

# Converged Address Book (CAB) Specification

Approved Version 1.1 – 21 Jun 2018

---

**Open Mobile Alliance**  
OMA-TS-CAB-V1\_1-20180621-A

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.**

© 2018 Open Mobile Alliance All Rights Reserved.

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

# Contents

<b>1. SCOPE</b> .....	<b>7</b>
<b>2. REFERENCES</b> .....	<b>8</b>
<b>2.1 NORMATIVE REFERENCES</b> .....	<b>8</b>
<b>2.2 INFORMATIVE REFERENCES</b> .....	<b>9</b>
<b>3. TERMINOLOGY AND CONVENTIONS</b> .....	<b>10</b>
<b>3.1 CONVENTIONS</b> .....	<b>10</b>
<b>3.2 DEFINITIONS</b> .....	<b>10</b>
<b>3.3 ABBREVIATIONS</b> .....	<b>12</b>
<b>4. INTRODUCTION (INFORMATIVE)</b> .....	<b>13</b>
<b>4.1 VERSION 1.0</b> .....	<b>13</b>
<b>4.2 VERSION 1.1</b> .....	<b>13</b>
<b>5. PROCEDURES AT CAB SERVER</b> .....	<b>14</b>
<b>5.1 AB SYNCHRONIZATION FUNCTION</b> .....	<b>14</b>
5.1.1 Retrieval and Restore of previously deleted Contact Entries .....	14
<b>5.2 CONTACT SUBSCRIPTION FUNCTION</b> .....	<b>15</b>
<b>5.3 CONTACT SHARE FUNCTION</b> .....	<b>16</b>
5.3.1 Procedures at the Originating Side.....	16
5.3.2 Procedures at the Receiving Side .....	17
<b>5.4 INTERWORKING FUNCTION</b> .....	<b>17</b>
5.4.1 Import from Non-CAB AB Systems.....	18
5.4.2 Import profile information from 3rd Party Systems.....	19
5.4.3 Export PCC information to 3rd Party Systems .....	20
5.4.4 Subscription to profile information in 3rd Party Systems.....	20
5.4.5 Contact Search with External Directories .....	22
5.4.6 Format Adaptation .....	23
<b>5.5 XDM AGENT</b> .....	<b>23</b>
<b>5.6 PRESENCE SOURCE</b> .....	<b>23</b>
<b>5.7 PRESENCE WATCHER</b> .....	<b>23</b>
<b>5.8 CONTACT STATUS FUNCTION</b> .....	<b>24</b>
5.8.1 Management of Contact Status in AB.....	24
5.8.2 Population of CAB Capability based on the Presence Enabler.....	24
5.8.3 Incoming Contact Subscription Requests .....	25
5.8.4 Contact Added .....	26
5.8.5 Common Connections.....	28
5.8.6 Suggestion of contact information .....	29
5.8.7 Service suggestions.....	30
5.8.8 Public content .....	31
<b>5.9 APPLYING CAB USER PREFERENCES</b> .....	<b>31</b>
<b>5.10 SUBSCRIPTION INVITATION FUNCTION</b> .....	<b>31</b>
5.10.1 Procedures at the Originating Side.....	31
5.10.2 Procedures at the Receiving Side.....	32
5.10.3 MIME Type for Cross-domain Subscription Invitations .....	33
<b>6. PROCEDURES AT CAB CLIENT</b> .....	<b>35</b>
<b>6.1 ADDRESS BOOK MANAGEMENT AND SYNCHRONIZATION</b> .....	<b>35</b>
6.1.1 Address Book Management and Synchronization using OMA DS .....	35
6.1.2 Address Book Management and Synchronization using OMA XDM .....	35
<b>6.2 PERSONAL CONTACT CARD (PCC) MANAGEMENT</b> .....	<b>36</b>
<b>6.3 CAB USER PREFERENCES MANAGEMENT</b> .....	<b>36</b>
<b>6.4 IMPORT NON-CAB ADDRESS BOOK</b> .....	<b>36</b>
<b>6.5 IMPORT PROFILE INFORMATION FROM 3<sup>RD</sup> PARTY SYSTEMS</b> .....	<b>37</b>
<b>6.6 EXPORT PCC INFORMATION TO 3<sup>RD</sup> PARTY SYSTEMS</b> .....	<b>38</b>
<b>6.7 SUBSCRIPTION TO PROFILE INFORMATION IN 3<sup>RD</sup> PARTY SYSTEMS</b> .....	<b>38</b>

- 6.8 CONTACT SHARE.....39**
- 6.9 CONTACT SUBSCRIPTION.....39**
- 6.10 ACCESS PERMISSIONS MANAGEMENT .....40**
  - 6.10.1 Access Permissions Document for Address Book .....40
  - 6.10.2 Access Permissions Document for PCC .....40
- 6.11 UPP DIRECTORY DOCUMENT MANAGEMENT .....40**
- 6.12 XDM PREFERENCES MANAGEMENT.....40**
  - 6.12.1 XDM Preferences Document for Address Book.....40
  - 6.12.2 XDM Preferences Document for PCC.....40
- 6.13 SUBSCRIPTION TO CAB XML DOCUMENTS CHANGES .....40**
  - 6.13.1 Subscriptions to CAB User’s own PCC Document changes .....40
  - 6.13.2 Subscriptions to CAB User’s own CAB User Preferences Document changes.....41
  - 6.13.3 Subscriptions to CAB User’s own CAB Feature Handler Document changes .....41
  - 6.13.4 Subscription to CAB User’s own CAB XML documents using Subscription Proxy.....41
- 6.14 CONTACT SEARCH .....42**
- 6.15 CONTACT STATUS MANAGEMENT.....42**
- 6.16 AUTHENTICATION.....42**
  - 6.16.1 Authentication for AB Synchronization.....42
  - 6.16.2 Authentication for XML Document Management.....42
- 6.17 SUBSCRIPTION INVITATION .....42**
- 6.18 RETRIEVAL OF HISTORY INFORMATION.....43**
  - 6.18.1 History Information of Address Book.....43
  - 6.18.2 History Information of PCC.....43
- 6.19 CONTACT SUGGESTION.....43**
- 7. CAB XDMS .....45**
- 8. CAB MANAGEMENT OBJECT .....46**
- APPENDIX A. CHANGE HISTORY (INFORMATIVE).....47**
  - A.1 APPROVED VERSION HISTORY .....47**
- APPENDIX B. STATIC CONFORMANCE REQUIREMENTS (NORMATIVE) .....48**
  - B.1 SCR FOR CAB CLIENT .....48**
  - B.2 SCR FOR CAB SERVER.....51**
- APPENDIX C. FLOWS (INFORMATIVE) .....58**
  - C.1 CONTACT SEARCH .....58**
    - C.1.1 Contact Search - PCC .....58
    - C.1.2 Contact Search - AB .....59
    - C.1.3 Contact Search – External Directories .....60
  - C.2 IMPORT FROM NON-CAB ADDRESS BOOK SYSTEMS.....61**
  - C.3 SAMPLE XCAP FLOWS FOR MANAGEMENT OF CAB XML DOCUMENTS .....62**
    - C.3.1 XCAP operations on CAB XML documents .....62
  - C.4 SAMPLE CONTACT SHARE FLOWS.....65**
    - C.4.1 Contact Share towards a CAB User .....66
    - C.4.2 Contact Share towards a Non CAB User .....76
  - C.5 CONTACT SUBSCRIPTION FLOWS .....78**
    - C.5.1 CAB Server Subscribes to contacts PCCs using Subscription Proxy .....78
    - C.5.2 CAB Client Subscribes to contact’s PCC through the reactive authorization .....80
  - C.6 MANAGING ADDRESS BOOK FLOW .....82**
    - C.6.1 CAB Client Address Book Modifications and Synchronization .....82
    - C.6.2 Address Book Modifications from Network .....83
  - C.7 CAB CAPABILITY FLOWS.....84**
    - C.7.1 CAB Server publishes CAB capability as Permanent Presence State .....84
    - C.7.2 CAB Server publishes CAB capability as a regular SIP Publish .....84
    - C.7.3 CAB Server subscribes to CAB Capability of its user’s contacts .....85
  - C.8 EXTERNAL PROFILE INFORMATION IMPORT FLOWS.....85**
    - C.8.1 Automatic approval of External Profile Information Import .....85
    - C.8.2 Manual approval of External Profile Information Import .....87

- C.9 PCC INFORMATION EXPORT FLOWS .....89**
- C.10 SUBSCRIPTION TO EXTERNAL PROFILE INFORMATION FLOWS.....91**
  - C.10.1 Automatic approval of Subscription to External Profile Information .....91
  - C.10.2 Manual approval of Subscription to External Profile Information.....92
- C.11 SUGGESTION OF CONTACT INFORMATION.....94**
  - C.11.1 Suggestion of contact information (via User Preferences).....94
  - C.11.2 Suggestion of contact information (via FH App Usage).....97
- C.12 SUBSCRIPTION INVITATION FLOW .....99**
  - C.12.1 Subscription Invitation using SIP:Message.....99
  - C.12.2 Subscription Invitation using XDM Forward.....101
- C.13 RETRIEVAL OF DELETED CONTACT INFORMATION.....102**
- C.14 COMMON CONNECTIONS .....103**
- C.15 PUBLIC CONTENT.....104**

## Figures

- Figure 1: Contact added data details .....26**
- Figure 2: Contact Search - PCC .....58**
- Figure 3: Contact Search - AB.....59**
- Figure 4 : Contact Search – External Directories .....60**
- Figure 5 : Import from non-CAB Address Book Systems.....61**
- Figure 6: CAB Client manipulating a CAB XML document.....62**
- Figure 7 : Flows of AB Forwarding in the originating side towards a CAB user .....66**
- Figure 8 : Flows of PCC forwarding in the originating side towards a CAB user .....68**
- Figure 9: Flows of AB forwarding in the terminating side towards a CAB user .....70**
- Figure 10: Flows of AB forwarding in the terminating side towards a CAB user .....71**
- Figure 11: Flows of PCC forwarding in the terminating side towards a CAB user.....73**
- Figure 12: Flows of PCC forwarding in the terminating side towards a CAB user.....75**
- Figure 13 : Flows of Contact Share towards a non CAB user .....76**
- Figure 14 : Contact Subscription flow using the Subscription Proxy .....78**
- Figure 15 : Contact Subscription flow through the reactive authorization.....80**
- Figure 16 : CAB Client Address Book Modifications and Synchronization Flow .....82**
- Figure 17 : Address Book Modifications from Network .....83**
- Figure 18 : CAB Server publishes CAB capability as Permanent Presence State .....84**
- Figure 19 : CAB Server publishes CAB capability as a regular SIP Publish .....84**
- Figure 20 : CAB Server subscribes to CAB Capability of its user’s contacts.....85**
- Figure 21 : External Profile Information Import Flows.....86**
- Figure 23 : PCC Information Export Flows .....90**
- Figure 23 : Subscription to External Profile Information Flows.....91**
- Figure 25 : Suggestion of contact information (via User Preferences) .....95**
- Figure 26 : Suggestion of contact information (via FH App Usage) .....97**
- Figure 27 : Subscription Invitation using SIP:Message .....100**
- Figure 28 : Subscription Invitation using XDM Forward.....101**

---

**Figure 29 : Retrieval of deleted contact information ..... 102**  
**Figure 30: Common Connections ..... 103**  
**Figure 31: Public Content ..... 105**

# 1. Scope

This document provides the Technical Specification of the CAB 1.1 Enabler to fulfil the requirements outlined in the Converged Address Book Requirements Document [CAB 1.1 RD] and in compliance to the architecture described in Converged Address Book Architecture Document [CAB 1.1 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 corresponding to the functionalities of the CAB Enabler. These flows explain the system concepts in a graphical manner and describe the relationships between Functional Components of the CAB Enabler architecture.

## 2. References

### 2.1 Normative References

#### IETF

- [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] “Functional Description of Event Notification Filtering”, H. Khartabil et al., September 2006, URL: <http://www.ietf.org/rfc/rfc4660.txt>
- [RFC4661] “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>

#### IMC

- [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>

#### OMA

- [CAB 1.0 AD] “Converged Address Book Architecture”, Version 1.0, Open Mobile Alliance™, OMA-AD-CAB-V1\_0, URL: <http://www.openmobilealliance.org/>
- [CAB 1.0 MO] “Converged Address Book Management Object”, Version 1.0, Open Mobile Alliance™, OMA-TS-CAB-MO-V1\_0, URL: <http://www.openmobilealliance.org/>
- [CAB 1.1 AD] “Converged Address Book Architecture”, Version 1.1, Open Mobile Alliance™, OMA-AD-CAB-V1\_1, URL: <http://www.openmobilealliance.org/>
- [CAB 1.1 FmtAd] “Converged Address Book (CAB) Specification – Format Adaptation”, Version 1.1, Open Mobile Alliance™, OMA-TS-CAB\_FormatAdaptation-V1\_0, URL: <http://www.openmobilealliance.org/>
- [CAB 1.1 RD] “Converged Address Book Requirements”, Version 1.1, Open Mobile Alliance™, OMA-RD-CAB-V1\_1, URL: <http://www.openmobilealliance.org/>
- [CAB 1.1 XDMS] “Converged Address Book XDM Specification”, Version 1.1, Open Mobile Alliance™, OMA-TS-CAB-XDMS-V1\_1, URL: <http://www.openmobilealliance.org/>
- [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/>
- [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 1.2.2, Open Mobile Alliance™, OMA-TS-DS\_DataSyncRep-V1\_2\_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/>



[SCRRULES]	“SCR Rules and Procedures”, Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[XDM AD]	“XML Document Management Architecture”, Version 2.1, Open Mobile Alliance™, OMA-AD-XDM-V2_1, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[XDM Core]	“XML Document Management Specification”, Version 2.1, Open Mobile Alliance™, OMA-TS-XDM_Core-V2_1, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[XDM List]	“List XDM Specification”, Version 2.1, Open Mobile Alliance™, OMA-TS-XDM_List-V2_1, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[XDM RD]	“XML Document Management Requirements”, Version 2.1, Open Mobile Alliance™, OMA-RD-XDM-V2_1, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[XDM UPP]	“UPP Directory XDM Specification”; Open Mobile Alliance™, OMA-TS-XDM_UPP_Directory-V1_0, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[XSD 1.0 cab pcc]	“XML Schema Definition: CAB Personal Contact Card document”, Version 1.0, Open Mobile Alliance™, OMA-SUP-XSD_cab_pcc-V1_0, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[XSD 1.1 cab pcc]	“XML Schema Definition: CAB Personal Contact Card document”, Version 1.1, Open Mobile Alliance™, OMA-SUP-XSD_cab_pcc-V1_1, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[XSD 1.1 cab subs invite]	“XML Schema Definition: CAB Subscription Invite Payload”, Version 1.1, Open Mobile Alliance™, OMA-SUP-XSD_cab_subsc_invite-V1_1, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[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: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>

## 2.2 Informative References

### OMA

[CAB APIs]	“OMA RESTful Converged Address Book (CAB) APIs”, Version 1.0, Open Mobile Alliance™, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[OMADICT]	“Dictionary for OMA Specifications”, Version 2.9, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_9, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[Pres]	“OMA Presence SIMPLE”, Version 2.0, Open Mobile Alliance™, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[Pres TS]	“OMA Presence SIMPLE Specification”, OMA-TS-Presence_SIMPLE-V2_0, Open Mobile Alliance™, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>
[PDE]	“OMA Presence SIMPLE Data Extensions V1.3” , Open Mobile Alliance™, URL: <a href="http://www.openmobilealliance.org/">http://www.openmobilealliance.org/</a>

## 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

<b>AB Document</b>	Uses definition from [CAB 1.1 AD].
<b>Access Permissions</b>	Uses definition from [XDM RD].
<b>Access Permissions Document</b>	Uses definition from [XDM AD].
<b>Active User Preferences Profile</b>	Uses definition from [XDM RD].
<b>Address Book</b>	See "Converged Address Book" in [CAB 1.1 RD].
<b>Application Unique ID</b>	Uses definition from [XDM Core].
<b>Application Usage</b>	Uses definition from [XDM Core].
<b>CAB Client</b>	Uses definition from [CAB 1.1 AD].
<b>CAB Feature Handler Document</b>	Uses definition from [CAB 1.1 AD].
<b>CAB Format</b>	An XML format containing a set of attributes representing contact information.
<b>CAB Server</b>	Uses definition from [CAB 1.1 AD].
<b>CAB User</b>	Uses definition from [CAB 1.1 RD].
<b>CAB User Preferences</b>	Uses definition from [CAB 1.1 AD].
<b>CAB User Preferences Document</b>	Uses definition from [CAB 1.1 AD].
<b>CAB Server</b>	Uses definition from [CAB 1.1 AD].
<b>CAB XDMS</b>	Uses definition from [CAB 1.1 AD].
<b>Contact Entry</b>	Uses definition from [CAB 1.1 RD].
<b>Contact Share</b>	Uses definition from [CAB 1.1 RD].
<b>Contact Status</b>	Uses definition from [CAB 1.1 AD].
<b>Contact Subscription</b>	Uses definition from [CAB 1.1 RD].
<b>Contact View</b>	Uses definition from [CAB 1.1 RD].
<b>Cross-Network Proxy</b>	Uses definition from [XDM AD].
<b>Document Reference</b>	Uses definition from [XDM AD].
<b>Document Selector</b>	Uses definition from [XDM Core].
<b>Data synchronization protocol</b>	Uses definition from [OMA DS Pro].
<b>DS Client</b>	Uses definition of "Client" from [OMA DS].
<b>DS Server</b>	Uses definition of "Server" from [OMA DS].
<b>Enabler</b>	Uses definition from [OMADICT].
<b>External Directories</b>	Uses definition from [CAB 1.1 AD].
<b>External Profile</b>	Uses definition from [CAB 1.1 RD].
<b>Favourite Contact</b>	Uses definition from [CAB 1.1 RD].
<b>Forward XDCP Request</b>	Uses definition from [XDM Core].

<b>Forwarding Notification List Document</b>	Uses definition from [XDM List].
<b>Global Document</b>	Uses definition from [XDM Core].
<b>Global Tree</b>	Uses definition from [XDM Core].
<b>History Information</b>	Uses definition from [XDM AD].
<b>Legacy Formats</b>	Uses definition from [CAB 1.1 RD].
<b>Limited XQuery over HTTP</b>	Uses definition from [XDM AD].
<b>Management Object</b>	Uses definition from [OMA DM TND].
<b>Non-CAB address book system</b>	Uses definition from [CAB 1.1 AD].
<b>Non-CAB User</b>	Uses definition from [CAB 1.1 AD].
<b>Personal Contact Card</b>	Uses definition from [CAB 1.1 RD].
<b>PCC Document</b>	Uses definition from [CAB 1.1 AD].
<b>PCC Update Document</b>	Uses definition from [CAB 1.1 XDMS].
<b>Principal</b>	Uses definition from [OMADICT].
<b>Published Contact Card</b>	Uses definition from [CAB 1.1 RD].
<b>Request History Information Document</b>	Uses definition from [XDM Core].
<b>Request-URI</b>	Uses definition from [XDM Core].
<b>UPP Directory</b>	Use definition from [XDM RD].
<b>UPP Directory Document</b>	Use definition from [XDM RD].
<b>URI</b>	Uses definition from [XDM Core].
<b>User Directory Document Selector</b>	Uses definition from [XDM Core].
<b>Users Tree</b>	Uses definition from [XDM Core].
<b>Search Document</b>	Uses definition from [XDM Core].
<b>Search Proxy</b>	Uses definition from [XDM AD].
<b>Search Request</b>	Uses definition from [XDM Core].
<b>SIP MESSAGE</b>	The SIP method MESSAGE as defined by [RFC3428]
<b>SIP NOTIFY</b>	Uses definition from [XDM Core].
<b>SIP SUBSCRIBE</b>	Uses definition from [XDM Core].
<b>SIP URI</b>	Uses definition from [XDM Core].
<b>Subscription Proxy</b>	Uses definition from [XDM AD].
<b>XDCP Document</b>	Uses definition from [XDM Core].
<b>XDCP Response</b>	Uses definition from [XDM Core].
<b>XDCP Request</b>	Uses definition from [XDM Core].
<b>XDM Agent</b>	Uses definition from [XDM AD].
<b>XDM Client</b>	Uses definition of "XDMC" from [XDM AD].
<b>XDM Document</b>	Uses definition from [XDM RD]
<b>XDM Preferences Document</b>	Uses definition from [XDM Core].
<b>XDM Resource</b>	Uses definition from [XDM RD]
<b>XDMS</b>	Uses definition from [XDM AD].

### 3.3 Abbreviations

<b>AB</b>	Address Book
<b>AUID</b>	Application Unique ID
<b>CAB</b>	Converged Address Book
<b>DM</b>	Device Management
<b>DS</b>	Data Synchronization
<b>DTD</b>	Document Type Definition
<b>HTTP</b>	Hyper Text Transfer Protocol
<b>IP</b>	Internet Protocol
<b>MIME</b>	Multipurpose Internet Mail Extension
<b>OMA</b>	Open Mobile Alliance
<b>PCC</b>	Personal Contact Card
<b>SCR</b>	Static Conformance Requirement
<b>SIP</b>	Session Initiation Protocol
<b>UPP</b>	User Preferences Profile
<b>URI</b>	Uniform Resource Identifier
<b>XCAP</b>	XML Configuration Access Protocol
<b>XDCP</b>	XDM Command Protocol
<b>XDM</b>	XML Document Management
<b>XDMC</b>	XML Document Management Client
<b>XDMS</b>	XML Document Management Server
<b>XML</b>	eXtensible Markup Language
<b>XQuery</b>	XML Query
<b>XUI</b>	XCAP User Identifier

## 4. Introduction (Informative)

The CAB 1.1 Technical Specification provides the CAB User with the features described in sub-clause [4.2](#) “*Version 1.1*”. The CAB Technical Specification utilizes the technologies of data management and synchronization [OMA DS Pro] and XML document management [XDM Core] to fulfil CAB requirements [CAB 1.1 RD] and is based on CAB architecture described in the [CAB 1.1 AD].

Since CAB 1.1 Enabler builds on existing CAB 1.0 Enabler functionality, the CAB 1.1 Technical Specification supports all the functionality defined in, and remains backwards compatible with CAB 1.0 Enabler.

