



Converged Address Book (CAB) Specification

Candidate Version 1.0 – 19 Oct 2010

Open Mobile Alliance
OMA-TS-CAB-V1_0-20101019-C

Use of this document is subject to all of the terms and conditions of the Use Agreement located at <http://www.openmobilealliance.org/UseAgreement.html>.

Unless this document is clearly designated as an approved specification, this document is a work in process, is not an approved Open Mobile Alliance™ specification, and is subject to revision or removal without notice.

You may use this document or any part of the document for internal or educational purposes only, provided you do not modify, edit or take out of context the information in this document in any manner. Information contained in this document may be used, at your sole risk, for any purposes. You may not use this document in any other manner without the prior written permission of the Open Mobile Alliance. The Open Mobile Alliance authorizes you to copy this document, provided that you retain all copyright and other proprietary notices contained in the original materials on any copies of the materials and that you comply strictly with these terms. This copyright permission does not constitute an endorsement of the products or services. The Open Mobile Alliance assumes no responsibility for errors or omissions in this document.

Each Open Mobile Alliance member has agreed to use reasonable endeavors to inform the Open Mobile Alliance in a timely manner of Essential IPR as it becomes aware that the Essential IPR is related to the prepared or published specification. However, the members do not have an obligation to conduct IPR searches. The declared Essential IPR is publicly available to members and non-members of the Open Mobile Alliance and may be found on the “OMA IPR Declarations” list at <http://www.openmobilealliance.org/ipr.html>. The Open Mobile Alliance has not conducted an independent IPR review of this document and the information contained herein, and makes no representations or warranties regarding third party IPR, including without limitation patents, copyrights or trade secret rights. This document may contain inventions for which you must obtain licenses from third parties before making, using or selling the inventions. Defined terms above are set forth in the schedule to the Open Mobile Alliance Application Form.

NO REPRESENTATIONS OR WARRANTIES (WHETHER EXPRESS OR IMPLIED) ARE MADE BY THE OPEN MOBILE ALLIANCE OR ANY OPEN MOBILE ALLIANCE MEMBER OR ITS AFFILIATES REGARDING ANY OF THE IPR'S REPRESENTED ON THE “OMA IPR DECLARATIONS” LIST, INCLUDING, BUT NOT LIMITED TO THE ACCURACY, COMPLETENESS, VALIDITY OR RELEVANCE OF THE INFORMATION OR WHETHER OR NOT SUCH RIGHTS ARE ESSENTIAL OR NON-ESSENTIAL.

THE OPEN MOBILE ALLIANCE IS NOT LIABLE FOR AND HEREBY DISCLAIMS ANY DIRECT, INDIRECT, PUNITIVE, SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR EXEMPLARY DAMAGES ARISING OUT OF OR IN CONNECTION WITH THE USE OF DOCUMENTS AND THE INFORMATION CONTAINED IN THE DOCUMENTS.

© 2010 Open Mobile Alliance Ltd. All Rights Reserved.

Used with the permission of the Open Mobile Alliance Ltd. under the terms set forth above.

Contents

1.	SCOPE	5
2.	REFERENCES	6
2.1	NORMATIVE REFERENCES	6
2.2	INFORMATIVE REFERENCES	7
3.	TERMINOLOGY AND CONVENTIONS	8
3.1	CONVENTIONS	8
3.2	DEFINITIONS	8
3.3	ABBREVIATIONS	9
4.	INTRODUCTION (INFORMATIVE)	10
4.1	VERSION 1.0	10
5.	PROCEDURES AT CAB SERVER	11
5.1	AB SYNCHRONIZATION FUNCTION	11
5.2	CONTACT SUBSCRIPTION FUNCTION	11
5.3	CONTACT SHARE FUNCTION	13
5.3.1	Procedures at the Originating Side.....	13
5.3.2	Procedures at the Receiving Side.....	14
5.4	INTERWORKING FUNCTION	14
5.4.1	Import from Non-CAB AB Systems.....	14
5.4.2	Contact Search with External Directories	15
5.4.3	Format Adaptation	16
5.5	XDM AGENT	24
5.6	CONTACT STATUS FUNCTION	25
5.6.1	Management of Contact Status in AB.....	25
5.6.2	Incoming Contact Subscription Requests	26
5.6.3	Contact Added	26
5.7	APPLYING CAB USER PREFERENCES	28
6.	PROCEDURES AT CAB CLIENT	29
6.1	ADDRESS BOOK MANAGEMENT AND SYNCHRONIZATION	29
6.2	PERSONAL CONTACT CARD (PCC) MANAGEMENT	29
6.3	CAB USER PREFERENCES MANAGEMENT	29
6.4	IMPORT NON-CAB ADDRESS BOOK	29
6.5	CONTACT SHARE	30
6.6	CONTACT SUBSCRIPTION	30
6.7	ACCESS PERMISSIONS MANAGEMENT	30
6.7.1	Access Permissions Document for Address Book	31
6.7.2	Access Permissions Document for PCC	31
6.8	UPP DIRECTORY DOCUMENT MANAGEMENT	31
6.9	XDM PREFERENCES MANAGEMENT	31
6.9.1	XDM Preferences Document for Address Book.....	31
6.9.2	XDM Preferences Document for PCC.....	31
6.10	SUBSCRIPTION TO CAB XML DOCUMENTS CHANGES	31
6.10.1	Subscriptions to CAB User's own PCC document changes	32
6.10.2	Subscriptions to CAB User's own CAB User Preferences document changes.....	32
6.10.3	Subscriptions to CAB User's own CAB Feature Handler document changes	32
6.10.4	Subscription to CAB User's own CAB XML documents using Subscription Proxy.....	32
6.11	CONTACT SEARCH	33
6.12	CONTACT STATUS MANAGEMENT	33
6.13	AUTHENTICATION	34
6.13.1	Authentication for AB Synchronization.....	34
6.13.2	Authentication for XML Document Management.....	34
7.	CAB XDMS	35

8. CAB MANAGEMENT OBJECT	36
APPENDIX A. CHANGE HISTORY (INFORMATIVE)	37
A.1 APPROVED VERSION HISTORY	37
A.2 DRAFT/CANDIDATE VERSION 1.0 HISTORY	37
APPENDIX B. STATIC CONFORMANCE REQUIREMENTS (NORMATIVE)	41
B.1 SCR FOR CAB CLIENT	41
B.2 SCR FOR CAB SERVER	43
APPENDIX C. FLOWS (INFORMATIVE)	47
C.1 CONTACT SEARCH	47
C.1.1 Contact Search - PCC	47
C.1.2 Contact Search - AB	48
C.1.3 Contact Search – External Directories	49
C.2 IMPORT FROM NON-CAB ADDRESS BOOK SYSTEMS	50
C.3 SAMPLE XCAP FLOWS FOR MANAGEMENT OF CAB XML DOCUMENTS	51
C.3.1 XCAP operations on CAB XML documents	51
C.4 SAMPLE CONTACT SHARE FLOWS	55
C.4.1 Contact Share towards a CAB User	56
C.4.2 Contact Share towards a Non CAB User	67
C.5 CONTACT SUBSCRIPTION FLOWS	69
C.5.1 CAB Server Subscribes to contacts PCCs using Subscription Proxy	69
C.6 MANAGING ADDRESS BOOK FLOW	71
C.6.1 CAB Client Address Book Modifications and Synchronization	71
C.6.2 Address Book Modifications from Network	72
APPENDIX D. X-CAB PARAMETERS AND FIELDS DEFINITIONS (INFORMATIVE)	74

Figures

Figure 1: Contact added data details	27
Figure 2: CAB Client manipulating a CAB XML document	51
Figure 3 : Flows of AB Forwarding in the originating side towards a CAB user	56
Figure 4: Flows of PCC forwarding in the originating side towards a CAB user	58
Figure 5: Flows of AB forwarding in the terminating side towards a CAB user	60
Figure 6: Flows of AB forwarding in the terminating side towards a CAB user	62
Figure 7: Flows of PCC forwarding in the terminating side towards a CAB user	64
Figure 8: Flows of PCC forwarding in the terminating side towards a CAB user	66
Figure 9 : Flows of Contact Share towards a non CAB user	68
Figure 10 : Contact Subscription flow using the Subscription Proxy	69
Figure 11 : CAB Client Address Book Modifications and Synchronization Flow	71
Figure 12 : Address Book Modifications from Network	72

Tables

Table 1: Mapping between CAB Format and Legacy Format(s)	24
---	-----------

1. Scope

This document provides the Technical Specification of the CAB Enabler to fulfil the requirements outlined in the Converged Address Book Requirements document [CAB RD] for CAB V1.0 and in compliance to the architecture described in Converged Address Book Architecture document [CAB AD]. The Technical Specification provides the definition of data elements of the CAB Enabler and the description of the procedures for the features supported by the CAB Enabler.

Additionally, this document describes a set of detailed flows for the functionalities of the CAB Enabler. These flows explain the system concepts in a graphical manner and describe the relationships between the functional components of the CAB Enabler architecture.

2. References

2.1 Normative References

- [CAB AD] “Converged Address Book Architecture”, Version 1.0, Open Mobile Alliance™, OMA-AD-CAB-V1_0, URL:<http://www.openmobilealliance.org/>
- [CAB RD] “Converged Address Book Requirements”, Version 1.0, Open Mobile Alliance™, OMA-RD-CAB-V1_0, URL:<http://www.openmobilealliance.org/>
- [CAB MO] "Converged Address Book Management Object", Version 1.0, Open Mobile Alliance™, OMA-TS-CAB-MO-V1_0, URL:<http://www.openmobilealliance.org/>
- [CAB XDMS] "Converged Address Book XDM Specification", Version 1.0, Open Mobile Alliance™, OMA-TS-CAB-XDMS-V1_0, URL:<http://www.openmobilealliance.org/>
- [OMA CPM CONV FCT TS] “CPM Conversation Functions”, Version 1.0, Open Mobile Alliance™, OMA-TS-CPM_Conv_Fnct-V1_0, URL:<http://www.openmobilealliance.org/>
- [OMA CPM IWF TS] “CPM Interworking”, Version 1.0, Open Mobile Alliance™, OMA-TS-CPM_Interworking-V1_0, URL:<http://www.openmobilealliance.org/>
- [OMA DS] “SyncML Representation Protocol, Data Synchronization Usage”, Version 1.2, Open Mobile Alliance™, OMA-TS-DS_DataSyncRep-V1_2, URL: <http://www.openmobilealliance.org/>
- [OMA DS DevInf] “OMA DS Device Information”, Version 1.2, Open Mobile Alliance™, OMA-TS-DS_DevInf-V1_2, URL: <http://www.openmobilealliance.org/>
- [OMA DS Pro] “DS Protocol”, Version 1.2.2, Open Mobile Alliance™, OMA-TS-DS_Protocol-V1_2_2, URL: <http://www.openmobilealliance.org/>
- [OMA XDM AD] “XML Document Management Architecture”, Version 2.1, Open Mobile Alliance™, OMA-AD-XDM-V2_1, URL:<http://www.openmobilealliance.org/>
- [OMA XDM Core] “XML Document Management Specification”, Version 2.1, Open Mobile Alliance™, OMA-TS-XDM_Core-V2_1, URL:<http://www.openmobilealliance.org/>
- [OMA XDM RD] “XML Document Management Requirements”, Version 2.1, Open Mobile Alliance™, OMA-RD-XDM-V2_1, URL:<http://www.openmobilealliance.org/>
- [OMA XDM UPP] “UPP Directory XDM Specification”; Open Mobile Alliance™, OMA-TS-XDM_UPP_Directory-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>
- [RFC2425] A MIME Content-Type for Directory Information. T. Howes, M. Smith, F. Dawson. September 1998. URL: <http://www.ietf.org/rfc/rfc2425.txt>
- [RFC2426] vCard MIME Directory Profile. F. Dawson, T. Howes. September 1998. URL: <http://www.ietf.org/rfc/rfc2426.txt>
- [RFC4660] IETF RFC 4660 “Functional Description of Event Notification Filtering”, H. Khartabil et al., September 2006 URL: <http://www.ietf.org/rfc/rfc4660.txt>
- [RFC4661] IETF RFC 4661 “An Extensible Markup Language (XML)-Based Format for Event Notification Filtering”, H. Khartabil et al., September 2006 URL: <http://www.ietf.org/rfc/rfc4661.txt>
- [SCRRULES] “SCR Rules and Procedures”, Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:<http://www.openmobilealliance.org/>
- [vCard2.1] “vCard The Electronic Business Card Version 2.1”, A versit Consortium Specification, September 18, 1996 URL: <http://www.imc.org/pdi/vcard-21.doc>
- [XSD cab pcc] “XML Schema Definition: CAB Personal Contact Card document”, Version 1.0, Open Mobile Alliance™, OMA-SUP-XSD_cab_pcc-V1_0, URL: <http://www.openmobilealliance.org/>

[XSD extSearch] “XML Schema Definition: CAB Search Document extension for External Directories”, Version 1.0, Open Mobile Alliance™, OMA-SUP-XSD_cab_search_external_directories-V1_0, URL: <http://www.openmobilealliance.org/>

2.2 Informative References

[OMADICT] “Dictionary for OMA Specifications”, Version 2.8, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_8, 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

Access Permissions	See [OMA XDM RD].
Aggregation Proxy	See [OMA XDM AD].
Address Book	See Converged Address Book in [CAB RD].
Application Unique ID	See [OMA XDM Core].
Application Usage	See [OMA XDM Core].
CAB Client	See [CAB AD].
CAB Format	An XML format containing a set of attributes representing contact information.
CAB User	See [CAB RD].
Contact Share	See [CAB RD].
Contact Status	See [CAB AD].
Contact Subscription	See [CAB RD].
Contact View	See [CAB RD].
DS Client	See Client [OMA DS].
DS Server	See Server [OMA DS].
External Directories	See [CAB AD].
History Information	See [OMA XDM AD].
Legacy Formats	See [CAB RD].
Limited XQuery over HTTP	See [OMA XDM AD].
Management Object	See [OMA DM TND].
Non-CAB Address Book System	See [CAB AD].
Personal Contact Card	See [CAB RD].
Published Contact Card	See [CAB RD].
UPP Directory	Use definition from [OMA XDM RD].
UPP Directory Document	Use definition from [OMA XDM RD].
Users Tree	See [OMA XDM Core].
XDCP Request	See [OMA XDM Core].
XDM Agent	See [OMA XDM AD].
XDM Client	See XDMC in [OMA XDM AD].

XDMS See [OMA XDM AD].

3.3 Abbreviations

AB	Address Book
AUID	Application Unique ID
DM	Device Management
DS	Data Synchronization
DTD	Document Type Definition
HTTP	Hyper Text Transfer Protocol
IP	Internet Protocol
OMA	Open Mobile Alliance
PCC	Personal Contact Card
SIP	Session Initiation Protocol
UPP	User Preferences Profile
XCAP	XML Configuration Access Protocol
XDCP	XDM Command Protocol
XDM	XML Document Management
XDMC	XML Document Management Client
XDMS	XML Document Management Server
XML	eXtensible Markup Language
XQuery	XML Query
XUI	XCAP User Identifier

4. Introduction (Informative)

The CAB Technical Specification provides the CAB User with the features described in sub clause 4.1. The CAB Technical Specification utilizes the technologies of data management and synchronization [OMA DS Pro] and XML document management [OMA XDM Core] to fulfil CAB requirements [CAB RD] and is based on CAB architecture described in the [CAB AD].

4.1 Version 1.0

CAB Technical Specification version 1.0 supports the following features as described in [CAB AD] sub clause 4.1:

- 1) Management of AB
- 2) Management of PCC
- 3) Contact Subscription
- 4) Notifications and status information
- 5) Sharing contact information
- 6) Searching contact information
- 7) Exposure of CAB information to external Enablers
- 8) Interaction with Non-CAB Address Book systems

5. Procedures at CAB Server

The CAB Server SHALL support the procedures described in the following sub `clauses.

5.1 AB Synchronization Function

The CAB Server SHALL use OMA DS Protocol [OMA DS Pro] as specified in CAB-01 interface [CAB AD] to synchronize the data modifications in the AB or subsets of AB with the CAB Client. .

The CAB Server SHALL support AB CAB Format for AB synchronization with CAB Client. In addition to the AB CAB Format, the CAB Server SHALL support the synchronization of Legacy Formats vCard 2.1 [vCard 2.1], and vCard 3.0 [RFC2425] [RFC2426]. Uses of vCard 2.1 and vCard 3.0 in synchronization are specified in

- sub clauses 5.3.26 to 5.3.29 of [OMA DS DevInf] in terms of properties and parameters, and
- sub clause 8 of [OMA DS] in terms of MIME type.

For synchronization of vCard 2.1 and vCard 3.0 formats, format adaptation is required. The CAB server SHALL perform the mapping as specified in “Format Adaptation” sub clause 5.4.3.

Note: The use of the Legacy Formats will result in reduced support of the contact information provided by the AB Application Usage.

The CAB Server SHALL act as a XDM Agent (see sub clause 5.5 “XDM Agent”) to manage (e.g. retrieve, create, modify, delete) AB XML document.

After successfully receiving address book changes from the CAB Client as part of the synchronization sequence, and if there is resulting data to be updated in the AB Application Usage, the CAB Server SHALL update the AB Application Usage before completing the procedure of OMA DS synchronization, i.e.. prior to sending the [OMA DS Pro] message (OMA DS Pkg #6) with the map acknowledgement to the CAB Client, or if OMA DS Pkg #5 and OMA DS Pkg #6 are not required, prior to sending OMA DS Pkg #4.

The CAB Server SHALL use “Server Alerted Sync” as defined in [OMA DS Pro], sub clause 12 “Server Alerted Sync”, to alert the CAB Client to initiate synchronization, when there are data updates in the AB Application Usage.

5.2 Contact Subscription Function

The Contact Subscription function is responsible for handling CAB User’s subscription requests towards other CAB Users and data resulting from Contact Subscription updates,.

The Contact Subscription SHALL use XDM Agent (see sub -clause 5.5 “XDM Agent”) to retrieve the list of contacts (through either XCAP GET or SIP SUBSCRIBE/NOTIFY to the CAB User preferences XML document [CAB XDMS]) to which it must establish Contact Subscription(s), from the CAB User Preference Application Usage, with the following clarifications:

- 1) SHALL use auid “org.openmobilealliance.cab-user-prefs”.
- 2) SHALL obtain the value of the <XUI> elements included in entries of the <subscription-list> element, from the CAB User Preference XML document [CAB XDMS].
- 3) SHALL use the value of the <XUI> elements to create PCC subscription request(s) directly to the PCC Application Usage or via Subscription Proxy as described in [OMA XDM Core].

If a <filter-set> element per [RFC4661] is associated with an <XUI>, then the Contact Subscription Function SHALL include the <filter-set> element in a MIME type "application/simple-filter+xml" [RFC4661] in the SIP SUBSCRIBE, in accordance with XDMC document change subscription procedures for an XDM Agent in sub clause 6.1.2 "Subscribing to Changes in the XDM Resources" [OMA XDM Core].

When the CAB Client (via XDMC) changes a <filter-set> element in an <entry>, the Contact Subscription Function SHALL restart the backend SIP subscription.

The Contact Subscription SHALL use XDM Agent, (see sub clause 5.5 “XDM Agent”), to generate subscription request(s) (via SIP SUBSCRIBE) towards the PCC Application Usage [CAB XDMS] of each of the contact(s) retrieved from <subscription-list>. These subscriptions SHALL be done directly to the PCC Application Usage or via Subscription Proxy as described in [OMA XDM Core].

In case the CAB User to which the Contact Subscription needs to be established is not a contact in the AB XML document, the Contact Subscription SHALL add it as a new Contact Entry in AB XML document and then it SHALL generate the Contact Subscription.

Upon receiving an incoming notification for a contact’s PCC update from a PCC Application Usage (via SIP NOTIFY), the Contact Subscription function:

- 1) SHALL use <contact-subscription-update> element from CAB User’s preference XML document [CAB XDMS] to identify how to process the incoming update, and the following procedures SHALL be applied:
 - a) When <contact-subscription-update> element is “true”, Contact Subscription Function SHALL update AB XML document in [CAB XDMS] with the resulting changes. If the CAB User Preferences element <notify-when-receive-contact-subscription> is set to ‘true’ then the Contact Subscription Function SHALL set in the AB XML document the <updated> element value to “contact subscription”.
 - b) When <contact-subscription-update> element is “false”, Contact Subscription function SHALL store the delta changes to a new <contact> element in AB XML document, and SHALL set the <temporary> element to “contact subscription”, SHALL set ‘contactIdRef’ attribute to the reference of the Contact Entry to which the temporary contact entry type is associated with.

The Contact Subscription Function SHALL set the <contact-status> in the AB XML document in [CAB XDMS]:

- 1) The value of the <contact-subscription status> element:
 - a) <contact-subscription-status> = “active”, when the value of the “Subscription-State” header of the SIP NOTIFY contains the value “active”, meaning that the subscription has been accepted and has been authorized by the contact.
 - b) <contact-subscription-status> = “pending”, when the value of the “Subscription-State” header of the SIP NOTIFY contains the value “pending”, meaning that the subscription has been received, but that the information in the contact’s PCC Access Permissions is insufficient to accept or deny the subscription.
 - c) <contact-subscription-status> = “denied”, when the value of the “Subscription-State” header of the SIP NOTIFY contains the value “terminated” and the reason code is “rejected”, or when the Contact Subscription Function receives a SIP “403 Forbidden” or “603 Decline” response, meaning that the subscription is not allowed by the CAB User’s access permission or service provider policy and the subscription is not active.
 - d) <contact-subscription-status> = “invalid_filter”, when the Contact Subscription Function receives a SIP “488 Not Acceptable Here” response per [RFC4660] or when the Contact Subscription Function receives a SIP NOTIFY response per [RFC4660] with a event-reason-value of value “badfilter”.
 - e) <contact-subscription-status> = “not_found”, when the value of the “Subscription-State” header of the SIP NOTIFY contains the value “terminated” and the reason code is “noresource”, or when the Contact Subscription Function receives a “404 Not Found” error code, meaning that the contact could not be identified as a CAB User and the subscription is not active.
 - f) <contact-subscription-status> = “other_error”, meaning that the subscription is not active and the Contact Subscription Function determines that the non-availability is not transient, so the Contact Subscription Function does not retry.

NOTE: In case the Contact Subscription Function receives a SIP NOTIFY request with the “Subscription-State” header value of “terminated” and determines that the non-availability is transient (e.g., when the reason code is “deactivated” or “probation”), the Contact Subscription Function tries re-subscription, and the Contact Subscription function decides whether the <contact-subscription-status> need to be updated.

5.3 Contact Share Function

The Contact Share Function is responsible for handling CAB User's Contact Share requests, initiating an XDCP request for forwarding Contact Share data, and invoking corresponding messaging actions towards non-CAB User.

5.3.1 Procedures at the Originating Side

The Contact Share Function SHALL support retrieval of Contact Share requests stored in the CAB Feature Handler Application Usage [CAB XDMS]. This is accomplished either by document management operations or subscribing to changes to the CAB Feature Handler Document. The retrieval is performed via the XDM Agent (see sub clause 5.5 "XDM Agent").

