CPM Message Storage

Candidate Version 2.0 – 11 Jun 2013

Open Mobile Alliance

OMA-TS-CPM_MessageStorage-V2_0-20130611-C

Contents

Figures

Tables

1. Scope

This document provides the technical specifications for the message storage functionality of the CPM Enabler. The document covers the storage of Media Objects, CPM Messages, CPM File Transfer Histories, CPM Session Histories and CPM Conversation Histories in the network and the interactions between client and server components to access the network storage. The technical specifications are designed to fulfil the requirements, architecture and system concepts that are described in [OMA-CPM-RD], [OMA-CPM-AD] and [OMA-CPM-SD] respectively.

As such, these technical specifications provide the formal definitions of the CPM-MSG interface that has been identified in [OMA-CPM-AD]. Also, these technical specifications formally define the expected behaviour of the Message Storage Client and Message Storage Server functional components that have been identified in [OMA-CPM-AD].

2. References

2.1 Normative References

[OMA-CPM-AD] "Converged IP Messaging Architecture", Open Mobile Alliance™, OMA-AD-CPM-V2_0, URL:http://www.openmobilealliance.org/

[OMA-CPM-RD] "Converged IP Messaging Requirements", Open Mobile Alliance™, OMA-RD-CPM-V2_0, URL:http://www.openmobilealliance.org/

[OMA-CPM-SD] "Converged IP Messaging System Description", Open Mobile Alliance™, OMA-TS-CPM_System_Description-V2_0, URL:http://www.openmobilealliance.org/

[OMA-SCRRULES] "SCR Rules and Procedures", Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:http://www.openmobilealliance.org/

[OMA-SEC-CF] "Security Common Functions Requirements", Open Mobile Alliance, OMA-RD-SEC_CF-V1.0, URL:http//www.openmobilealliance.org/

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL:http://www.ietf.org/rfc/rfc2119.txt

[RFC2177] "IMAP IDLE command", B. Leiba, June 1997, URL:http://www.ietf.org/rfc/rfc2177.txt

[RFC2246] "The TLS Protocol Version 1.0", T. Dierks et al, January 1999, URL:http://www.ietf.org/rfc/rfc2246.txt

[RFC2387] "The MIME Multipart/Related Content-type", E. Levinson, August 1998, URL:http://www.ietf.org/rfc/rfc2387.txt

[RFC2392] "Content-ID and Message-ID Uniform Resource Locator", E. Levinson, August 1998, URL:http://www.ietf.org/rfc/rfc2392.txt

[RFC3501] "Internet Message Access Protocol - version 4rev1", M. Crispin, March 2003, URL:http://www.ietf.org/rfc/rfc3501.txt

[RFC3516] "IMAP4 Binary Content Extension", L. Nerenberg, April 2003, URL:http://www.ietf.org/rfc/rfc3516.txt

[RFC3862] "Common Presence and Instant Messaging (CPIM) : Message Format", G. Klyne et al, August 2004, URL:http://www.ietf.org/rfc/rfc3862.txt

[RFC4314] "IMAP4 Access Control List (ACL) Extension", A. Melnikov, December 2005, URL:http://www.ietf.org/rfc/rfc4314.txt

[RFC4467] "Internet Message Access Protocol (IMAP) – URLAUTH Extension", M. Crispin, May 2006, URL:http://www.ietf.org/rfc/rfc4467.txt

[RFC4551] "IMAP Extension for Conditional STORE Operation or Quick Flag Changes Resynchronization", A. Melnikov et al, June 2006, URL:http://www.ietf.org/rfc/rfc4551.txt

[RFC5092] "IMAP URL Scheme", A. Melnikov, Ed. et al, November 2007, URL:http://www.ietf.org/rfc/rfc5092.txt

[RFC5161] "The IMAP ENABLE Extension", A. Gulbrandsen, Ed. et al, March 2008, URL:http://www.ietf.org/rfc/rfc5161.txt

[RFC5162] "IMAP4 extension for Quick Mailbox Resynchronization", A. Melnikov, March 2008, URL:http://www.ietf.org/rfc/rfc5162.txt

[RFC5257] "Internet Message Access Protocol - ANNOTATE Extension", C. Daboo et al, June 2008, URL:http://www.ietf.org/rfc/rfc5257.txt

[RFC5259] "Internet Message Access Protocol -CONVERT extension", A. Melnikov, Ed. et al, July 2008, URL:http://www.ietf.org/rfc/rfc5259.txt

[RFC5322] "Internet Message Format", P. Resnick, October 2008. URL:http://www.ietf.org/rfc/rfc5322.txt

[RFC5423] "Internet Message Store Event", R. Gellens, et al, March 2009, URL:http://www.ietf.org/rfc/rfc5423.txt

[RFC5464] "The IMAP METADATA extension", C. Daboo, February 2009, URL:http://www.ietf.org/rfc/rfc5464.txt

[RFC5465] 'The IMAP NOTIFY Extension", A. Gulbrandsen, et al, February 2009, URL:http://www.ietf.org/rfc/rfc5465.txt

[RFC5551] ”Lemonade Notifications Architecture’, R. Gellens, August 2009, URL:http://www.ietf.org/rfc/rfc5551.txt

[RFC5788] “IMAP4 Keyword Registry”, A. Melnikov et al, March 2010, URL:http://www.ietf.org/rfc/rfc5788.txt

2.2 Informative References

[OMADICT] “Dictionary for OMA Specifications”, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_9, URL:http://www.openmobilealliance.org/

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2 Definitions

CPM Address	See [OMA-CPM-RD].
CPM Chat Message	See [OMA-CPM-RD].
CPM Conversation	See [OMA-CPM-RD].
CPM Conversation History	See [OMA-CPM-RD].
CPM Conversation Identity	See [OMA-CPM-SD].
CPM File Transfer	See [OMA-CPM-RD].
CPM File Transfer History	See [OMA-CPM-RD].
CPM Message	See [OMA-CPM-RD].
CPM Participating Function	See [OMA-CPM-AD].
CPM Pre-defined Group	See [OMA-CPM-RD].
CPM Session History	See [OMA-CPM-RD].
CPM Standalone Message	See [OMA-CPM-RD].
CPM User	See [OMA-CPM-RD].
Media Object	See [OMA-CPM-AD].
CPM Session	See [OMA-CPM-RD].
CPM Session Invitation	See [OMA-CPM-RD].
Message Storage Client	See [OMA-CPM-AD].
Message Storage Server	See [OMA-CPM-AD].
Principal	See [OMADICT].
Unique IDentifier	A 32-bit value assigned to each stored object and used with the UID validity value to form a 64-bit value that is unique to a stored object and cannot be referred to any other stored object in the folder or any subsequent folder of the message storage [RFC3501].
UIDVALIDITY	A 32-bit representation of the creation date and time of a folder

3.3 Abbreviations

ACL	See [OMADICT]
CPIM	Common Presence and Instant Messaging
CPM	See [OMADICT]

IMAP	See [OMADICT]
OMA	See [OMADICT]
MIME	See [OMADICT]
PSK	Pre Shared Key
SASL	Simple Authentication and Security Layer
TLS	See [OMADICT]
UID	Unique (Message) Identifier
URL	See [OMADICT]

Note: Abbreviations defined in the OMA Dictionary complements this section.

4. Introduction

The CPM message storage functionality allows the storage of CPM Messages, CPM File Transfer Histories, CPM Session Histories, CPM Conversation Histories, and any potential Media objects either stand-alone or attached to CPM Messages and CPM Session Histories in a network-based storage on behalf of CPM Users.

The CPM message storage functionality authenticates and authorizes CPM Users to being able to retrieve, organize, set permissions, receive event notifications, synchronize with CPM Users device's local message storage and manage (e.g., copy, remove, move etc.,) the storage objects that are stored on it. It also allows CPM Users to search the storage objects with key words.

4.1 CPM Version 1.0

The CPM 1.0 offers:

- Storage objects
    - CPM Standalone Messages, CPM File Transfer Histories, CPM Session Histories and CPM Conversation Histories
    - any potential Media objects either stand-alone or attached to CPM Standalone Messages, CPM File Transfer Histories and CPM Session Histories
- User authentication and authorization mechanisms
- Operations
    - folder operations e.g., create, list, set active folder, move, remove, search
    - stored object operations e.g., store, fetch, copy, remove, preview
    - metadata operations on stored objects e.g., update metadata
    - access rights on stored objects e.g., set, get, delete
- generate references to stand-alone or attached to CPM Standalone Messages and CPM Session History objects
- allows fetch of stored Media objects by reference
- synchronize between network-based storage and device's local storage
- Notifications
    - notifications about changes in stored resources

4.2 CPM Version 2.0

Backward compatibility of version 2.0 with CPM Version 1.0 excludes support of session history object as defined in CPM version 1.0. This has been replaced in CPM version 2.0 by the session history folder and session info object.

5. Common Procedures

5.1 Authorization and Authentication

The IMAPv4 [RFC3501] protocol is used to access the Message Storage Server. This section defines authentication and authorization mechanisms of IMAPv4 used by the Message Storage Server.

5.1.1 Authentication

CPM's message storage functionality supports the two authentication mechanisms listed below, as defined for IMAPv4 [RFC3501];

1. SASL authentication via the AUTHENTICATE command ([RFC3501]); and
2. Username/password in plain text authentication via the LOGIN command ([RFC3501]).

The username/password for the second authentication mechanism is separately managed by the CPM service. The password may be pre-configured by the CPM system when the CPM user subscribes to the CPM service.

In addition to these authentication mechanisms, TLS/PSK-TLS, as defined in [RFC2246] and [OMA-SEC-CF], is optional and complementary to simple authentication-only SASL mechanisms or deployed clear-text password login commands. In this way, IMAPv4 can be immune to eavesdropping and hijacking attacks. Using TLS/PSK-TLS, the Message Storage Client also can authenticate the Message Storage Server by checking the certificate supplied by the Message Storage Server.

5.1.2 Authorization

The Message Storage Server uses standard IMAP4 [RFC3501] functionality to enforce access to the stored resources, extended with the possibility for CPM Users to define access control lists (ACL) for their own stored resources.

The ACL management operations and related standard rights are defined in IMAPv4 ACL extension [RFC4314].

In addition to that, the Message Storage Server also allows the use of the IMAP4 URLAUTH extension, as defined in [RFC4467] and [RFC5092]. This URLAUTH extension provides a means by which a Message Storage Client can use URLs carrying authorization information to access limited data on the Message Storage Server.

5.2 Storage Folder and Objects

Per the description provided in section 5.5 of the CPM System Description [OMA-CPM-SD], a CPM Message Storage Server may contain the following items.

- message object,
- session history folder,
- file transfer history object,
- conversation history folder,
- stand-alone Media Object,
- user folder,
- session info object,
- group state object.

In this section, these storage objects are specified in terms of their names and identities, which can be used for various CPM message storage operations in accordance with IMAP4 [RFC3501] and its extensions.