### 4.1 Version 1.0

CAB 1.0 Technical Specification supports the following features as described in [CAB 1.0 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

### 4.2 Version 1.1

The CAB 1.1 Technical Specification supports the following features, in addition to the features described in sub-clause [4.1](#) “*Version 1.0*”:

- 1) Import/Export of profile information from/to 3<sup>rd</sup> party systems
- 2) Subscription to profile information in 3<sup>rd</sup> party systems
- 3) Contact Status Enhancements
  - a. Identifying Common Connections in the AB
  - b. Contact-added notifications
  - c. Contact Suggestions
  - d. Service Suggestions
  - e. Display of public content
  - f. Marking of Favourite Contacts
- 4) Subscription Invitations
- 5) Architecture flexibility i.e. the use of both XDM and/or DS to access AB XML document from CAB Client

## 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 Data synchronization protocol [OMA DS Pro] as specified in CAB-1 interface [CAB 1.1 AD] to synchronize data modifications in the AB. Further, the CAB Server SHALL support managing (e.g. retrieve, modify, delete, keep up to date) subsets of AB with the CAB Client, as specified in sub-clause 5.13 “*Data Sync Record and Field Level Filtering*” of [OMA DS].

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 “*ParamName*” to 5.3.29 “*PropParam*” of [OMA DS DevInf] in terms of properties and parameters, and
- sub-clause 8 “*Base Media and Content formats*” 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 [CAB 1.1 FmtAd].

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 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 Data synchronization, i.e. prior to sending the [OMA DS Pro] message (Pkg #6) with the map acknowledgement to the CAB Client, or if Pkg #5 and Pkg #6 are not required, prior to sending 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.

#### 5.1.1 Retrieval and Restore of previously deleted Contact Entries

To support management of ‘deleted’ Contact Entries, the CAB Server supports the following:

- 1) When receiving an OMA DS “Delete” command of Contact Entry with <temporary> element not equal to the value “deleted” of the contact, the CAB Server :
  - a) SHALL set the <temporary> element under the <entry-status> of the Contact Entry to the value “deleted”,
  - b) SHALL act as a XDM Agent (see sub-clause 5.5 “*XDM Agent*”) to manage changes in AB Document.
- 2) When receiving a OMA DS “Delete” command of Contact Entry with the <temporary> element equal to the value “deleted” of the contact, the CAB Server:
  - a) SHALL delete the <temporary> element under the <entry-status> of the Contact Entry from CAB User AB Document;
  - b) SHALL act as a XDM Agent (see sub-clause 5.5 “*XDM Agent*”) to manage of changes in AB Document.
- 3) Restore operation:
  - a) When receiving an OMA DS “Add” command of Contact Entry with the <temporary> element equal to the value “deleted” of the contact, and the ‘accept’ attribute set to “yes” the CAB Server:
    - i. SHALL remove the value of the <temporary> element under the <entry-status> of the Contact Entry,
    - ii. SHALL act as a XDM Agent (see sub-clause 5.5 “*XDM Agent*”) to manage of changes in AB Document.

## 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 by document management operations or subscribing to changes to the CAB User Preferences Document [CAB 1.1 XDMS]) to which it must establish Contact Subscription(s), from the CAB User Preferences Application Usage, with the following clarifications:

- 1) SHALL use auid "org.openmobilealliance.cab-user-prefs".
- 2) SHALL obtain the value of the 'id' attribute of <entry> elements included in entries of the <subscription-list> element, from the CAB User Preferences Document [CAB 1.1 XDMS].
- 3) SHALL use the value of the 'id' attribute of <entry> elements to create PCC subscription request(s) directly to the PCC Application Usage or via Subscription Proxy as described in [XDM Core].

If a <filter-set> element per [RFC4661] is associated with the 'id' attribute of <entry> element, 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 XML document change subscription procedures for an XDM Agent in sub-clause 6.1.2 "Subscribing to Changes in the XDM Resources" of [XDM Core].

When the CAB Client (via XDMC) changes a <filter-set> element in an <entry> of the <subscription-list> element, 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 1.1 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 [XDM Core].

In case the CAB User to which the Contact Subscription needs to be established is not a contact in the AB Document, the Contact Subscription SHALL add it as a new Contact Entry in AB 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 Preferences Document [CAB 1.1 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 Document in [CAB 1.1 XDMS] with the resulting changes. If the CAB User Preferences element <notify-on-contact-subscription-update> is set to 'true' then the Contact Subscription Function SHALL set in the AB 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 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 Document in [CAB 1.1 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 an 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 either CAB User or 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 1.1 XDMS]. This is accomplished either by polling the document using 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 with the following clarification:

- 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 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) . For recipients that are in the CAB User’s AB, the recipient type may be determined based on <contact-type> element of <contact-status> of the AB App Usage;

If the recipient is a CAB User:

- a. it SHALL construct the XDCP Request using the data retrieved from the CAB Feature Handler Document [CAB 1.1 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 Request; and



- v. SHALL include <delivery-report> element in the XDCP Request with the value “true” if <delivery-report-request> element in the <contact-share> is set to “true”.
- b. SHALL send the XDCP request towards the recipient’s CAB XDMS via the XDM Agent.

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. It SHALL check the <format> element (if present) from the retrieved Contact Share request data, to determine the format to be used for encoding and delivery of the Contact Share data. If <format> element is not present, the default format value from the <contact-share-format> element in CAB User Preferences XML [CAB 1.1 XDMS] document is used.
- c. 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 (i.e. based on the previous step) via the Interworking Function (see sub-clause 5.4 “Interworking Function”).
- d. When CPM Interworking Function is used as messaging delivery mechanism, it SHALL construct a message as described in [CPM CONV FCT TS] and send a message towards the CPM Interworking Selection Function [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 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 Document [XDM List].

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 [XDM List] through either by polling the document using document management operations or by subscribe/notify mechanisms of [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 [XDM List] for the Forward XDCP Request.
  - The “Pending” value is set when there is no delivery Report available yet for the Forward XDCP Request.
  - The “Successful” value is set when the <status> element of the Forward Delivery Report XDCP Request is either “delivered” or “rejected”.
  - The “Failure” value is set when the <status> element of the Forward Delivery Report XDCP Request 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 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 based on the section 5.8.7.2 “Forward-prefs Elements” of [XDM Core] is “accept”, it SHALL store the contact shared data into the appropriate <person-details>, <group-details> and/or <org-details> elements under the <contact> element and set 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 based on the section 5.8.7.2 “Forward-prefs Elements” of [XDM Core] is “confirm”, it SHALL store the contact shared data into the appropriate <person-details>, <group-details> and/or <org-details> elements under the <contact> and set <temporary> child element of the <entry-status> 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)
- Import profile information from 3<sup>rd</sup> party system(s)
- Export PCC information to 3<sup>rd</sup> party system(s)
- Subscription to profile information in 3<sup>rd</sup> party 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 1.1 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 <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 <expiration-time> element to indicate the time period of scheduling the import request to the 3<sup>rd</sup> party system. The maximum and default time periods for scheduling a import request SHALL be determined by the Service Providers local policy. If <expiration-time> is not present a recurrent import is kept active until a request to terminate it is received or until it is terminated according to the Service Providers local policy;
- MAY use either the <credentials> element or <token> element 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 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 Document to “Failure”.

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

- 1) SHALL use <import-update> element from CAB User Preferences Document [CAB 1.1 XDMS] to identify how to store the imported contacts, as follows:
  - a) When <import-update> element is “true”, the Interworking Function SHALL update AB Document in [CAB 1.1 XDMS] with the resulting changes and SHALL set the value of <updated> element as “contact imported”.
  - b) When <import-update> element is “false”, Interworking Function SHALL store the imported contact to a new <contact> element in AB Document [CAB 1.1 XDMS], and SHALL set the value of <temporary> element as “contact imported”.
- 2) SHALL set the value of <contact-source> element in AB Document [CAB 1.1 XDMS] to the value of <non-CAB-source> element in CAB Feature Handler Document [CAB 1.1 XDMS].
- 3) SHALL set the value of the <code> element under <response> element of the corresponding <import-non-cab> element of the CAB Feature Handler Document [CAB 1.1 XDMS] to “Success”.
- 4) SHALL wait the time defined in the element <scheduled-interval> before performing a subsequent import, unless <scheduled-interval> is set to ‘0’ in which case no further import will be performed.

The CAB Server performs the AB synchronization as specified in sub-clause [5.1](#) “AB Synchronization Function” in order to inform the CAB Client of the AB updates.

## 5.4.2 Import profile information from 3rd Party Systems

The data for the scheduled import requests of profile information from 3<sup>rd</sup> party systems into CAB user's PCC SHALL be retrieved from the CAB Feature Handler Application Usage [CAB 1.1 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 3<sup>rd</sup> party system from which to import
- SHALL use the <scheduled-interval> element to determine the interval between successive accesses to the 3<sup>rd</sup> party system
- MAY use the <expiration-time> element to indicate the time period of scheduling the import request to the 3<sup>rd</sup> party system. The maximum and default time periods for scheduling a import request SHALL be determined by the Service Providers local policy. If <expiration-time> is not present a recurrent import is kept active until a request to terminate it is received or until it is terminated according to the Service Providers local policy;
- MAY use either the <credentials> element or <token> element to obtain access to the 3<sup>rd</sup> party system
- SHALL use the a <filter-set> element, if included, to filter the profile information to be imported from the 3<sup>rd</sup> party system

While processing the import request, the Interworking Function MAY set the value of the <code> under <response> element of the corresponding <import-profile> element of the CAB Feature Handler 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-profile> element of the CAB Feature Handler Document to "Failure".

After successful imported and format adaptation of the received profile information, the Interworking Function:

- 1) SHALL use <pcc-import-update> element from CAB User Preferences Document [CAB 1.1 XDMS] to identify how to store the results from imported External Profile information:
  - a) When <pcc-import-update> element is "true", Interworking Function SHALL automatically update the PCC Document in [CAB 1.1 XDMS] with the resulting changes and store the meta information of updated PCC Document in the PCC Document as follows:
    - a) SHALL create a new <pcc-update> element;
    - b) SHALL NOT include the <update-ref> element unless the stored PCC Update Document does exist;
    - c) SHALL set the <update-type> element to the value "import-profile";
    - d) SHALL set the <update-source> element with the value of <non-CAB-source> from the CAB Feature Handler Document request;
    - e) SHALL set the <time-stamp> element with the current 'dateTime' value; and
    - f) SHALL set the <approval-type> element to the value "automatic".
  - b) When <pcc-import-update> element is "false", Interworking Function SHALL store the imported External Profile information into a new PCC Update Document in [CAB 1.1 XDMS], and store the reference to the PCC Update Document in the PCC Document as follows:
    - a) SHALL create a new <pcc-update> element;
    - b) SHALL set the <update-ref> element to a valid XCAP URI pointing to the stored PCC Update Document;
    - c) SHALL set the <update-type> element to the value "import-profile";
    - d) SHALL set the <update-source> element with the value of <non-CAB-source> from the CAB Feature Handler Document request;

- e) SHALL set the <time-stamp> element with the current 'dateTime' value; and
  - f) SHALL set the <approval-type> element to the value "manual".
- 2) SHALL set the value of the <code> element under <response> element of the corresponding <import-profile> element of the CAB Feature Handler Document [CAB 1.1 XDMS] to "Success".
  - 3) SHALL wait the time defined in the element <scheduled-interval> before performing a subsequent import, unless <scheduled-interval> is set to '0' in which case no further import will be performed.

The CAB Client is notified of the change in the PCC either by SIP NOTIFY or Push OTA Message depending on whether the subscription was a SIP-based subscription or an XDCP-based subscription as described in [6.13.1](#) 'Subscriptions to CAB User's own PCC Document changes'.

### 5.4.3 Export PCC information to 3rd Party Systems

The data for the scheduled export requests of PCC information to 3<sup>rd</sup> party systems SHALL be retrieved from the CAB Feature Handler Application Usage [CAB 1.1 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 export 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 3<sup>rd</sup> party system to which to export
- SHALL use the <scheduled-interval> element to determine the interval between successive export operations to the 3<sup>rd</sup> party system
- MAY use the <expiration-time> element to indicate the time period of scheduling the export request to the 3<sup>rd</sup> party system. The maximum and default time periods for scheduling a export request SHALL be determined by the Service Providers local policy. If <expiration-time> is not present a recurrent export is kept active until a request to terminate it is received or until it is terminated according to the Service Providers local policy;
- MAY use either the <credentials> element or <token> element to obtain access to the 3<sup>rd</sup> party system
- SHALL use the <filter-set> element, if included, to filter the profile information to be exported to the 3<sup>rd</sup> party system

While processing the export request, the Interworking Function MAY set the value of the <code> under <response> element of the corresponding <export-profile> element of the CAB Feature Handler 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 <export-profile> element of the CAB Feature Handler Document to "Failure".

After successful export the Interworking Function:

- 1) SHALL set the value of the <code> element under <response> element of the corresponding <export-profile> element of the CAB Feature Handler Document [CAB 1.1 XDMS] to "Success".
- 2) SHALL wait the time defined in the element <scheduled-interval> before performing a subsequent export, unless <scheduled-interval> is set to '0' in which case no further export will be performed.

### 5.4.4 Subscription to profile information in 3rd Party Systems

The data for the subscription requests to profile information in 3<sup>rd</sup> party systems SHALL be retrieved from the CAB Feature Handler Application Usage [CAB 1.1 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 subscription 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 3<sup>rd</sup> party system to which to subscribe to
- MAY use the <expiration-time> element to indicate the time period of keeping the subscription request active to the 3<sup>rd</sup> party system. The maximum and default time periods for keeping a subscription request active SHALL be determined by the Service Providers local policy. If <expiration-time> is not present the subscription is kept active until a request to terminate it is received or until it is terminated according to the Service Providers local policy;
- MAY use either the <credentials> element or <token> element to obtain access to the 3<sup>rd</sup> party system
- SHALL use the <filter-set> element, if included, to filter the profile information to be subscribed from the 3<sup>rd</sup> party system

While processing the subscribe request, the Interworking Function MAY set the value of the <code> under <response> element of the corresponding <subscribe-profile> element of the CAB Feature Handler 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 <subscribe-profile> element of the CAB Feature Handler Document to “Failure”.

After successful reception of updates from the 3<sup>rd</sup> party system and format adaptation of the received profile information, the Interworking Function:

- 1) SHALL use <pcc-subscribe-update> element from CAB User Preferences Document [CAB 1.1 XDMS] to identify how to store the results from subscription to External Profile information:
  - a) When <pcc-subscribe-update> element is “true”, the Interworking Function SHALL automatically update the PCC Document in [CAB 1.1 XDMS] with the resulting changes and store the meta information of updated PCC Document in the PCC Document as follows:
    - SHALL create a new <pcc-update> element;
    - SHALL NOT include the <update-ref> element unless the stored PCC Update Document does exist;
    - SHALL set the <update-type> element to the value “subscribe-profile”;
    - SHALL set the <update-source> element with the value of <non-CAB-source> from the CAB Feature Handler Document request;
    - SHALL set the <time-stamp> element with the current ‘dateTime’ value; and
    - SHALL set the <approval-type> element to the value “automatic”.
  - b) When <pcc-subscribe-update> element is “false”, Interworking Function SHALL store the resulting changes from External Profile information into a new PCC Update Document in [CAB 1.1 XDMS], and store the reference to the PCC Update Document in the PCC Document as follows:
    - SHALL create a new <pcc-update> element;
    - SHALL set the <update-ref> element to a valid XCAP URI pointing to the stored PCC Update Document;
    - SHALL set the <update-type> element to the value “subscribe-profile”;
    - SHALL set the <update-source> element with the value of <non-CAB-source> from the CAB Feature Handler Document request;
    - SHALL set the <time-stamp> element with the current ‘dateTime’ value; and
    - SHALL set the <approval-type> element to the value “manual”.
- 2) SHALL set the value of the <code> element under <response> element of the corresponding <subscribe-profile> element of the CAB Feature Handler Document [CAB 1.1 XDMS] to “Success”.
- 3) SHALL process subsequent updates from 3<sup>rd</sup> party system, accordingly to service provider’s policy.

The CAB Client is notified of the change in the PCC either by SIP NOTIFY or Push OTA Message depending on whether the subscription was a SIP-based subscription or an XDMP-based subscription as described in [6.13.1](#) ‘Subscriptions to CAB User’s own PCC Document changes’.



## 5.4.5 Contact Search with External Directories

The CAB Interworking Function SHALL support searches to External Directories, by hosting a standard XML format (see below sub-clause [5.4.5.1](#) “*Application Usage for External Directories Search*”) 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 [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 [5.4.5.1.5](#) “*Extension for <dataSource>*”) 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 results prior to sending the response back to the Search Proxy.

### 5.4.5.1 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 [5.4.5.1.4](#) “*XML Schema*”) and stored in the Global Tree of the Application Usage as described in [XDM Core] sub-clause 5.5 “*Global Documents*”.

#### 5.4.5.1.1 Application Unique ID

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

#### 5.4.5.1.2 Default Namespace

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

#### 5.4.5.1.3 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 [5.4.5.1.4](#) “*XML Schema*” and the following rules apply in addition to the procedures defined in sub-clause 5.4.1 “*Search Document*” of [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 [XDM Core] in sub-clause 6.2.3 “*Searching for Data in XML Documents*”.

#### 5.4.5.1.4 XML Schema

The standard XML format for external search SHALL conform to the PCC XML schema as described in [XSD 1.1 cab pcc].

#### 5.4.5.1.5 Extension for <dataSource>

Additional CAB extension to the search document as described in [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]

## 5.4.6 Format Adaptation

The format adaptation between CAB Format and Legacy Format(s) SHALL be supported by the Interworking Function as described in [CAB 1.1 FmtAd].

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

If CAB Server populates the CAB capability (i.e. CAB / non-CAB) through Presence Enabler [Pres] and the CAB Server publishes the CAB capability as Permanent Presence State, then the XDM Agent SHALL interact with the Presence XDMS for this purpose as described in [Pres TS] Section 5.1.3 “*Manipulation of Permanent Presence State using XCAP*”.

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

- Procedures of document management of CAB XML documents [CAB 1.1 XDMS] based on sub-clause 6.1.1 “*Document Management*” from [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 1.1 XDMS] as described in sub-clause “6.1.2 “*Subscribing to Changes in the XML Resources*” with the exception of sub-clause 6.1.2.1.2 “*XDMC*” in [XDM Core].
- Procedure of Document Reference information management for CAB AB Document as described in sub-clause 6.1.1.3.1 “*Document Reference Operations*” (only XDM Agent part) of [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 Resources*” with the exception of sub-clause 6.1.2.1.2 “*XDMC*” [XDM Core].
- Procedures of history information access for CAB XML documents [CAB 1.1 XDMS] based on sub-clause 6.1.4.2 “*Request History Information*” in [XDM Core]
- Procedures of XDM Forwarding for CAB XML documents [CAB 1.1 XDMS] based upon sub-clause 6.1.1.3.2 “*XDM Resource Forwarding Operations*” in [XDM Core]

The XDM Agent SHALL support the Application Usages specified in [CAB 1.1 XDMS].

## 5.6 Presence Source

If CAB Server populates the CAB capability (i.e. CAB / non-CAB) through Presence Enabler [Pres] and the CAB Server publishes the CAB capability as a SIP Publish, then the Presence Source SHALL interact with the Presence Server for this purpose as described in [Pres TS] Section 5.1.2 “*Publication of Presence Information using SIP*”.

## 5.7 Presence Watcher

If CAB Server populates the CAB capability (i.e. CAB / non-CAB) through Presence Enabler [Pres], then the Presence Watcher SHALL subscribe to the Presence information of the contacts of the CAB Users served by this CAB Server as described in [Pres TS] Section 5.2 “*Watcher*”, in order to receive the CAB capability of those contacts as part of the Presence information updates.

## 5.8 Contact Status Function

The Contact Status Function SHALL manage the Contact Status information contained in the <contact-status> element of the Contact Entry following the rules and procedures as specified in AB Application Usage [CAB 1.1 XDMS].

### 5.8.1 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.

While several mechanisms are possible to determine the CAB capability of other users (i.e. CAB / non-CAB <contact-type>), such as: one time Contact Subscription, contact search, exchange of CAB capability through Presence Server [Pres], none is mandated by 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.
- Contact Share – See sub-clause [5.3](#) “*Contact Share Function*” for procedures to populate the Contact Status information associated with incoming Contact Share information.
- 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.
- Contact type update – See sub-clause [5.8.2](#) “*Population of CAB Capability based on the Presence Enabler*” for procedures to populate the Contact Status information associated with a contact type update received via the Presence Enabler.
- Incoming Contact Subscription Request - See sub-clause [5.8.3](#) “*Incoming Contact Subscription Requests*” for procedures to populate the Contact Status information associated with the incoming Contact Subscription Requests.
- Contact Added – See sub-clause [5.8.4](#) “*Contact Added*” for procedures to populate the Contact Status associated with contact added information.
- Common Connections – See sub-clause [5.8.5](#) “*Common Connections*” for procedures to populate the Contact Status associated with Common Connections information.
- Contact suggestions – See sub-clause [5.8.6](#) “*Suggestion of contact information*” for procedures to populate the Contact Status to indicate contact suggestions.
- Service suggestions – See sub-clause [5.8.7](#) “*Service suggestions*” for procedures to populate the Contact Status to indicate service suggestions.
- Public content – See sub-clause [5.8.8](#) “*Public content*” for procedures to populate the Contact Status to indicate public contact per contact in the AB.
- Subscription Invitation – See sub-clause [5.10](#) “*Subscription Invitation Function*”, for procedures to populate the Contact Status information associated with the incoming subscription invitations.

### 5.8.2 Population of CAB Capability based on the Presence Enabler

If CAB Server populates the CAB capability (i.e. CAB / non-CAB) through Presence Enabler [Pres] then the following procedures SHALL be followed.

The CAB tuple in the presence document is defined according to [PDE], using the <service-description> registered by OMNA:

```

<tuple >      -><status>-><basic>->open/closed
               -><willingness>-><basic>->open/closed
               -><service-description>-><service-id>->org.openmobilealliance:CAB
               -><version>->1.1
  
```



→<contact>→ tel:+1-123-456-7890

When a user becomes a CAB user then the <status><basic> and <willingness><basic> elements in the CAB tuple SHALL be set to “open” and the <contact> element to the CAB XUI. This can be done either as a SIP Publish by the Presence Source or by modification of the Permanent Presence State by the XDM Agent.

When a user unsubscribes from CAB then the <status><basic> and <willingness><basic> elements in the CAB tuple SHALL be set to “closed”. This can be done either as a SIP Publish by the Presence Source or by modification of the Permanent Presence State by the XDM Agent.

The Presence Watcher SHALL subscribe to the Presence information of the CAB Users’ contacts served by the CAB Server by either of the following procedures:

- Individual Presence subscriptions to each of the contacts in the address book of each of the CAB Users, on behalf of these users. Multiple subscriptions to a specific contact may be performed if that contact is included in the address books of more than one CAB User.
- Single anonymous Presence subscription for each contact included in at least one address book of the CAB Users served by that CAB server.

When the Presence Watcher receives a presence notification then it will check the changes in the CAB tuple and update the <contact-type> element in the AB Application Usage as detailed below. Notifications associated to an anonymous subscription will result in updates to the AB Application Usage of all CAB Users with that contact in their address books. Notifications associated to an individual Presence subscription will result in updates to the AB Application Usage of the specific CAB User on which behalf that subscription was issued.

