HP Enterprise Collaboration

For the REST API

Software Version: 1.01

Developers Guide

Document Release Date: February 2012

Software Release Date: February 2012

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

UNIX® is a registered trademark of The Open Group.

This product includes an interface of the 'zlib' general purpose compression library, which is Copyright © 1995-2002 Jean-loup Gailly and Mark Adler.

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to:

http://h20230.www2.hp.com/selfsolve/manuals

This site requires that you register for an HP Passport and sign in. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

Or click the **New users - please register** link on the HP Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Support

Visit the HP Software Support Online web site at:

http://www.hp.com/go/hpsoftwaresupport

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

Contents

Chapter 1

Introduction

HP Enterprise Collaboration (**EC**) is a collaboration platform which enhances and facilitates the collaboration that takes place in almost any flow in the IT organization.

It does this by connecting the structured data which is managed in various applications in the IT workspace, with the unstructured collaboration that supports its creation and maintenance.

This document is the Developers Guide to the REST API.

Chapter 2

Basic Concepts

Make sure you understand the following concepts before reviewing or using the REST API.

Conversation

The conversation is an elementary entity in EC and within it users collaborate.

Subject

A short title that describes the conversation's context. Subject may be modified by any Participant.

Post

A message or comment by a Participant.

Participant

A user that may add posts to a conversation or change its subject.

Participants may add other participants if the conversation's Control was not restricted to Owners.

A Participant may remove himself from the conversation if he is not the Owner, as well as removing Attachments he had added.

Owner

The Participant that created the conversation.

The Owner may change the Visibility, Control and Activation modes of the conversation, as well as delete any attachment, and remove Participants from the conversation except himself.

Viewer

A user that may view a conversation, but is not a Participant.

Viewers are not defined explicitly in EC.

Visibility

Can be modified by the Owner to:

- **Public:** Everyone may view this conversation.
- **Participants:** Only Participants may view this conversation.

Control

The Owner may set this to:

- **Everyone:** Every user may add users to the conversation.
- **Participants:** Only Participants may add users to the conversation.
- **Owners:** Only Owners may add users to the conversation.

Activation

Can be modified by the Owner from not active ("draft mode") to active. If a conversation is not activated, it is visible only to the Owner. In this way, a client can provide the functionality of editing a conversation before it is being published.

Required

A Participant can be marked as required for a conversation by anyone that is allowed to view the conversation. In this case, he will be notified by other communication channels about the conversation.

Attachment

There are 3 types of attachments in Enterprise Collaboration. All are attached to the conversation itself **and not to a certain Post** as is currently common in other collaborations and social systems.

The conversation's Owner or its uploader can remove attachments from the conversation if they are still a Participant. The different types of attachment are:

- **File Attachment:** Describes a file that was attached to the conversation.
- **URL Attachment:** Describes a URL that was attached to the conversation.
- **Context Object Attachment**: Describes a Context Object that was attached to the conversation.

Read

When a user is a Participant, additional data is provided in a section named as **Conversation Participant Info**:

- EC saves an indication of whether the conversation was read by him.
- **It is the responsibility of the client to mark the conversation as read**; the only time EC assumes a conversation is read, is on its initial creation.

Profile

The profile is a basic entity in EC which contains the user's personal and organizational information.

> **Note:** The profile should not be confused with Participant.

Presence

Presence is the aggregated state, which is composed of the user's presence states in Enterprise Collaboration and the IM provider (for example OCS).

Chapter 3

REST API

The API is entirely HTTP-based and attempts to conform to the design principles of Representational State Transfer (REST).

EC supports both JSON & XML formats. We recommend using JSON for better performance. Since XMLs are easier to read, they are also provided for debugging and are used in the examples.

When using JSON data format, the BadgerFish convention should be followed.

Operations

GET operations are used to retrieve representations. A GET operation does not change any state on the server side.

POST operations are used to create resources. A representation of the created resource is returned.

PUT operations are used to update existing resources. Usually, a representation of the updated resource is returned.

DELETE operations are used to remove a resource. No content is returned on a successful operation.

Web Application Description Language (WADL)

EC server provides WADLs so clients may use them for viewing the structure of the representations while developing, as well as auto-generating entities from them.

Chapter 4

Enterprise Collaboration REST Services

The following APIs are provided by EC and described in depth in:

- "REST Conversations API" (on page 16))
- "REST Profiles API" (on page 35)
- "REST Notifications API" (on page 40)
- "REST Presences API" (on page 53)

REST Conversations API

The API is responsible for the creation, update, deletion and retrieval of conversations and their sub-entities: Posts, Attachments & Participants.

WADL:
http://host:port/diamond/rest/api/V1/conversations?_wadl&_type=xml

REST Profiles API

The API is responsible for retrieving user profiles and updating them.

WADL:
http://host:port/diamond/rest/api/V1/profiles?_wadl&_type=xml

REST Notifications API

The API is responsible for retrieving notifications from EC. Use this API to receive real-time changes in conversations being participated in or followed.

WADL:
http://host:port/diamond/rest/api/V1/notifications?_wadl&_type=xml

A synchronization algorithm should be implemented when retrieving up to date notifications.

REST Presences API

The API is responsible for retrieving EC users aggregated presence status.

WADL:
http://host:port/diamond/rest/api/V1/presences?_wadl&_type=xml

Chapter 5

Security

Authentication

Basic authentication is used to authenticate the current user.

Encoding

Several fields are encoded using an Apache StringEscapeUtils approach. The client, retrieving a conversation, might need to decode those fields using the same approach:

- Conversation subject
- Part of the post body inside <code>…</code> block

Sanitation

Sanitation of user input is performed by Enterprise Collaboration. The following fields are sanitized:

- Post body
- Attachment file, URL and description
- User job title
- Adapters output.

Chapter 6

REST Conversations API

REST Conversations API URI Space

Resource	URI	Supported Methods
Attachment	http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}/attachments/{attachmentId}	DELETE
Attachment Data	http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}/attachments/{attachmentId}/data	GET
Attachments	http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}/attachments	POST
Conversation	http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}	GET, PUT
Conversations	http://{host}:{port}/diamond/rest/api/V1/conversations/	GET, POST
Particpant	http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}/{participantId}	DELETE
Participants	http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}/participants	POST, PUT
Posts	http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}/posts	GET, POST
SearchResult	http://{host}:{port}/diamond/rest/api/V1/conversations/searchResult	GET

Attachment

Description

Attachment resource.

Provides the ability to remove an attachment from a conversation.

URL

http://{host}:{-port}/diamond/rest/api/V1/conversations/{conversationId}/attachments/{attachmentId}

Remark

Only the conversation's Owner and the uploader of the attachment (as long as he is a Participant) may remove the attachment.

DELETE Method

DELETE	
Action	Removes an attachment
Request parameters	attachmentId; the ID of the attachment being deleted
Request body	None
Request example	DELETE http://localhost:8080/rest/api/V1/conversations /7d15bf97-f833-4583-8bbb-c980b30c9703/attachments/12
Response	None
Status codes	204. The request was fulfilled, or Error Message.

Attachment Data

Description

Attachment data resource.

Provides the ability to retrieve the binary data of an attachment.

URL

http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}/attachments/{attachmentId}/data

Remark

This operation is only valid for a file attachment.

