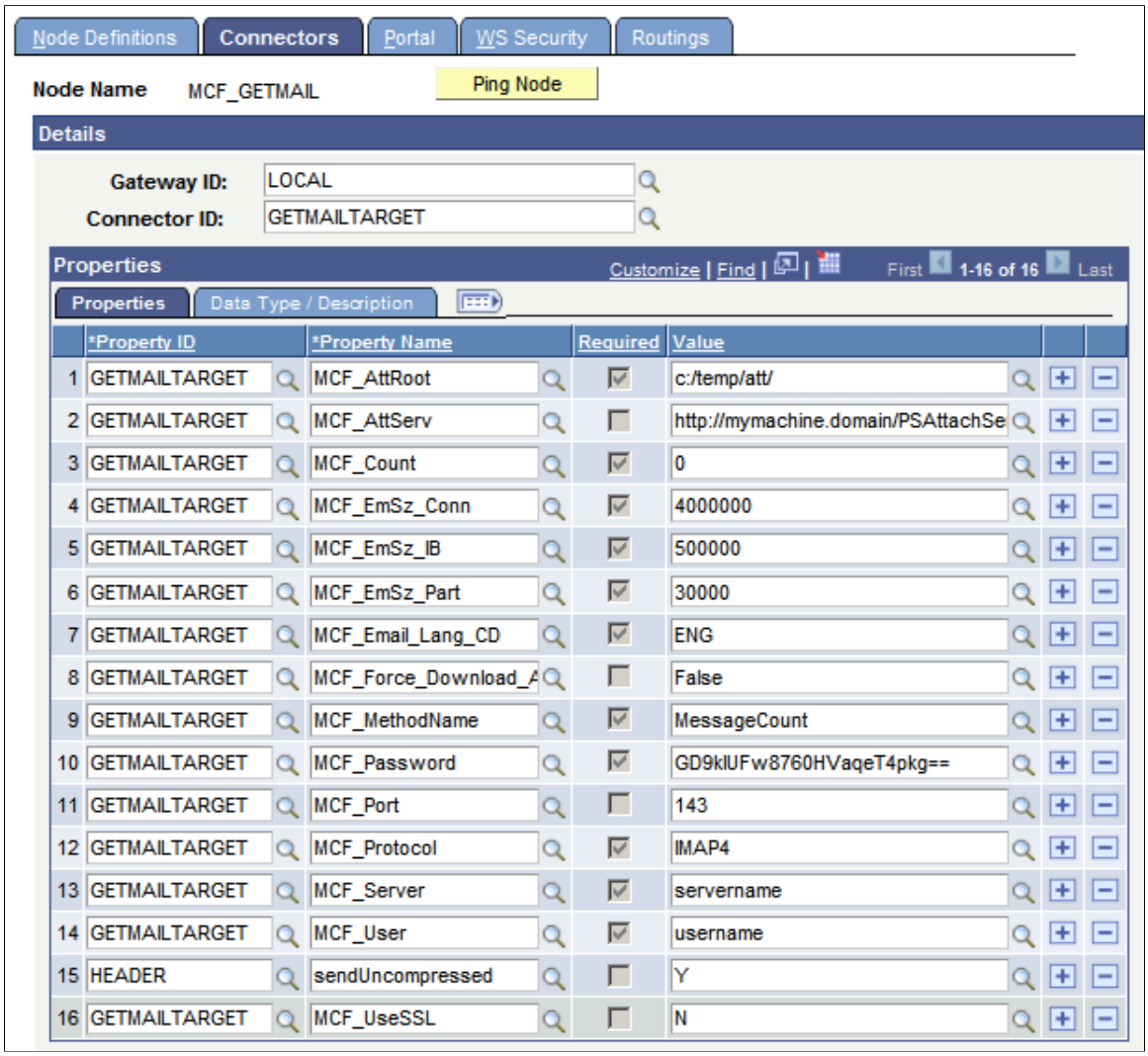
Configuring GETMAILTARGET Connector Properties

Access the Connector Properties page using the following navigation path:

PeopleTools, Integration Broker, Integration Setup, Nodes, Connectors

Image: Connectors page

This example illustrates the fields and controls on the Connectors page.



Configure GETMAILTARGET connector properties on the MCF_GETMAIL node. You can also set threshold values for email routed to the attachment repository.

The following table lists node properties:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Property Name	***Default Value***	***Description***
MCF_AttRoot	*c:\temp\att*	Defines the location of the attachment repository on the gateway. For example, if the attachment root is c:\temp\att, then attachments for emails addressed to a mail box *support* are downloaded to c:\temp\att\<mail server host name>\<mailbox>. **Note:** If the attachments are to be stored by a gateway that is remote to the webserver specified in the MCF_AttServ property, be sure that the file system permissions are set so that the directory may be accessed by both. The user permissions for the directory specified by MCF_AttRoot should be set such that the node (such as the local node) that will retrieve the attachments has read permissions. **Note:** The user name or mailbox portion of the attachment root path may include encoded characters. All email attachments, and email parts not of the content types specified for the MCF_ContentTypes property, are always written to the attachment repository. A relative URL is returned to the application that allows the application to access the attachment. The attachment retrieval depends on the file system of the operating system. For Windows NTFS/FAT 16 file system, the folder length or maximum characters of the file extension is 256/512. For Unix and Solaris UFS file system, the length is 256. Attachment security is managed at a user and role level using the PT_MCF_EMAIL application package. Users can configure attachment folder paths using metastrings to enable multiple folders based on several variables: • *%OPRID%:* PeopleSoft user ID. • *%DBNAME%:* database name. • *%CURRDATE%:* current date. • *%CURRHOUR%:* current hour.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Property Name	***Default Value***	***Description***
MCF_RepositoryType	*FILE*	Specifies the location for storing email attachments. Email attachments can be stored either in the attachment repository or in the database. The default value is FILE. If you select DATABASE, the email attachments are stored in the database. Also, when you select DATABASE, the value in the MCF_AttRoot property is ignored. If you select FILE, the email attachments are stored in the attachment repository.
MCF_AttServ	*http://<machine_name>.<domain_name>/PSAttachServlet/ps/*	Defines the location of the MCF attachment repository servlet, located on the gateway web server. Attachment relative URLs are appended to this address to create a fully qualified URL to reference an attachment in the repository. **Note:** When you are setting a remote gateway to access email attachments stored on the remote machine, the database that the gateway/psattachservlet accesses should have the permissions to view the attachments. **Note:** If your web server is installed with a PeopleSoft Pure Internet Architecture site name other than *ps,* include the site name in the specified path.
MCF_ClientCertAlias	ClientCert	Provide the alias of the client certificate. **Note:** The client certificate must be imported in the pskey keystore. **Note:** The MCF_ClientCertAlias property must be set only if the Email Server is set for client authentication, that is, the Email Server is set to request the client's certificate for authentication. Additionally, you must ensure that the trusted certificates and the certificate chain (if any) of the client certificate are loaded in the Email Server. See "Installing Integration Gateway-Based Digital Certificates" (PeopleTools 8.54: Integration Broker Administration).

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Property Name	***Default Value***	***Description***
MCF_ClientCertPass	password	Enter the password of the private key that you provided when you created the client certificate. **Note:** The alias and the password must match the alias and password of the imported client certificate in the pskey keystore. **Note:** The MCF_ClientCertPass property must be set only if the Email Server is set for client authentication, that is, the Email Server is set to request the client's certificate for authentication. Additionally, you must ensure that the trusted certificates and the certificate chain (if any) of the client certificate are loaded in the Email Server.
MCF_ContentTypes	*text/plain*	Specifies email content types that will be stored in the database. Content types not specified will be stored as attachments. Enter content types separated by commas. For example, you can Enter *text/html* and *text/xml* to enable storage in the database of HTML email and email containing XML. The content type *text/plain* is always stored in the database, regardless of values specified for MCF_ContentTypes. Do not include binary encoded content types in the content types list unless the application can handle base64 encoded text.
MCF_Count	*0*	Defines the default number of emails retrieved from the mail server unless otherwise specified in the SyncRequest. The PeopleSoft MultiChannel Framework email application package always specifies the number of emails to be retrieved and does not reference this property.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Property Name	***Default Value***	***Description***
MCF_EmSz_Conn	*4000000*	Defines the connector threshold, in bytes. This parameter ensures that emails retrieved from a POP3 mail server do not exceed the available memory on the gateway server. Email content retrieved from IMAP4 servers can be streamed to a file on the attachment repository, but content retrieved from POP3 servers must first be read into memory before being written to a file. Therefore, the MCF_EmSz_Conn threshold can be set very high for IMAP4 mail servers. PeopleSoft MultiChannel Framework does not process an email if its size is greater than this value. The status returned to the application is *1*. The triggering email is neither downloaded nor deleted from the mail server. Only the email header is returned to the application. See "Understanding PeopleSoft MultiChannel Framework Mail Classes" (PeopleTools 8.54: PeopleCode API Reference).
MCF_EmSz_IB	*500000*	Defines the PeopleSoft Integration Broker threshold, in bytes. This is the maximum size message that the GETMAILTARGET connector can send through PeopleSoft Integration Broker. The purpose of this threshold is to ensure that retrieved emails do not exceed the available memory on the gateway server and that the server running the application that requested the emails. A PeopleSoft Integration Broker message can contain one or more emails, depending on the number of emails the application retrieves per message. As soon as this threshold is reached, the remainder of the triggering email is written to the repository, and no more emails are retrieved. Emails fetched prior to the triggering email are returned normally. The triggering email is returned with a return status of *2,* indicating that some of its parts were written to the repository. See "Understanding PeopleSoft MultiChannel Framework Mail Classes" (PeopleTools 8.54: PeopleCode API Reference).

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Property Name	***Default Value***	***Description***
MCF_EmSz_Part	*30000*	Defines the text part threshold, in bytes. If any text part of an email exceeds this value , the part is routed to the attachment repository. This ensures that excessively large objects of text are not written to the database. For databases with limitations on the maximum size of rows or long text field lengths, this value must be lower than that limit to avoid SQL errors when saving emails.
MCF_Email_Lang_CD	*ENG*	Defines the expected language of emails downloaded by this node. This code is included in the email header returned to the application. Configure a node for each language you expect to handle, because Simple Mail Transfer Protocol (SMTP) mail servers and clients do not guarantee correct identification of the email's language when creating an email.
MCF_FetchSize	*819200*	Determines the number of bytes retrieved for an IMAP email in a single fetch. It also controls the size of the buffer (in bytes) used to write email attachments to the repository. Setting MCF_FetchSize to a higher number uses more memory, but it may also increase the speed of download for larger emails and attachments.
MCF_Force_Download_Attachments	*False*	Enables downloading email parts that might otherwise be interpreted as text. If set to True, all email parts, including text/plain, irrespective of their size, are downloaded to the attachment repository. This property enables reading of non-ASCII attachments. If set to False, email parts are evaluated individually, and either interpreted as email text or as an attachment.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Property Name	***Default Value***	***Description***
MCF_MethodName	*MessageCount*	Defines the methods associated with the application class package. The defined method is the default method used unless otherwise specified in the SyncRequest. The email application package always specifies the method to be used and does not reference this property. See "Understanding PeopleSoft MultiChannel Framework Mail Classes" (PeopleTools 8.54: PeopleCode API Reference).
MCF_Password	None	Defines the default password for the mailbox unless otherwise specified in the SyncRequest. See "Understanding PeopleSoft MultiChannel Framework Mail Classes" (PeopleTools 8.54: PeopleCode API Reference).
MCF_Port	*143*	Defines the mail server port to be used. By default, POP3 servers use port 110 and IMAP4 servers use port 143. Confirm the port number with your system administrator and set it accordingly here.
MCF_Protocol	*IMAP4*	Defines the protocol used by the connector to access emails on the mail server. Supported values are *POP3* or *IMAP4*.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Property Name	Default Value	Description
MCF_Quarantine	*Quarantine*	Defines a quarantine folder for email meeting the following criteria: • Connector size overflow. • Unsupported encoding. • Unknown Java exception. • JavaMail content error. • No attachment repository. • Mail parse exception. The quarantine folder is created for each user account and must be managed by each email user, not by system administrators. **Note:** Folders are supported only on IMAP4 mail servers.
MCF_Server	None	Defines the fully qualified host name of the mail server. This is the default fully qualified host name of the mail server, unless otherwise specified in the SyncRequest. For example, bigserver.peoplesoft.com. See "Understanding PeopleSoft MultiChannel Framework Mail Classes" (PeopleTools 8.54: PeopleCode API Reference).
MCF_User	None	Defines the user name for the email account (mailbox) being accessed. This is the default user name used unless otherwise specified in the SyncRequest. For example, if emails are addressed to support@peoplesoft.com, the user name is *support*.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Property Name	*Default Value*	*Description*
MCF_UseSSL	*N*	A value of *Y* indicates that SSL should be used. The SSL settings will be loaded from the GETMAILTARGET connector, including client certificate settings. For information on SSL connections, see Handling Email, SSL Connections. A value of *N* indicates that a default connection is preferred. No SSL settings will be loaded from the GETMAILTARGET connector, including no client certificate settings. **Note:** The default connection may be an SSL connection if the Email Server facilitates that connection.

Note: The POP3 and IMAP4 email protocols calculate message size differently. POP3 does not include header size, but IMAP4 does. As a result, message sizes affecting thresholds behave differently depending on the protocol used.
Multipurpose Internet Mail Extensions (MIME), is an Internet standard for representing multipart and multimedia data in email so that they can be exchanged between different email systems. The parts may be nested. PeopleSoft embeds the Sun JavaMail API to implement support for the MIME standard. However, all parts of an email are represented in PeopleSoft as level 1 rowsets, regardless of whatever hierarchy existed in the original email. Email clients determine the MIME format of the emails sent from them, so identical email content sent from different email clients may have a different MIME structures.