Upon reception of a Presence notification with the <status><basic> element in a contact’s CAB tuple with value “open” and in case that contact’s <contact-type> in the AB Application Usage was not present (meaning a non-CAB user) then it SHALL be included, the sub-element < type> SHALL be set to “CAB”, the sub-element <contact-type-source> SHALL be set to “presence” (reflecting that this information has been obtained through the Presence Enabler) and the address received in the <contact> element in the CAB tuple SHALL be marked as the CAB XUI in the AB Application Usage by setting the corresponding “xui-type” attribute to “CAB”.

Upon reception of a Presence notification with no CAB tuple for a contact or with the <status><basic> element in a contact’s CAB tuple with value “closed” and provided that contact’s < type> was set to “CAB” and the <contact-type-source> was set to “presence” (meaning that this information was obtained through the Presence Enabler), then the <contact-type> element SHALL be removed altogether (meaning that the user is no longer a CAB user). In case that the <contact-type-source> is not set to “Presence” then no action is performed as the <contact-type> information may have been obtained from other sources and the absence of the CAB tuple may mean that the Presence Server has no information about the CAB capability of its users.

### 5.8.3 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:

Note: The first two procedures are pre-conditions for the Contact Status Function to be able to detect the incoming Contact Subscription Request.

- The Contact Status Function SHALL subscribe to changes of the Request History Information Document in the PCC Application Usage via XDM Agent based on the procedures described in section 6.1.2 “*Subscribing to Changes in the XDM Resources*” of [XDM Core] in order to detect the incoming Contact Subscription request.
- On behalf of the CAB User, the Contact Status Function SHALL set the <history-prefs> element in XDM Preferences Document in PCC Application Usage [CAB 1.1 XDMS] as described in [XDM Core], sub-clause 5.8 “*XDM Preferences Document*” in such a way that the Request History Information Document (as described in [XDM Core], sub-clause 5.7.2 “*Request History Information Document*”) is updated with unsuccessful incoming Contact Subscription requests.
- Upon receiving the change notification for the Request History Document of the PCC Application Usage, the Contact Status Function SHALL use the <notify-when-receive-contact-subscription> element in CAB User

Preferences Document to check the notification preferences of the CAB User for incoming Contact 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 Contact Subscription requests via XDM Agent (see sub-clause 5.5 “XDM Agent”), and:
  - o If the CAB User’s AB Document [CAB 1.1 XDMS] contains a Contact Entry associated with the requestor of the incoming Contact 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 Document [CAB 1.1 XDMS] does not contain a Contact Entry associated with the requestor of the incoming Contact Subscription request, a new <contact> element SHALL be created in the AB 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.

### 5.8.4 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 Document. The interdomain conveyance of CAB User addition to other CAB User’s Address Book is achieved through SIP MESSAGE method as described in the following sub-clause.

#### 5.8.4.1 Originating Side

If the <send-notification-contact-added> element of the CAB User Preferences Document [CAB 1.1 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 1.1 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 1.1 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 [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 [XDM Core] and send the SIP MESSAGE according to the procedures of the SIP/IP Core that is referenced in [XDM Core];

#### 5.8.4.2 Receiving Side

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

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

- Check if the CAB feature tag "3gpp-service.ims.icsi.oma.cab\_1.0" is 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:

- If the adding CAB User is already in the CAB User's AB, one of the following conditions SHALL apply,

- if a <connection> element is not already created under the adding CAB User's Contact Entry in the CAB User's AB, the Contact Status Function SHALL create a new <connection> element under the <common-connections> element of the <contact-status> element and;
  - SHALL set the <display-name> element with the value of 'display-name' parameter of the "From" header of the received SIP MESSAGE fragment
  - SHALL set the <XUI> element with the value of the XUI present in the "From" header
  - SHALL create new <type> element under the <type-list> element and set the value of <type> element to "contact-added-mutual" to indicate that the connection is of type contact-added-mutual, since CAB User and the contact have both mutually added each other.
- if a <connection> element is already created under the adding CAB User's Contact Entry in the CAB User's AB, the Contact Status Function SHALL perform the following:
  - If a <type> element with a value of "contact-added-non-mutual", it SHALL be replaced with a value of "contact-added-mutual"
  - If a <type> element with a value of "contact-added-non-mutual" does not exist, then it SHALL create a new <type> element and set the value of <type> element to "contact-added-mutual"
- If the adding CAB User is not in the CAB User's AB, the Contact Status Function SHALL create a new Contact Entry in the AB Application Usage [CAB 1.1 XDMS], and set the <temporary> element under the <entry-status> element within the <contact-status> with a value 'contact added'. The <contact> element MAY be populated with the available PCC information of the added CAB User, based on the XUI value in the 'CAB Contact Added' fragment received.

## 5.8.5 Common Connections

The CAB Enabler supports the following types of Common Connections as described under the Data Semantics section of the AB App Usage [CAB 1.1 XDMS].

- Address book
- Watcher
- Contact Added Mutual (see sub-clause [5.8.4](#) "Contact Added" for procedures)
- Contact Added Non-mutual (see sub-clause [5.8.4](#) "Contact Added" for procedures)

The detection of Common Connections for types 'Address Book' and 'Watcher' is out of scope of this specification. However, when such Common Connections are identified, the CAB Server SHALL use Contact Status in the CAB User's AB to indicate the Common Connections to the CAB User, subject to CAB User Preferences i.e. <notify-common-connections> element in the CAB User Preferences XML Document [CAB 1.1 XDMS].

### 5.8.5.1 Address Book

If a Common Connection of type 'address-book' is identified for a Contact Entry in the CAB User's AB, one of the following conditions SHALL apply;

- If a <connection> element is not already present for the identified Common Connection, the Contact Status Function SHALL create a new <connection> element under the <common-connections> element of the <contact-status> element and:
  - SHALL set the <display-name> element with the value of 'display-name' of the Common Connection
  - SHALL set the <XUI> element to the value of 'XUI' of the Common Connection
  - SHALL create a new <type> element under the <type-list> element and set the value of <type> element to "address book"
- If a <connection> element is already present for the identified Common Connection, the Contact Status Function:
  - SHALL create a new <type> element under the <type-list> element and set the value of <type> element to "address book"

### 5.8.5.2 Watcher

If a Common Connection of type ‘watcher’ is identified for a Contact Entry in the CAB User’s AB, one of the following conditions SHALL apply;

- If a <connection> element is not already present for the identified Common Connection, the Contact Status Function SHALL create a new <connection> element under the <common-connections> element of the <contact-status> element and:
  - o SHALL set the <display-name> element with the value of ‘display-name’ of the Common Connection
  - o SHALL set the <XUI> element to the value of ‘XUI’ of the Common Connection
  - o SHALL create a new <type> element under the <type-list> element and set the value of <type> element to “watcher”
- If a <connection> element is already present for the identified Common Connection, the Contact Status Function:
  - o SHALL create a new <type> element under the <type-list> element and set the value of <type> element to “watcher”

## 5.8.6 Suggestion of contact information

The suggested contact data is delivered to the CAB User through Contact Status associated with the Contact Entry. The suggested contact data may be retrieved from an external public contact database or from within the CAB system.

Note: The logic of generating the suggested data, however, is out of scope of this specification.

### 5.8.6.1 Triggering of contact suggestions

The contact suggestions may be triggered with one of the following mechanisms;

- 1) Enabling <allow-suggested-contact-info> preference in the CAB User Preferences App Usage;
  - a) The CAB Server SHALL use either subscription operation or document management operation as specified in section 6.1 “*Procedures at the XDMC and the XDM Agent*” of [XDM Core] to obtain the CAB User Preferences XML Document [CAB 1.1 XDMS], to check if the recipient CAB User’s <allow-suggested-contact-info> user preference is set to ‘true’
 

If the <allow-suggested-contact-info> is ‘true’, then CAB Server SHALL generate the contact suggestions, and provide them to the CAB User as described in sub-clause [5.8.5.2](#) “*Delivery of contact suggestions*”
- 2) Receiving a request from CAB Client using <contact-suggest> element via the CAB Feature Handler Document.
  - a) The CAB Server SHALL use either subscription operation or document management operation as specified in section 6.1 “*Procedures at the XDMC and the XDM Agent*” of [XDM Core] to obtain the <contact-suggest> element of the CAB Feature Handler Document [CAB 1.1 XDMS]

The CAB Server SHALL obtain the data for the contact suggestions request via XDM Agent (see sub-clause [5.5](#) “*XDM Agent*”) from the CAB Feature Handler Document as follows.

- SHALL set the AUID to “org.openmobilealliance.cab-feature-handler”
- MAY use the <non-CAB-source> element to identify the external source (e.g. domain name) with which to receive contact suggestions.
- MAY use the <scheduled-interval> element to determine the interval between successive contact suggestion requests
- MAY use the <expiration-time> element to indicate the time period of scheduling the contact suggestions request to external sources. The maximum and default time periods for scheduling a contact suggestion request SHALL be determined by the Service Providers local policy. If <expiration-time> is not present, a recurrent contact suggestion request is kept active until a request to terminate it is received or until it is terminated according to the Service Providers local policy;
- MAY use either the <credentials> element or <token> element to obtain access to the external source
- MAY use the <criteria> element indicating suggestion criteria

- b) While processing the contact suggestions request, the CAB Server MAY set the value of the <code> under <response> element of the corresponding <contact-suggest> element of the CAB Feature Handler Document to “Pending”.
- c) If the CAB Server failed to process the request, it SHALL set the value of the <code> under <response> element of the corresponding <contact-suggest> element of the CAB Feature Handler Document to “Failure”.
- d) After successfully processing the request including format adaptation (if applicable), the suggestions are provided to the CAB User as described in sub-clause [5.8.5.2](#) “*Delivery of contact suggestions*”

### 5.8.6.2 Delivery of contact suggestions

In order to deliver the contact suggestions, the CAB Server SHALL perform the following:

- SHALL analyze the suggestions with the contact information available within the CAB system, to identify the contact suggestions to be provided to the CAB User.
- SHALL create a new <contact> element in AB Document [CAB 1.1 XDMS], and SHALL set the value of <temporary> element under <entry-status> element as “suggested”;
- SHALL populate the <contact> element with available contact information to suggest to the CAB User;
- SHALL set the <contact-source> element with name of the source from where the suggested contact information was obtained from;
- If the suggested contact information is associated with an existing Contact Entry, the CAB Server SHALL set the ‘contactIdRef’ attribute to match the ‘id’ attribute value of an existing <contact> element

The CAB Server performs the AB synchronization as specified in sub-clause 5.1 “*AB Synchronization Function*” in order to inform the CAB Client of the AB updates.

Note: The CAB User then accepts or rejects the <temporary> element (see sub-clause [6.15](#) “*Contact Status Management*”). In the case of the CAB User accepts the suggested contact, the CAB Client removes the <temporary> element associated with the contact.

### 5.8.7 Service suggestions

Service suggestions are delivered to the CAB User through Contact Status associated with the Contact Entry. The suggested services (and related information) are provided by the Service Provider

Note: The logic of generating the suggested service information, however, is out of scope of this specification.

Prior to delivering the service suggestions to the CAB User, the CAB Server SHALL use either subscription operation or document management operation as specified in section 6.1 “*Procedures at the XDMC and the XDM Agent*” of [XDM Core] to obtain the CAB User Preferences XML Document [CAB 1.1 XDMS], to check if the recipient CAB User’s <notify-service-suggestions> user preference is set to ‘true’

If the <notify-service-suggestions> preference is ‘true’, the CAB Server SHALL perform the following for each service suggestion:

- Identify the <contact> element to which the service suggestions is associated to;
- SHALL create a new <suggestion> element under <service-suggestions> element of the <contact-status> element.
- SHALL set the <description> element with the description of the suggested service;
- MAY set the <label> element with the name of the suggested service;
- MAY set the <url> element with a valid URL of the suggested service
- MAY set the <address> element with the address and/or location of the suggested service
- MAY set the <comm-addr> element with the communication address for the suggested service
- MAY set the <icon> element with a valid URL of the service icon representing the suggested service
- MAY set the <expiration> element with the validity period expressed as an absolute time of the suggested service

The CAB Server MAY remove the service suggestions based on the Service Provider's policy, and in conjunction with the value of <expiration> element.

## 5.8.8 Public content

Public content per contact is delivered to the CAB User through Contact Status associated with the Contact Entry. The public content (e.g. blogs, RSS feeds, news) is generated by the Service Provider, and subject to CAB User Preferences [CAB 1.1 XDMS].

Note: The logic of generating the public content, however, is out of scope of this specification.

Prior to delivering the public content to the CAB User, the CAB Server SHALL use either subscription operation or document management operation as specified in section 6.1 "*Procedures at the XDMC and the XDM Agent*" of [XDM Core] to obtain the CAB User Preferences XML Document [CAB 1.1 XDMS], to check if the recipient CAB User's <notify-public-content> user preference is set to 'true'

If the <notify-public-content> preference is 'true', the CAB Server SHALL perform the following for each instance of public content that is generated:

- Identify the <contact> element to which the public content is associated to;
- SHALL create a new <content-entry> element under <public-content> element of the <contact-status> element;
- SHALL set the <type> element indicating the type of public content (e.g. blog, RSS, news, etc)
- SHALL set the <url> element indicating the URL to the public content entry;
- MAY set the <title> element indicating the title of the public content entry;
- MAY set the <source> element indicating the source of the public content entry;
- MAY set the <time> element indicating the date and time of the public content entry.

The CAB Server MAY remove or update the existing public content entries based on the Service Provider's policy, and in conjunction with the value of <time> element.

## 5.9 Applying CAB User Preferences

The CAB Server SHALL act as XDM Agent as described in sub-clause [5.5](#) "*XDM Agent*" to retrieve the UPP Directory Document [XDM UPP] of each CAB User and identify the Active User Preferences Profile associated with each of CAB User's devices. After identifying the Active User Preferences Profile, the CAB Server SHALL apply the preferences stored in the CAB User Preferences Document [CAB 1.1 XDMS] related to the Active User Preferences Profile.

## 5.10 Subscription Invitation Function

The Subscription Invitation Function is responsible for handling CAB User's outgoing subscription invitation requests to other CAB Users, and incoming subscription invitation requests.

### 5.10.1 Procedures at the Originating Side

The CAB Server SHALL use XDM Agent (see sub-clause [5.5](#) "*XDM Agent*") to retrieve CAB User's subscription invitation requests stored in the CAB Feature Handler Application Usage [CAB 1.1 XDMS]. This is accomplished either by polling the document using document management operations or subscribing to changes to the CAB Feature Handler Document, with the following clarifications:

- SHALL set the AUID to "org.openmobilealliance.cab-feature-handler"
- SHALL obtain the subscription invitation requests by the extracting the data under <subscription-invite> element and its child elements

Cross-domain subscription invitation requests are supported through the following two mechanisms (i.e. using [SIP:MESSAGE](#) message or XDCP Forward).

#### 5.10.1.1 Using cross-domain SIP:MESSAGE

Upon obtaining the subscription invitation requests, the CAB Server SHALL process the request(s) as follows:



- SHALL use the <recipients-list> element to determine the recipients of the subscription invitation request.
- SHALL use the <contact-view> element to determine the Contact View to be transmitted to the recipient
- SHALL use the <subscription-authorization> element to determine whether access permissions are to be granted to the recipient for future incoming Contact Subscription requests. If set to 'true', the CAB Server SHALL modify CAB User's PCC Access Permissions as described in sub-clause 5.2.1.11 "Access Permissions" of PCC Application Usage [CAB 1.1 XDMS].
- SHALL use the <invitation-message> element to determine the custom personal message to be associated with the outgoing invitation request

After processing the subscription invitation request, the CAB Server SHALL construct and send the cross-domain [SIP:MESSAGE](#) [RFC 3428] according to SIP/IP Core [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 'uri' attribute of the <entry> element of the <recipients-list>).
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 CAB User's XUI and the 'display-name' parameter of the From header to the CAB User'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 CAB User's XUI;
5. Set the Accept-Contact header field with the value of the CAB feature tag, i.e. "3gpp-service.ims.icsi.oma.cab\_1.1";
6. Set the P-Preferred-Service header field with the value of the CAB feature tag, i.e. "3gpp-service.ims.icsi.oma.cab\_1.1";
7. Set the MIME to application/vnd.oma.cab-sub-invite+xml. See sub-clause 5.10.3 "MIME Type for Cross-domain Subscription Invitations" for the MIME definition.
8. Set the other required SIP headers according to [XDM Core] and send the SIP MESSAGE according to the procedures of the SIP/IP Core that is referenced in [XDM Core];

### 5.10.1.2 Using XDCCP Forward

Upon obtaining the subscription invitation requests, the CAB Server SHALL initiate an XDCCP Request as described in sub-clause 5.3.1 "Procedures at the Originating Side" of section 5.3 "Contact Share Function", to forward subscription invite with the following clarifications;

- SHALL update CAB User's PCC Access Permission for granting the subsequent contact subscription by the recipients if the recipient is a CAB User.
- SHALL include <note> element which is mapped to <invitation-message> element containing a personal invitation message to be associated with the invitation request.

## 5.10.2 Procedures at the Receiving Side

### 5.10.2.1 Using cross-domain SIP:MESSAGE

The CAB Server SHALL receive the incoming subscription invitation requests from the remote domain using SIP MESSAGE request.

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

- Check if the CAB feature tag "3gpp-service.ims.icsi.oma.cab\_1.1" is 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 headers (i.e. validate the value of "To" and "From" headers);
- 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 a MIME type 'application/vnd.oma.cab-sub-invite+xml', the CAB Server SHALL interpret the message to be an incoming subscription invitation request. See sub-clause 5.10.3 "MIME Type for Cross-domain Subscription Invitations" for the MIME definition.
- 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 are invalid.

Upon successfully receiving and processing the incoming subscription invitation request, the CAB Server SHALL verify if the recipient CAB User's preference <notify-contact-subscription-invite> [CAB 1.1 XDMS] is set to 'true', and then deliver the notification of the incoming contact subscription to the CAB User by performing the following steps, via the Contact Status in AB;

- If the requesting CAB User not in the CAB User's AB, the CAB Server:
  - o SHALL create a new Contact Entry and set the <temporary> element under the <entry-status> element of the <contact-status> with a value 'incoming subscription invite';
  - o SHALL set the 'invite-message' attribute associated with the <temporary> element, if custom invitation message is present in [SIP:MESSAGE](#), and;
  - o MAY populate the <contact> element with available PCC information of requesting CAB User.
- If the requesting CAB User is already in the CAB User's AB
  - o the CAB Server SHALL create a new Contact Entry and set the <temporary> element under the <entry-status> element of the <contact-status> with a value 'incoming subscription invite' and set the 'contactIdref' attribute to match the 'id' attribute value of an existing <contact> element.
  - o SHALL set the 'invite-message' attribute associated with the <temporary> element, if custom invitation message is present in [SIP:MESSAGE](#), and;
  - o MAY populate the <contact> element with available PCC information of requesting CAB User.

### 5.10.2.2 Using XDCP Forward

Upon getting notified of the temporary document (corresponding to the invitation request for subscription invitation) from PCC Application Usage, the CAB Server SHALL update the AB Document of the recipient with the following clarifications;

- If the preference set by the CAB User in the XDM Preferences Document based on the section 5.8.7.2 "Forward-prefs Element" of [OMA XDM Core] is "accept" or "confirm", and the received contact (i.e. requesting CAB User) is not the CAB User's AB, it SHALL store the received contact information as follows:
  - o SHALL create a new Contact Entry with the PCC information of the requesting CAB User and set the <temporary> element under the <entry-status> element of the <contact-status> with a value 'incoming subscription invite';
  - o SHALL set the 'invite-message' attribute associated as a child element of the <temporary> element with the value of <note> from the received XDCP request.
- If the preference set by the CAB User in the XDM Preferences Document based on the section 5.8.7.2 "Forward-prefs Element" of [OMA XDM Core] is "accept" or "confirm", and the received contact (i.e. requesting CAB User) is already in the CAB User's AB, it SHALL store the received contact information as follows:
  - o SHALL create a new Contact Entry with the PCC information of the requesting CAB User and set the <temporary> element under the <entry-status> element of the <contact-status> with a value 'incoming subscription invite' and set the 'contactIdref' attribute to match the 'id' attribute value of an existing <contact> element;
  - o SHALL set the 'invite-message' attribute associated as a child element of the <temporary> element with the value of <note> from the received XDCP request.

### 5.10.3 MIME Type for Cross-domain Subscription Invitations

The cross-domain subscription invitation requests SHALL conform to the structure and semantics described in the following sub-clauses.

### 5.10.3.1 Structure

The root element of the subscription invitation request SHALL be <subscription-invite>.

The <subscription-invite> element:

- SHALL include a <pcc-uri> element, indicating a valid XCAP URI of the originator's PCC Document.
- MAY include <invite-message> element, indicating a custom (text) invitation message to be delivered to the recipient of the subscription invitation request.
- Any other elements from any other namespaces for the purpose of extensibility.

### 5.10.3.2 Data Semantics

The <pcc-uri> element value is of type "anyURI". It SHALL be a valid XCAP URI.

The <invitation-message> element value is of type "String"

### 5.10.3.3 XML Schema

The XML schema definition is described in [XSD 1.1 cab subs invite].

### 5.10.3.4 Default Namespace

The default namespace is "urn:oma:xml:cab:subs-invite",

### 5.10.3.5 MIME Type

The MIME type used for cross-domain subscription invitation requests is 'application/vnd.oma.cab-sub-invite+xml'

## 6. Procedures at CAB Client

### 6.1 Address Book Management and Synchronization

To enable a flexible architecture by allowing a CAB Client to be either ‘XDM-only’ based or ‘XDM and DS’ based, the CAB Client SHALL support address book management and synchronization using either:

- OMA DS as detailed in section [6.1.1](#); or
- OMA XDM as detailed in section [6.1.2](#).

To further clarify, the CAB Client SHALL conform to either section [6.1.1](#) or [6.1.2](#) for synchronization and management of AB.

#### 6.1.1 Address Book Management and Synchronization using OMA DS

The CAB Client SHALL use OMA Data synchronization protocol [OMA DS Pro] as specified in section 5.3.2.8 “*Interface CAB-1: CAB Server*” in [CAB 1.1 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 as defined in [RFC2425] and [RFC2426] respectively 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. (a mapping table is specified in sub-clause [5.4.6](#) “*Format Adaptation*”).

CAB Client SHALL be able to receive “Server Alerted Sync” 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 1.1 XDMS].

##### 6.1.1.1 Retrieval and Restore of previously deleted Contact Entries

To support management of deleted Contact Entries, a CAB Client indicates to the CAB Server by using a combination of OMA DS “Delete”, “Get”, and “Add” commands.