Contact Share function SHALL obtain the data for the Contact Share request via XDM Agent (see sub clause 5.5 "XDM Agent") from the CAB Feature Handler Application Usage as follows:

- SHALL set the AUID to "org.openmobilealliance.cab-feature-handler"

Upon obtaining the CAB Feature Handler document, the Contact Share Function SHALL extract the data from the <contact-share> element.

The Contact Share Function

1. SHALL use the <recipients-list> element to determine the recipients and it identifies the recipients type (i.e. whether the recipient is a CAB or a non-CAB User);

If the recipient is a CAB User:

- a. it SHALL construct the XDCP Forward Request using the data retrieved from the CAB XDMS [CAB XDMS] with the following clarifications:
 - i. SHALL set the request URI of the XDCP Request as "http://[XCAP Root URI]/org.openmobilealliance.cab-address-book/users/[XUI]" if the <data> element in the retrieved Contact Share request contains <AB> child element(s);
 - ii. SHALL set the request URI of the XDCP Request as "http://[XCAP Root URI]/org.openmobilealliance.cab-pcc/users/[XUI]" if the <data> element in the retrieved Contact Share request contains <PCC> child element(s);
 - iii. SHALL create the XDCP Document to be included in the XDCP Request by copying the values from the retrieved elements of the <contact-share> ;
 - iv. SHALL bind the value of the "id" attribute of the <cab-feature> element that contains the <contact-share> element, with the <request-id> element of the XDCP Forward request; and
 - v. SHALL include <delivery-report> element in the XDCP Forward request with the value "true" if <delivery-report-request> element in the <contact-share> is set to "true".

If the recipient is a non-CAB User:

- a. It SHALL retrieve appropriate data from CAB User's PCC and/or AB Application Usage based on the data type indicated in the request.
- b. When applicable and subject to service provider policies, it SHALL request the CAB Interworking Function to convert the data to be shared to a Legacy Format via the Interworking Function (see sub clause 5.4 "Interworking Function").

- c. When CPM Interworking Function is used as messaging delivery mechanism, it SHALL construct a message as described in [OMA CPM CONV FCT TS] and send a message towards the CPM Interworking Selection Function [OMA CPM IWF TS].

The Contact Share function SHALL update the <response> element in the CAB Feature Handler Document based on the response received for the XDCP Forward Request. If <delivery-report-request> element for this <contact-share> request was set to “true”, it SHALL further update the <delivery-report-status> in the CAB Feature Handler Document based on the updates made by the CAB XDMS to the Forwarding Notification List [List XDMS].

For each of the recipients, it SHALL create a <entry-report> element per recipient and set the “uri” attribute of the recipient accordingly. The Contact Share function SHALL obtain the Forwarding Notification List Document [List XDMS] through either document management or subscribe/notify mechanisms of [OMA XDM Core] and populate the <delivery-status> element as follows:

- The <code> element contains the value of either “Pending”, “Successful” or “Failure” based on the data retrieved from Forwarding Notification List Document for the XDCP request.
 - The “Pending” value is set when there is no XDM Forward Delivery Report available yet for the XDCP Forward request.
 - The “Successful” value is set when the <status> element of the XDM Forward Delivery Report is either “delivered” or “rejected”.
 - The “Failure” value is set when the <status> element of the XDM Forward Delivery Report is “expired”.
- It MAY populate the <phrase> element with descriptive text corresponding to the delivery status.

5.3.2 Procedures at the Receiving Side

In case of Sharing PCC, at the receiving side, upon getting notified of the temporary document (i.e. ContactSharePCC.xml) from PCC Application Usage, the Contact Share Function SHALL update AB XML document of the recipient after performing the needed conversion with the following clarifications:

- if the preference set by the CAB User in the XDM Preferences Document is “accept”, it SHALL store the contact shared data into a <contact> element and setting the <updated> element of the <contact-status> to the value “contact share”.
- if the preference set by the CAB User in the XDM Preferences Document is “confirm”, it SHALL store the contact shared data into a <contact> and setting <temporary> child element of the <contact-status> to the value “contact share”.

5.4 Interworking Function

The Interworking Function SHALL support the following procedures, described in the sub clauses below:

- Import from non-CAB AB system(s)
- Contact Search with External Directories
- Format adaptation (between CAB Format and Legacy Formats)

5.4.1 Import from Non-CAB AB Systems

The data for the scheduled import requests of contact(s) from non-CAB AB systems into CAB SHALL be retrieved from the CAB Feature Handler Application Usage [CAB XDMS]. This is accomplished either by document management operations or

subscribing to changes to the CAB Feature Handler Document. The retrieval is performed via the XDM Agent (see sub clause 5.5 “XDM Agent”).

The Interworking Function SHALL obtain the data for the import request via XDM Agent (see sub clause 5.5 “XDM Agent”) from the CAB Feature Handler Application Usage as follows .

- SHALL set the AUID to “org.openmobilealliance.cab-feature-handler”
- SHALL use the <non-CAB-source> element to identify the source of the non-CAB system from which to import
- SHALL use the <operations> element to identify the operation associated with import request.
- SHALL use the <scheduled-interval> element to determine the interval between successive accesses to the non-CAB address book system (i.e. in order to determine the level of synchronization between the non-CAB system and the CAB User’s AB)
- MAY use the <credentials> element and the values of the child elements <username> and <password> to obtain access to the non-CAB system(s)

While processing the import request, the Interworking Function MAY set the value of the <code> under <response> element of the corresponding <import-non-cab> element of the CAB Feature Handler XML document to “Pending”.

If the Interworking Function failed to process the request, the Interworking Function SHALL set the value of the <code> under <response> element of the corresponding <import-non-cab> element of the CAB Feature Handler XML document to “Failure”.

After successful imported and format adaptation of the each imported contact, the Interworking Function:

- 1) SHALL use <contact-import-update> element from CAB User’s preference XML document [CAB XDMS] to identify how to store the imported contacts, as follows:
 - a) When <contact-import-update> element is “true”, the Interworking Function SHALL update AB XML document in [CAB XDMS] with the resulting changes and SHALL set the value of <updated> element as “contact imported”.
 - b) When <contact-import-update> element is “false”, Interworking Function SHALL store the imported contact to a new <contact> element in AB XML document [CAB XDMS], and SHALL set the value of <temporary> element as “contact imported”.
- 2) SHALL set the value of <contact-source> element to the value of <non-CAB-source> element.
- 3) SHALL set the value of the <code> element under <response> element of the corresponding <import-non-cab> element of the CAB Feature Handler XML [CAB XDMS] document to “Success”.

The CAB Server performs the AB synchronization as specified in sub clause 5.1 in order to inform the CAB Client of the AB updates.

5.4.2 Contact Search with External Directories

The CAB Interworking Function SHALL support searches to External Directories, by hosting a standard XML format (see sub clause 5.4.2.1) and receiving search requests via XDM-7i interface (i.e. Limited XQuery over HTTP).

The contact search requests and responses SHALL be based the search document schema as described in [OMA XDM Core] sub clause 5.4.1 “Search Document”.

Upon receiving the search request, the CAB Interworking Function SHALL use <dataSource> child element (see sub clause 5.4.2.1.5) to construct an external search request.

CAB Interworking Function translates the search requests received via XDM-7i to external search requests based on the format supported by the External Directories, in the case when External Directories do not support the standard XML format. When the External Directories support the standard XML format for search requests, translation MAY not be required.

Note: The interactions with External Directories and mapping the requests/responses to/from External Directories are out of scope of this specification.

The CAB Interworking Function SHALL route back the search requests responses from the External Directory to the Search Proxy via XDM-7i interface. If multiple External Directories are searched, the CAB Interworking Function SHALL aggregate the results prior to sending the response back to the Search Proxy.

5.4.2.1 Application Usage for External Directories Search

This Applications Usage specifies the standard XML format for supporting searches towards External Directories.

If the External Directories content is stored at the CAB Server, the content SHALL conform to the standard XML format (see sub clause 5.4.2.1.4) and stored in the Global Tree of the Application Usage as described in [OMA XDM Core] sub clause 5.5 “Global Documents”.

5.4.2.1.1 Application Unique ID

The AUID SHALL be “org.openmobilealliance.cab-external-search”.

5.4.2.1.2 Default Namespace

The CAB Interworking Function document default element namespace is "urn:oma:xml:cab:external-search".

5.4.2.1.3 Search Capabilities

This Application Usage SHALL support search requests towards External Directories based on standard XML search document that conforms to the XML schema defined in sub clause 5.4.2.1.4 and the following rules apply in addition to the procedures defined in [OMA XDM Core]:

1. support a collection “org.openmobilealliance.cab-external-search/global”;
2. The basic XQuery expression supported by this Application Usage SHALL be as follows:
xquery version "1.0";
declare default element namespace "urn:oma:xml:cab-external-search";

All search requests that do not comply with the basic XQuery expression as defined in this sub clause SHALL be responded with an HTTP “409 Conflict” error response as defined by [OMA XDM Core] in sub clause 6.2.3.

5.4.2.1.4 XML Schema

The standard XML format for external search SHALL conform to the PCC XML schema as described in [XSD cab pcc].

5.4.2.1.5 Extension for <dataSource>

Additional CAB extension to the search document as described in [OMA XDM Core] sub clause 5.4.1 “Search Document”:

- The <dataSource> child element of the <search> element indicates the specific external directory source to which the <request> is targeted or the specific external directory source from which the <response> is received. The value SHALL be of type String.

This extension is described in [XSD extSearch]

5.4.3 Format Adaptation

The format adaptation between CAB Format and Legacy Format(s) SHALL be supported by the Interworking Function.

The following table defines the supported properties and corresponding semantics between the CAB Format and the Legacy Format (s) vCard object, [vCard 2.1], [RFC2425], [RFC2426].

As all vCard 2.1 or vCard 3.0 properties support a LANGUAGE parameter, it is assumed that this LANGUAGE parameter corresponds implicitly to the "xml:lang" [W3C-XML] attribute which is allowed for all Contacts and PCCs properties (both "xml:lang" and LANGUAGE refer to [RFC5646]).

A few of these properties are defined as informative vCard extensions in Appendix D.

Contacts or PCC Property or attribute	vCard 2.1 Property/parameter	vCard 3.0 Property/parameter
For PCCs only <pcc>		
"pcc-type" attribute		
For Contacts only <contact>		
<contact-status> Cardinality: (0,1)		
<contact-type> Cardinality: (0,1)		
<entry-status> Cardinality: (0,1)		
<updated> Cardinality: (0,n)		
<temporary> with "contactIdRef" attribute Cardinality: (0,1)		
"contactIdRef" attribute		
<contact-subscription-status> Cardinality: (0,1)		
<contact-source> Cardinality: (0,1)		
For PCCs and Contacts <pcc> and <contacts>		
<person-details> with "index" attribute PCC: Cardinality: (0,n) Contact: Cardinality: (0,1)	See conditions & details below The mapping corresponds to one <person-details>.	
"index" attribute		
<name> with "pref", "name-type", "index" attributes Cardinality: (1,n)	<N> without the Pref attribute value and without name-type attribute value ← only the one with minimum pref value	<N> (or <Nickname> when name- type='KnownAs'), without the Pref attribute value and without name-type attribute value ← only the one with minimum pref value (without name-type)
"pref" attribute		
"name-type" attribute		
"index" attribute		
<title> with 'display-order' attribute Cardinality: (-0,1)	<N>; Name Prefix (fourth field)	<N>; Honorary Prefix (fourth field)
"display-order"		

attribute		
<given> with 'display-order' attribute Cardinality: (0,1)	<N>; Given Name (second field)	
"display-order" attribute		
<middle> with 'display-order' attribute Cardinality: (0,1)	<N>; Additional Names (third field)	
"display-order" attribute		
<family> with 'display-order' attribute Cardinality: (0,1)	<N>; Family Name (first field)	
"display-order" attribute		
<gen-id> with "display-order" attribute Cardinality: (1,n)	<N>; Name Suffix (fifth field)	<N>; Honorific Suffix (fifth field)
"display-order" attribute		
<degree> with "display-order" attribute Cardinality: (1,n)	<N>; Name Suffix (fifth field)	<N>; Honorific Suffix (fifth field)
"display-order" attribute		
<phonetic-given> Cardinality: (1,n)		
<phonetic-family> Cardinality: (1,n)		
<display name> Cardinality: (1,1)	<FN>	
<address> with "index", "pref", "addr-type" attributes Cardinality: (0,n)	<ADR> with TYPE=addr-type value without the Index. Pref attributes values Or <LABEL> with TYPE=addr-type value without the Index. Pref attributes values ← only the first value of <address> or the one with minimum pref value see conditions & details below	
"index" attribute		
"pref" attribute		
"addr-type" attribute	TYPE parameter	
<location> Cardinality: (0,1)		
<label> Cardinality: (0,1)		
<addr-string> Cardinality: (0,1)		
<country> Cardinality: (0,1)	<ADR>; Country (seventh field)	
<region> with "region-type" attribute Cardinality: (0,1)	<ADR>; region (fifth field) ← only the region-name i.e. without the region-type	
"region-type" attribute		
<region-name> Cardinality: (1,1)	<ADR>; region (fifth field)	

<sub-region> with "region-type" attribute	
"region-type" attribute	
<locality> with "location-type" attribute Cardinality: (0,1)	See conditions & details below
"location-type" attribute	
<locality-element> Cardinality: (1,1)	<ADR>; locality-element (fourth field)
<sub-locality> with "sublocality-type" attribute	
"subloc-type" attribute	
<street> Cardinality: (0,1)	<ADR>; Street (third field) "str-name and str-number" are just put one after the other (i.e. without separating comma) <u>Example</u> : ADR;TYPE=home: ;str-name str-number ;.....
< str-name> Cardinality: (1,1)	<ADR>; Street (third field)
<str-number> Cardinality: (1,1)	<ADR>; Street (third field)
<intersection>	
<int-name> Cardinality: (1,1)	
<int-number>	
<post_code> Cardinality: (0,1)	See conditions & details below
<post-code-main> Cardinality: (1,1)	<ADR>; post-code-main (sixth field)
<sub-post-code> Cardinality: (1,1)	
<postal-delivery-point> Cardinality: (0,1)	<ADR>; Post Office Box (first field) only the first postal-delivery-point-name without index and pref attributes
<postal-delivery-point-name> with "index", "pref" attributes Cardinality: (1,n)	<ADR>; Post Office Box (first field) only the first postal-delivery-point-name without index and pref attributes
"index" attribute	
"pref" attribute	
<post-office> Cardinality: (0,1)	
<postal-office-name> with "index", "pref" attribute Cardinality: (1,n)	
"index" attribute	
"pref" attribute	
<rural-delivery-point> with "index", "pref" attributes Cardinality: (0,1)	
"index" attribute	
"pref" attribute	
<extended-address> with "index" attribute Cardinality: (1,n)	<ADR>; Extended address (second field) ← only the first value of <extended-address> without index attribute
"index" attribute	
<premises> with "premise-type" attribute	<ADR>; Extended address (second field) ← only the first value of <extended-address> without index attribute

"premises-type" attribute	
<premises-name>	
<premises-number>	
<sub-premises> with "sub-premises-type" attribute	
"sub-premises-type" attribute	
<sub-premises-name>	
Card : (0,n)	
<sub-premises-number>	
Card : (0,n)	
<location> Cardinality: (0,1)	<GEO> and <TZ> see conditions & details below
< label>	
<latitude>	<GEO>; lat (first field) in decimal form with "decimal value=+/- degrees + minutes/60 + seconds/3600"
<degrees-measure>	Decimal value is positive when lat-sign='N', negative when lat-sign='S'
<minutes-measure>	
<seconds-measure>	
<lat-sign>	
<longitude>	<GEO>; lon (second field) in decimal form with "decimal value=+/- degrees + minutes/60 + seconds/3600"
<degrees-measure>	Decimal value is positive when long-sign='E', negative when long-sign='W'
<minutes-measure>	
<seconds-measure>	
<long-sign>	
<altitude>	
<time-zone>	See conditions & details below
<tz-label>	
<utc-offset>	<TZ>
<tz-uri>	
<comm.-addr> Cardinality: (1,1)	<TEL> and <EMAIL> with parameter PREF See conditions & details below
< uri-entry> with "index", "pref", "addr-uri-type", attributes Cardinality: (0,n)	See conditions & details below
"index" attribute	
"pref" attribute	a) When addr-uri-type = 'email' <EMAIL>;PREF;TYPE=addr-uri-type values:uri PREF is only used for the uri with the minimum "pref" value b) When addr-uri-type="SIP URI" or "IM" or "pres URI" <IMPP>;PREF:uri PREF is only used for the uri with the minimum "pref" value
"addr-uri-type" attribute	
<addr-uri> Cardinality: (1,1)	
< label> (i.e communication mean)	
<tel> with "index", "pref", "tel-type" Cardinality: (0,n)	See conditions & details below
"index" attribute	
"pref" attribute	<TEL>;PREF;TYPE=tel-type: tel-nb and extension translated to an X500 value PREF is only used for the tel with the minimum "pref" value
"tel-type" attribute	
<tel-nb> which is <tel-str> or <tel-uri> or <E.164> Cardinality: (1,1)	

	<extension>	
	<label>	
<birth> Cardinality: (0,1)	<BDAY> see conditions & details below	
< date> Cardinality: (0, 1)	<BDAY>	
<non-greg-date> with "cal-type" attribute Cardinality: (0,n)		
"cal-type" attribute		
"index" attribute		
<place>> with "index" attribute Cardinality: (1,n)		
"index" attribute		
<anniversary-list> Cardinality: (0,1)		
<anniversary-entry> Cardinality: (1,n)		
"index" attribute		
<anniversary-date> Cardinality: (0,1)		
<date> Cardinality: (0,1)		
<non-greg-date> with "cal-type" and index attributes Cardinality: (0,n)		
"cal-type" attribute		
"index" attribute		
< label>		
<gender> Cardinality: (0,1)		
<language-list> Cardinality: (0,1)		
<language-entry> with "language-proficiency- type", "language- fluency-type", "index", "pref" attributes Cardinality: (1,n)		
"language- proficiency-type" attribute		
"language- fluency-type" attribute		
"index" attribute		
"pref" attribute		
<media-list> Cardinality: (0,1)	<SOUND> and <LOGO> and <PHOTO> see conditions & details below	

<media-entry> with "media-content", "media-type", "index", "pref" attributes Cardinality: (1,n)	See conditions and details below
	When media-content is "video"
"media-content" attribute	
"media-type" attribute	
"index" attribute	
"pref" attribute	
"media"	When media-content is "sound" or "logo" or "photo" <SOUND> or <LOGO> or <PHOTO> with TYPE="a translation of media-type attribute i.e TYPE=the media type value when media-content = photo or logo (see RFC4288)"
<media-label>	
<category-list> Cardinality: (0,1)	<CATEGORIES>
<category-entry> with "index" attribute Cardinality: (1,n)	<CATEGORIES>
"index" attribute	
<web resources> Cardinality: (0,1)	<URL> see conditions & details below
<web-entry> with "index" attribute	
"index" attribute	
<url> Cardinality: (1,1)	<URL> ← only the first value of <url> without the pref attribute value
< label>	
<key-list> Cardinality: (0,1)	<KEY> with TYPE= key-type value
<key-entry> with "display-order", "key-type" attributes	
"display-order" attribute	
"key-type" attribute	<KEY> with TYPE= key-type value
<key> Cardinality: (1,1)	
<label>	
<service-list> Cardinality: (0,1)	
<service-entry> with "index" attribute Cardinality: (1,n)	
"index" attribute	
< label>	
<alias>	
< url>	
<expertise-list> Cardinality: (0,1)	
<expertise-entry> with "e-	

level", "index" attributes Cardinality: (1,n)	
"e-level" attribute	
"index" attribute	
<hobby-list> Cardinality: (0,1)	
<hobby-entry > with "h-level", "index" attributes Cardinality: (1,n)	
"h-level" attribute	
"index" attribute	
<interest-list> Cardinality: (0,1)	
<interest-entry > with "i-level", "index" attributes Cardinality: (1,n)	
"i-level" attribute	
"index" attribute	
<career-history> Cardinality: (0,1)	
<history-entry> with "index" attribute Cardinality: (1,n)	
"index" attribute	
<history-description> with "history-type" attribute Cardinality: (1,1)	
"history-type" attribute	
<start-date> with "cal-type"	
"cal-type" attribute	
"end-date" with "cal-type" attribute	
"cal-type" attribute	
<note> Cardinality: (0,1)	<NOTE>
<public-note> Cardinality: (0,1)	
Any other elements	
<organization-details> with "index" attribute PCC: Cardinality: (0,n) Contact: Cardinality: (0,1)	See conditions & details below The mapping corresponds to one <organization-details>
"index" attribute	
<org-name> with "pref", "org-name-type", "index" attributes Cardinality: (1,n)	<ORG> see conditions & details below
"pref" attribute	
"org-name-type" attribute	
"index" attribute	

<display-name> Cardinality: (0,1)	<ORG> ← only the display-name and unit values included in the first occurrence of <org-name> without the pref, "index" and org-name-type attributes values
<unit> Cardinality: (0,1)	
<entity> Cardinality: (0,1)	
<org_member-list> Cardinality: (0,1)	
<org_directory> Cardinality: (0,1)	
<comm.-addr> Cardinality: (0,1)	
<media-list> Cardinality: (0,1)	
<web resources> Cardinality: (0,1)	
<key-list> Cardinality: (0,1)	
Any other elements	

<group-details> with "index" attribute PCC: Cardinality: (0,n) Contact : Cardinality : (0,1)	
"index" attribute	
<group-name> Cardinality: (1,n)	
<group_member-list> Cardinality: (0,1)	
<group_uri> Cardinality: (0,1)	
<comm.-addr> Cardinality: (0,1)	
<media-list> Cardinality: (0,1)	
<web resources> Cardinality: (0,1)	
<key-list> Cardinality: (0,1)	
Any other elements	

Any other elements	
--------------------	--

Table 1: Mapping between CAB Format and Legacy Format(s)

5.5 XDM Agent

The XDM Agent acts as a supporting entity to other CAB Server functions and supports interactions (i.e. document management operations, subscriptions/notifications, history access, forwarding, document share by reference) with CAB

Application Usages [CAB XDMS]. The document management operation supported for PCC Application Usages is limited to read-only

The XDM Agent SHALL support the following procedures as described in [OMA XDM Core] sub clause 6.1 “Procedures at the XDMC and the XDM Agent”:

- Procedures of document management of CAB XML documents [CAB XDMS] based on sub clause 6.1.1 “Document Management” from [OMA XDM Core].
In case the AB Synchronization Function requests XDM Agent to access AB Application Usage, the XDM Agent SHALL use the identity (XUI) of the CAB User. In all the other cases when the CAB Server performs write operations into the AB Application Usage (e.g. triggered by Contact Subscription), the XDM Agent SHALL use XUI assigned to the CAB Server, (i.e. different from XUIs of CAB Users).
- Procedures of subscription to changes in CAB XML documents [CAB XDMS] as described in sub clause “6.1.2 Subscribing to changes in the XML documents” with the exception of sub clause 6.1.2.1.2 “XDMC” as described in [OMA XDM Core].
- Procedure of Document Reference information management for CAB AB documents as described in sub clause 6.1.1.3.1 “Document Reference Operations” (only XDM Agent part) from [OMA_XDM_core].
- Procedures of subscription to changes in UPP Directory Document stored in UPP Directory XDMS[OMA-XDM-UPP] as described in sub clause 6.1.2 “Subscribing to changes in the XML documents” with the exception of sub clause 6.1.2.1.2 “XDMC” as described in [OMA XDM Core].
- Procedures of history information access for CAB XML documents [CAB XDMS] based on sub clause 6.1.4.2 “Request History Information [OMA XDM Core]
- Procedures of XDM Forwarding for CAB XML documents [CAB XDMS] based upon sub clause 6.1.1.3.2 “XDM Resource Forwarding Operations [OMA XDM Core]

The XDM Agent SHALL support the Application Usages specified in [CAB XDMS].