Related Links

"Understanding Managing Integration Gateways" (PeopleTools 8.54: Integration Broker Administration)

Enabling Virus Scanning

This section discusses how:

- Locate virusscan.xml file for your web server.
- Configure the virusscan.xml file.
- Use virus scan error logs.

Locating virusscan.xml File for Your Web Server

Virus scanning can be enabled for inbound MCF Email and AddAttachment, by configuring the virusscan.xml file for your virus scan engine. The location of this file depends on your web server:

- WebLogic:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

<PIA_HOME>/webserv/<domain>/applications/peoplesoft/PSIGW.war/WEB-INF/classes/psft/pt8/virusscan

- Websphere:

  <PIA_HOME>/webserv/<domain>/installedApps/<domain>NodeCell/<domain>.ear/PSIGW.war/WEB-INF/classes/psft/pt8/virusscan

See "Continue" (PeopleTools 8.54: PeopleCode Language Reference), "ConvertChar" (PeopleTools 8.54: PeopleCode Language Reference).

Configuring the virusscan.xml File

To enable the virus scanning feature:

1. Open VirusScan.xml.

```
<?xml version="1.0" encoding="UTF-8"?>
<Providers disableAll="True" logFile="./servers/PIA/logs/VirusScan%u.log">
    <!-- Sample Configuration for Symantec Engine
 <Provider>
        <name>Symantec</name>
      <class>psft.pt8.virusscan.provider.GenericVirusScanProviderImpl</class>
      <icapversion>ICAP/1.0</icapversion>
         <service-name>/SYMCScanResp-AV</service-name>
         <policycommand>?action=SCAN</policycommand>
         <address>152.68.144.44</address>
         <port>1344</port>
         <disable>false</disable>
    </Provider>-->

    <!-- Configure your own proivider -->
    <Provider>
    <!-- Provider Name of the Scan Engine -->
    <name></name>
    <!-- Provider Class of the Scan Engine.
         psft.pt8.virusscan.provider.GenericVirusScanProviderImpl is
         the default
         provider class.  -->
    <class>psft.pt8.virusscan.provider.GenericVirusScanProviderImpl</class>
    <!-- ICAP version -->
    <icapversion>ICAP/1.0</icapversion>
    <!-- ICAP ServiceName. The Service Name changes from Scan Engine to
   Scan Engine.
         This is the name Scan Engine Service is will be hosted with -->
    <service-name></service-name>
    <!-- RESPMOD extra commands, These are the RESPMOD commands
   (SEE ICAP Protocol).
         Usually these commands will be chainging from Engine to Engine
    -->
    <policycommand></policycommand>
    <!-- IP Address of Scan Engine host> -->
    <address></address>
    <!-- IP Port of Scan Engine host -->
    <port></port>
    <!-- Disable scanning for this provider -->
    <disable></disable>
    <!--
         Default codes = 200 and 204 for clean, 201,403 for infected
         Use these tags to change the behaivior if needed
         <clean>200,204</clean>
         <infected>201,403</infected>
         -->
  <virusheader></virusheadercheck>
</Provider>
```

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

```
</Providers>
```

Note: A sample configuration for Symantec Engine is provided in the remarks.

2. In the Providers tag, set the attribute *disableAll* to False .

```
<Providers disableAll="False" logFile="./servers/PIA/logs/VirusScan%u.log">
```

3. Multiple scan engine can be configured under <Providers>. Each <Provider> tag represents one scan engine. All configured scan engines will check for viruses. For each <Provider> tag enter values for the tags:

Tag	***Description***	***Example Value***
<name>	Provider Name of the Scan Engine	Symantec
<class>	Provider Class of the Scan Engine Default provider class is: `psft.pt8.virusscan.provider.GenericVirusScanProviderImpl`	`psft.pt8.virusscan.provider.GenericVirusScanProviderImpl`
<icapversion>	ICAP version	ICAP/1.0
<service-name>	Service name for the scan engine host.	/SYMCScanResp-AV
<policycommand>	Policy command used by the Scan Engine. Only SCAN is supported.	?action=SCAN
<address>	IP address of Scan Engine host.	IP address of the machine where the scan engine is running
<port>	IP port of Scan Engine host.	Port where the scan engine is running
<disable>	Disable scanning for this provider.	false
<clean>	Default codes = 200 and 204 for clean. You can use this tag to change the behavior if needed.	200,204
<infected>	Default codes = 201 and 403 for infected You can use this tag to change the behavior if needed.	201,403

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Tag	*Description*	*Example Value*
<virusheadercheck>	This tag contains comma separated pair of configurable header and error. This tag can be configured only for these two errors: INFECTED and SCANERROR. For example, <virusheadercheck>X-Violation::INFECTED, FileAttributeError::SCANERROR</virusheadercheck>. In this example, X-Violation will be checked in the response header from the Scan Engine and if found, INFECTED will be returned. If X-Violation is not found, FileAttributeError will be checked in the response header from the Scan Engine. If FileAttributeError is found, SCANERROR will be returned. **Note:** If error is not configured for X-Violation, INFECTED will be returned by default. **Note:** The <virusheadercheck> tag is not applicable when <clean> or <infected> is configured. **Note:** If the first header is found in the response header from the Scan Engine, the subsequent headers are not checked. **Note:** When <clean> or <virusheadercheck> is not configured, the default codes for clean (200, 204) and for infected (201, 403) will be checked in the response header and either CLEAN or INFECTED will be returned.	<virusheadercheck>X-Violation::INFECTED, FileAttributeError::SCANERROR</virusheadercheck>

Using Virus Scan Error Logs

There are two type of logs generated virus scanning logs and error logs.

Virus Scanning Logs

Virus Scanning logs are the only interface with the scanning engine. These logs are located in the path indicated by the *logfile* property in VirusScanning.xml. If there is a failure, the details will be logged *ig.errorLog.filename* in integrationGateway.properties

Error Logs

If there is a failure, the details will be logged *ig.errorLog.filename* in integrationGateway.properties For example, ig.errorLog.filename in <PIA_HOME>/webserv/<domain>/applications /peoplesoft/PSIGW.war/WEB-INF/integrationGateway.properties.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The return value when the virus scan for mail attachments is REPOSITORY_FAILURE = 8.

See "Error Messages Returned by MCFGetMail Class Methods" (PeopleTools 8.54: PeopleCode API Reference).

If there are any errors during file processing the error codes listed in this table will be generated.

Error Code	*Description*
%Attachment_ViolationFound	File violation detected by Virus scan engine.
%Attachment_VirusScanError	Virus scan engine error.
%Attachment_VirusConfigError	Virus scan engine configuration error.
%Attachment_VirusConnectError	Virus Scan engine connection error.

Demonstrating the Email Channel

To demonstrate the email channel, use the Email Sample Pages (MCFEM_DEMOERMS_CMP) component.

This section discusses how to:

- Use the GetMail - Server page.
- Use the MailStore - DB page.

The PeopleSoft system provides email sample pages to demonstrate the functionality of the PT_MCF_MAIL application package.

Note: The email sample pages are not intended for any purpose other than demonstration and troubleshooting. They should not be used in production.

The email sample pages can be also used to test the configuration of your integration gateway and MCF_GETMAIL node.

Note: Using the email sample pages requires access to a mail server.

Using the GetMail - Server Page

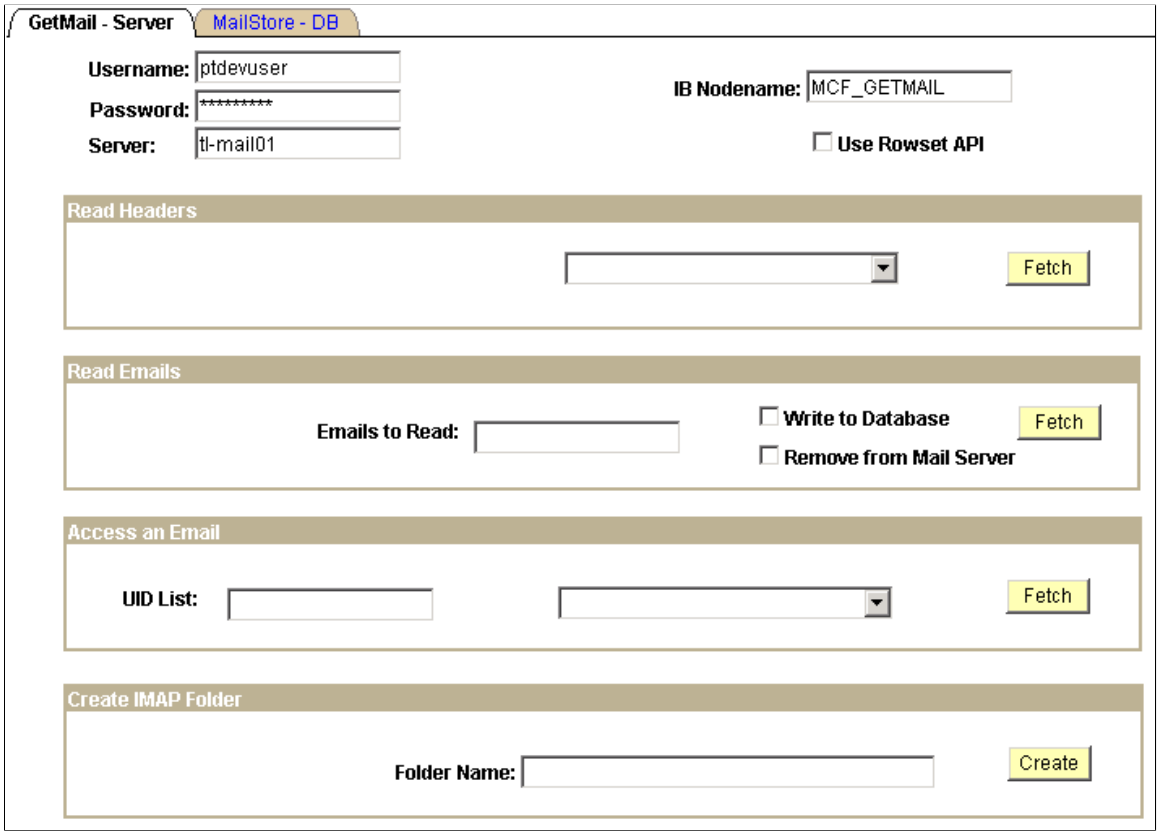
Access the GetMail - Server page using the following navigation path:

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

PeopleTools, MultiChannel Framework, Email, Sample Pages, GetMail - Server

Image: The GetMail - Server page having the following editable fields: User Name, Password, IB Nodename, Use Rowset API, Read Headers, Emails to Read, Write to Database, Remove from Mail Server, UID List, and IMAP Folder Name.

This example illustrates the fields and controls on the GetMail - Server page. You can find definitions for the fields and controls later on this page.



This page demonstrates reading or deleting emails from the mail server. It also demonstrates the storage and retrieval of emails to and from the database tables. Output of the response to each of the requests is written to a file at %PS_SERVDIR%/files/mcfdata.out on the application server. Check this file after each test for the output data.

To demonstrate email functionality, send an email to the user name and server that you enter on the GetMail - Server page.

Username and Password	Enter a valid username and password for an account on the mail server.
Server	Enter the mail server.
IB Nodename (Integration Broker node name)	Enter the PeopleSoft Integration Broker node used to access email from the mail server. The default is MCF_GETMAIL
Use Rowset API	Select to use the rowset-based GetMail API.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Deselect to use the new Mail Classes API.

Read Headers

Select a method to be called from the drop-down list box, then click Fetch. Select from:

Message Count: Writes the number of emails in the specified mailbox to the output file mcfdata.out.

ReadHeadersWithAttach: Writes headers and attachment information for all emails in the specified mailbox to the output file mcfdata.out.

Read Emails

Enter the number of emails to read, select other parameters, then click Fetch to read emails.

Emails to Read Enter the number of emails to retrieve from the specified mailbox and write to the output file mcfdata.out.

Write to Database Select to have the retrieved emails written to the database as well as to the output file mcfdata.out.

Remove from Mail Server Select to delete emails from the mail server after they have been retrieved .

Access an Email

Enter an email UID list, select a method from the drop-down list box, and then click Fetch to access the selected emails.

UID List (unique identifier list) Enter the unique ID of an email to be retrieved or deleted. To delete, you can enter a comma-delimited list of unique email IDs (UID) to be deleted.

Each email has a unique number (the UID) associated with it that is permanently guaranteed not to refer to any other email in the mailbox. To find the UID of an email, run the *ReadHeadersWith Attach* option first, which returns the UID for each email. If an invalid UID is specified, one empty row is returned.

Read Email w/ attachment Retrieves the specified email from the specified mailbox and writes it to the output file mcfdata.out.

Remove Message Deletes the specified email from the specified mailbox.

Create IMAP Folder

Create an email folder; only valid for IMAP4 mail servers.