5.2.1 Message Object

A message object, matching the message concept described in [RFC3501], is a Message Storage Server stored item. For execution of various IMAP commands, the message object is identified by a 3-component identifier consisting of a folder name, a message identification number associated to the message object and a folder validity value as specified below:

1. The folder name is the name of the folder in the Message Storage Server where the message is stored,
2. The message identification number is either a message sequence number or a 32-bit Unique Identifier (UID), which is specified according to [RFC3501],
3. The folder validity value or the Unique IDentifier validity (UIDVALIDITY) is another 32-bit value as defined in [RFC3501] distinguishing folders of the same name from each other.

NOTE: For private messages exchanged during a CPM Group Session: if the actual recipient list for each message is recorded, this MAY be recorded in the CPIM "To" headers of the MSRP messages.

5.2.2 Session History Folder

The session history folder is viewed as a special kind of sub-folder and is realized via the mailbox concept of IMAP4. The name of a session history folder SHALL be the Contribution-ID of the corresponding CPM Session. For more information, please refer to the storage representation examples in Appendix I.2.

A session history folder SHALL contains:

- One session info object, and
- Zero or more message object(s), and/or
- Zero or more file transfer object(s), and/or
- Zero or more stand-alone Media Object(s), and/or
- Zero or more group state objects

The name of the session history folder MUST be unique within the scope of the parent Conversation History Object.

5.2.3 File Transfer History Object

The definition and identity specification of the message object SHALL be applicable to the file transfer history object, which consists of a UID, UIDVALIDITY and their values.

The file transfer history object is a message formatted according to [] and the clarifications given in below.

The content inserted in the headers of this message is retrieved from the CPM File Transfer invitation.

The body of this message SHALL be a "multipart/related" body, structured as follows:

- the first body part SHALL contain metadata about the CPM File Transfer History and is formatted according to section 5.2.3.1 "Application/X-CPM-File-Transfer Content Type Definition"
- the other body parts SHALL contain the files that were sent and received via MRSP during the CPM File Transfer.

An example of a file transfer history object is shown in Appendix E.

Internet Message Format [RFC5322] header	Internet Message Format Parameter status	Content
From	Mandatory	Set to the address of the initiator of the CPM File Transfer, retrieved from the authenticated originator's CPM Address in the SIP INVITE request.
Date	Mandatory	Set to, in order of preference: - the Date of the SIP INVITE request if available, otherwise, - the date and time when the CPM File Transfer recording was started.
Subject	Optional	Set only if "Subject" header is set in the SIP INVITE request.
Conversation-ID	Mandatory	Set to the Conversation-ID of the SIP INVITE request. NOTE: this is a new field, extending the Internet Message Format, as defined in 3.6.8 of [].
Contribution-ID	Mandatory	Set to the Contribution -ID of the SIP INVITE request. NOTE: this is a new field, extending the Internet Message Format, as defined in 3.6.8 of [].
InReplyTo-Contribution-ID	Optional	Set to the InReplyTo -Contribution -ID of the SIP INVITE request. NOTE: this is a new field, extending the Internet Message Format, as defined in 3.6.8 of [].
Content type	Mandatory	Set to the following value : multipart/related;boundary=cpm; type="Application/X-CPM-File-Transfer" NOTE: the new Application/X-CPM File-Transfer content-type is defined in section 5.2.4.1.
Message body	Mandatory	Contains a list of MIME entities separated by the "cpm" boundary. The first MIME entity is the "root", as defined in [RFC2387] and its content-type is "Application/X-CPM-File-Transfer". The other MIME entities are the files that were exchanged during the CPM File

		Transfer.

Table 1: CPM file transfer history object format

5.2.3.1 Application/X-CPM-File-Transfer Content Type Definition

This new MIME type is used to store metadata about the CPM File Transfer.

The body of a message with the content-type header set to Application/X-CPM-File-Transfer SHALL be formatted as follows :

<file-transfer-type>: this element indicates the type of file transfer. It may be set to "Ad-Hoc", "Pre-Defined" or "1-1".

<invited-participants>: this element contains the list of addresses of the invited participants, separated by a semi-column, or the address of the Pre-defined Group.

Next, for each file, metadata information MAY be stored within a <file-object> element as follows:

<file-object>

<cid>cid: set to the Content-ID, as defined in [RFC2392], of the corresponding stored file.**</cid>**

<sdp>SDP parameters associated with the corresponding stored file (e.g., file name).**</sdp>**

</file-object>

5.2.4 Conversation History Folder

The conversation history folder is realized via the mailbox concept of IMAP4. A conversation history folder stores all items related to a single CPM Conversation as objects (e.g. message objects, file transfer history objects and stan-dalone Media Objects) and sub-folders (i.e. session history folders). The name of the conversation history folder is set to the CPM Conversation Identity used in that CPM Conversation.

5.2.5 Stand-alone Media Object

The standalone Media Object is realized by the message concept of IMAP4, whereby the Media Object is wrapped into a MIME formatted object in order to fit into an IMAP4 message. Other than the formatting of the contents, the standalone Media Object is the same as the message object defined in section 5.2.1 "***Error! Reference source not found.***", including the definition and naming specification of the message object in terms of UID, next UID and their values.

5.2.6 User Folder

The user folder is a Message Storage folder realized by the mailbox concept of IMAP4, described in [RFC3501]. The user folder is identified by the name given to it. The CPM user folder aligns with the rules and procedures for names of the mailbox concept of IMAP4, as described in [RFC3501]. For additional information, see section 5.5.1.1 of the CPM System Description [OMA-CPM-SD].

5.2.7 Session Info Object

A session info object contains only the session information of the recorded CPM Session. The session info object is only relevant to the session history folder it belongs to and SHALL NOT be moved to another folder or removed from its session history folder.

The definition and identity specification of the message object SHALL be applicable to the session info object, which consists of a UID, UIDVALIDITY and their values. The session info object is a message formatted according to [] and the clarifications given in above.

The content inserted in the headers of this message is retrieved from the CPM Session Invitation.

The body of this object SHALL contain metadata about the CPM Session History and is formatted according to the table below.

An example of a session info object is shown in Appendix D.

Internet Message Format [RFC5322] header	Internet Message Format Parameter status	Content
From	Mandatory	Set to the address of the initiator of the CPM Session, retrieved from the authenticated originator's CPM Address in the SIP INVITE request.
Date	Mandatory	Set to, in order of preference: - the Date of the SIP INVITE request if available, otherwise, - the date and time when the CPM Session recording was started.
Subject	Optional	Set only if "Subject" header is set in the SIP INVITE request.
Conversation-ID	Mandatory	Set to the Conversation-ID of the SIP INVITE request. NOTE: this is a new field, extending the Internet Message Format in accordance with 3.6.8 of [].
Contribution-ID	Mandatory	Set to the Contribution -ID of the SIP INVITE request. NOTE: this is a new field, extending the Internet Message Format in accordance with 3.6.8 of [].
InReplyTo-Contribution-ID	Optional	Set to the InReplyTo -Contribution -ID of the SIP INVITE request. NOTE: this is a new field, extending the Internet Message Format in accordance with 3.6.8 of [].
Content type	Mandatory	Set to the following value : multipart/related;boundary=cpm; type="Application/X-CPM-Session" NOTE: the new "Application/X-CPM-Session" content-type is defined in section 5.2.3.1.

Table 2: CPM session info object format

5.2.8 Group State Object

The Group State object is an XML object with XML Content Type "application/group-state-object+xml" containing the last valid CPM Group Session Identity for the Group Chat, as well as the last set of active participants at the time the Group Chat was torn down because of inactivity.

The format of the Group State object includes the following elements and attributes:

1) SHALL include one <groupstate> element containing:
   a) SHALL include one "lastfocussessionid" attribute, containing the CPM Group Session Identity;
   b) SHALL include one "timestamp" attribute;
   c) SHALL include one "group-type" attribute, containing the type of CPM Group Session (e.g. Closed);
2) SHALL include one or more <participant> element containing:
   a) SHALL include "name" attribute of the current participant in the CPM Session;
   b) SHALL include :comm-addr: attribute containing the communication address (i.e. URI) of the participant.

An example of the Group State object is shown in **Error! Reference source not found.** below:

Example:

```
<groupstate
            timestamp="2012-06-13T16:39:57-05:00"
            lastfocussessionid="da9274453@company.com"
            group-type="closed">
      <participant name="bob" comm-addr="tel:+16135551212"/>
      <participant name="alice" comm-addr="tel:+15195551212"/>
      <participant name="bernie" comm-addr="tel:+15145551212"/>
</groupstate>
```

Table 3: Group State Object Example

5.3 Identification of Storage Objects

The combination of a folder name, its stored object's UID, and the folder's UIDVALIDITY MUST permanently and persistently refer to one and only one stored object in a Message Storage Server. In particular, the internal date, size, envelope, body structure, and message texts (HEADER, TEXT and all BODY fetch data items) MUST never change according to [RFC3501]. This requirement does not include message numbers, nor does it include object attributes that can be set by a STORE command (e.g., FLAGS).

Associated with any folder object, there is a next UID value, which is the predicted value that will be assigned to a new storage object in that folder. Provided by the CPM Message Storage Server, this next UID is intended to provide a means for the CPM Message Storage Client to determine whether or not any messages have been delivered to the folder since the previous time it checked this value.

NOTE: The "next UID" value of a folder changes whenever a new object is stored in that folder.

5.4 Notifications

The Message Storage Server SHALL notify the Message Storage Client of the changes in the stored resources either solicited as a result of a client request or unsolicited and unilaterally according to the IMAP extension for the general notification model [RFC5551].

The data content of a notification may include a wide range of information as follows:

1. Folder size and stored objects status updates, e.g., addition and/or removal of messages to prevent synchronization errors,
2. "Next UID" for the changes to the storage requested by a third entity, e.g., CPM PF,

3. Stored objects' attribute flags, e.g., "\Recent" flag for recently arrived objects in a folder.

5.5 Metadata Structure

CPM's message storage functionality supports a metadata model that consists of three distinct parts:

1. A set of metadata flags (i.e. IMAP system flags and keywords) that are associated with message objects, file transfer history objects and stan-dalone Media Objects. These flags indicate additional state information about the stored object, and
2. A set of metadata annotations that can be associated with folder objects (i.e. mailboxes) and message objects (i.e. messages)stored in the Message Storage Server. These annotations provide system or user-defined information that the system or the user associates with these stored objects.
3. A set of metadata annotations that can be associated with the server (i.e. server annotations). These annotations provide system- or user-defined information that the system or the user associates with the server rather than individual objects.

The Message Storage Client and the Message Storage Server SHALL at least support metadata flags, folder metadata annotations and server annotations, and, in addition, MAY support message metadata annotations.

With respect to the metadata flags, the Message Storage Client and Message Storage Server SHALL support at least the following flags defined in [RFC3501], [RFC5788] and Appendix C:

- \Seen (message has been read),
- \Answered (message has been answered),
- \Flagged (message is "flagged" for urgent and/or special attention),
- \Deleted (message is "deleted" for removal by later EXPUNGE),
- \Draft (message has not completed composition (marked as a draft),
- \Recent (message is "recently" arrived in this mailbox),
- $MDNSent (A disposition notification has been sent for this message),
- $Forwarded (message has been forwarded), and
- \read-report-sent (A read receipt has been sent for this message) as defined in Appendix C.

NOTE: All flag assignments and operations SHALL be handled according to the procedures specified in [RFC3501].