5.6 Contact Status Function

The Contact Status Function SHALL manage the Contact Status information contained in the <contact-status> element of the Contact Entry as specified in AB Application Usage [CAB XDMS].

5.6.1 Management of Contact Status in AB

When there is a change to the contacts in the AB (e.g. to an existing or new Contact Entry), the CAB Server records the status of the change to the <contact-status> element of the corresponding Contact Entry. The CAB Server SHALL use the <contact-status> element and its child elements to convey the status information to the CAB User.

The discovery mechanism that is used to populate the <contact-type> element value of the <contact-status> element in the Contact Entry of the AB (for indicating the CAB Status of the CAB User’s contacts) is out of scope of this specification.

The Contact Status in the AB SHALL be populated for the following operations that occur in the CAB Server:

- Contact Subscription – See sub clause 5.2 “Contact Subscription Function” for procedures to populate the Contact Status information associated with the outgoing Contact Subscriptions.
- Import non-CAB – See sub clause 5.4.1 “Import from Non-CAB AB Systems” for procedures to populate the Contact Status information associated with the Import non-CAB requests.
- Incoming Contact Subscription Request - See sub clause 5.6.2 “Incoming Contact Subscription Requests” for procedures to populate the Contact Status information associated with the incoming Contact Subscription Requests.
- Contact Share – See sub clause 5.3 “Contact Share Function” for procedures to populate the Contact Status information associated with incoming Contact Share information.

5.6.2 Incoming Contact Subscription Requests

The incoming Contact Subscription requests for reactive authorization are notified to the CAB User using the Contact Status information in CAB User’s AB.

The Contact Status Function SHALL detect and notify the incoming Contact Subscription request(s) that require re-active authorization from the CAB User using the following procedures:

- SHALL set the <history-prefs> in XDM Preferences Document in PCC Application Usage [CAB XDMS] as described in [OMA XDM Core], sub clause 5.8 “XDM Preferences Document” in such a way that the Request History Information document (as described in [OMA XDM Core], sub clause 5.7.2 “Request History Information Document”) is updated with unsuccessful incoming subscription requests are stored in the PCC Application Usage.
- SHALL use the <notify-when-receive-contact-subscription> element in CAB User Preferences XML document check the notification preferences of the CAB User for incoming subscription requests.
- If <notify-when-receive-contact-subscription> element is set to “true”, the Contact Status Function SHALL retrieve the information associated with unsuccessful incoming requests via XDM Agent (see sub clause 5.5 “XDM Agent”), and:
 - o If the CAB User’s AB XML document [CAB XDMS] contains a Contact Entry associated with the requestor of the incoming subscription request, the <updated> element under the <entry-status> of the Contact Entry SHALL be set to the value ‘incoming subscription request’.
 - o If the CAB User’s AB XML document [CAB XDMS] does not contain a Contact Entry associated with the requestor of the incoming subscription request, a new <contact> element SHALL be created in the AB XML document, SHALL set the <temporary> element under the <entry-status> with the value ‘incoming subscription request’, and corresponding contact information of the requestor is populated under the <contact> element.

5.6.3 Contact Added

The CAB Server SHALL detect the addition of a CAB User to another CAB User’s AB, once the added contact is marked as CAB User in the <contact-type> element in the AB. The detection of a contact being a CAB User is out of scope of this specification. The interdomain conveyance of CAB User addition to other CAB User’s Address Book is achieved though SIP MESSAGE method as described in the following sub clause.

5.6.3.1 Originating Side

If the <send-notification-contact-added> element of the CAB User Preferences XML document [CAB XDMS] is set to "true", the following procedures SHALL apply:

The ‘CAB Contact Added’ data as described in the table below SHALL be used to share between multiple domains to resolve CAB-HLF-012 [CAB RD]. The ‘CAB Contact Added’ data is generated at the CAB Server when a CAB User adds another CAB User (who belongs to the remote domain) to his/her AB.

AB Owner – is referred to the CAB User who is performing the Add operation

Added Contact – is referred to the contact that is added by the CAB User.

Element	Data Type	Cardinality	Description
AB owner			This represents the AB owner data. It SHALL contain the owner-XUI element
Owner-XUI	xs:anyURI	1	The XUI of the originating CAB User performing the

			Add operation
Added Contact			This represents the Added Contact data. It SHALL contain the added-XUI of the contact.
Added-XUI	xs:anyURI	1	The XUI of the contact added by the CAB User

Figure 1: Contact added data details

The CAB Server SHALL send the fragment representing ‘CAB Contact Added’ data as described in the above table using the CAB-NNI-1 interface [CAB AD] to the remote CAB domain, based on the XUI of the Added Contact.

Prior to sending the fragment to the remote domain, the CAB Server SHALL check the CAB User’s PCC Access Permissions to verify if the CAB User’s XUI publication to the Added Contact is allowed. The publication is considered allowed if at least <allow-retrieve> access level is granted to the Added Contact. The CAB Server SHALL proceed as follows:

- If the PCC Access Permissions do not allow publication of the CAB User’s XUI to the Added Contact’s XUI, the CAB Server SHALL NOT send the fragment.
- If the PCC Access Permissions allow publication of the CAB User’s XUI to the Added Contact’s XUI, the CAB Server SHALL send the fragment using SIP MESSAGE[RFC 3428] method as described below:

The CAB Server SIP MESSAGE request SHALL be constructed and sent according to SIP/IP Core [OMA XDM Core] using the following steps:

1. Set the Request-URI of the SIP MESSAGE with the value of the recipient’s XUI (i.e. the value of added-XUI).
2. Set the To header field of the SIP MESSAGE with the Request-URI;
3. Set the From header field of the SIP MESSAGE with the value of the originator’s XUI (i.e. the value of owner-XUI) and the ‘display-name’ parameter of the From header to the Owner’s display-name from his/her PCC that has the highest ‘pref’ value.;
4. Set the P-Asserted-Identity header field of the SIP MESSAGE with the value of the originator’s XUI (i.e. the value of owner-XUI);
5. Set the Accept-Contact header field with the value of the CAB feature tag, i.e. “3gpp-service.ims.icsi.oma.cab_1.0”;
6. Set the P-Preferred-Service header field with the value of the CAB feature tag, i.e. “3gpp-service.ims.icsi.oma.cab_1.0”;
7. The SIP MESSAGE SHALL have an empty body.
8. Set the other required SIP headers according to [OMA XDM Core] and send the SIP MESSAGE according to the procedures of the SIP/IP Core that is referenced in [OMA XDM Core];

5.6.3.2 Receiving Side

The CAB Server SHALL receive the ‘CAB Contact Added’ data from the remote domain using MESSAGE request.

The CAB Server SHALL process the received SIP MESSAGE request according to the procedures of the SIP/IP Core referenced by [OMA XDM Core], as follows:

- Check if the CAB feature tag “3gpp-service.ims.icsi.oma.cab_1.0” is the present in the Accept-Contact header field. If the CAB feature tag is not present or the value is not recognized, the CAB Server SHALL reject the request with SIP 403 "Forbidden" response.
- Retrieve and process the SIP MESSAGE request (i.e. validate the value of “To” and “From” headers to match the semantics of added-XUI and owner-XUI respectively);
- The CAB Server SHALL use the ‘display-name’ parameter from the From header in the notification to the recipient CAB User.
- If the SIP MESSAGE has an empty body, the CAB Server SHALL interpret it as a contact-added notification.
- Respond with 200 OK SIP response code according to [RFC 3428]; or
- Reject with 488 Not Acceptable Here SIP response code if the values of “To” and “From” values do not match the semantics of added-XUI and owner-XUI respectively.

Upon identifying that a CAB User has been added by another CAB User (i.e., by receiving a SIP MESSAGE from the remote domain or determined otherwise within the same domain) the CAB Server SHALL verify the following:

- i. if the recipient CAB User’s <receive-notification-when-contact-added> user preference is set to ‘true’, and
- ii. if the recipient CAB User’s AB Access Permissions for the originating CAB User XUI allow the notification of the Contact Added data

then the Contact Status Function on recipient side SHALL perform the following:

5.7 Applying CAB User Preferences

The CAB Server SHALL act as XDM Agent as described in sub clause 5.5 to retrieve the UPP Directory Document [OMA XDM UPP] of each CAB User and identify the Active Profile associated with each of CAB User’s devices. After identifying the Active Profile, the CAB Server SHALL apply the preferences stored in the CAB User Preferences Document [CAB XDMS] related to the Active Profile.

6. Procedures at CAB Client

6.1 Address Book Management and Synchronization

The CAB Client SHALL use OMA DS Protocol [OMA DS Pro] as specified in CAB-1 interface [CAB AD] to synchronize the data modifications in the address book with the CAB Server.

The CAB Client SHALL support at least one of AB CAB Format or the Legacy Formats vCard 2.1 [vCard 2.1], vCard 3.0 [RFC2425], [RFC2426] for AB synchronization with CAB Server.

Note: Syncing with the Legacy Formats may result in reduced support of the AB contact information provided by the AB Application Usage. (mapping table is specified in sub clause 5.4.3 "Format Adaptation").

CAB Client SHALL be able to receive "Server Alerted Synch" as specified in [OMA DS Pro], sub clause 12 "Server Alerted Sync", to synchronize with the CAB Server i.e. as a result of updates from AB Application Usage [CAB XDMS].

6.2 Personal Contact Card (PCC) Management

The CAB Client SHALL format the requests for CAB User's PCC document management as described in the [OMA XDM Core] sub clause 6.1.1 "Document Management" with the following clarifications:

- It SHALL populate the AUID of the XCAP URI with "org.openmobilealliance.cab-pcc" application usage for the PCC, as defined in [CAB XDMS].

6.3 CAB User Preferences Management

The CAB Client SHALL format the requests for CAB User Preferences document management as described in the [OMA XDM Core] sub clause 6.1.1 "Document Management" with the following clarifications:

- It SHALL populate the AUID of the XCAP URI with "org.openmobilealliance.cab-user-prefs" Application Usage for CAB User preferences as defined in [CAB XDMS].

6.4 Import Non-CAB Address Book

The CAB Client SHALL use the <import-non-cab> element in the CAB Feature Handler XML document [CAB XDMS] to store the non-CAB address import request data, and SHALL use the procedures as described in the [OMA XDM Core] sub clause 6.1.1 "Document Management" with the following clarifications:

- 1) SHALL use the AUID "org.openmobilealliance.cab-feature-handler".
- 2) SHALL populate <non-CAB-source> and <credential> elements.

The CAB Client SHALL use the <response> element of the <import-non-cab> element to indicate the status of the request to the CAB User, with the following clarifications:

- 1) SHALL use either subscription operation or document management operation as specified in [OMA XDM Core] to obtain the CAB Feature Handler XML document [CAB XDMS].
- 2) SHALL use <import-non-cab> "id" attribute that is associated with the same request.
- 3) MAY delete the whole import request of <import-non-cab> element, when AB synchronization is completed.

The CAB Client retrieves the imported Contact Entries (i.e. with either <updated> or <temporary> element value set to "contact imported"), through the AB synchronization performed as specified in sub clause 6.4.

6.5 Contact Share

The CAB Client SHALL use the <contact-share> element in the CAB Feature Handler XML document [CAB XDMS], to store the Contact Share request data and SHALL use the procedures as described in the [OMA XDM Core] sub clause 6.1.1 “Document Management” with the following clarifications:

- SHALL use the AUID “org.openmobilealliance.cab-feature-handler”.
- SHALL populate the <recipients-list> element
- SHALL populate the <data> element with their child elements i.e. <PCC> and/or <AB> elements.
- MAY populate <note>, <display-name>, and <delivery-report-request> elements.

The CAB Client SHALL use the <response> element of the <contact-share> element to indicate the status of the Contact Share request to the CAB User, with the following clarifications:

- 1) SHALL use either subscription operation or document management operation as specified in [OMA XDM Core] to obtain the <response> element of the CAB Feature Handler XML document [CAB XDMS].
- 2) SHALL use <contact-share> “id” attribute that is associated with the same request.
- 3) When the <delivery-report-request> is set to “false”, or when <delivery-report-status> element has the value “Successful” for all recipients, it MAY delete the <contact-share> element of the request, when the <code> element of the <response> has the value “Success“, subject to Service Provider policies.

Note: Handling of contact share requests for multiple recipients with multiple <delivery-reports-status> values is out of scope.

The CAB Client SHALL use the <delivery-report-status> element of the <contact-share> element to indicate the status of the delivery report for the corresponding Contact Share request to the CAB User.

6.6 Contact Subscription

In order to subscribe or unsubscribe to a contact’s PCC document changes, the CAB Client SHALL follow the procedures as described in the [OMA XDM Core] sub clause 6.1.1 “Document Management” to respectively add or delete the corresponding <entry> element in the <subscription-list> element of his/her CAB User Preferences Application Usage [CAB XDMS] with the following clarifications:

- It SHALL populate the AUID of the XCAP URI with “org.openmobilealliance.cab-user-prefs” Application Usage for CAB User Preferences as defined in [CAB XDMS];
- It SHALL populate the <XUI> sub-element of the <entry> element with the XUI of the contact to subscribe to;
- It MAY populate a <filter-set> sub-element of the <entry> element by following the procedure described in sub clause 6.1.2.1.2 “XDMC” of [OMA XDM Core].

6.7 Access Permissions management

If the Access Permissions Document is used, the procedure to manage the Access Permissions Document at CAB Client SHALL conform to the sub clause of 6.1.1 “Document Management” and the sub clause of 5.6 “Access Permissions Document” in [OMA XDM_Core] with the clarification given in this sub clause.

6.7.1 Access Permissions Document for Address Book

The Access Permissions Document for Address Book SHALL be addressed using the User Directory Document Selector '/oma-ap/access-permissions' with Address Book AUID. The HTTP Request-URI for the document management operations SHALL be set with the following value:

- "http://[XCAP_Root_URI]/org.openmobilealliance.cab-address-book/users/[XUI]/oma_ap/access-permissions"

6.7.2 Access Permissions Document for PCC

The Access Permissions Document for PCC SHALL be addressed using the User Directory Document Selector '/oma-ap/access-permissions' with PCC AUID. The HTTP Request-URI for the document management operations SHALL be set with the following value:

- "http://[XCAP_Root_URI]/org.openmobilealliance.cab-pcc/users/[XUI]/oma_ap/access-permissions"

6.8 UPP Directory Document Management

The procedure to manage the UPP Directory Document [OMA XDM UPP] at CAB Client SHALL conform to the sub clause 6.1.1 "Document Management". The HTTP Request-URI for the document management operations SHALL be set with the following value:

- "http://[XCAP_Root_URI]/org.openmobilealliance.upp-directory/users/[XUI]"

6.9 XDM Preferences Management

The CAB Client SHALL use the XDM Preferences document to manage the AB and PCC XDM Forwarding preferences for Contact Share operations and the PCC Request History preferences for reactive authorization sub clause 5.6.2 "Incoming subscription requests". The procedures to manage the XDM Preferences Document at CAB Client SHALL conform to the sub clause 6.1.1 "Document Management" and the sub clause 5.8 "XDM Preferences Document" in [OMA XDM_Core] with the clarification given in this sub clause.

6.9.1 XDM Preferences Document for Address Book

The XDM Preferences Document for Address Book SHALL be addressed using the User Directory Document Selector '/oma_xdm_pref/preferences' with Address Book AUID. The HTTP Request-URI for the document management operations SHALL be set with the following value:

- "http://[XCAP_Root_URI]/org.openmobilealliance.cab-address-book/users/[XUI]/oma_xdm_pref/preferences"

6.9.2 XDM Preferences Document for PCC

The XDM Preferences Document for PCC SHALL be addressed using the User Directory Document Selector '/oma_xdm_pref/preferences' with PCC AUID. The HTTP Request-URI for the document management operations SHALL be set with the following value:

- "http://[XCAP_Root_URI]/org.openmobilealliance.cab-pcc/users/[XUI]/oma_xdm_pref/preferences"

6.10 Subscription to CAB XML documents changes

There are two mechanisms (i.e. SIP and XDCP/Push) through which a CAB Client can subscribe to document changes stored in CAB XDMS(s), except AB XDMS. The CAB Client SHALL support one of the two mechanisms.

6.10.1 Subscriptions to CAB User's own PCC document changes

The CAB Client SHALL generate the subscription requests to CAB User's own PCC document changes, as described in [OMA XDM Core] sub clause 6.1.2 "Subscribing to changes in XDM Resources" with the following clarifications:

- If the SIP method is used, the Request-URI SHALL be set to the XUI of the CAB User with the URI Parameter: `aid=<"org.openmobilealliance.cab-pcc">`;
- If the XDCP/Push method is used, the AUID in the Request-URI as described in [OMA XDM Core] sub clause 6.1.1.3 "XDM Operations using XDCP" SHALL be set to the "org.openmobilealliance.cab-pcc".

6.10.2 Subscriptions to CAB User's own CAB User Preferences document changes

The CAB Client SHALL generate the subscription requests to CAB User's own CAB User Preferences document changes, as described in [OMA XDM Core] sub clause 6.1.2 "Subscribing to changes in XDM Resources" with the following clarifications:

- If the SIP method is used, the Request-URI SHALL be set to the XUI of the CAB User with the URI Parameter: `aid=<"org.openmobilealliance.cab-user-prefs">`;

If the XDCP/Push method is used, the AUID in the Request-URI as described in [OMA XDM Core] sub clause 6.1.1.3 "XDM Operations using XDCP" SHALL be set to the "org.openmobilealliance.cab-user-prefs".

6.10.3 Subscriptions to CAB User's own CAB Feature Handler document changes

The CAB Client SHALL generate the subscription requests to CAB User's own CAB Feature Handler document changes, as described in [OMA XDM Core] sub clause 6.1.2 "Subscribing to changes in XDM Resources" with the following clarifications:

- If the SIP method is used, the Request-URI SHALL be set to the XUI of the CAB User with the URI Parameter: `aid=<"org.openmobilealliance.cab-feature-handler">`;
- If the XDCP/Push method is used, the AUID in the Request-URI as described in [OMA XDM Core] sub clause 6.1.1.3 "XDM Operations using XDCP" SHALL be set to the "org.openmobilealliance.cab-feature-handler".

6.10.4 Subscription to CAB User's own CAB XML documents using Subscription Proxy

If the Subscription Proxy was provisioned to the XDMC of the CAB Client and subscription to more than one CAB User's own CAB XML documents, the CAB client SHALL generate the subscription requests to the Subscription Proxy, as described in [OMA XDM Core] sub clause 6.1.2 "Subscribing to changes in XDM Resources" with the following clarifications:

- If the SIP method is used, the Request-URI SHALL be set to the SIP URI of the Subscription Proxy and the body of the SIP SUBSCRIBE request SHALL contain the resource list as defined in [OMA XDM Core] with all relevant XDM resource entries, in which the Document Selector SHALL be set to one or more document selectors in the following list:
 - "org.openmobilealliance.cab-pcc/users/[XUI]/PCC.xml for the PCC document,
 - "org.openmobilealliance.cab-user-prefs/users/[XUI]/CAB-UP.xml for the CAB User Preferences document,

- “org.openmobilealliance.cab-feature-handler /users/[XUI]/feature-handler.xml for the CAB Feature Handler document,

The XUI SHALL be set to the XUI of the CAB User.

- If the XDCP/Push method is used, the Request-URI as described in [OMA XDM Core] sub clause 6.1.1.3 “XDM Operations using XDCP” SHALL be set to “http://[XCAP_Root_URI]/org.openmobilealliance.xdcp.sp”, targeting to the Subscription Proxy, and the payload of the XCAP request SHALL contain the resource list as defined in [OMA XDM Core] with all relevant XDM resource entries, in which the Document Selector SHALL be set to one or more document selectors in the following list:
 - “org.openmobilealliance.cab-pcc/users/[XUI]/PCC.xml for the PCC document,
 - “org.openmobilealliance.cab-user-prefs/users/[XUI]/CAB-UP.xml for the CAB User Preferences document,
 - “org.openmobilealliance.cab-feature-handler /users/[XUI]/feature-handler.xml for the CAB Feature Handler document,

The XUI SHALL be set to the XUI of the CAB User.

6.11 Contact Search

When the CAB User requests contact search, CAB Client SHALL perform the following:

1. Construct a search request document according to the rules and procedures described in [OMA XDM Core] sub clause 5.4.1 “Search Document” and sub clause 6.2.3 “Searching for Data in XML Documents” with the following clarifications:
 - a. When searching AB, the collection parameter SHALL be “org.openmobilealliance.cab-address-book/users/[XUI]/AB”, where [XUI] represents the XUI of a CAB User and AB represents the Address Book document name. This search is limited to the CAB User’s AB.
 - b. When searching PCC, the collection parameter SHALL be “org.openmobilealliance.cab-pcc/users/”.
 - c. When searching External Directories the collection parameter SHALL be “org.openmobilealliance.cab-external-search/global/”, and SHALL use the <dataSource> child element CAB extension [XSD extSearch] of the <search> element to indicate the specific external directory source to which the <request> is targeted or the specific external directory source from which the <response> is received.
2. Send the search request document using a HTTP POST request to the Aggregation Proxy according to the rules and procedures described in [OMA XDM Core] sub clause 6.1.3 “Searching for Data in XML Documents” with the following clarification:
 - a. When searching PCC, the value of “domain” parameter SHALL be set to any of the following values: domain=[home, all, or target domains] and be subject to service provider policies.
 - b. When searching External Directories, the value of “domain” parameter SHALL be “home”.

6.12 Contact Status Management

The CAB Client SHALL use the <contact-status> element from the AB XML document to convey the status information of the AB contacts to the CAB User.

The following procedures SHALL be performed by the CAB Client, based on the values within the <entry-status> element of Contact Status:

- The CAB Client SHALL remove the <update> element associated with the Contact Entry from CAB User AB, when the CAB User consumes the updated contact.