- 1) For delete Contact Entry:
  - a) CAB Client SHALL use OMA DS “Delete” command, as specified in [OMA DS] sub-clause 6.5.5 “*Delete*”, to delete a Contact Entry.
  - b) CAB Client SHALL use OMA DS “Delete” command, with the <temporary> element equal to the value “deleted” of the contact, as specified in [OMA DS] sub-clause 6.5.5 “*Delete*”, to delete a Contact Entry with the <temporary> element equal to the value “deleted”.
- 2) For restore of a Contact Entry by the CAB Client:
  - a) CAB Client SHOULD perform an OMA DS “Get” command with the <temporary> element equal to the value “deleted” to retrieve deleted Contact Entry.
  - b) CAB Client SHALL perform an OMA DS “Add” command of the deleted Contact Entry to local address book.
  - c) CAB Client SHALL initiate AB synchronization to synchronize the data modification in the address book with the CAB Server.
- 3) For restore of a Contact Entry by CAB Server:
  - a) CAB Client SHOULD perform an OMA DS “Get” command with the <temporary> element equal to the value “deleted” to retrieve deleted Contact Entry.
  - b) CAB Client SHALL perform an OMA DS “Add” command with the <temporary> element equal to the value “deleted” of the contact, and the ‘accept’ attribute set to “yes”.

#### 6.1.2 Address Book Management and Synchronization using OMA XDM

The CAB Client SHALL format the requests for CAB User’s AB document management as described in the [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-address-book” application usage for the AB, as defined in [CAB 1.1 XDMS].

There are two mechanisms (i.e. SIP and XDCP/Push) through which a CAB Client can subscribe to document changes in AB XML document stored in AB App Usage [CAB 1.1 XDMS] AB XDMS, for AB synchronization.

The CAB Client SHALL generate the subscription requests to CAB User’s own AB Document changes, as described in [XDM Core] sub-clause 6.1.2 “*Subscribing to changes in the 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: `aud=<“org.openmobilealliance.cab-address-book”>`;

If the XDCP/Push method is used, the AUID in the Request-URI as described in [XDM Core] sub-clause 6.1.1.3 “*XDM Operations using XDCP*” SHALL be set to the “org.openmobilealliance.cab-address-book”.

### 6.1.2.1 Retrieval and Restore of previously deleted Contact Entries

To support management of deleted Contact Entries, the CAB Client uses a combination of XDM History and XDM Restore operations.

For retrieval of previously deleted Contact Entries, the CAB Client SHALL support retrieving the Modification History Information for Address Book as described in sub-clause 6.18.1 “*History Information of Address Book*”.

For restore of previously deleted Contact Entries, the CAB Client SHALL make an XDCP ‘restore’ request as described in the [XDM Core] sub-clause 6.1.1.3.7 “*XDM Restore*”.

## 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 [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 1.1 XDMS].

When undoing a deleted PCC Document, the CAB Client SHALL make an XDCP Request containing an XDCP Document as described in the [XDM Core] sub-clause 5.4.2 “*XDCP Document*” with the following clarifications:

- It SHALL include a <restore> element as child element of the <request> element. The <restore> element;
  - a) SHALL have a “type” attribute with the value set to “undo” to indicate the request is intended to perform undo operation to rollback the version right before the current version of PCC Document;
  - b) SHALL include the <current-etag> element containing the value of the E-Tag of the PCC Document the CAB Client currently possesses; and
  - c) SHALL NOT include <back-to-etag> element containing the E-Tag value of PCC Document which is requested to restore to.

## 6.3 CAB User Preferences Management

The CAB Client SHALL format the requests for CAB User Preferences document management as described in the [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 1.1 XDMS].

## 6.4 Import Non-CAB Address Book

The CAB Client SHALL use the <import-non-cab> element in the CAB Feature Handler Document [CAB 1.1 XDMS] to store the non-CAB address import request data, and SHALL use the procedures as described in the [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>, <scheduled-interval> and <credential> elements.
- 3) MAY populate <expiration-time> element.

In case the CAB client uses a token as credentials for a periodic import request it SHOULD ensure the token is valid at least until the expiration time of the periodic import request.

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 section 6.1 “Procedures at the XDMC and the XDM Agent” of [XDM Core] to obtain the CAB Feature Handler Document [CAB 1.1 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 6.1 “Address Book Management and Synchronization”.

## 6.5 Import profile information from 3<sup>rd</sup> Party Systems

The CAB Client SHALL use the <import-profile> element in the CAB Feature Handler Document [CAB 1.1 XDMS] to store the data for the request to import of profile information from 3<sup>rd</sup> party systems, and SHALL use the procedures as described in the [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>, <scheduled-interval> and <credential> elements.
- 3) MAY populate <expiration-time> and <filter-set> elements.

In case the CAB client uses a token as credentials for a periodic import request it SHOULD ensure the token is valid at least until the expiration time of the periodic import request.

The CAB Client SHALL use the <response> element of the <import-profile> 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 section 6.1 “Procedures at the XDMC and the XDM Agent” of [XDM Core] to obtain the CAB Feature Handler Document [CAB 1.1 XDMS].
- 2) SHALL use <import-profile> “id” attribute that is associated with the same request.
- 3) MAY delete the whole import request of <import-profile> element, when the CAB Client is notified about the result of the request and about the PCC updates with the imported profile information or when the CAB User requests that the import of External Profile shall be cancelled.

The CAB Client is notified of the changes in the PCC (via the <pcc-update> element of the PCC Document if the CAB User Preference <pcc-import-update> is set to “false”, or directly in the PCC Document if the CAB User Preference <pcc-import-update> is set to “true”) either by SIP NOTIFY or Push OTA Message depending on whether the subscription was a SIP-based subscription or an XDCP-based subscription as described in 6.13.1 “Subscriptions to CAB User’s own PCC Document changes”.

In case the CAB Client performs manual approval of changes to a PCC Document, it SHALL subscribe to changes to PCC Document in the PCC Application Usage and check for PCC Document including <pcc-update> element including a <update-ref> element as described in [CAB 1.1 XDMS] with the following clarifications;

- a) SHALL, when the CAB Client detects a <update-ref> element included in <pcc-update> element, retrieve the PCC Update Document referenced in the <update-ref> and present the proposed changes to the PCC Document included in the XCAP Diff document in the body of the response to the CAB User for manual approval;
- b) SHALL update the PCC Document with the changes that were approved by the CAB User by generating a Differential Write XDCP request as described in [OMA XDM Core] section “Differential Write” with the XCAP Diff document received where all not approved document changes to the PCC Document are removed.  
Note: For the acknowledgement that the manual update approval procedure has been performed, the Differential Write XDCP request shall include the <remove> element that has the node selector of <update-ref> element to indicate associated PCC Update Document.

## 6.6 Export PCC information to 3<sup>rd</sup> Party Systems

The CAB Client SHALL use the <export-profile> element in the CAB Feature Handler Document [CAB 1.1 XDMS] to store the data for the request to export of PCC information to 3<sup>rd</sup> party systems, and SHALL use the procedures as described in the [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>, <scheduled-interval> and <credential> elements.
- 3) MAY populate <expiration-time> and <filter-set> elements.

In case the CAB client uses a token as credentials for a periodic export request it SHOULD ensure the token is valid at least until the expiration time of the periodic export request.

The CAB Client SHALL use the <response> element of the <export-profile> 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 section 6.1 “*Procedures at the XDMC and the XDM Agent*” of [XDM Core] to obtain the CAB Feature Handler Document [CAB 1.1 XDMS].
- 2) SHALL use <export-profile> “id” attribute that is associated with the same request.
- 3) MAY delete the whole export request of <export-profile> element, once CAB Client is notified about the result of the request.

## 6.7 Subscription to profile information in 3<sup>rd</sup> Party Systems

The CAB Client SHALL use the <subscribe-profile> element in the CAB Feature Handler Document [CAB 1.1 XDMS] to store the data for the request to subscribe to profile information in 3<sup>rd</sup> party systems, and SHALL use the procedures as described in the [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.
- 3) MAY populate <expiration-time> and <filter-set> elements.

In case the CAB client uses a token as credentials it SHOULD ensure the token is valid at least until the expiration time of the subscription. If no expiration time has been defined for the subscription the CAB client SHOULD issue a new subscription request with a new token before the expiration of the previously provided token.

The CAB Client SHALL use the <response> element of the <subscribe-profile> 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 section 6.1 “*Procedures at the XDMC and the XDM Agent*” of [XDM Core] to obtain the CAB Feature Handler Document [CAB 1.1 XDMS].
- 2) SHALL use <subscribe-profile> “id” attribute that is associated with the same request.
- 3) MAY delete the whole subscribe request of <subscribe-profile> element, when the CAB Client is notified about the result of the request and about the PCC updates with the received profile information or when the CAB User requests that the subscription to External Profile shall be cancelled.

The CAB Client is notified of the changes in the PCC (via the <pcc-update> element of the PCC Document if the CAB User Preference <pcc-subscribe-update> is set to “false”, or directly in the PCC Document if the CAB User Preference <pcc-subscribe-update> is set to “true”) either by SIP NOTIFY or Push OTA Message depending on whether the subscription was a SIP-based subscription or an XDCP-based subscription as described in [6.13.1](#) “*Subscriptions to CAB User’s own PCC Document changes*”.

In case the CAB Client performs manual approval of changes to a PCC Document, it SHALL subscribe to changes to PCC Document in the PCC Application Usage and check for PCC Document including <pcc-update> element including a <update-ref> element as described in [CAB 1.1 XDMS] with the following clarifications;

- a) SHALL, when the CAB Client detects a <update-ref> element included in <pcc-update> element, retrieve the PCC Update Document referenced in the <update-ref> and present the proposed changes to the PCC Document included in the XCAP Diff document in the body of the response to the CAB User for manual approval;
- b) SHALL update the PCC Document with the changes that were approved by the CAB User by generating a Differential Write XDCP request as described in [OMA XDM Core] section “*Differential Write*” with the XCAP Diff document received where all not approved document changes to the PCC Document are removed.  
Note: For the acknowledgement that the manual update approval procedure has been performed, the Differential Write XDCP request shall include the <remove> element that has the node selector of <update-ref> element to indicate associated PCC Update Document.

## 6.8 Contact Share

The CAB Client SHALL use the <contact-share> element in the CAB Feature Handler Document [CAB 1.1 XDMS], to store the Contact Share request data and SHALL use the procedures as described in the [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.
- MAY populate the <format> element.

The CAB Client SHALL use the <response> element of the corresponding <contact-share> element of the CAB Feature Handler Document [CAB 1.1 XDMS] to indicate the status of the Contact Share request to the CAB User, with the following clarifications:

- 1) SHALL use either subscription operation or polling mechanism using document management operations following the rules and procedures specified in section 6.1 “*Procedures at the XDMC and the XDM Agent*” of [XDM Core] to obtain the <response> element of the CAB Feature Handler Document [CAB 1.1 XDMS].
- 2) SHALL use <contact-share> “id” attribute that is associated with the corresponding Contact Share request.
- 3) When the <delivery-report-request> is set to “false”, or when <delivery-report-status> element indicates the result as ‘Successful’ or ‘Failure’ for all recipients, it MAY delete the <contact-share> element of the request, when the <code> element of the <response> element has the value “Success” or “Failure” respectively, 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.

## 6.9 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 [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 1.1 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 1.1 XDMS];
- It SHALL populate the ‘id’ attribute 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 [XDM Core].



## 6.10 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 6.1.1 “*Document Management*” and the sub-clause 5.6 “*Access Permissions Document*” in [XDM\_Core] with the clarifications provided in this sub-clause.

### 6.10.1 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”

### 6.10.2 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”

## 6.11 UPP Directory Document Management

The procedure to manage the UPP Directory Document [XDM UPP] at a CAB Client SHALL conform to [XDM Core] 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]”

## 6.12 XDM Preferences Management

The CAB Client SHALL use the XDM Preferences Document to manage the forwarding preferences of AB and PCC Document for Contact Share operations and the history preferences of PCC Request History Information Document for reactive authorization as described in sub-clause [5.8.3](#) “*Incoming Contact 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*” of [XDM\_Core] with the clarifications given in this sub-clause.

### 6.12.1 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”

### 6.12.2 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”

## 6.13 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.

### 6.13.1 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 [XDM Core] sub-clause 6.1.2 “*Subscribing to changes in the 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 [XDM Core] sub-clause 6.1.1.3 “*XDM Operations using XDCP*” SHALL be set to the “org.openmobilealliance.cab-pcc”.

### 6.13.2 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 [XDM Core] sub-clause 6.1.2 “*Subscribing to changes in the 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 [XDM Core] sub-clause 6.1.1.3 “*XDM Operations using XDCP*” SHALL be set to the “org.openmobilealliance.cab-user-prefs”.

### 6.13.3 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 [XDM Core] sub-clause 6.1.2 “*Subscribing to changes in the 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 [XDM Core] sub-clause 6.1.1.3 “*XDM Operations using XDCP*” SHALL be set to the “org.openmobilealliance.cab-feature-handler”.

### 6.13.4 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 [XDM Core] sub-clause 6.1.2 “*Subscribing to changes in the 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 [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 [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 [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.

## 6.14 Contact Search

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

1. Construct a Search Request containing a Search Document according to the rules and procedures described in [XDM Core] sub-clause 5.4.1 “Search Document” and sub-clause 6.1.3 “*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 AB 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 by using a HTTP POST request containing a Search Document to the Aggregation Proxy according to the rules and procedures described in [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”.

## 6.15 Contact Status Management

The CAB Client SHALL use the <contact-status> element from the AB 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.

## 6.16 Authentication

### 6.16.1 Authentication for AB Synchronization

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

### 6.16.2 Authentication for XML Document Management

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

## 6.17 Subscription Invitation

The CAB Client SHALL use the <subscription-invite> element in the CAB Feature Handler Document [CAB 1.1 XDMS], to store the contact subscription invitation request data and SHALL use the procedures as described in the [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
- MAY populate the <contact-view> element and its child elements
- MAY populate <subscription-authorization> element
- MAY populate <invitation-message> element.

The CAB Client SHALL use the <response> element of the corresponding <subscription-invite> element of the CAB Feature Handler Document [CAB 1.1 XDMS] to indicate the status of the contact subscription invitation request to the CAB User, with the following clarifications:

- 1) SHALL use either subscription operation or polling mechanism using document management operations following the rules and procedures specified in section 6.1 “*Procedures at the XDMC and the XDM Agent*” of [XDM Core] to obtain the <response> element of the CAB Feature Handler Document [CAB 1.1 XDMS].
- 2) SHALL use <subscription-invite> “id” attribute that is associated with the corresponding contact subscription invitation request.

## 6.18 Retrieval of History Information

The CAB Client SHALL support the retrieval of History Information Documents as described in section 6.1.4 “*Retrieval of History Information*” in [XDM Core] with the following clarifications:

- 1) For ‘XDM and DS’ based CAB Client: Retrieval of History Information Documents SHALL be supported for CAB PCC Application Usage, as defined in [CAB 1.1 XDMS].
- 2) For ‘XDM-only’ based CAB Client: Retrieval of History Information Documents SHALL be supported for both CAB AB and PCC Application Usages, as defined in [CAB 1.1 XDMS].

### 6.18.1 History Information of Address Book

The Request History Information for Address Book SHALL be retrieved using the User Directory Document selector ‘/oma\_requests/history’ with AB AUID. The HTTP Request-URI SHALL be set with the following value:

- “http://[XCAP\_Root\_URI]/org.openmobilealliance.cab-address-book/users/[XUI]/oma\_requests/history”

The Modification History Information for Address Book SHALL be retrieved using the User Directory Document selector ‘/oma\_hist/address-book’ with AB AUID. The HTTP Request-URI SHALL be set with the following value:

- “http://[XCAP\_Root\_URI]/org.openmobilealliance.cab-address-book/users/[XUI]/oma\_hist/address-book”

### 6.18.2 History Information of PCC

The Request History Information for PCC SHALL be retrieved using the User Directory Document selector ‘/oma\_requests/history’ with PCC AUID. The HTTP Request-URI SHALL be set with the following value:

- “http://[XCAP\_Root\_URI]/ org.openmobilealliance.cab-pcc /users/[XUI]/oma\_requests/history”

The Modification History Information for PCC SHALL be retrieved using the User Directory Document selector ‘/oma\_hist/pcc’ with PCC AUID. The HTTP Request-URI SHALL be set with the following value:

- “http://[XCAP\_Root\_URI]/ org.openmobilealliance.cab-pcc /users/[XUI]/oma\_hist/pcc ”

## 6.19 Contact Suggestion

The CAB Client SHALL use one of the following mechanisms to initiate Contact suggestions:

- 1) Enabling <allow-suggested-contact-info> preference in the CAB User Preferences App Usage [CAB 1.1 XDMS];
- 2) Making a Contact suggestion request via CAB Feature Handler App Usage [CAB 1.1 XDMS]

If initiated by the CAB User Preferences App Usage, the CAB Client SHALL modify the <allow-suggested-contact-info> element in the CAB User Preferences App Usage, as described in sub-clause 6.3 “*CAB User Preferences Management*”.

If initiated by the CAB Feature Handler App Usage, the CAB Client SHALL use the <contact-suggest> element in the CAB Feature Handler Document [CAB 1.1 XDMS], to store the contact suggestion request data and SHALL use the procedures as described in the [XDM Core] sub-clause 6.1.1 “*Document Management*” with the following clarifications:

- SHALL use the AUID “org.openmobilealliance.cab-feature-handler”.
- MAY populate <non-CAB-source>, <scheduled-interval> and <credential> elements.
- MAY populate <expiration-time>, <filter-set> elements and <criteria> element.

In case the CAB Client uses a token as credentials for a periodic contact suggestion request it SHOULD ensure the token is valid at least until the expiration time of the periodic contact suggestion request.

The CAB Client SHALL use the <response> element of the corresponding <contact-suggest> element of the CAB Feature Handler Document [CAB 1.1 XDMS] to indicate the status of the contact suggestion request to the CAB User, with the following clarifications:

- 1) SHALL use either subscription operation or polling mechanism using document management operations following the rules and procedures specified in section 6.1 “*Procedures at the XDMC and the XDM Agent*” of [XDM Core] to obtain the <response> element of the CAB Feature Handler Document [CAB 1.1 XDMS].
- 2) SHALL use <contact-suggest> “id” attribute that is associated with the corresponding contact suggestion request.
  - The CAB Client receives the Contact suggestions via AB (i.e. Contact Entry with <temporary> element value set to “suggested”), through the AB synchronization performed as specified in sub-clause 6.1 “*Address Book Management and Synchronization*”.

## 7. CAB XDMS

The CAB XDMS(s) SHALL support the XDMS procedures described in [XDM Core], sub-clause 6.2 “*Procedures at the XDM Server*”, and the Application Usages described in [CAB 1.1 XDMS].

## 8. CAB Management Object

The CAB Management Object (MO) for configuration and provisioning of CAB Client is described in [CAB 1.0 MO].

Note: No new configuration parameters have been introduced in CAB 1.1 Enabler, therefore CAB 1.0 MO is used as-is

## Appendix A. Change History (Informative)

### A.1 Approved Version History

Reference	Date	Description
OMA-TS-CAB-V1_1-20180621-A	21 Jun 2018	Status changed to Approved by COM Doc Ref # OMA-COM-2018-0003-INP_CAB_V1_1_ERP_for_final_Approval

## 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-AB-C-001-M	Support for AB Management and Synchronization	Section 6.1	CAB-AB-C-002-O, CAB-AB-C-003-O, CAB-AB-C-006-O, OR CAB-AB-C-007-O, CAB-AB-C-008-O
CAB-AB-C-002-O	Support for AB Management and Synchronization with CAB Server using CAB-1 interface (OMA DS)	Section 6.1.1	CAB-AB-C-004-O OR CAB-AB-C-005-O
CAB-AB-C-003-O	Support for receiving ‘Server-Alerted Sync’	Section 6.1.1	
CAB-AB-C-004-O	Support for AB synchronization with CAB Server using CAB Format	Section 6.1.1	
CAB-AB-C-005-O	Support for AB synchronization with CAB Server using Legacy Format(s)	Section 6.1.1	
CAB-AB-C-006-O	Support for Retrieval and Restore of previously deleted contacts (OMA DS)	Section 6.1.1.1	
CAB-AB-C-007-O	Support for AB Management and Synchronization with AB XDMS (OMA XDM)	Section 6.1.2	
CAB-AB-C-008-O	Support for Retrieval and Restore of previously deleted contacts (OMA XDM)	Section 6.1.2.1	
CAB-PCC-C-001-M	Support for managing CAB PCC Document in PCC Application Usage	Section 6.2	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-UP-C-001-M	Support for managing CAB User Preferences in CAB User Preferences Application Usage	Section 6.3	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-IMP-NonCAB-C-001-M	Support for managing import non-CAB	Section 6.4	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M,



Item	Function	Reference	Requirement
	requests in CAB Feature Handler Application Usage		XDM_Core-XOP-C-003-M
CAB-IMP-NonCAB-C-002-M	Support for managing import non-CAB responses in CAB Feature Handler Application Usage	Section 6.4	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-IMP-Prof-C-001-M	Support for managing import profile information requests in CAB Feature Handler Application Usage	Section 6.5	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-IMP-Prof-C-002-M	Support for managing import profile information responses in CAB Feature Handler Application Usage	Section 6.5	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-EXP-PCC-C-001-M	Support for managing export PCC information requests in CAB Feature Handler Application Usage	Section 6.6	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-EXP-PCC-C-002-M	Support for managing export PCC information responses in CAB Feature Handler Application Usage	Section 6.6	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-SUBS-Prof-C-001-M	Support for managing subscription to profile information requests in CAB Feature Handler Application Usage	Section 6.7	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-SUBS-Prof-C-002-M	Support for managing subscription to profile information responses in CAB Feature Handler Application Usage	Section 6.7	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-SHR-C-001-M	Support for managing Contact Share requests in CAB Feature Handler Application Usage	Section 6.8	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-SHR-C-002-M	Support for managing Contact Share responses and delivery report in CAB Feature Handler Application Usage	Section 6.8	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-CS-C-001-M	Support for managing Contact Subscription list in CAB User Preferences Application Usage	Section 6.9	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-AP-C-001-M	Support for managing	Section 6.10.1	XDM_Core-XOP-C-001-M,

Item	Function	Reference	Requirement
	AB Access Permissions		XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-AP-C-002-M	Support for managing PCC Access Permissions	Section 6.10.2	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-UPP-C-001-M	Support for managing UPP Directory Document	Section 6.11	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-XDM-Pref-C-001-M	Support for managing XDM Preferences for AB	Section 6.12.1	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-XDM-Pref-C-002-M	Support for managing XDM Preferences for PCC	Section 6.12.2	
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 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-SRCH-C-001-M	Support for constructing and sending the contact search request to CAB AB Application Usage	Section 6.14	XDM_Core-SRC-C-001-O XDM_Core-SRC-C-002-O
CAB-SRCH-C-002-M	Support for constructing and sending the contact search request to CAB	Section 6.14	XDM_Core-SRC-C-001-O XDM_Core-SRC-C-002-O