With respect to the server metadata and folder metadata annotations, the Message Storage Client and the Message Storage Server SHALL support the structure defined in [RFC5464] for metadata annotations as well as the METADATA entries defined in Appendix H for folder objects.

With respect to the message metadata annotations, if supported, the Message Storage Client and the Message Storage Server SHALL support the structure defined in [RFC5257] for message objects, for file transfer history objects and for standalone Media Objects.

6. Procedures at Message Storage Client

The Message Storage Client is a functional component of the CPM enabler, which allows the CPM User to view and manage (store, fetch, delete etc) the resources stored in the Message Storage Server. In addition to that the Message Storage Client notifies the CPM User of any changes to the stored resources in the Message Storage Server (e.g. new message arrived, message got read on another client).

The Message Storage Client SHALL act as anIMAP4 client as defined in [RFC3501]. In addition to that, the Message Storage Client SHALL support the "ACL" IMAP4 extension as defined in [RFC4314], the "URLAUTH" IMAP4 extension as defined in [RFC4467], the "CONDSTORE" IMAP4 extension as defined in [RFC4551], the "ENABLE" IMAP4 extension as defined in [RFC5161RFC5161], the "QRESYNC" IMAP4 extension as defined in [RFC5162] and the "METADATA" IMAP4 extension as defined in [RFC5464]. Also, the Message Storage Client MAY support the "ANNOTATE" IMAP4 extension as defined in [RFC5257] and/or the "CONVERT" IMAP4 extension as defined in [RFC5259].

6.1 General Operations

6.1.1 Authenticate Operation

When the CPM Client needs to authenticate with the Message Storage Server, it SHALL either use the SASL method or the plain-text username/password method.

When the Message Storage Client wants to use the SASL method, the Message Storage Client SHALL send to the Message Storage Server an AUTHENTICATE request as defined in [RFC3501] with the authentication mechanism that it wants to use and then SHALL complete the authentication process as defined in [RFC3501].

When the Message Storage Client wants to use the plain-test username/password method, the Message Storage Client SHALL send to the Message Storage Server a LOGIN request as defined in [RFC3501] with the username and password associated with the Message Storage Client's CPM User.

The CPM Client MAY set up a TLS session prior to authenticating with the Message Storage Server. In order to do so, the Message Storage Client SHALL send to the Message Storage Server a STARTTLS request as defined in [RFC3501] and complete the TLS session setup. The Message Storage Server SHALL support either TLS or PSK-TLS as defined in [RFC2246] and [OMA-SEC-CF] for the TLS protocol negotiations.

6.1.2 Set Active Folder Operation

When a Message Storage Client needs to set a particular folder as the active folder, the Message Storage Client SHALL send to the Message Storage Server a SELECT request as defined in [RFC3501] with the folder name of the folder that is to be set as the active folder.

6.2 Access Control List Operations

6.2.1 Set Access Control List

When a Message Storage Client needs to set the access rights for another Principal on one of the folders of its associated CPM User, the Message Storage Client SHALL send to the Message Storage Server a SETACL request as defined in [RFC4314] with the folder name, the granted access rights and the identifier of the Principal to which access is given.

6.2.2 Get Access Control List

When a Message Storage Client needs to get the access control list on one of the folders of its associated CPM User, the Message Storage Client SHALL send to the Message Storage Server a GETACL request as defined in [RFC4314] with the folder name.

6.2.3 Delete Access Control List

When a Message Storage Client needs to delete the access rights for another Principal on one of the folders of its associated CPM User, the Message Storage Client SHALL send to the Message Storage Server a DELETEACL request as defined in [RFC4314] with the folder name and the identifier of the Principal to which access is given.

6.2.4 Access Rights Retrieval Operations

When a Message Storage Client needs to retrieve the access rights for another Principal on one of the folders of its associated CPM User, the Message Storage Client SHALL send to the Message Storage Server a LISTRIGHTS request as defined in [RFC4314] with a folder name and the identifier of the Principal whose access rights are to be retrieved.

6.3 Message and History Operations

6.3.1 Object Store Operation

When a Message Storage Client needs to store a message object, a file transfer history object or a standalone Media Object into a folder on the Message Storage Server, the Message Storage Client SHALL send to the Message Storage Server an APPEND request as defined in [RFC3501] including the name of the folder and the data of the message object, file transfer history object or standalone Media Object data. The Message Storage Client MAY include an initial set of metadata flags in the APPEND request towards the Message Storage Server, as defined in [RFC3501]. The Message Storage Client also MAY include a set of metadata annotations in the APPEND request towards the Message Storage Server, as defined in [RFC5257].

NOTE: The set of metadata flags and metadata annotations associated with the stored object can be changed later using the metadata update operation defined in section **Error! Reference source not found.** "***Error! Reference source not found.***".

6.3.2 Object Fetch Operation

When a Message Storage Client needs to fetch a message object, file transfer history object or stand-alone Media Object from the active folder on the Message Storage Server, the Message Storage Client SHALL send to the Message Storage Server a FETCH or a UID FETCH request as defined in [RFC3501] with the UID(s) pointing to a stored object(s) in Message Storage Server.

NOTE: The Message Storage Client may specify that only specific parts of a message object, file transfer history object or standalone Media Object are to be fetched, as defined in [RFC3501].

6.3.3 Object Preview Fetch Operation

When a Message Storage Client needs to fetch a preview of a message object, file transfer history object or standalone Media Object from the active folder on the Message Storage Server, the Message Storage Client SHALL send to the Message Storage Server a CONVERT or a UID CONVERT request as defined in [RFC5259] with the UID pointing to the stored object on the Message Storage Server to be previewed.

NOTE 1: The Message Storage Client may specify that only a preview of specific parts of a message object, file transfer history object or standalone Media Object is to be fetched, as defined in [RFC3501] and [RFC5259].

NOTE 2: The CONVERT and UID CONVERT commands can be used to transcode the media type of a MIME part into another media type and/or into the same media type with different encoding parameters.

NOTE 3: A "Preview Fetch" operation MAY involve a server-side content adaptation in response to the Message Storage Client's request for the stored object in a compacted or digested form rather than in its original full size and shape.

NOTE 4: Conversions only affect what is sent to the Message Storage Client; the original data in the Message Storage Server SHALL NOT be converted.

6.3.4 Object Copy Operation

When a Message Storage Client needs to copy an existing message object, an existing file transfer history object or an existing standalone Media Object in the Message Storage Server, the Message Storage Client SHALL send to the Message Storage Server a COPY request as defined in [RFC3501] with the UID pointing to an object to be copied from the Message Storage Server and specifying the destination folder.

6.3.5 Object Remove Operation

When a Message Storage Client needs to remove a stored message object, a file transfer history object, a group state object or a stored standalone Media Object, the Message Storage Client SHALL send to the Message Storage Server a STORE request as defined in [RFC3501] with the UID pointing to the stored object to update the flag list associated with the object's data and setting the "\Deleted" flag. The Message Storage Client SHALL NOT remove a stored session info object unless it is done in the context of the entire session history folder itself being (re)moved.

After setting the "\Deleted" flag, the Message Storage Client SHALL send to the Message Storage Server an EXPUNGE request as defined in [RFC3501] in order to permanently remove the message(s) that have been identified for removal from the list of objects with the "\Deleted" flag set.

NOTE: The Message Storage Client can use the EXPUNGE request to permanently remove multiple messages, i.e. there may be multiple STORE commands to set the "Deleted" flag before an EXPUNGE command is executed.

6.4 Folder Operations

6.4.1 Folder Create Operation

When a Message Storage Client needs to create a new folder, the Message Storage Client SHALL send to the Message Storage Server a CREATE request as defined in [RFC3501] including the name of the folder.

6.4.2 List Folder Operation

When a Message Storage Client needs to list the contents of the currently selected folder, the Message Storage Client SHALL send to the Message Storage Server a LIST request as defined in [RFC3501].

6.4.3 Folder Move Operation

NOTE: The folder move operation is also used for renaming folders.

When a Message Storage Client needs to rename a folder or to move a folder, the Message Storage Client SHALL send to the Message Storage Server a RENAME request as defined in [RFC3501] including the old name and the new name of the folder to be renamed or moved.

6.4.4 Folder Remove Operation

When a Message Storage Client needs to delete a folder, the Message Storage Client SHALL send to the Message Storage Server a DELETE request as defined in [RFC3501] including the name of the folder that is to be deleted.

6.4.5 Folder Search Operation

When a CPM Client needs to search in the active folder on the Message Storage Server, the Message Storage Client SHALL send to the Message Storage Server SEARCH request as defined in [RFC3501] including one or more search key data.

6.5 Reference Operations

6.5.1 Generate Reference Operation

When a Message Storage Client needs to generate a reference for (part of) a message object, a file transfer history object or a stand-alone Media Object, the Message Storage Client SHALL send to the Message Storage Server a GENURLAUTH request as defined in [RFC4467] including an IMAP URL (as per [RFC5092] and [RFC4467]) pointing to the (part of) the object for which a reference needs to be created.

6.5.2 Fetch by Reference Operation

When a Message Storage Client needs to fetch the (part of) a message object, a file transfer history object or a stand-alone Media Object on the basis of the reference, the Message Storage Client SHALL send to the Message Storage Server a URLFETCH request as defined in [RFC4467].

6.6 Metadata Management Operations

The Message Storage Client SHALL support:

- updating the metadata flags defined for IMAP4 message objects in section **Error! Reference source not found.** *"**Error! Reference source not found.**"*,
- fetching the metadata flags defined for IMAP4 message objects in section **Error! Reference source not found.** *"**Error! Reference source not found.**"*,
- updating server metadata annotations defined for IMAP4 mailboxes in section **Error! Reference source not found.** *"**Error! Reference source not found.**"*.
- fetching server metadata annotations defined for IMAP4 mailboxes in section **Error! Reference source not found.** *"**Error! Reference source not found.**"*.

The Message Storage Client MAY support:

- updating message and folder metadata annotations defined for IMAP4 message objects in section **Error! Reference source not found.** *"**Error! Reference source not found.**"*.
- fetching message and folder metadata annotations defined for IMAP4 message objects in section **Error! Reference source not found.** *"**Error! Reference source not found.**"*.

6.6.1 Metadata Update Operation

When a Message Storage Client needs to update the metadata flags of a message object, the Message Storage Client SHALL send to the Message Storage Server a STORE request as defined in [RFC3501] including the UID of the message object and the changes to the flags (e.g., \Seen, \Deleted, etc.).

When a Message Storage Client needs to update message metadata annotations, if supported, the Message Storage Client SHALL send to the Message Storage Server a STORE request as defined in [RFC3501] including the UID of the message object and the 'ANNOTATION' data item as defined in [RFC5257].

When a Message Storage Client needs to update server or folder metadata, the Message Storage Client SHALL send to the Message Storage Server a SETMETADATA request as defined in [RFC5464] including:

- the name of the folder whose metadata is to be updated,
- an entry specifier and value pair (i.e. the entry specifier and the value corresponding to the entry specifier).

NOTE: The name of the folder SHALL be an empty string (i.e. "") to update server annotations.

The SETMETADATA request allows a requestor to include multiple update in a request. To reduce network overhead and delays, it is RECOMMENDED that a requestor updates all annotations at once by including all entry specifier and value pairs in a single request.

