Oracle® Real-Time Collaboration

Application Developer's Guide

10*g* Release 1 (10.1.2)

B25461-01

November 2005

Oracle Real-Time Collaboration Application Developer’s Guide, 10*g* Release 1 (10.1.2)

B25461-01

Primary Author: Raymond Gallardo

Contributor: Vimal Chopra, Ramesh Dommeti, Anindo Roy, Paddu Vedam, Gerardo Viedma

Contents

Preface

This Preface contains these topics:

Audience

Oracle Real-Time Collaboration Application Developer's Guide is intended for any programmers and developers who want to integrate the functionality of Oracle Real-Time Collaboration with their applications.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documents

For more information, see the following manuals in the Oracle Collaboration Suite documentation set:

- *Oracle Real-Time Collaboration Administrator's Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction to Oracle Real-Time Collaboration Integration Services

Oracle Real-Time Collaboration Integration Services bring real-time collaboration within the context of the business applications in an enterprise. The Oracle Real-Time Collaboration system can be used to cobrowse Web documents, share desktops, conduct polls, chat, stream voice, and much more. Oracle Real-Time Collaboration facilitates real-time collaboration using a company's corporate intranet and the public Internet.

Oracle Real-Time Collaboration Integration Services can be used by the following groups:

- Lines of business that want to easily use real-time collaboration functionality from within the context of their applications
- Online e-businesses that want to quickly implement very valuable features like **live help** or "shop-together" from all their sites' Web pages
- Help desks that want to dramatically improve customer satisfaction by providing desktop sharing with the click of a button
- Sales organizations that want to quickly implement a system where its sales agents can use an existing application to provide live demonstrations of their products to their customers
- Online learning organizations that want to enhance existing applications by providing recorded presentations of classroom material or interactive classrooms

Oracle Real-Time Collaboration Integration Services provides easy, industry-standard methods for any application to tightly integrate with the Oracle Real-Time Collaboration deployment within an enterprise. These services allow you to extend existing interfaces with real-time collaboration features. Examples of such scenarios include the following:

- **Extending web portals with SOAP services, servlets, and JavaScript modules**: Support divisions can enable Oracle Real-Time Collaboration functionality in their end user interfaces. Support pages can be enhanced to provide indicators about the availability of support personnel, or groups or support personnel, and can provide hooks to initiate real-time chat conversations between the end users and the support personnel. These chat conversations can be upgraded to a Web Conference seamlessly.

  Sales divisions can use Oracle Real-Time Collaboration functionality when demonstrating product features. A telephone call can be upgraded to a live demonstration using Web conferencing. Online sales collateral can be enhanced with presence indicators that users can use to initiate conversations with sales agents regarding product features.

- **Extending enterprise applications with SOAP services**: Enterprise applications can be enhanced to provide real-time collaboration features in their workflow. For example, Human Resource applications can be enhanced to provide real-time access to HR personnel who are signed using their instant messenger client. Queries can be answered in real-time and employee requests can be processed faster, having a positive impact on employee productivity in the organization.
- **Extending desktop software with plug-ins, COM objects, Java SDKs and Messenger add-ins**: Desktop software, such as Microsoft Outlook and Word, can be enhanced to provide real-time collaboration functionality. For example, a user preparing a document can seamlessly share the document with his supervisor for real-time feedback. Users can schedule appointments as Web conferences directly from common desktop applications, such as Outlook.

The Oracle Real-Time Collaboration system has been designed to let you have a single, centrally managed Oracle Real-Time Collaboration system, deployed in an enterprise, that provides real-time collaboration functionality to multiple lines of business at the same time. Each line of business can determine which Oracle Real-Time Collaboration features to expose and customize how they appear to their users.

Technology Requirements

Oracle Real-Time Collaboration Integration Services have been designed to use industry-standard SOAP XML messages and HTTP protocol for integration with Oracle Real-Time Collaboration system. The architecture facilitates a single deployment of Oracle Real-Time Collaboration in an enterprise to meet the requirements of all types of integrating applications. The integrating applications might not be co-located and managed by the same administrator as the Oracle Real-Time Collaboration deployment.

This model of integration lends itself very well in situations where Web applications are located in a distributed environment and need to expose real-time capabilities by integrating with a central deployment of Oracle Real-Time Collaboration.

Figure 1–1 Centrally-Managed Oracle Real-Time Collaboration System Supporting Multiple Lines of Business

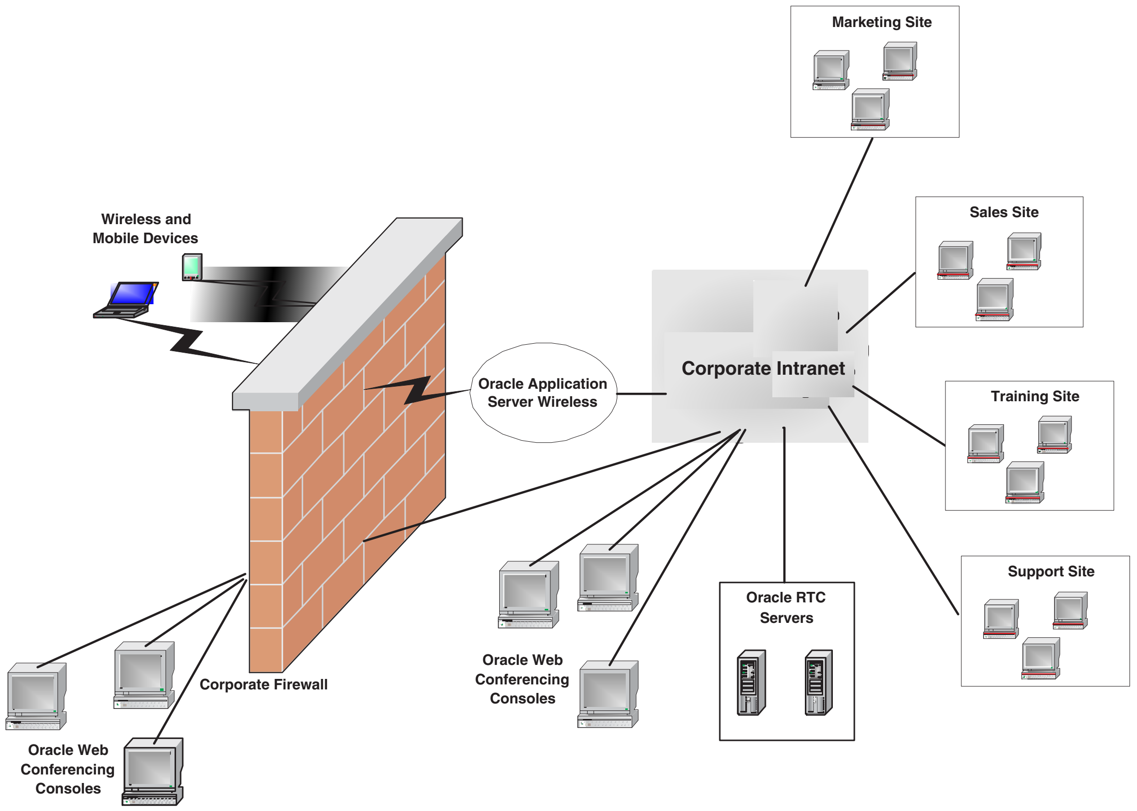

Marketing Site
Sales Site
Wireless and Mobile Devices
Corporate Intranet
Oracle Application Server Wireless
Training Site
Support Site
Oracle RTC Servers
Oracle Web Conferencing Consoles
Corporate Firewall
Oracle Web Conferencing Consoles

Figure 1–1 illustrates how Oracle Real-Time Collaboration integrates with other applications. The Oracle Real-Time Collaboration servers are behind the corporate firewall. The following sites are located on the corporate intranet: Marketing, Sales, Training, Support. Through Oracle Real-Time Collaboration Integration Services, these sites expose Oracle Real-Time Collaboration functionality, which is provided by Oracle Real-Time Collaboration servers. Wireless and mobile devices can access Oracle Real-Time Collaboration functionality through Oracle Application Server Wireless. The Oracle Real-Time Collaboration servers also provide Oracle Web Conferencing functionality, which users can access through the Oracle Web Conferencing Console.

All interfaces of Oracle Real-Time Collaboration Integration Services are defined as Web-based services. These interfaces are implemented as a set of servlets that can be accessed using a URL. The parameters of the interfaces are sent as XML documents. These documents are sent as the payload of the HTTP request.

This integration model provides language and location independence.

Invoking a servlet requires a client with the ability to construct and parse XML schemas and send the XML document over HTTP. There is no dependency on any specific XML library.

2 Oracle Real-Time Collaboration Integration Services Architecture and Concepts

This chapter discusses the following concepts that are essential to understanding and using Oracle Real-Time Collaboration Integration Services:

- Architecture
- Sites
- Users
- Security and Authentication
- Invoking Oracle Real-Time Collaboration Integration Services
- Oracle Real-Time Collaboration Integration Services SOAP Operations
- Steps to Integrate Oracle Real-Time Collaboration Integration Services

> **See Also:** See the following Web site for more information about the architecture of Oracle Real-Time Collaboration Integration Services:
>
> `http://`*`hostname`*`:`*`port`*`/imtapp/ws`
>
> Contact your administrator for the hostname and port of Oracle Real-Time Collaboration Integration Services. This Web site also includes the following additional resources:
>
> - Sample applications, including demonstration programs that call Web services through HTTP POST operations and sample Java code.
> - Links to service URLs, WSDL files, proxy JAR files, and endpoints for each Web service.
> - Sample input and output XML
> - Additional details about specific concepts and services

Architecture

Oracle Real-Time Collaboration Integration Services is a suite of SOAP-based Web services and direct URL-based services that allow you to extend any enterprise or desktop application with the capabilities of Oracle Real-Time Collaboration.

The use of SOAP allows for the consumption of the services by clients from multiple platforms. As a result, the only requirement is the ability to compose and parse XML. At the same time, the SOAP standard allows rapid productivity gains in common developmental platforms, such as Java and C#, through the use of standard SOAP libraries. Proxies for Java are provided for each service with the Oracle Real-Time Collaboration deployment. The use of the HTTP(S) protocol allows the most common network architectures to use Oracle Real-Time Collaboration Integration Services.

Sites

An application that uses Oracle Real-Time Collaboration Integration Services is called a site. An identifier for such an integrating application is called a site ID.

Oracle Real-Time Collaboration uses this concept of sites to provide an integrating application specific customization of an Oracle Real-Time Collaboration deployment. The site ID is used for authenticating a site when it invokes Oracle Real-Time Collaboration Integration Services.

Creating Sites

A site can be created through the Oracle Real-Time Collaboration Web application Sites tab. After creating a site, you can set up site-specific properties and authorizations to control the interaction of the site with Oracle Real-Time Collaboration.

See Also: Chapter 8, "Oracle Real-Time Collaboration Sites" in *Oracle Real-Time Collaboration Administrator's Guide* and Oracle Real-Time Collaboration online help for information about creating and configuring sites.

Users

Oracle Real-Time Collaboration uses a standards-based LDAP store (Oracle Internet Directory) as the source of users. When Oracle Real-Time Collaboration is deployed, it is configured with a particular instance of an LDAP store. This directory determines the Oracle Real-Time Collaboration user community.

Users of integrating applications fall into the following two categories:

- Same user address space as Oracle Real-Time Collaboration
- Different user address space from Oracle Real-Time Collaboration

Applications that share the same user address space can maintain their users' identity in Oracle Real-Time Collaboration.

Guest (or proxy) users can be provided access to a certain limited set of features through Oracle Real-Time Collaboration Integration Services including the following:

- Viewing presence of regular users
- Participating in chat sessions with regular users

Security and Authentication

When any integrating application invokes a Web service, the integrating application (or site) needs to be authenticated as a valid site that is authorized to invoke Oracle Real-Time Collaboration's functionality.

Using Site Credentials

A site may call Oracle Real-Time Collaboration Integration Services in the following ways:

- **Tight integration**: In response to a user action, a site can call Oracle Real-Time Collaboration Integration Services as a Web service. The results from the Web Service calls are then presented to the user.
- **Loose integration**: In some cases, such as when an application has a browser-based interface, a site can provide URLs that point directly to the Oracle Real-Time Collaboration server. An example of this is a presence URL in which the site generates and includes an URL that fetches presence information directly from the Oracle Real-Time Collaboration Server. See Chapter 4, "Presence and Live Help Integration" for more information.)

In either case, the site has to communicate with Oracle Real-Time Collaboration Integration Services using its site ID and site authorization token as initial credentials. Once a session is established, the session credentials are used for subsequent requests.