Item	Function	Reference	Requirement
	PCC Application Usage		
CAB-SRCH-C003-M	Support for constructing and sending the contact search request to the External Directories	Section 6.14	XDM_Core-SRC-C-002-O XDM_Core-SRC-C-003-O
CAB-COS-C-001-M	Support for Contact Status information	Section 6.15	CAB-AB-C-001-M
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-SUBS-INV-C-001-M	Support for managing subscription invitation requests in CAB Feature Handler Application Usage	Section 6.17	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-M
CAB-SUBS-INV-C-002-M	Support for managing subscription invitation responses in CAB Feature Handler Application Usage	Section 6.17	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-HIST-C-001-O	Support for retrieval of History Information for AB	Section 6.18.1	XDM_Core-RHI-C-001-O XDM_Core-MHI-C-001-O
CAB-HIST-C-002-M	Support for retrieval of History Information for PCC	Section 6.18.2	XDM_Core-RHI-C-001-O XDM_Core-MHI-C-001-O
CAB-SUGS-C-001-M	Support for receiving contact suggestions	Section 6.19	CAB-COS-C-001-M
CAB-SUGS-C-002-M	Support for initiating contact suggestions	Section 6.19	CAB-SUGS-C-003-O OR CAB-SUGS-C-004-O
CAB-SUGS-C-003-O	Support for initiating contact suggestion via CAB User Preferences	Section 6.19	CAB-UP-C-001-M
CAB-SUGS-C-004-O	Support for initiating contact suggestion via CAB Feature Handler	Section 6.19	XDM_Core-XOP-C-001-M, XDM_Core-XOP-C-002-M, XDM_Core-XOP-C-003-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

Item	Function	Reference	Requirement
			CAB-SYNC-S-007-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 5.4.6	
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 Application Usage	Section 5.1 Section 5.5	CAB-XDMA-S-001-M
CAB-SYNC-S-007-M	Support for Retrieval and Restore of previously deleted contacts	Section 5.1.1	
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 Application Usage	Section 5.2, Section 5.5	CAB-XDMA-S-001-M
CAB-CSUB-S-002-O	Support for handling the contact subscription request with <filter-set>	Section 5.2	
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 Document using XDM document management operation with resulting	Section 5.2, Section 5.5	CAB-XDMA-S-001-M

Item	Function	Reference	Requirement
	data from Contact Subscriptions, and updating of contact subscription status		
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 Application 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 Application 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 Application 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 [CPM CONV FCT TS]	Section 5.3	
CAB-CSF-S-006-M	Support for sending the message defined in [CPM CONV FCT TS] towards the CPM Interworking Selection Function [CPM IWF TS]	Section 5.3	
CAB-CSF-S-007-M	Support for initiating XDCCP Request for AB or PCC forwarding 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 Application 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 Application 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	Section 5.4.1 Section 5.5	CAB-XDMA-S-001-M

Item	Function	Reference	Requirement
	Application Usage and updating the contact status		
CAB-IWF-S-003-M	Supporting for storing the response (import non-CAB) in CAB Feature Handler Application Usage	Section 5.4.1 Section 5.5	CAB-XDMA-S-001-M
CAB-IWF-S-004-M	Support for obtaining the request to import profile information from 3 <sup>rd</sup> party (i.e. using document management or subscription to changes) from CAB Feature Handler Application Usage	Section 5.4.1, Section 5.5	CAB-XDMA-S-001-M
CAB-IWF-S-005-M	Supporting the storage of imported profile information in PCC Application Usage	Section 5.4.1 Section 5.5	CAB-XDMA-S-001-M CAB-IWF-S-008-M
CAB-IWF-S-006-M	Supporting for storing the response (import profile) in CAB Feature Handler Application Usage	Section 5.4.1 Section 5.5	CAB-XDMA-S-001-M
CAB-IWF-S-007-M	Support for obtaining the request to export PCC information to 3 <sup>rd</sup> party (i.e. using document management or subscription to changes) from CAB Feature Handler Application Usage	Section 5.4.1, Section 5.5	CAB-XDMA-S-001-M
CAB-IWF-S-008-M	Supporting for storing the response (export PCC) in CAB Feature Handler Application Usage	Section 5.4.1 Section 5.5	CAB-XDMA-S-001-M
CAB-IWF-S-009-M	Support for obtaining the request to subscribe to profile information from 3 <sup>rd</sup> party (i.e. using document management or subscription to changes) from CAB Feature Handler Application Usage	Section 5.4.1, Section 5.5	CAB-XDMA-S-001-M
CAB-IWF-S-010-M	Supporting the storage of changes to profile information in PCC Application Usage	Section 5.4.1 Section 5.5	CAB-XDMA-S-001-M CAB-IWF-S-008-M
CAB-IWF-S-011-M	Supporting for storing	Section 5.4.1	CAB-XDMA-S-001-M

Item	Function	Reference	Requirement
	the response (profile subscription) in CAB Feature Handler Application Usage	Section 5.5	
CAB-IWF-S-012-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-013-M	Support for CAB extension <dataSource> to the Search Document for External Directories Search	Section 5.4.2.1.5	
CAB-IWF-S-014-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-015-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-016-M	Support for the format adaptation between CAB format and Legacy format(s)	Section 5.4.3	
CAB-IWF-S-017-M	Support for obtaining the request to import profile from 3 <sup>rd</sup> Party System (i.e. using document management or subscription to changes) from CAB Feature Handler Application Usage	Section 5.4.2, Section 5.5	CAB-XDMA-S-001-M
CAB-IWF-S-018-M	Support for obtaining the request to export PCC to 3 <sup>rd</sup> Party System (i.e. using document management or subscription to changes) from CAB Feature Handler Application Usage	Section 5.4.3, Section 5.5	CAB-XDMA-S-001-M
CAB-IWF-S-019-M	Support for obtaining the request to subscribe to profile in 3 <sup>rd</sup> Party System (i.e. using	Section 5.4.4, Section 5.5	CAB-XDMA-S-001-M

Item	Function	Reference	Requirement
	document management or subscription to changes) from CAB Feature Handler Application Usage		
CAB-IWF-S-020-M	Supporting the storage in PCC Application Usage of profile information received via import or subscription to 3 <sup>rd</sup> Party system	Section 5.4.1 Section 5.5	CAB-XDMA-S-001-M
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 XDM_Core-MHI-A-001-O XDM_Core-RHI-A-001-O XDM_Core-PRF-A-001-O
CAB-COS-S-001-M	Support for managing the Contact Status information	Section 5.8	
CAB-COS-S-002-O	Support for CAB Capability publication as SIP Publish	Section 5.8.2 Section 5.5	CAB-XDMA-S-001-M
CAB-COS-S-003-O	Support for CAB Capability publication as Permanent Presence State	Section 5.8.2 Section 5.6	
CAB-COS-S-004-O	Support for subscription to Presence to obtain CAB Capability of contacts	Section 5.8.2 Section 5.7	
CAB-COS-S-005-M	Support for Incoming Contact Subscription requests	Section 5.8.3 Section 5.5	CAB-XDMA-S-001-M
CAB-COS-S-006-M	Support for generating and sending SIP Message with Contact Added data to a remote domain	Section 5.8.4.1	
CAB-COS-S-007-M	Support for receiving SIP Message with Contact Added data from a remote domain	Section 5.8.4.2	
CAB-COS-S-008-M	Support for populating Contact Added information in AB	Section 5.8.4.2 Section 5.5	CAB-XDMA-S-001-M
CAB-COS-S-009-M	Support for populating Common Connections	Section 5.8.5	CAB-XDMA-S-001-M



Item	Function	Reference	Requirement
	information in AB	Section 5.5	
CAB-COS-S-010-M	Support for checking the contact suggestion triggers	Section 5.8.6.1 Section 5.5	CAB-XDMA-S-001-M
CAB-COS-S-011-M	Support for delivery of contact suggestions	Section 5.8.6.2 Section 5.5	CAB-XDMA-S-001-M
CAB-COS-S-012-M	Support for populating service suggestions in AB	Section 5.8.7 Section 5.5	CAB-XDMA-S-001-M
CAB-COS-S-013-M	Support for populating public content in AB	Section 5.8.8 Section 5.5	CAB-XDMA-S-001-M
CAB-UPP-S-001-M	Support for managing UPP Directory Document	Section 5.9 Section 5.5	CAB-XDMA-S-001-M
CAB-SUBS-INV-001-M	Support for obtaining the subscription invitation requests from CAB Feature Handler Application Usage	Section 5.10.1 Section 5.5	CAB-XDMA-S-001-M
CAB-SUBS-INV-002-M	Support for sending and receiving cross-domain subscription requests and responses	Section 5.10.1 Section 5.10.2	CAB-SUBS-INV-003-O CAB-SUBS-INV-005-O OR CAB-SUBS-INV-004-O CAB-SUBS-INV-006-O
CAB-SUBS-INV-003-O	Support for generating, and sending SIP Message with subscription invite data to a remote domain	Section 5.10.1.1 Section 5.10.3 Section 5.5	CAB-XDMA-S-001-M
CAB-SUBS-INV-004-O	Support for sending an XDCP request with subscription invite data to a remote domain	Section 5.10.1.2 Section 5.5	CAB-XDMA-S-001-M
CAB-SUBS-INV-005-O	Support for receiving SIP Message with subscription invite data from a remote domain	Section 5.10.2.1 Section 5.10.3 Section 5.5	CAB-XDMA-S-001-M
CAB-SUBS-INV-006-O	Support for receiving an XDCP request with subscription invite data from a remote domain	Section 5.10.2.2 Section 5.5	CAB-XDMA-S-001-M

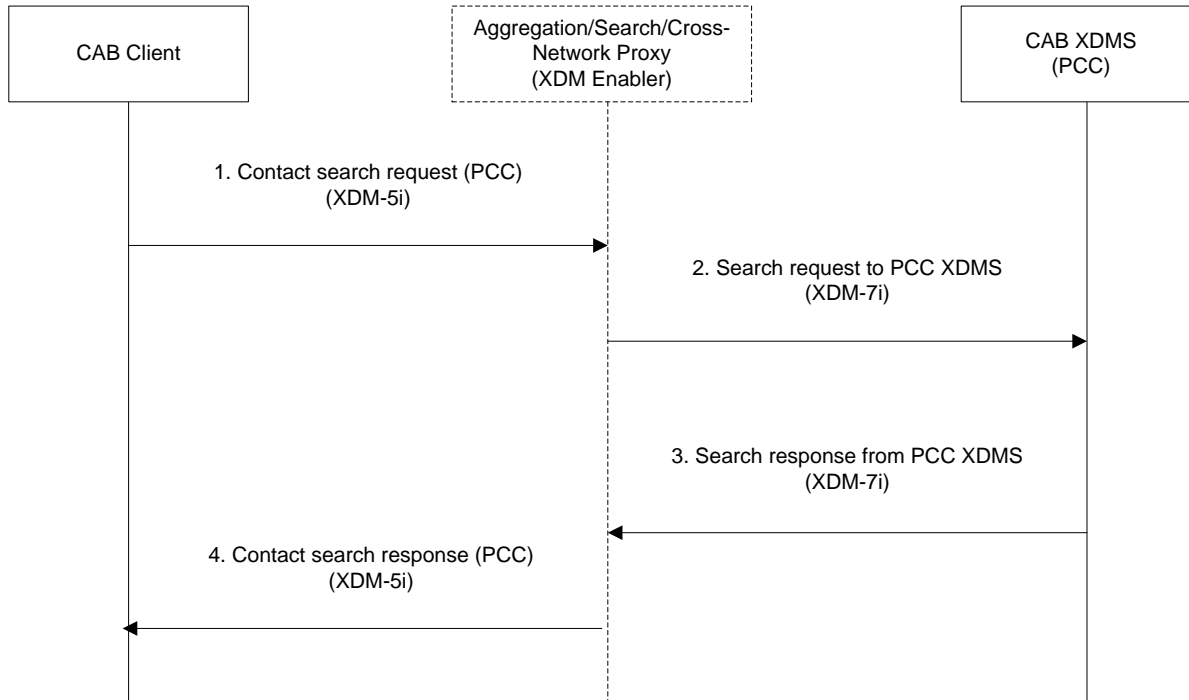
## Appendix C. Flows

(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



**Figure 2: Contact Search - PCC**

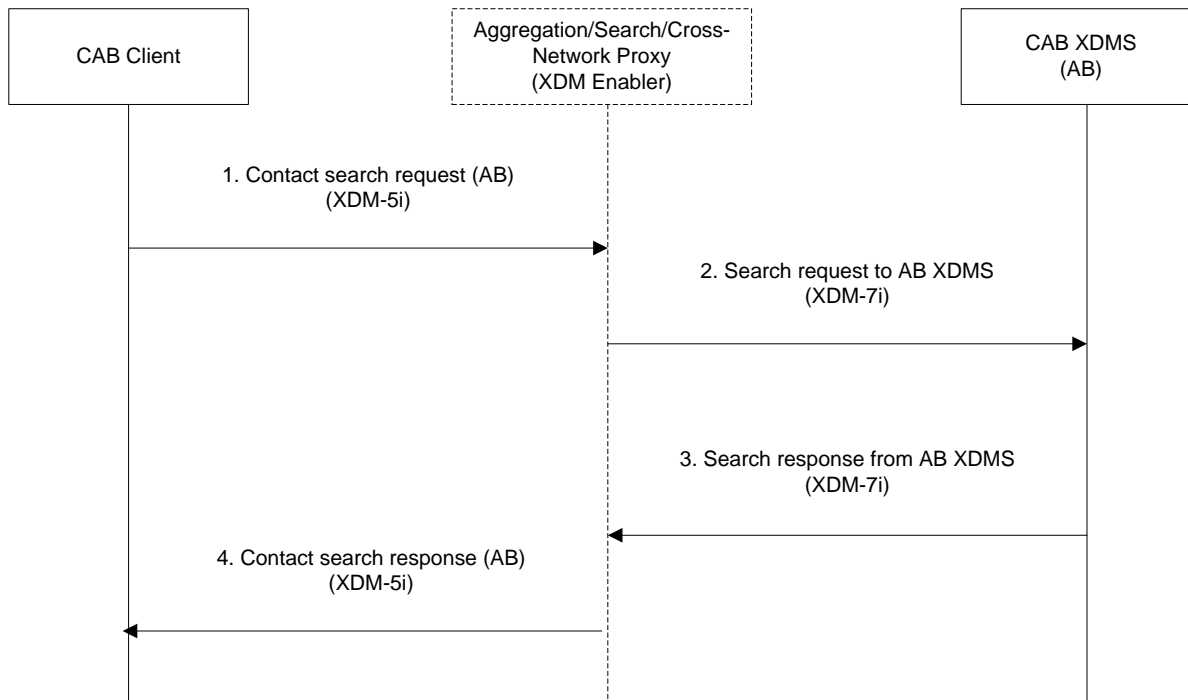
Step 1: The CAB Client makes a contact search request to CAB Users' PCC(s) via the XDM Enabler proxies (Aggregation Proxy, Search Proxy, Cross-Network Proxy) using Limited XQuery. The request is formulated based on the PCC XML schema hosted by the PCC Application Usage. This request is based on XDM-5i.

Step 2: Upon receiving the contact search request, the Search Proxy routes the request to the PCC Application Usage as indicated in the original request.

Step 3: The search response including the search results from the PCC Application Usage 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 Step 1. The response is based on XDM-5i.

## C.1.2 Contact Search - AB



**Figure 3: Contact Search - AB**

Step1: 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 Application Usage. This request is based on XDM-5i.

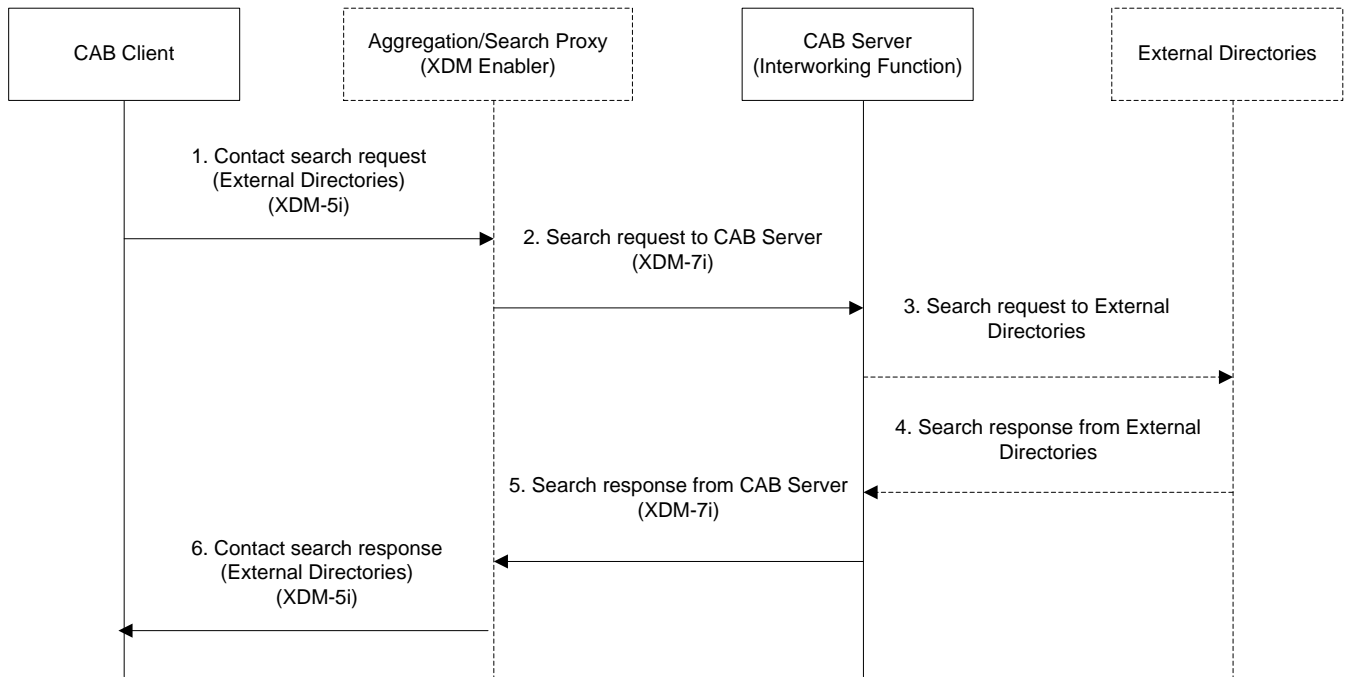
Step 2: Upon receiving the contact search request, the Search Proxy routes the request to the AB Application Usage as indicated in the original request.

Step 3: The search response including the search results from the AB Application Usage 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 Step1. 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 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's AB data.

### C.1.3 Contact Search – External Directories



**Figure 4 : Contact Search – External Directories**

Step1: 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 contact search 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.

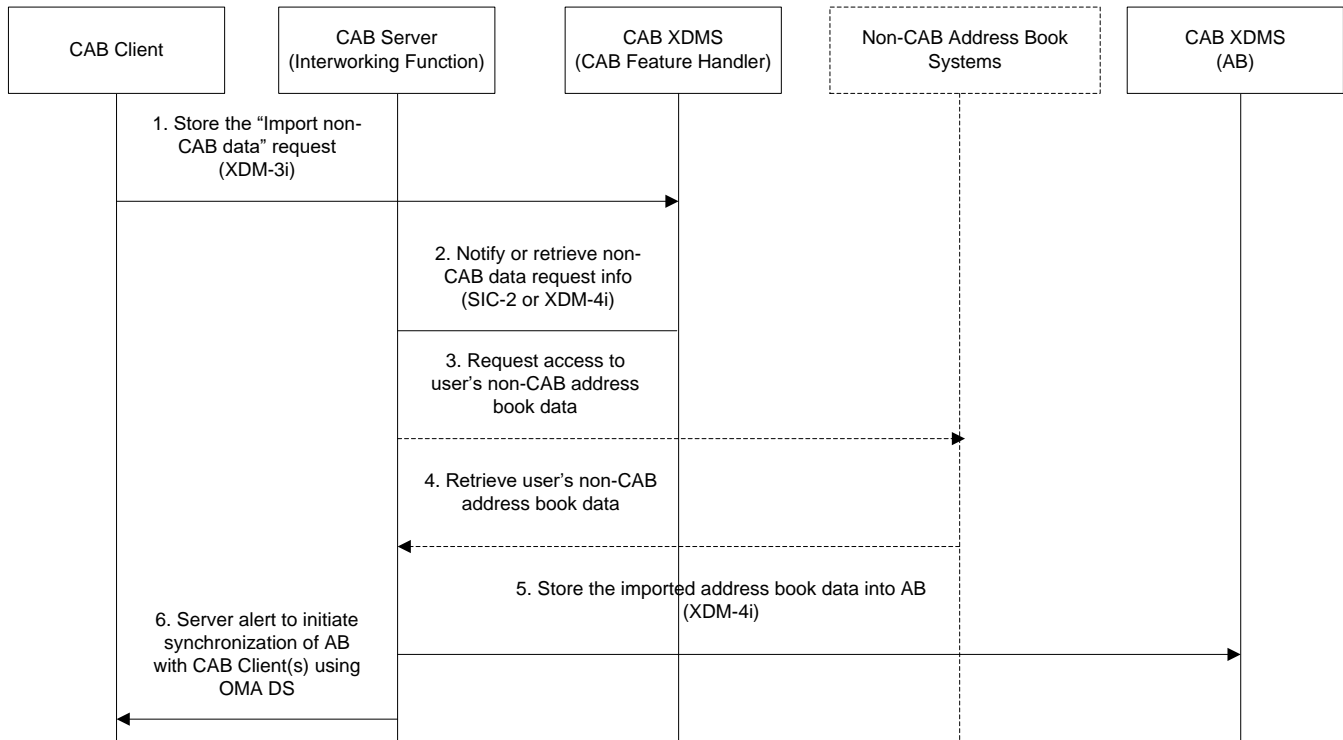
Step 3: The Interworking Function initiates the translated search request(s) to the External Directories over the appropriate interface(s). The translation and initiation of search requests over such external interfaces is out of scope of this specification.

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 of this specification.

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



**Figure 5 : 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 1.1 XDMS].

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

Step 2a: CAB Feature Handler Application Usage notifies a change to the CAB Feature Handler Document to the CAB Server assuming a prior subscription to the changes is in place by the CAB Server.

Step 2b: CAB Feature Handler Application Usage is polled by the CAB Server for changes in the CAB Feature Handler 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 (e.g. receipt of contact information including any Favourite Contacts) 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 1.1 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 the AB Synchronization Function of the CAB Server to the CAB Client to initiate synchronization as described in section 5.1 “AB Synchronization Function”.