- The CAB Client SHALL remove the <temporary> element associated with the Contact Entry, when the CAB User accept the temporary contact.
- The CAB Client SHALL remove the entire Contact Entry associated with the <temporary> element from the address book, when the CAB User rejects the temporary contact.

6.13 Authentication

6.13.1 Authentication for AB Synchronization

The CAB Client SHALL follow the authentication procedures described in [OMA DS Pro] sub clause 7 “Authentication”.

6.13.2 Authentication for XML Document Management

The CAB Client SHALL follow the authentication procedures described in [OMA XDMS Core] sub clause 5.1.1 “Authentication”.

7. CAB XDMS

The CAB XDMS(s) SHALL support the XDMS procedures described in [OMA XDM Core] “*Procedures at the XDM Server*”, and the Application Usages described in [CAB XDMS].

8. CAB Management Object

The CAB Management Object (MO) for configuration and provisioning of CAB Client is described in [CAB MO].

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 1.0 History

Document Identifier	Date	Sections	Description
Draft Versions OMA-TS-CAB-V1_0	07 May 2009	All	Initial draft
	03 Jul 2009	1,2,3,5,6, App. C	Incorporated the following CRs: OMA-MWG-CAB-2009-0190R01-INP_TS_Contents, OMA-MWG-CAB-2009-0191R01-CR_TS_Scope, OMA-MWG-CAB-2009-0192R01-CR_TS_References, OMA-MWG-CAB-2009-0201R01-CR_TS_Definitions, OMA-MWG-CAB-2009-0206R01-CR_TS_Appendix_Contact_Search_Flow, and OMA-MWG-CAB-2009-0208R01-CR_TS_Abbreviations.
	21 Jul 2009	4, App. C	Incorporated the following CRs: OMA-MWG-CAB-2009-0193R01-CR_TS_Introduction, and OMA-MWG-CAB-2009-0205R01-CR_TS_Appendix_Interaction_with_non_CAB_Address_Book_Systems_Flow_. Updated the flow diagrams in C.1.
	27 Jul 2009	6.1	Incorporated the following CR: OMA-MWG-CAB-2009-0236R01-CR_TS_Search.
	20 Aug 2009	2.1, 3.3, C.3, 4, 7, 8	Incorporated the following CRs: OMA-MWG-CAB-2009-0253R02-CR_TS_Model_Flow_AI037, OMA-MWG-CAB-2009-0264R01-CR_CAB_TS_Version, and OMA-MWG-CAB-2009-0265R01-CR_CAB_TS_new_sections_for_CAB_XDMS_and_MO.
	07 Sep 2009	2.1, 3.2, 4, 5, 5.6, 6.1, C.3	Incorporated the following CRs: OMA-MWG-CAB-2009-0267R01-CR_CAB_TS_Section_5 and OMA-MWG-CAB-2009-0274-CR_TS_OMA_XDM_reference.
	06 Oct 2009	3.2, 5.1, 5.2, 5.3, 5.4, C.2	Incorporated the following CRs: OMA-MWG-CAB-2009-0280R02-CR_CAB_data_Format, OMA-MWG-CAB-2009-0289R01-CR_Contact_Subscription_Function_CAB_TS, OMA-MWG-CAB-2009-0290R02-CR_Contact_Share_Function_CAB_TS, and OMA-MWG-CAB-2009-0291R01-CR_Interworking_Function_CAB_TS.
	23 Nov 2009	5.4.2, C.4, C.5	Incorporated the following CRs: OMA-MWG-CAB-2009-0303R02-CR_TS_ContactShareFlow, OMA-MWG-CAB-2009-0305R02-CR_ContactSubscription_with_SUP, and OMA-MWG-CAB-2009-0317R01-CR_Contact_Search_with_External_Directories.
	06 Jan 2010	6.2, 6.3, 6.4, 6.5, C.6	Incorporated the following CRs: OMA-MWG-CAB-2009-0352R02-CR_TS_was_284R01, OMA-MWG-CAB-2009-0373R01-CR_CAB_Client, and OMA-MWG-CAB-2009-0379R01-CR_CAB_Client_TS_Structure.

Document Identifier	Date	Sections	Description
	17 Feb 2010	2.1, 5.4.2, 6.2.2, 6.2.3, 6.5, 9, C.4.1.2, C.7	Corrected subclause numbers in 5.4.2. Incorporated the following CRs: OMA-MWG-CAB-2009-0390R04-CR_AI_069_HLF_013_Trigger_and_data_sharing, OMA-MWG-CAB-2010-0001R01-CR_Procedures_at_CAB_Client, OMA-MWG-CAB-2010-0013R02-CR_TS_CAB_AccessPermission_DocManagement, OMA-MWG-CAB-2010-0028R01-CR_Search_Document_Extension, OMA-MWG-CAB-2010-0046R01-CR_Flows_Contact_Share, OMA-MWG-CAB-2010-0052R01-CR_auth_for_AB_doc_management, and OMA-MWG-CAB-2010-0053R01-CR_auth_for_PCC_doc_management.
	22 Feb 2010	2.1, 5.4.3, B	Incorporated the following CRs: OMA-MWG-CAB-2010-0021R03-CR_TS_CAB_SCR_Table and OMA-MWG-CAB-2010-0074R01-CR_PCC_Data_Structure_Name_vCard_mapping.
	05 Mar 2010	5.1, 6.4, 9, C.4.1.1, C.7.2, All (editorial)	Incorporated the following CRs: OMA-MWG-CAB-2010-0060R01-CR_Procedures_for_HLF_012, OMA-MWG-CAB-2010-0075R01-CR_Client_Server_AB_synch, OMA-MWG-CAB-2010-0081R01-CR_Contact_Share_flows, and OMA-MWG-CAB-2010-0085R01-CR_CAB_TS_Core_clean_up_and_fixes,
	08 Mar 2010	6.2.2.3, C.4.1.1	Unincorporated the following CR: OMA-MWG-CAB-2010-0081R01-CR_Contact_Share_flows. Incorporated the following CRs: OMA-MWG-CAB-2010-0081R02-CR_Contact_Share_flows and OMA-MWG-CAB-2010-0099R01-CR_Client_contact_subscription_list.
	09 Mar 2010	All	Editorial cleanup.
	23 Apr 2010	3.2, 3.3, 5.1, 5.2, 5.3, 5.4.1, 5.5, 5.6, 9.2, C.4.1.1, C.4.1.2, 6.2.2.2	Incorporated the following CRs: OMA-COM-CAB-2010-0172-CR_CONR_Resolution_C005_C006_C042_C044_C047_C052_C057_C064_C0169_C0252, OMA-COM-CAB-2010-0174R01-CR_CONR_Resolution_C0205, OMA-COM-CAB-2010-0177-CR_CONRR_on_abbreviation_table, and OMA-COM-CAB-2010-0178R01-CR_solution_for_C037.
	03 May 2010	2.1, 3.2	Incorporated the following CR: OMA-COM-CAB-2010-0161R01-CR_CONR_Resolution_TS_Comments_C028_C029_C030.
	06 May 2010	2.1, 2.2, 5.1, 5.2, 5.3, 5.4.1, 5.4.2, 5.4.3, 5.5, C.6.2	Incorporated the following CRs: OMA-COM-CAB-2010-0152R02-CR_Comments_C071_to_C099_Section5_2_CAB_TS, OMA-COM-CAB-2010-0168R02-CR_CONR_Resolution_TS_C055_C056_C060_C063_C069, OMA-COM-CAB-2010-0192R01-CR_CONRR_TS_C018_to_C027, and OMA-COM-CAB-2010-0198-CR_CONRR_C0318_on_C6.2.
	17 May 2010	1, 2.1, 4, 5.1, 5.2, C.2	Incorporated the following CRs: OMA-COM-CAB-2010-0202R02-CR_CONR_Resolution_HLF_013, OMA-COM-CAB-2010-0205-CR_CONRR_TS_C010_to_C017, OMA-COM-CAB-2010-0206R01-CR_CONRR_TS_C034_to_C036, OMA-COM-CAB-2010-0207R01-CR_CONRR_TS_C046, OMA-COM-CAB-2010-0212R01-CR_CONR_Resolution_TS_Core_Appendix_C.2, and OMA-COM-CAB-2010-0215R02-CR_CONR_Resolution_CAB_TS_filter_subscription.
	07 Jun 2010	5.1	Incorporated the following CR: OMA-COM-CAB-2010-0236R01-CR_CONR_Resolution_C066.
	24 Jun 2010	6.2.2.3, 6.4, C.7	Incorporated the following CRs: OMA-COM-CAB-2010-0247-CR_CAB_User_List_clean_up, OMA-COM-CAB-2010-0248R01-CR_CONR_Resolution_TS_C222_C231, and OMA-COM-CAB-2010-0267R01-CR_CONRR_TS_C207_to_C212.

Document Identifier	Date	Sections	Description
	13 Jul 2010	5.1, 5.2, 5.4.1, 5.4.2, 5.4.2.1.3, 5.4.2.1.4, 5.4.2.1.5, 5.5, 6.2.2.1, 6.3.1, 6.3.2, 6.3.3, 6.3.4, 6.6, C.6.1, C.6.2	Incorporated the following CRs: OMA-COM-CAB-2010-0197R01-CR_CONRR_C0317_on_C6.1, OMA-COM-CAB-2010-0209R02-CR_Comments_Section5_2_part2_CAB_TS, OMA-COM-CAB-2010-0227R02-CR_CONRR_C0200_to_C0203_section_541_6221, OMA-COM-CAB-2010-0255R01-CR_resolution_AI_A022_TS_CAB, OMA-COM-CAB-2010-0272R01-CR_CONRR_C0137_C0141_C0143_C0147_on_5.4.2, OMA-COM-CAB-2010-0273R01-CR_CONRR_C0142_C0150_C0151_on_5.4.2.1.3, OMA-COM-CAB-2010-0276R01-CR_CONRR_TS_C217_to_C221, and OMA-COM-CAB-2010-0286R01-CR_TS_AB_Personalization.
	04 Aug 2010	2.1, 5.2, 5.4.2, 5.4.3, 6.1, C.4	Incorporated the following CRs: OMA-COM-CAB-2010-0253R02-CR_CONR_Contact_Search_C001_to_C013, OMA-COM-CAB-2010-0284R01-CR_CONR_Resolution_C0311, OMA-COM-CAB-2010-0304R01-CR_CONR_Resolution_for_C0160_C0165, OMA-COM-CAB-2010-0307R01-CR_CONR_CAB_TS_External_Directories_Search_section_5.4.2, OMA-COM-CAB-2010-0320-CR_action_item_031
	11 Aug 2010	5.4, 5.4.1, 5.4.3, 6.2.2.2, 6.4, C.2	OMA-COM-CAB-2010-0305R01-CR_CONR_Resolution_C054, OMA-COM-CAB-2010-0333-CR_CONR_Resolution_for_C0160_C0165_Second_Phase, OMA-COM-CAB-2010-0336R02-CR_CONR_Resolution_CAB_Client_Contact_Share, OMA-COM-CAB-2010-0337R02-CR_CONR_Resolution_Interworking_Import_non_CAB, OMA-COM-CAB-2010-0342-CR_CAB_TS_clean_up,
	13 Aug 2010	5.6, 9.1	OMA-COM-CAB-2010-0214R04-CR_CONR_Resolution_TS_section_9.2
	01 Sep 2010	5.2, 5.4.3, 5.6, 5.6.1.1, 5.6.1.2, 6.2.2.1, 6.2.2.2, 6.2.4, 9.0, Appendix B.1, Appendix B.2	OMA-COM-CAB-2010-0334R01-CR_CONR_Resolution_for_C0160_C0165_Third_phase, OMA-COM-CAB-2010-0340-CR_Core_TS_SCR_table_completion, OMA-COM-CAB-2010-0351R01-CR_reactive_authorization_e_vote_Option_B_CAB_TS, OMA-COM-CAB-2010-0357R01-CR_CAB_contact_add_data_via_SIP_Message, OMA-COM-CAB-2010-0367R01-CR_C216_CAB_TS, OMA-COM-CAB-2010-0379R01-CR_Action_Item_A035, OMA-COM-CAB-2010-0397R01-CR_Action_Item_A053
	07 Sep 2010	2.1, 3.2, 3.3, 5.1, 5.3, 5.3.1, 5.3.2, 5.4.3, 5.5, 5.6, 5.6.1, 6, 6.2.2.2, 6.2.4, 6.4, 6.6, 7, c.1.2, C.1.3, c.3, C.4.2, D	OMA-COM-CAB-2010-0335R01-CR_TS_Core_CONRR_resolution_AB_Synchronization_sections , OMA-COM-CAB-2010-0338R01-CR_Re_order_CAB_Client_sections_C0181_C0198, OMA-COM-CAB-2010-0349R01-CR_C118_C119_CAB_TS, OMA-COM-CAB-2010-0358R01-CR_TS_CAB_CONRR_C0282_solution, OMA-COM-CAB-2010-0359-CR_TS_CAB_CONRR_C0308_C0309_resolution, OMA-COM-CAB-2010-0364R01-CR_CONR_Resolution_for_C0160_C0165_Fourth_Phase, OMA-COM-CAB-2010-0365R02-CR_CONR_Resolution_C0164, OMA-COM-CAB-2010-0374R04-CR_CONR_Resolution_TS, OMA-COM-CAB-2010-0381R01-CR_CONRR_C051_and_Consistency, OMA-COM-CAB-2010-0383R01-CR_CONRR_C079_C0175_C0234, OMA-MWG-CAB-2010-0388R01-CR_TS_CONRR_C0174_shareByReference, OMA-COM-CAB-2010-0396R02-CR_CONR_Resolution_for_C0160_C0165_Fifth_Phase, OMA-COM-CAB-2010-0400-CR_CONR_Resolution_C281_C291
	09 Sep 2010	5.3.1, C.4.1.1, C.4.1.2, C.5	OMA-COM-CAB-2010-0374R05-CR_CONR_Resolution_TS

Document Identifier	Date	Sections	Description
	12 Sept 2010	5.2, 5.4.2.1.3, 5.6.3.1, 5.6.3.2, C.3.1, C.7.1	Remove Editor's Notes through all the sections with no action.
	14 Sept 2010	C.7	Removal of Section C.7
Candidate Version: OMA-TS-CAB-V1_0	19 Oct 2010	All	Status changed to Candidate by TP: OMA-TP-2010-0431-INP_CAB_V1_0_ERP_for_Candidate_Approval Editorial fixes

Appendix B. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [SCRRULES].

The SCRs defined in the following tables includes SCR for:

- CAB Client
- CAB Server

B.1 SCR for CAB Client

Note: All the Requirements that prefix with “XDM Core” are a reference to the [XDM Core] specification.

Item	Function	Reference	Requirement
CAB-CS-C-001-M	Support for constructing and sending the contact search request to CAB AB App Usage	Section 6.1	XDM_Core-SRC-C-001-O XDM_Core-SRC-C-002-O
CAB-CS-C-002-M	Support for constructing and sending the contact search request to CAB PCC App Usage	Section 6.1	XDM_Core-SRC-C-001-O XDM_Core-SRC-C-002-O
CAB-CS-C003-M	Support for constructing and sending the contact search request to the External Directories	Section 6.1	XDM_Core-SRC-C-002-O XDM_Core-SRC-C-003-O
CAB-DM-C-001-M	Support for managing CAB PCC XML document in PCC Application Usage	Section 6.2.1	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-DM-C-002-Y	Support for managing CAB User Preferences in CAB User Preferences App Usage	Section 6.2.2	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-DM-C-003-M	Support for managing import requests in CAB Feature Handler App Usage	Section 6.2.2.1	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-DM-C-004-M	Support for managing import responses in CAB Feature Handler App Usage	Section 6.2.2.1	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M CAB-SUB-C-003-M CAB-SUB-C-006-M
CAB-DM-C-005-M	Support for managing Contact Share requests in CAB Feature Handler App Usage	Section 6.2.2.2	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-DM-C-006-M	Support for managing Contact Share responses and delivery report in CAB Feature Handler	Section 6.2.2.2	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M

Item	Function	Reference	Requirement
	App Usage		CAB-SUB-C-003-M CAB-SUB-C-006-M
CAB-DM-C-007-M	Support for managing Contact Subscription list in CAB User Preferences App Usage	Section 6.2.2.3	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-DM-C-008-M	Support for managing AB Access Permissions	Section 6.2.3.1	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-DM-C-009-M	Support for managing PCC Access Permissions	Section 6.2.3.2	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-SUB-C-001-M	Support for subscribing to CAB User's own PCC document changes	Section 6.3.1	XDM_Core-SUB-C-001-O, XDM_Core-SUB-C-002-O, XDM_Core-SUB-C-003-O
CAB-SUB-C-002-M	Support for subscribing to to and receiving notifications from CAB's own CAB User User Preferences document changes	Section 6.3.2	XDM_Core-SUB-C-001-O, XDM_Core-SUB-C-002-O, XDM_Core-SUB-C-003-O
CAB-SUB-C-003-M	Support for subscribing to and receiving notifications from CAB User's own CAB Feature Handler document changes	Section 6.3.3	XDM_Core-SUB-C-001-O, XDM_Core-SUB-C-002-O, XDM_Core-SUB-C-003-O
CAB-SUB-C-004-M	Support for subscribing to and receiving notifications from CAB PCC document changes via Subscription Proxy	Section 6.3.4	XDM_Core-SUB-C-001-O, XDM_Core-SUB-C-002-O, XDM_Core-SUB-C-003-O
CAB-SUB-C-005-M	Support for subscribing to and receiving notifications from CAB User Preferences document changes via Subscription Proxy	Section 6.3.4	XDM_Core-SUB-C-001-O, XDM_Core-SUB-C-002-O, XDM_Core-SUB-C-003-O
CAB-SUB-C-006-M	Support for subscribing to and receiving notifications from CAB Feature Handler document changes via Subscription Proxy	Section 6.3.4	XDM_Core-SUB-C-001-O, XDM_Core-SUB-C-002-O, XDM_Core-SUB-C-003-O
CAB-SYNC-C-001-M	Support for synchronization with CAB Server using CAB-1 interface (OMA DS)	Section 6.4	CAB-SYNC-C-002-O, OR CAB-SYNC-C-003-O CAB-SYNC-C-004-M
CAB-SYNC-C-002-O	Support for AB synchronization with CAB Server using CAB Format	Section 6.4	

Item	Function	Reference	Requirement
CAB-SYNC-C-003-O	Support for AB synchronization with CAB Server using Legacy Format(s)	Section 6.4	
CAB-SYNC-C-004-M	Support for receiving 'Server-Alerted Sync'	Section 6.4	
CAB-AUT-C-001-M	Support for authentication for synchronization (OMA DS)	Section 6.5.1	
CAB-AUT-C-001-M	Support for authentication for XML document management (OMA XDM)	Section 6.5.2	XDM_Core-SEC-C-001-M XDM_Core-SEC-C-002-M XDM_Core-SEC-C-003-O
CAB-COS-C-001-M	Support for Contact Status information	Section 6.6	CAB-SYNC-C-001-M

B.2 SCR for CAB Server

Item	Function	Reference	Requirement
CAB-SYNC-S-001-M	Support for synchronization with CAB Client using CAB-1 interface (OMA DS)	Section 5.1	CAB-SYNC-S-002-M CAB-SYNC-S-003-O CAB-SYNC-S-004-O CAB-SYNC-S-005-M
CAB-SYNC-S-002-M	Support for AB synchronization with CAB Client using CAB Format	Section 5.1	
CAB-SYNC-S-003-O	Support for AB synchronization with CAB Client using Legacy Format(s)	Section 5.1	
CAB-SYNC-S-004-O	Support for Format Adaptation	Section 5.1 Section 5.4.3	
CAB-SYNC-S-005-M	Support for sending 'Server-Alerted Sync'	Section 5.1	
CAB-SYNC-S-006-M	Support for managing AB document in AB App Usage	Section 5.1 Section 5.5	CAB-XDMA-S-001-M
CAB-CSUB-S-001-M	Support for obtaining (i.e. using document management or subscription to changes) the list of contact subscriptions from CAB User's subscription list in the CAB User Preferences App Usage	Section 5.2, Section 5.5	CAB-XDMA-S-001-M
CAB-CSUB-S-002-O	Support for handling the contact subscription	Section 5.2	

Item	Function	Reference	Requirement
	request with <filter-set>		
CAB-CSUB-S-003-M	Support for handling the contact (to be subscribed to) who is not a contact in AB XDMS	Section 5.2	
CAB-CSUB-S-004-M	Support for subscribing to CAB PCC document changes	Section 5.2, Section 5.5	CAB-XDMA-S-001-M, CAB-CSUB-S-005-M, CAB-CSUB-S-006-M
CAB-CSUB-S-005-M	Initial subscription using SIP SUBSCRIBE message for CAB PCC document changes	Section 5.2, Section 5.5	XDM_Core-SUB-A-001-O
CAB-CSUB-S-006-M	Processing Received Notification for CAB PCC document changes (XDM v2.0)	Section 5.2, Section 5.5	XDM_Core-SUB-A-002-O
CAB-CSUB-S-07-M	Support for storing or updating CAB AB XML document using XDM document management operation with resulting data from Contact Subscriptions, and updating of contact subscription status	Section 5.2, Section 5.5	CAB-XDMA-S-001-M
CAB-CSF-S-001-M	Support for obtaining the Contact Share request (i.e. using document management or subscription to changes) from CAB Feature Handler App Usage	Section 5.3, Section 5.5	CAB-XDMA-S-001-M
CAB-CSF-S-002-M	Support for determining the recipient type, i.e. CAB or non-CAB	Section 5.3	
CAB-CSF-S-003-M	Support for retrieving the data from CAB PCC App Usage to be shared with non-CAB recipients	Section 5.3, Section 5.5	CAB-XDMA-S-001-M
CAB-CSF-S-004-M	Support for retrieving the data from CAB AB App Usage to be shared with non-CAB recipients	Section 5.3, Section 5.5	CAB-XDMA-S-001-M
CAB-CSF-S-005-M	Support for constructing the message defined in [OMA CPM CONV FCT TS]	Section 5.3	
CAB-CSF-S-006-M	Support for sending the message defined in [OMA CPM CONV FCT TS] towards the CPM Interworking Selection Function	Section 5.3	