6.6.2 Metadata Fetch Operation

When a Message Storage Client needs to retrieve the metadata flags of a message object, the Message Storage Client SHALL send to the Message Storage Server a FETCH request as defined in [RFC3501] including the UID of the message object and the 'FLAGS' data item name.

When a Message Storage Client needs to retrieve the metadata flags of a message object, if supported, the Message Storage Client SHALL send to the Message Storage Server a FETCH request as defined in [RFC3501] including the UID of the message object and the 'ANNOTATION' data item name, as defined in [RFC5257].

When a Message Storage Client needs to retrieve server or folder metadata annotations, the Message Storage Client SHALL send to the Message Storage Server a GETMETADATA request as defined in [RFC5464] including:

- the name or the folder whose metadata is requested,
- the entry specifier.

NOTE: The name of the folder SHALL be an empty string (i.e. "") to retrieve server annotations.

The GETMETADATA request allows a requestor to extend the list of entry specifiers in a fetch operation using the DEPTH command option. To reduce network overhead and delays, it is RECOMMENDED that a requestor retrieves annotations in batch(es), using the DEPTH command option. Therefore, a requestor's request SHOULD include:

- the DEPTH command option with the value "infinity",
- the entry specifier as a higher level specifier, e.g. "/shared/OMNA", "/shared/OMNA/OMACPM20", etc.

6.7 Synchronization

[OMA-CPM-SD] gives a description of the synchronization process between the Message Storage Client and the Message Storage Server. This process only consists of operations described above and therefore needs no further explanation in this section.

While executing these operations, the Message Storage Client SHALL support and use the IMAP4 extensions described in [RFC4551], [RFC5161] and [RFC5162] to get an optimized and quick synchronization between the – potentially offline – Message Storage Client and the Message Storage Server.

6.8 Notification Operations

The IMAP Extension [RFC5465] allows the Message Storage Client to request for solicited notifications about events in specified folders of a Message Storage Server. For making this request, the Message Storage Client SHALL send a NOTIFY command to the Message Storage Server to limit its unsolicited notifications to certain selected folders and certain events such as objects being added to or deleted from those selected folders. Without this Message Storage Client NOTIFY command, an IMAP server will only send information about the changes in the Message Storage Server to the client in the following cases:

1. as the result of a Message Storage Client command such as FETCH responses to a FETCH or STORE command),
2. as unsolicited responses sent just before the end of a command (e.g., EXISTS or EXPUNGE) as the result of changes in other sessions, and
3. during the Message Storage Client's IDLE command

NOTE: Per [RFC2177], the IDLE command provides a way for the client to go into a mode where the Message Storage Server pushes its notifications about the server events in selected folders.

Upon registration and receiving of either solicited or unsolicited notifications from the Message Storage Server, the Message Storage Client SHALL update its corresponding information in the client(s)' locally stored resources.

7. Procedures at Message Storage Server

The Message Storage Server is a functional component of the CPM enabler, which allows authorized and/or authenticated principals (such as Message Storage Clients or CPM Participating Functions) to access a resource in the Message Storage Server.

The Message Storage Server SHALL act as an IMAP4 server as defined in [RFC3501]. In addition to that, the Message Storage Server SHALL support the "ACL" IMAP4 extension as defined in [RFC4314], the "URLAUTH" IMAP4 extension as defined in [RFC4467], the "CONDSTORE" IMAP4 extension as defined in [RFC4551], the "ENABLE" IMAP4 extension as defined in [RFC5161], the "QRESYNC" IMAP4 extension as defined in [RFC5162] and the "METADATA" IMAP4 extension as defined in [RFC5464]. Also, the Message Storage Server MAY support the "ANNOTATE" IMAP4 extension as defined in [RFC5257] and the "CONVERT" IMAP4 extension as defined in [RFC5259]

7.1 General Operations

7.1.1 Authenticate Operation

Upon receiving a STARTTLS request, the Message Storage Server SHALL process the request and return a response according to the STARTTLS command as defined in [RFC3501]. The Message Storage Server SHALL at least support TLS and PSK-TLS as defined in [RFC2246] and [OMA-SEC-CF] for the TLS protocol negotiations.

Upon receiving an AUTHENTICATE request, the Message Storage Server SHALL process the request and return a response according to the AUTHENTICATE command as defined in [RFC3501] and then complete the authentication process as defined in [RFC3501].

Upon receiving a LOGIN request, the Message Storage Server SHALL process the request and return a response according to the LOGIN command as defined in [RFC3501].

7.1.2 Set Active Folder Operation

Upon receiving a SELECT request, the Message Storage Server SHALL process the request and return a response according to the SELECT command as defined in [RFC3501].

7.2 Access Control List Operations

7.2.1 Set Access Control List

Upon receiving a SETACL request, the Message Storage Server SHALL process the request and return a response according to the SETACL command as defined in [RFC4314].

7.2.2 Get Access Control List

Upon receiving a GETACL request, the Message Storage Server SHALL process the request and return a response according to the GETACL command as defined in [RFC4314].

7.2.3 Delete Access Control List

Upon receiving a DELETEACL request, the Message Storage Server SHALL process the request and return a response according to the DELETEACL command as defined in [RFC4314].

7.2.4 Access Rights Retrieval Operations

Upon receiving a LISTRIGHTS request the Message Storage Server SHALL process the request and return response according to the descriptions as defined in [RFC4314].

7.3 Objects Operations

7.3.1 Object Store Operation

Upon receiving an APPEND request including the object to store, the Message Storage Server SHALL handle the request according to the APPEND command as defined in [RFC3501] and store the specified object to the end of the specified destination folder.

7.3.1.1 Handling Deferred CPM Message Objects

Upon receiving a request for storing a Deferred CPM Message object that is associated with an expiry time, the Message Storage Server SHALL:

1. handle the request according to the APPEND command as defined in [RFC3501] and store the object to the end of the specified destination folder designated for temporarily holding deferred messages and
2. notify the Message Storage Client of the arrival of the new message.

The stored message object will remain in the designated folder until either it is moved by the Message Storage Client's corresponding commands or it reaches its deferred expiry time and is deleted from the folder.

7.3.2 Object Fetch Operation

Upon receiving a FETCH request with the UID pointing to a stored object, the Message Storage Server SHALL handle the request according to the FETCH or UID FETCH command as defined in [RFC3501].

7.3.3 Object Preview Fetch Operation

Upon receiving a CONVERT or UID CONVERT request with the UID pointing to a stored object of the Message Storage Server, the Message Storage Server SHALL handle the preview request for the supplied format and dimensions according to the CONVERT or UID CONVERT command as defined in [RFC5259]

7.3.4 Object Copy Operation

Upon receiving a COPY request with the UID pointing to a stored object of the Message Storage Server, the Message Storage Server SHALL copy the specified Message(s) or Message History to the end of the specified destination folder according to the COPY command as defined in [RFC3501]

7.3.5 Object Remove Operation

Upon receiving an IMAP STORE command with a UID pointing to the stored object to update the flag list of the object's data to include the "\Deleted" flag, the Message Storage Server SHALL handle the flagging request by setting the stored object's "\Deleted" flag according to the STORE command as defined in [RFC3501]. A session info object can only be removed via session folder removal operation as described in sect. 6.4.4 "Folder Remove Operation".

Upon receiving an EXPUNGE request to remove all stored objects from the CPM User's Message Storage Server that have the "\Deleted" flag set for removal, the Message Storage Server SHALL handle the request to permanently remove these stored objects according to the EXPUNGE command as defined in [RFC3501].

7.4 Metadata Management Operations

The Message Storage Server SHALL support:

- updating the metadata flags defined for IMAP4 message objects in section **Error! Reference source not found.** *"**Error! Reference source not found.**"*,

- fetching the metadata flags defined for IMAP4 message objects in section **Error! Reference source not found.** *"Error! Reference source not found."*,
- updating server metadata annotations defined for IMAP4 mailboxes in section **Error! Reference source not found.** *"Error! Reference source not found."*.
- fetching server metadata annotations defined for IMAP4 mailboxes in section **Error! Reference source not found.** *"Error! Reference source not found."*.

The Message Storage Server MAY support:

- updating message and folder metadata annotations defined for IMAP4 message objects in section **Error! Reference source not found.** *"Error! Reference source not found."*.
- fetching message and folder metadata annotations defined for IMAP4 message objects in section **Error! Reference source not found.** *"Error! Reference source not found."*.

7.4.1 Metadata Update Operation

Upon receiving a STORE request with the UID addressing a stored object of the Message Storage Server, the Message Storage Server SHALL update the metadata flags as indicated in the request according to the STORE command as defined in [RFC3501]. If the STORE request includes 'ANNOTATION' data item and the Message Storage Server supports updating message metadata annotations (i.e. indicates support for the ANNOTATE IMAP4 extension defined in [RFC5257]), the Message Storage Server SHALL handle any metadata annotations specified in the STORE request according to the ANNOTATE IMAP4 extension defined in [RFC5257].

Upon receiving a SETMETADATA request with the name of a folder stored object on the Message Storage Server - or the empty string (i.e. "") in case of server annotations -, the Message Storage Server SHALL apply appropriate metadata updates (i.e. add, remove or modify) according to the SETMETADATA command as defined in [RFC5464].

7.4.2 Metadata Fetch Operation

Upon receiving a FETCH request with the UID addressing one or more stored objects of the Message Storage Server, the Message Storage Server SHALL return the metadata flags according to the FETCH command as defined in [RFC3501]. If the FETCH request includes the 'ANNOTATION' data item name and the Message Storage Server supports fetching message metadata annotations (i.e. indicates support for the ANNOTATE IMAP4 extension defined in [RFC5257]), the Message Storage Server SHALL return any message metadata annotations according to the ANNOTATE IMAP4 extension defined in [RFC5257].

Upon receiving a GETMETADATA request with the name of a folder stored on the Message Storage Server - or the empty string (i.e. "") in case of server annotations -, the Message Storage Server SHALL return the appropriate metadata annotations according to the GETMETADATA command as defined in [RFC5464].

7.5 Folder Operations

7.5.1 Folder Create Operation

Upon receiving a CREATE request, the Message Storage Server SHALL create the folder (e.g., mailbox) with the requested name according to the CREATE command as defined in [RFC3501].

7.5.2 List Folders Operation

Upon receiving a LIST request, the Message Storage Server SHALL determine and return the names of all folders of the CPM User on the Message Storage Server according to the LIST command as defined in [RFC3501].

7.5.3 Folder Move Operation

NOTE: The folder move operation is also used for renaming folders.

Upon receiving a RENAME request with the request-folder name and its new name, the Message Storage Server SHALL rename the indicated folder according to the RENAME command as defined in [RFC3501].

7.5.4 Folder Remove Operation

Upon receiving a DELETE request with the request-folder name, the Message Storage Server SHALL remove the indicated folder according to the DELETE command as defined in [RFC3501].

7.5.5 Folder Search Operation

Upon receiving a SEARCH request, with one or more search key data, the Message Storage Server SHALL handle the request according to the IMAP SEARCH command as defined in [RFC3501].

7.6 Reference Operations

7.6.1 Generate Reference Operation

Upon receiving a GENURLAUTH request, the Message Storage Server SHALL process the request and return a response according to the GENURLAUTH command as defined in [RFC4467].