GET Method

GET	
Action	Retrieves an attachment
Request parameters	Attachment ID to retrieve
Request body	None
Request example	GET http://localhost:8080/rest/api/V1/conversations /7d15bf97-f833-4583-8bbb-c980b30c9703/attachments/12/data
Response	The requested data in byte form
Status codes	200. The request was fulfilled, or Error Message.

Attachments

Description

Attachments resource.

Provides the ability to add attachments to a conversation.

Note: Attachments can be obtained by requesting a conversation.

URL

http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}/attachments

Remark

The response includes the server's current sync number, which the client uses in further notification polling.

There are 3 types of attachment in EC which can be created by providing any one of the following objects:

fileParams: For adding a file to the conversation. Should not exceed 50MB.

urlParams: For adding a URL to the conversation.

contextObjectIdParams: For adding a context object attachment, when the context object id is already known in advance.

contextObjectParams: For adding a contextObject to EC, and then creating an attachment in the conversation.

POST Method

POST	
Action	Creates a new attachment. See "Limitations" (on page 76) for more information.
Request parameters	None
Request body	attachmentsParams
Request example	POST http://localhost:8080/rest/api/V1/conversations/{conversationId}/attachments
Response	attachmentsResponse
Status codes	200. The request was fulfilled and the response includes the created attachments and a sync number, or Error Message.

Conversation

Description

Conversation resource

Provides the ability to retrieve a specific conversation and update it.

URL

http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}

Remark

The response includes the server's current sync number, which the client uses in further notification polling.

GET Method

GET	
Action	Retrieves a conversation.
Request parameters	See below
Request body	None
Request example	GET http://localhost:8080/rest/api/V1/conversations/7d15bf97-f833-4583-8bbb-c980b30c9703?includeattachments=false
Response	conversationResponse
Status codes	200. The request was fulfilled and the response includes the requested conversation and a sync number, or Error_Message.

GET Parameters

Parameter	Type	Description	Mandatory?	Default
Includeattachments	Boolean	Retrieves the requested conversation with or without attachments.	No	False

PUT Method

PUT	
Action	Updates the conversation's settings.
Request parameters	None
Request body	See below
Request example	PUT http://localhost:8080/rest/api/V1/conversation/7d15bf97-f833-4583-8bbb-c980b30c9703
Response	conversationResponse
Status codes	200. The request was fulfilled and the response includes the updated conversation and a sync number, or Error Message.

PUT Body

conversationUpdateParams element	Type	Description	Default
<conversationUpdateParams>			
<subject/>	String	A new subject for this conversation.	-
<control/>	Enum	A new control mode.	-
<visibility/>	Enum	A new visibility.	-
<isActive/>	Boolean	Activate the conversation.	-
<isRead/>	Boolean	Whether this conversation was read.	-
</conversationParams>			

PUT Body Remarks

- A conversation cannot be deactivated.
- A conversation cannot have **Participants** visibility and **Everybody** control at the same time.
- Only the Owner is allowed to change control, visibility and activation.

Conversation PUT Body Example

```
<ns3:conversationUpdateParams .. ..>
<subject>Some subject</subject>
<control>PARTICIPANTS</control>
<visibility>PUBLIC</visibility>
<isActive>true</isActive>
<isRead>true</isRead>
</ns3:conversationUpdateParams>
```

Conversations

Description

The Conversations resource

This resource provides the ability to create and retrieve conversations.

URL

http://{host}:{port}/diamond/rest/api/V1/conversations/

Remark

The response includes the server's current sync number, which the client uses in further notification polling.

GET Method

GET	
Action	Retrieves conversations
Request parameters	See below
Request body	None
Request example	GET http://localhost:8080/rest/api/V1/conversations?pagesize=50&from=1&isread=false
Response	conversationsResponse
Status codes	200. The request was fulfilled and the response includes the requested conversations and a sync number.

GET Parameters

Parameter	Type	Description	Mandatory?	Default
pagesize	Integer	The maximum number of conversations to return.	No	20
from	Integer	The index of the first conversation returned. The count starts from 1.	No	1
contextobjectid	String	The context object id of an object that must be included in the returned conversations. (Search conversations is by context object.) The returned list is in descending order of the conversations' last edited date. The operation does not support the **isread** and **isrequired** parameters.	No*	-
isread	Boolean	Returns read or unread conversations. The returned list is not ordered. Cannot be combined with the **contextobjectid** parameter.	No*	-
isrequired	Boolean	Returns required or non-required conversations. The returned list is not ordered. Cannot be combined with the **contextobjectid** parameter.	No*	-

*At least one should be provided.

GET Response

This element is referenced from the following: Conversations - GET method.

conversationsResponse element	Type	Description
<conversationsResponse>		
<syncNumber/>	Long	Server current sync number.
<conversations>	Element	The list of requested conversations.
<conversation/>	Element	See "conversation Element" (on page 61) for more information.
</conversations>		
</conversationsResponse>		

POST Method

POST	
Action	Creates a new conversation
Request parameters	None
Request body	conversationParams
Request example	POST http://localhost:8080/rest/api/V1/conversations
Response	conversationResponse
Status codes	200. The request was fulfilled and the response includes the created conversations and a sync number, or Error Message.

POST Body

conversationParams element	Type	Description	Default
<conversationParams>			
<subject/>	String	*The subject of this conversation.	-
<visibility/>	Enum	The visibility of this conversation.	Public
<control/>	Enum	This conversation's control.	Everyone
<participantsIds>		A list of initial participants.	
<participantId/>	String	The id of a participant to be added to the conversation.	-
<participantsIds/>			
<isActive/>	Boolean	Whether this conversation should be created as active or draft.	False
<firstPostParams/>	Element	An initial post. See "postParams Element" (on page 71) for more information.	-
<attachmentsParams/>	Element	Initial attachments. See "attachmentsParams Element" (on page 58) for more information.	-
</conversationParams>			

*At least one should be provided

POST Body Remarks

- A subject or a first post must be provided.
- The creator of the conversation is added implicitly to the conversation. (There is no need to provide this in the list of participants.)
- A conversation cannot have **Private** visibility and **Everybody** control at the same time.
- See "Security" (on page 15) for more information about subject and post body sanitation and encoding.

Conversations POST Body Example

```
<ns2:conversationParams
xmlns:ns2="HP.Collaboration.Diamond.Api.Conversation" .. .. .. >
<subject>Hello World!</subject>
<visibility>PUBLIC</visibility>
<control>PARTICIPANTS</control>
<participantsIds>
<participantId>john1</participantId>
<participantId>joseph</participantId>
<participantId>eyal</participantId>
</participantsIds>
<isActive>false</isActive>
<firstPostParams>
<body>This is the first message</body>
</firstPostParams>
<attachmentsParams>
<attachmentParams xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="ns2:urlParams">
<url>http://www.hp.com</url>
<description>HP’s official web-site</description>
</attachmentParams>
</attachmentsParams>
</ns2:conversationParams>
```

Participant

Description

Participant resource.

Provides the ability to remove a Participant from a conversation.

URL

http://{host}:{-port}/diamond/rest/api/V1/conversations/{conversationId}/participants/{participantId}

Remark

An Owner may remove every Participant except himself.

A Participant may remove any Participant he added.

DELETE Method

DELETE	
Action	Removes a Participant
Request parameters	Participant ID to remove
Request body	None
Request example	DELETE http://localhost:8080/rest/api/V1/conversations/7d15bf97-f833-4583-8bbb-c980b30c9703/participants/john2
Response	None
Status codes	204. The request was fulfilled, or Error Message.