Item	Function	Reference	Requirement
	[OMA CPM IWF TS]		
CAB-CSF-S-007-M	Support for initiating XDM Forward message to the recipient CAB User	Section 5.3 Section 5.5	CAB-XDMA-S-001-M
CAB-CSF-S-008-M	Supporting for storing the response and delivery report in CAB Feature Handler App Usage	Section 5.3 Section 5.5	CAB-XDMA-S-001-M
CAB-IWF-S-001-M	Support for obtaining the request to import the contacts from non-CAB system (i.e. using document management or subscription to changes) from CAB Feature Handler App Usage	Section 5.4.1, Section 5.5	CAB-XDMA-S-001-M
CAB-IWF-S-002-M	Supporting the storage of imported contacts in AB Application Usage and updating the contact status	Section 5.4.1 Section 5.5	CAB-XDMA-S-001-M
CAB-CSF-S-003-M	Supporting for storing the response and delivery report in CAB Feature Handler App Usage	Section 5.4.1 Section 5.5	CAB-XDMA-S-001-M
CAB-IWF-S-004-Y	Support for receiving and processing Search request via XDM-7i (Limited XQuery over HTTP) (XDM v2.1) for External Directories Search	Section 5.4.2	XDM_Core-SRC-S-001-O, XDM_Core-SRC-S-003-O
CAB-IWF-S-005-M	Support for CAB extension <dataSource> to the Search Document for External Directories Search	Section 5.4.2.1.5	
CAB-IWF-S-006-M	Support for translating/mapping the received Search request (XDM-7i) to the external Search request based upon the format supported by the External Directories	Section 5.4.2	
CAB-IWF-S-007-M	Support for generating the response to Search Proxy via XDM-7i (XDM v2.1)	Section 5.4.2	XDM_Core-SRC-S-001-O, XDM_Core-SRC-S-003-O
CAB-IWF-S-008-Y	Support for the	Section 5.4.3	

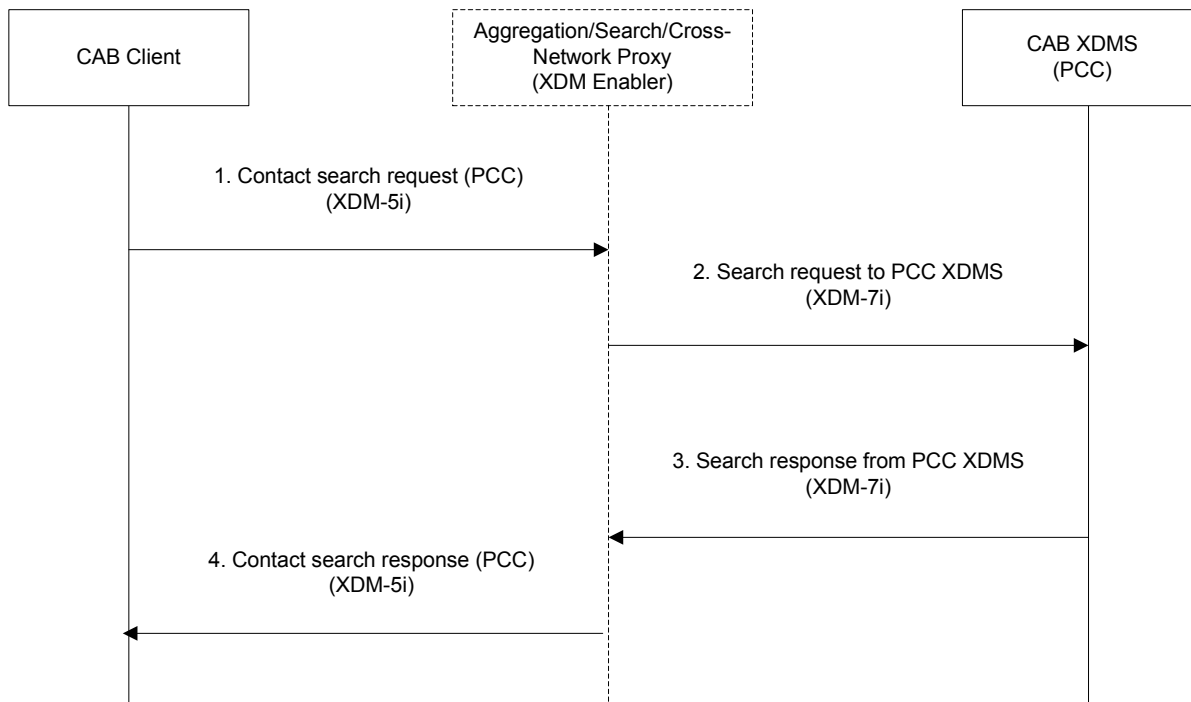
Item	Function	Reference	Requirement
	formatadaptation between CAB format and Legacy format(s)		
CAB-XDMA-S-001-M	Support for XDM Agent	Section 5.5	XDM_Core-XOP-A-001-M XDM_Core-XOP-A-002-M XDM_Core-XOP-A-003-M XDM_Core-SUB-A-001-O XDM_Core-SUB-A-002-O XDM_Core-SEC-A-003-M XDM_Core-RES-A-001-O
CAB-COS-S-001-M	Support for managing the Contact Status information	Section 5.6	
CAB-COS-S-002-M	Support for generating and sending SIP Message with Contact Added data to a remote domain	Section 5.6	
CAB-COS-S-003-M	Support for receiving SIP Message with Contact Added data from a remote domain	Section 5.6	

Appendix C. Flows (Informative)

C.1 Contact Search

The following provides three basic message flows for contact search based on the target destination i.e. PCC, AB, and External directories. Any combination of these flows may be employed by the CAB Enabler implementations.

C.1.1 Contact Search - PCC



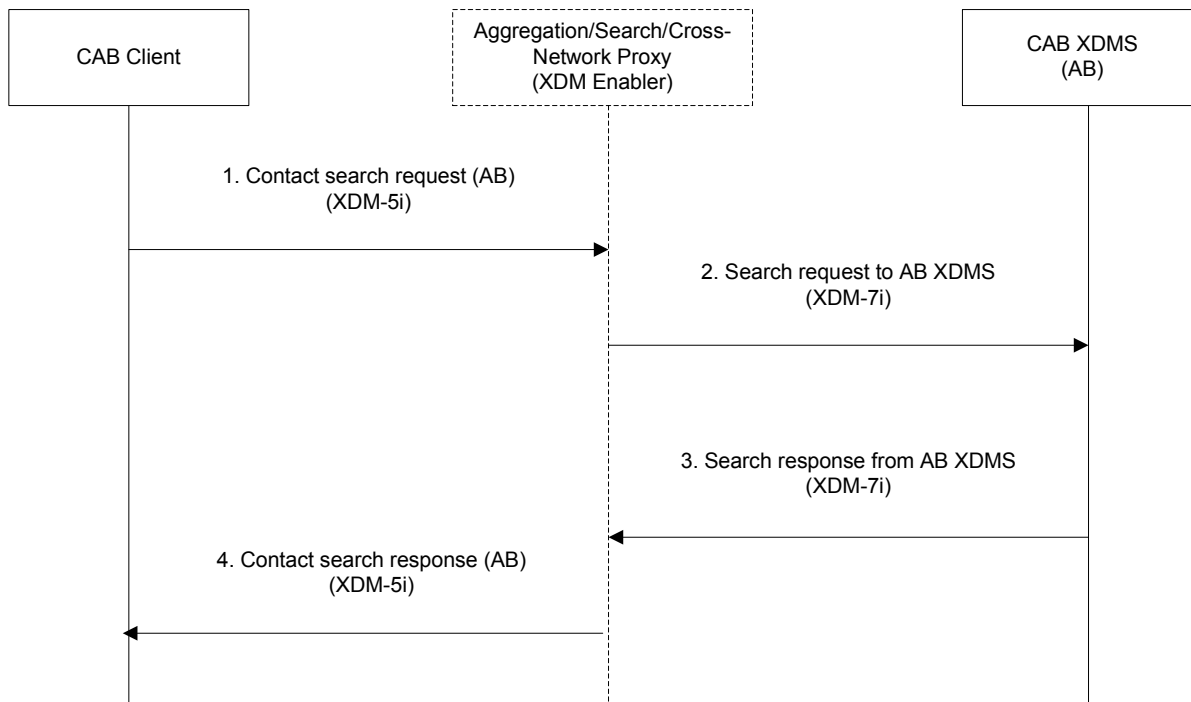
Step1: The CAB Client makes a contact search request to CAB Users’ PCC(s) via the XDM Enabler Proxies (Aggregation Proxy, Search Proxy, Cross-Network Proxy) using Limited XQuery. The request is formulated based on the PCC XML schema hosted by the PCC XDMS. This request is based on XDM-5i.

Step 2: Upon receiving the CAB Client request, the Search proxy routes the request to the PCC XDMS as indicated in the original request.

Step 3: The search response including the search results from the PCC XDMS is delivered in the response to the Search proxy. The search results may be further formatted or aggregated in the Search proxy. The PCC search results are subject to CAB User’s PCC Access Permissions.

Step 4: The contact search response (PCC) is sent back to the CAB Client which includes the list of results corresponding to the initial search request in Step1. The response is based on XDM-5i.

C.1.2 Contact Search - AB



Step 1: The CAB Client makes a contact search request to the CAB User's own AB via the XDM Enabler Proxies (Aggregation Proxy, Search Proxy, Cross-Network Proxy) using Limited XQuery. The request is formulated based on the AB XML schema hosted by the AB XDMS. This request is based on XDM-5i.

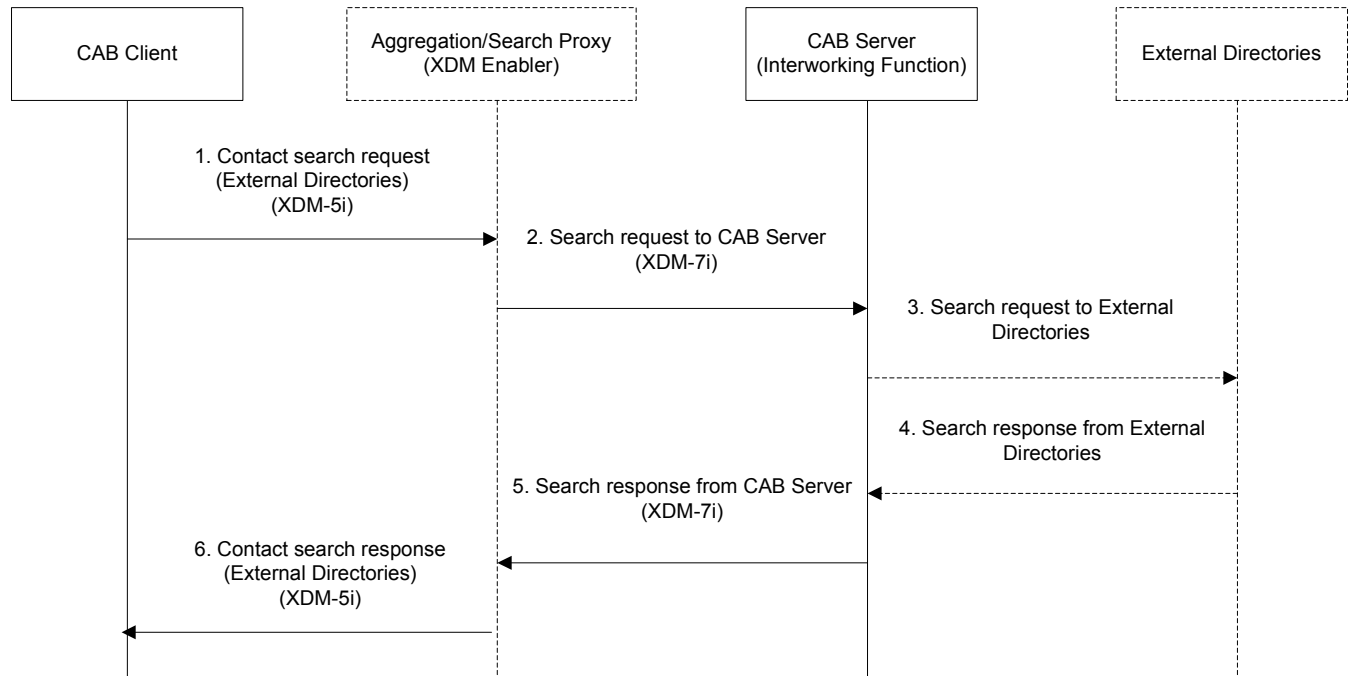
Step 2: Upon receiving the CAB Client request, the Search proxy routes the request to the AB XDMS as indicated in the original request.

Step 3: The search response including the search results from the AB XDMS is delivered in the response to the Search proxy. The search results may be further formatted or aggregated in the Search proxy. The AB search results are subject to CAB User's AB Access Permissions.

Step 4: The contact search response (AB) is sent back to the CAB Client which includes the list of results corresponding to the initial search request in Step 1. The response is based on XDM-5i.

Note: the format of the search request for AB needs to identify the target as a specific CAB User's AB XML document, to satisfy the Use Case where the CAB User A might have given Access permissions to other CAB User B to search the CAB User A AB data.

C.1.3 Contact Search – External Directories



Step 1: The CAB Client makes a contact search request towards External Directories via the Aggregation/Search Proxy using Limited XQuery. The request is formulated based on the standard XML search format (for External directories) hosted by the Interworking Function. This request is based on XDM-5i.

Step 2: Upon receiving the CAB Client request, the Search proxy routes the request to the CAB Server (i.e. Interworking Function) as indicated in the original request via XDM-7i. The Interworking Function translates the standard XML search request to external search request. The interactions and mapping between standard XML search request and external search request is out of scope.

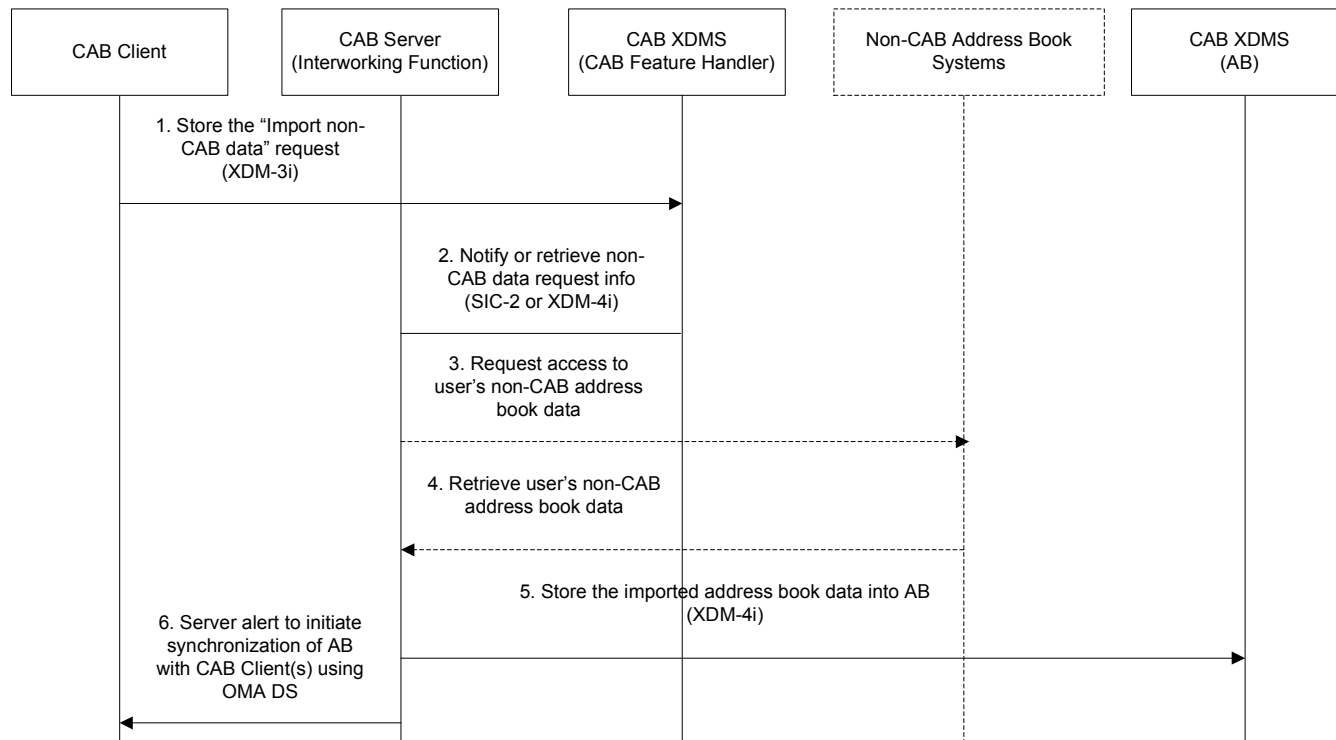
Step 3: The Interworking Function initiates the translated search request(s) to the External Directories over the appropriate interface(s). The translation and initiation of search requests over such external interfaces is out of scope.

Step 4: The search responses from the External Directories are received by the Interworking Function. If more than one search responses are received, the Interworking Function shall aggregate all responses into one search result. The aggregation of responses based on the external interfaces and data models is out of scope.

Step 5: The Interworking Function returns the search result(s) from External Directories to the Search proxy. The response is based on XDM-7i. The search result(s) may be further formatted or composed in the Search proxy.

Step 6: The contact search response (External Directories) is sent back to the CAB Client and includes the result(s) corresponding to the initial search request in Step1. The response is based on XDM-5i.

C.2 Import from non-CAB Address Book Systems



Step 1: The CAB Client makes an “import non-CAB AB data” request by writing/storing the request information formatted to the <import-non-cab> element as described in the CAB Feature Handler Application Usage [CAB XDMS].

Step 2: The CAB Server retrieves the “import non-CAB AB data” request information from the CAB Feature Handler App Usage by either of the following methods:

Step 2a: Feature Handler App Usage notifies a change to the feature handler XML document to the CAB Server assuming a prior subscription to the changes is in place by the CAB Server.

Step 2b: Feature Handler App Usage is polled by the CAB Server for changes in the feature handler XML document based on a pre-determined time interval using XCAP.

Step 3: The CAB Server (i.e. Interworking Function) on behalf of the CAB User requests the non-CAB address book system(s) access to the CAB User’s legacy address book data, by supplying the necessary access parameters.

Step 4: Upon obtaining the access, the CAB Server (i.e. Interworking Function) retrieves/receives the non-CAB address book data of the CAB User from the non-CAB address book system(s).

Step 4a: The CAB Server (i.e. Interworking Function) transforms the imported data into the CAB Format.

Step 5: The CAB Server stores the resulting data in the AB Application Usage [CAB XDMS] subject to CAB User’s preferences.

Step 6: The data in the CAB User’s AB is then subsequently synchronized with the CAB Client(s) using OMA DS server-alerted notification sent by DS Server to CAB Client to initiate synchronization.

C.3 Sample XCAP flows for management of CAB XML documents

The flows in figure below describe the management operations on the data in the CAB XML documents, based on [OMA XDM Core] document management operations: create, retrieve, update, delete.

Note: The AB XML document data management operations are captured in the flows sub clause 6.1 “Address Book Management and Synchronization”. The flows in this section apply to all the CAB XML documents except the AB XML document.

C.3.1 XCAP operations on CAB XML documents

The management operations on the CAB XML documents are realized through XCAP operations as described in Appendix C of [OMA XDM Core]. This example describes the message flows used by CAB Client to manipulate a CAB XML document* in CAB XDMS(s) after authentication.

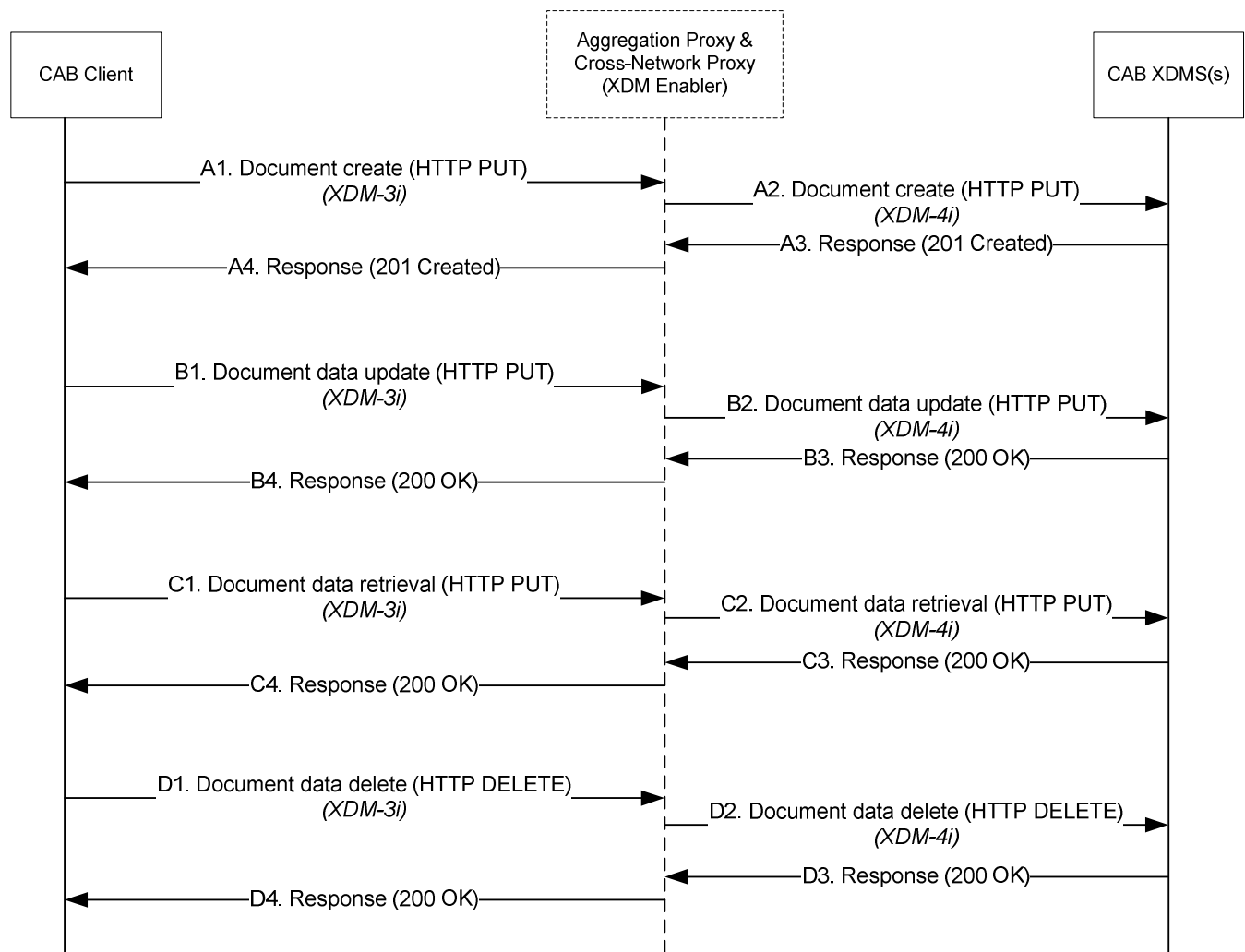


Figure 2: CAB Client manipulating a CAB XML document

NOTE 1 : The request messages (A, B, C, D) are shown in one diagram for the convenience of the reader, but there is no implication that all of them have to be performed.

NOTE 2: The Cross-Network Proxy is present in the flows in the case of an Authorized Principal managing PCC XDMS documents from a remote domain.

Description of steps below use as example operations on CAB User Preferences document:

Operation A: Creation of a document in CAB XDMS(s)

A1) The CAB Client sends an HTTP PUT request using the XDM-3i interface, via the Aggregation Proxy/Cross-Network Proxy to create a new CAB document “index” for a user in any of the CAB XDMS(s).