7.6.2 Fetch by Reference Operation

Upon receiving a URLFETCH request, the Message Storage Server SHALL process the request and return a response according to the URLFETCH command as defined in [RFC4467].

7.7 Message and History Synchronization Operations

[OMA-CPM-SD] gives a description of the synchronization process between the Message Storage Client and the Message Storage Server. This process only consists of operations described above and therefore needs no further explanation in this section.

In addition to that, the Message Storage Server SHALL support the IMAP4 extensions described in [RFC4551], [RFC5161] and [RFC5162] to allow the Message Storage Client to have an optimized and quick synchronization process.

7.8 Notifications Operations

Upon receiving the Message Storage Client's NOTIFY command and/or occurrence of changes in the Message Storage Server's resources, the Message Storage Server SHALL reflect the presence of changes via sending Notification messages. The changes in the Message Storage Server's resources MAY include one or more of the following specific operations per [RFC5423]:

1. message addition and deletion
2. Message flags, e.g., read, clear
3. Access accounting, e.g., login, logout
4. Folder management, e.g., create, delete, rename

The notification messages SHALL be sent by the Message Storage Server to the Message Storage Client upon the occurrence of the following events:

1. Changes in the stored objects due to performing any operations by the Message Storage Server on the stored objects during the period when the CPM Message Storage Client is registered and the notification feature is activated.
2. Changes occurred since the CPM Message Storage Client's last de-registration or CPM User's deactivation of the notification feature.

NOTE: While IMAP IDLE, NOTIFY, or other status change indications serve the purpose of notification, they are not necessarily for notifications only. They are part of the greater synchronization mechanism.

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
n/a	n/a	No prior version

A.2 Draft/Candidate Version 2.0 History

Document Identifier	Date	Sections	Description
Draft Versions OMA-TS-CPM_MessageStorage-V2_0	20 Apr 2012	All	Initial Draft with content from v1.0 document.
	22 Nov 2012	4.2, 5.2. Appendix F and Appendix G added.	CR53R02 and
	18 Dec 2012	5.2.8 and Appendix G	CR78
	04 Apr 2013		Incorporated CR0064R01
	17 Apr 2013		Incorporated CR0076R01
	25 Apr 2013		Editorial changes
	05 May 2013		Incorporated CRs: CR0087, CR0088
	09 May 2013		Incorporated CRs: CR0086
	17 May 2013		Minor editorial fixes
	23 May 2013		Incorporated CRs: CR123R03, CR124R01
Candidate Version OMA-TS-CPM_MessageStorage-V2_0	11 Jun 2013	n/a	Status changed to Candidate by TP TP Ref # OMA-TP-2013-0163R01-INP_Converved_IP_Messaging_2.0_ERP_and_ETR_Candidate_Approval

Appendix B. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [**Error! Reference source not found.**].

B.1 SCR for Message Storage Client

Item	Function	Reference	Requirement
CPM-TS-MS-C-001_M	Authentication – setting up TLS connection	6.1.1	
CPM-TS-MS-C-002_M	Authentication – SASL method	6.1.1	
CPM-TS-MS-C-003_M	Authentication – username/password method	6.1.1	
CPM-TS-MS-C-004_M	Requesting to set an active folder	6.1.2	
CPM-TS-MS-C-005_M	Setting an ACL	6.2.1	
CPM-TS-MS-C-006_M	Getting an ACL	6.2.2	
CPM-TS-MS-C-007-M	Deleting an ACL	6.2.3	
CPM-TS-MS-C-008_M	Retrieving access rights	6.2.4	
CPM-TS-MS-C-009_M	Requesting to store an object(i.e. Message and Message histories)	6.3.1	
CPM-TS-MS-C-010_M	Requesting to fetch message(s)	6.3.2	
CPM-TS-MS-C-011_O	Requesting to fetch a preview object	6.3.3	
CPM-TS-MS-C-012_M	Requesting to copy objects to another folder	6.3.4	
CPM-TS-MS-C-013_M	Requesting to remove objects	6.3.5	
CPM-TS-MS-C-014_M	Requesting to create a folder	6.4.1	
CPM-TS-MS-C-015_M	Requesting to get a list of folders	6.4.2	
CPM-TS-MS-C-016_M	Requesting to rename the folder	6.4.3	
CPM-TS-MS-C-017_M	Requesting to remove the folder	6.4.4	
CPM-TS-MS-C-018_M	Requesting to search contents by a few key words	6.4.5	
CPM-TS-MS-C-019_M	Updating metadata flags of a stored object	6.6.1	
CPM-TS-MS-C-019a_M	Fetching metadata flags of a stored object	6.6.2	
CPM-TS-MS-C-020_O	Updating metadata annotations of a stored object (ANNOTATE extension)	6.6.1	
CPM-TS-MS-C-020a_O	Fetching metadata annotations of a stored	6.6.2	

Item	Function	Reference	Requirement
	object (ANNOTATE extension)		
CPM-TS-MS-C-021_M	Updating metadata annotations of a folder (METADATA extension)	6.6.1	
CPM-TS-MS-C-021a_O	Fetching metadata annotations of a folder (METADATA extension)	6.6	
CPM-TS-MS-C-021b_M	Support CPM METADATA annotations	Appendix H	
CPM-TS-MS-C-021c_M	Support CPM METADATA server annotations	6.6	
CPM-TS-MS-C-022_M	Synchronization	6.6	
CPM-TS-MS-C-023_O	Notifications	6.7	

B.2 SCR for Message Storage Server

Item	Function	Reference	Requirement
CPM-TS-MS-S-001_M	Authentication – setting up TLS connection	7.1.1	
CPM-TS-MS-S-002_M	Authentication – SASL method	7.1.1	
CPM-TS-MS-S-003_M	Authentication – username/password method	7.1.1	
CPM-TS-MS-S-004_M	Requesting to set an active folder	7.1.2	
CPM-TS-MS-S-005_M	Setting an ACL	7.2.1	
CPM-TS-MS-S-006_M	Getting an ACL	7.2.2	
CPM-TS-MS-S-007-M	Deleting an ACL	7.2.3	
CPM-TS-MS-S-008_M	Retrieving access rights	7.2.4	
CPM-TS-MS-S-009_M	Storing objects to a folder	7.3.1	
CPM-TS-MS-S-010_M	Sending objects	7.3.2	
CPM-TS-MS-S-011_O	Converting the object as requested	7.3.3	
CPM-TS-MS-S-012_M	Copying the objects to the another folder	7.3.4	
CPM-TS-MS-S-013_M	Removing the object	7.3.5	
CPM-TS-MS-S-014_M	Creating a folder with a requested folder name	7.4.1	
CPM-TS-MS-S-015_M	Sending the name of all folders	7.4.2	
CPM-TS-MS-S-016_M	Renaming the folder	7.4.3	
CPM-TS-MS-S-017_M	Deleting the folder	7.4.4	
CPM-TS-MS-S-018_M	Searching objects by a few keywords	7.4.5	

Item	Function	Reference	Requirement
CPM-TS-MS-S-019_M	Updating metadata flags for a stored object	7.4.1	
CPM-TS-MS-S-019a_M	Fetching metadata flags for a stored object	7.4.2	
CPM-TS-MS-S-020_O	Updating metadata annotations of a stored object (ANNOTATE extension)	7.4.1	
CPM-TS-MS-S-020a_O	Fetching metadata annotations of a stored object (ANNOTATE extension)	7.4.2	
CPM-TS-MS-S-021_M	Updating metadata annotations of a folder (METADATA extension)	7.4.1	
CPM-TS-MS-S-021a_O	Fetching metadata annotations of a folder (METADATA extension)	7.4	
CPM-TS-MS-S-021b_M	Support CPM METADATA annotations	Appendix H	
CPM-TS-MS-S-021c_M	Support CPM METADATA server annotations	7.4	
CPM-TS-MS-S-022_M	Processing the synchronization	7.6	
CPM-TS-MS-S-023_O	Notifications	7.7	

Appendix C. CPM-defined IMAP Flag Extensions

C.1 \read-report-sent

\ read-report-sent is a CPM-defined IMAP flag extension associated with a Message Object stored in Message Storage Server.

The definition of the flag is:

```
\read-report-sent
   Read report is sent
```

Formal syntax of \read-report-sent is:

```
flag-extension = "\read-report-sent"
```

Example:

```
C: A003 STORE 2:4 +FLAGS (\read-report-sent)
```

\ read-report-sent status flag is visible to the Message Storage Client but is masked to the CPM User. This flag is used to record whether a read report has been sent or not for a message object in the Message Storage Server.

Appendix D. Example of Session History Folder

Below is an example of a 1-1 session history folder with the following content:

- 1 CPM Chat Message.
- 1 audio stream.

```
From: John Doe <jdoe@machine.example>
Date: Fri, 21 Nov 1997 09:55:06 -0600
Subject: the weather will be fine today
Conversation-ID: f81d4fae-7dec-11d0-a765-00a0c91e6bf6
Contribution-ID: abcdef-1234-5678-90ab-cdef01234567
InReplyTo-Contribution-ID: 01234567-89ab-cdef-0123-456789abcdef
Content-type: multipart/related;boundary=cpm; type="Application/X-CPM-Session"

--cpm
Content-Type: Application/X-CPM-Session

<session-type>1-1</session-type>
<invited-participants>jdoe@machine.example; sip:alice@example.com</invited-
participants>

<media-object>
      <cid>cid:<1234@example.com></cid>
      <sdp>
        a=sendonly
        m=audio 49170 RTP/AVP 0 97
      </sdp>
</media-object>

--cpm
Content-type: Message/CPIM
From: John Doe <jdoe@machine.example>
To: Alice <sip:alice@example.com>
DateTime: 2000-12-13T13:40:00-08:00
Content-type: text/plain; charset=utf-8

Here is the text of my message.

--cpm
Content-Type: audio/basic
Content-Transfer-Encoding: base64
Content-ID:<1234@example.com>

       ... base64-encoded 8000 Hz single-channel
           mu-law-format audio data goes here ...
```