Participants

Description

Participants resource.

Provides the ability to add and update conversation Participants.

URL

http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}/participants

Remark

The response includes the server's current sync number, which the client uses in further notification polling.

POST Method

POST	
Action	Adds new Participants to the conversation. See "Limitations" (on page 76) for more information.
Request parameters	None
Request body	participants
Request example	POST http://localhost:8080/rest/api/V1/conversations/{conversationId}/participants
Response	profiles
Status codes	200. The request was fulfilled, and the response includes the created participants and a sync number, or Error Message.

PUT Method

PUT	
Action	Updates the status of participants in a conversation.
Request parameters	None
Request body	participants
Request example	PUT http://localhost:8080/rest/api/V1/conversations/{conversationId}/participants
Response	None
Status codes	204. The request was fulfilled, with no content to return, or Error Message.

Posts

Description

Posts resource.

This provides the ability to add and retrieve posts to and from a conversation.

URL

http://{host}:{port}/diamond/rest/api/V1/conversations/{conversationId}/posts

GET Method

GET	
Action	Retrieves posts from a specific conversation.
Request parameters	See below
Request body	None
Request example	GET http://localhost:8080/rest/api/V1/conversations/{convesationId}?pagesize=50&from=1&order=desc
Response	postsResponse
Status codes	200. The request was fulfilled and the response includes the requested posts and a sync number, or Error_Message.

GET Parameters

Parameter	Type	Description	Mandatory?	Default
pagesize	Integer	The maximum number of conversations to return. See "Limitations" (on page 76) for more information.	No	50
from	Integer	The index of the first conversation returned. The count starts from 1.	No	1
order	String	**desc** for descending order. **asc** for ascending order. Sort order is based on the posts' date of creation.	No	desc

GET Response

This element is referenced from the following: Posts - GET method.

postsResponse element	Type	Description
<postsResponse>		
<syncNumber/>	Long	Server current sync number.
<posts>	Element	The list of requested posts.
<post/>	Element	See "post Element" (on page 70) for more information.
</posts>		
</postsResponse>		

POST Method

POST	
Action	Creates a new post in a specific conversation. See "Limitations" (on page 76) for more information.
Request parameters	None
Request body	postParams
Request example	POST http://localhost:8080/rest/api/V1/conversations/{conversationId}/posts
Response	postResponse
Status codes	200. The request was fulfilled and the response includes the created post and a sync number, or Error_Message.

POST Body Remarks

- A subject or a first post must be provided.
- The creator of the conversation is added implicitly to the conversation. (There is no need to provide this in the list of participants.)
- A conversation cannot have **Private** visibility and **Everybody** control at the same time.
- See "Security" (on page 15) for more information about subject and post body sanitation and encoding.

POST Body Example

```
<ns2:conversationParams
xmlns:ns2="HP.Collaboration.Diamond.Api.Conversation" .. .. .. >
<subject>Hello World!</subject>
<visibility>PUBLIC</visibility>
<control>PARTICIPANTS</control>
<participantsIds>
<participantId>john1</participantId>
<participantId>joseph</participantId>
<participantId>eyal</participantId>
</participantsIds>
<isActive>false</isActive>
<firstPostParams>
<body>This is the first message</body>
</firstPostParams><attachmentsParams>
<attachmentParams xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="ns2:urlParams">
<url>http://www.hp.com</url>
<description>HP's official website</description>
</attachmentParams>
</attachmentsParams>
</ns2:conversationParams>
```

Search Result

Description

The Search Result resource.

This resource provides the ability to search conversations by free text. The search is performed in Subjects, Posts, Users that participate or participated in conversations, Context Objects & Facets (See "Remark" (on page 32)).

URL

http://{host}:{port}/diamond/rest/api/V1/conversations/searchResult

Remark

The indexing of conversations in Enterprise Collaboration is not done at real time. Therefore, if a user posts some text, it is unlikely that they will be able to find that text in an immediate search. By default, indexing is performed every 10 minutes, but can be configured to a different value by the administrator.

EC's server also supports “Facets” while REST API does not. In this case, the facet object will not be included, but the string phrase found by the search is returned. In this way, the user can see that the conversation contains the string even though it was added by an advanced UI client.

GET Method

GET	
Action	Retrieves search results
Request parameters	"GET Parameters" (on page 32)
Request example	GET http://{host}:{port}/diamond/rest/api/V1/conversations/searchResult?searchString=papaya&limit=10
Response	"conversationSearchResponse Element" (on page 65)
Status codes	200. The request was fulfilled and the response includes the requested searchResult and a sync number.

GET Parameters

Parameter	Type	Description	Mandatory?	Default
searchString	String	A "Google-like" free text search.	Yes	
Limit	Integer	The maximum number of searchResults to return.		1000

SearchResult GET Response Example

(searchString="papaya")

```
<ns2:conversationSearchResponse
xmlns:ns2="HP.Collaboration.Diamond.Api.Conversation"
   xmlns:ns3="HP.Collaboration.Diamond.Api.Profile"
   xmlns:ns4="HP.Collaboration.Diamond.Api.Repository">
<syncNumber>31</syncNumber>
<searchResult>
 <hitResults>
  <hitResult>
   <highlightFields>
    <highlightField>
     <type>POST_BODY</type>
      <fragments>
        <fragment>The <em>papaya</em> (from Carib via Spanish),
papaw,
              or pawpaw is the fruit of the plant Carica
<em>papaya</em>,
        </fragment>
      </fragments>
     </highlightField>
    </highlightFields>
    <conversation>...</conversation>
  </hitResult>
 <hitResult>
<highlightFields>
 <highlightField>
   <type>POST_BODY</type>
    <fragments>
     <fragment>
      Carica <em>papaya</em> was the first fruit tree
       to have its genome deciphered.
      </fragment>
     </fragments>
  </highlightField>
  <highlightField>
    <type>SUBJECT</type>
     <fragments>
      <fragment><em>Papaya</em> Genome</fragment>
     </fragments>
   </highlightField>
  </highlightFields>
   <conversation>...</conversation>
  </hitResult>
 <hitResult>
    <highlightFields>
     <highlightField>
```

```
      <type>SUBJECT</type>
       <fragments>
        <fragment><em>Papaya</em></fragment>
       </fragments>
      </highlightField>
    </highlightFields>
      <conversation>...</conversation>
 </hitResult>
</hitResults>
</searchResult>
</ns2:conversationSearchResponse>
```

Chapter 7

REST Profiles API

REST Profiles API URI Space

Resource	URI	Supported Methods
Profiles	http://{host}:{port}/diamond/rest/api/V1/profiles/{userIds}	GET
Profiles	http://{host}:{port}/diamond/rest/api/V1/profiles	GET
Images	http://{host}:{port}/diamond/rest/api/V1/profiles/my/images	DELETE, GET, PUT

Profiles (Explicit Retrieval)

Description

Profiles resource.

This provides the ability to get user profiles.

URL

http://{host}:{port}/diamond/rest/api/V1/profiles/{userIds}

Remark

{userIds} should be provided with a comma as delimiter.

GET Method

GET	
Action	Get users' profiles according to the given user ids
Request parameters	The IDs of the requested users
Request body	None
Request example	GET http://localhost:8080/rest/api/V1/profiles/user1,user2,user3?isLight={isLight}
Response	profilesResponse
Status codes	200. The request was fulfilled, or Error_Message.