### 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 [XDM Core] document management operations: create, retrieve, update, and delete.

Note: The AB Document data management operations are captured in the flows of sub-clause C.6.1 “CAB Client Address Book Modifications and Synchronization”. The flows in this section apply to all the CAB XML documents except the AB 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.2 “Sample XCAP Message Flow” of [XDM Core]. This example describes the message flows used by the CAB Client to manipulate a CAB XML document in CAB XDMS(s) after authentication.

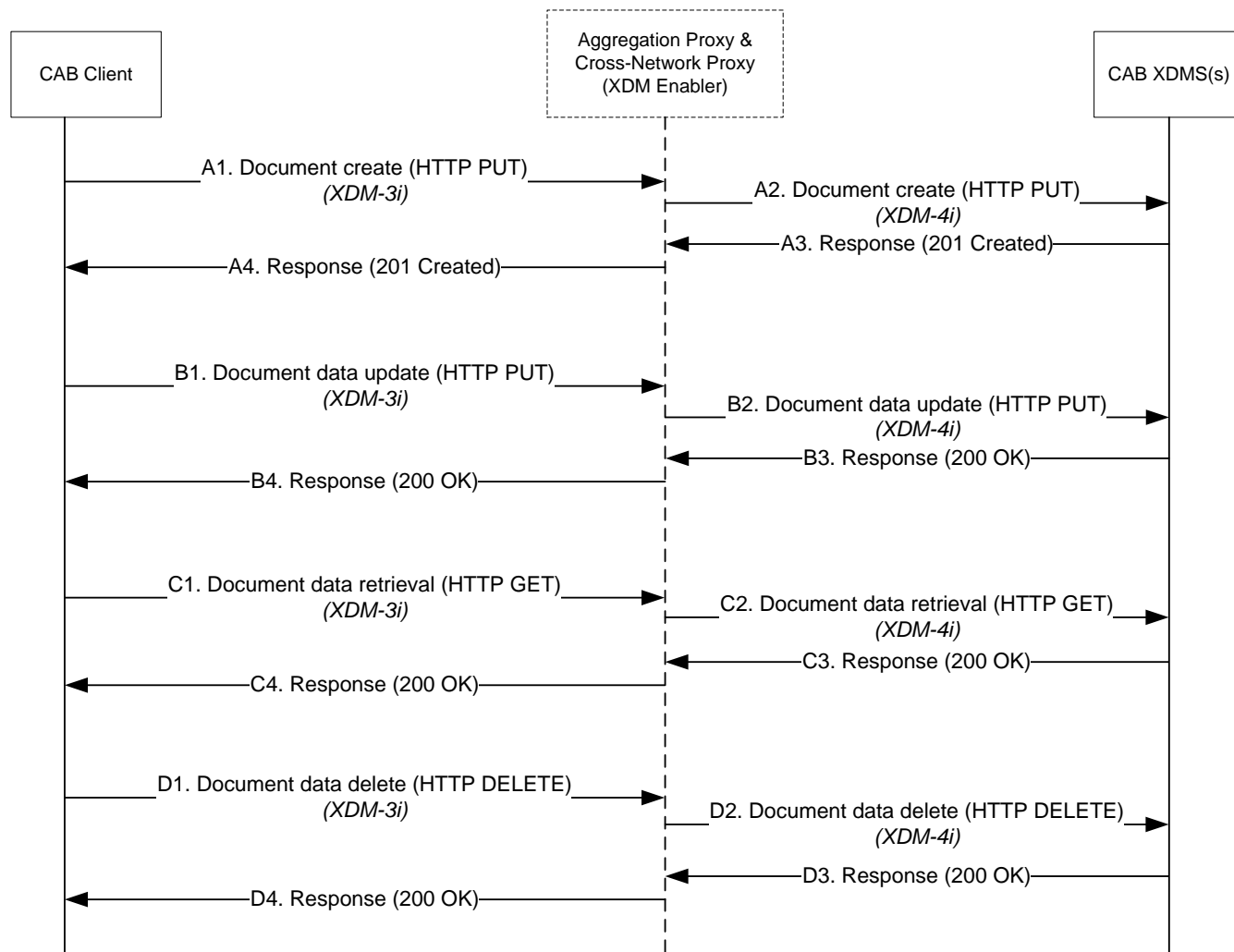


Figure 6: CAB Client manipulating a CAB XML document

NOTE 1 : All the operations are shown in one diagram for the convenience of the reader, but there is no implication that all of them have to be performed in sequence.

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

The following description of steps is based on the rules and procedures as defined in section 6.1 “*Procedures at the XDMC and the XDM Agent*” and section C.2 “*Sample XCAP Message Flow*” of [XDM Core] and use as example operations on CAB User Preferences XML Document [CAB 1.1 XDMS]

#### 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 XML 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 XML Document that is owned by user with XUI of “sip:joebloggs@example.com” in the example.com domain.

```
PUT /org.openmobilealliance.cab-user-prefs/users/sip:joebloggs@example.com/index HTTP/1.1
Host: xcap.example.com
...
Content-Type: application/vnd.oma.cab-user-prefs+xml; charset="utf-8"
Content-Length: (...)

<?xml version="1.0" encoding="UTF-8"?>
<cab-user-prefs xmlns="urn:oma:xml:cab:cab-user-prefs">
  <cab-upp>
    <cab-upp-set>
      <profile id='1234'>
        <display-name> CAB UPP of Joe </display-name>
      </profile>
    </cab-upp-set>
  </cab-upp >
</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:joebloggs@example.com”’s home directory to add a new XUI “sip:friends01@example.com” to the <entry> element of <subscription-list> element.

```
PUT /org.openmobilealliance.cab-user-prefs/users/sip:joebloggs@example.com/index /~/cab-user-prefs
/cap-upp HTTP/1.1
Host: xcap.example.com
...
Content-Type: application/xcap-el+xml; charset="utf-8"
Content-Length: (...)

<cab-upp>
  <subscription-list>
    <entry id='sip:friends01@example.com' />
  </subscription-list>
</cab-upp>
```

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 the <subscription-list> information from the CAB User Preferences Application Usage.

```
GET /org.openmobilealliance.cab-user-prefs/users/sip:joebloggs@example.com/index/~/cab-user-
  prefs/cab-upp/subscription-list 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: (...)
<cab-upp>
  <subscription-list>
    <entry id='sip:friends01@example.com' />
  </subscription-list>
</cab-upp>
```

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 <entry> element identified by (i.e. with an ‘id’ attribute value) ‘sip:friends01@example.com’ from “sip:joebloggs@example.com”’s <subscription-list> element in the CAB User Preferences Application Usage.

```
DELETE /org.openmobilealliance.cab-user-prefs/users/sip:joebloggs@example.com/index/~/cab-user-
  prefs/cab-upp/subscription-list/entry/%5Bid=%22sip:friends01@example.com%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), can be achieved in many ways, 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

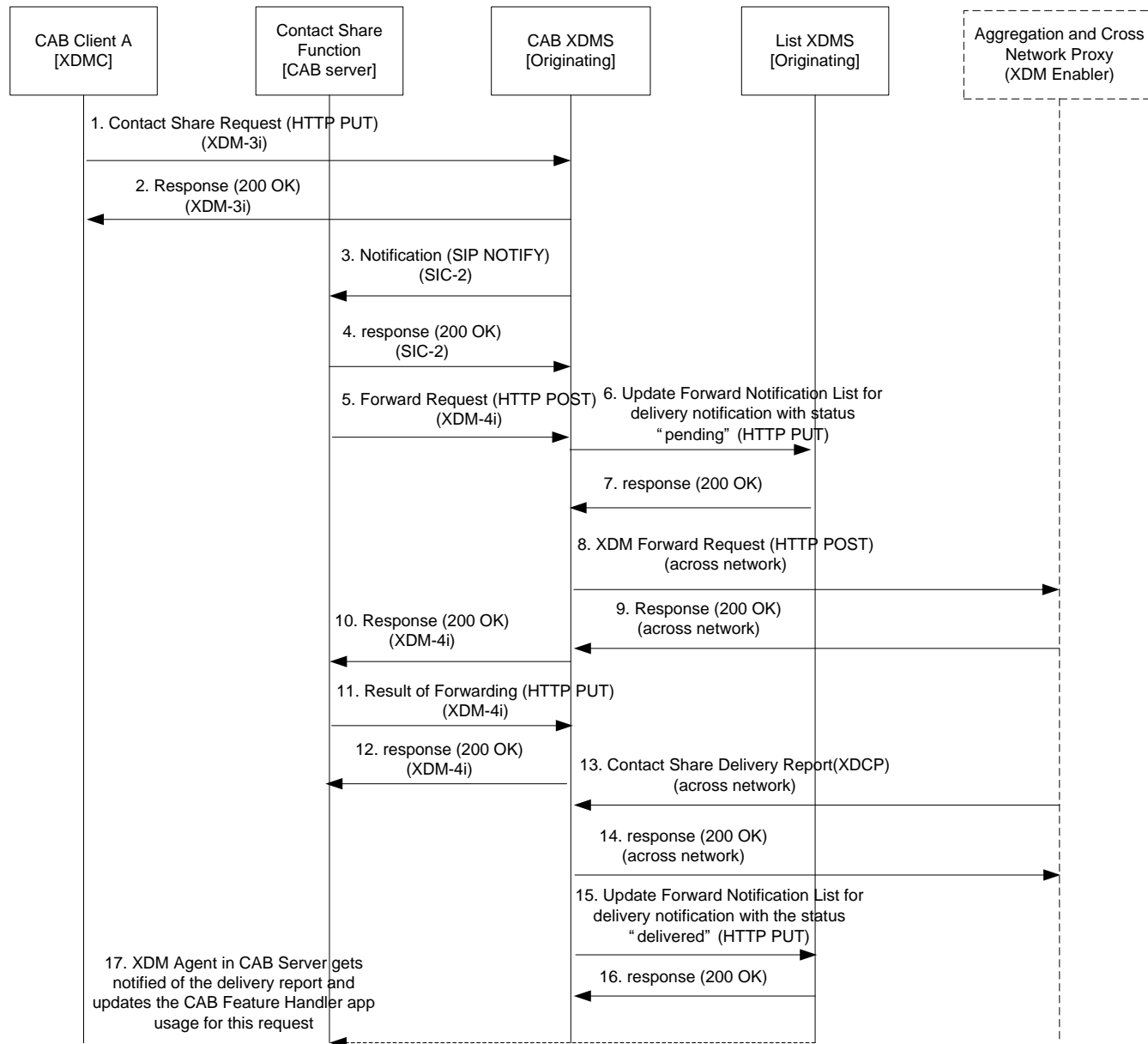


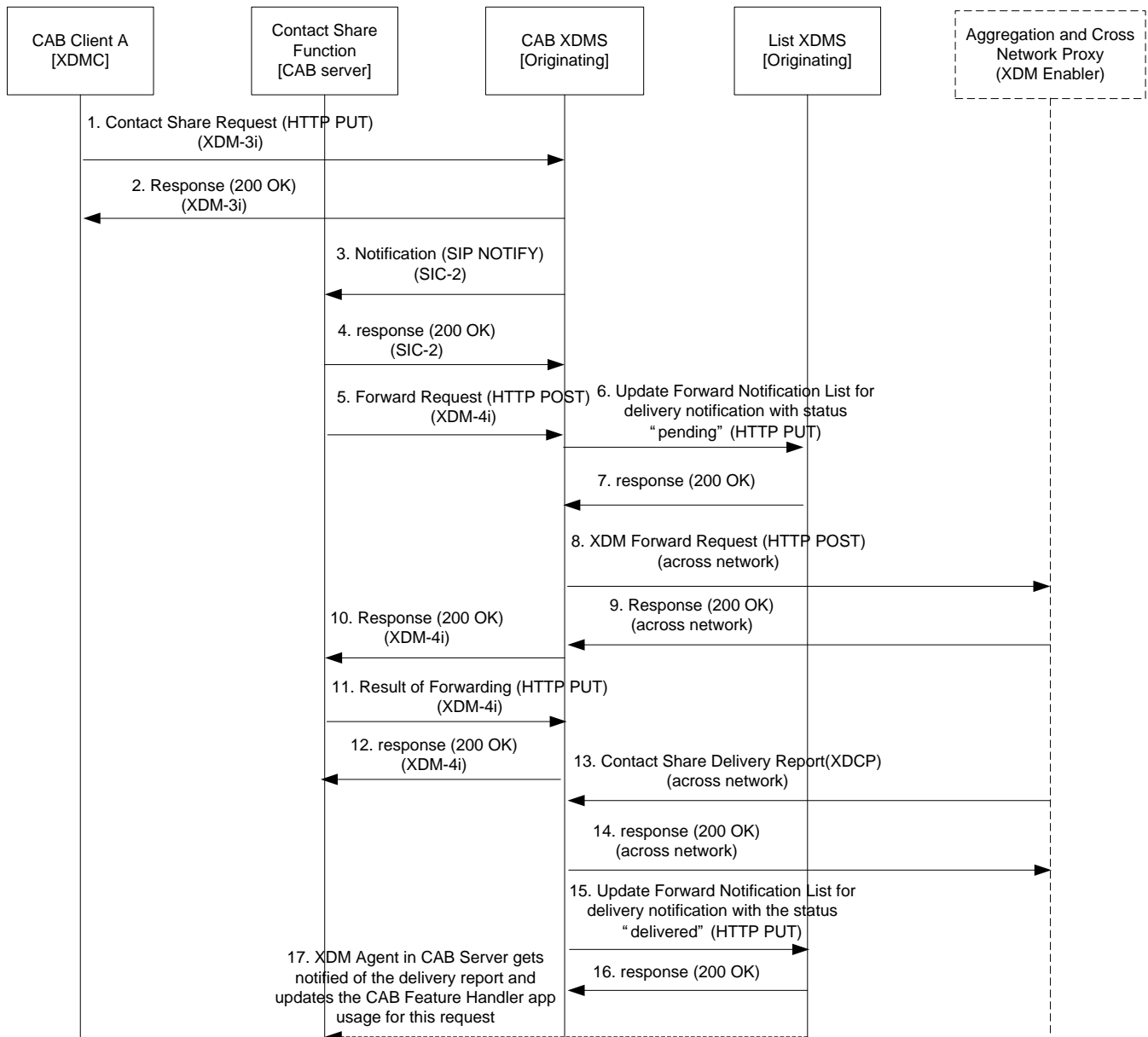
Figure 7 : Flows of AB Forwarding in the originating side towards a CAB user

- 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 (CAB Feature Handler Application Usage) sends the response back to CAB Client using XDM-3i interface.
- Step 3: CAB XDMS (CAB Feature Handler 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-4i 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: The List XDMS 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 Forward XDCCP Request to the recipient's CAB XDMS(s) in the remote domain, using the XDM Enabler interfaces (e.g. XDM-8.2i) as described in the [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: Upon receiving the remote forward response, the CAB XDMS (AB Application Usage) of CAB Client A creates 200 OK response to the forward request and sends it to the Contact Share Function of CAB Server.
- 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 Application Usage 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.4.1.2 PCC forwarding



**Figure 8 : 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 (CAB Feature Handler Application Usage) sends the response back to CAB Client using XDM-3i interface.

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

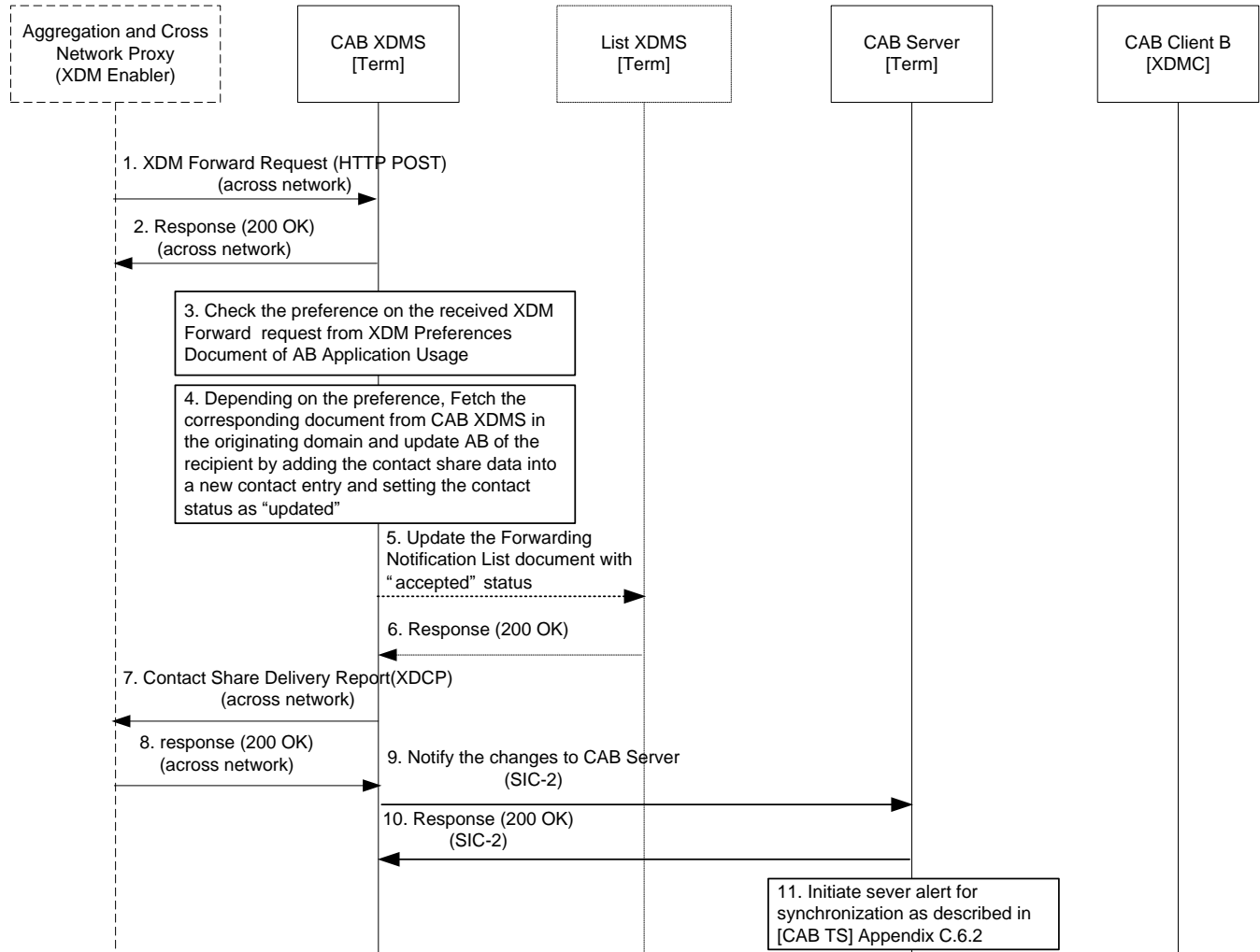
Note: alternatively, the CAB Server can poll via XDM-4i the CAB Feature Handler Document and perform the steps below after detecting a change in the data.

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 XDPC 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: The List XDMS 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 Forward XDPC request to the recipient's CAB XDMS(s) in the remote domain, using the XDM Enabler interfaces (e.g. XDM-8.2i) as described in the [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: Upon receiving the remote forward response, the CAB XDMS (PCC Application Usage) of CAB Client A creates 200 OK response to the forward request and sends it to the Contact Share Function of CAB Server.
- 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 Application Usage 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 (XDPC) from the terminating network.
- Step 14: CAB XDMS (PCC Application Usage) of CAB Client A sends 200 OK response with the XDPC 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.4.1.3 Terminating Side

#### C.4.1.3.1 AB forwarding (Accept)



**Figure 9: 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 Forward XDCCP request from 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 XDM Preferences Document. 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 XDCCP 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 XDCP 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 “Address Book Modifications from Network”.
- Following this, the CAB Client B may initiate synchronization as described in [CAB TS] Appendix C.6.1 “CAB Client Address Book Modifications and Synchronization”.