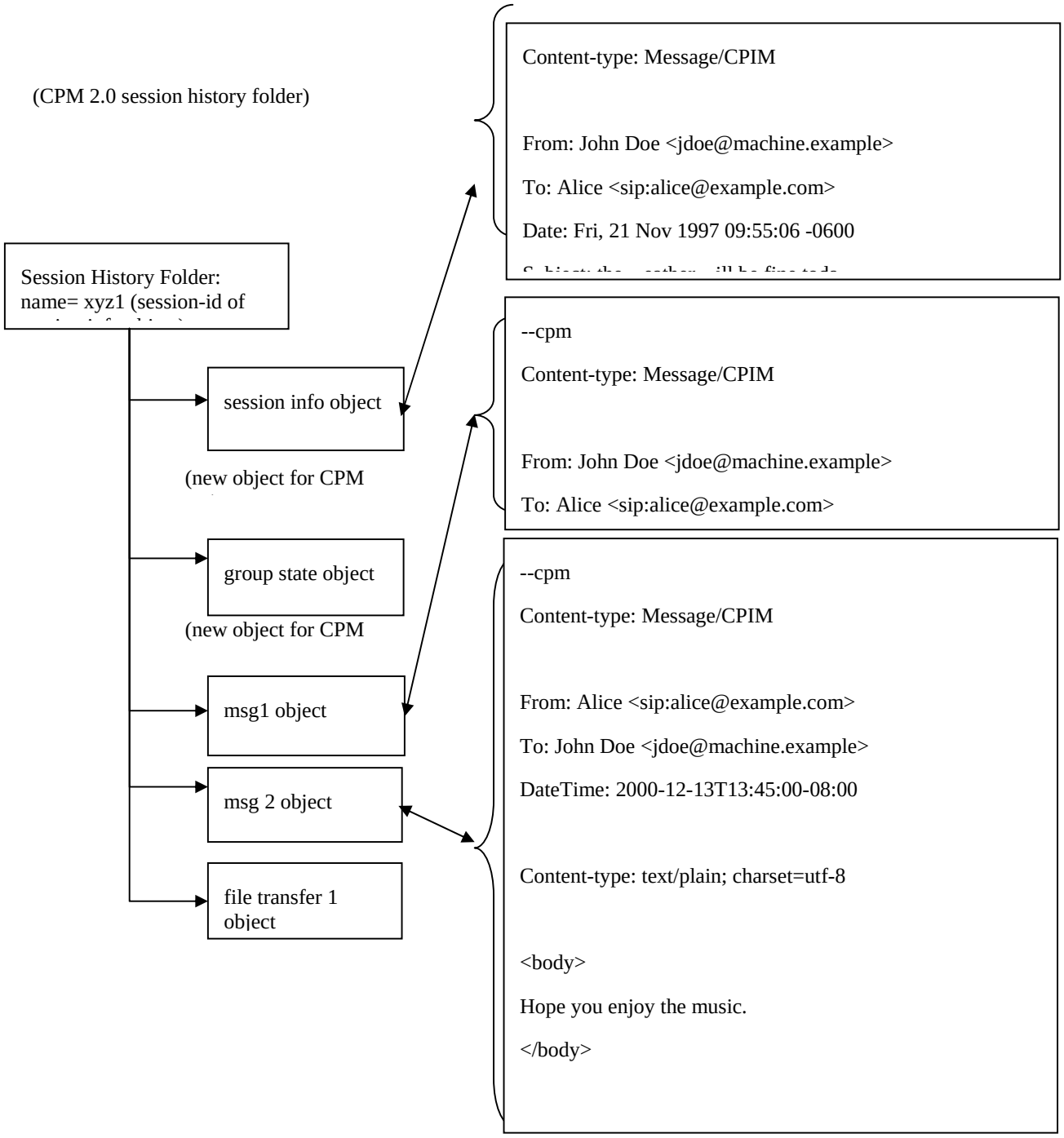
Appendix E. Example of File Transfer History Object

Below is an example of a 1-1 file transfer history object in which one file was received.

```
From: John Doe <jdoe@machine.example>
Date: Fri, 21 Nov 1997 09:55:06 -0600
Conversation-ID: f81d4fae-7dec-11d0-a765-00a0c91e6bf6
Contribution-ID: abcdef-1234-5678-90ab-cdef01234567
InReplyTo-Contribution-ID: 01234567-89ab-cdef-0123-456789abcdef
Content-type: multipart/related;boundary=cpm; type=”Application/X-CPM-File-Transfer”

--cpm
Content-Type: Application/X-CPM-File-Transfer

<file-transfer-type>1-1</file-transfer-type>
<invited-participants>jdoe@machine.example; sip:alice@example.com</invited-
participants>

<file-object>
      <cid>cid:<1234@example.com></cid>
      <sdp>
        i=This is my latest picture
        a=sendonly
        a=file-selector:name:"My picture.jpg" type:image/jpeg size:4092
        a=file-disposition:render
        a=file-date:creation:"Mon, 15 May 2006 15:01:31 +0300"
      </sdp>
</file-object>

--cpm
Content-Type: image/jpeg
Content-Transfer-Encoding: binary
Content-ID: <1234@example.com>

   ... My picture.jpg...
```

Appendix F. Storage of CPM Session

The illustration below includes the main header fields in the SIP, MSRP and CPIM messages. While other header fields may exist, they are not shown below for simplification.



F.1 Illustration of content of the CPM Session History folder

Appendix G. Group State Object schema

The recommended schema for the Group State Object is described below:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:xml="http://www.w3.org/XML/1998/namespace"
        elementFormDefault="qualified">
    <xs:import namespace="http://www.w3.org/XML/1998/namespace"
            schemaLocation="http://www.w3.org/2009/01/xml.xsd"/>
    <xs:element name="groupstate">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="participant" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:attribute name="name" type="xs:string" use="required"/>
                        <xs:attribute name="comm-addr" type="xs:anyURI"
                        use="required"/>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
            <xs:attribute name="contributionid" type="xs:string" use="required"/>
            <xs:attribute name="lastfocussessionid" type="xs:string" use="required"/>
            <xs:attribute name="timestamp" type="xs:datetime" use="required"/>
            <xs:attribute name="group-type" type="xs:string" use="required"/>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

G.1: Group State Object schema

Appendix H. CPM METADATA annotations (Normative)

This section contains the metadata entries defined by the OMA CPM Enabler.

Each CPM metadata entry is listed in **Error! Reference source not found.** along with the following information:

- the valid namespace(s),
- whether the metadata entry can be used as a server and/or a mailbox annotation,
- its purpose, and;
- a reference to the metadata entry format description.

Entry name	Valid namespace(s)	Server annotation	Mailbox annotation	Purpose	Format
DefaultFolderLocation	/shared/OMNA/OMACPM20	Y	N	Indicates the location of the "default" system folder; see section "5.5.1.1 Folders" in [OMA-CPM-SD].	H.1.1

Table 4: CPM METADATA annotation reference sheet

H.1 Metadata entry formats

While all metadata entries contain strings in general, it is important to define the format of contents of these strings. This section contains a sub-section describing the format of each metadata entry defined in **Error! Reference source not found.**.

H.1.1 DefaultFolderLocation

The metadata entry MUST contain a fully qualified mailbox name suffixed with the server's hierarchy separator character (i.e. a parameter for the SELECT command).

For example:

```
/INBOX/CPM2/-=default=-
```

Appendix I. Representation of CPM Conversations in the CPM Message Store (Informative)

This section contains examples of conversations that may take place between CPM Users, focusing on illustrating how the communication representation corresponds to the storage representation. The examples illustrate basic scenarios where a conversation takes place using the same channel, more advanced scenarios where a conversation takes place using different channels but only one channel at a time. The last example included is the most advanced scenario where a single conversation takes place using various channel in parallel.

The basis for these examples is provided by the original high-level diagrams in section 4 of the CPM Requirements document. [OMA-CPM-RD].

Please note that these examples do not indicate directions. It does not matter if the message or file was sent or received – it is reflected in the storage just the same.

I.1 Standalone exchanges

The following figure illustrates a new conversation, taking place solely using standalone exchanges such as standalone messages (pager mode, large message mode) and file transfers. Each CPM Standalone Message corresponds to a single message object and each CPM File Transfer corresponds to a single file transfer history object.

A watchful observer will notice that the InReplyTo-Contribution-ID is not always included. However, when it is included, it establishes a link between two distinct contributions (e.g. a standalone message and a file transfer in the example below).

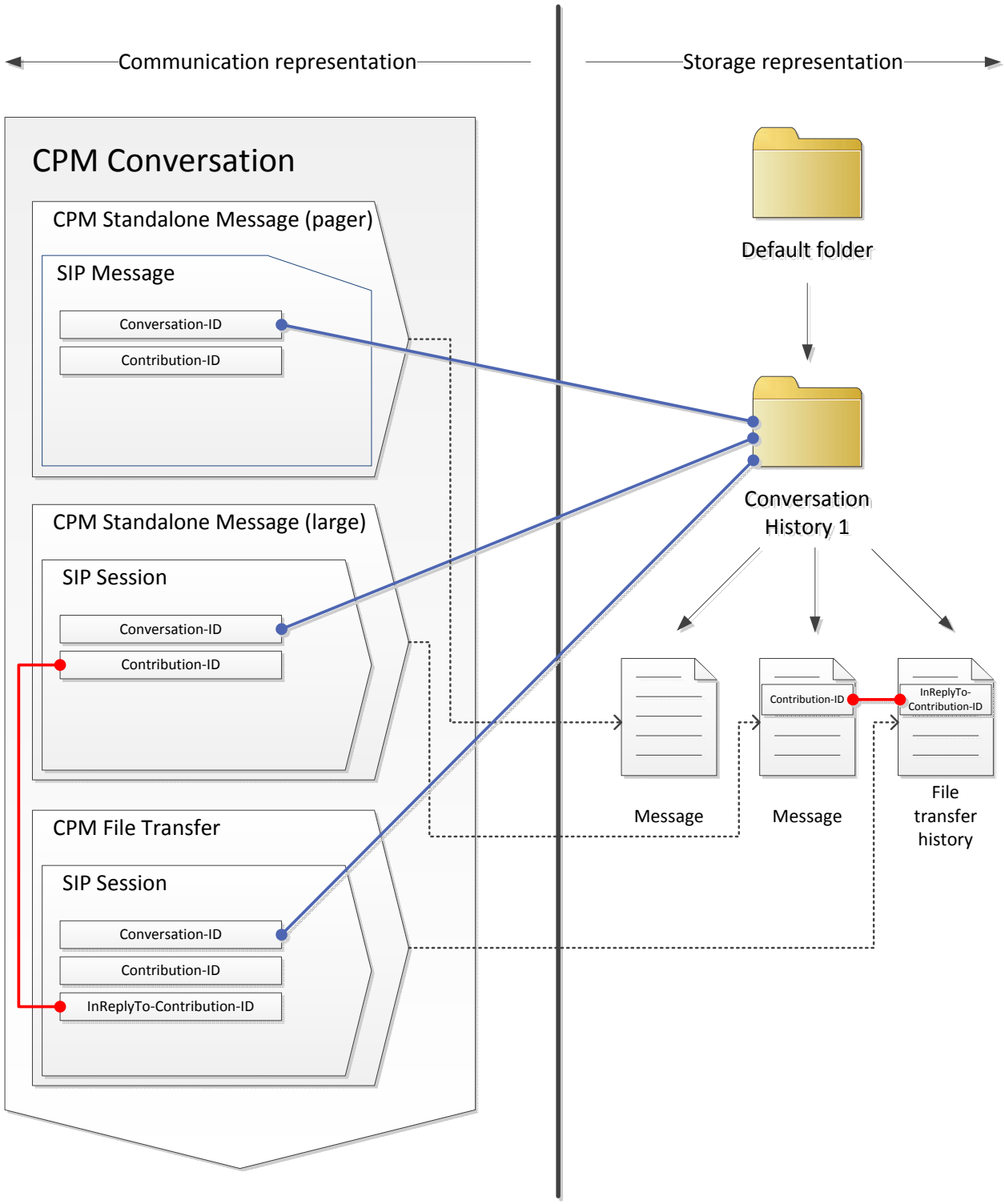


Figure 1: Example of storage representation; standalone exchanges

I.2 Chat exchanges

The following figure illustrates a new conversation, taking place solely using chat exchanges (applies to 1-1 chat or group chat) such as chat message and file transfers. At the beginning of the chat session, the session info object is stored into the session history folder. In this example all chat messages and file transfers are stored at the end of the chat session. Additionally, the Group State Object is stored. Each chat message corresponds to a single message object and each CPM File Transfer corresponds to a single CPM File Transfer History Object.