GET Parameters

Parameter	Type	Description	Mandatory?	Default
isLight	Boolean	Whether a detailed or light profile should be returned.	No	True

Profiles (Search)

Description

Profiles resource. This resource provides the ability to search for internal users.

The client should be able to implement a user chooser component based on this resource. The search is performed by relating the strings as prefixes. Results are sorted by first name and last name respectively (according to admin configurations). For a free search (by searchString parameter), the system returns the results sorted by a predefined priority.

URL

http://{host}:{port}/diamond/rest/api/V1/profiles

GET Method

GET	
Action	Searches for profiles according to the given method and text.
Request parameters	None.
Request body	None.
Request example	GET http://{host}:{port}/diamond/rest/api/V1/profiles?searchString=john
Response	profilesResponse Light profiles will be returned.
Status codes	200. The request was fulfilled. 206. The request was fulfilled but not all profiles were returned due to system limit.

GET Parameters

Parameter	Type	Description	Mandatory?	Default
searchString	String	A free search string. The server will try to retrieve users on its own.	No*	None
firstName**	String	A prefix of the first name.	No*	None
lastName**	String	A prefix of the last name.	No*	None
email	String	A prefix of an email address.	No*	None
limit	Integer	The maximum number of results to be returned. See"Limitations" (on page 76) for more information.	No	50

* At least one of these parameters must be provided.

** firstName and lastName can be combined.

Images

Description

Images resource.

This provides the ability to retrieve and update a profile's image.

URL

http://{host}:{port}/diamond/rest/api/V1/profiles/my/images

GET Method

GET	
Action	Get user images according to user id and the image scale
Request parameters	The ID of the user whose image is retrieved
Request body	None
Request example	GET http://localhost:8080/rest/api/V1/profiles/my/images?scale={scale}
Response	Byte array of the requested representation
Status codes	200. The request was fulfilled, or Error_Message.

GET Parameters

Parameter	Type	Description	Mandatory?	Default
scale	Enum	Image scale: • Thumbnail • Small • Normal • Large	No	Thumbnail

Parameter Remarks

The client should use **Thumbnail** whenever possible, since the server will perform fetching optimizations on the image.

Scales are subjected to server administration. By default the scales are:

- Thumbnail: 60 x 60 pixels
- Small: 60 x height pixels
- Normal: 100 x height pixels
- Large: 200 x height pixels

PUT Method

PUT	
Action	Updates a user's image
Request parameters	None
Request body	None
Request example	PUT http://localhost:8080/rest/api/V1/profiles/my/images
Response	None
Status codes	200. The request was fulfilled, or Error_Message.

DELETE Method

PUT	
Action	Removes a user's image
Request parameters	None
Request body	None
Request example	DELETE http://localhost:8080/rest/api/V1/profiles/my/images
Response	None
Status codes	200. The request was fulfilled, or Error_Message.

Chapter 8

REST Notifications API

Notifications fulfill the requirement of EC to respond to the users in real time, meaning no browser refresh is needed to get new events. The API allows EC clients to poll the server and get new notifications events when they are issued.

How to Refer to Notifications Fields

The notification structure includes the same entities that can be retrieved by the Conversation & Profile API (Conversations, Posts, Profiles, etc).

The Complete Case

In order to indicate what the exact change was, all differences are sent.

In this way, a client is able to build a complete picture of the change.

Example: If a post is added to a conversation, the notification includes the added post and the conversation containing it, with its new edited date. Since other fields in the conversation are not changed, they are not sent (except its ID of course).

The Partial Case

In case the server is not willing to send an entire entity as a notification, it sends it with partial information. Only some of the changed fields are provided and the **isPartial** flag is set to true.

Note: If **isPartial** is not provided, you should regard it as false.

Important: Fields that are not provided, cannot be address as unchanged.

For simplicity, it is recommended to fetch the entire entity directly from the Conversation\Profile API if the client needs this entity.

Example: A new conversation is created. The server sends a notification which includes the conversation with its created date, subject and **isPartial** flag turned on. This is an indication that the conversation contains more information (probably at least participants).

The Deletion Case

If an entity is removed, the **isDeleted** flag is set to true.

REST Notifications Mechanism

From the client perspective, the notification mechanism works in the following manner:

The client performs these general activities:

1. Register the client
2. Retrieve the resource
3. Retrieve notifications
4. Synchronize between resource and notification retrieval

Register the Client

- When the client starts up, it performs registration with the server.
- The registration response includes the global notification sync number, which represents the server's current state.
- The client uses this sync number to get further notifications from the server.
- The sync number grows sequentially in a global manner. For example, two sequential sync numbers could correspond to different resources.

> **Note:** The notification sync number could be updated in the next phase (Retrieve the Resource). Therefore, in most cases, the sync number retrieved in the registration phase may be ignored.

Retrieve the Resource

- When the client decides to retrieve a resource, it should call the Conversation REST API\Profile REST API and retrieve conversation and user data.
- The data includes the conversation sync number. The client should use this when polling further notification events about the changes in the corresponding resource.

Retrieve Notifications

- After refreshing the resources, the client is able to poll for notifications for those resources; all resources in the polling request should be in the **refreshed** state.
- The polling is long polling.
- As result of the single resource change, multiple notification events might be generated. (For example adding a new post will generate two events: (a) post added; and (b) conversation subject changed.) Each notification event includes a global sync number.
- When the client retrieves notification events, it receives the sync number with each notification event and remembers it for further work.
- The client handles the notification events and saves every sync number after successful handling. The client polls the server for further notification events, while specifying the sync number from which it would like to retrieve the notifications.

- The client is able to recognize whether a notification event is relevant or outdated. This could be done by comparing the event's sync number to the sync number of the resource, which was remembered before. If the client retrieves a notification event with a sync number less than the sync number it saved when retrieving the resource, it should ignore the notification.
- The notification event could be outdated. For example, because the client is multithreaded and performs a refresh of resources and retrieves notifications in separate threads.
- If the client fails to handle the notification event, it retrieves it again the next time it polls the server, since it uses the sync number of the last successfully handled event. This prevents situations when the notification event is lost.
- If the client receives an HTTP 404 message while polling, it means that the server does not have the notification with this sync number, and the client should refresh the resource.

Synchronize Between Refreshes and Notification Retrieval

- The client performs refreshes and notification retrieval in multiple threads, so the synchronization between refreshes and notification retrieval should be done.
- The sync number, used in the next polling should be the minimum between the sync number brought by the refresh and the sync number brought by the last notification event.
- This could cause some notification event to be retrieved from the server more than once, and the client should ignore these (after comparison), as mentioned above.

Technology

Long polling (Comet) is used to emulate server push.

REST Notifications API URI Space

Resource	URI	Supported Methods
Registrations	http://{host}:{port}/diamond/rest/api/V1/notifications/registrations	GET
Notifications	http://{host}:{port}/diamond/rest/api/V1/notifications	GET

Registrations

Description

Registration resource.

URL

http://{host}:{port}/diamond/rest/api/V1/notifications/registrations

Remark

This resource provides registration of REST clients for notifications.The response includes the server's current sync number, which the client uses in further notification polling.

GET Method

GET	
Action	Registers and synchronizes the client with the current revision of the server.
Request parameters	None.
Request body	None.
Request example	GET http://localhost:8080/diamond/rest/api/V1/notifications/registrations
Response	registrationResponse
Status codes	200. The request was fulfilled and the response includes registration data, or Error_Message.