Using User Credentials

Oracle Real-Time Collaboration Integration Services can also be used to also create end-user authenticated applications. These applications are authenticated with end-user credentials or proxy credentials obtained by the site on behalf of the user, which are then submitted for authentication.

Applications that submit user credentials when using the SOAP services are limited to functionality that is relevant only to that user. For example, they cannot schedule or manage conferences for others or obtain presence information of other users.

Invoking Oracle Real-Time Collaboration Integration Services

Oracle Real-Time Collaboration Integration Services can be invoked from any of the following:

- Application server
- Native client
- Browser

Oracle Real-Time Collaboration Integration Services SOAP Operations

The following services are available:

- Authentication Services: Provide a secure authentication and authorization mechanism for users and sites to access all other Oracle Real-Time Collaboration Integration Services. The authentication mechanism involves the issuing and validation of Authentication services tokens.
- Configuration Services: Allow clients to query and update configuration information from an Oracle Real-Time Collaboration deployment.
- Pre-Conference Management Services: Schedule, update, and list conferences
- Post-Conference Management Services: Retrieve and specify conference publishing options, and retrieve playback-related information for a particular conference.
- Presence Services: Allow clients to obtain the presence of an entity.

- Reporting Services: Provide a mechanism for clients to generate reports based on finished conferences. Clients can retrieve conference attendees, lists of finished conferences based on their date and conference ID, as well as more specific conference details.

Steps to Integrate Oracle Real-Time Collaboration Integration Services

This section describes the steps you would typically perform to build applications with Oracle Real-Time Collaboration Integration Services.

Step 1: Review Oracle Real-Time Collaboration Integration Services Web Services

Oracle Real-Time Collaboration Integration Services provides SOAP-based Web services that enable you to integrate Oracle Real-Time Collaboration with any application. Select the Web services you would like to integrate into your application.

See Chapter 3, "Oracle Real-Time Collaboration Integration Services SOAP Methods" for a list of these Web services.

Step 2: Download Service Definitions

Determine which Web services you will be using and download the corresponding service definition, defined in a WSDL. The WSDL file allows for the auto generation of client stubs in most development languages.

See Chapter 3, "Oracle Real-Time Collaboration Integration Services SOAP Methods" for URLs to the service definitions of the Web services.

Step 3: Obtain Host Name, Site ID and Authentication Token

Obtain the host name of the Oracle Real-Time Collaboration deployment from your administrator.

Ask your administrator to create an Oracle Real-Time Collaboration site for you. You will need the site ID and the authentication token for this site.

See Chapter 8, "Oracle Real-Time Collaboration Sites" in *Oracle Real-Time Collaboration Administrator's Guide* for information about creating sites, site IDs, and authentication tokens.

Step 4: Integrate Oracle Real-Time Collaboration into Your Application

Call the appropriate Web services from your application. You may do this through an HTTP POST operation or a Java client.

Call Oracle Real-Time Collaboration Integration Services through HTTP POST Operations

You may do this from your application by submitting the required parameters through an HTTP POST operation. Each request sent must be accompanied with the initial site or user credentials, or appropriate tokens obtained through Authentication Services.

Call Oracle Real-Time Collaboration Integration Services from Java Client

You may also call Oracle Real-Time Collaboration Integration Services from a Java client by using the appropriate proxy JAR file. To obtain the URL of this proxy JAR file, see the Web service's endpoint URL.

See Chapter 3, "Oracle Real-Time Collaboration Integration Services SOAP Methods" for endpoint URLs of a particular Web service.

3 Oracle Real-Time Collaboration Integration Services SOAP Methods

This chapter describes the available Oracle Real-Time Integration Services methods, the parameters for each of these methods, and the data types used in the requests and responses.

The following services are available:

- Authentication Services
- Configuration Services
- Pre-Conference Management Services
- Post-Conference Management Services
- Presence Services
- Reporting Services

For an online demonstration of the operations included in each of these services, as well as the services' Java proxy, refer to the Service URL of the service. For a formal definition of the service, refer to the Service Definition URL.

Authentication Services

Authentication Services provides a secure authentication and authorization mechanism for users and sites to access Oracle Real-Time Collaboration Integration Services. The authentication mechanism involves the issuing and validation of Authentication Services tokens.

A site or a user that wishes to access Oracle Real-Time Collaboration Integration Services would first obtain an Authentication Services token by providing site or user credentials. From there on, until the time that the token expires, the site or user will not be required to provide the password or authentication token for further Oracle Real-Time Collaboration Integration Services requests. The token obtained may then be used for further Oracle Real-Time Collaboration Integration Services requests.

Authentication Services exposes the following Web services:

- Create Session
- Request Token
- Request Multiple Tokens
- Request Token for Credentials
- End Session
- Get Token Status

Service URL

http://<*hostname:portnum*>/imtapp/ws/AuthenticationService

Service Definition

http://<*hostname:portnum*>/imtapp/ws/AuthenticationService?WSDL

Create Session

Returns an Authentication Services session token (a TokenObject) when provided with an authentic username and password or SiteID and AuthToken as input. An additional parameter that defines the type of client requesting the token is required. This is set to "USER" or "SITE".

Session tokens may be used to generate more tokens. Session tokens currently only have a time-based expiration policy; this policy can be configured at install time.

Request Parameters of createSession

The following are the request parameters for the createSession service:

Table 3–1 Parameters of createSession

Parameter	Description
param0	String containing username or site ID
param1	String containing password or an authentication token
param2	String containing either "USER" or "SITE". This defines the type of client requesting the token.

TokenObject

Represents an Authentication Services token.

Table 3–2 Parameters of TokenObject

Parameter	Description
token	String representing the token
tokenAttributes	Array of type AdhocAttribute. This array contains two elements: ▪ name 1: TOKENID ▪ value 1: The value of TOKENID ▪ name 2: EXPIRY_TIME ▪ value 2: The value of the token's expiration time in UTC
tokenIndex	String

AdhocAttribute

Represents an attribute (a name-value pair).

Table 3–3 Parameters of AdhocAttribute

Parameter	Description
name	String containing name of the attribute
value	String containing value of the attribute

Request Token

Returns a TokenObject when executed successfully. It takes the user ID or Site ID, session token, TokenProperties (properties of the token being requested) and a PolicyObject as input.

Request Parameters of requestToken

The following are the request parameters for the requestToken service:

Table 3–4 Parameters of requestToken

Parameter	Description
param0	String containing username or site ID
param1	String containing password, or session token
param2	TokenProperties
param3	PolicyObject

TokenProperties

Encapsulates the properties of the token being requested. Typically, the purpose (policy) for which the requested token will be used is contained in this object.

Table 3–5 Parameters of TokenProperties

Parameter	Description
tokenPolicy	String, the token policy, such as "SESSION::PRESENCE" This string may be a combination of the following strings separated by a double colon (::): ■ SESSION ■ PRESENCE ■ PROXY ■ REPORTING ■ PRECONFERENCE ■ POSTCONFERENCE ■ GUESTCHAT
tokenAttributes	AdhocAttribute. If you have requested a token with a proxied policy (for example, if the requested token has a policy of "PROXY::GUESTCHAT"), you must specify the following attribute: ■ name: PROXY_USER ■ value: The username of the proxy user, for example orcladmin

PolicyObject

Encapsulates additional information required by Authentication Services in order to authorize the requestToken request specific to the policy of the token requested. Information provided in this object would typically depend on the tokenPolicy set in the TokenProperties object of the request.

Table 3–6 Parameters of PolicyObject

Parameter	**Description**
policyArray	AdhocAttribute

Request Multiple Tokens

Returns an array of TokenObject when executed successfully. It takes the user ID or Site ID, session token, TokenProperties (properties of the tokens being requested) and an array of PolicyObjects as input.

Request Parameters of requestMultipleTokens

The following are the request parameters for the requestMultipleTokens service:

Table 3–7 Parameters of requestMultipleTokens

Parameter	Description
param0	String containing user name or site ID
param1	String containing user password or the site authentication token
param2	TokenProperties object that encapsulates the properties of the token being requested
param3	Array of PolicyObject that describes the types of tokens being requested

Request Token for Credentials

Returns a TokenObject when executed successfully. It takes a CredentialObject, TokenProperties (properties of the token being requested), and a PolicyObject (specifies any parameters that are specific to the implementation of the policy scheme that governs the generation of the token being requested) as input.

Request Parameters of requestTokenForCredentials

The following are the request parameters for the requestTokenForCredentials service:

Table 3–8 Parameters of requestTokenForCredentials

Parameter	Description
param0	CredentialObject
param1	TokenProperties
param2	PolicyObject

CredentialObject

Represents information required by the Request Token for Credential service.

Table 3–9 Parameters of CredentialObject

Parameter	Description
credentialType	String containing type of credential being used: ▪ "TOKEN" if an Authentication Service Token is being used ▪ “PASSWORD” if the SiteAuthKey or the user password is being used
entityType	String containing type of entity (“SITE” or “USER”)
entityCredential	String containing password or SiteAuthKey or session token
entityID	String containing site ID or user ID

End Session

This service invalidates a token and returns the invalidated TokenObject when provided with an authentic username and token or siteID and the token to be invalidated as input. The End Session operation determines if the token requested to be invalidated was issued to the user or site passed in the request. If this condition is satisfied, the token and all its child tokens are invalidated.

Request Parameters of endSession

The following are the request parameters for the endSession service:

Table 3–10 Parameters of endSession

Parameter	Description
param0	String representing user name or site ID
param1	String representing token of user or site

Get Token Status

This service returns the status of a token encapsulated in a TokenObject object when provided with an authentic username and token or siteID and the token as input.

Request Parameters of getTokenStatus

The following are the request parameters for the getTokenStatus service:

Table 3–11 Parameters of getTokenStatus

Parameter	Description
param0	String representing user name or site ID
param1	String representing token of user or site

Configuration Services

Implemented as a SOAP RPC-style Web service and provides the ability for clients to query and update configuration information from an Oracle Real-Time Collaboration deployment.

The following are some examples of its functionality:

- Obtain and modify Oracle Real-Time Collaboration configuration settings. The following are some examples of settings:
  - Whether it is mandatory to set a conference key for conferences scheduled by a given site
  - The value of the "early join time" for a conference
- Obtain Oracle Real-Time Collaboration Integration Services catalog information (for example, supported versions of an Oracle Real-Time Collaboration Web service available from the deployment)

Since Configuration Services provide information about other Oracle Real-Time Collaboration Web services, the signatures of the operations that it exposes will not change. New operations may be added in future releases, but what is released will not change. The following are the only features that will change in a new release:

- The list of configuration information available (new entries may be added)
- New operations that provide new functionality

Configuration Services exposes the following Web service:

- Get Configuration

Service URL

http://<*hostname:portnum*>/imtapp/ws/ConfigurationService

Service Definition

http://<*hostname:portnum*>/imtapp/ws/ConfigurationService?WSDL

Configuration Entries

Configuration entries form the basis for accessing the functionality offered by Configuration Services. Any configuration information that is exposed through Configuration Services has an entry in the Configuration Services directory. Each entry in the Configuration Services directory is identified by a name is used to access it. The entries in the Configuration Services directory is structured in a tree as follows:

```
rtc
+--servicecatalog
|  +--authenticationservice
|  |  +--versions
|  |  +--url
|  |     +--endpoint
|  |     +--wsdl
|  +--preconferenceservice
|     +--versions
|     +--url
|        +--endpoint
|        +--wsdl
|
+--application
|  +--conferencekeyrequired
|  +--presenceenabled
|  +--schedulingenabled
|
+--clients
   +--win32
   |  +--addin
   |  |  +--version
   |  |  +--url
   |  +--messenger
   +--addin
   |  +--win32
   +--messenger
      +--win32
```

Fullname

The fullname of a configuration entry is the handle to the configuration in the Configuration Services directory. It is a string that references the configuration entry and is composed as a period (.) delimited path to the configuration entry.

The following are examples of fullnames:

- The fullname for the property that determines if a conference key is mandatory for conferences scheduled by a given site is rtc.application.conferencekeyrequired
- The fullname for the Authentication Services versions supported is rtc.servicecatalog.authenticationservice.versions

The table at the end of this section lists all the entries in the Configuration Services directory along with the fullnames required to reference them.

Namespace and Property Name

Every entry in the Configuration Services directory can also be referenced by its property name along with its namespace. A namespace represents a grouping of Configuration Services entries and the property name is the name of a specific entry under a given namespace.