The example below is an HTTP PUT operation on the CAB User Preferences document that is owned by user with XUI of “sip:jobloggs@example.com” in the example.com domain.

```
PUT /cab-user-prefs/users/sip:jobloggs@example.com/index HTTP/1.1
Host: xcap.example.com
...
Content-Type: application/<TBD> +xml; charset="utf-8"
Content-Length: (...)

<?xml version="1.0" encoding="UTF-8"?>
<cab-user-prefs xmlns="urn:oma+xml:cab:cab-user-prefs">
  <FFS name="X">
    <ABC> </ABC>
  </FFS>
</ cab-user-prefs >
```

A2) Assuming that the CAB Client was successfully authenticated, the CAB XDMS(s) receive the request over the XDM-4i interface from the XDM proxies (Aggregation Proxy/Cross-Network proxy).

A3) The CAB XDMS(s) acknowledge the creation of the index document with a HTTP “201 Created” message, assuming that the CAB Client had the right Access Permissions to perform the create operation and the operation was successful.

```
HTTP/1.1 201 Created
Etag: "cdcdcdcd"
...
Content-Length: 0
```

A4) The HTTP “201 Created” message is received by the CAB Client.

Operation B: Document data update in CAB XDMS(s)

B1) The CAB Client sends a HTTP PUT request over the XDM-3i interface, via the Aggregation Proxy/Cross-Network Proxy to the just-created “index” document in “sip:jobloggs@example.com”’s home directory to add a new <ABC> sub-element “YZW” to the <FFS> element identified as “X”.

```
PUT /cab-user-prefs/users/sip:jobloggs@example.com/index /~/cab-user-prefs
/FFS5B@name=%22X%22%5D/ABC5B@=%22YZW%22%5D HTTP/1.1
Host: xcap.example.com
...
Content-Type: application/xcap-el+xml; charset="utf-8"
Content-Length: (...)

<ABC="YZW"></ABC>
```

B2) Assuming that the CAB Client was successfully authenticated, the CAB XDMS(s) receive the request over the XDM-4i interface from the XDM proxies (Aggregation Proxy/Cross-Network proxy).

B3) The CAB XDMS(s) acknowledge the data update request of the index document with a HTTP “200 OK” reply, assuming that the CAB Client had the right Access Permissions to perform the update operation and the operation was successful.

```
HTTP/1.1 200 OK
Etag: "efefefef"
...
Content-Length: 0
```

B4) The HTTP “200 OK” message is received by the CAB Client.

Operation C: Document data retrieval from CAB XDMS(s)

C1) The CAB Client sends a HTTP GET request over the XDM-3i interface, via the Aggregation Proxy/Cross-Network Proxy to retrieve “sip:joebloggs@example.com”’s the FFS “X” from CAB User preference XDMS.

```
GET /cab-user-prefs/users/sip:joebloggs@example.com/index/~/cab-user-prefs/FFS5B@name=%22X%22%5D
HTTP/1.1
Host xcap.example.com
```

C2) Assuming that the CAB Client was successfully authenticated, the CAB XDMS(s) receive the request over the XDM-4i interface from the XDM proxies (Aggregation Proxy/Cross-Network proxy).

C3) The CAB XDMS(s) returns the data in the body of an HTTP “200 OK” reply, assuming that the CAB Client had the right Access Permissions to perform the retrieval operation and the operation was successful.

```
HTTP/1.1 200 OK
...
Etag: "efefefef"
Content-Type: application/xcap-el+xml; charset="utf-8"
Content-Length: (...)

<FFS name="X">
  <ABCYZW</ABC>
</FFS
```

C4) The HTTP “200 OK” message is received by CAB Client.

Operation D: Document data deletion from CAB XDMS(s)

D1) The CAB Client sends a HTTP DELETE request over the XDM-3i interface, via the Aggregation Proxy/Cross-Network Proxy to delete an <ABC> identified by the “YZW” from “sip:joebloggs@example.com”’s FFS “X” in the CAB User preference XDMS.

```
DELETE /cab-user-prefs/users/sip:joebloggs@example.com/index/~/cab-user-prefs/
FFS5B@name=%22X%22%5D/ABC5B@=%YZW%22%5D HTTP/1.1
Host: xcap.example.com
```

D2) Assuming that the CAB Client was successfully authenticated, the CAB XDMS(s) receive the delete request over the XDM-4i interface from the XDM proxies (Aggregation Proxy/Cross-Network proxy).

D3) The CAB XDMS(s), after checking the Access Permissions of the CAB Client, perform the deletion and acknowledges it by returning the body of an HTTP “200 OK” reply.

```
HTTP/1.1 200 OK
Etag: "ghghgh"
...
Content-Length: 0
```

D4) The HTTP “200 OK” message is received by CAB Client.

C.4 Sample Contact Share flows

The sample flows below capture the Contact Share operations covering all supported scenarios:

- Sending Side:

- Contact Share towards a CAB User
- Contact Share towards a non-CAB User

- Receiving side

- Contact Share received by a CAB User.

The CAB User data that can be shared by the CAB Enabler is:

- AB data (Contact Entries)
- PCC data (Contact Views)

The sample flows below use the case where AB Contact Entries or PCC Contact Views are the subject of Contact Share data.

Note: The determination of the recipient type (.i.e. if a CAB User or not), it is out of scope of the CAB specifications. There are many ways to achieve it, but none is mandated by the CAB Enabler. This applies to both flows (CAB to CAB and CAB to non-CAB) between steps 4 and 5.

C.4.1 Contact Share towards a CAB User

C.4.1.1 Originating Side

C.4.1.1.1 AB forwarding

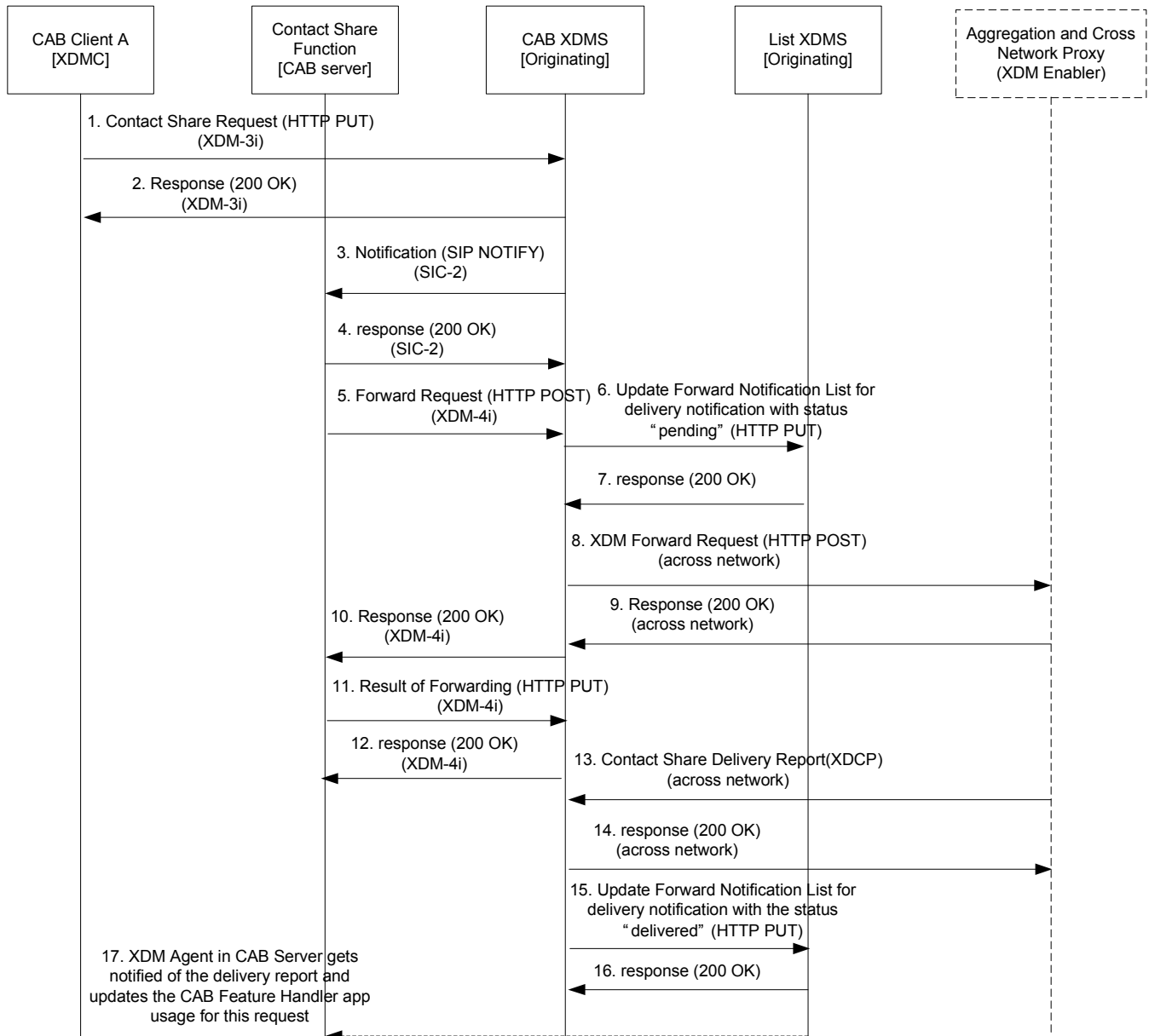


Figure 3 : Flows of AB Forwarding in the originating side towards a CAB user

Step 1; CAB Client A performs a HTTP PUT containing Contact Share data in CAB Feature Handler document in the CAB XDMs, using XDM-3i interface

- Step 2: CAB XDMS (Feature Handler Application Usage) sends the response back to CAB Client using XDM-3i interface.
- Step 3: CAB XDMS (AB Application Usage) notifies Contact Share Function (through the CAB Server's XDM Agent) about document changes using SIC-2 interface.
- Note: alternatively, the CAB Server can poll via XDM-3i the CAB Feature Handler document and perform the steps below after detecting a change in the data.
- Step 4: Contact Share Function sends the response (200 OK) to CAB XDMS (CAB Feature Handler Application Usage) using SIC-2 interface
- Step 5: Contact Share Function uses the CAB Server's XDM Agent to initiate an XDCCP Request (HTTP POST) for forwarding Contact Share data to the corresponding CAB XDMS through the XDM-4i interface.
- Step 6: CAB XDMS (AB application usage) updates the Forwarding Notification List of CAB Client A for delivery notification with the status "pending".
- Step 7: CAB XDMS (AB application usage) updates the Forwarding Notification List by adding the <delivery-notification> entry received and sends 200 OK response.
- Step 8: Since the recipient is in different domain CAB XDMS (AB Application Usage) creates the Contact Share data to be forwarded in a temporary storage and then does the XDCCP Forward Remote request to the recipients' CAB XDMS(s), using the XDM Enabler interfaces (e.g. XDM-8.2i) as described in the [OMA XDM AD].
- Step 9: CAB XDMS (AB Application Usage) of CAB Client A receives the response (200 OK) with result of forwarding back from the terminating network.
- Step 10: CAB XDMS (i.e. AB Application Usage) of CAB Client A on receiving the remote forward response creates 200 OK response to the forward requested received from CAB Client.
- Step 11: Contact Share Function performs an XCAP PUT using XDM-4i containing the result of forwarding in the <response> element of the CAB Feature Handler document in the CAB XDMS.
- Step 12: CAB Feature Handler from the CAB XDMS responds with 200 OK response back to Contact Share Function using XDM-4i interface.
- Step 13: Since CAB Client A has requested for the Delivery Report, CAB XDMS (AB Application Usage) of CAB Client A receives the Contact Share Delivery Report request (XDCCP) from the terminating network.
- Step 14: CAB XDMS (AB Application Usage) of CAB Client A sends 200 OK response with the XDCCP Response containing <done> element.
- Step 15: CAB XDMS (AB Application Usage) of CAB Client A updates the Forwarding Notification List entry created in the Step 6 of this flow by changing the status attribute value to "delivered"
- Step 16: The List XDMS sends 200 OK response.
- Step 17: CAB Server of User A would get notified about the contact share delivery status (the subscription to Forwarding Notification List document changes for CAB User's Forwarding Notification List document from the CAB server is assumed to have occurred prior to the notification). The CAB Server updates the delivery status in the CAB Feature Handler document for the corresponding request.

C.4.1.1.2 PCC forwarding

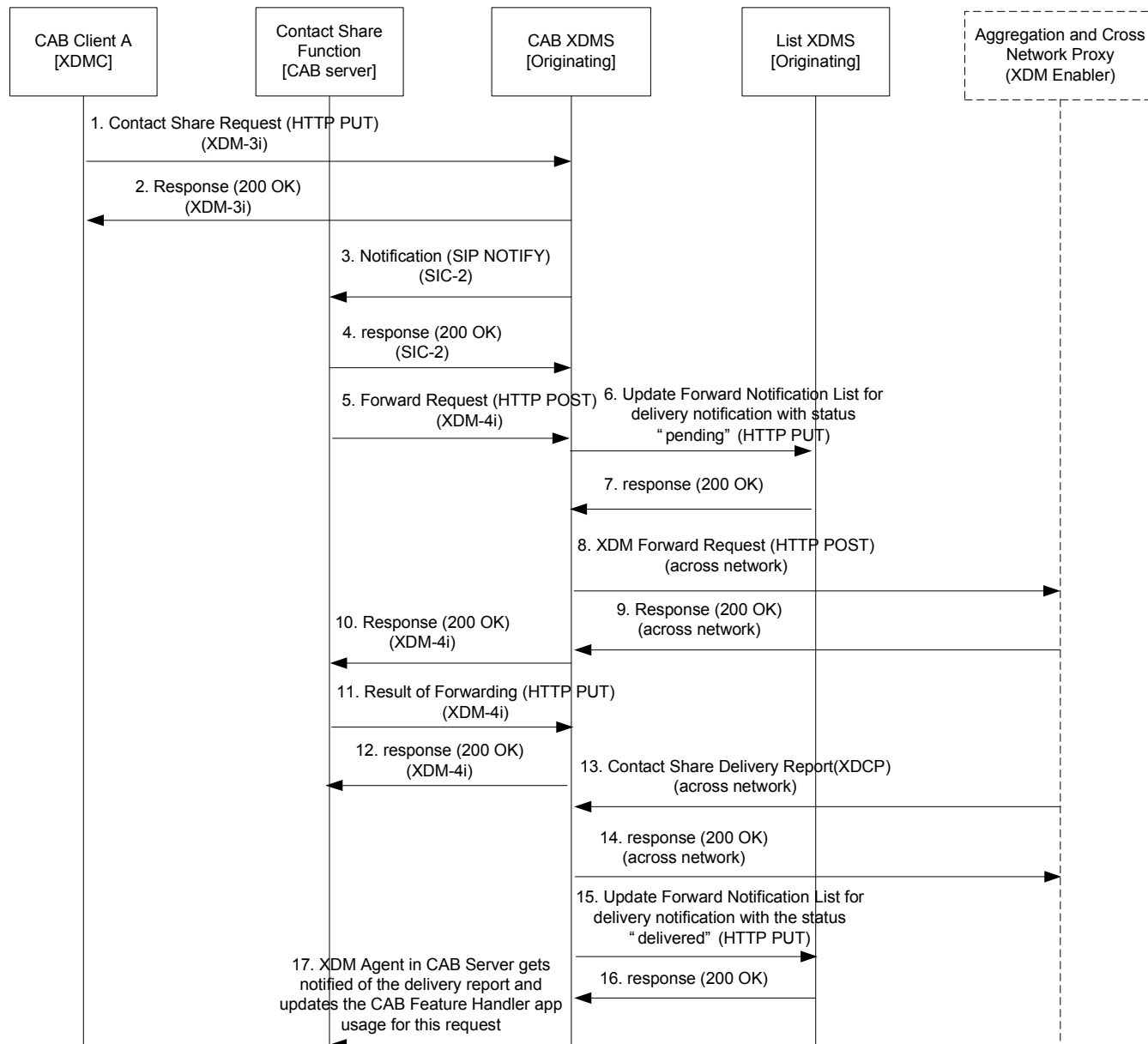


Figure 4: Flows of PCC forwarding in the originating side towards a CAB user

Step 1: CAB Client A performs a HTTP PUT containing Contact Share data in CAB Feature Handler document in the CAB XDMS, using XDM-3i interface

Step 2: CAB XDMS (Feature Handler Application Usage) sends the response back to CAB Client using XDM-3i interface.

Step 3: CAB XDMS (PCC Application Usage) notifies Contact Share Function (through the CAB Server’s XDM Agent) about document changes using SIC-2 interface.

Note: alternatively, the CAB Server can poll via XDM-3i the CAB Feature Handler document and perform the steps below after detecting a change in the data.

- Step 4: Contact Share Function sends the response (200 OK) to CAB XDMS (CAB Feature Handler Application Usage) using SIC-2 interface
- Step 5: Contact Share Function uses the CAB Server's XDM Agent to initiate an XDMP Request (HTTP POST) for forwarding Contact Share data to the corresponding CAB XDMS through the XDM-4i interface.
- Step 6: CAB XDMS (PCC application usage) updates the Forwarding Notification List of CAB Client A for delivery notification with the status "pending".
- Step 7: CAB XDMS (PCC application usage) updates the Forwarding Notification List by adding the <delivery-notification> entry received and sends 200 OK response.
- Step 8: Since the recipient is in different domain CAB XDMS (PCC Application Usage) creates the Contact Share data to be forwarded in a temporary storage and then does the XDMP Forward Remote request to the recipients' CAB XDMS(s), using the XDM Enabler interfaces (e.g. XDM-8.2i) as described in the [OMA XDM AD].
- Step 9: CAB XDMS (PCC Application Usage) of CAB Client A receives the response (200 OK) with result of forwarding back from the terminating network.
- Step 10: CAB XDMS (i.e. PCC Application Usage) of CAB Client A on receiving the remote forward response creates 200 OK response to the forward requested received from CAB Client.
- Step 11: Contact Share Function performs an XCAP PUT using XDM-4i containing the result of forwarding in the <response> element of the CAB Feature Handler document in the CAB XDMS.
- Step 12: CAB Feature Handler from the CAB XDMS responds with 200 OK response back to Contact Share Function using XDM-4i interface.
- Step 13: Since CAB Client A has requested for the Delivery Report, CAB XDMS (PCC Application Usage) of CAB Client A receives the Contact Share Delivery Report request (XDMP) from the terminating network.
- Step 14: CAB XDMS (PCC Application Usage) of CAB Client A sends 200 OK response with the XDMP Response containing <done> element.
- Step 15: CAB XDMS (PCC Application Usage) of CAB Client A updates the Forwarding Notification List entry created in the Step 6 of this flow by changing the status attribute value to "delivered"
- Step 16: The List XDMS sends 200 OK response.
- Step 17: CAB Server of User A would get notified about the contact share delivery status (the subscription to Forwarding Notification List document changes for CAB User's Forwarding Notification List document from the CAB server is assumed to have occurred prior to the notification). The CAB Server updates the delivery status in the CAB Feature Handler document for the corresponding request.

C.4.1.2 Terminating Side

C.4.1.2.1 AB forwarding (Accept)

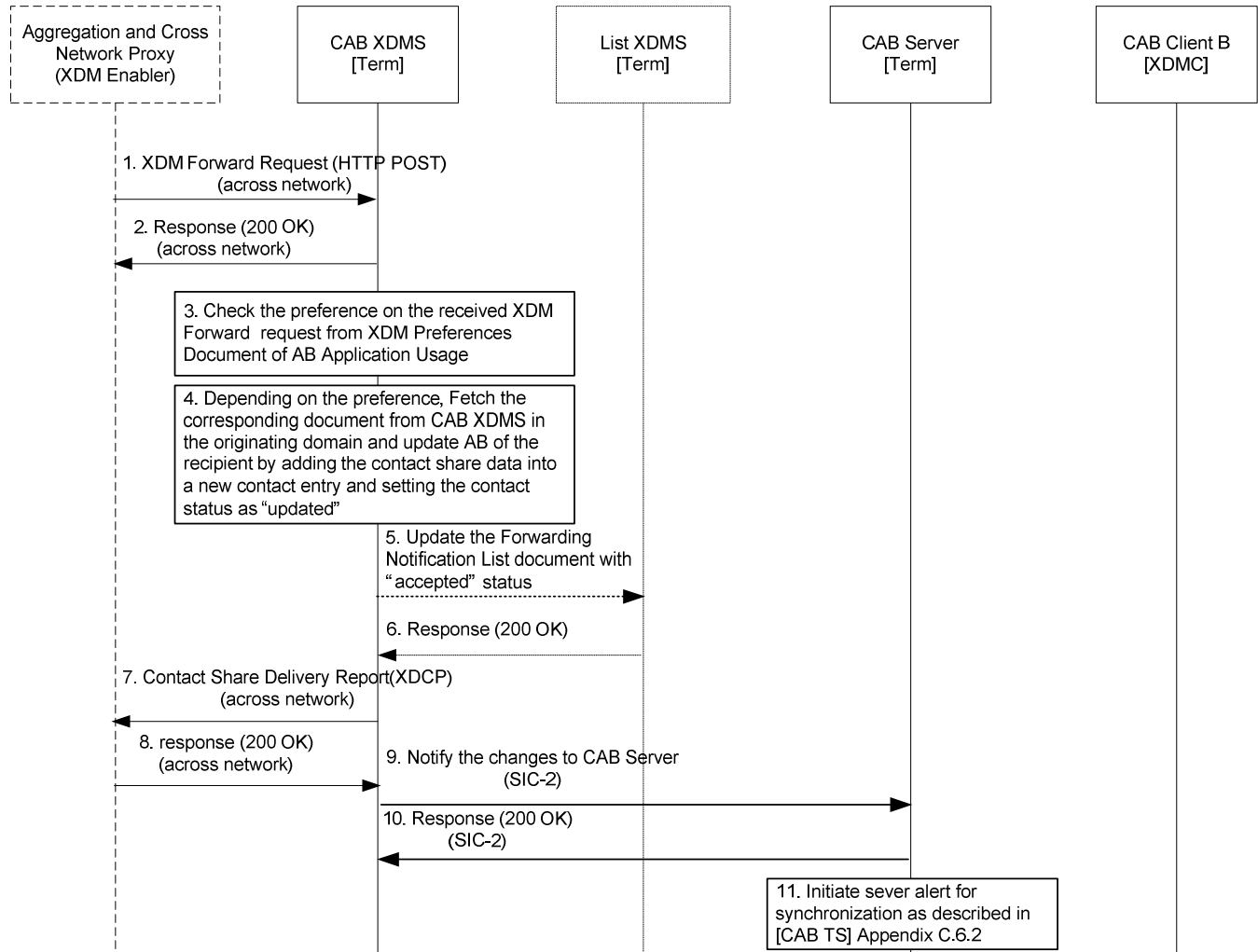


Figure 5: Flows of AB forwarding in the terminating side towards a CAB user

- Step 1: CAB XDMS (AB Application Usage) of CAB Client B receives the XDCP Forward Remote request form the originating network.
- Step 2: CAB XDMS (AB Application Usage) of CAB Client B on receiving the remote forward request checks whether the recipients listed in the request are in its domain and then creates the 200 OK response and sends to the CAB XDMS (AB Application Usage) of CAB Client A.
- Step 3: CAB XDMS (AB Application Usage) of CAB Client B checks the preferences of the CAB Client B for handling the forward request from the AB forward preferences stored in the AB Application Usage. Here the case shown is that CAB Client B wants to accept the XDM Resource received from CAB Client A
- Step 4: CAB XDMS (AB Application Usage) fetches the XDM Resource using the URI received in the Forward Remote request and stores the Contact Share data in CAB Client’ AB Application Usage and sets the contact status as “updated”.