Folder Name Enter the name of the folder to create.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

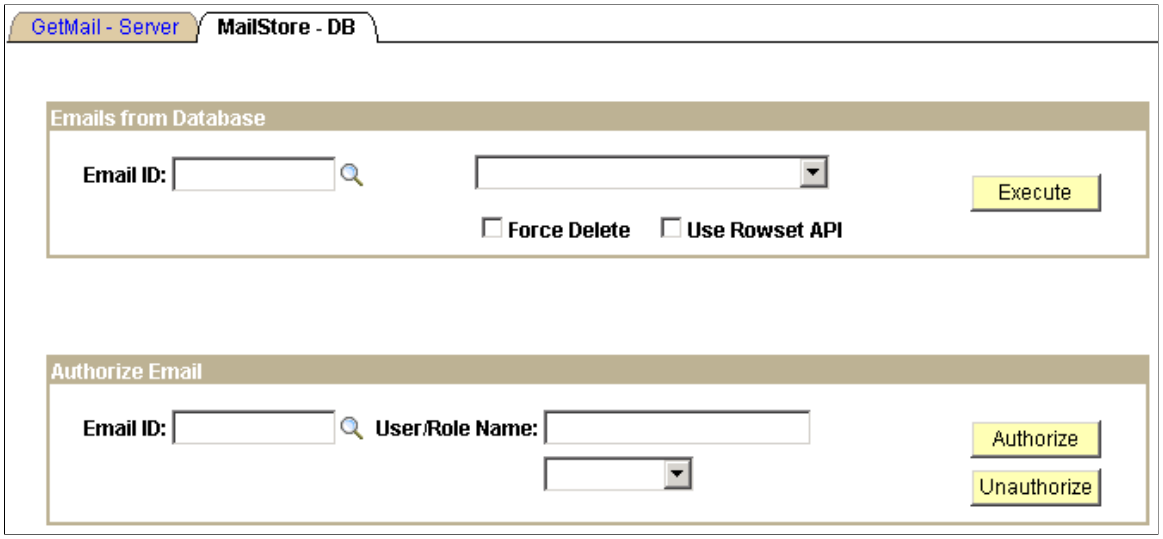
Using the MailStore - DB Page

Access the MailStore - DB page using the following navigation path:

PeopleTools, MultiChannel Framework, Email, Samples Pages, MailStore - DB

Image: The MailStore - DB page having the Emails from Database and Authorize Email group boxes.

This example illustrates the fields and controls on the MailStore - DB page. You can find definitions for the fields and controls later on this page.



View an example of how to access emails from the database (how to retrieve and delete email) and how to authorize email attachments for viewing or deletion using the PT_MCF_MAIL application package provided with PeopleSoft MCF.

Email ID Enter the PeopleSoft email ID (not to be confused with the UID) with which the email was saved to the database.

Emails from Database

Enter an email ID, select an action from the drop-down list box, and then click Execute to perform the selected action.

Delete Email from Database Deletes the specified email from the database and any corresponding attachments from the repository.

Retrieve Email Retrieves the specified email from the database and writes it to the output file mcfdata.out.

Force Delete Select after entering *Delete Email from Database* to force the deletion even if an error occurs when deleting associated attachments from the repository.

Use Rowset API Select to use the rowset-based GetMail API.

Deselect to use the new Mail Classes API.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Authorize Email

Enter an email ID, a user role or name, and click Authorize or Unauthorize to perform the specified action.

Email ID Enter the PeopleSoft email ID (not to be confused with the UID) with which the email was saved to the database.

User/Role Name Enter a PeopleSoft user ID or role to be authorized or unauthorized to view the attachment associated with the selected email.

After entering an ID or role, select *User* or *Role,* as appropriate, from the drop-down list box.

Related Links

Using the Email Sample Page

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Chapter 15

Configuring Instant Messaging in PeopleSoft MultiChannel Framework

Understanding Instant Messaging

PeopleSoft MCF extends the HTML chat functionality delivered in PeopleTools 8.42 to popular instant messaging networks. Users can initiate an outbound chat with a customer using instant messaging networks such as IBM/Lotus Sametime, or Yahoo! Messenger. This functionality enables an external user to participate in a business transaction through instant messaging.

PeopleSoft MCF enables instant messaging (IM) through public networks (GTALK, MSN, and Yahoo! Messenger), an enterprise Lotus Sametime Connect network or XMPP Instant Messaging server.

You can enable a user to initiate an instant messaging session from any page using either a button or PeopleCode.

See PeopleCode API Reference.

PeopleSoft users must install appropriate client software for supported instant messaging domain. However, users presence can also be detected without the installation of the client software.

Note: The instant messaging presence detection in PeopleTools is subject to the respective IM vendor's network availability. Oracle is not responsible for the availability or performance of any of the IM networks or for any change in an IM vendor's protocol that may render the IM Integration inoperable.

Configuring Instant Messaging Server Details

In MCF you can enable instant messaging for :

- GTALK
- SAMETIME
- XMPP
- YAHOO
- MSN

This section discusses how to configure instant messaging.

The supported public instant messaging networks such as Yahoo, GTALK, and XMPP require installation of client software. Installation and configuration of the client software is the responsibility of your administrators and customers, and is beyond the scope of this document.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Lotus Sametime Connect is an instant messaging network running on an enterprise Sametime/Domino server, which must be configured to support PeopleSoft MultiChannel Framework instant messaging. To administer instant messaging with Sametime Connect, install Sametime Links and configure the server to enable anonymous users to connect to the server and query for the presence of other known users. Refer to your Lotus/Domino server documentation for more information.

For XMPP Instant Messaging servers, you will need to configure the XMPP domain in order to detect user presence, whereas other domains are configured by default.

Important! Google no longer supports chatback badges, so MCF does not use chatback badges for presence detection in GTALK. MCF supports GTALK only through XMPP. You may configure GTALK over XMPP to enable MCF to detect user presence and to initiate the chat client. The GTALK user presence detection works only for the contacts on the contact (buddy) list. After MCF initiates a chat, GTALK is responsible for maintaining the chat session.

Important! Microsoft has retired Messenger and no longer supports any presence detection features, so MCF does not support presence detection in MSN.

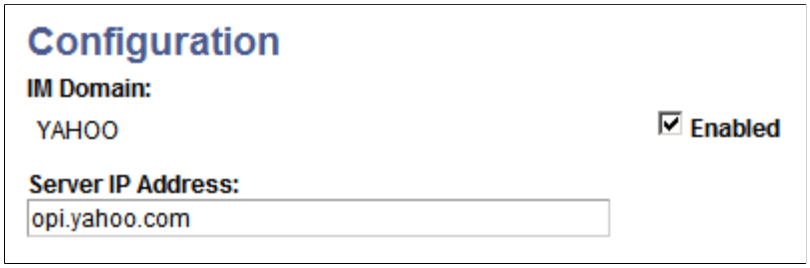
Important! Skype no longer supports presence detection feature for it's contact list, hence MCF does not support presence detection in Skype.

Configuration for Yahoo

Select PeopleTools, MultiChannel Framework, Instant Messaging, Configuration.

Image: Configuration page

This example illustrates the fields and controls on the Configuration page. You can find definitions for the fields and controls later on this page.



IM Domain The instant messaging service name.

Server IP Address The host name and port (if applicable) of the instant messaging service.

Warning! Do not alter the server address for Yahoo or GTALK . The server addresses for Yahoo, MSN, XMPP, and GTALK are preconfigured. However, you must enter the address for your Sametime Connect server.

Enabled Select Enabled to enable the configured instant messaging service.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

If the enabled flag is deselected, and the network is available, the buttons will always show as offline.

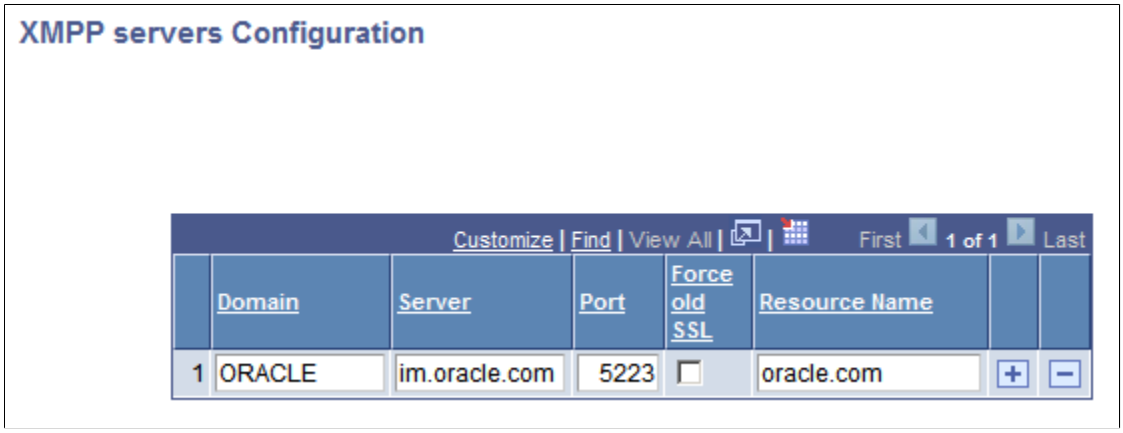
Configuring XMPP Domains

To configure XMPP servers:

1. Select PeopleTools, MultiChannel Framework, Instant Messaging, Configuration.
2. Click the Search button and select XMPP.
3. Click the Configure XMPP Servers link from the Configuration page.

Image: XMPP servers Configuration page

This example illustrates the fields and controls on the XMPP servers Configuration page. You can find definitions for the fields and controls later on this page.



Configure as many XMPP domains as required.

Domain	Enter IM server domain.
Server	Enter server information.
Port	Enter the port for the IM server.
Force old SSL	Select to use old SLL port 5223.
Resource Name	Resource name for the domain name of the XMPP server. For example: oracle.com.

Configuring IM Users for XMPP Servers

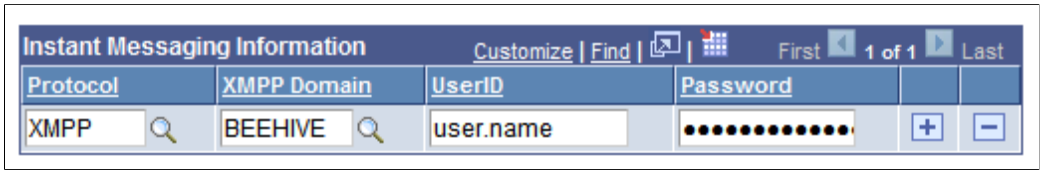
In order to obtain user presence information from configured XMPP IM servers, the user must login to the server. Each user needs to enter their userid and password for the XMPP server. If XMPP login information has been entered in user profile, the login process to the configured XMPP server will happen automatically when the PeopleSoft user signs on to PeopleSoft. This login process takes place in the background and will not stop the user from performing other operations.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Select My System Profile or PeopleTools, Security, User Profile, User Profile and click the Instant Messaging Information link.

Image: Instant Messaging Information page

This example illustrates the fields and controls on the Instant Messaging Information page.



Enter your user ID and password information for each XMPP domain you are using.

It is necessary to relogin to the application after configuring the settings.

Note: Username without domain has to be entered under UserId field. Example: ‘firstname.lastname’ for firstname.lastname@oracle.com.

Using the Instant Messaging Sample Pages

To demonstrate instant messaging, use the Instant Messaging Sample Pages (MCF_IM_DEMO_CMP) component.

This section discusses how to:

- Use the PeopleCode Sample page.
- Use the Button Sample page.

Pages Used to Test Instant Messaging

Below are the details:

Page Name	*Definition Name*	*Navigation*	*Usage*
PeopleCode Sample	MCF_IM_DEMO_PG	PeopleTools, MultiChannel Framework, Instant Messaging, Sample Pages, PeopleCode Sample	Set up buddy lists and view presence for configured instant messaging clients
Button Sample	MCF_IM_DEMO1_PG	PeopleTools, MultiChannel Framework, Instant Messaging, Sample Pages, Button Sample	View presence for configured instant messaging clients using the instant messaging button.

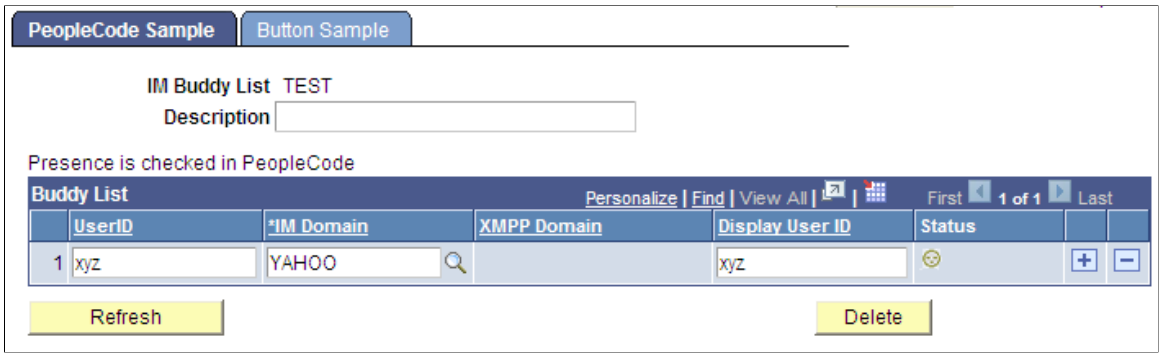
Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Using the PeopleCode Sample Page

Select PeopleTools, MultiChannel Framework, Instant Messaging, Sample Pages and add a new value.

Image: PeopleCode Sample page

This example illustrates the fields and controls on the PeopleCode Sample page. You can find definitions for the fields and controls later on this page.



Note: The PeopleCode Sample does not support presence detection for GTALK and XMPP domains.

The PeopleCode Sample page demonstrates the use of instant messaging PeopleCode. The sample pages are intended for demonstration purposes and should not be used in production.