### C.4.1.3.2 AB forwarding (Confirm)

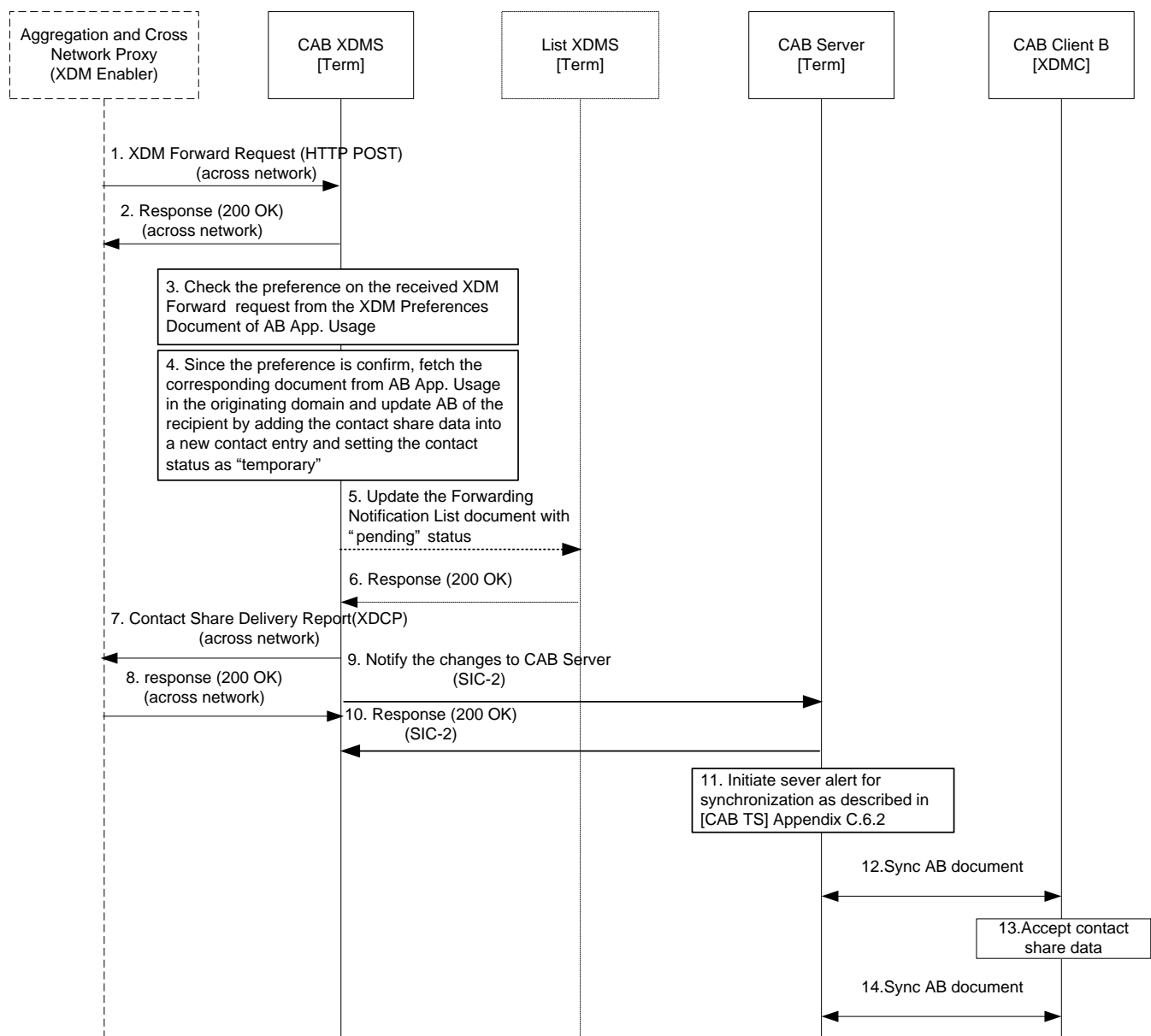
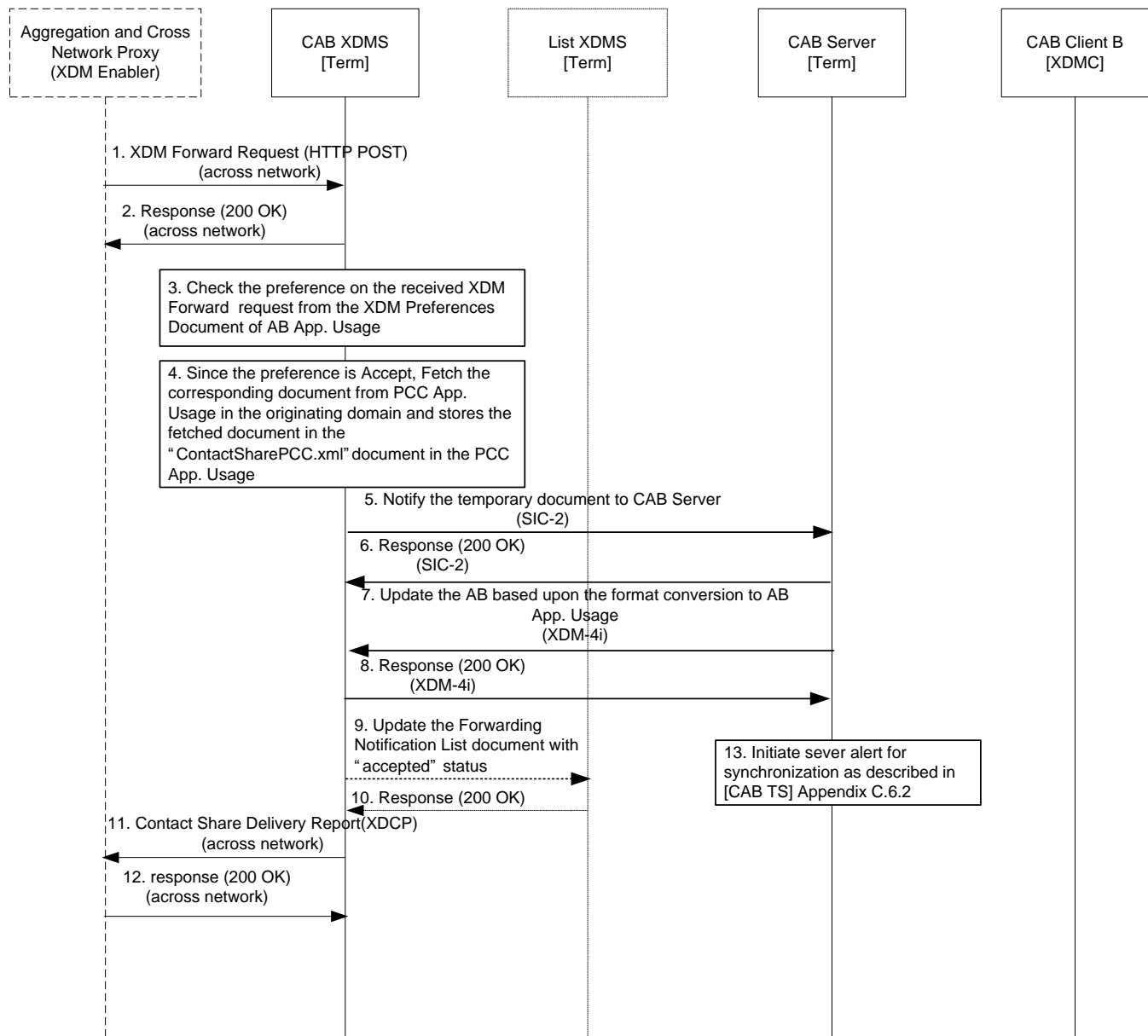


Figure 10: Flows of AB forwarding in the terminating side towards a CAB user

- Step 1: CAB XDMS (AB Application Usage) of CAB Client B receives the Forward XDCP Forward request from 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 XDM Preference document. 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 XDCP 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 XDCP 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 "*Address Book Modifications from Network*".
- Step 12: Following this, the CAB Client B may initiate synchronization as described in [CAB TS] Appendix C.6.1 "*CAB Client Address Book Modifications and Synchronization*".
- 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 "*CAB Client Address Book Modifications and Synchronization*".



### C.4.1.3.3 PCC forwarding (Accept)



**Figure 11: 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 Forward XDCP 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 XDM Preferences Document. 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 XDCP 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 XDCCP 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 “*Address Book Modifications from Network*”.
- Following this, the CAB Client B may initiate synchronization as described in [CAB TS] Appendix C.6.1 “*CAB Client Address Book Modifications and Synchronization*”.

### C.4.1.3.4 PCC forwarding (Confirm)

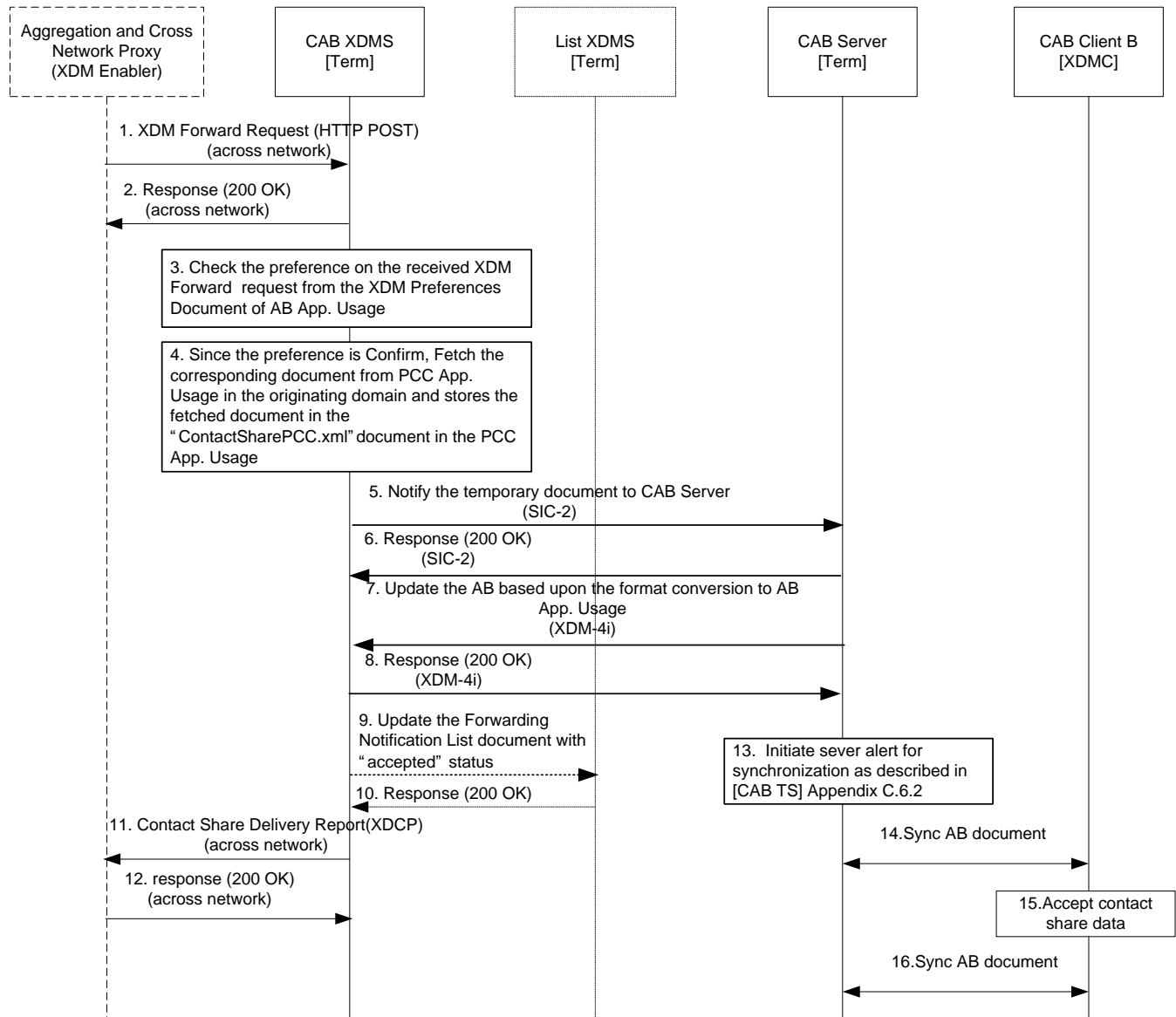
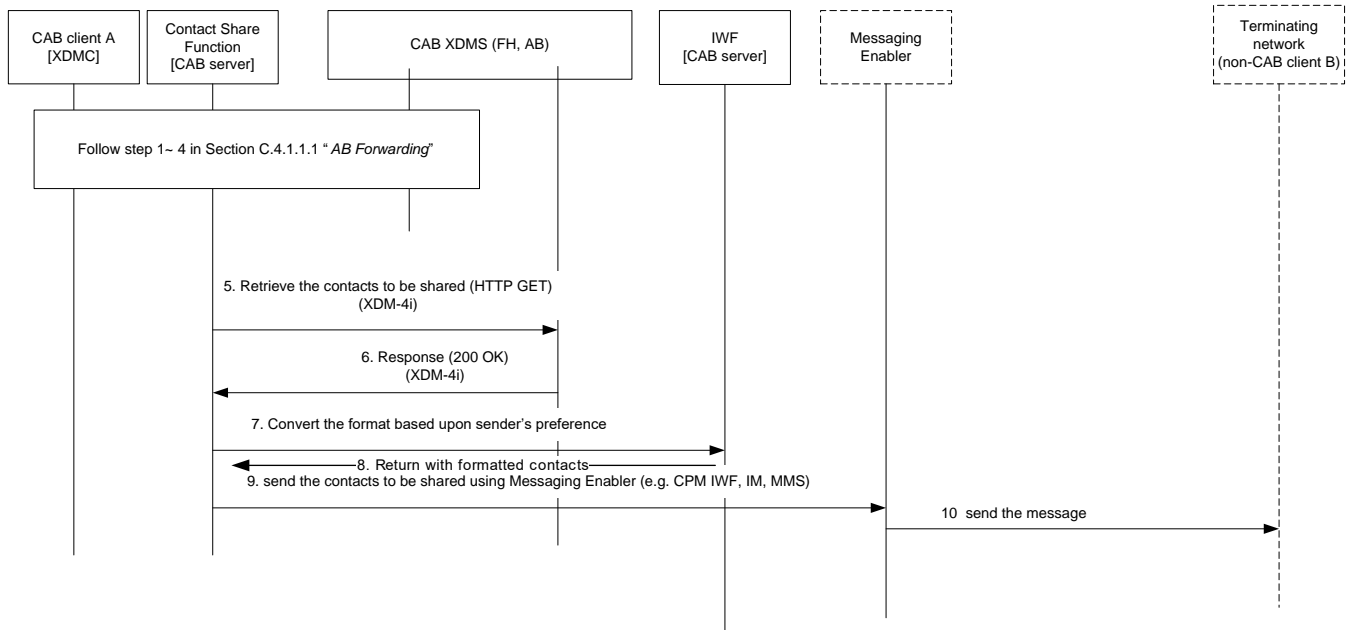


Figure 12: Flows of PCC forwarding in the terminating side towards a CAB user

- Step 1: CAB XDMs (PCC Application Usage) of CAB Client B receives the Forward XDCCP 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 XDM Preferences Document. 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 XDMP 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 “CAB Client Address Book Modifications and Synchronization”.
- 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. “CAB Client Address Book Modifications and Synchronization”

### C.4.2 Contact Share towards a Non CAB User



**Figure 13 : Flows of Contact Share towards a non CAB user**

- Steps 1 through 4 are same as the flow in section C.4.1.1.1 “AB Forwarding”
- 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 AB application usage of the CAB XDMS
- Step 6: Contact Share Function receives the success response (200 OK) from CAB XDMS.

- Step 7: Contact Share Function may request internally to CAB IWF to convert the format of the contact(s) to be shared following section 5.4.6 “*Format Adaptation*”.
- 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 Preferences 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 management of subscription list in the CAB User Preferences Application Usage follows the same flow as described in the Appendix C.3 “Sample XCAP flows for management of CAB XML documents”.

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

Note: The SIP/IP Core between different networks is not shown for simplification.

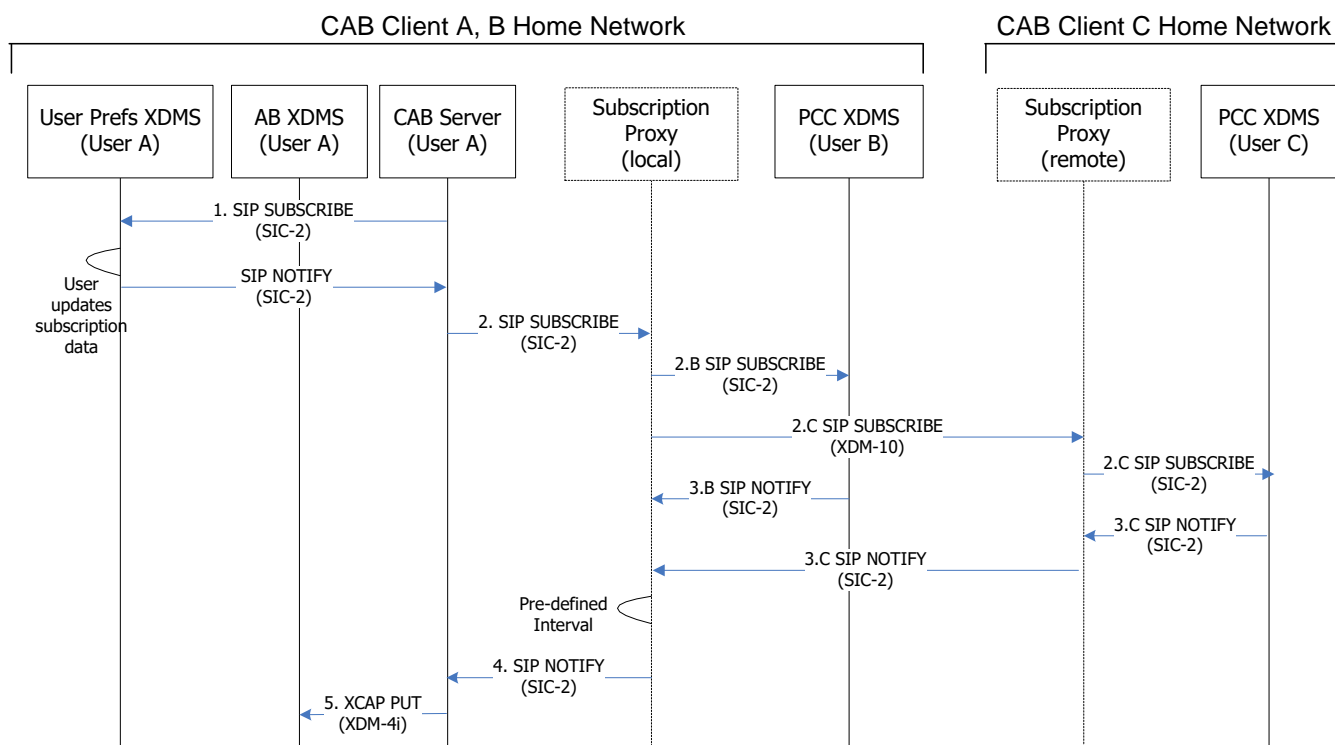


Figure 14 : Contact Subscription flow using the Subscription Proxy

Step 1: Following the subscription of the CAB Server of CAB User A through the interface SIC-2 to the updates of the subscription list in the CAB User Preferences Application Usage, the Contact Subscription Function of the CAB Server of the 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 of the CAB User A’s AB Application Usage.

Step 2: The Contact Subscription Function of the CAB Server of the CAB user A sends a SIP SUBSCRIBE request to the Subscription Proxy via the SIP/IP Core using the xcap-diff event package following the rules and procedures described in section 6.1.2 “Subscribing to Changes in the XDM Resources” of [XDM Core]. The Request URI of the SIP SUBSCRIBE 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 [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 SIP SUBSCRIBE request, and return 200 OK response to the CAB Server through the SIP/IP Core

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

Upon receiving the SIP SUBSCRIBE request the PCC Application Usage of the CAB User B verifies the Access Permissions regarding the CAB User A. Assuming the CAB User A is successfully authorized, the PCC Application Usage of User B responds with 200 OK response through the SIP/IP Core to the CAB Server.

The received SIP SUBSCRIBE request is stored in the Request History Information Document of PCC Application Usage, subject to history preferences defined in XDM Preferences Document, following the rules defined in section 5.7.2 “*Request History Information Document*” of [XDM Core]. Contact Status Function of the CAB Server may use the Request History Document of PCC Application Usage, subject to CAB User preferences, as described in section 5.8.3 “*Incoming Contact Subscription Requests*”.

Step 2.C: The SIP SUBSCRIBE request is sent to the PCC Application Usage of the CAB User C in the remote network via the SIP/IP Core, and possibly via Subscription Proxy in the remote network.

Upon receiving the SIP SUBSCRIBE request with the Request URI set to “sip:userC@other\_domain.com;aud=org.openmobilealliance.cab-pcc”, the PCC Application Usage of the CAB User C verifies Access Permissions regarding the CAB User A. Assuming the CAB User A is successfully authorized, the PCC Application Usage of the CAB User C responds with 200 OK response following the same path back to the CAB Server.

The received SIP SUBSCRIBE request is stored in the Request History Information Document of PCC Application Usage, subject to history preferences defined in XDM Preferences Document, following the rules defined in section 5.7.2 “*Request History Information Document*” of [XDM Core]. Contact Status Function of the CAB Server may use the Request History Document of PCC Application Usage, subject to CAB User preferences, as described in section 5.8.3 “*Incoming Contact Subscription Requests*”.

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

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

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

Until this state, the initial notifications from the PCC Application Usage(s) in the same network and remote network were received. The Contact Subscription Function of the CAB server A will respond all the way back with 200 OK response.

Step 5: The Contact Subscription Function of the CAB Server A receives the SIP NOTIFY request with the states of the subscriptions for the CAB User B and the CAB User C and updates the Contact Status of the Contact Entries corresponding to the CAB User B and CAB User C in the AB Application Usage of the CAB User A based upon the user preference of the CAB User A. The updates in Contact Status of the 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 the CAB Server A sends an alert to the CAB User A to indicate changes occurred in the CAB User A's AB Application Usage as described in step 5 of [CAB TS] Appendix C.6.2 "Address Book Modifications from Network". Following this, the CAB Client A may initiate synchronization as described in [CAB TS] Appendix C.6.1 "CAB Client Address Book Modifications and Synchronization".

### C.5.2 CAB Client Subscribes to contact's PCC through the reactive authorization

The CAB Client triggers the Contact Subscription to contacts PCC by the updates in the CAB User's subscription list in the CAB User Preferences Application Usage.

This example flow describes the CAB User A's Contact Subscription to CAB User B through the reactive authorization.

For the activation of the reactive authorization for CAB User A, CAB User B needs to update the XDM Preferences Document for the PCC Application Usages in such way that the Request History Information Documents for the PCC Application Usage are updated with information about unsuccessful XDM Requests using procedures in section 6.1.1.2 of [XDM Core]. And the CAB Server of CAB User B also needs to subscribe for changes to the Request History Information Documents as described in [XDM Core] Appendix G. "Reactive Authorization of XDM Requests using Request History Information Documents".

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

Note: The SIP/IP Core or Subscription Proxy between different networks is not shown for simplification.

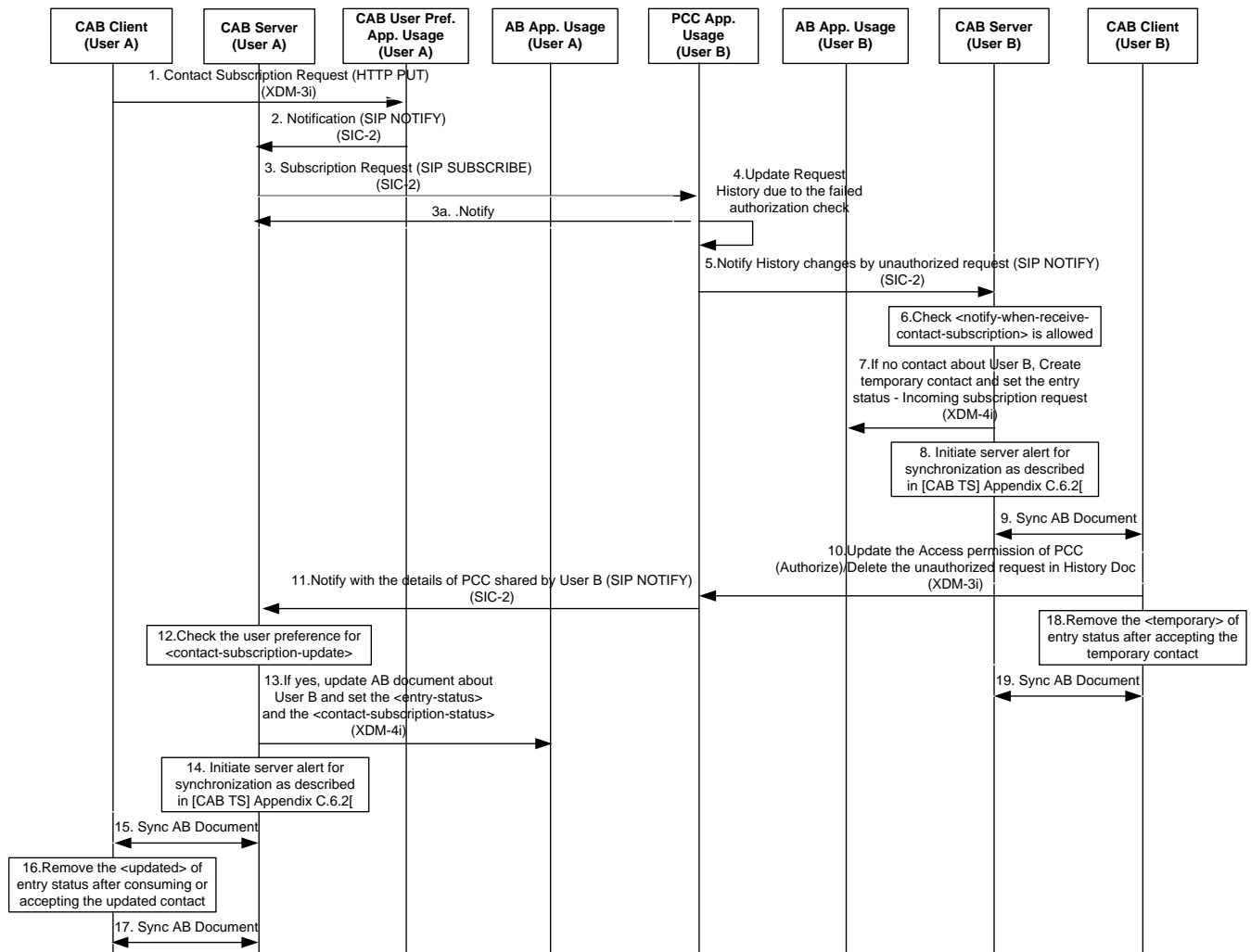


Figure 15 : Contact Subscription flow through the reactive authorization



Step 1: CAB Client A performs a HTTP PUT containing Contact Subscription data in CAB User Preferences Document in the CAB XDMS, using XDM-3i interface

Step 2: CAB User Preferences Application Usage of CAB XDMS notifies Contact Subscription Function (through the CAB Server's XDM Agent) about document changes using SIC-2 interface.