GET Response

registrationResponse element	Type	Description	Default
<registrationResponse>			
<syncNumber/>	Long	Server current sync number.	-
<clientId/>	String	Client UUID.	-
</registrationResponse>			-

Registrations GET Response Example

```
<ns4:registrationResponse
xmlns:ns2="HP.Collaboration.Diamond.Api.Profile"
xmlns:ns3="HP.Collaboration.Diamond.Api.Conversation"
xmlns:ns4="HP.Collaboration.Diamond.Api.Notification">
<syncNumber>182048</syncNumber>
<clientId>8cf1375e-a7be-4ae2-b117-909ca1ff98b4</clientId>
</ns4:registrationResponse>
```

Notifications

Description

Notifications resource.

URL

http://{host}:{port}/diamond/rest/api/V1/notifications

Remark

Retrieves notifications for the user.

GET Method

GET	
Action	Retrieves notifications from the server.
Request parameters	See below
Request body	None
Request example	GET http://localhost:8080/diamond/rest/api/V1/notifications?startFrom=25 GET http://localhost:8080/ diamond/rest/api/V1/notifications?startFrom=25&conversationIds=1,2,3
Response	notificationResponse
Status codes	200. The request was fulfilled and the response includes registration data. 401. Authentication problem; user or password are wrong.

GET Parameters

Parameter	Type	Description	Default
conversationIds	String, comma separated	Specific conversation IDs to get notifications for the user.	Empty
startFrom	Long	Get notifications starting from this sync number.	-

GET Response

notificationResponse element	Type	Description	Default
<notificationResponse>			
<notifications>	List	List of notifications.	
<notification/>	Element	See "Notification Element" (on page 49) for more information.	
</notifications>			
<syncNumber/>	Long	Server current sync number.	
</notificationResponse>			

Notifications GET Response Example

```
<ns4:notificationResponse
xmlns:ns2="HP.Collaboration.Diamond.Api.Profile"
xmlns:ns3="HP.Collaboration.Diamond.Api.Conversation"
xmlns:ns4="HP.Collaboration.Diamond.Api.Notification">
 <notifications>
  <notification>
   <actionType>CONVERSATION_PARTICIPANTS_REQUIRED_ADD</actionType>
   <syncNumber>183408</syncNumber>
   <clientId>709025fd-c152-4cbe-a762-4b365ace5b26</clientId>
   <initiator>
     <id>joseph.gutin@hp.com</id>
     <displayName>Joseph Gutin</displayName>
     <jobTitle>Diamond server side developer</jobTitle>
     <emails>
       <email>joseph.gutin@hp.com</email>
     </emails>
     <source>INTERNAL</source>
     <thumbnailImageResource>…</thumbnailImageResource>
     <mobileImageResource>…</mobileImageResource>
     <smallImageResource>…</smallImageResource>
     <normalImageResource>…</normalImageResource>
   </initiator>
   <conversations>
     <conversation>
       <id>e3a2b4a7-8d62-4d9b-807a-a2114bffe43f</id>
       <editedDate>1323007965494</editedDate>
       <participantsInfo>
         <participantInfo>
           <isRequired>true</isRequired>
           <lightProfile>
             <id>michal.foster@hp.com</id>
             <emails/>
           </lightProfile>
         </participantInfo>
       </participantsInfo>
     </conversation>
   </conversations>
 </notification>
</notifications>
<syncNumber>183408</syncNumber>
</ns4:notificationResponse>
```

Notification Element

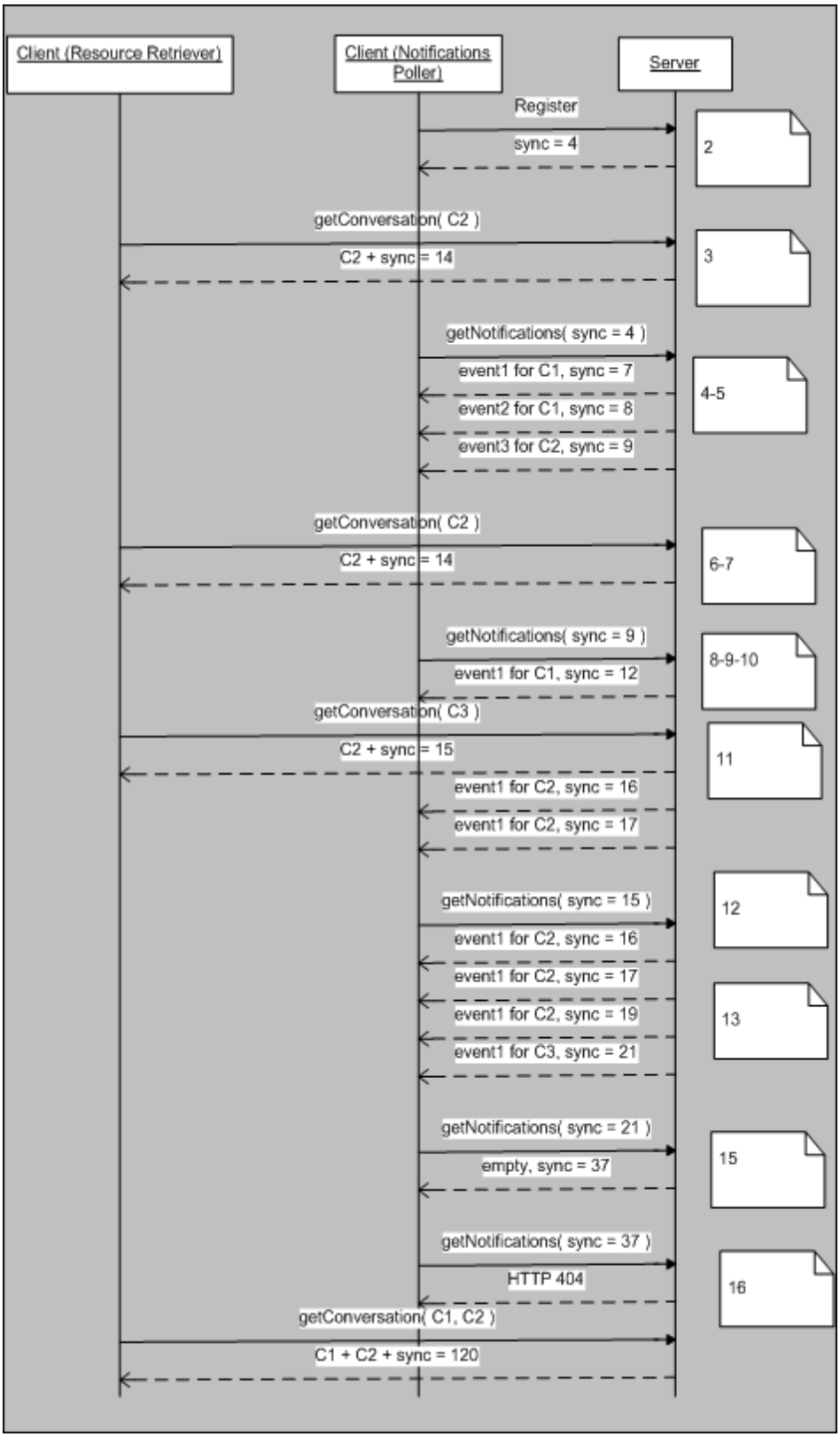
Element	Type	Description
<notification>	Notification element	Notification
<syncNumber/>	Long	Notification sync number
<clientId/>	UUID	Client UUID
<initiator/>	Element	See "initiator Element" (on page 66) for more information.
<actionType/>	Enum	For future use.
<conversations>	List	List of conversations.
<conversation/>	Conversation element	See "conversation Element" (on page 61) for more information.
</conversations>		
<posts>	List	List of posts.
<post/>	Post element	See "post Element" (on page 70) for more information.
</posts>		
<profiles>	List	List of profiles.
<profile/>	Profile element	See "profile Element" (on page 72) for more information.
</profiles>		
</notification>		