See "Understanding the MCFIMInfo Class" (PeopleTools 8.54: PeopleCode API Reference).

IM Buddy List Displays an identifier for the list.

Description Enter a description for the list.

User ID Enter the user ID.

The user ID must be that of a valid user of the instant messaging service specified in the IM Domain field.

See Determining your Screen Name.

IM Domain Select the instant messaging service associated with the specified user ID from the drop-down list box.

XMPP Domain Select IM server domain.

Display User Name Name used for display purpose.

Status An icon displays the presence status of the specified user. The icons vary with the instant messaging service.

Click the icon to initiate an instant messaging session with the selected user. The user must be online.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: User presence is not updated automatically. Use the Refresh Presence button to check the current status of Yahoo users.

Refresh Presence Click the Refresh Presence button to check the status of Yahoo users.

Note: If you get a warning symbol with the network, even though you have specified the correct address of IM server, provide values for Proxy host and Proxy port in the pasappsrv.cfg file.

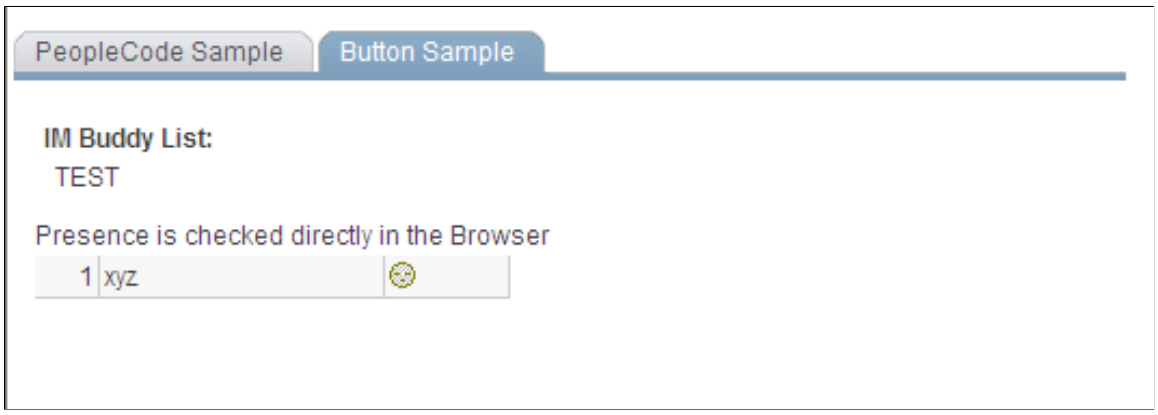
Using the Button Sample Page

Access the Button Sample page using the following navigation path:

PeopleTools, MultiChannel Framework, Instant Messaging, Sample Pages, Button Sample

Image: Button Sample page

This example illustrates the fields and controls on the Button Sample page. You can find definitions for the fields and controls later on this page.



The Button Sample page demonstrates the use of the instant messaging button implemented in PeopleSoft Application Designer. The sample pages are intended for demonstration purposes and should not be used in production.

See "Specifying Type Properties for Push Buttons or Links" (PeopleTools 8.54: Application Designer Developer's Guide).

IM Buddy List Displays an identifier for this list.

Description Enter a description for this list.

User and Network Displays user ID and instant messaging service ID.

Status Displays the presence status of the specified user.

Click the icon to initiate an instant messaging session with the selected user. You can send an instant message even if the user is not online.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: For the SAMETIME icon to be visible, do not select the Sun JVM in the browser. If Sun JVM is selected, the icon does not display in the sample page.

Developing an Application with Instant Messaging Action Button for Presence Detection

In order to create an application with instant messaging action button for presence detection:

1. Create a page and add a button.
2. Click on properties of a push button and choose *Instant Messaging Action* for the destination field.
3. Choose a record and field name where screen name is in the format for the instant messaging domain, as listed in this table:

Instant Messaging Domain	***Required format of screen name***
YAHOO	screenname@YAHOO
XMPP	screenname@DOMAIN@XMPP
GTALK	screenname@GTALK

4. Save page.

Using Single Button Presence

The single presence icon represents a collective presence of all screen names. When a user moves the mouse pointer over a single presence icon, it will open a menu with all user screen names and their presence

This section provides an overview of presence detection and discusses how to:

- Configure servers.
- Configure screen names.
- Test single presence button.
- Develop an application with Single Presence button for presence detection.

Understanding Presence Detection

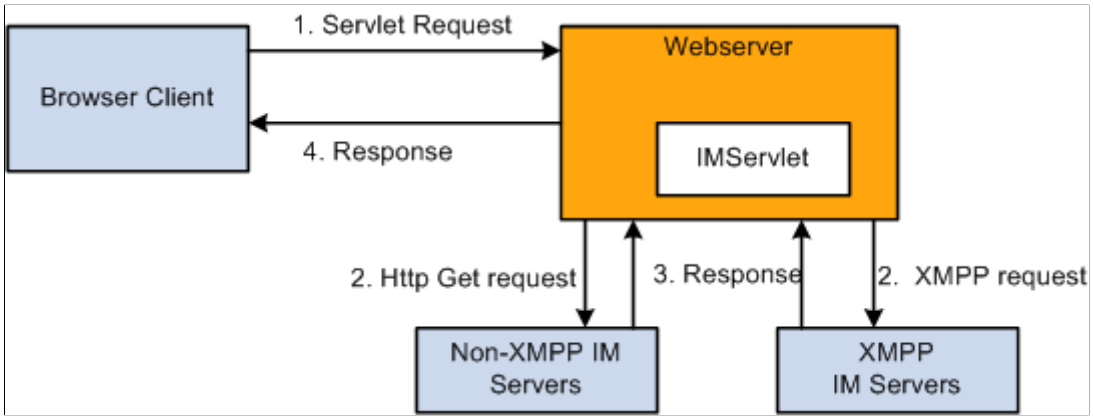
The IM Servlet is implemented and hosted in the web server for the purpose of communicating with IM servers to get presence information. Therefore, browsers can send http Get request to IM Servlet and get presence information. The IM Servlet directly communicates with non-XMPP IM servers to fetch presence information and uses Smack API to communicate with XMPP servers. To improve the performance and response time the IM Servlet supports bulk requests for getting presence of multiple

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

screen names using one request. In order to get the presence of a buddy from an XMPP server, the user needs to login into the XMPP server and maintain an active session. For non-XMPP servers the user does not need to login into IM server to fetch the presence information. This active session is maintained in the IM Servlet.

Image: Instant messaging architecture

This diagram shows the Instant messaging architecture.



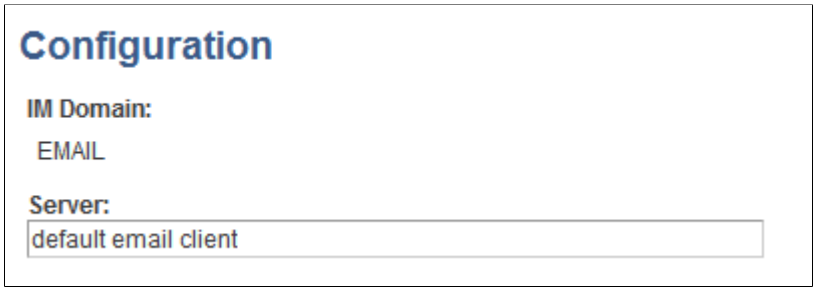
1. Browser requests to get the presence from IM Servlet
2. For non-XMPP servers, IM Servlet will send http get request to corresponding Instant Messaging server to find out presence information. For XMPP servers, IM Servlet will contact XMPP server via smack API to get presence information.
3. Presence information is returned to IM Servlet.
4. Presence is sent to the client browser.

Instant Messaging Servers Configuration for Single Presence Button

PeopleTools, MultiChannel Framework, Instant Messaging, Presence, Servers Configuration.

Image: Configuration page

This example illustrates the fields and controls on the Configuration page.



The configuration page is the same page used to configure instant messaging servers. When you select the page under the Presence menu, you can also configure Email, MSN and Skype.

Related Links

Configuration for Yahoo

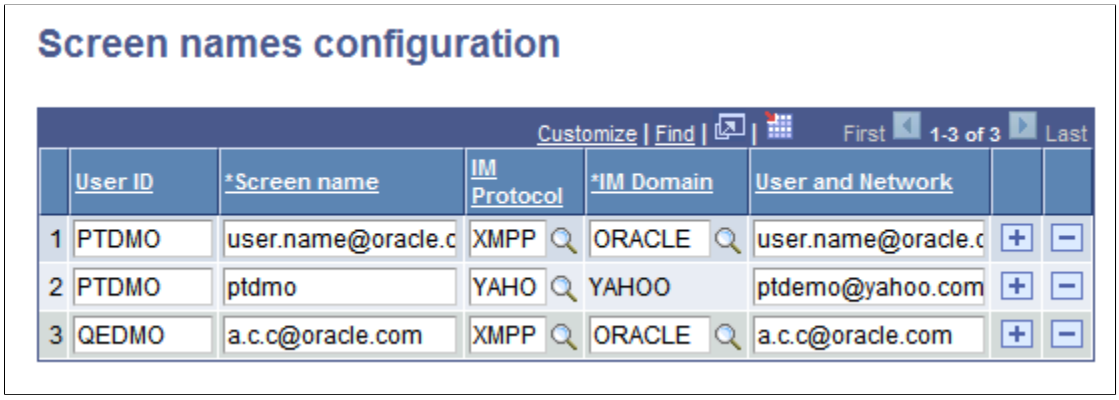
 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Configuring Screen Names

Select PeopleTools, MultiChannel Framework, Instant Messaging, Presence, Screen names configuration.

Image: Screen names configuration page

This example illustrates the fields and controls on the Screen names configuration page. You can find definitions for the fields and controls later on this page.



User ID Enter PeopleSoft user ID.

Screen name ID used to obtain presence for this user.

IM Protocol Select the protocol.

IM Domain Select the instant messaging domain.

User and Network The value entered here is used for display purposes in Single Button Presence.

Determining your Screen Name

The value of the ScreenName column will vary, depending on your Instant Messaging domain.

IM Domain	*User ID Description*	*Email ID*	*Corresponding ScreenName*
YAHOO	User name without domain	abc.xyz@yahoo.co.in	abc.xyz
MSN	MSN badge ID	p_abc@hotmail.com	f2a66c8e51870b5
XMPP	User name with domain	abc.xyz@oracle.com	abc.xyz@oracle.com

Creating a MSN Badge

To create MSN user IDs:

1. Use your browser to point to the following URL: http://settings.messenger.live.com/applications/WebSettings.aspx

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

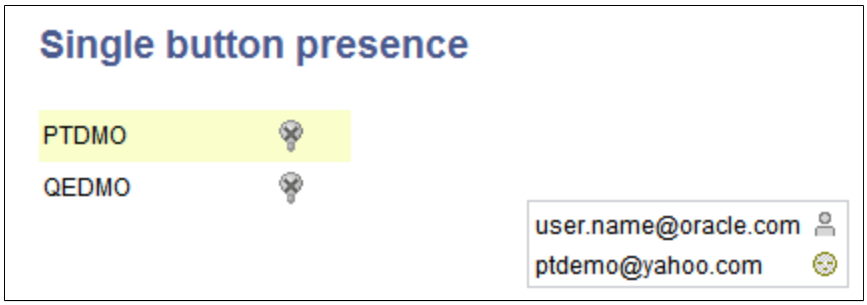
2. Sign on with MSN Credentials
3. Select the option Allow anyone on the web to see my presence and send me messages. Note: Any website can show your status and people you have blocked may see your presence on the web.
4. Click Save
5. Verify the MSN badge: <iframe src="http://settings.messenger.live.com/Conversation/IMMe.aspx?invitee=8e109e9602507581@apps.messenger.live.com&mkt=en-in" width="300" height="300" style="border: solid 1px black; width: 300px; height: 300px;" frameborder="0" scrolling="no"></iframe>
6. Select the Create HTML tab, and view the HTML appearing at the bottom of the page. The badge value appears after “invitee=” and ” @apps”. The system uses the badge to determine the user’s presence information which is also used as the MSN user ID.

Testing Single Presence Button

Select PeopleTools, MultiChannel Framework, Instant Messaging, Presence, Sample page.

Image: Single button presence page

This example illustrates the fields and controls on the Single button presence page.



The single presence icon represents a collective presence of all screen names. When a user moves the mouse pointer over a single presence icon, it will open a menu with all user screen names and their presence. If any of the screen names are online, you can click the online icon and it will launch the corresponding IM Messenger to send IM messages. When the user clicks on an email icon, it will launch default configured email client. When you click on Skype, it will launch Skype application and allow you to login to Skype and if skype is not installed there appears a dialog box with two buttons to download or upgrade skype

Developing a Sample Application with Single Presence Button

To develop a sample application with Single Presence Button:

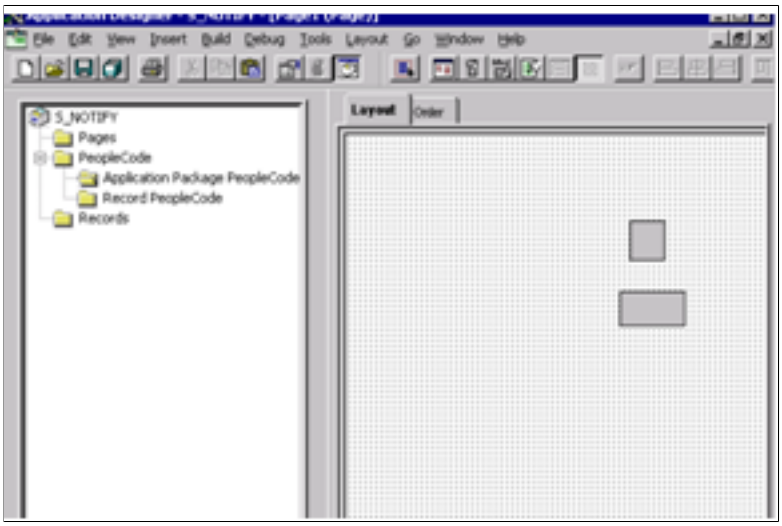
1. Create a new page with two hidden HTML areas, one Hidden HTML area(as in, MCF_XMPPHEADER) is for keeping Javascript content, and the other HTML area (as in, MCF_IM_HTMLAREA) is for showing the presence icon.
2. Ensure that all screen names are in the required format and that they are available in a record (as in,MCF_IMSCRNAMES).