A watchful observer will notice that the Contribution-ID links a chat session and the file transfers in the same CPM Session together.

Note: while this example is valid for long-lived chat sessions as well, the point of the long-lived chat sessions is better illustrated on Figure 3.

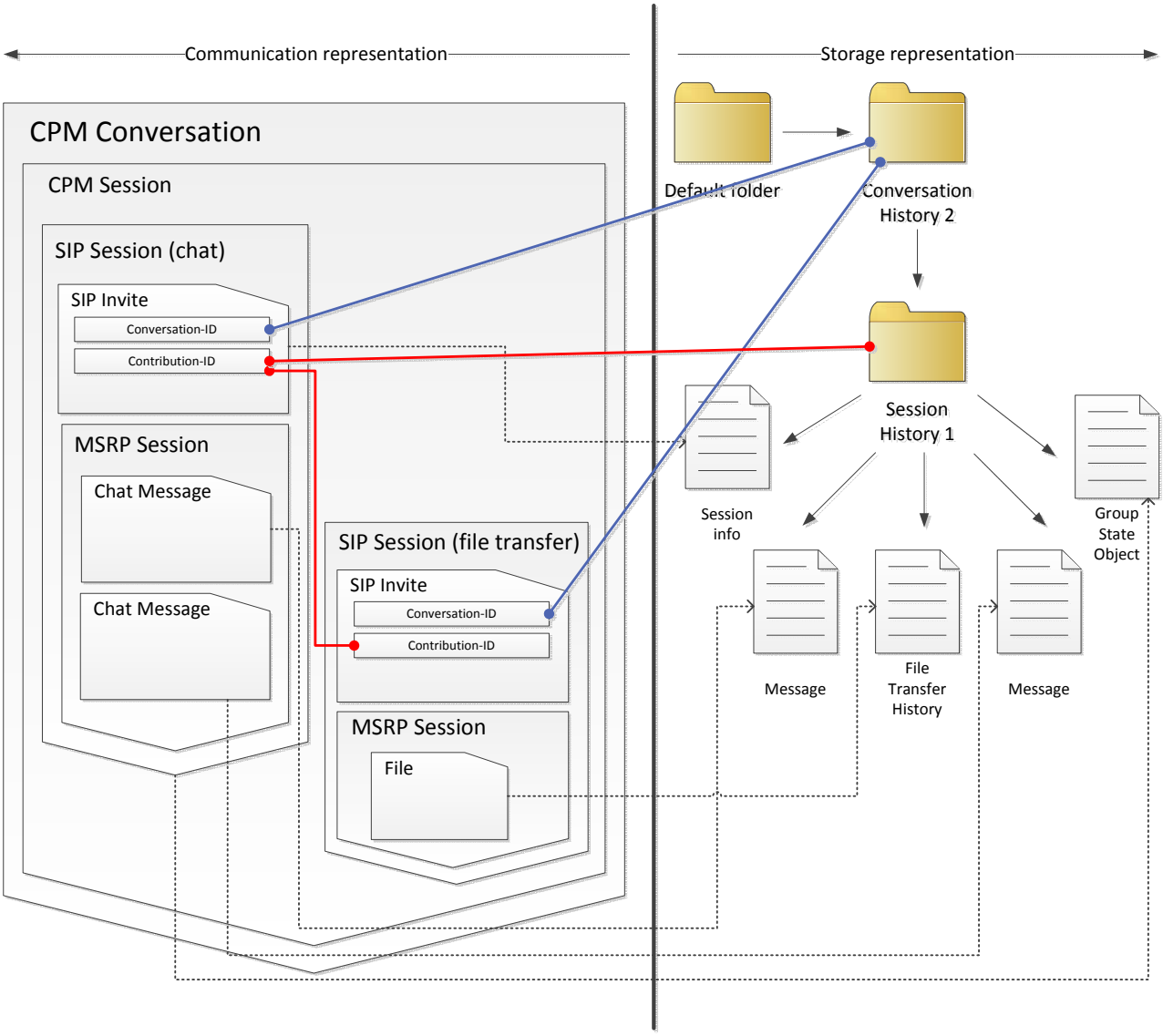


Figure 2: Example of storage representation; chat exchanges

I.3 Long-lived chat exchanges

The following figure illustrates a new long-lived group chat conversation, taking place solely using long-lived group chat exchanges and file transfers. At the beginning of the long-lived group chat session, the session info object is stored into the session history folder. In this example, all chat messages and file transfers are stored at the end of each SIP session. Additionally, the Group State Object is stored. Each chat message corresponds to a single message object and each CPM File Transfer corresponds to a single CPM File Transfer History Object.

After some time, the long-lived group chat session is restarted, using the Conversation-ID and Contribution-ID to establish the link. The session info object is already stored for this CPM Session; hence a new session info object is not stored. In this example, all chat messages and file transfers are stored at the end of each SIP session. Additionally, a second Group State Object is stored). Each chat message corresponds to a single message object and each CPM File Transfer corresponds to a single CPM File Transfer History Object.

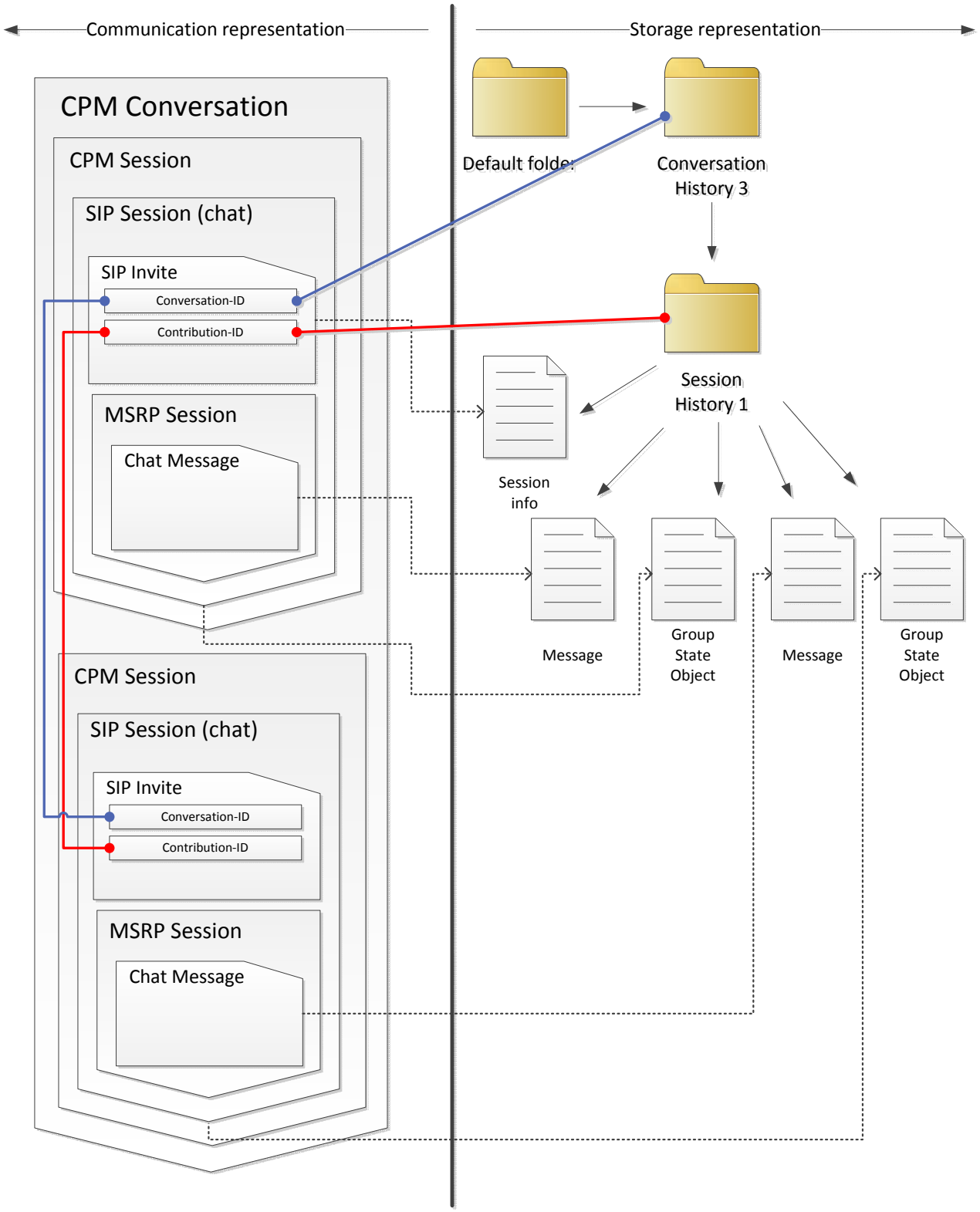


Figure 3: Example of storage representation; long lived chat exchanges

I.4 Standalone to 1-1 chat

The following figure illustrates a new conversation, that takes place initially using standalone exchanges and later on it takes place using a 1-1 chat, which is an example of a conversation that begun on a mobile device and continued over a desktop computer.

The 1-1 chat invitation could contain an InReplyTo-Contribution-ID referencing any of the Standalone Messages in the conversation, but it would not make any difference in practice as the link is established solely using the Conversation-ID, which is the same in all Standalone Messages and the 1-1 chat. If a different Conversation-ID was used in the 1-1 chat, a new conversation would be started instead, resulting in a new Conversation History, as shown in Figure 2.

There is nothing preventing the users from having the same conversation using both standalone exchanges and 1-1 chat in parallel: the messages would be reflected into storage exactly the same way. However, the timestamp of the stored messages would reveal that these two did, in fact, happen in parallel.