Example of Retrieving Notifications for Multiple Conversations

Step	Description	Current Refresh Sync Numbers	Next Polling Sync Number
1	The client starts up		
2	The client registers itself with the server, receives current sync number 4 and saves it as current polling sync.		
3	The user enters conversation C1, the client retrieves conversation C1 using conversation API. The sync number returned by the API is 6. The client saves sync number 6 as sync of C1 and also as current refresh sync. The client calculates next polling sync after refresh as min(4, 6) = 4	C1 (6)	4
4	The client starts polling for user notifications (all related conversations for this user) starting from sync number 4.		
5	The client retrieves notification result from the server. It includes several notification events with syncs 7,8,9, while syncs 7 and 8 are for conversation C1. The client starts handling all event,s and after each successful handling saves sync of the event as both current sync of conversation C1 (7 and 8) and current polling sync. In the end, the client saves sync number 9 as current polling sync. The client uses the last successfully updated sync as next polling sync (9)	C1 (8)	9
6	User adds explicitly conversation C2 (which the user is not a participant in) to the view.		
7	The client retrieves conversation C2 using conversation API. The sync number returned by the API is 14. The client saves sync number 14 as current sync for conversation C2 and as current refresh sync. The client calculates next polling sync after refresh as min(9, 14) = 9	C1 (8) C2 (14)	9
8	The client starts polling for notifications for all user conversations and C2 starting from sync number 9.		
9	The client receives notification result from the server. It includes several notification events: for C1 with sync 12, for C2 with sync 16, and for C2 with sync 17		

Step	Description	Current Refresh Sync Numbers	Next Polling Sync Number
10	The client compares current sync of conversation C1 (9) with just retrieved event for C1 with sync 12, accepts the event with sync 12, and saves 12 as current sync of C1 and current polling sync. The client uses the last successfully update sync as next polling sync (17)	C1 (12) C2 (17)	17
11	The client adds conversation C3 in parallel to the polling in stage 8. The client retrieves C3 with sync 15. The client calculates next polling sync after refresh as min(17, 15) = 15	C1 (12) C2 (17) C3 (15)	15
12	The client starts polling starting with sync 15 for all user conversations and C2.	C1 (12) C2 (17) C3 (15)	15
13	The client receives notification result from the server. It includes several notification events: for C2 with sync 16, for C2 with sync 17, for C2 with sync 19, and for C3 with sync 21. The client ignores syncs 16 and 17 since the last refresh sync of C2 is already 17. The client saves syncs 19 and 21	C1 (12) C2 (19) C3 (21)	21
14	The user exits from conversation C3. The client removes the entry for C3 and stops updating it, even if some notification for C3 is retrieved while polling for all user notifications.		
15	The client polls for notifications, but there are no new events. The client receives current sync number (37) and updates its current polling sync.		37
16	After some long time (for example due to a network problem), the client continues polling starting from sync 37 but there is no notification event with this sync on the server since it is purged. The client receives an HTTP status 404 message and refreshes all conversations it monitors, receiving their updated sync number. The client continues polling using these syncs (min of the refresh sync and current polling sync)		
17	And so on…		

The following image is a diagram showing how notifications are retrieved from multiple conversations.


Client (Resource Retriever)
Client (Notifications Poller)
Server
Register
sync = 4
2
getConversation(C2)
C2 + sync = 14
3
getNotifications(sync = 4)
event1 for C1, sync = 7
event2 for C1, sync = 8
event3 for C2, sync = 9
4-5
getConversation(C2)
C2 + sync = 14
6-7
getNotifications(sync = 9)
event1 for C1, sync = 12
8-9-10
getConversation(C3)
C2 + sync = 15
11
event1 for C2, sync = 16
event1 for C2, sync = 17
getNotifications(sync = 15)
12
event1 for C2, sync = 16
event1 for C2, sync = 17
event1 for C2, sync = 19
event1 for C3, sync = 21
13
getNotifications(sync = 21)
empty, sync = 37
15
getNotifications(sync = 37)
HTTP 404
16
getConversation(C1, C2)
C1 + C2 + sync = 120

Chapter 9

REST Presences API

REST Presences API URI Space

Resource	URI	Supported Methods
presences	http://{host}:{port}/diamond/rest/api/V1/presences	GET

Description

Presences resource.

URL

http://{host}:{port}/diamond/rest/api/V1/presences

Remark

This resource retrieves aggregated presence statuses of users.

{userIds} should be provided with a comma as delimiter.

GET Method

GET	
Action	Retrieves aggregated presence statuses of users.
Request parameters	The IDs of the requested users.
Request body	None.
Request example	GET http://localhost:8080/diamond/rest/api/V1/presences/user1,user2,user3
Response	See "GET Response" (on page 54) for more information.
Error handling	If the external presence query has timed out, the aggregated presence state will be calculated from the EC presence state (Online > Online, Offline > Offline). If the list of user IDs includes a duplicate ID, the result will include it once only: HTTP 404 if the input list of user IDs is empty or contains invalid user ID. HTTP 200 in the case of success

GET Response

presenceResponse element	Type	Description
<presenceResponse>		
<presences>	Element	Presence statuses list
<presence>	Element	Presence status
<userId/>	Enum	Aggregated presence state: • ONLINE

presenceResponse element	Type	Description
		• OFFLINE • BUSY • AWAY
</presence>		
</presences>		