For example, the entry for Authentication Services versions supported can be referenced with namespace rtc.servicecatalog.authenticationservice and property name "versions". It may alternatively also be referenced with namespace rtc.servicecatalog and property name authenticationservice.versions.

Configuration Service Directory

The following table lists the fullnames of available configuration entries:

Table 3–12 Configuration Entries

Configuration Entry Fullname	Description	Public?
rtc.application	Has no value by itself. It is the top-level namespace for Oracle Real-Time Collaboration configuration properties.	Yes
rtc.application.admin_email	Represents the Oracle Real-Time Collaboration administrator email address.	Yes
rtc.application.auth_chat_enabled	Determines if a site or a user is authorized to obtain chat tokens.	No
rtc.application.auth_presence_enabled	Determines if a site or a user is authorized to obtain presence tokens.	No

Table 3–12 (Cont.) Configuration Entries

Configuration Entry Fullname	Description	Public?
rtc.application.auth_proxy_enabled	Determines if a site or a user is authorized to obtain proxy tokens.	No
rtc.application.auth_session_enabled	Determines if a site or a user is authorized to obtain session tokens.	No
rtc.application.conference_key_required	Determines if the conferences scheduled by a site or a user are required to have a conference key.	No
rtc.application.current_server_time	Represents the current time on the server.	Yes
rtc.application.early_join_minutes	Represents the early join minutes for conferences created.	Yes
rtc.application.guest_user_access_enabled	Determines if guest user access is enabled.	No
rtc.application.im_nickname	*To be documented*	No
rtc.application.im_suffix	*To be documented*	No
rtc.application.max_request_size	Determines the maximum size of the Oracle Real-Time Collaboration Web services requests.	Yes
rtc.application.max_response_size	Determines the maximum size of Oracle Real-Time Collaboration Web services responses.	Yes
rtc.application.scheduling_access_enabled	Determines if a site or user is authorized for scheduling.	No
rtc.application.token_life	Determines the life of Authentication Services tokens in seconds.	Yes
rtc.application.version_identifier	Represents the version of the Oracle Real-Time Collaboration application.	No
rtc.application.im_domain	Represents the Oracle Messenger domain of the Oracle Messenger server	
rtc.servicecatalog	Has no value by itself. It is the top-level namespace for all service catalog entries.	Yes
rtc.servicecatalog.service_home	Represents Oracle Real-Time Collaboration Web services home (no-ssl home if enabled. If SSL is required, the SSL home is returned.)	Yes
rtc.servicecatalog.service_ssl_home	Represents Oracle Real-Time Collaboration Web services SSL home.	Yes
rtc.servicecatalog.authenticationservice	Has no value by itself. It is the top-level namespace for Authentication Services.	Yes
rtc.servicecatalog.authenticationservice.versions	Lists all supported versions of Authentication Services in the form of a comma-delimited string.	Yes
rtc.servicecatalog.authenticationservice.url	Has no value.	Yes
rtc.servicecatalog.authenticationservice.url.endpoint	Represents Authentication Services endpoint URL relative to the Web services home.	Yes
rtc.servicecatalog.authenticationservice.url.wsdl	Represents Authentication Services WSDL URL relative to the Web services home.	Yes
rtc.servicecatalog.preconferenceservice	Has no value by itself. It is the top-level namespace for Pre-Conference Management Services.	Yes
rtc.servicecatalog.preconferenceservice.versions	Lists all supported versions of Pre-Conference Management Services in the form of a comma-delimited string.	Yes
rtc.servicecatalog.preconferenceservice.url	Has no value.	Yes

Table 3–12 (Cont.) Configuration Entries

Configuration Entry Fullname	Description	Public?
rtc.servicecatalog.preconferenceservice.url.endpoint	Represents Pre-Conference Management Services endpoint URL relative to the Web services home.	Yes
rtc.servicecatalog.preconferenceservice.url.wsdl	Represents Pre-Conference Management Services WSDL URL relative to the Web services home.	Yes
rtc.servicecatalog.presenceservice	Has no value by itself. It is the top-level namespace for Presence Services.	Yes
rtc.servicecatalog.presenceservice.versions	Lists all supported versions of Presence Services in the form of a comma-delimited string.	Yes
rtc.servicecatalog.presenceservice.url	Has no value.	Yes
rtc.servicecatalog.presenceservice.url.endpoint	Represents Presence Services endpoint URL relative to the Web services home.	Yes
rtc.servicecatalog.presenceservice.url.wsdl	Represents Presence Services WSDL URL relative to Web services home.	Yes
rtc.servicecatalog.postconferenceservice	Has no value by itself. It is the top-level namespace for Post-Conference Services.	Yes
rtc.servicecatalog.postconferenceservice.versions	Lists all supported versions of Post-Conference Services in the form of a comma-delimited string.	Yes
rtc.servicecatalog.postconferenceservice.url	Has no value.	Yes
rtc.servicecatalog.postconferenceservice.url.endpoint	Represents Post-Conference Services endpoint URL relative to the Web services home.	Yes
rtc.servicecatalog.postconferenceservice.url.wsdl	Represents Post-Conference Services WSDL URL relative to the Web services home.	Yes
rtc.clients	Has no value. It is the top-level namespace for Oracle Real-Time Collaboration clients.	Yes
rtc.clients.addin	Has no value	Yes
rtc.clients.addin.win32	Has no value	Yes
rtc.clients.addin.win32.version	Returns the currently supported addin version	Yes
rtc.clients.addin.win32.url	Returns the download URL for the addin client	Yes
rtc.clients.messenger	Has no value	Yes
rtc.clients.messenger.win32	Has no value	Yes
rtc.clients.messenger.win32.version	Returns the currently supported Oracle Messenger client version	Yes
rtc.clients.messenger.win32.url	Returns the download URL for the Oracle Messenger client	Yes
rtc.clients.win32	Has no value	Yes
rtc.clients.win32.addin	Has no value	Yes
rtc.clients.win32.addin.version	Returns the currently supported addin version	Yes
rtc.clients.win32.addin.url	Returns the download url for the addin client	Yes
rtc.clients.win32.messenger	Has no value	Yes
rtc.clients.win32. messenger.version	Returns the currently supported Oracle Messenger client version	Yes
rtc.clients.win32. messenger.url	Returns the download URL for the Oracle Messenger client	Yes

Get Configuration

Obtains Oracle Real-Time Collaboration configuration information; returns a GetConfigurationResponse.

Request Parameters of getConfiguration

The following are the request parameters for the getConfiguration service:

Table 3–13 Parameters of getConfiguration

Parameter	Description
param0	GetConfigurationRequest object

GetConfigurationRequest

Represents information required by the Get Configuration service.

Table 3–14 Parameters of GetConfigurationRequest

Parameter	Description
entityID	String containing site ID or user ID of the entity making the Web service call.
entityToken	String containing authentication token for this entity. The token can be obtained using Authentication Services.
namespace	The optional namespace of the properties being requested.
properties	An array of property name strings for which values are requested. These property strings, when concatenated to the end of the namespace with a period (".") should form a valid configuration service entry.
adhocAttribute	Optional array of AdhocAttribute, name-value pair attributes that can be used to provide additional type-specific data.

GetConfigurationResponse

Represents information returned by the Get Configuration service.

Table 3–15 Parameters of GetConfigurationResponse

Parameter	Description
statusCode	Integer status code for the Web service call
statusMessage	Character string, authentication token for this entity. The token can be obtained using Authentication Services.
statusDetails	The optional namespace of the properties being requested
namespace	String containing additional information regarding the status of the Web service call
properties	An AdhocAttribute array that contain property names and values for all the configuration entries in the query. If there was an issue obtaining the value of a specific configuration entry, its value is set to null.
adhocAttributes	Array of AdhocAttribute, name-value pair attributes that can be used to specify additional output elements from the Web service

The following table describes the status codes of GetConfigurationResponse:

Table 3–16 Status Codes of GetConfigurationResponse

Status	Status Code	Description
NO_ERROR	0	Web service call completed successfully
INVALID_SITE_ID	4	The site ID provided was invalid
NOT_AUTHORIZED	1032	The entity is not authorized to make the request
UNEXPECTED_ERROR	1090	An unexpected error occurred and the operation cannot be completed successfully

Pre-Conference Management Services

Schedules, updates, deletes, and lists conferences; supports the following services:

- Instant Conference
- Schedule Conference
- Update Conference
- Delete Conference
- List Upcoming Conferences

Service URL

http://<*hostname:portnum*>/imtapp/ws/PreConferenceService

Service Definition

http://<*hostname:portnum*>/imtapp/ws/PreConferenceService?WSDL

Instant Conference

This URL service allows an integrating application to create an instant conference on the Oracle Real-Time Collaboration system. Once the instant conference has been successfully created, the system returns a join URL to the host. The host can then use the join URL to launch the Oracle Real-Time Collaboration console. The difference between this service and the Schedule Conference service is that this service schedules and starts a conference instantly.

Service URL

http://<*hostname:portnum*>/imtapp/app/OracleRTCService?operation=instantMeetin g&xmlin=<*xml-document*>

Input Parameters

The parameter "xmlin" is the XML document that contains all the input parameters for the service. This document should be based on the following XML schema:

http://<*hostname:portnum*>/imtapp/app/instantMtgReq.xsd

The following table describes each of the elements in the XML schema.

Table 3–17 Input Parameters of Instant Conference

XML Schema Element	Required?	Data Type	Valid Values	Description
meetingType	No	String	REGULAR, PUBLIC, RESTRICTED, and REGISTERED	Meeting type
siteid	Yes	String		Site ID of the integrating application
password	No	String		Conference password, if any
meetingTitle	No	String		Name of the conference being scheduled
hostExitURL	No	URL	Valid URL	URL of the page that pops-up on the host side after a conference has ended. This feature can be used to collect feedback/exit surveys related to the conference.

Table 3–17 (Cont.) Input Parameters of Instant Conference

XML Schema Element	Required?	Data Type	Valid Values	Description
attendeeExitURL	No	URI	Valid URL	URL of the page that pops-up on the attendee side after a conference has ended. This feature can be used to collect feedback/exit surveys related to the conference.
attendeeStartURL	No	URI	Valid URL	URL of the page that pop-up on the attendee side when the attendee joins a conference. This feature can be used to lead the attendee to some informational page about the conference. Example, sales related page in a sales conference.
returnType	No	String	XML, text	When return type is set to XML, an XML document is returned. When return type is set to text, then the return type is a URL redirect.

Return Parameters

The format of the return document depends on the returnType parameter.

If the returnType parameter is XML, then upon successful execution, the response is an XML document whose schema is defined by meetingResp.xsd:

http://<*hostname:portnum*>/imtapp/app/meetingResp.xsd

Table 3–18 Return Parameters of Instant Conference

XML Schema Element	Required?	Data Type	Description
responseCode	Yes	Integer	Return code is always 0 on successful execution
mtgId	Yes	String	ID of the scheduled conference
playbackURL	Yes	Any URL	URL to be used for playing back the conference

If the returnType is text, then the join conference URL is returned with the redirect header. This is useful when the Instant Conference service is called from a browser window. In this case, the browser window is automatically redirected to the URL location, which automatically launches Oracle Web Conferencing to start the Web conference.

Error Handling

The format of the error document depends on the returnType parameter.

If the returnType is text, then the appropriate error message is returned as a text/HTML page.

If the returnType is XML, if there is any error (response code is non-zero), the response to the call is an XML document based on errorMsg.xsd.

http://<*hostname:portnum*>/imtapp/app/errorMsg.xsd

Table 3–19 Error Handling Parameters of Instant Conference

XML Schema Element	Required?	Data Type	Description
responseCode	Yes	Integer	Return code is always 0 on successful execution
message	Yes	String	A short description of the error

Schedule Conference

Allows an integrating site or end user application to schedule a conference; returns a ScheduleResponse object.

Request Parameters of scheduleConference

The following are the request parameters for the scheduleConference service:

Table 3–20 Parameters of scheduleConference

Parameter	Description
param0	ScheduleRequest object

ScheduleRequest

Represents information required by the scheduleConference and updateConference services.

Table 3–21 Parameters of ScheduleRequest

Parameter	Description
entityID	Character string representing the site ID or user ID of the entity making the Web service call.
entityToken	Character string representing the authentication token for this entity. The token can be obtained from Authentication Services.
scheduleOptions	ScheduleOptions object representing various scheduling parameters
adhocAttributes	Optional array of AdhocAttribute objects containing name-value pair attributes that can be used to specify additional inputs to the Web service.

ScheduleOptions

Specifies various scheduling parameters.