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

3. Write page PeopleCode for the page activate event:

   Image: Application Designer — page activate event

   This example illustrates the page activate event in Application Designer for writing PeopleCode.




```
import PT_MCF_IM:*;

Global PT_MCF_IM:MCFIMSingleButton &single;
Local Rowset &RS1;

&single = create PT_MCF_IM:MCFIMSingleButton();
/* Number of milliseconds that we want to show sliding window even after we   */
/* remove cursor from single presence button.                                 */

&single.hideDelay = 5325;
&RS1 = CreateRowset(Record.MCF_IMSCRNNAMES);
/* &RS1 have all screen names, right is direction of sliding window. 190 is   */
/* X coordinate of top left corner of sliding window, 200 is Y coordinate of  */
/* top left corner of sliding window, 350 is height of sliding window, 450    */
/* is width of sliding window,  1 is some unique no of sliding window on that */
/* page. If there are many single presence icons on page, then it would be    */
/* useful for uniqness.                                                       */

RECORDNAME.MCF_IM_HTMLAREA.Value = &single.generateHTML(&RS1, "right", 190, 200, 35⇒
0, 450, 1);
RECORDNAME.MCF_XMPPHEADER = &single.generateJavaScript();
```

See "MCFIMSingleButton Class Methods" (PeopleTools 8.54: PeopleCode API Reference).

Installing Instant Messenger Clients

MSN is launched through a web client. It is not required to install the MSN messenger.

GTALK needs GTALK chat client to be installed on the client machine.

Yahoo needs Yahoo messenger to be installed on the client machine and can also be configured in Pidgin.

XMPP needs Pidgin client installed on client machine.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Working with Domain Offline and Online Buttons

This table shows the various state of your Instant Messenger:

Instant Messenger	*State*	*Description*
XMPP	Online	Clicking the online icon of XMPP will launch Pidgin if it is installed in the local machine. Chat Window is opened if logged in to Pidgin. If Pidgin is not installed, the message webpage cannot be displayed appears in the browser. **Note:** In certain browsers, pop up window will be opened before pidgin is launched.
	Offline	Clicking the offline icon of XMPP will launch Pidgin if it is installed in the local machine. It will open a chat window if logged in to Pidgin already. An error will be displayed if pidgin is not installed.
Yahoo	Online	Clicking the online icon of YAHOO will launch Yahoo Messenger if it is installed in local machine. It will open a chat window if logged in to Yahoo Messenger already and if the user is not logged it will prompt to login to Yahoo.. If Yahoo is not installed, it will not open a chat window but if Yahoo is registered with Pidgin, it opens a chat window in Pidgin.
	Offline	Clicking the offline icon of YAHOO will launch Yahoo Messenger if it is installed in local machine. It will open a chat window if logged into Yahoo Messenger already and it will allow you to send offline messages. If Yahoo is not installed and registered with Pidgin, it opens a chat window in Pidgin allowing you to send offline messages.
MSN	Online\Offline	Clicking the online icon of MSN will launch MSN webchat window. Clicking on the Offline icon of MSN will launch webchat, however, user will not be able to send offline message.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Instant Messenger	*State*	*Description*
Google	Online\Offline	Clicking the online GTALK icon will launch the installed GTALK chat client. Clicking the offline GTALK icon will launch the installed GTALK chat client window and will display the status of the particular contact.
Email icon		Clicking the email icon will launch default configured Email client.
Voice icon		Clicking the VOICE icon will launch Skype if it is installed in the local machine.

Debugging IM

When troubleshooting or monitoring the IMServlet, you can view the IMServlet.log to gather information specific to the IMServlet.

Viewing IMServlet Logs

The log file is located in:

PIA_HOME\servers\PIA\logs

Since IMServlet is using web server logging, IM Log configuration can be done in the following location:

PIA_HOME\piaconfig\properties\ logging.properties

Levels OFF, INFO, SEVERE can be set via com.peoplesoft.pt.mcf.im.level according to the needs.

By default, log is set to SEVERE as the following: com.peoplesoft.pt.mcf.im.level = SEVERE

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Appendix A

JSMCAPI Quick Reference

JSMCAPI Classes

This section lists the JSMCAPI classes in alphabetical order, along with the constructor, fields, methods, callback event methods, and returns associated with each class.

_Address

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
_Address()	_Address object

Fields

Field	***Type***
caps	Object
id	String

Callback Event Methods

Callback Event Method	***Returns***
onError(*event*)	None

_UQAddress

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

The _UQAddress class extends the _Address class.

See _Address.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	***Returns***
_UQAddress()	_UQAddress object

Fields

Field	***Type***
tasks	Associative array

Methods

Method	***Returns***
acceptTask(*task*, *reason*)	None
dequeueTask(*task*)	Object

Callback Event Methods

The _UQAddress class inherits the onError callback event method from the _Address class.

See _Address.

Callback Event Method	***Returns***
onAccepted(*event*)	None
onAcceptingTask(*event*)	None
onDequeueingTask(*event*)	None
onNotify(*event*)	None
onTaskAdded(*event*)	None
onTaskRemoved(*event*)	None
onUnassigned(*event*)	None

_User

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	*Returns*
_User()	_User object

Fields

Field	*Type*
agentId	String
caps	Associative array
id	String
name	String
presences	Associative array
states	Associative array of the constants beginning with ST_*.
ST_LOGGEDIN	String
ST_LOGGEDOUT	String
ST_NOTREADY	String
ST_READY	String
ST_UNKNOWN	String
ST_WORKNOTREADY	String
ST_WORKREADY	String
statistics	Object
statistics1	Object
statistics2	Object

A2AChat

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

The A2AChat class extends the _Address class.

See _Address.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	***Returns***
A2AChat(*event*, *address*, *chatType*)	A2AChat object

Fields

Field	***Type***
address	Object
agentId	String
agentName	String
appData	Object
chatType	String
customerName	Object
id	String
isConference	Object
jr	String of the PS_JR constant.
type	String of the following constants: • TYPE_ANSWER • TYPE_CONSULT
uniqueueId	String

Methods

Method	***Returns***
getUrl(*defaultUrl*)	String

A2AChatAddress

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	***Returns***
A2AChatAddress()	A2AChatAddress object

Fields

The A2AChatAddress class inherits the following fields from the _Address class:

- id
- caps

See _Address.

Field	***Type***
id	String
tasks	Associative array

Methods

Method	***Returns***
initiateChat(*agentId*)	None

Callback Event Methods

The A2AChatAddress class inherits the onError callback event method from the _Address class.

See _Address.

Callback Event Method	***Returns***
onChatEnded(*event*)	None
onInitiatingChat(*event*)	None
onNotify(*event*)	None

AgentStatistics

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	*Returns*
AgentStatistics()	AgentStatistics object

Fields

Field	*Type*
averageCallDuration	Number
averageHoldDuration	Number
callsHandled	Number
data	Associative array
percentIdleTime	Number
percentTimeAvailable	Number
percentTimeInCurrentState	Number
percentTimeUnavailable	Number
taskAcceptedCurrentLogin	Number
timeCurrentLogin	Number
timeWorking	Number
totalTaskAcceptedLogin	Number
totalTaskDoneLogin	Number
totalTaskDoneLogin	Number
unavailableDuration	Number
waitDuration	Number

AppData

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	***Returns***
AppData()	AppData object

Fields

Field	***Type***
data	Associative array
groupId	String
jr	String
question	String
strData	String
subject	String
uniqueId	String
url	String of the constant URL
userId	String
username	String
wizUrl	String

Methods

Method	***Returns***
addKeyValue(*key*, *value*)	None

Buddy

This section lists the constructor, methods, and callback event methods associated with this class.

The Buddy class extends the _User class.

See _User.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	*Returns*
Buddy()	Buddy object

Fields

The Buddy class inherits the following fields from the _User class:

- agentID
- caps
- id
- name
- presences
- states
- statistics
- statistics1
- statistics2

See _User.

Callback Event Methods

Callback Event Method	*Returns*
onStat1(*event*)	None
onStat2(*event*)	None
onState(*event*)	None

Call

This section lists the constructor, fields, and returns (if applicable) associated with this class.

The Call class extends the Task class.

See Task.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	***Returns***
Call(*strCall*)	Call object

Fields

The Call class inherits the following fields from the Task class:

- caseid
- cost
- customerid
- group
- id
- onStat
- priority
- type
- urlAbs
- urlRel

See Task.

Field	***Type***
line	Object
statistics	Object
userData	String

CallData

This section lists the constructor, fields, methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
CallData()	CallData object

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Fields

Field	*Type*
ani	String
callId	String
callType	String
data	String
dnis	String

Methods

Method	*Returns*
addKeyValue(*key* , *value*)	None

CallStatistics

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Constructor

Constructor	*Returns*
CallStatistics()	CallStatistics object

Fields

Field	*Type*
data	Associative array
holdTime	String
queueTime	String
talkTime	String

Chat

This section lists the constructor, fields, methods, and returns (if applicable) associated with this class.

The Chat class extends the Task class.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

See Task.

Constructor

Constructor	***Returns***
Chat(*event*, *address*)	Chat object

Fields

The Chat class inherits the following fields from the Task class:

- caseid
- cost
- customerid
- group
- id
- onStat
- priority
- type
- urlAbs
- urlRel

See Task.

Field	***Type***
address	Object
agentId	String
appData	Object
chatconnection	Object
chatType	String
customerName	String
groupId	String
question	String
statistics	Object

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Field	*Type*
subject	String
userData	Object

Methods

Method	*Returns*
gettpUrl(*defaultUrl*)	String
getUrl(*defaultUrl*)	String

ChatAddress

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

The ChatAddress class extends the _UQAddress class.

See _UQAddress.

Constructor

Constructor	*Returns*
ChatAddress()	ChatAddress object

Fields

The ChatAddress class inherits the following fields from the _Address class:

- id
- caps

Field	*Type*
chatconnections	Object
tasks	Object

See _Address.

The ChatAddress class inherits the tasks field from the _UQAddress class.

See _UQAddress.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Methods

The ChatAddress class inherits the following methods from the _UQAddress class:

- acceptTask
- dequeueTask

Method	***Returns***
chat (*agentId*, *buddyId*, *reason*)	None
getChatconnectionByConnectionId (*connectionId*)	Object
getChatconnectionindexByConnectionId (*connectionId*)	Index
getFreeChatconnection	Object
getFreeChatconnectionIndex	Object

See _UQAddress.

Callback Event Methods

The ChatAddress class inherits the onError callback event method from the _Address class.

See _UQAddress.

The ChatAddress class inherits the following callback event methods from the _UQAddress class:

- onAccepted
- onAcceptingTask
- onDequeueingTask
- onNotify
- onTaskAdded
- onTaskRemoved
- onUnassigned

See _UQAddress.

Callback Event Method	***Returns***
onCapabilitiesChanged (*event*)	None

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

ChatConnection

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
ChatConnection(*event*)	ChatConnection object

Fields

Field	***Returns***
caps	Object
chat	String
connectionId	String
id	String
state	String

Methods

Method	***Returns***
answer (*agentid*, *reason*)	None
conference (*agentid*, *reason*)	None
forward (*fromagentid*, *toagentid*, *qid*, *userdata*, *chatdata*, *reason*)	None
gethistory (*agentid*, *noofLines*, *reason*)	None
getUrl (*deafultURL*)	String
message (*agentid*, *message*, *reason*)	None
pushURL(*URL*)	None
reject(*agentid*, *reason*)	None
release(*agentid*, *reason*)	None
typing (*agentid*, *reason*)	None

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Callback Event Methods

Callback Event Method	***Returns***
onAccepted (*event*)	None
onAnswering (*event*)	None
onCapabilitiesChanged (*event*)	None
onChatdataChanged (*event*)	None
onConferencing (*event*)	None
onDialing (*event*)	None
onDropped (*event*)	None
onError (*event*)	None
onForwarded (*event*)	None
onForwardError (*event*)	None
onForwarding (*event*)	None
onHistory (*event*)	None
onIncomingChat (*event*)	None
onMessage (*event*)	None
onPartyAdded (*event*)	None
onPartyChanged (*event*)	None
onPartyRemoved (*event*)	None
onProperties (*event*)	None
onPushURL (*event*)	None
onRejected (*event*)	None
onReleased (*event*)	None
onReleasing (*event*)	None
onRevoked (*event*)	None
onTalking (*event*)	None

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Callback Event Method	***Returns***
onTyping (*event*)	None
onUserdataChanged (*event*)	None

ChatConnectionCaps

This section lists the constructor, fields, methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
ChatConnectionCaps(*event*)	ChatConnectionCaps object

Fields

Field	***Type***
canAnswer	Boolean
canConference	Boolean
canConferenceSingle	Boolean
canForward	Boolean
canIndicateTyping	Boolean
canPushURL	Boolean
canReject	Boolean
canSendMessage	Boolean

ChatData

This section lists the constructor, fields, methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
ChatData(*event*)	ChatData object

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Fields

Field	***Type***
data	Object

Methods

Method	***Returns***
addKeyValue(*key*, *value*)	None

Connection

This section lists the constructor, methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
Connection (*event*)	ConnectionEvent object

Methods

Method	***Returns***
eventUserDataChanged(*event*)	None

ConnectionListener

This section lists the constructor, methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
ConnectionListener (*event*)	ConnectionListener object

Methods

Method	***Returns***
requestAttachUserdata(*object*, *request*)	None

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

ConnectionRequest

This section lists the constructor, methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
ConnectionRequest (*event*)	ConnectionRequest object

Methods

Method	***Returns***
getUserData( )	UserData

Email

This section lists the constructor, fields, methods, and returns (if applicable) associated with this class.