Presence GET Response Example

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<ns2:presenceResponse
xmlns:ns2="HP.Collaboration.Diamond.Api.Presence">
<presences>
 <presence>
  <userId>john.smith@hp.com</userId>
  <state>ONLINE</state>
 </presence>
 <presence>
  <userId>barbara.collins@hp.com</userId>
  <state>OFFLINE</state>
 </presence>
  <presence>
   <userId>sandra.clinton@hp.com</userId>
   <state>AWAY</state>
 </presence>
 <presence>
   <userId>christopher.jones@hp.com</userId>
   <state>BUSY</state>
 </presence>
</presences>
</ns2:presenceResponse>
```

Chapter 10

Common Data Types

This chapter includes:

attachment Element

This element is referenced from the following:

- conversation
- Attachments - POST Method.

Element	Type	Description
<attachment>		Attachment element
<id/>	String	Attachment ID
<description/>	String	Attachment description
<uploadDate/>	Long	Attachment upload date. See "Date and Time" (on page 66).
<uploaderId/>	String	Attachment uploader user ID
<isDeleted/>	Boolean	Is attachment deleted
<attachmentType/>	Enum	• FILE • CONTEXT_OBJECT • URL
<mimeType/>	String	Attachment MIME type. (Where **attachmentType** is FILE.)
<extension/>	String	Attachment file extension. (Where **attachmentType** is FILE.)
<contextObjectId/>	String	BTO object ID. (Where **attachmentType** is CONTEXT_OBJECT.)
<type/>	String	BTO object type. (Where **attachmentType** is CONTEXT_OBJECT.)
<url/>	String	URL. (Where **attachmentType** is URL.)
</attachment>		

attachmentsParams Element

This element is referenced from the following:

- Conversations - POST Body
- Attachments - POST Method

Element	Type	Description	Mandatory?	Default
<attachmentsParams>				
<attachmentParams>	fileParams	Should be provided when attaching a file	No	
<fileName/>	String	A file name	No	
<description/>	String	Some description	No	File name field
<data/>	Byte	The binary data	Yes	
<attachmentParams/>				
<attachmentParams>	urlParams	Should be provided when attaching a URL	No	
<url/>	String	The URL being attached	Yes	
<description/>	String	Description of the URL	No	URL field
<attachmentParams>	contextObjectIdParams	Should be provided when attaching an existing context object	No	
<id>	String	The ID of the existing context object being attached	Yes	
</attachmentParams>				
<attachmentParams>	contextObjectParams		No	
<contextObject>				
<id>	String	The ID of the context object	Yes	
<type>	String	The type of the object	Yes	
<description>	String	Description of the object	Yes	

Element	Type	Description	Mandatory?	Default
<createdDate>	Long	The date the object is created	Yes	
<relatedContacts>		Contacts for the object	No	
<contact>		Contact's name		
<email>		Contact's email address		
<reason>		Connection to the object		
</contact>				
</relatedContacts>				
<keyValues>		Client application key values		
<keyValue>				
<name/>				
<value/>				
</keyValue>				
</keyValues>				
</contextObject>				
</attachmentParams>				
</attachmentsParams>				

Attachments POST Body Example

Attaching 4 attachments; one from each params type.

```
<ns3:attachmentsParams
xmlns:ns2="HP.Collaboration.Diamond.Api.Profile"
xmlns:ns3="HP.Collaboration.Diamond.Api.Conversation"
xmlns:ns4="HP.Collaboration.Diamond.Api.Notification"
xmlns:ns5="HP.Collaboration.Diamond.Api.Repository">
<attachmentParams xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="ns3:fileParams">
<fileName>readme.txt</fileName>
<description>HP EC Manual</description>
<data>AQIDBA……==</data>
</attachmentParams>
```

```
<attachmentParams xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="ns3:urlParams">
<url>http://www.hp.com</url>
<description>HP's web-site</description>
</attachmentParams>
<attachmentParams xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="ns3:contextObjectIdParams">
<id>1234</id>
</attachmentParams>
<attachmentParams xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:type="ns3:contextObjectParams">
<contextObject>
<id>IM10002</id>
<type>Incident</type>
<description>Webmail login failure</description>
<createdDate>2147483647</createdDate>
<relatedContacts>
<contact>
<email>eyal@hp.com</email>
<reason>Owner</reason>
</contact>
<contact>
<email>olga@hp.com</email>
<reason>Assigned</reason>
</contact>
</relatedContacts>
<keyValues>
<keyValue>
<name>Priority</name>
<value>Urgent</value>
</keyValue>
</keyValues>
</contextObject>
</attachmentParams>
</ns3:attachmentsParams>
```

attachmentsResponse Element

This element is referenced from the following: Attachments - POST Method.

Element	Type	Description
<attachmentsResponse>		
<syncNumber/>	Long	Server current sync number
<attachments>	List	The created attachments
<attachment/>	Element	See attachment for more information.
</attachments>		
</attachmentsResponse>		

conversation Element

This element is referenced from the following:

- Conversations - GET Response
- notification
- conversationResponse
- conversationSearchResponse

Element	Type	Description
<conversation>		Conversation element
<id/>	String	Conversation ID
<isPartialInfo/>	Boolean	Does conversation element contain partial or full information
<isDeleted/>	Boolean	Is conversation deleted
<createdDate/>	Long	Creation date
<editedDate/>	Long	Edited date
<subject/>	String	Subject
<totalNumberOfPosts/>	Integer	Total number of posts
<visibility/>	Enum	Conversation visibility: • **Public:** Everyone may view this conversation.

Element	Type	Description
		• **Participants:** Only Participants may view this conversation.
<control/>	Enum	Conversation control: • **Everyone**: Every user may add users to the conversation • **Participants**: Only Participants may add users to the conversation • **Owners**: Only Owners may add users to the conversation
<isActive/>	Boolean	Whether conversation is active
<conversationUserInfo/>	Element	See "conversationParticipantInfo Element" (on page 63) for more information.
<attachments>	List	List of attachments
<attachment/>	Element	See "attachment Element" (on page 57) for more information.
</attachments>		
<participantsInfo>	List	List of Participants' information
<participantInfo/>	Element	See "participantInfo Element" (on page 68) for more information.
</participantsInfo>		
</conversation>		

conversationParticipantInfo Element

This element is referenced from the following: conversation.

Element	Type	Description
<conversationUserInfo>		conversationUserInfo element
<viewerType/>	Enum	Set to PARTICIPANT
<isRead/>	Boolean	Is the conversation read for the user who retrieved the notification
<markedTime/>	Date	For future use
</conversationUserInfo>		

conversationParticipantInfo Example

```
<conversationUserInfo xsi:type="ns3:conversationParticipantInfo"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
     <viewerType>PARTICIPANT</viewerType>
     <isRead>true</isRead>
     <markedTime>1323081305081</markedTime>
</conversationUserInfo>
```

conversationResponse Element

This element is referenced from the following:

- Conversation – GET Method
- Conversation - PUT Method
- Conversations - POST Method

Element	Type	Description
<conversationResponse>		
<syncNumber/>	Long	Server current sync number
<conversation/>	Element	See "conversation Element" (on page 61) for more information.
</conversationResponse>		

conversationSearchResponse Element

This element is referenced from the following:

- " GET Method" (on page 32)

Element	Type	Description
<conversationSearchResponse>		
<syncNumber/>	Long	Server current sync number.
<searchResult>	Element	A wrapper for the results.
<hitResults>	List	A list of results.
<hitResult>	Element	A result on a specific conversation.
<highlightFields>	List	A list of fields that corresponds to the search string.
<higlightField>	Element	A field that corresponds to the search string.
<type>	Enum	The field type: • SUBJECT • POST_BODY • USER • CONTEXT_OBJECT • FACET
<fragments>	List	A list of fragments in the field that corresponds to the search string.
<fragment>	String	A fragment that contains the corresponding search string. The string is outlined with a tags.
</higlightField>		
</highlightFields>		
<conversation/>	conversation	The conversation which contains the corresponding string. See "conversation Element" (on page 61).
</hitResult>		
</hitResults>		
</searchResult>		
</conversationSearchResponse>		

Date and Time

Date and time values in EC REST API are number of milliseconds since Jan 1, 1970, 00:00:00 GMT.

initiator Element

This element is referenced from the following: notification.

Element	Type	Description
<initiator>		Initiator element
All fields are the same as in the lightProfile element.		
</initiator>		

lightProfile Element

This element is referenced from the following:

- participantInfo
- initiator

Element	Type	Description
<lightProfile>		lightProfile element
<id/>	User ID	User ID
<jobTitle/>	String	User job title
<displayName/>	String	User display name
<emails>	List of strings	List of emails
<email/>	String	Email
</emails>		
<mobileImageResource/>	String	Mobile image resource
<thumbnailImageResource/>	String	Thumbnail image resource
<smallImageResource/>	String	Small image resource
<normalImageResource/>	String	Normal image resource
<source/>	profileSource enum	• INTERNAL, where the user exists within the organization (LDAP) • EXTERNAL, where this is not a user within the organization
</lightProfile>		

participantInfo Element

This element is referenced from the following: conversation.

Element	Type	Description
<participantInfo>		participantInfo element
<role/>	Enum	User role in the conversation: • OWNER • PARTICIPANT
<isRequired/>	Boolean	Is the user required in the conversation
<isDeleted/>	Boolean	Is the user deleted from the conversation
<addedDate/>	Date	When was the user added to the conversation. See "Date and Time" (on page 66).
<adderUserId/>	String	Who added the user to the conversation
<lightProfile/>	Element	See "lightProfile Element" (on page 67) for more information.
</participantInfo>		

participantInfo Element Example