Table 3–22 Parameters of ScheduleOptions

Parameter	Description
attendees	Array of ParticipantData objects for the conference attendees
conferenceId	String representing the conference ID
description	String representing description for the conference
registeredUserOnly	Boolean flag indicating whether only registered users may attend the Web conference. If set to null, a default value of true is used.
publiclyListed	Boolean flag indicating whether this Web conference can be publicly listed. If set to null, a default value of false is used.
hostInvitedOnly	Boolean flag indicating whether only registered users invited by the host can attend the Web conference. If set to true, only registered users will be allowed to attend the conference and the registeredUserOnly flag must be set to true. If set to null, a default value of true is used.

Table 3–22 (Cont.) Parameters of ScheduleOptions

Parameter	Description
enrollmentRequired	Boolean flag indicating whether the attendees for this conference need to go through an enrollment process in order to join the conference. If set to true, only registered users will be allowed for the conference and the registeredUserOnly flag must be set to true. In addition, a conference key must be provided. If enrollmentRequired is set to null, a default value of false is used.
dialinInfo	String
startDate	Date object, conference start time
endDate	Date object, conference end time
organizer	ParticipantData object, conference organizer
conferenceKey	String, conference key
publishAttendees	Boolean flag indicating whether the conference's list of attendees should be included in your e-mail invitation to each attendee. If set to null, a default value of false is used.
sendEmailNotification	Boolean value indicating whether e-mail invitations should be sent for the conference.
siteId	String, site ID
title	String, conference title
adhocAttributes	Array of AdhocAttribute objects containing name-value pair attributes that can be used to specify additional scheduling options.

ParticipantData

Represents a conference attendee or organizer.

Table 3–23 Parameters of ParticipantData

Parameter	Description
firstName	String, participant’s first name
lastName	String, participant’s last name
emailAddr	String, e-mail address of participant
userguid	String, user’s GUID if the participant is a registered user
userid	String, user’s ID if the participant is a registered user
adhocAttributes	Array of AdhocAttribute objects containing name-value pair attributes that can be used to specify additional scheduling options.

ScheduleResponse

Represents information returned from the Schedule Conference and Update Conference services.

Table 3–24 Parameters of ScheduleResponse

Parameter	Description
statusCode	Integer, status code for the Web service call

Table 3–24 (Cont.) Parameters of ScheduleResponse

Parameter	Description
statusMessage	Character string, authentication token for this entity. The token can be obtained using Authentication Services.
statusDetails	The optional namespace of the properties being requested
scheduleResult	ScheduleResult object encapsulating the result of the scheduling operation; includes the conference ID along with the attendee and host URLs used to join the conference.
adhocAttributes	Array of AdhocAttribute, name-value pair attributes that can be used to specify additional output elements from the Web service

The following table describes the status codes of scheduleConference and updateConference services:

Table 3–25 Status Codes of scheduleConference and updateConference Services

Status	Status Code	Description
NO_ERROR	0	Web service call completed successfully
INVALID_SITE_ID	4	The site ID provided was invalid
INVALID_USER_ID	5	The attendee list could not be retrieved for this conference.
INVALID_USER_GUID	6	The entity is not authorized to make the request
MEETING_START_TIME_IS_NULL	351	A null conference start time was provided.
MEETING_END_TIME_IS_NULL	352	A null conference end time was provided.
MEETING_START_TIME_ERR	356	There was a problem with the conference start time.
MEETING_END_TIME_ERR	357	There was a problem with the conference end time.
MEETING_MTGNOTCREATED_ERR	358	The conference could not be created.
MEETING_PUBLIC_MTGTYPE_ERR	362	The current site does not allow public conferences to be scheduled.
MEETING_INVALID_PASSWORD	367	An invalid conference key was provided.
PROCESSED_WITH_NON_FATAL_ERRORS	1013	A non-critical error occurred while processing some of the conference attendees.
NOT_AUTHORIZED	1032	The entity is not authorized to make the request
ORGANIZER_ID_MISSING	1061	The host information is missing for this conference.
USER_CANNOT_SCHEDULE_FOR_ ORGANIZER	1062	The user who submitted the schedule request was different from the conference host.

Table 3–25 (Cont.) Status Codes of scheduleConference and updateConference

Status	Status Code	Description
ERROR_PROCESSING_PARTICIPANT	1064	An error occurred while processing conference participants.
ERROR_SENDING_EMAIL_INVITES	1065	An error occurred while attempting to send email invites to conference participants.
UNEXPECTED_ERROR	1090	An unexpected error occurred and the operation cannot be completed successfully

ScheduleResult

Represents information returned from the Schedule Conference and Update Conference services.

Table 3–26 Parameters of ScheduleResult

Parameter	Description
attendeeURL	String, URL where attendees can join the conference
hostURL	String, URL where the host can join the conference
conferenceID	String, conference ID
adhocAttributes	Optional array of AdhocAttribute containing name-value pair attributes that can be used to specify additional output elements from the Web service

Update Conference

Allows an integrating site or end user application to update a previously scheduled conference. This function returns a ScheduleResponse object.

Request Parameters of updateConference

The following are the request parameters for the updateConference service:

Table 3–27 Parameters of updateConference

Parameter	Description
param0	ScheduleRequest object

Delete Conference

Allows an integrating site or end user application to delete a previously scheduled conference.

Returns a StatusResponse object.

Request Parameters of deleteConference

The following are the request parameters for the deleteConference service:

Table 3–28 Parameters of deleteConference

Parameter	Description
param0	DeleteConferenceRequest object

DeleteConferenceRequest

Represents information required by the deleteConference service.

Table 3–29 Parameters of DeleteConferenceRequest

Parameter	Description
entityID	Character string representing the site ID or user ID of the entity making the Web service call.
entityToken	Character string representing the authentication token for this entity. The token can be obtained from Authentication Services.
conferenceID	String containing the conference ID of the conference to be deleted.
adhocAttributes	Optional array of AdhocAttribute objects containing name-value pair attributes that can be used to specify additional inputs to the Web service.

StatusResponse

Represents information returned from the deleteConference service.

Table 3–30 Parameters of StatusResponse

Parameter	Description
statusCode	Integer, status code for the Web service call
statusMessage	Character string, authentication token for this entity. The token can be obtained using Authentication Services.
statusDetails	The optional namespace of the properties being requested
adhocAttributes	Array of AdhocAttribute, name-value pair attributes that can be used to specify additional output elements from the Web service

The following table describes the status codes of the deleteConference service:

Table 3–31 Status Codes of deleteConference Service

Status	Status Code	Description
NO_ERROR	0	Web service call completed successfully

Table 3–31 (Cont.) Status Codes of deleteConference Service

Status	Status Code	Description
INVALID_MEETING_ID	3	An invalid conference ID was provided
INVALID_SITE_ID	4	The site ID provided was invalid
MEETING_CANNOT_BE_UPDATED_NOW	354	The meeting could not be modified at the current time
NOT_AUTHORIZED	1032	The entity is not authorized to make the request
UNEXPECTED_ERROR	1090	An unexpected error occurred and the operation cannot be completed successfully

List Upcoming Conferences

Allows an integrating site to retrieve the list of upcoming conferences for a site, with options to filter based on conferenceType, userID (for a site), start date and end date.

Returns an UpcomingConferencesResponse object that contains a list of upcoming conferences for a site or user.

Request Parameters of listUpcomingConferences

The following are the request parameters for the listUpcomingConferences service:

Table 3–32 Parameters of listUpcomingConferences

Parameter	Description
param0	UpcomingConferencesRequest object

UpcomingConferencesRequest

Represents information required by the listUpcomingConferences service.

Table 3–33 Parameters of UpcomingConferencesRequest

Parameter	Description
entityId	String, site ID or user ID of the entity making the Web service call
entityToken	String representing the authentication token for this entity. Obtain this token by using Authentication services.
userID	String containing the user ID used to filter upcoming conferences. Conferences where this user participates as either host or attendee will be returned. If a null user ID is provided, the upcoming conferences will not be filtered based on host or attendee user IDs.
registeredUserOny	If Boolean flag is non-null, upcoming conferences having a registeredUserOnly flag equal to the filter value will be retrieved. Otherwise, upcoming conferences retrieved will not be filtered based on the value of this flag.
publiclyListed	If Boolean flag is non-null, upcoming conferences having a publiclyListed flag equal to the filter value will be retrieved. Otherwise, upcoming conferences retrieved will not be filtered based on the value of this flag.
hostInvitedOnly	If Boolean flag is non-null, upcoming conferences having a hostInvitedOnly flag equal to the filter value will be retrieved. Otherwise, upcoming conferences retrieved will not be filtered based on the value of this flag.
fromStartDate	Optional Date object for filtering the list by earliest start date
toStartDate	Optional Date object for filtering the list by latest start date
adhocAttrributes	Optional array of AdhocAttribute objects containing name-value pair attributes that can be used to specify additional inputs to the Web service.

UpcomingConferencesResponse

Represents information returned from the listUpcomingConferences service.

Table 3–34 Parameters of UpcomingConferencesResponse

Parameter	Description
statusCode	Integer, status code for the Web service call
statusMessage	Character string, authentication token for this entity. The token can be obtained using Authentication Services.
statusDetails	The optional namespace of the properties being requested
conferenceDetails	UpcomingConferenceDetails object including conference ID and title, name of host, conference start and end times, and conference URL
adhocAttributes	Array of AdhocAttribute, name-value pair attributes that can be used to specify additional output elements from the Web service

The following table describes the status codes of the listUpcomingConferences service:

Table 3–35 Status Codes of deleteConference Service

Status	Status Code	Description
NO_ERROR	0	Web service call completed successfully
INVALID_USER_ID	5	An invalid user ID was provided
ERROR_RETRIEVING_CONFERENCE_LIST	376	An error occurred while retrieving conference list
NOT_AUTHORIZED	1032	The entity is not authorized to make the request
UNEXPECTED_ERROR	1090	An unexpected error occurred and the operation cannot be completed successfully

UpcomingConferenceDetails

Represents information returned by the listUpcomingConferences service:

Table 3–36 Parameters of UpcomingConferenceDetails

Parameter	Description
attributes	Array of AdhocAttribute objects for the conference.
conferenceID	String, site ID or user ID of the entity making the Web service call
conferenceStatus	String containing the conference status. Can have a value of "Not Started", "In progress", or "Waiting for Host".
conferenceTitle	String containing the conference title
registeredUserOnly	Boolean flag indicating whether only registered users may be allowed to attend the Web conference
publiclyListed	Boolean flag indicating whether this web conference can be publicly listed
hostInvitedOnly	Boolean flag indicating whether only registered users invited by the host can attend the Web conference
conferenceURL	String representing the URL where this conference can be joined
hostGUID	String representation of the host's GUID.

Table 3–36 (Cont.) Parameters of UpcomingConferenceDetails

Parameter	**Description**
hostName	String containing the name of the host
scheduledEndTime	Date object for the conference end time
scheduledStartTime	Date object for the conference start time
statusDetail	String containing detailed status information for the conference
adhocAttrributes	Optional array of AdhocAttribute objects containing name-value pair attributes that can be used to specify additional inputs to the Web service.

Post-Conference Management Services

Retrieves playback-related information for a conference.

Supports the following services:

- Get Playback URL

Service URL

http://*<hostname:portnum>*/imtapp/ws/PostConferenceService

Service Definition

http://*<hostname:portnum>*/imtapp/ws/PostConferenceService?WSDL

Get Playback URL

Obtains playback-related information for a conference. In particular, it returns the URL (in a PlaybackURLResponse object) where the conference playback can be retrieved, as well as flags indicating whether the conference was finished, published, or recorded.

Request Parameters of getPlaybackURL

The following are the request parameters for the getPlaybackURL service:

Table 3–37 Parameters of getPlaybackURL

Parameter	Description
param0	PlaybackURLRequest object

PlaybackURLRequest

Represents information required by the getPlaybackURL service.

Table 3–38 Parameters of PlaybackURLRequest

Parameter	Description
entityID	Character string representing the site ID or user ID of the entity making the Web service call.
entityToken	Character string representing the authentication token for this entity. The token can be obtained from Authentication Services.
conferenceID	The conference ID for the conference whose playback details are to be retrieved
includeConferenceKey	Optional boolean flag indicating whether the conference key should be included (in encrypted form) as part of the playback URL. If set to null, the conference key will not be included.
adhocAttributes	Optional array of AdhocAttribute containing name-value pair attributes that can be used to specify additional inputs to the Web service.