The Email class extends the Task class.

See Task.

Constructor

Constructor	***Returns***
Email(*event*)	Email object

Fields

The Email class inherits the following fields from the Task class:

- caseid
- cost
- customerid
- group
- id
- onStat
- priority
- type
- urlAbs

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- urlRel

See Task.

Field	***Type***
address	Object
agentId	Object
appData	Object
customerName	String
emailconnection	Object
emailId	String
groupId	String
question	String
statistics	Object
subject	String
userData	Object

Methods

Method	***Returns***
gettpUrl(*defaultUrl*)	Object
getUrl(*defaultUrl*)	String

EmailAddress

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

The EmailAddress class extends the _UQAddress class.

See _UQAddress.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	***Returns***
EmailAddress()	EmailAddress object

Fields

The EmailAddress class inherits the following fields from the _Address class:

- id
- caps

See _Address.

The EmailAddress class inherits the tasks field from the _UQAddress class.

See _UQAddress.

Field	***Type***
agent	Object
emailconnections	Object

Methods

The EmailAddress class inherits the following methods from the _UQAddress class:

- acceptTask
- dequeueTask

See _UQAddress.

Method	***Returns***
getEmailconnectionByConnectionId (*connectionId*)	Object
getEmailconnectionindexByConnectionId (*connectionId*)	Index
getFreeEmailconnection	Object

Callback Event Methods

The EmailAddress class inherits the onError callback event method from the _Address class.

See _Address.

The EmailAddress class inherits the following callback event methods from the _UQAddress class:

- onAccepted

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- onAcceptingTask
- onDequeueingTask
- onNotify
- onTaskAdded
- onTaskRemoved
- onUnassigned

See _Address.

EmailConnection

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
EmailConnection(*event*)	EmailConnection object

Fields

Field	***Type***
caps	Object
connectionId	String
email	String
id	String
state	String

Methods

Method	***Returns***
abandon (*reason*)	None
answer (*reason*)	None
complete (*reason*)	None

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Method	***Returns***
forward (*fromagentid*, *toagentid*, *qid*, *userdata*, *Emaildata*, *reason*)	None
reject(*agentid*, *reason*)	None
withdraw_RES (*reason*)	None

Callback Event Methods

Callback Event Method	***Returns***
onAnswering (*event*)	None
onCapabilitiesChanged (*event*)	None
onCompleted (*event*)	None
onDropped (*event*)	None
onEmaildataChanged (*event*)	None
onError (*event*)	None
onForwarded (*event*)	None
onForwardError (*event*)	None
onForwarding (*event*)	None
onIncoming (*event*)	None
onProcessing (*event*)	None
onRejected (*event*)	None
onRevoked (*event*)	None
onUserdataChanged (*event*)	None
onWithdraw_REQ (*event*)	None

EmailConnectionCaps

This section lists the constructor, fields, methods, and returns (if applicable) associated with this class.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	***Returns***
EmailConnectionCaps(*event*)	EmailConnectionCaps object

Fields

Field	***Type***
canAnswer	Boolean
canComplete	Boolean
canForward	Boolean
canReject	Boolean

EmailData

This section lists the constructor, fields, methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
EmailData(*event*)	EmailData object

Fields

Fields	***Type***
data	Object

Methods

Method	***Returns***
addKeyValue(*key*, *value*)	None

Extension

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

The Extension class extends the _Address class.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

See _Address.

Constructor

Constructor	***Returns***
Extension(*numOfLines*)	Extension object

Fields

The Extension class inherits the following fields from the _Address class:

- id
- caps

See _Address.

Field	***Type***
forwardMode	Object
isDnd	Boolean
lines	Associative array
numOfLines	Number

Methods

Method	***Returns***
cancelDnd(*reason*)	None
cancelForwardSet(*reason*)	None
forwardSet(*number*, *mode*, *reason*)	None
getDialingLine()	Object
getFreeLine()	Object
getLineByConnectionId(*connectionId*)	Object
getOffHookLine()	Object
setDnd(*reason*)	None

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Callback Event Methods

The Extension class inherits the onError callback event method from the _Address class.

See _Address.

Callback Event Method	***Returns***
onCancelingDnd(*event*)	None
onCancelingForward(*event*)	None
onDnd(*event*)	None
onDndCanceled(*event*)	None
onForwardCanceled(*event*)	None
onForwarded(*event*)	None
onForwarding(*event*)	None
onSettingDnd(*event*)	None

ExtensionCaps

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
ExtensionCaps(*strCaps*)	ExtensionCaps object

Fields

Field	***Type***
canCancelDnd	Boolean
canDial	Boolean
canFwdBusy	Boolean
canFwdBusyNoAnswer	Boolean
canFwdCancelForward	Boolean
canFwdDefault	Boolean

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Field	*Type*
canFwdNoAnswer	Boolean
canFwdUnconditional	Boolean
canRefreshState	Boolean
canSetDnd	Boolean

ForwardMode

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Constructor

Constructor	*Returns*
ForwardMode()	ForwardMode object

Fields

Field	*Type*
mode	String of the following constants: • BUSY • BUSYNOANSWER • DEFAULT • NOANSWER • NONE • UNCONDITIONAL

GenericAddress

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

The GenericAddress class extends the _UQAddress class.

See _UQAddress.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	***Returns***
GenericAddress()	GenericAddress object

Fields

The GenericAddress class inherits the following fields from the _Address class:

- id
- caps

See _Address.

The GenericAddress class inherits the tasks field from the _UQAddress class.

See _UQAddress.

Field	***Type***
agent	Object
genericconnections	Object

Methods

The GenericAddress class inherits the following methods from the _UQAddress class:

- acceptTask
- dequeueTask

See _UQAddress.

Method	***Returns***
getGenericconnectionByConnectionId (*connectionId*)	Object
getGenericconnectionindexByConnectionId (*connectionId*)	Index
getFreeGenericconnection	Object

Callback Event Methods

The GenericAddress class inherits the onError callback event method from the _Address class.

See _Address.

The GenericAddress class inherits the following callback event methods from the _UQAddress class:

- onAccepted

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

- onAcceptingTask
- onDequeueingTask
- onNotify
- onTaskAdded
- onTaskRemoved
- onUnassigned

See _UQAddress.

GenericConnection

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
GenericConnection(*event*)	GenericConnection object

Fields

Field	***Type***
caps	Object
connectionId	String
generic	String
id	String
state	String

Methods

Method	***Returns***
abandon (*reason*)	None
answer (*reason*)	None
complete (*reason*)	None

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Method	***Returns***
forward (*fromagentid*, *toagentid*, *qid*, *userdata*, *Genericdata*, *reason*)	None
reject(*agentid*, *reason*)	None
withdraw_RES (*reason*)	None

Callback Event Methods

Callback Event Method	***Returns***
onCapabilitiesChanged (*event*)	None
onCompleted (*event*)	None
onDropped (*event*)	None
onError (*event*)	None
onForwarded (*event*)	None
onForwardError (*event*)	None
onForwarding (*event*)	None
onGenericdataChanged (*event*)	None
onIncoming (*event*)	None
onProcessing (*event*)	None
onRejected (*event*)	None
onRevoked (*event*)	None
onUserdataChanged (*event*)	None
onWithdraw_REQ (*event*)	

GenericConnectionCaps

This section lists the constructor, fields, methods, and returns (if applicable) associated with this class.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	*Returns*
GenericConnectionCaps(*event*)	GenericConnectionCaps object

Fields

Field	*Type*
canAnswer	Boolean
canComplete	Boolean
canForward	Boolean
canReject	Boolean

GenericData

This section lists the constructor, fields, methods, and returns (if applicable) associated with this class.

Constructor

Constructor	*Returns*
GenericData(*event*)	GenericData object

Fields

Field	*Type*
data	Object

Methods

Method	*Returns*
addKeyValue(*key*, *value*)	None

GenericTask

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

The GenericTask class extends the Task class.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

See Task.

Constructor

Constructor	***Returns***
GenericTask(*event*)	GenericTask object

Fields

The GenericTask class inherits the following fields from the Task class:

- caseid
- cost
- customerid
- group
- id
- onStat
- priority
- type
- urlAbs
- urlRel

See Task.

Field	***Type***
address	String
agentId	String
appData	Object
customerName	String
genericconnection	Object
genericId	String
groupId	String
question	String
subject	String

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Field	Type
userData	Object

Methods

Method	Returns
gettpUrl(*defaultUrl*)	Object
getUrl(*defaultUrl*)	String

GLOBALS

This section lists the fields and methods associated with this class.

Fields

Field	Type
A2AChat.PS_JR	Constant
A2AChat.TYPE_ANSWER	Constant
A2AChat.TYPE_CONSULT	Constant
Server.TYPE.CTI	Constant
Server.TYPE_UQ	Constant
Task.TYPE_A2ACHAT	Constant
Task.TYPE_CHAT	Constant
Task.TYPE_CTI	Constant
Task.TYPE_EMAIL	Constant
Task.TYPE_GENERIC	Constant

Methods

Method	Returns
initJSMCAPI()	None
isValid(*obj*)	Boolean

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Method	***Returns***
MCFBroadcast(*cluster*, *cluster*, *cluster*, *cluster*, *cluster*, *cluster*, *cluster*, *cluster*, *cluster*, *cluster*)	None

Group

This section lists the constructor, fields, callback event methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
Group()	Group object

Fields

Field	***Type***
id	String
name	String
registered	Boolean
statistics	Object
statistics1	Object
statistics2	Object

Callback Event Methods

Callback Event Methods	***Returns***
onStat(*event*)	None
onStat1(*event*)	None
onStat2(*event*)	None
onTaskAdded(*event*)	None
onTaskRemoved(*event*)	None

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

GroupStatistics

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
GroupStatistics()	GroupStatistics object

Fields

Field	***Type***
data	Associative array
listOfTasksInTheQueueByTaskType	Number
maxTaskCompletionTime	Number
newestTask	Number
newestTaskCompletionTime	Number
numberOfAbandoned	String
numberOfLoggedIn	String
numberOfQueued	String
numUnassignedTasks	Number
queuedWaitTime	String
queueUpTime	Number
relativeQueueLoad	String
timeElapsedOldestTask	Number

GroupStatistics1

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
GroupStatistics1(*data*)	GroupStatistics1 object

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Fields

Field	*Type*
mostRecentTaskDone	String
mostRecentTaskDoneData	String
mostRecentTaskEnqueued	String
mostRecentTaskEnqueuedData	String
numAgentsAvailable	String
numAgentsLoggedIn	String
numEscalation	String
numOverflow	String
numTaskAccepted	String
numTaskDone	String
numTaskQueued	String
reasonFlag	String
taskTotalTimeInSystem	String
timeSinceStart	String

GroupStatistics2

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Constructor

Constructor	*Returns*
GroupStatistics2(*data*)	GroupStatistics2 object

Fields

Field	*Type*
averageTaskDuration	String
averageWaitTime	String

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Field	*Type*
oldestTask	String
recentTask	String
timeElapsedOldestTask	String
timeElapsedRecentTask	String

Line

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

Constructor

Constructor	*Returns*
Line()	Line object

Fields

Field	*Type*
call	Object
caps	Object
connectionId	String
id	String
isMuted	Boolean
state	String of the following constants: • ST_DIALING • ST_DROPPED • ST_HELD • ST_IDLE • ST_OFFHOOK • ST_RINGING • ST_TALKING

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Methods

Method	***Returns***
alternate(*reason*)	None
answer(*reason*)	None
attachUserData(*userdata*, *reason*)	None
clear(*reason*)	None
complete(*reason*)	None
conference(*destination*, *reason*, *userdata*, *calldata*)	None
conferenceSingle(*destination*, *reason*, *userdata*, *calldata*)	None
dial(*number*, *reason*, *userdata*)	None
dropParty(*destination*, *reason*)	None
getAni()	String
getDescr()	String
getDnis()	String
getReferenceId()	String
getUrl(*defaultUrl*)	String
grabCall(*destination*, *reason*)	String
hold(*reason*)	String
join (*reason*, *conferenceId*)	None
mute (*reason*)	None
park(*destination*, *reason*)	None
reconnect(*reason*)	None
reject(*reason*)	None
release(*reason*)	None
retrieve(*reason*)	None
sendDTMF(*reason*, stringDTMF)	None

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Method	***Returns***
setcallresult(*result*)	None
setcallresultDNC(*reason*, *number*)	None
setcallresultReschedule(*hours*, *minutes*,*day*, *month*,*year*)	None
transfer(*destination*, *reason*, *userdata*, *calldata*)	None
transferMute(*destination*, *reason*, *userdata*, *calldata*)	None
unmute(*reason*)	None
updateCallData(*calldata*, *reason*)	None