- Step 5: CAB XDMS (AB Application Usage) of CAB Client B updates the Forwarding Notification List of CAB Client B for request notification with the details of the received remote forward request
- Step 6: The List XDMS of CAB Client B updates the Forwarding Notification List of CAB Client B with the above request notification entry and sends 200 OK response.
- Step 7: Since CAB Client A has requested for the Contact Share Delivery Report, CAB XDMS (AB Application Usage) of CAB Client B generates the Forward Delivery Report request.
- Step 8: CAB XDMS (AB Application Usage) of CAB Client B receives 200 OK response with the XDCCP Response containing <done> element from the originating network.
- Step 9: CAB XDMS (AB Application Usage) of CAB Client B notifies the CAB Server about document changes using SIC-2 interface.
- Step 10: The CAB Server sends the response (200 OK) to CAB XDMS (AB Application Usage) using SIC-2 interface.
- Step 11: The CAB Server sends an alert to CAB Client B to indicate changes occurred in the CAB User's AB as described in the step 5 of [CAB TS] Appendix C.6.2
- Following this, the CAB Client B may initiate synchronization as described in [CAB TS] Appendix C.6.1.

C.4.1.2.2 AB forwarding (Confirm)

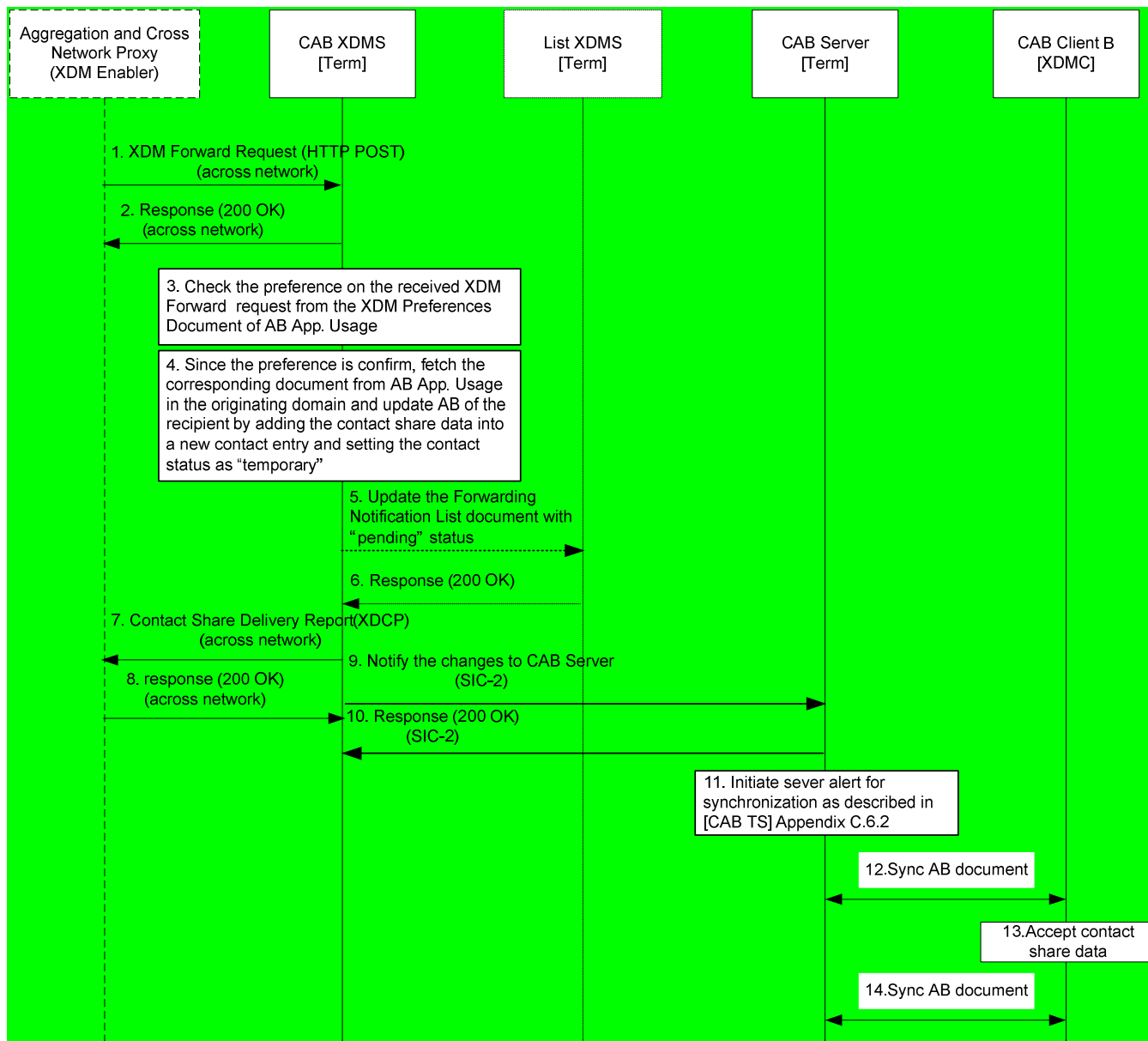


Figure 6: Flows of AB forwarding in the terminating side towards a CAB user

- Step 1: CAB XDMS (AB Application Usage) of CAB Client B receives the XDCP Forward Remote request form the originating network.
- Step 2: CAB XDMS (AB Application Usage) of CAB Client B on receiving the remote forward request checks whether the recipients listed in the request are in its domain and then creates the 200 OK response and sends to the CAB XDMS (AB Application Usage) of CAB Client A.
- Step 3: CAB XDMS (AB Application Usage) of CAB Client B checks the preferences of the CAB Client B for handling the forward request from the AB forward preferences stored in the AB Application Usage. Here the case shown is that CAB Client B wants to confirm the XDM Resource received from CAB Client A

- Step 4: CAB XDMS (AB Application Usage) fetches the XDM Resource using the URI received in the Forward Remote request and stores the Contact Share data in CAB Client B's AB Document and sets the contact status as "temporary".
- Step 5: [optional] CAB XDMS (AB Application Usage) of CAB Client B updates the Forwarding Notification List of CAB Client B for request notification with the details of the received remote forward request
- Step 6: [optional] The List XDMS of CAB Client B updates the Forwarding Notification List of CAB Client B with the above request notification entry and sends 200 OK response.
- Step 7: Since CAB Client A has requested for the Contact Share Delivery Report CAB XDMS (AB Application Usage) of CAB Client B generates the Forward Delivery Report request.
- Step 8: CAB XDMS (AB Application Usage) of CAB Client B receives 200 OK response with the XDMP Response containing <done> element from the originating network.
- Step 9: CAB XDMS (AB Application Usage) of CAB Client B notifies the CAB Server about document changes using SIC-2 interface.
- Step 10: The CAB Server sends the response (200 OK) to CAB XDMS (AB Application Usage) using SIC-2 interface.
- Step 11: The CAB Server sends an alert to CAB Client to indicate changes occurred in the CAB User's AB as described in the step 5 of [CAB TS] Appendix C.6.2
- Step 12: Following this, the CAB Client B may initiate synchronization as described in [CAB TS] Appendix C.6.1
- Step 13: CAB Client B accepts the contact share data stored as "temporary" contact in the AB Document and removes <temporary> element in the address book.
- Step 14: Following this, the CAB Client B may initiate synchronization as described in [CAB TS] Appendix C.6.1

C.4.1.2.3 PCC forwarding (Accept)

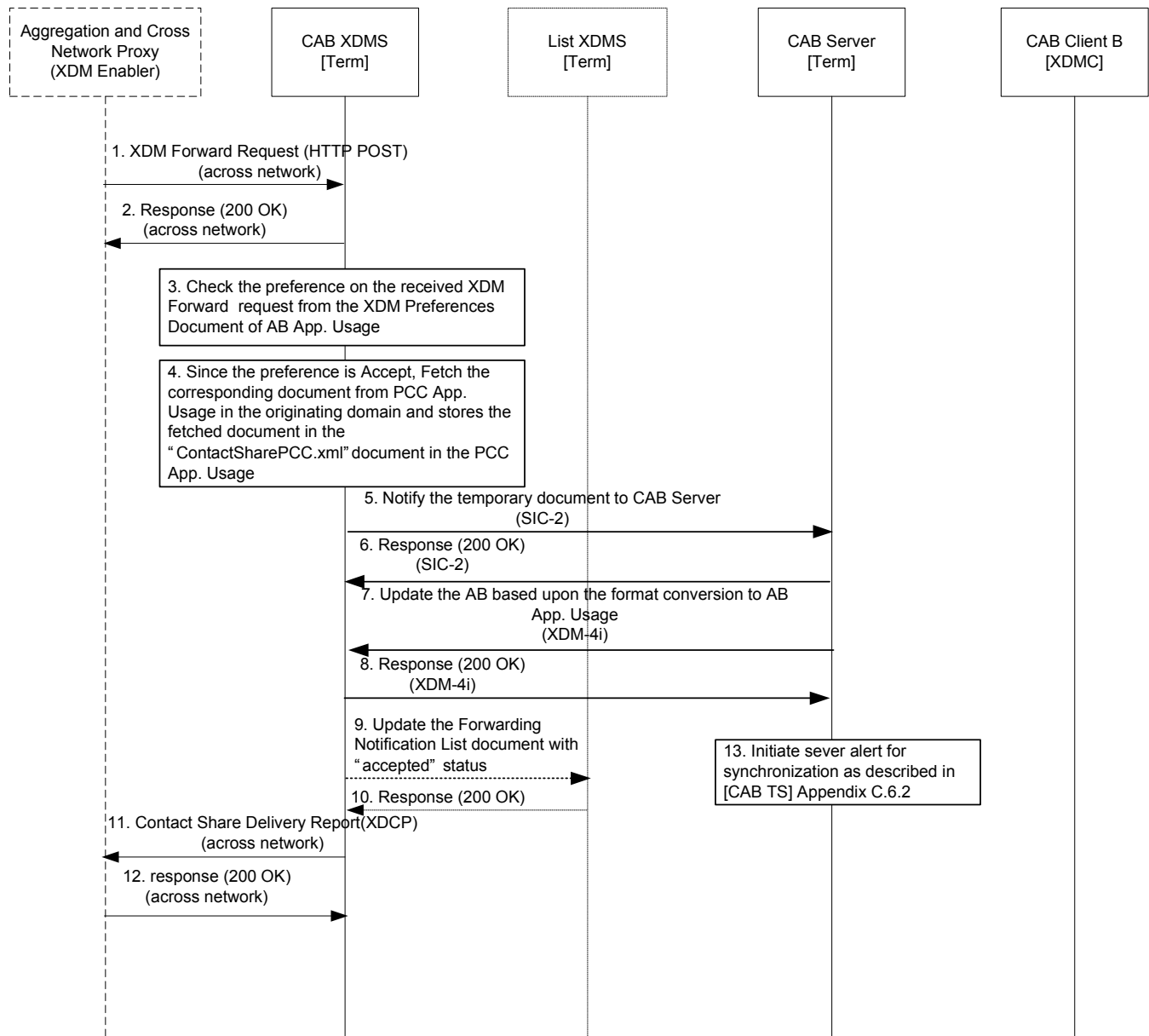


Figure 7: Flows of PCC forwarding in the terminating side towards a CAB user

- Step 1: CAB XDMs (PCC Application Usage) of CAB Client B receives the XDCP Forward Remote request form the originating network.
- Step 2: CAB XDMs (PCC Application Usage) of CAB Client B on receiving the remote forward request checks whether the recipients listed in the request are in its domain and then creates the 200 OK response and sends to the CAB XDMs (PCC Application Usage) of CAB Client A.
- Step 3: CAB XDMs (PCC Application Usage) of CAB Client B checks the preferences of the CAB Client B for handling the forward request from the AB forward preferences stored in the AB Application Usage. Here the case shown is that CAB Client B wants to accept the XDM Resource received from CAB Client A
- Step 4: CAB XDMs (PCC Application Usage) fetches the XDM Resource using the URI received in the Forward Remote request and stores the fetched content in the ContactSharePCC.xml document in the PCC Application Usage.

- Step 5: The CAB Server gets notified of the changes to the ContactSharePCC.xml document from CAB XDMS (PCC Application Usage)
- Step 6: The CAB Server sends the response (200 OK) to CAB XDMS (PCC Application Usage) using SIC-2 interface.
- Step 7: The CAB Server updates the AB document with the content of received contact share data residing in the ContactSharePCC.xml document after performing the needed format conversion and sets the contact status as “updated”.
- Step 8: The CAB XDMS (AB Application Usage) sends the response (200 OK) to CAB Server.
- Step 9: [Optional] CAB XDMS (PCC Application Usage) of CAB Client B updates the Forwarding Notification List of CAB Client B for request notification with the details of the received remote forward request
- Step 10: [Optional] The List XDMS of CAB Client B updates the Forwarding Notification List of CAB Client B with the above request notification entry and sends 200 OK response.
- Step 11: Since CAB Client A has requested for the Contact Share Delivery Report CAB XDMS (PCC Application Usage) of CAB Client B generates the Forward Delivery Report request.
- Step 12: CAB XDMS (PCC Application Usage) of CAB client B receives 200 OK response with the XDMP Response containing <done> element from the originating network.
- Step 13: The CAB Server sends an alert to CAB Client B to indicate changes occurred in the CAB User’s AB as described in the step 5 of [CAB TS] Appendix C.6.2
- Following this, the CAB Client B may initiate synchronization as described in [CAB TS] Appendix C.6.1

C.4.1.2.4 PCC forwarding (Confirm)

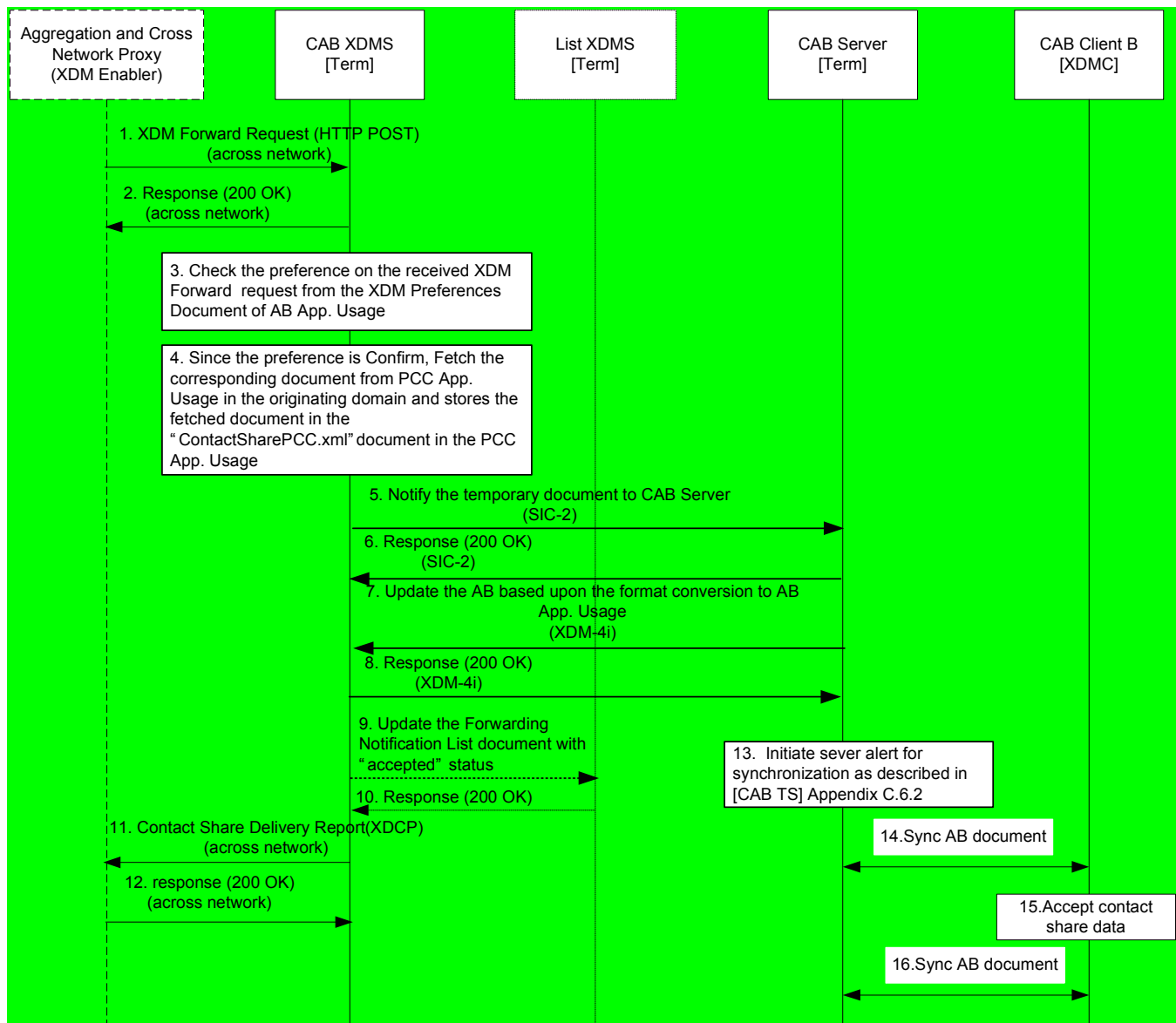


Figure 8: Flows of PCC forwarding in the terminating side towards a CAB user

- Step 1: CAB XDMS (PCC Application Usage) of CAB Client B receives the XDCP Forward Remote request from the originating network.
- Step 2: CAB XDMS (PCC Application Usage) of CAB Client B on receiving the remote forward request checks whether the recipients listed in the request are in its domain and then creates the 200 OK response and sends to the CAB XDMS (PCC Application Usage) of CAB Client A.
- Step 3: CAB XDMS (PCC Application Usage) of CAB Client B checks the preferences of the CAB Client B for handling the forward request from the AB forward preferences stored in the AB Application Usage. Here the case shown is that CAB Client B wants to accept the XDM Resource received from CAB Client A
- Step 4: CAB XDMS (PCC Application Usage) fetches the XDM Resource using the URI received in the Forward Remote request and creates the fetched content in the ContactSharePCC.xml document as the temporary document in the PCC Application Usage.

- Step 5: The CAB Server gets notified of the temporary document from CAB XDMS (PCC Application Usage)
- Step 6: The CAB Server sends the response (200 OK) to CAB XDMS (PCC Application Usage) using SIC-2 interface.
- Step 7: The CAB Server updates the AB document with the content of received contact share data residing in the ContactSharePCC.xml document after performing the needed format conversion and sets the contact status as “temporary”.
- Step 8: The CAB XDMS (AB Application Usage) sends the response (200 OK) to CAB Server.
- Step 9: [Optional] CAB XDMS (PCC Application Usage) of CAB Client B updates the Forwarding Notification List of CAB Client B for request notification with the details of the received remote forward request
- Step 10: [Optional] The List XDMS of CAB Client B updates the Forwarding Notification List of CAB Client B with the above request notification entry and sends 200 OK response.
- Step 11: Since CAB Client A has requested for the Contact Share Delivery Report CAB XDMS (PCC Application Usage) of CAB Client B generates the Forward Delivery Report request.
- Step 12: CAB XDMS (PCC Application Usage) of CAB client B receives 200 OK response with the XDCCP Response containing <done> element from the originating network.
- Step 13: The CAB Server sends an alert to CAB Client to indicate changes occurred in the CAB User’s AB as described in the step 5 of [CAB TS] Appendix C.6.2
- Step 14: Following this, the CAB Client B may initiate synchronization as described in [CAB TS] Appendix C.6.1
- Step 15: CAB Client B accepts the contact share data stored as “temporary” contact in the address book and removes <temporary>element for the confirmed contact..
- Step 16: Following this, the CAB Client B may initiate synchronization as described in [CAB TS] Appendix C.6.1.

C.4.2 Contact Share towards a Non CAB User

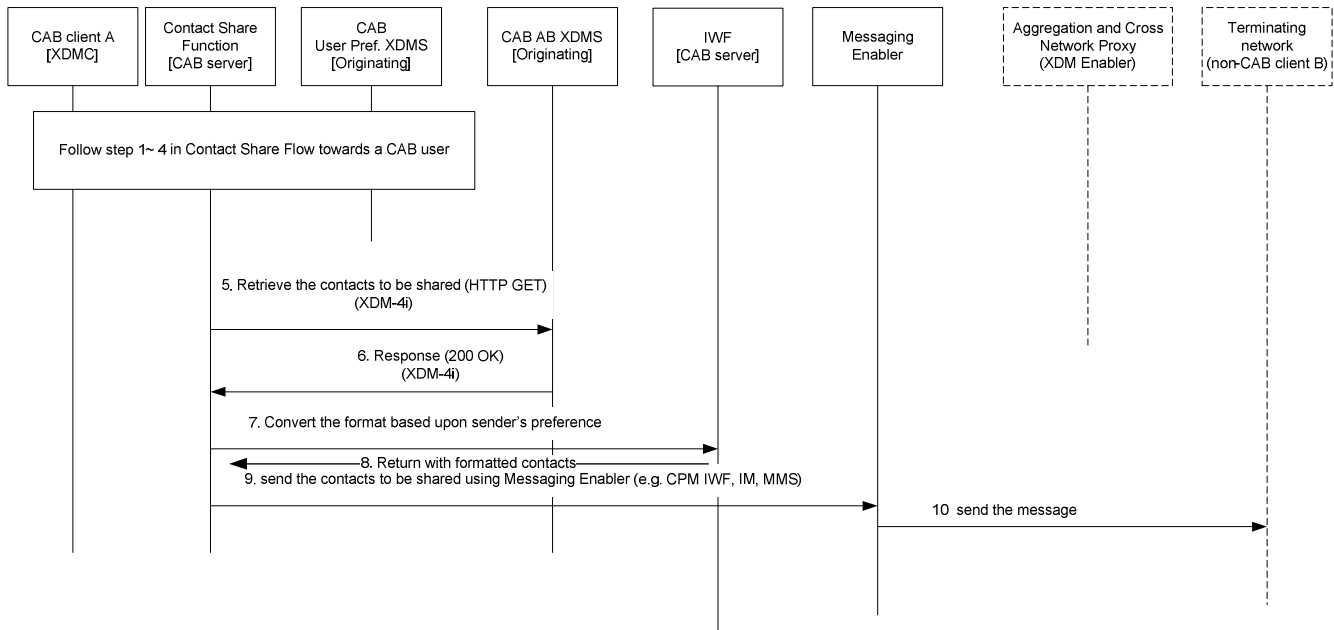


Figure 9 : Flows of Contact Share towards a non CAB user

Steps 1 through 4 are same as the above flow “Contact Share towards a CAB User”

Step 5: Contact Share Function uses the CAB Server’s XDM Agent to perform a HTTP GET using XDM-4i to retrieve the contact(s) to be shared from CAB XDMS

Step 6: Contact Share Function receives the success response (200 OK) from CAB XDMS.