PlaybackURLResponse

Represents information returned by the Get Playback URL service:

Table 3–39 Parameters of PlaybackURLResponse

Parameter	Description
statusCode	Integer status code for the Web service call.
statusMessage	Descriptive string for the status of the Web service call.
statusDetails	String containing additional information regarding the status of the Web service call
playbackURLDetail	PlaybackURLDetails object, encapsulates the URL where the conference playback can be retrieved, as well as flags indicating whether the conference was finished, published, or recorded
adhocAttributes	Array of AdhocAttribute containing name-value pair attributes that can be used to specify additional output elements from the web service.

The following table describes the status codes of the getPlaybackURL service:

Table 3–40 Status Codes of getPlaybackURL Service

Status	Status Code	Description
NO_ERROR	0	Web service call completed successfully
MEETING_ID_IS_NULL	2	The conference ID provided was set to null
INVALID_MEETING_ID	3	An invalid conference ID was provided
NOT_AUTHORIZED	1032	The entity is not authorized to make the request

PlaybackURLDetails

Represents information related to the playback URL of a conference.

Table 3–41 Parameters of PlaybackURLDetails

Parameter	Description
playbackURL	String containing the URL for the conference playback
finishedFlag	Boolean indicating whether the conference has been finished
recordedFlag	Boolean indicating whether the conference has been recorded
publishedFlag	Boolean indicating whether the conference has been published
adhocAttributes	Optional array of AdhocAttribute containing name-value pair attributes that can be used to specify additional information relating to the playback details

Presence Services

Provides the following SOAP-based and servlet-based services that facilitate clients to obtain the presence of an entity; supports the following services

- Get Presence URL Service
- Get Presence SOAP Service
- Bulk Presence SOAP Service

Get Presence URL Service

Used when the integrating application or web site does not need to directly process the presence information but instead wants to display the presence information in a web page served to end users.

The Get Presence URL service is available in the following forms:

- Image URL service
- JavaScript/Text URL service

Service URL

http://*<hostname:portnum>*/imtapp/Presence?MODE=mode&OP=op&RES=resource &TID=tokenID&UID=userID&K1=hashKey

Input Parameters

The Presence URL service takes the following inputs as parameters in the URL:

Table 3–42 Parameters of getPresence

Parameter	Description
R_MODE	Optional parameter to specific Image ("IMG") or JavaScript ("JS") mode. Default is "JS".
R_OP	Optional parameter specific to the request mode. For example, this parameter can be used to request alternate images when using the "IMG" mode by setting its value between 1 and 5.
R_RES	Optional parameter to explicitly retrieve presence of specific resource
TID	Required parameter. TokenID for a token authorized for the Presence Service. Obtain this parameter with Authentication Services by setting "PRESENCE" in the policy option
R_UID	Required parameter. User ID (or JID) of the target presentity.
K1	SHA1 hash key used to secure the request. Further details about generating this key are given below.

The value of the K1 parameter is computed using the SHA1 Digest algorithm on a customized message. This added security is needed because the presence request would be coming from the end-user browser and not a site. Hence, it is necessary to ensure that only presence requests authorized by a site are served. The input message to the SHA1 hash algorithm is a String obtained from the URL parameters using the following logic:

- The URL parameters should be arranged in alphabetical order.
- If a parameter does not have a value assigned, it should be ignored.
- For parameters with an assigned value (other than K1), the following string should be added to the message, excluding the quotes: "<*param_name*><*param_value*>".
- After all parameters have been added, the following string should be added to the message, excluding the quotes: "<token><*token_value*>".

The Secure Hash Algorithm (SHA1) is a standard message digest algorithm, with libraries available in most programming languages. In Java, the java.security.MessageDigest class can be used for this purpose.

Response (Success)

If presence information for the JID is available and if the hash key (K1) is validated correctly, the service generates the following response:

- If the OPT parameter is set to "IMG": the service returns an image representing the presence status.
- If the OPT parameter is set to "JS": the service returns text with the Presence Status and Presence Show attributes. The show attribute represents states such as "Available", "Away", "Offline", or "DnD" ("DnD" represents the "Do Not Disturb" state). The Status attribute represents any custom display text that the Presentity may have set for that connection (for example, "Call me on my Cell").

Response (Error)

If an invalid JID is specified, or if the hash key (K1)is not validated, the service generates the following response:

- If the OPT parameter is set to "IMG": The Service returns an image representing the Error status. Deployments can customize this image to be the same as an "Offline" image, or may have a specific image to represent an "Error" status.
- If the OPT parameter is set to "JS": The Service returns text with the Presence Status and Presence Show values set to "Offline".

Get Presence SOAP Service

The SOAP service is used when an integrating application needs to obtain presence information for internal processing. This service returns detailed presence information about a target presentity (representing a user, application, or service, for example).

The service returns a PresentityStatus object, which is an aggregation of the online resources for that JID. Based on the input parameters, the returned information has the following variations:

- If a resource is specified: Returns the status of the particular resource. If the specified resource is not one of the online resources, the PresentityStatus object will not contain any resource objects.
- If no resource is specified and showAllResourcesFlag is set to true: The services returns a list of all the resource objects for that JID.

Service URL

http://<*hostname:portnum*>/imtapp/ws/PresenceService

Service Definition

http://<*hostname:portnum*>/imtapp/ws/PresenceService?WSDL

Request Parameters of getPresence

The following are the request parameters for the getPresence service:

Table 3–43 Parameters of getPresence

Parameter	Description
param0	String representing the siteID or userID of the entity making the Web service call
param1	String representing the authentication token for this entity. Obtain this token by using Authentication services.
param2	String representing the JID of the presentity whose presence is to be retrieved
param3	Optional string representing the resource of the presentity whose presence is to be retrieved. If this parameter is not set, the default resource is retrieved
param4	Optional boolean flag to indicate if all the resources should be retrieved. This is effective only if param3 is not set.

PresentityStatus

Specifies the target presentity for Get Presence SOAP Service.

Table 3–44 Parameters of PresentityStatus

Parameter	Description
allResources	Array of PresentityResource containing each of the presentity's available online resources.
userJID	String, JID for the user

PresentityResource

Specifies the resource for presentities.

Table 3–45 Parameters of PresentityResource

Parameter	Description
lastModifiedDate	Date object for the time when this resource was last modified
priority	Long indicating the priority of this resource
resourceName	String containing the resource name
show	String indicating whether the presentity should be shown as either online or offline on the current resource
status	String representing the presentity's detailed online status on the current resource

Bulk Presence SOAP Service

Used by integrating applications to obtain presence information for a list of presentities in bulk through a single Web service call. The SOAP service is an RPC-style service that returns list of detailed presence information for each of the target presentities.

Used to obtain a list of detailed presence information for each of the target presentities (such as a user, application, or service). Returns an object of type BulkPresenceResponse.

Request Parameters of getBulkPresence

The following are the request parameters for the getBulkPresence operation:

Table 3–46 Parameters of getBulkPresence

Parameter	Description
param0	BulkPresenceRequest object

BulkPresenceRequest

Represents information required for the getBulkPresence service.

Table 3–47 Parameters of BulkPresenceRequest

Parameter	Description
entityID	Character string representing the site ID or user ID of the entity making the Web service call.
entityToken	Character string representing the authentication token for this entity. The token can be obtained from Authentication Services.
entityJIDs	Array of character strings representing the JIDs of the presentities whose presence is to be retrieved
resources	Optional array of character strings representing the resource to be retrieved for each of the specified presentities. If an individual resource name is set to null, the default resource for the corresponding presentity will be retrieved. If the array itself is set to null, the default resource will be retrieved for all presentities.
allResources	Optional boolean flag indicating whether all the resources for a presentity should be retrieved. This flag will be ignored for those presentities for which a specific resource name is requested from the resources array. If set to null, only the default resource will be retrieved for all presentities for which a resource name was not provided.
adhocAttributes	Optional array of AdhocAttribute, name-value pair attributes that can be used to specify additional inputs to the Web service.

BulkPresenceResponse

Represents information returned from the getBulkPresence service.

Table 3–48 Parameters of BulkPresenceResponse

Parameter	Description
statusCode	Integer status code for the Web service call.

Table 3–48 (Cont.) Parameters of BulkPresenceResponse

Parameter	Description
statusMessage	Descriptive string for the status of the Web service call.
statusDetails	String containing additional information regarding the status of the Web service call.
presentityStatusList	Array of detailed presence information for the target presentities, encapsulated in an array of PresentityStatus objects.
adhocAttributes	Array of AdhocAttribute, name-value pair attributes that can be used to specify additional output elements from the Web service.

The following table describes the status codes of the getBulkPresence service:

Table 3–49 Status Codes of getBulkPresence Service

Status	Status Code	Description
NO_ERROR	0	Web service call completed successfully
SERVICE_INPUT_ERROR	1010	Invalid input was provided to the Web service
REQUEST_SIZE_EXCEEDED	1011	The request exceeded the maximum number of allowed request elements. No data is returned. It is the responsibility of clients invoking this service to limit the size of their requests.
ADDITIONAL_DATA_AVAILABLE	1012	The request was only partially processed since the response reached the maximum allowed response size. Elements of the response are processed in the same order as they are found in the request until the maximum response size is exceeded. Any subsequent request elements will be ignored and the corresponding response elements will not included as part of the overall response. It is therefore the responsibility of clients invoking this service to provide the logic to handle any such dropped request elements.
NOT_AUTHORIZED	1032	The entity is not authorized to make the request

Reporting Services

Provides a mechanism for clients to generate reports based on finished conferences. It allows clients to retrieve conference attendees, lists of finished conferences based on their date and conference ID, as well as more specific conference details.

Reporting Services exposes the following Web services:

- Report Attendee List
- Report Finished Conference Details
- Report Finished Conferences By Date
- Report Finished Conferences By ID

Service URL

http://<*hostname:portnum*>/imtapp/ws/ReportingService

Service Definition

http://<*hostname:portnum*>/imtapp/ws/ReportingService?WSDL

Report Attendee List

Used by integrating applications to obtain attendee-related information for a particular conference. The SOAP service is an RPC-style service that returns a list of attendees.

This service returns an object of type AttendeeListResponse.

Request Parameters of reportAttendeeList

The following are the request parameters for the reportAttendeeList operation:

Table 3–50 Parameters of reportAttendeeList

Parameter	Description
param0	AttendeeListRequest object

AttendeeListRequest

Represents information for a particular conference whose attendees are to be retrieved.

Table 3–51 Parameters of AttendeeListRequest

Parameter	Description
entityID	Character string representing the site ID or user ID of the entity making the Web service call.
entityToken	Character string representing the authentication token for this entity. The token can be obtained from Authentication Services.
conferenceID	String containing the ID of the conference whose attendees are to be retrieved
adhocAttributes	Optional array of AdhocAttribute objects containing name-value pair attributes that can be used to specify additional inputs to the Web service..

AttendeeListResponse

Represents attendee-related information of a particular conference.

Table 3–52 Parameters of AttendeeListResponse

Parameter	Description
statusCode	Integer status code for the Web service call.
statusMessage	Descriptive string for the status of the Web service call.
statusDetails	String containing additional information regarding the status of the Web service call.
attendees	Array of Attendee for this conference.
adhocAttributes	Optional array of AdhocAttribute objects containing name-value pair attributes that can be used to specify additional inputs to the Web service..

The following table describes the status codes of the reportAttendeeList service:

Table 3–53 Status Codes of reportAttendeeList Service

Status	Status Code	Description
NO_ERROR	0	Web service call completed successfully
INVALID_SITE_ID	4	The site ID provided was invalid
ERROR_RETRIEVING_ATTENDEE_LIST	377	The attendee list could not be retrieved for this conference
ERROR_RETRIEVING_CONFERENCE_DETAILS	379	The conference details could not be retrieved for this conference.
NOT_AUTHORIZED	1032	The entity is not authorized to make the request
UNEXPECTED_ERROR	1090	An unexpected error occurred and the operation cannot be completed successfully

Attendee

Represents an attendee of a conference.

Table 3–54 Parameters of Attendee

Parameter	Description
name	String containing the attendee's name.
emailAddress	String containing the attendee's email address
timeEntered	Long value representing time attendee entered conference
timeExited	Long value representing time attendee exited conference
userID	String containing the attendee's user ID
guest	Boolean flag indicating whether the attendee was a guest of the conference
adhocAttributes	Optional array of AdhocAttribute containing name-value pair attributes that can be used to provide additional type-specific data.

Report Finished Conference Details

Used by integrating applications to obtain detailed information for a particular finished conference. The SOAP service is an RPC-style service that returns the conference details for the requested conference. It obtains detailed information for a finished conference. In particular, it returns the requested conference's details including information such as title, host name, start and end times, duration, number of attendees, and status details, and can optionally include attendee and playback URL information for the conference.