Callback Event Methods

Callback Event Method	***Returns***
onAlternating(*event*)	None
onAnswering(*event*)	None
onAttachingUD(*event*)	None
onCallDataChanged(*event*)	None
onCapabilitiesChanged(*event*)	None
onClearing(*event*)	None
onCompleting(*event*)	None
onConferencing(*event*)	None
onDialing(*event*)	None
onDropped(*event*)	None
onError(*event*)	None
onGrabbing(*event*)	None
onHeld(*event*)	None
onHolding(*event*)	None
onJoining(*event*)	None

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Callback Event Method	***Returns***
onMuted(*event*)	None
onOffHook(*event*)	None
onOnHook(*event*)	None
onParking(*event*)	None
onPartyAdded(*event*)	None
onPartyChanged(*event*)	None
onPartyRemoved(*event*)	None
onReconnecting(*event*)	None
onRejected(*event*)	None
onRejecting(*event*)	None
onReleasing(*event*)	None
onRetrieving(*event*)	None
onRinging(*event*)	None
onSetcallresult(*event*)	None
onSetcallresultDNC(*event*)	None
onSetcallresultReschedule(*event*)	None
onTalking(*event*)	None
onTransfering(*event*)	None
onUnmuted(*event*)	None
onUpdatingCD(*event*)	None
onUserDataChanged(*event*)	None

LineCaps

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	***Returns***
LineCaps(*strCaps*)	LineCaps object

Fields

Field	***Type***
canAlternate	Boolean
canAnswer	Boolean
canAttachUserData	Boolean
canClear	Boolean
canComplete	Boolean
canConference	Boolean
canConferenceSingle	Boolean
canDropParty	Boolean
canHold	Boolean
canMute	Boolean
canPark	Boolean
canReconnect	Boolean
canReject	Boolean
canRelease	Boolean
canRetrieve	Boolean
canSendDTMF	Boolean
canSetcallresult	Boolean
canSetcallresultDNC	Boolean
canSetcallresultReschedule	Boolean
canTransfer	Boolean

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Field	*Type*
canTransferMute	Boolean
canUnmute	Boolean
canUpdateCallData	Boolean

MCEvent

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Constructor

Constructor	*Returns*
MCEvent()	MCEvent object

Fields

Field	*Type*
extension	Object
group	Object
reason	Object
user	Object

MediaType

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

Constructor

Constructor	*Returns*
MediaType()	MediaType object

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Fields

Field	*Type*
type	String of the following constants: • MT_EMAIL • MT_VOICE

PSMC

This section lists the constructor, fields, methods, and returns (if applicable) associated with this class.

Constructor

Constructor	*Returns*
PSMC()	PSMC object

Fields

Field	*Type*
renServer	Object
servers	Associative array
sessions	Associative array

Methods

Method	*Returns*
closeSession(*serverId*)	None
getCallById(*id*)	Object
getChatById(*id*)	Object
getEmailById(*id*)	Object
getGenericTaskById(*id*)	Object
getLineById(*id*)	Object
openSession(*serverId*)	None

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Method	***Returns***
start()	None
stop()	None

Reason

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
Reason(*code*, *desc*)	Reason object

Fields

Field	***Type***
code	String
description	String
reasonData1	String
reasonData2	String
reasonData3	String

RenServer

This section lists the constructor, fields, callback event methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
RenServer()	RenServer object

Fields

Field	***Type***
isRunning	Boolean

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Field	*Type*
url	String

Callback Event Methods

Callback Event Method	*Returns*
onDown(*event*)	None
onUp(*event*)	None

Server

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

Constructor

Constructor	*Returns*
Server(*serverID*)	Server object

Fields

Field	*Type*
id	String
info	String
state	String of the following constants: • ST_INSERVICE • ST_OUTOFSERVICE
type	String of the following constants: • TYPE_CTI • TYPE_UQ

Callback Event Methods

Callback Event Method	*Returns*
onBroadcast(*event*)	None

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Callback Event Method	***Returns***
onHbLost(*event*)	None
onHbRecovered(*event*)	None
onInService(*event*)	None
onOutOfService(*event*)	None
onRestart(*event*)	None

Session

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
Session(*serverID*)	Session object

Fields

Field	***Type***
addresses	Associative array
buddies	Associative array
groups	Associative array
id	String
intervalBetweenReqs	Number
numberRegsPerBulkReq	Number
serverID	String

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Field	***Type***
state	String of the following constants: • ST_ACTIVE • ST_CLOSED • ST_CLOSING • ST_IDLE
user	User object

Methods

Method	***Returns***
broadcastSubscribe(*cluster*, *queue*, *task*, *state*, *presence*, *method*)	None
broadcastUnsubscribe(*type*)	None
close()	None
open()	None
registerAddress(*address*)	None
registerBuddy(*buddy*)	None
registerBuddiesBulk(*buddies*, *numRegsPerBulkReq*, *intervalBetweenReqs*)	None
registerGroup(*group*)	None
registerGroupsBulk(*groups*, *numRegsPerBulkReq*, *intervalBetweenReqs*)	None
registerUser(*user*)	None
setAutoRecovery(*autoRecover*)	None
statPublish(*userStat*, *userStat*)	None
unregisterAddress(*address*)	None
unregisterBuddy(*buddy*)	None
unregisterGroup(*group*)	None

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Method	***Returns***
unregisterUser(*user*)	None

Callback Event Methods

Callback Event Method	***Returns***
onAddressRegistered(*event*)	None
onAddressUnregistered(*event*)	None
onBuddyRegistered(*event*)	None
onBuddyUnregistered(*event*)	None
onClosed(*event*)	None
onError(*event*)	None
onGroupRegistered(*event*)	None
onGroupUnregistered(*event*)	None
onInfo(*event*)	None
onOpened(*event*)	None
onUserRegistered(*event*)	None
onUserUnregistered(*event*)	None

Task

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
Task()	Task object

Fields

Field	***Type***
caseid	String

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Field	***Type***
cost	String
customerid	String
group	String
id	String
onStat	String
priority	String
type	String; one of the following: • TYPE_A2ACHAT • TYPE_CHAT • TYPE_CTI • TYPE_EMAIL • TYPE_GENERIC
urlAbs	Object
urlRel	Object

Methods

Method	***Returns***
setUserData(*UserData data*)	None
getUserData( )	UserData

TaskStatistics

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
TaskStatistics()	TaskStatistics object

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Fields

Field	*Type*
data	Associative array
holdTime	String
queueTime	String
talkTime	String

User

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

The User class extends the _User class.

See _User.

Constructor

Constructor	*Returns*
User(*id*, *name*, *agentId*, *agentPassword*)	User object

Fields

The User class inherits the following fields from the _User class:

- agentId
- caps
- id
- name
- presences
- states
- statistics
- statistics1
- statistics2

See _User.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Field	***Type***
addresses	Associative array
agentPassword	String
language	String
registeredAddresses	Associative array
statesUq	Object

Methods

Method	***Returns***
disableMedia(*group*, *mediaType*, *extension*, *reason*)	None
enableMedia(*group*, *mediaType*, *extension*, *reason*)	None
isMediaEnabled(*mediaType group*)	None
login(*group*, *extension*, *reason*)	None
loginUq(*group*)	None
logout(*group*, *extension*, *reason*)	None
logoutUq(*group*)	None
register(*extension*, *reason*)	None
setNotReady(*group*, *presence*, *extension*, *reason*)	None
setPresence(*group*, *presence*, *extension*, *reason*)	None
setPresenceUq(*group*, *presenceText*, *reason*)	None
setReady(*group*, *presence*, *extension*, *reason*)	None
setWorkNotReady(*group*, *presence*, *extension*, *reason*)	None
setWorkReady(*group*, *presence*, *extension*, *reason*)	None
unRegister(*extension*, *reason*)	None

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Callback Event Methods

Callback Event Method	***Returns***
onCapabilitiesChanged(*event*)	None
onCtiBusy(*event*)	None
onCtiClear(*event*)	None
onDropped(*event*)	None
onError(*event*)	None
onInfo(*event*)	None
onLoggedIn(*event*)	None
onLoggedOut(*event*)	None
onLoggingIn(*event*)	None
onLoggingOut(*event*)	None
onMediaDisabled(*event*)	None
onMediaEnabled(*event*)	None
onNotReady(*event*)	None
onPresenceChanged(*event*)	None
onReady(*event*)	None
onRegistered(*event*)	None
onRegistering(*event*)	None
onSettingNotReady(*event*)	None
onSettingPresence(*event*)	None
onSettingReady(*event*)	None
onSettingWorkNotReady(*event*)	None
onSettingWorkReady(*event*)	None
onStat(*event*)	None
onStat1(*event*)	None

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Callback Event Method	***Returns***
onStat2(*event*)	None
onUnkown(*event*)	None
onUnregistered(*event*)	None
onUnregistering(*event*)	None
onWorkNotReady(*event*)	None
onWorkReady(*event*)	None

UserCaps

This section lists the constructor, fields, method, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
UserCaps()	UserCaps object

Fields

Field	***Type***
canHandleChat	Boolean
canHandleEmail	Boolean
canHandleGeneric	Boolean
canHandleVoice	Boolean
canLogin	Boolean
canLogout	Boolean
canRefreshState	Boolean
canSetNotReady	Boolean
canSetPresence	Boolean
canSetReady	Boolean

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Field	***Type***
canSetWorkNotReady	Boolean
canSetWorkReady	Boolean

UserData

This section lists the constructor, constants, methods, and returns (if applicable) associated with this class.

Constructor

Constructor	***Returns***
UserData()	UserData object

Constants

The UserData class uses the following constants:

- COMPONENT
- DESCR
- ISCRIPTPROGRAMNAME
- ICTYPE
- MARKET
- MENU
- REFERENCEID
- TARGET
- URLABS
- URLREL

Methods

Method	***Returns***
add(*key*, *value*)	None
addKeyValue(*key*, *value*)	None
getValue(String key)	String
remove(*key*)	None

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Method	*Returns*
getKeys( )	String array

UserStatistics1

This section lists the constructor, fields, and returns (if applicable) associated with this class.

Constructor

Constructor	*Returns*
UserStatistics1()	UserStatistics1 object

Fields

Field	*Type*
availableCost	String
ctiBusy	String
currentQueue	String
mostRecentTaskData	String
mostRecentTaskId	String
numTaskAccepted	String
numTasksDone	String
numTasksUnassigned	String
presenceText	String
reasonFlag	String
state	String
timeInCurrentState	String
timeSinceLoggedIn	String

UserStatistics2

This section lists the constructor, fields, methods, callback event methods, and returns (if applicable) associated with this class.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Constructor

Constructor	*Returns*
UserStatistics2()	UserStatistics2 object

Fields

Field	*Type*
currentQueue	String
timeIdle	String
timeInCurrentState	String
timeNotReady	String
timeSinceLogin	String
totalTimeAvailable	String
totalTimeUnavailable	String

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Appendix B

Installing Digital Certificates for REN SSL

Installing Digital Certificates

Digital certificates are required to provide client and server authentication for REN SSL. A digital certificate is an electronic means of establishing your credentials for web or business transactions that are issued by a certification authority (CA). The CA is a trusted third party who signs and issues the certificates for users after verifying their authentication using secure means.

The Installing Digitial Certificates for REN SSL topic presents a sample way of installing digital certificates and configure REN SSL. PeopleSoft customers may have their own means of obtaining and installing digital certificates for REN SSL.

This section outlines the basic steps to install digital certificates. Before installing digital certificates, you must create the application server domain.

Note: The application server domain must have write permissions. All certificates are stored under <PS_HOME>/appserv/<domain name>. The cacerts file has write permissions under <PS_HOME>/JRE/lib/security.

The following overview lists the steps that are required to install digital certificates. The subsequent sections describe each step in detail.

To install digital certificates and configure REN SSL:

1. Install the CA server certificate.

   See Installing the CA Server Certificate.

2. Install the REN server certificate.

   See Installing the REN Server Certificate.

3. Configure digital certificates.

   See Configuring Digital Certificates.

4. Import certificates into Java keystore for any REN Java Clients.

   See Importing Certificates in Java Keystore for any REN Java Clients.

5. Configure the REN server.

   See Configuring the REN Server.

6. Configure REN clusters.

   See Configuring REN Clusters.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

7. Install certificates for local node.

   See Installing Certificates for Local Node.

8. Generate the client certificate.

   See Generating the Client Certificate.

9. Install PSMCAPI certificates.

   See Installing PSMCAPI Certificates.

10. Configure an external keystore.

    See Configuring External Keystore in REN Server

Installing the CA Server Certificate

To install the CA server certificate:

1. Generate the RSA private key for certificate authority.
2. Generate the Certificate Signing Request (CSR) for certificate authority.
3. Generate the PEM file.

   Note: If a CA certificate already exists in PEM format, the preceding three steps can be omitted.

4. Import the CA Certificate in PEM format using PeopleTools, Security, Security Objects, Digital Certificates.

The preceding steps are explained in detail in the section Configuring Digital Certificates.

See Configuring Digital Certificates, "Implementing Client Authentication" (PeopleTools 8.54: Integration Broker Administration).

Installing the REN Server Certificate

To install the REN server certificate:

1. Generate REN server CSR using PeopleTools, Security, Security Objects, Digital Certificates.
2. Get the CSR signed by a CA.

   Note: The certificate must be in PEM format.