Step 7: Subject to CAB Client A Preferences, Contact Share Function may request internally to CAB IWF to convert the format of the contact(s) to be shared, based on CAB Client A Preferences.

Step 8: The CAB IWF returns with the formatted contact(s)

Step 9: Upon successfully obtaining the data format to be sent to the recipient(s), Contact Share Function sends the contacts to be shared using a Messaging Enabler, e.g. CPM ISF through its exposed interface.

Step 10: Messaging Enabler sends the message that contains the shared contacts to the recipient(s). The delivery and responses depend on each of the messaging means employed and it is out of scope of this specification.

C.5 Contact Subscription flows

C.5.1 CAB Server Subscribes to contacts PCCs using Subscription Proxy

This flow is triggered by the updates in the CAB User’s subscription list in the CAB User Preference Application Usage, such as deleting, adding new contacts or modifying existing contacts uris that are used in the subscription process.

This example flow uses the case of addition of multiple users into the CAB User A’s subscription list: CAB User B and CAB user C to be subscribed to. The Subscription List in the CAB User Preference Application Usage follows the same flow as described in the Appendix C.3 “Sample XCAP flows for management of CAB XML documents”.

In this example, the CAB User A is identified through the following URI: “sip:joe.bloggs@example.com”.

Note: the 200 OK responses and ACK are not shown in the figure and steps for simplification.

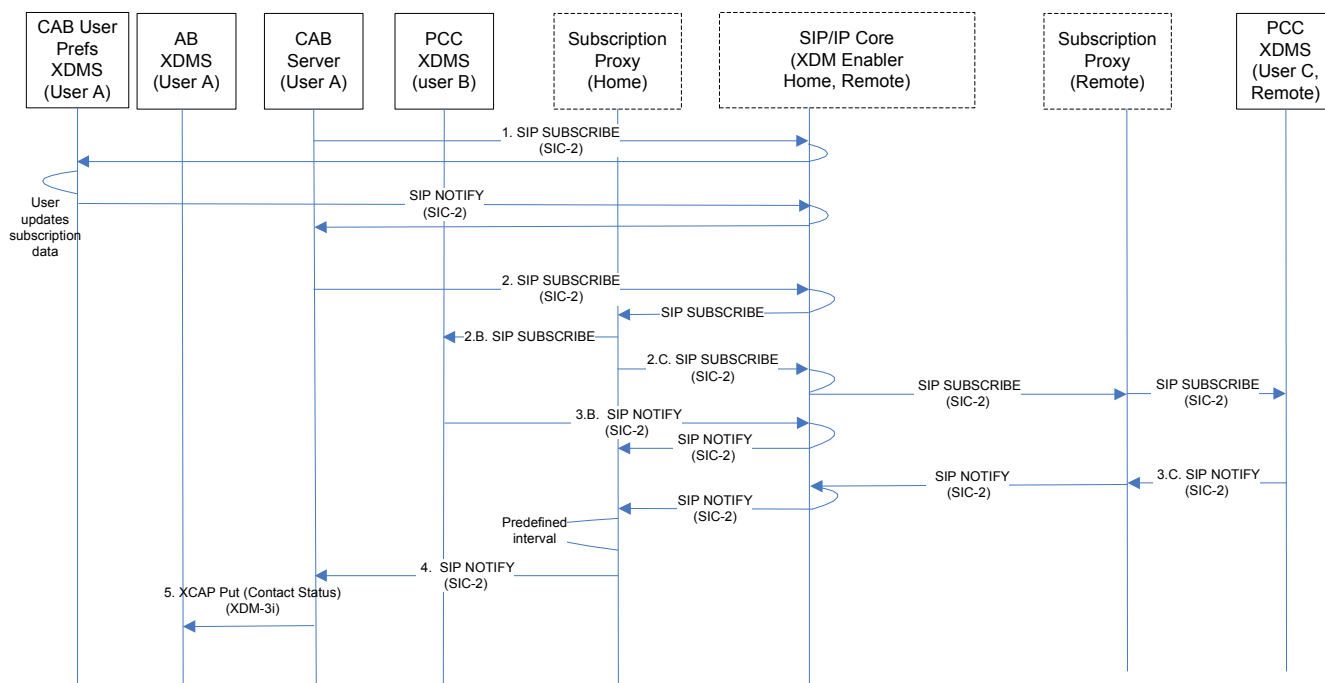


Figure 10 : Contact Subscription flow using the Subscription Proxy

Step 1: Following the subscription of the CAB Server of CAB User A through interface SIC-2 to the updates of the Subscription List in the CAB User Preference Application Usage, the CAB Server of CAB User A gets notified of changes. Note: This trigger can also be achieved through regular polling of CAB User Preferences document by the CAB Server. The Contact Subscription status is set to “pending” for both contacts B and C in the Contact Status.

Step 2: CAB Server of CAB user A sends a SIP SUBSCRIBE request to the Subscription Proxy via the SIP/IP Core using the xcap-diff event package as defined in [OMA XDM Core]. The Request URI of the SIP SUBSCRIPTION request is set to the SIP address of the Subscription Proxy as obtained during provisioning. The body of the SIP SUBSCRIBE request contains the resource list with three entries as specified in [OMA XDM Core]:

- AUID “org.openmobilealliance.cab-pcc”
- URI pointing to PCC of user B: “userB@example.com”

- URI pointing to PCC of user C: “userC@other_domain.com”

Upon receiving a SIP SUBSCRIBE request for the “xcap-diff” event package, the Subscription Proxy creates a subscription dialog to “xcap-diff” event package to provide the changes of the data identified by the body of SUBSCRIBE request, and return 200 OK to the CAB server through the SIP/IP Core

Step 2.B: Based on the received initial subscription, the Subscription Proxy generates SIP SUBSCRIBE requests for back-end subscriptions, through the SIP/IP core for each of the users listed in the body. First, the Subscription Proxy sets the Request URI to the value “sip:userB@example.com;auid=org.openmobilealliance.cab-pc” and sends a SIP SUBSCRIBE request to the PCC Application Usage of CAB User B.

The PCC XDMS of CAB User B receives the SIP SUBSCRIBE with the Request URI “sip:userB@example.com;auid=org.openmobilealliance.cab-pc” and verifies the CAB User B Access permissions regarding CAB user A. Assuming the CAB User A passed authorization checks, the PCC Application Usage of User B replies with 200 OK, through SIP/IP Core to the CAB Server.

Step 2.C: The next back-end subscription is targeted to PCC Application Usage in a remote network. Such SIP SUBSCRIBE request is sent to PCC Application Usage of User C in the remote network via the SIP/IP Core, and possibly via Subscription Proxy in the remote network.

The PCC Application Usage in the remote network receives the SIP SUBSCRIBE with the Request URI “sip:userC@other_domain.com;auid=org.openmobilealliance.cab-pc” and verifies the CAB User C Access permissions regarding CAB user A. Assuming the CAB User A passed authorization checks, the PCC Application Usage of User C replies with 200 OK following the same path back to the CAB Server.

Step 3.B: The PCC Application Usage of CAB User B generates and sends an initial SIP NOTIFY via same path, containing initial reference to XDM document listed in the body of SIP SUBSCRIBE request, i.e. the PCC XML document of the CAB User B. The SIP NOTIFY is received by the Subscription Proxy in Home domain of CAB User A, which will reply with 200 Ok all the way to the PCC Application Usage.

Step 3.C: The PCC Application Usage of CAB User C generates and sends an initial SIP NOTIFY via same path, containing initial reference to XDM document listed in the body of SIP SUBSCRIBE request, i.e. the PCC XML document of the CAB User C. The SIP NOTIFY is received by the Subscription Proxy in Home domain of CAB User A, which will reply with 200 Ok all the way to the PCC Application Usage.

Step 4: In this example, it is assumed that initial notifications from PCC Application Usages in the same domain are received without any significant delay. The Subscription Proxy, after the predefined interval, generates initial notification to the subscribed CAB server of CAB User A.

Until this state, the initial notifications from PCC Application Usage (s) in the same network and remote network were received. The initial notification contains the body indicating these states. The SIP NOTIFY request is sent by the Subscription Proxy via the SIP/IP Core, to the CAB Server of CAB User A. The CAB server A will reply all the way back with 200 OK.

Step 5: CAB Server A receives the SIP Notify with the states of the subscriptions for the CAB user B and CAB user C and updates the Contact Status of the Contact Entries corresponding to CAB User B and CAB User C in the AB Application Usage of the CAB User A based upon the user preference of CAB User A. The updates in Contact Status of contacts B and C consists in the subscription status data (in this case “active”).

The update of Contact Status in AB Application Usage gets notified to the CAB Server A and CAB Server A sends an alert to CAB User A to indicate changes occurred in the CAB User A’s AB as described in the step 5 of [CAB TS] Appendix C.6.2. Following this, the CAB Client B may initiate synchronization as described in [CAB TS] Appendix C.6.1.

C.6 Managing Address Book Flow

C.6.1 CAB Client Address Book Modifications and Synchronization

Figure 11 depicts a CAB Client originated address book modifications and synchronization flow. The flow assumes that the CAB Server and CAB Client had up-to-date copies of the network-based address book, prior to the CAB Client making modifications to the address book. For illustrative purposes, a two-way DS synchronization with initialization separate from data sync is shown.

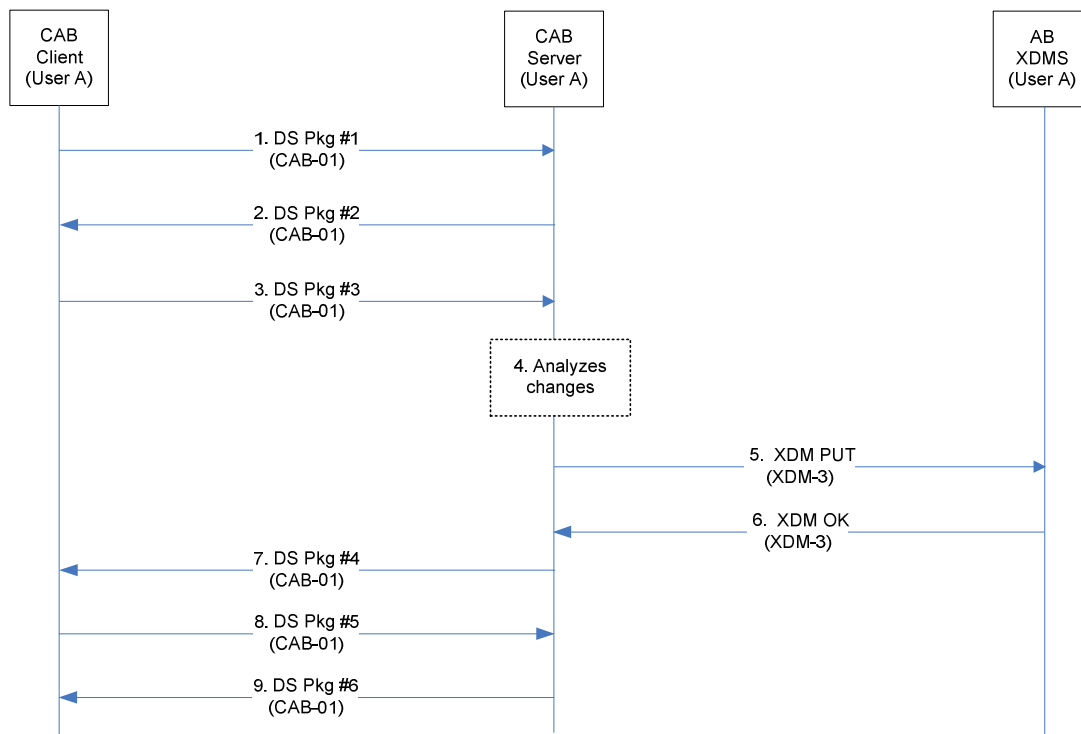


Figure 11 : CAB Client Address Book Modifications and Synchronization Flow

1. Based on modifications on the CAB User's client address book, the CAB Client sends an OMA DS initialization message (OMA DS Pkg #1) to the CAB Server. The message includes server challenge (if needed), device capability information, DS sync type, and data type (i.e., the user's Address Book) to be synchronized with associated anchors.
2. The CAB Server computes the authentication challenge response, if a challenge was included, and then sends a DS message (OMA DS Pkg #2) to the CAB client that includes the challenge response to the CAB Client, if the CAB Client had challenged the CAB Server, a client challenge, server capability information, sync type, and synchronization information including associated anchors.
3. The CAB Client computes the challenge response for the CAB Server, and sends a DS message (OMA DS Pkg #3) with proposed address book modifications to the CAB Server.
4. The CAB Server processes address book modifications from the CAB Client.
5. The CAB Server updates the AB XDMS. This may involve multiple XDM requests, but only one request is depicted in the figure.

6. The AB XDMS acknowledges the write operation. There is one XDM response for each XDM request in the previous step.
7. The CAB Server sends a DS message (OMA DS Pkg #4) with a status for the client modifications and server modifications to the client.
8. The CAB Client updates its local address book cache, and sends a DS status message to the server (OMA DS Pkg #5).
9. The CAB Server sends an DS message (OMA DS Pkg #6) with the map acknowledgement to the CAB Client.

Note: Depending on the DS packages implementation, Steps 5 and 6 may occur at different times during OMA DS synchronization procedure. If there is data resulting from Step 4 that needs to be written into AB XDMS, this has to be done prior to closing the procedure of OMA DS synchronization (i.e. Step 9 or Step 7) to avoid data inconsistency in case of further reads of AB XDMS data.

C.6.2 Address Book Modifications from Network

Figure 12 depicts a network originated address book modifications flow e.g. when there is a change to the user address book that occurred in the network. The flow assumes that the local and network address books were up-to-date prior to the network modifications.

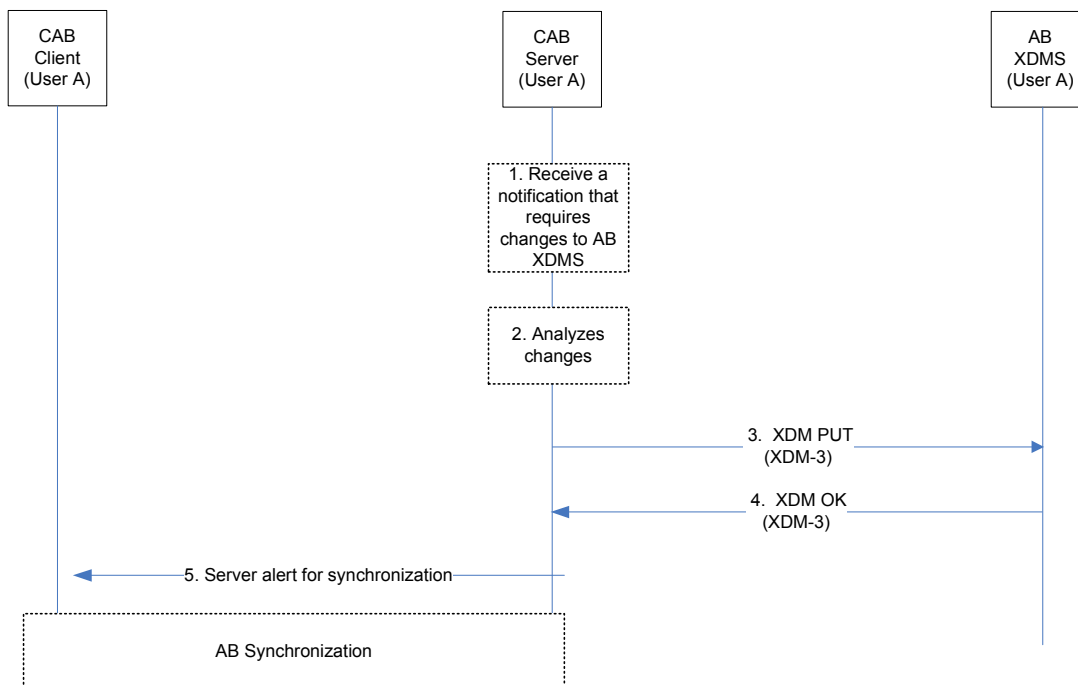


Figure 12 : Address Book Modifications from Network

1. The CAB Server receives a notification that may require a change to the user's AB in the network (e.g., receives a SIP NOTIFY request from an XDM Subscription Proxy or PCC XDMS of the Published Contact Card, based on an active subscription, change from multiple devices).
2. The CAB Server analyzes the changes found in the notifications.
3. The CAB Server updates the AB XDMS.
4. The AB XDMS applies access permission rules, updates AB document, and acknowledges the update.
5. The CAB Server sends an alert to CAB Client to indicate changes occurred in the CAB User's network address book.

Following this, the CAB Client may initiate synchronization as described in C6.1.

Appendix D. X-CAB parameters and fields definitions (Informative)

This annex defines some informative vCard extensions (i.e. X-CAB based) that can be helpful to be used when format adaptation between CAB Format with Legacy Format is performed.

Note: The following table includes extensions for some of the CAB format elements, but it is not exhaustive.

CAB attributes	Initial vCard parameter	vCard extensions (parameters)
"pcc-type"		Used as a property not as a parameter (see 2 nd table below)
"contactIdRef"		X-CAB-CONTACT-ID-REF (used in X-CAB-CONTACT-STATUS to indicate, when temporary element is used, a reference to the Contact Entry to which the contact activity-status is associated with)
"index"		X-CAB-INDEX (possible values : token)
"language-proficiency-type"		X-CAB-LANGUAGE-PROFICIENCY-TYPE (possible values : "read only", "speak", "read/write")
"language-fluency-type"		X-CAB-LANGUAGE-FLUENCY-TYPE (possible values : "beginner", "average", "fluent")
"q-level"		"X-CAB-LEVEL" (possible values : "beginner", "average", "expert")
"h-level"		"X-CAB-LEVEL" (possible values : "high", "medium", "low")
"i-level"		"X-CAB-LEVEL" (possible values : "high", "medium", "low")
"org-name-type"	"TYPE"	as in last vCard specification with specific values (possible values : "LegalName", "FormerName", "OfficialName")

PCC document attribute	Initial Vcard property	vCard extensions (fields)
pcc-type		<p>X-CAB-PCC-TYPE</p> <p>Purpose: To specify the kind of object the vCard represents.</p> <p>Value type: A single text value.</p> <p>Cardinality: (0,1)</p> <p>Special note: The value may be one of: "individual", "group" and "organization" .</p> <p>If this property is absent, "individual" MUST be assumed as default.</p>

<p>Contact-type</p> <p>&entry-status</p> <p>& Contact-subscription-status</p> <p>& Contact-source</p>		<p>X-CAB-CONTACT-STATUS</p> <p>Purpose: To specify the CAB status of the object the vCard represents.</p> <p>Value type: A single structured value consisting of 4 values separated by the SEMI-COLON character (ASCII decimal 59) :</p> <p>1 contact-type (possible value : "CAB" if the contact is a CAB user)</p> <p>2 entry-status. This field is composed of one or both of the following sub-fields :</p> <ul style="list-style-type: none"> - updated : indicating that the contact has been updated by the CAB server, as a result of automatic updates from incoming subscription request(s) (possible values : "incoming subscription request") and contact share (possible values: "contact share") (values are separated by commas). - temporary (possible values : "contact subscription", "contact imported", "contact added", "incoming subscription")
---	--	---

		<p>request" and contact share (possible values: "contact share"). This field may include a X-CAB-CONTACT-ID-REF parameter (field value and parameter are separated by a comma).</p> <p>3 contact-subscription-status (possible values : "active", "pending", "denied", "invalid filter", "not found", "other_error")</p> <p>4 contact-source indicating the latest source from which the contact data was obtained or updated (default value "CAB")</p> <p>Cardinality: (1,1)</p>
--	--	---

Person-details elements

Gender		<p>X-CAB-SEX</p> <p>Purpose: To specify the sex of the object the vCard represents, as defined in [ISO.5218.2004].</p> <p>Value type: A single integer value.</p> <p>Cardinality: (0,1)</p> <p>Special note: The value 0 stands for "not known", 1 stands for "male", 2 stands for "female", 3 stands for "other" and 9 stands for "not applicable".</p>
Language-entry		<p>X-CAB-LANG</p> <p>Purpose: To specify the language(s) that may be used for contacting the individual associated with the vCard.</p> <p>Value type: A single language-tag value.</p> <p>Cardinality: (0,n)</p> <p>Special note: This property can include the "PREF" parameter to indicate a preferred-language (possible values : from 1 to 100). This property can include "X-CAB-LANGUAGE-PROFICIENCY-TYPE", "X-CAB-LANGUAGE-FLUENCY-TYPE" parameters. This property can include an "X-CAB-INDEX" parameter.</p>
Service-entry		<p>X-CAB-SERVICE</p> <p>Purpose: To specify the aliases used on different sites by the object that the vCard refers to.</p> <p>Value type: A single structured value consisting of 3 values separated by the SEMI-COLON character (ASCII decimal 59) :</p> <p>1 label</p> <p>2 alias</p> <p>3 url</p> <p>Cardinality: (0,n)</p> <p>Special note: This property can include the "X-CAB-INDEX" parameter</p>
Expertise-entry		<p>X-CAB-EXPERTISE</p> <p>Purpose: To specify the expertise(s) of the object that the vCard refers to.</p> <p>Value type: A single string value.</p> <p>Special note: This property can include the X_CAB-LEVEL parameter (possible values : "beginner", "average", "expert"). This property can include the "X-CAB-INDEX" parameter.</p> <p>Cardinality: (0,n)</p>
Hobby-entry		<p>X-CAB-HOBBY</p> <p>Purpose: To specify the hobbies of the object that the vCard refers to.</p> <p>Value type: A single string value.</p> <p>Special note: This property can include the X_CAB-LEVEL parameter (possible values : "high", "medium", "low"). This property can include the "X-CAB-INDEX"</p>

		parameter. Cardinality: (0,n)
Interest-entry		X-CAB-INTEREST Purpose: To specify the interest(s) of the object that the vCard refers to. Value type: A single string value Special note: This property can include the X-CAB-LEVEL parameter (possible values : "high", "medium", "low"). This property can include the "X-CAB-INDEX" parameter. Cardinality: (0,n)
Public-note		X-CAB-PUBLICNOTE Purpose: To specify additional information associated with the object the vCard refers to. Value type: A single string value Cardinality: (0,n)

Organization-directory

Org-directory	<ORG>	X-CAB-ORG-DIRECTORY Purpose: To specify the organization-directory of the object the vCard represents. Value type: A single structured value consisting of : - directory (a URI) Cardinality: (0,n) Special note: This property can include the PREF and X-CAB-INDEX parameters.
---------------	-------	---