This service returns an object of type FinishedConferenceDetailsResponse.

Request Parameters of reportFinishedConferenceDetails

The following are the request parameters for the reportFinishedConferenceDetails operation:

Table 3–55 Parameters of reportFinishedConferenceDetails

Parameter	Description
param0	FinishedConferenceDetailsRequest object

FinishedConferenceDetailsRequest

Represents information required by the Report Finished Conference Details service.

Table 3–56 Parameters of FinishedConferenceDetailsRequest

Parameter	Description
entityID	Character string representing the site ID or user ID of the entity making the Web service call.
entityToken	Character string representing the authentication token for this entity. The token can be obtained from Authentication Services.
conferenceID	String containing the ID of the finished conference whose details are to be retrieved.
includeAttendees	Optional boolean flag indicating whether the conference attendees should be included with the conference details. If set to null, the conference attendees will not be included with the conference details.
includePlaybackURLDetails	Optional boolean indicating whether the conference playback URL details should be included with the conference details. If set to null, the conference's playback URL details will not be included with the conference details.
adhocAttributes	Optional array of AdhocAttribute, name-value pair attributes that can be used to specify additional inputs to the Web service.

FinishedConferenceDetailsResponse

Represents information returned by the returnedFinishedConferenceDetails service.

Table 3–57 Parameters of FinishedConferenceDetailsResponse

Parameter	Description
statusCode	Integer status code for the Web service call.
statusMessage	Descriptive string for the status of the Web service call.

Table 3–57 (Cont.) Parameters of FinishedConferenceDetailsResponse

Parameter	Description
statusDetails	String containing additional information regarding the status of the Web service call.
conferenceDetails	The finished conference details; object of type FinishedConferenceDetails.
adhocAttributes	Array of AdhocAttribute containing name-value pair attributes that can be used to specify additional output elements from the Web service.

The following table describes the status codes of the reportFinishedConferenceDetails service:

Table 3–58 Status Codes of reportFinishedConferenceDetails Service

Status	Status Code	Description
NO_ERROR	0	Web service call completed successfully
MEETING_ID_IS_NULL	2	The conference ID provided was set to null
INVALID_MEETING_ID	3	The conference ID provided was invalid
INVALID_SITE_ID	4	The site ID provided was invalid
ERROR_RETRIEVING_ATTENDEE_LIST	377	The attendee list could not be retrieved for this conference.
ERROR_RETRIEVING_CONFERENCE_ DETAILS	379	The conference details could not be retrieved for this conference
NOT_AUTHORIZED	1032	The entity is not authorized to make the request
UNEXPECTED_ERROR	1090	An unexpected error occurred and the operation cannot be completed successfully

FinishedConferenceDetails

Represents a finished conference.

Table 3–59 Parameters of FinishedConferenceDetails

Parameter	Description
conferenceID	String containing the conference ID
title	String containing the conference title
hostName	String containing the name of the conference host
startTime	Date object for the time at which the conference started
endTime	Date object for the time at which the conference ended
duration	Integer equal to the duration of conference in minutes
numAttendees	Integer containing the number of attendees
rating	A string rating for the conference

Table 3–59 (Cont.) Parameters of FinishedConferenceDetails

Parameter	Description
comments	String containing comments relating to the conference
statusDetails	String containing status details for the conference. The following are the possible options: ■ WFH: Waiting for host ■ IP: In progress ■ ABANDONED: Meeting scheduled end time has passed and meeting has been already created ■ GRACEFUL: Meeting ended gracefully ■ ABORTED: Meeting was aborted. ■ FAILED: Meeting failed. ■ FORCED: Meeting was forcefully stopped
attendees	Array of Attendee objects with attendee-related information for the conference
playbackURLDetails	PlaybackURLDetails object containing playback URL details for the conference
adhocAttributes	Optional array of AdhocAttribute containing name-value pair attributes that can be used to provide additional type-specific data.

Report Finished Conferences By Date

Used by integrating applications to retrieve conference details for a number of conferences filtered by date. The SOAP service is an RPC-style service that returns a list of conference details for the matching conferences. It obtains conference details for all conferences that occurred between the specified start and end dates. In particular, it returns a list of conference details including information such as title, host name, start and end times, duration, number of attendees, and status details, and can optionally include attendee and playback URL information for each of the matching conferences.

This service returns an object of type BulkFinishedConferenceDetailsResponse.

Request Parameters of reportFinishedConferencesByDate

The following are the request parameters for the reportFinishedConferencesByDate operation:

Table 3–60 Parameters of reportFinishedConferencesByDate

Parameter	Description
param0	FinishedConferencesByIDRequest object

FinishedConferencesByDateRequest

Represents information required by the Report Finished Conferences By Date service.

Table 3–61 Parameters of FinishedConferencesByDateRequest

Parameter	Description
entityID	Character string representing the site ID or user ID of the entity making the Web service call.
entityToken	Character string representing the authentication token for this entity. The token can be obtained from Authentication Services.
fromEndDate	The lower-bound end date for which matching conferences should be retrieved. If set to null, the standard base time of January 1, 1970, 00:00:00 GMT will be used by default.
toEndDate	The upper-bound end date for which matching conferences should be retrieved. If set to null, the current date at the server will be used by default.
includeAttendees	Optional boolean flag indicating whether the conference attendees should be included with the conference details. If set to null, the conference attendees will not be included with the conference details.
includePlaybackURLDetails	Optional boolean indicating whether the conference playback URL details should be included with the conference details. If set to null, the conference's Playback URL details will not be included with the conference details.
adhocAttributes	Optional array of AdhocAttribute containing name-value pair attributes that can be used to specify additional inputs to the Web service.

BulkFinishedConferenceDetailsResponse

Represents details of finished conferences.

Table 3–62 Parameters of BulkFinishedConferenceDetailsResponse

Parameter	Description
statusCode	Integer status code for the Web service call.
statusMessage	Descriptive string for the status of the Web service call.
statusDetails	String containing additional information regarding the status of the Web service call.
conferenceDetailsList	Array of FinishedConferenceDetails, finished conference details matching the request.
adhocAttributes	Array of AdhocAttribute containing name-value pair attributes that can be used to specify additional output elements from the Web service.

The following are the status codes of the bulkFinishedConferencesByDate service:

Table 3–63 Status Codes of bulkFinishedConferencesByDate Service

Status	Status Code	Description
NO_ERROR	0	Web service call completed successfully
INVALID_SITE_ID	4	The site ID provided was invalid
ERROR_RETRIEVING_CONFERENCE_LIST	376	The list of conferences could not be retrieved for this request
ERROR_RETRIEVING_ATTENDEE_LIST	377	The attendee list could not be retrieved for this conference.
ERROR_RETRIEVING_CONFERENCE_ DETAILS	379	The conference details could not be retrieved for this conference
NOT_AUTHORIZED	1032	The entity is not authorized to make the request
UNEXPECTED_ERROR	1090	An unexpected error occurred and the operation cannot be completed successfully

Report Finished Conferences By ID

Used by integrating applications wishing to retrieve conference details for a batch of conferences whose conference IDs are known. The SOAP service is an RPC-style service that returns a list of conference details for the matching conferences. It obtains conference details for those conferences specified in the list of conference IDs. In particular, it returns a list of conference details including information such as title, host name, start and end times, duration, number of attendees, and status details, and can optionally include attendee and playback URL information for each of the matching conferences.

This service returns an object of type BulkFinishedConferenceDetailsResponse.

Request Parameters of reportFinishedConferencesByID

The following are the request parameters for the reportFinishedConferencesByID operation:

Table 3–64 Parameters of reportFinishedConferencesByID

Parameter	Description
param0	FinishedConferencesByIDRequest object

FinishedConferencesByIDRequest

Represents information required by the Report Finished Conferences By ID service.

Table 3–65 Parameters of FinishedConferencesByIDRequest

Parameter	Description
entityID	Character string representing the site ID or user ID of the entity making the Web service call.
entityToken	Character string representing the authentication token for this entity. The token can be obtained from Authentication Services.
conferenceIDs	String array containing the list of conference IDs for those conferences whose details should be retrieved.
includeAttendees	Optional boolean flag indicating whether the conference attendees should be included with the conference details. If set to null, the conference attendees will not be included with the conference details.
includePlaybackURLDetails	Optional boolean indicating whether the conference playback URL details should be included with the conference details. If set to null, the conference's Playback URL details will not be included with the conference details.
adhocAttributes	Optional array of AdhocAttribute containing name-value pair attributes that can be used to specify additional inputs to the Web service.

The following are the status codes of the reportFinishedConferencesByID service:

Table 3–66 Status Codes of reportFinishedConferencesByID Service

Status	Status Code	Description
NO_ERROR	0	Web service call completed successfully
INVALID_SITE_ID	4	The site ID provided was invalid
ERROR_RETRIEVING_ATTENDEE_LIST	377	The attendee list could not be retrieved for this conference.
ERROR_RETRIEVING_CONFERENCE_ DETAILS	379	The conference details could not be retrieved for this conference
NOT_AUTHORIZED	1032	The entity is not authorized to make the request
UNEXPECTED_ERROR	1090	An unexpected error occurred and the operation cannot be completed successfully

4 Presence and Live Help Integration

Business can extend their application with Oracle Real-Time Collaboration Integration Services for enhanced interactions with external customers or partners. In this way, businesses can equip employees with the same Oracle Real-Time Collaboration tools in external as well as internal collaboration flows. A typical scenario is a live help scenario in which customer support Web pages indicate the availability of support personnel, with options to allow the customer to start an instant messaging session with the support personnel.

This chapter describes steps you would typically perform to integrate presence and live help into your applications.

Steps in Enabling Live Help Flow

The live help flow typically involves allowing guest users, who are most likely known only to an integrating site and not to the Oracle Real-Time Collaboration base installation, to participate in Oracle Real-Time Collaboration features like instant messaging. This flow involves the following high-level steps:

Step 1 Provision Site

Communication with Oracle Real-Time Collaboration Integration Services involves the concept of a logical entity called a *site*. Provision the site for the presence and guest user chat policies. The presence policy allows a site to include presence information in its Web pages through the Presence URL service. The guest user chat policy permits a site to request a chat session for a guest user.

Step 2 Create Support Groups

Create a live help support group for the site that will provide live help support. Register your support personnel with this group. This is the group to which live help messages will be broadcasted when a guest user requests live help.

Step 3 Enable Presence in Your Web Pages

Providing presence information in customer interfaces is a powerful way of letting users know if support personnel are available to service a customer's request. An indicator reflecting whether support personnel are online and available, and possibly indicators about predicted hold times or queue lengths, can inform the customer appropriately and set the right expectations of service.

Presence indicators also provide natural invocation hooks for a customer to invoke the next step in the flow: involving actual chat with support personnel.

Step 4 Integrate Live Help into Your Application

This involves the user clicking on a presence indicator (or an equivalent link) to invoke the appropriate Oracle Messenger client to start a chat session.

Provision Site

An administrator can provision a site through the Oracle Real-Time Collaboration Web application interface. To enable live help, provision the site for the following two policies:

- PRESENCE
- GUESTCHAT

The PRESENCE policy allows a site to include presence information in its Web pages through the Presence URL Service. The GUESTCHAT policy permits a site to request a chat session for a guest user.

A provisioned site is provided with a set of site credentials: a site ID and a site authentication token. A site authenticates itself by providing its credentials to Oracle Real-Time Collaboration Authentication services.

Create Support Groups

Users with administrative privileges can add support groups and register support personnel to these groups. An administrator would use the command `rtcctl addGroup` to create a group for this purpose. For more information about the `rtcctl` command, see Chapter 4, "rtcctl Command-Line Utility for Oracle Real-Time Collaboration" in *Oracle Real-Time Collaboration Administrator's Guide*.

Assume within the context of this chapter that the name of the support group is otn.support@sites.oracle.com.