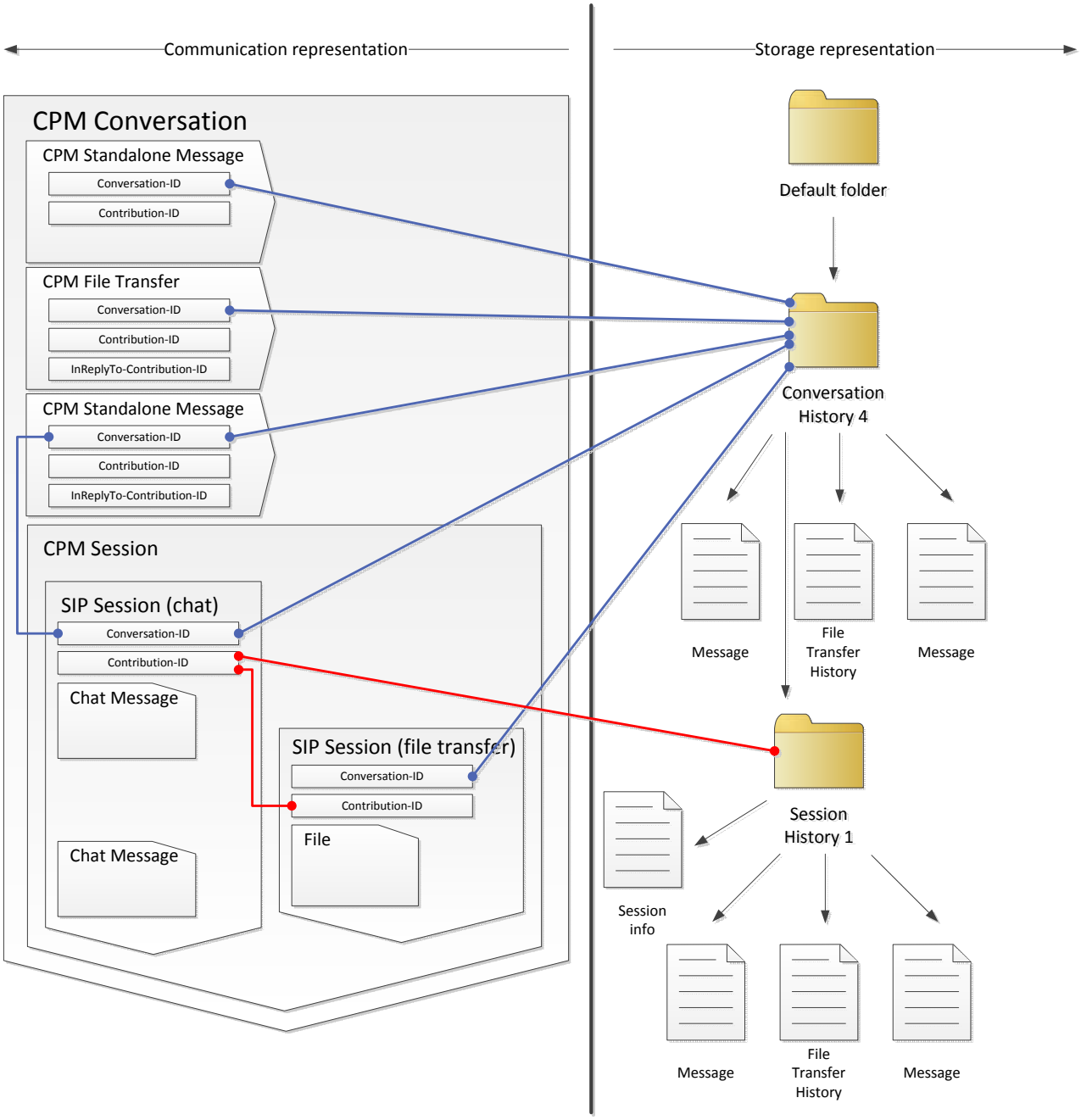


Figure 4: Example of storage representation; standalone to 1-1 chat

I.5 Extending 1-1 chat to group chat

The following figure illustrates a new conversation, that takes place initially using 1-1 chat and later on it is extended to a group chat by replacing the 1-1 chat with a group chat using the "*Extending a CPM 1-1 Session to a CPM Group Session*" procedure.

The link between the 1-1 chat and the group chat is established solely using the Conversation-ID and the Session-Replaces header field. If the Session-Replaces header field were missing, it would mean that a new, parallel CPM Group Chat Session

is starting (i.e. the 1-1 chat session is retained) in the same conversation – and consequently, in the same Conversation History, as shown in Figure 6.

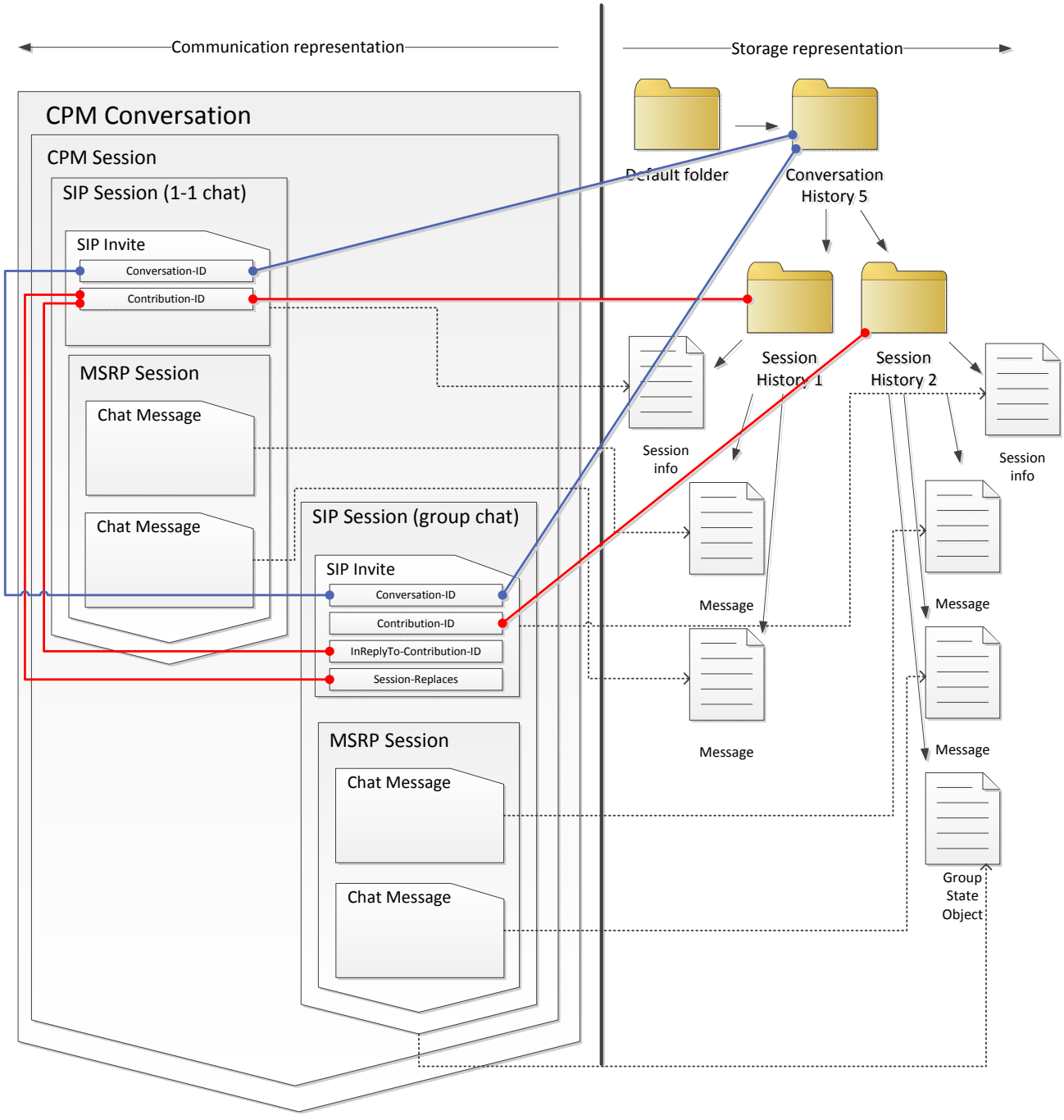


Figure 5: Example of storage representation; extending 1-1 chat to group chat

I.6 Conversation via parallel channels

The following figure illustrates a new conversation that takes place using standalone exchanges, a 1-1 chat and a long-lived group chat in parallel.

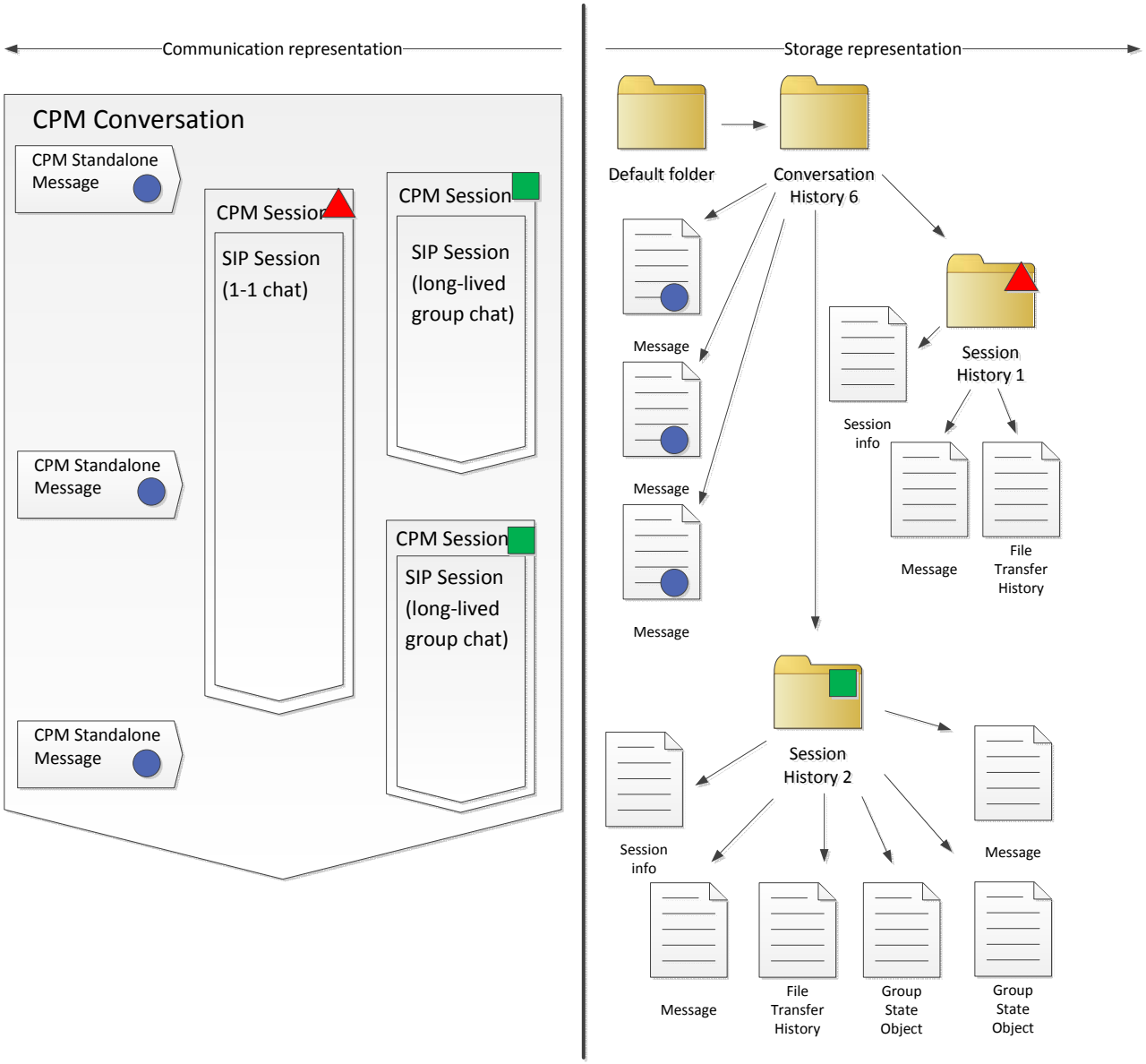


Figure 6: Example of storage representation; conversation via parallel channels