```
<participantInfo>
     <role>PARTICIPANT</role>
     <isRequired>false</isRequired>
     <addedDate>1323081321800</addedDate>
     <lightProfile>
     <id>john.collins@hp.com</id>
     <displayName>John Collins</displayName>
     <jobTitle>Diamond server side developer</jobTitle>
     <emails>
       <email>collins.john@hp.com</email>
       <email>john.collins@hp.com</email>
     </emails>
     <source>INTERNAL</source>
     <thumbnailImageResource…</thumbnailImageResource>
     <mobileImageResource>…</mobileImageResource>
     <smallImageResource>…</smallImageResource>
     <normalImageResource>…</normalImageResource>
     </lightProfile>
     <adderUserId>mark.roberts@hp.com</adderUserId>
</participantInfo>
```

participants Element

This element is referenced from the following: Participants - POST Method.

Element	Type	Description	Mandatory?	Default
<participants>				
<participant>				
<id/>	String	The Participant's ID	Yes	
<isRequired/>	Boolean	Whether the Participant should be marked as required	No	False
<participant/>				
</participants>				

Participants POST Body Example

```
<ns3:participants..>
  <participant>
    <id>moses1</id>
  </participant>
  <participant>
    <id>joseph80</id>
    <isRequired>true</isRequired>
  </participant>
</ns3:participants>
```

post Element

This element is referenced from the following:

- notification
- postResponse

Element	Type	Description
<post>		Post element
<id/>	String	Post ID
<conversationId/>	String	Conversation ID
<isPartialInfo/>	Boolean	Whether post element contains partial information
<isDeleted/>	Boolean	Whether post deleted
<createdDate/>	Date	Post creation date. See "Date and Time" (on page 66).
<editedDate/>	Date	Post edit date. See "Date and Time" (on page 66).
<author/>	String	Post author user ID
<body/>	String	Post body
<postUserInfo/>	Element	See "postUserInfo Element" (on page 71) for more information.
</post>		

post Element Example

```
<post>
  <id>fe59229c-a51a-41d9-ae11-de842bf7e51a</id>
  <conversationId>a137e3b5-25e1-4c04-801e-
   94d3d8158f96</conversationId>
  <createdDate>1323081307722</createdDate>
  <editedDate>1323081307722</editedDate>
  <author>joseph.gutin@hp.com</author>
  <body>asdasd</body>
  <postUserInfo xsi:type="ns3:postParticipantInfo" xmlns:xsi="…">
  <viewerType>PARTICIPANT</viewerType>
  <isRead>false</isRead>
  </postUserInfo>
</post>
```

postParams Element

This element is referenced from the following:

- Conversations - POST Body
- Posts - POST Method

Element	Type	Description
<postParams>		
<body/>	String	The post body. See "Limitations" (on page 76) for more information.
</postParams>		

postResponse Element

This element is referenced from the following: Posts - POST Method.

Element	Type	Description
<postResponse>		
<syncNumber/>	Long	Server current sync number
<post/>	Element	See "post Element" (on page 70) for more information.
</postResponse>		

postUserInfo Element

This element is referenced from the following: post.

Element	Type	Description
<postUserInfo>		postUserInfo element
<viewerType/>	Enum	• VIEWER • PARTICIPANT
<isRead/>	Boolean	If **viewerType** is PARTICIPANT, this represents whether the post is read by the user who received the notification.
</postUserInfo>		

profile Element

The profile element is extended from lightProfile; it includes all its fields.

This element is referenced from the following:

- notification
- profileResponse

Element	Type	Description
<profile>		profile element
<isPartialInfo/>	Boolean	Whether the element contains partial information.
<firstName/>	String	User's first name
<lastName/>	String	User's last name
<sip/>	String	User's sip address
<largeImageResource/>	String	Large image resource
<isImageExists/>	Boolean	True if this user has uploaded a personal image.
</profile>		

profiles Element

This element is referenced from the following: Participants - POST Method.

Element	Type	Description
<profiles>		
<profile/>	Element	See "profile Element" (on page 72) for more information.
</profiles>		

profilesResponse Element

This element is referenced from the following:

- Participants - POST Method
- Profiles - GET Method

Element	Type	Description
<profilesResponse>		
<syncNumber/>	Long	Server current sync number
<profiles>	A list of profiles	
<profile/>	Profile	A profile
</profiles>		
<illegalUsersIdsResponses>		
<illegalUserIdResponse>		
<userId/>	String	A list of illegal IDs from the request
<illegalUserIdResponse>		
</illegalUsersIdsResponses>		
</profilesResponse>		

Chapter 11

Error Messages

Code	Exception
400	“Max number of characters in post exceeded.” “Cannot upload an image since it exceeds maximal size.” “The given job title is too long.” “File attachment exceeds size limitation.” “Participant is already in the conversation.” “Unable to add a post since the maximal number of allowed posts was achieved.” “Unable to add participants since to maximal number of allowed participants in conversation will exceed.” “Unable to add attachments since the maximal number of allowed attachments in the conversation will exceed.” "The given limit for conversations search is out of range."
401	“User or password specified are wrong.”
403	“User is not permitted to remove the requested user from the conversation.” “Unable to perform an action, since the given user is not a participant in the conversation.” “Only the conversation owner can add an external user to this conversation.”
404	“Conversation does not exist.” “Attachment does not exist.”
415	“Unknown image type. Verify your uploaded data is of an image and is not corrupted.”
500	Other errors

Chapter 12

HTTP Codes

Code	Cause
200	Successful operation
204	No content
206	Partial content
400	Bad request
401	Unauthorized
403	Unauthorized operation
404	Resource not found
415	Unsupported media type
500	Internal server error

Chapter 13

Limitations

The following table lists the default limitations in the EC server. Note that the EC administrator can change these values.

Failure to stay within a limitation results in a BAD_REQUEST error. HTTP Code 400; see "HTTP Codes" (on page 75) for more information.

Limitation	Value
Maximum number of posts in a conversation	10,000
Maximum number of attachments in a conversation	5,000
Maximum number of participants in a conversation	100
Maximum size of a file attachment	50 megabytes
Maximum number of posts that can be retrieved in a single request	200
Maximum size of a profile image (when uploaded.)	10 megabytes
Maximum subject length	255 bytes
Maximum post length	4,000 bytes
Maximum job title length	60 letters
Maximum number of letters for profiles search	4 letters
Maximum profiles search results	50