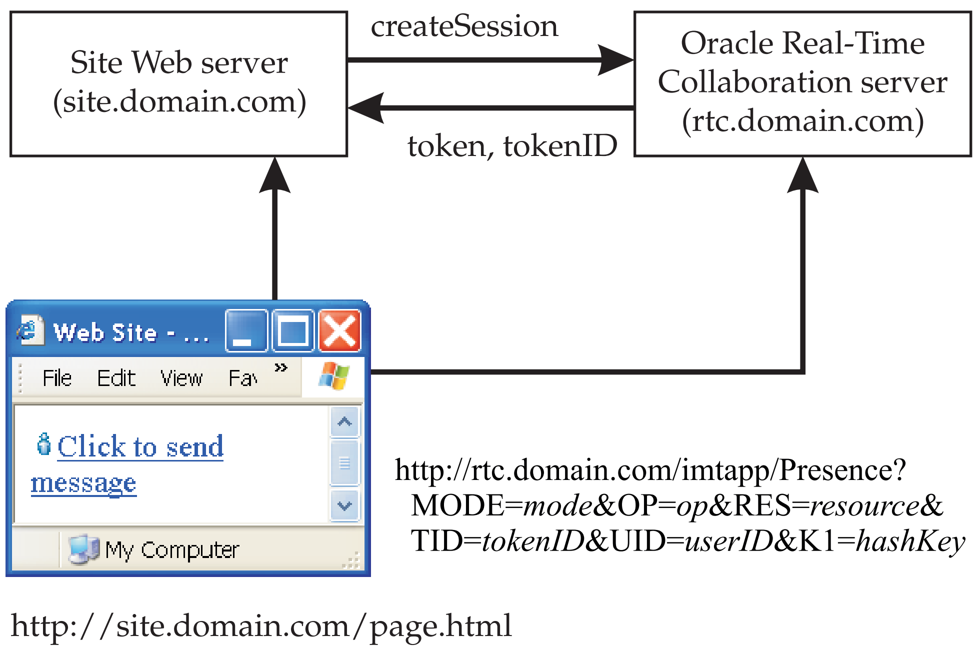
Enable Presence in Your Web Pages

The presence integration scenario uses the Presence URL service. This service provides a registered user with a unique URL that displays user presence information securely in an application. Presence information can be obtained in the form of an image or a status string using the JavaScript variant of Presence URL Services.

The logic involved in creating the URL lies in generating a unique, asymmetric hash key for each user. This key is generated from valid session information that is common to all users that belong to a given site along with user specific information.

The following diagram illustrates the flow of enabling presence:

Figure 4–1 Enabling Presence Flow



Prerequisites

Before proceeding further, make sure you have the following system and account information:

- Access to an Oracle Real-Time Collaboration deployment
- Valid site ID and authentication token for the Web site you would like to include presence information
- Site ID provisioned for the PRESENCE policy

Consuming the Presence URL Service

The Presence URL Service allows integrating sites to query and provide presence information directly in their web pages. This allows for requests to come directly from the user's browser, eliminating the need for server-side processing of presence information by the site. (If the sites does need to process presence information on the server, the Presence SOAP Service may be consumed.) In addition, client-side processing is possible, using the JavaScript version of the service.

Presence Service URL Parameters

The following is an example presence URL:

```
https://rtc.domain.com/imtapp/Presence?
  TID=11704&R_UID=OTN.SUPPORT@GROUPS.DOMAIN.COM&
  R_MODE=IMG&K1=659314C60136CF096DBA01B969992AB9F
```

The URL uses the following parameters:

- **R_TID** (Required): Token ID for a token authorized for the Presence Service. This can be obtained using Authentication Services, by setting "PRESENCE" in the policy option.

- **R_UID** (Required): User ID (or JID) of the target presentity.
- **R_K1** (Required): A SHA1 hash key used to secure the request.
- **R_MODE** (Optional): Optional parameter to specific Image ("IMG") or JavaScript ("JS") mode. Default is "IMG"
- **R_OP** (Optional): Optional parameter specific to the request mode. For example, this parameter can be used to request alternate images when using the "IMG" mode, by setting its value between 1 and 5.
- **R_RES** (Optional): Optional parameter to explicitly retrieve presence of a specific resource.

Presence Integration Steps

To integrate presence in a Web site, follow these steps:

1. Obtain Authentication Token for PRESENCE Policy
2. Compute Presence Hash Key (K1) and URL
3. Embed Presence URLs in Web Page

Obtain Authentication Token for PRESENCE Policy

In order to generate the presence URL, the integrating site needs to first obtain an authentication token from Oracle Real-Time Collaboration Authentication Services. This is a two-step process:

Step 1 Create Session

The integrating site calls the Create Session operation to create an Oracle Real-Time Collaboration Authentication Services session. The inputs to this operation are siteID, siteauthkey and entityType (SITE). The output of this operation is a TokenObject (sessionToken in the following code extract):

```
AuthenticationServiceEIProxy asProxy = new AuthenticationServiceEIProxy();
asProxy._setSoapURL("http://rtc.domain.com/imtapp/AuthenticationService");
try
{
  oracle_imt_service_data_TokenObject tObj =
    asProxy.createSession("12345","MSJSJASDJJASDJASDASD=","SITE");
  String sessionToken = tObj.getToken();
}
catch (Exception e)
{
  // Process Error
}
```

Step 2 Request PRESENCE Token

Since the site wants to display the presence of the live help support group, it needs to then obtain a token for the PRESENCE policy.

A site may call the Request Token operation to request an Oracle Real-Time Collaboration Authentication Services token for a specific purpose. The inputs to this operation are the siteID, sessionToken, TokenProperties and PolicyObject. The following code extract requests an Authentication Services token for the PRESENCE policy:

```
try
{
  oracle_imt_service_data_TokenProperties tProps =
    new oracle_imt_service_data_TokenProperties();
  tProps.setTokenPolicy("PRESENCE");
  oracle_imt_service_data_TokenObject tObj =
    asProxy.requestToken("12345", sessionToken, tProps, null);
  String presenceToken = tObj.getToken();
  String tokenID;
  oracle_imt_service_data_AdhocAttribute[] adAttrs = tObj.getTokenAttributes();
  for (int i = 0; i < adAttrs.length; i++)
  {
    if (adAttrs[i].getName().equalsIgnoreCase("TOKENID"))
      tokenID = adAttrs[i].getValue();
  }
}
catch (Exception e)
{
  // Process Exception
}
```

Compute Presence Hash Key (K1) and URL

The value of the K1 parameter is computed using the SHA1 Digest algorithm on a custom message composed using the URL parameters. This is needed since the presence request would be coming from the end-user browser and not a site. Hence, it is necessary to ensure that only presence requests authorized by a site are served. The input message to the SHA1 hash algorithm is a message obtained from the URL parameters using the following logic:

- Arrange the URL parameters should be arranged in alphabetical order
- Ignore parameters that do not have an assigned value
- For parameters with an assigned value (other than K1), add the following string to the message:

  `<param_name><param_value>`

- In front of the above parameter-value string, include the following string:

  `<TOKEN><authentication_token_presence>`

The Secure Hash Algorithm (SHA1) is a standard message digest algorithm, with libraries available in most programming languages. In Java, the java.security.MessageDigest class can be used for this purpose.

Assume for the sake of this example that the presenceToken returned by the requestToken step is `1000641_QAHnRTN3L8E1DVEHL3UFKYuxHcW8ce7p` and the tokenID is `100641`. The presence hash (K1) can be calculated as shown in the following code extract:

```
String tokenString = "<TOKEN>" +
  "<1000641_QAHnRTN3L8E1DVEHL3UFKYuxHcW8ce7p><R_MODE><IMG>" +
  "<R_UID><john.doe@oracle.com>";
String K1 = getJavaDigest(tokenString);
```

The presence URL can then be constructed as follows for the image version:

```
https://rtc.domain.com/imtapp/Presence?
  TID=100641&R_UID=john.doe@oracle.com&
  R_MODE=IMG&K1=B1CA85982C3221F29E0808F6DB91AFB6AB84008D
```

There is also a Javascript version of the presence URL that can also be generated in a similar fashion as the image version. The following is the Javascript version of the presence URL:

```
https://rtc.domain.com/imtapp/Presence?
  TID=100641&R_UID=john.doe@oracle.com&
  R_MODE=JS&K1=DFGJKCA85982DHJKRU28EWKJ3EWK391KKED9QKDJK
```

The following is an implementation of the `getJavaDigest` method:

```
private static String getJavaDigest(String inputString)
  throws NoSuchAlgorithmException, FileNotFoundException, IOException
{
  MessageDigest md = MessageDigest.getInstance("MD5"); //SHA1 is also supported
  byte[] dataBytes = new byte[2048];
  dataBytes = inputString.getBytes();
  int nread = dataBytes.length;
  md.update(dataBytes, 0, nread);
  byte[] mdbytes = md.digest();
  String s = byteArray2Hex(mdbytes);
  return s;
}

private static char[] hexChars = {
  '0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
  'A', 'B', 'C', 'D', 'E', 'F'};

public static String byteArray2Hex(byte[] ba)
{
  StringBuffer sb = new StringBuffer();
  for (int i = 0; i < ba.length; i++) {
    int hbits = (ba[i] & 0x000000f0) >> 4;
    int lbits = ba[i] & 0x0000000f;
    sb.append("" + hexChars[hbits] + hexChars[lbits] + "");
  }
  return sb.toString();
}
```

Embed Presence URLs in Web Page

Directly embed the presence URL into HTML pages by using the SRC option in an IMG element. The following HTML extract illustrates this (line breaks inserted for clarity):

```
<img src="http://rtc.domain.com/imtapp/Presence?
  TID=100641&R_UID=OTN.SUPPORT@GROUPS.ORACLE.COM&
  R_MODE=IMG&K1=659314C60136CF096DBA01B2432395969992AB9F">
```

Use Embeddable Presence to publish your presence (availability) to others. Navigate to **Preferences > Embeddable Presence** after logging in to the Oracle Real-Time Collaboration Web application for more details.

In order for an action to be performed when a user clicks the image, enclose the IMG tag in an HTML address element as follows:

```
<a href="customaction.jsp"><img src="imageurl"></a>
```

To enable custom actions based on presence status, use the JavaScript version of the Presence URL. The JavaScript version returns JavaScript script that assigns values to standard variables (status, show). The following code extract illustrates this (line breaks inserted in URLs for clarity):

```
<SCRIPT
  LANGUAGE="JavaScript"
  src="http://rtc.domain.com/imtapp/Presence?
    TID=100641&R_UID=OTN.SUPPORT@GROUPS.ORACLE.COM&
    R_MODE=JS&K1=659314C60136CF096DBA01B2432395969992AB9F">
</SCRIPT>
<SCRIPT LANGUAGE="JavaScript">
  gotoURL = "default.jsp"
  if (show == "AVAILABLE") {
    gotoURL = "chat.jsp"
  } else if (show == "OFFLINE") {
    gotoURL = "email.jsp"
  }
  document.write("<A href=\'")
  document.write(gotoURL)
  document.write("\'>Contact Me (I\'m" + status + ")</A>")
</SCRIPT>
```

The first <SCRIPT> segment sets the values of the "status" and "show" variables based on the presence status. The second <SCRIPT> segment consumes these values to enable the appropriate actions. In this way, a site can route its users to the appropriate destinations based on the status obtained by Presence Services. One of the possible actions is an invocation of a guest user chat session.

Integrate Live Help into Your Application

The live help scenario uses Oracle Real-Time Collaboration Authentication Services: John Doe purchases a product from Vision Corp. He runs into some issues while using the product and decides to visit the company Web site to see if he can find some information there. He notices that there is a "Live Help Available" button on the Web site. He clicks the button. Immediately, a chat window opens and a company service representative starts debugging John's issue. The problem is resolved in a matter of minutes and John Doe is extremely satisfied with the overall support experience he has had with Vision Corp.

This section describes how to implement this live help scenario in a Web application. It involves creating a URL for the "Live Help Available" button. To create this URL, you generate a unique, asymmetric hash key for a particular user. This key is generated using valid session information that is common to all users that belong to a given site along with user-specific information.