3. Import the certificate into PeopleTools, Security, Security Objects, Digital Certificates.

The preceding steps are explained in detail in the section Configuring Digital Certificates.

See Configuring Digital Certificates.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Configuring Digital Certificates

Before configuring digital certificates, you must generate the private keys, CSR, and PEM file.

To configure digital certificates:

1. Select PeopleTools, Security, Security Objects, Digital Certificates.
2. Click +.
3. Select *ROOTCA* from the Type drop-down list box.
4. Enter an alias name for the CA in Alias, and click Add Root.

   The Add Root Certificate dialog box appears.
5. Open the ca.pem file.

   The root CA certificate is generated.
6. Copy the contents of the ca.pem file, paste them into the Add Root Certificate dialog box, and click OK.
7. Click +.
8. Select *Cert* from the Type drop-down list box.
9. Enter an alias name in Alias, such as PSFTCA.
10. Click Add Root.
11. Select the CA certificate alias of step 4 from the Issuer Alias lookup button.
12. Click Request.

    The Request New Certificate dialog box appears.
13. Complete the Common Name, Org Unit (organization unit), Organization, Locality, State/Province, Country, Algorithm, Key Size, Email Address, and Challenge Pswdfields.

    Note: The common name must be the machine name of the REN server machine, for example, PTA112.peoplesoft.com, where PTA112 is the machine name and .peoplesoft.com is the domain name.

    ---
14. Click OK.

    The Certificates Signing Request dialog box appears.
15. Copy and paste the text from the Certificates Signing Request dialog box, and save the text in a file named ren.csr in <PS_HOME>\appserv\<domain name>\.
16. Click OK.

    The Import link appears.
17. Submit ren.csr to the CA that issued the selected root certificate.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

The CA may send you the signed public key certificate by email or require you to download it from a specified web page.

18. Open the saved certificate file in a text editor, and then highlight and copy its entire contents.
19. Select PeopleTools, Security, Security Objects, Digital Certificates.
20. Click Import.

    The Import Certificate page appears.

21. Paste the copied certificate content into the long edit box, and click OK.

See "Implementing Node Authentication" (PeopleTools 8.54: Integration Broker Administration), "Implementing Nonrepudiation" (PeopleTools 8.54: Integration Broker Administration).

Importing Certificates in Java Keystore for any REN Java Clients

In order for REN Java clients, such as a queue server, MCF Log server, or CTI adapter, to communicate with an SSL-enabled REN server, these clients need to have SSL certificates available in a Java keystore. You must import the REN server certificates into the keystore used by each REN Java client. For the Queue servers and MCF Log servers, this will be the keystore used by the JRE in the PeopleSoft domain where the queue server and MCF Log server are running. For CTI adapters, this will be the JRE used by the adapter.

The following example shows how to install the REN server certificates into the JRE trust store for a queue server and MCF log server. You will need to repeat this procedure for every queue server domain being used. You will also need to perform a similar procedure for CTI adapters or any other REN Java client, if you are using them.

Note: If you have configured client authentication, you will need to import client certificates into the JRE keystore.

To import certificates in Java keystore:

1. Open a command prompt.
2. Enter the following command:

```
<PS_HOME>\jre\bin\keytool -import -trustcacerts -alias <alias-name>
-file <CA Certificate Pem file> -keystore <full path to keystore used by the R⇒
EN Java client>
-storepass <password>
```

   Example:

```
<PS_HOME>\jre\bin\keytool -import -trustcacerts -alias PSFTCA -file ca.pem
-keystore <PS_HOME>\jre\lib\security\cacerts -storepass changeit
```

Note: You will get an error message, sslv3 alert certificate unknown, if the certificate is not imported correctly.

See "Implementing Client Authentication" (PeopleTools 8.54: Integration Broker Administration).

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Configuring the REN Server

To configure the REN server for SSL:

1. Select PeopleTools, REN Server Configuration, REN Server Definition.

   The REN Server Definition page appears.

2. Select the SSL Only check box.
3. Select your REN Server certificate from the Certificate Alias drop-down list box.
4. Click *Save.*

Related Links

Defining REN Servers

Configuring REN Clusters

To configure REN clusters:

1. Select PeopleTools, REN Server Configuration, REN Server Cluster.

   The REN Server Cluster page appears.

2. Update the REN server cluster URL using https and the SSL port.
3. Update the REN server browser URL using https and the SSL port.
4. Click Save.

Related Links

Defining a REN Server Cluster

Installing Certificates for Local Node

Apart from the CA and REN server certificates, client authentication requires local node certificates, a client certificate for the browser, and a PSMCAPI certificate.

To install certificates for the local node:

1. Select PeopleTools, Security, Security Objects, Digital Certificates.
2. Click +.
3. Select *Local Node* from the Type drop-down list box.
4. Enter the local node name in Alias, and click Add Root.
5. Select the alias of the CA certificate from the Issuer Alias lookup button.
6. Click Request.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

7. Complete the Common Name, Org Unit, Organization, Locality, State/Province, Country, Algorithm, Key Size, Email Address, and Challenge Pswd fields.

   Note: The common name must be the machine name of the REN server machine, for example, PTA112.peoplesoft.com, where PTA112 is the machine name and .peoplesoft.com is the domain name.

8. Click OK.

   The Certificates Signing Request dialog box appears.

9. Copy and paste the text from the Certificates Signing Request dialog box, and save the text in a file.

10. Click OK.

    The Import link appears.

11. To obtain your local node certificate, submit the certificate request text to the CA that issued the selected root certificate.

    The process of obtaining digital certificates varies, depending on the CA. Typically, a CA requires you to paste the content of the PEM-formatted CSR into a form that you submit online. The CA may send you the signed public key certificate by email or require you to download it from a specified web page.

12. Open the saved certificate file in a text editor, and then highlight and copy its entire contents.

13. Select PeopleTools, Security, Security Objects, Digital Certificates.

14. Click the Import link.

    The Import Certificate page appears.

15. Paste the copied certificate content into the long edit box, and click OK.

See "Implementing Node Authentication" (PeopleTools 8.54: Integration Broker Administration), "Implementing Nonrepudiation" (PeopleTools 8.54: Integration Broker Administration).

Generating the Client Certificate

You can generate the client certificate by openssl or keytool in P12 format and import it in the browser. Importing the certificates depends on the browser.

The following steps are an example of generating a client certificate using openssl. Clients can use keytool or Microsoft CA to generate the client certificate.

To generate the client certificate using openssl:

1. Generate the RSA private key.

   ```
   openssl genrsa -des3 -out <Private key file>
   ```

   Example:

   ```
   openssl genrsa des3 out renclient.key passout pass:pass 1024
   ```

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

2. Generate a CSR file.

```
openssl req -config <filename> -new -M;key <Private key file> -out <CSR file>
```

Example:

```
openssl req -config ..\apps\openssl.cnf -new -key renclient.key -x509 -days 36⇒
5 -out renclient.csr
```

3. Generate a PEM format file.

```
openssl x509 -req -days 365 in <CSR file> -CA <CA PEM File> -CAkey<CA
Key File> CAcreateserial -out <RenServer PEM file> -outform PEM
```

Note: <CA key file> and <CA PEM file> are Certificate Authority Key file and Certificate Authority in PEM format respectively.

Example:

```
openssl x509 req days 365 in RENCLIENT.csr -CA ca.pem -CAkey ca.key -CAcreates⇒
erial -out renclient1.pem -outform PEM -passout pass:pass
```

4. Generate a .p12 certificate for the browser.

```
openssl pkcs12 -export -in <RenServer PEM file>  -out <.p12 file> -inkey
<Private key file> -name <alias name>
```

Example:

```
openssl pkcs12 -export -in renclient.pem -out renclient.p12 -inkey
renclient.key -M;name renclient
```

Installing PSMCAPI Certificates

To install PSMCAPI certificates:

1. Generate a private key in the keystore using the following command:

```
<PS_HOME>\jre\bin\keytool genkey dname CN=Company Name,
OU=Organization Unit, O=M;Organization, L=Locality, S=State/Provenance,
C=Country alias <alias Name>  -M;keyalg RSA validity 365 keystore
<PS_HOME>\jre\lib\cacerts storepass changeit keypass password
```

2. Generate the CSR using the following command:

```
PS_HOME>\jre\bin\keytool certreq alias <alias name> -file <certificate
file name> -keystore <full path to the Java keystore used by the CTI adapter> ⇒
storepass
<password> keypass <password>
```

3. To obtain your certificate, submit the CSR to the CA that issued the selected root certificate.

4. Import the signed certificates into Java keystore using the following command:

```
<PS_HOME>\jre\bin\keytool import alias <alias name> -file <certificate
file .pem> -keystore <full path to the Java keystore used by the CTI adapter> ⇒
storepass <password>
keypass <password>
```

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Note: The clients must import the CA certificate in Java keystore of JRE of PSMCAPI for SSL communication with the REN server using the keytool command.

See "Implementing Client Authentication" (PeopleTools 8.54: Integration Broker Administration).

Configuring External Keystore in REN Server

Certificates that are used to secure a Web server can be reused to secure a REN server. The Java keystore that stores these certificates can be converted to a PKCS #12 keystore. Subsequently, you can configure the REN server to use the newly created external PKCS #12 keystore when you define a REN server the SSL and the UseExternalKeystore options are selected.

To convert a Java keystore to a PKCS #12 keystore, you can use the PSExportToPKCS12.bat file that is delivered by Oracle.

1. Access PS_HOME/peoplesoft/Piabin and locate the PSExportToPKCS12.bat file.

   Note: Ensure that the JAVA_HOME environment variable is set correctly.

2. Run PSExportToPKCS12.bat in Windows or PSExportToPKCS12.sh in UNIX.

   Note: The PSExportToPKCS12.bat and PSExportToPKCS12.sh scripts must be run using a version of the Sun Java Runtime environment compatible with your PeopleTools installation.

   Image: Convert Java keystore to PKCS #12 keystore

   This example illustrates the conversion of Java keystore to PKCS #12 keystore using the PSExportToPKCS12.bat file.

```
C:\PT_INSTALLS\903-I1_Copysys\webserv\peoplesoft\piabin>PSExportToPKCS12
Enter Source Java Keystore file
[Full path of Keystore file you want to export as PKCS12 file]:
c:\keystore.jks
Enter Source Keystore Password
[This is password of Keystore file you want to export]:
password
Enter Source Keystore Alias
[This is Alias of the KeyPair in the Keystore file you want to export]:
TESTREN
Enter Source Key Password
[Key Password. Press enter if same as Keystore Password ]:

Enter Destination Keystore file
[Full path of exported PKCS12 file]:
C:\Users\kkameshwar\psft\pt\8.52\appserv\Q8529031\psrenserver\modules\nsopenssl\TESTREN1.p12
Enter Destination Keystore Password
[This is password of Exported Keystore file ]:
password
Enter Destinition Alias
[This is Alias of the KeyPair in exported file]:
TESTREN
Enter Destinition Trusted Store
[Full path of exported TrustStore file]:
C:\Users\kkameshwar\psft\pt\8.52\appserv\Q8529031\psrenserver\modules\nsopenssl\TRUSTCA.pem
Exported PKCS12 file:C:\Users\kkameshwar\psft\pt\8.52\appserv\Q8529031\psrenserver\modules\nsopenssl\TESTREN1.p12
Exported TrustStore file:C:\Users\kkameshwar\psft\pt\8.52\appserv\Q8529031\psrenserver\modules\nsopenssl\TRUSTCA.pem
C:\PT_INSTALLS\903-I1_Copysys\webserv\peoplesoft\piabin>_
```

Note: The PSExportToPKCS12.bat file must be run prior to defining a REN server in the REN Server Configuration page. The file must not be run during the boot sequence of the application server.

 Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

3. Enter values for the following:
   - Source Java Keystore — the full path of Java Keystore.
   - Source Keystore Password — the password of the Java Keystore
   - Source Keystore Alias — alias of the KeyPair in the Java Keystore.
   - Source Key Password — if the password is the same as Source Keystore Password, you can leave it blank.
   - Destination Keystore — the full path of exported PKCS #12 keystore including the file extension (*.p12* is preferred).
   - Destination Keystore Password — the password of exported keystore.
   - Destination Alias — alias of the KeyPair in exported keystore.
   - Destination Trusted Store — the full path of the exported TrustStore file including the file extension (*.pem* is preferred).

   The Java Keystore is converted to PKCS #12 keystore.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Appendix C

PSRENCONFIG Quick Reference

PSRENCONFIG Parameters

This section lists the REN server configuration parameters. Each parameter listed below is grouped by its section within the psrenconfig.txt file. Identically named parameters in different sections will affect different functions of the REN server, and are independent of each other.

The list of parameters described here is not complete. For a complete list of parameters, see the psrenconfig.txt file created with your appserver domain.

Note: Default values may differ between releases including patch releases. For the most up to date default value information, see the psrenconfig.txt file in your installation. These default values may not be the recommended settings for your specific type of installation.

Global Parameters

Parameter	*Default Value*	*Description*
listenbacklog	32	The maximum length of the pending connection queue on the listening sockets.

Server Parameters

Parameter	*Default Value*	*Description*
maxthreads	40	The maximum number of connection threads that will be created.
minthreads	5	The starting number of connection threads.

Event Routing

Parameter	*Default Value*	*Description*
reaper_interval	3600	How frequently, in seconds, the REN server will do internal cache cleanup.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.

Parameter	***Default Value***	***Description***
default_kn_expires	“+3600”	How quickly, in seconds, cached items will, by default, become ready for cleanup.

Copyright © 1988, 2016, Oracle and/or its affiliates. All rights reserved.