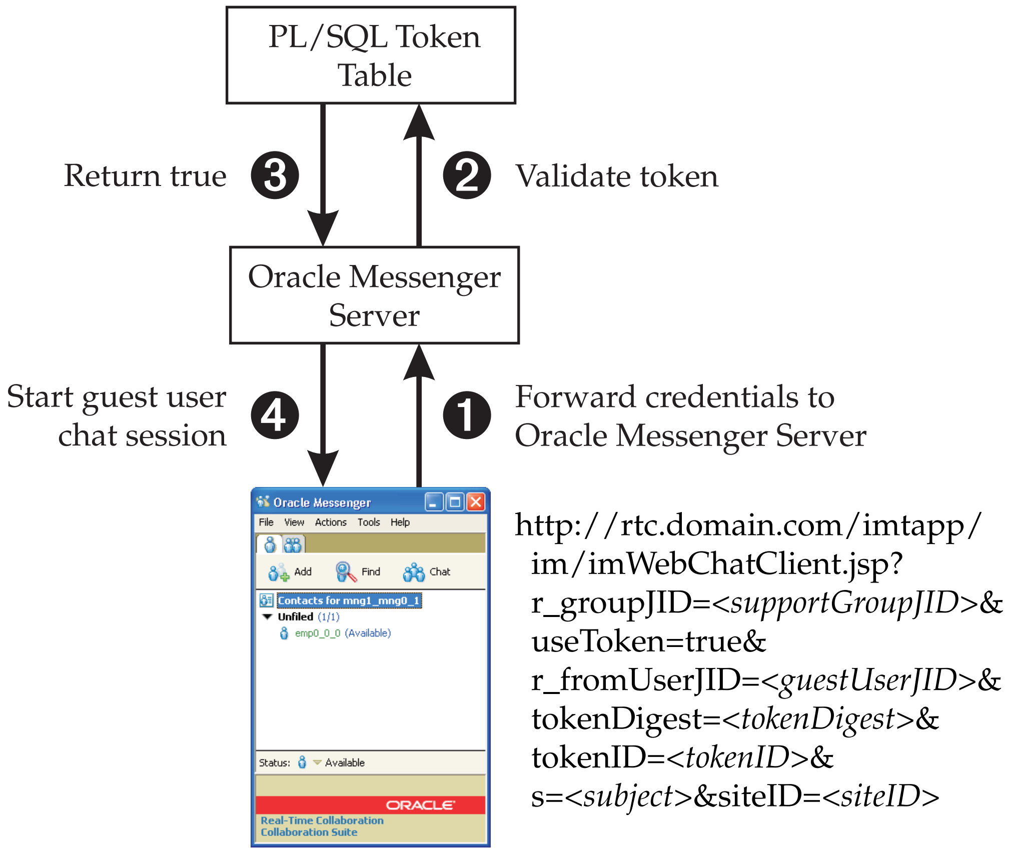
This hash key, along with other parameters, define a context for a request for a guest user chat. These context parameters are submitted, through an HTTP POST operation, to a custom Java Servlet page that calls imRequestorMsg.jsp, the interface file for the Web chat client of Oracle Real-Time Collaboration. This page calls a Java Applet client that establishes an XMPP (Extensible Messaging and Presence Protocol) connection with the Oracle Messenger server.

The Oracle Messenger server receives the context parameters through the XMPP connection established by the Java Applet client. The Oracle Messenger server validates the guest user chat request by recomputing the hash key by using the context parameters it has received. If the recomputed hash key matches the one submitted in the context parameters, the server will allow the guest user chat request.

The server creates a restricted temporary user for the session and allows a guest user chat with the company service representative. The life of the temporary user may extend beyond the chat session depending on Oracle Real-Time Collaboration configuration parameters, which would allow the message history be retrieved for the same user. Therefore, it is important that the live help site does not reuse userJID (a unique identifier that identifies Oracle Real-Time Collaboration users) values between different users.

The following diagram illustrates the flow of live help:

Figure 4–2 Live Help Flow



http://rtc.domain.com/imtapp/im/imWebChatClient.jsp?r_groupJID=<*supportGroupJID*>&useToken=true&r_fromUserJID=<*guestUserJID*>&tokenDigest=<*tokenDigest*>&tokenID=<*tokenID*>&s=<*subject*>&siteID=<*siteID*>

Prerequisites

Before proceeding further, make sure you have access to the following:

- An Oracle Real-Time Collaboration deployment
- A valid site ID and authentication token for the Web site you would like to include presence information
- The site ID provisioned for the GUESTCHAT policy
- Information about groups for group presence services. These groups respond to guest user chat requests. An administrator would use the command `rtcctl addGroup` to create a group for this purpose. For more information about the rtcctl command, see Chapter 4, "rtcctl Command-Line Utility for Oracle Real-Time Collaboration" in *Oracle Real-Time Collaboration Administrator's Guide*.

Live Help Integration Steps

As described in the section "Enable Presence in Your Web Pages," custom actions can be enabled based on presence status obtained from the Presence URL service. In order to establish a guest user chat session, complete the following steps:

1. Establish Context for Chat: Establish a guest user chat context from the site
2. Forward Request to Appropriate Clients: Forward the request context to clients such as the HTML chat client
3. Request Processing by Clients: The request context is processed by the clients and forwarded to the Oracle Real-Time Collaboration server in a way that they can be securely validated.
4. Request Processing by Server: If the request context is validated, then the server grants a guest user session.

Establish Context for Chat

After a user clicks the guest user chat invocation widget, the site should intercept the request to construct the necessary context. Establish this context by supplying the following parameters:

- **r_fromUserJID** (for example, guest_user_12345@sites.oracle.com): This parameter is used to identify the end user uniquely to the Oracle Real-Time Collaboration system. A site can leverage this parameter by establishing a unique identifier for each of its users. Thus if the user visits this functionality again, it would be possible to retrieve message history for the user from the Oracle Real-TIme Collaboration system by using the same unique identifier.
- **guestDisplayName** (for example, OTN Guest): This parameter is displayed as the guest user's name in chat sessions.
- **useToken**: This parameter should always be set to true.
- **r_groupJID** (for example, otn.support@groups.oracle.com): This parameter is the target group with which the chat session is to be established.
- **r_toJID**: Use this parameter if you want to establish a chat session with an individual user instead of a target group.
- **s**: Subject (for example, Pricing on OCS)

In addition, the following security parameters are needed:

- **tokenID**: This is the tokenID of a Authentication Token obtained from Authentication Services using a policy of "GUESTCHAT"
- **tokenDigest**: A hash key used for secure validation on the server side. Details about how to compute this parameter are given in the following section.

Obtain Authentication Service Token for GUESTCHAT Policy

In order to implement guest chat, the integrating site needs to obtain an Authentication Token for the GUESTCHAT policy from Authentication Services. This is a two-step process:

Step 1 Create Session

Call the Create Session operation from the integrating site to create an Authentication Services session. The inputs to this operation are the siteID, siteauthkey and entityType (SITE). The output of this operation is a TokenObject (`sessionToken` in the following code extract):

```
AuthenticationServiceEIProxy asProxy = new AuthenticationServiceEIProxy();
asProxy._setSoapURL("http://rtc.domain.com/imtapp/AuthenticationService");
try
{
  oracle_imt_service_data_TokenObject tObj =
  asProxy.createSession("12345","MSJSJASDJJASDJASDASD=","SITE");
  String sessionToken = tObj.getToken();
}
catch (Exception e)
{
  // Process Error
}
```

Step 2 Request GUESTCHAT Token

Since the site wants to display the presence of the live help support group, it needs obtain a token for the GUESTCHAT policy.

Call the Request Token operation from the site to request an Authentication Services token for a specific purpose. The inputs to this operation are the siteID, sessionToken, TokenProperties and PolicyObject. The following code excerpt requests an Authentication Token for the GUESTCHAT policy:

```
try
{
  oracle_imt_service_data_TokenProperties tProps =
    new oracle_imt_service_data_TokenProperties();
  tProps.setTokenPolicy("GUESTCHAT");
  oracle_imt_service_data_TokenObject tObj =
    asProxy.requestToken("12345", sessionToken, tProps, null);
  String presenceToken = tObj.getToken();
  String tokenID;
  oracle_imt_service_data_AdhocAttribute[] adAttrs = tObj.getTokenAttributes();
  for (int i = 0; i < adAttrs.length; i++)
  {
    if (adAttrs[i].getName().equalsIgnoreCase("TOKENID"))
      tokenID = adAttrs[i].getValue();
  }
}
catch (Exception e)
{
  // Process Exception
}
```

Compute the Request Hash Key

The tokenDigest parameter is a wrapper around an MD5 message digest (hashkey) of a message composed from context parameters. The tokenDigest parameter is computed in a manner similar to how the presence URL hash key is generated. Specifically, the following are the steps for creating this hash key are:

- Arrange the context parameters in alphabetical order.
- For parameters with an assigned value (other than tokenDigest), add the following string to the message (including the angle brackets):

  <param_name><param_value>

- After all parameters have been added, add the following string to the beginning of the message (including the angle brackets):

  ```
  <token><token_value>
  ```

Note: Currently only three context parameters are supported (r_fromUserJID, r_toJID in the case of target users, and r_groupJID in the case of target groups). An example of the final concatenated string would look similar to the following:

```
<token><12345_ASDASLjaasljdaSAD><r_fromUserJID>
  <guest_user_12345@sites.oracle.com><r_groupJID><otn.support@groups.oracle.com>
```

The resulting message is then processed by the MD5 algorithm to obtain a hashKey. The MD5 algorithm is a standard message digest algorithm, with libraries available in most programming languages. In Java, the java.security.MessageDigest class can be used for this purpose.

Once the MD5 digest (`md5digestValue`) is generated successfully, the tokenDigest parameter is a hexadecimal conversion of the string that is constructed as follows:

```
12345_<digest><md5digestValue><r_fromUserJID>
  <guest_user_12345@sites.oracle.com><r_groupJID><otn.support@groups.oracle.com>
```

Note: The above string is constructed as follows:

```
tokenID + "_" + "<digest>" + "<" +  md5digestValue + ">" + attributeString
```

The value of `attributeString` is constructed as follows:

```
attributeString = "<r_fromUserJID><guest_user_12345@sites.oracle.com>
  <r_groupJID><otn.support@groups.oracle.com>"
```

The following code extract illustrates these steps. See "Compute Presence Hash Key (K1) and URL" for an implementation of the getJavaDigest and byteArray2Hex methods.

```
String inputString = "<token><" + tObj.getToken() + ">" +
  "<r_fromUserJID><guest_user02@sites.oracle.com>" +
  "<r_groupJID><otn.support1@groups.domain.com>";

String javaDigest = getJavaDigest(inputString);

String guestUserDigestUnencrypted = tokenID +
  "_<digest><" + javaDigest + ">" + "<r_fromUserJID>" +
  "<guest_user02@sites.oracle.com><r_groupJID>" +
  "<otn.support1@groups.domain.com>";

byte[] dataBytes = new byte[1024];
dataBytes = guestUserDigestUnencrypted.getBytes();
String tokenDigest = byteArray2Hex(dataBytes);
```

The following is an example of a guest chat URL (line breaks inserted for clarity):

```
http://rtc.domain.com/imtapp/im/imWebChatClient.jsp?
  r_groupJID=otn.support1@groups.domain.com&
  useToken=true&
  r_fromUserJID=guest_user02@sites.domain.com&
  tokenDigest=313030303634305F3C6469676573743E3C313636363144303035393031303035&
  tokenID=1000640&
  s=Hello&
  siteID=12345
```

In addition to establishing the request context, this interception by the site gives the site to re-confirm the user's credentials. This step is recommended to ensure that the guest user feature is not being invoked from a Web page that has been rendered outside the context in which it was generated. For example, if the original Web page was emailed to a group, the recipients should not automatically be able to invoke the guest user chat feature unless they can be authenticated and authorized by the site.

Forward Request to Appropriate Clients

The parameters described in the previous section define a context for the guest user chat. These parameters are communicated to the appropriate clients though a Web page (`http://rtc.domain.com/imtapp/im/imWebChatClient.jsp`), which is hosted by the Oracle Real-Time Collaboration server. The parameters should be submitted to this page through an HTTP POST operation. Currently, only a Java applet client is supported. Hence, the receiving page will download this Applet with the context parameters. This page will automatically provide additional parameters needed by the client. For example, the IP Address and port number of the Oracle Messenger server may be provided as additional parameters to the client.

Request Processing by Clients

The Java applet client will establish an XMPP connection with the Oracle Messenger server and submit the required user and security context parameters. Since these parameters are submitted by the applet from the user's console, it is necessary to ensure that the request parameters can be secured. This is achieved using the tokenDigest.

Request Processing by Server

The Oracle Messenger server receives the context parameters through the XMPP connection established by the client. It is the duty of the Oracle Messenger Server to ensure the validity of the request and enforce the guest user chat restrictions communicated through the context parameters. Oracle Messenger validates the request by recomputing the request hash key using the parameters it receives (the Oracle Real-Time Collaboration server may do this by itself or delegate it to Authentication Services). If the hash key computed by the server matches the one submitted in the parameters, the server will allow the request.

The Oracle Real-Time Collaboration server will create a restricted temporary user for the session and allow a guest user chat session with the target user. The life of the user may extend beyond the session, based on the Oracle Real-Time Collaboration server configuration parameters. As a result, the message history may be retrieved for subsequent interaction by the same user. Hence, it is important that the site does not reuse the userJID values between different users.

Index