Note: alternatively, the CAB Server can poll via XDM-4i the CAB User Preferences Document and perform the steps below after detecting a change in the data.

Step 3: The Contact Subscription Function of CAB Server of the CAB User A sends a SIP SUBSCRIBE request towards the PCC Application Usage of CAB User B via the Subscription Proxy/SIP/IP Core as described in [CAB TS] Appendix C.5.1 "*CAB Server Subscribes to contacts PCCs using Subscription Proxy*".

Step 4: The Request History Information Document of the PCC Application Usage of CAB User B according to XDM Preferences Document set by the CAB User B is updated with the information about unsuccessful XDM operations.

Step 5: The CAB Server of CAB User B receives the notification from the CAB XDMS (PCC Application Usage).

Note: alternatively, the CAB Server can poll via XDM-4i the Request History Information Document and perform the steps below after detecting a change in the data.

Step 6: The CAB Server of CAB User B checks the preferences of the CAB User B for indicating whether to notify the CAB User for incoming Contact Subscription request.

Step 7: The CAB Server of CAB User B creates the temporary contact and set the <updated> element of entry status as "incoming subscription request".

Step 8: The CAB Server of CAB User B sends an alert to CAB Client of CAB User B to indicate changes occurred in the CAB User's AB Document as described in the step 5 of [CAB TS] Appendix C.6.2.

Step 9: The CAB Client of CAB User B may initiate synchronization as described in [CAB TS] Appendix C.6.1.

Step 10: The CAB User B checks if the synchronized AB Document contains the entry status for old unauthorized XDM requests. If that is the case, the CAB User B's UE prompts the User with the list of unauthorized old XDM requests and asks what to do with them and continues the possible choices per the entry status as described in [XDM Core] Appendix G. "*Reactive Authorization of XDM Requests using Request History Information Documents*". If it contains a new unauthenticated XDM Request, the CAB User B's UE prompts as described in [XDM Core] Appendix G. "*Reactive Authorization of XDM Requests using Request History Information Documents*" but with only one XDM request.

In this example flow, the CAB User B selects to authorize the CAB User A. Thus, CAB Client of CAB User B updates the Access Permissions Document related to the PCC Application Usage of CAB User B by using procedure described on specified in [XDM Core] section 6.1.1.2.4 to grant the CAB User A's access to the requested XDM Document. The CAB Client of CAB User B also uses procedures described in [XDM Core] section 6.1.1.2.5 to delete the part in the Request History Document that contained request information about the now authorized user. This is done because the CAB User B needs to make sure that the user is not prompted again with old unauthorized requests from e.g. other UEs that the user might have.

Step 11: The CAB Server of CAB User A gets notified of the details of PCC Document shared by the CAB User B, following the change in authorization for CAB User A's subscription (initiated in Step 3).

Note: If the subscription expires before the authorization change occurs, the SIP SUBSCRIBE will need to be either refreshed to extend the expiry or re-initiated.

Step 12: The CAB Server of CAB User A checks the preferences of the CAB User A for indicating whether the AB Application Usage of CAB User A is updated automatically when information resulting from Contact Subscription is received

Step 13: The CAB Server of CAB User A updates AB Document about CAB User B and set the <updated> element of entry status as "contact subscription request" and the <contact-subscription-status> as "active".

Step 14: The CAB Server of CAB User A sends an alert to CAB Client of CAB User A to indicate changes occurred in the CAB User's AB Document as described in the step 5 of [CAB TS] Appendix C.6.2.

Step 15: The CAB Client of CAB User A may initiate synchronization as described in [CAB TS] Appendix C.6.1.

Step 16: The CAB User A removes the <updated> element of <entry-status> element after consuming the updated contact.

Step 17: The CAB Client of CAB User A may initiate synchronization as described in [CAB TS] Appendix C.6.1.

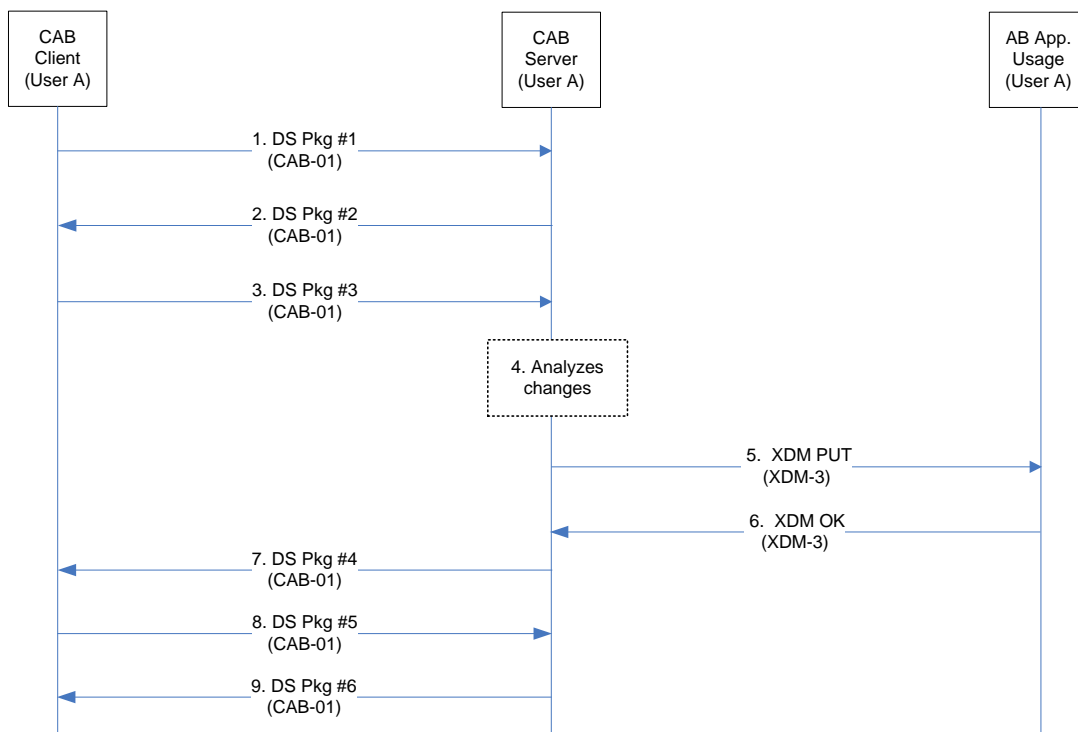
Step 18: The CAB User B removes the <temporary> element of <entry-status> element after accepting the updated contact.

Step 19: The CAB Client of CAB User 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 16 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 16 : CAB Client Address Book Modifications and Synchronization Flow**

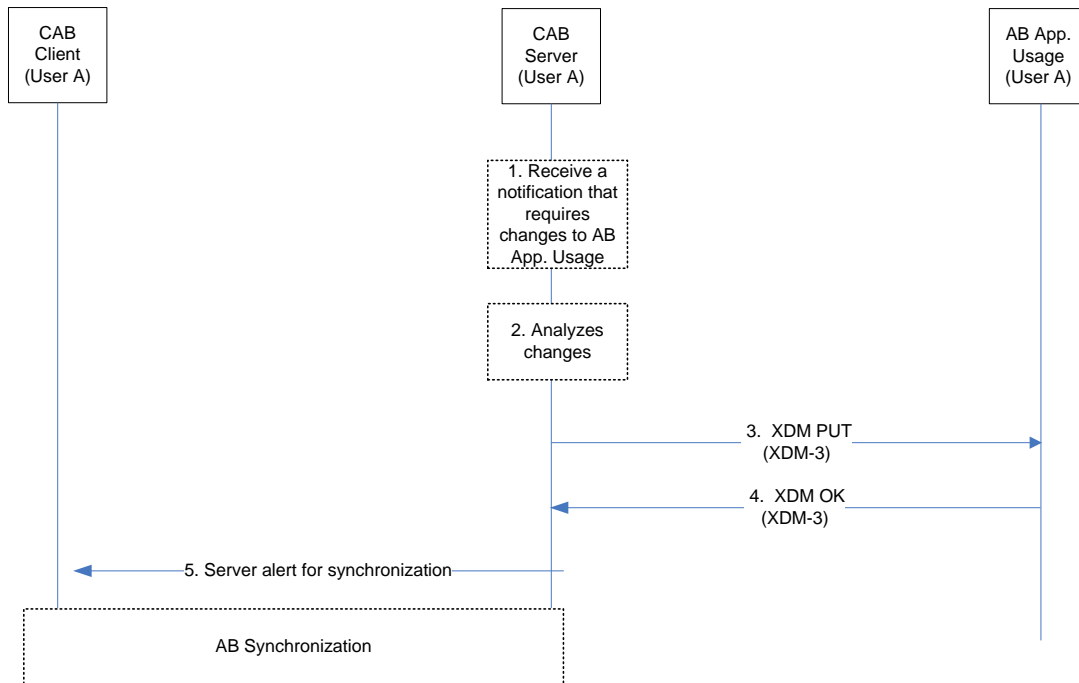
1. Based on modifications on the CAB User's client address book, the CAB Client sends an OMA DS initialization message (OMA DS Pkg #1) to the CAB Server. The message includes server challenge (if needed), device capability information, DS sync type, and data type (i.e., the user's Address Book) to be 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 Application Usage. This may involve multiple XDM requests, but only one request is depicted in the figure.

6. The AB Application Usage 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 Application Usage, 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 data.

## C.6.2 Address Book Modifications from Network

Figure 17 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 17 : Address Book Modifications from Network**

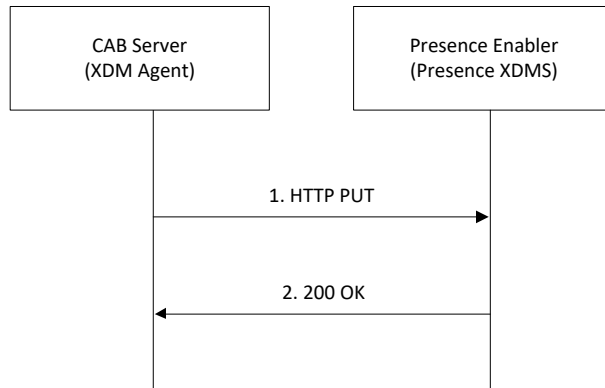
1. The CAB Server receives a notification that may require a change to the user's AB in the network (e.g., receives a SIP NOTIFY request from an XDM Subscription Proxy or PCC Application Usage of the Published Contact Card, based on an active subscription, change from multiple devices).
2. The CAB Server analyzes the changes found in the notifications.
3. The CAB Server updates the AB Application Usage.
4. The AB Application Usage 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. “*CAB Client Address Book Modifications and Synchronization*”.

## C.7 CAB Capability Flows

The following provides three basic message flows for exchange of CAB capability information based on the Presence Enabler. The first two flows are different alternatives for the same functionality, namely the publication of the CAB Capability.

### C.7.1 CAB Server publishes CAB capability as Permanent Presence State

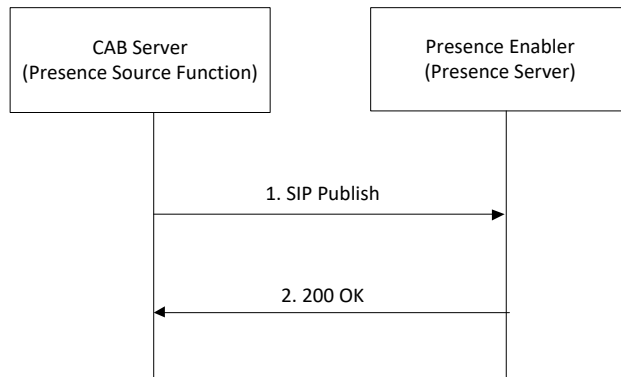


**Figure 18 : CAB Server publishes CAB capability as Permanent Presence State**

Step 1: A user becomes CAB user. The CAB Server through the XDM Agent updates the CAB capability of the new CAB User at that user’s Permanent Presence State at the Presence Enabler (Presence XDMS).

Step 2: The response is sent back to the CAB Server.

### C.7.2 CAB Server publishes CAB capability as a regular SIP Publish



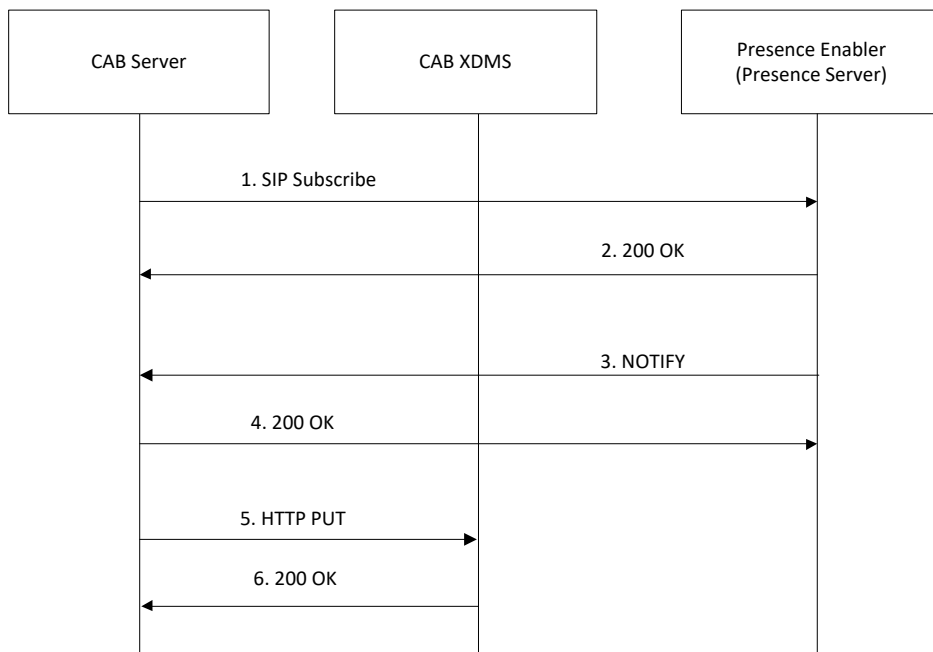
**Figure 19 : CAB Server publishes CAB capability as a regular SIP Publish**

Step 1: A user becomes CAB user. The CAB Server through the Presence Source Function publishes the CAB capability of the new CAB User to that user’s Presence Document at the Presence Enabler (Presence Server).

Step 2: The response is sent back to the CAB Server.

These requests are exchanged through the SIP/IP Core, which is not shown for simplicity

### C.7.3 CAB Server subscribes to CAB Capability of its user’s contacts



**Figure 20 : CAB Server subscribes to CAB Capability of its user’s contacts**

Step 1: CAB server (Presence Watcher Function) subscribes to the Presence (CAB service tuple) of the contacts in the address book of each user it serves. This can be also incremental subscriptions (e.g. when a user adds a new contact). The subscription is performed by either of the following procedures:

Step 1a: individual subscriptions to each of the contacts in the address book of each of the users (with multiple subscriptions to a specific contact if included in the address books of more than one user)

Step 1b: single anonymous subscription for each contact included in at least one address book of the users served by that CAB server.

Step 2: The response is sent back to the CAB Server.

Step 3: A notification is received from the Presence Enabler, informing of a change in the CAB Service tuple of a subscribed contact. Multiple notifications for a single contact may be received if Step 1a was followed

Step 4: The response is sent back to the Presence Enabler.

Step 5: The CAB server (XDM Agent) updates the CAB XDMS, by updating the “contact type” (CAB/non-CAB) of the corresponding contact in the address book of the user with that specific contact in his/her address book, and subject to the information received in the notifications in Step 3 (e.g. if Step 1a was followed no update will be performed in those users’ address book for which no notification was received, meaning the contact did not grant access to his/her Presence data).

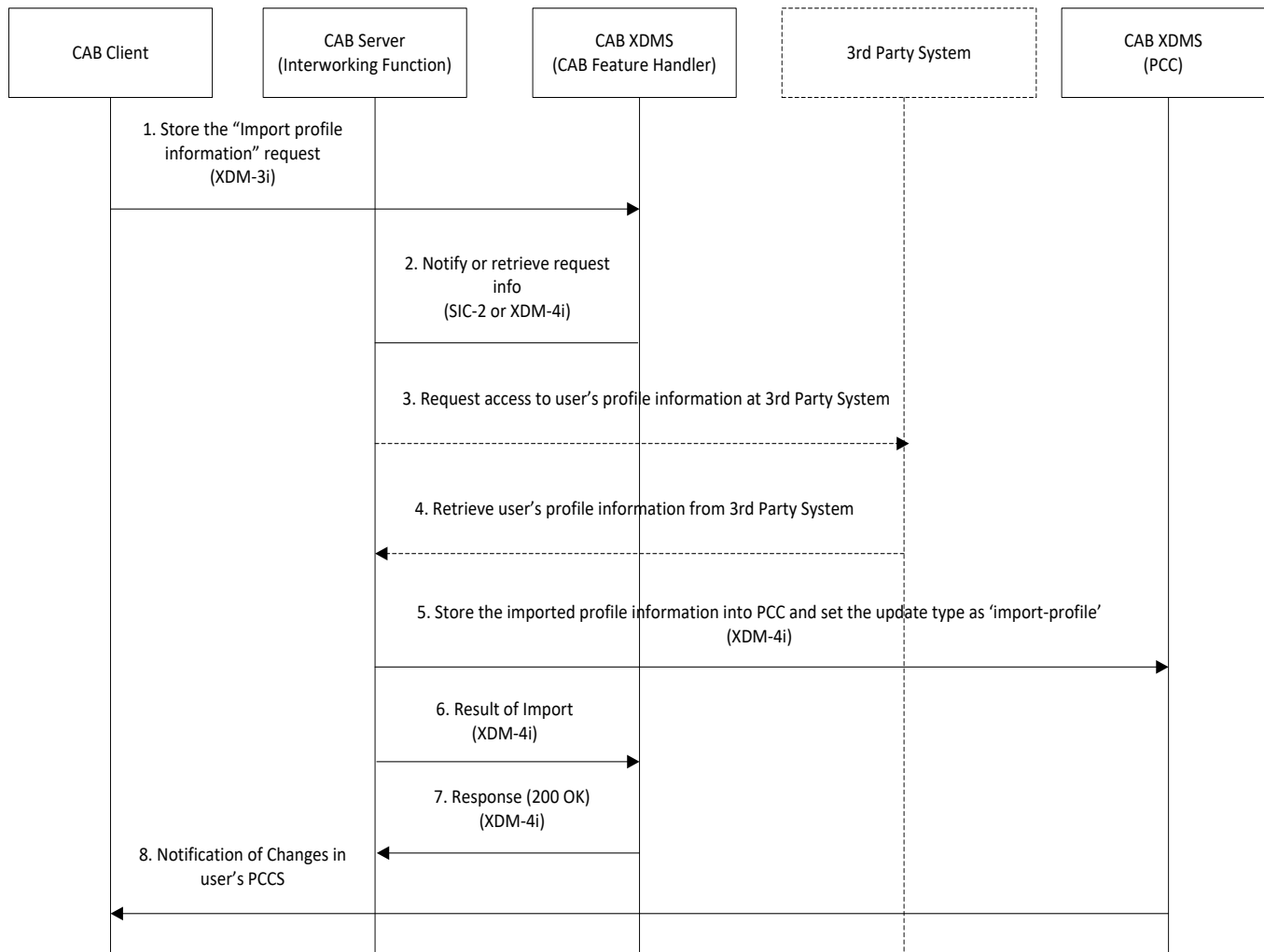
Step 6: The response is sent back to the CAB Server.

The SIP requests (steps 1, 2, 3 and 4) are exchanged through the SIP/IP Core, which is not shown for simplicity

## C.8 External Profile Information Import Flows

### C.8.1 Automatic approval of External Profile Information Import

The following provides basic message flows for the import of External Profile information from 3rd parties in case the CAB User set his/her preference as ‘automatic’ i.e. automatic approval of PCC Changes.



**Figure 21 : External Profile Information Import Flows**

Step 1: The CAB Client makes an “import profile” request by writing/storing the request information formatted to the <import-profile> element as described in the CAB Feature Handler Application Usage [CAB 1.1 XDMS].

Step 2: The CAB Server retrieves the “import profile” request information from the CAB Feature Handler Application Usage by either of the following methods:

Step 2a: CAB Feature Handler Application Usage notifies a change to the CAB Feature Handler Document to the CAB Server assuming a prior subscription to the changes is in place by the CAB Server.

Step 2b: CAB Feature Handler Application Usage is polled by the CAB Server for changes in the CAB Feature Handler 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 3rd party system access to the CAB User’s External Profile information, by supplying the necessary access parameters.

Step 4: Upon obtaining the access, the CAB Server (i.e. Interworking Function) retrieves/receives the CAB User’s External Profile information from the 3rd party system and transforms the imported data into the CAB Format.

Different approaches are possible to implement steps 3 and 4, namely using:

- Proprietary mechanisms e.g. via proprietary APIs exposed by the 3<sup>rd</sup> party systems.
- Native CAB Enabler procedures e.g. if 3<sup>rd</sup> party systems support receiving native CAB cross-domain PCC subscription requests (one-time subscriptions with expiry set to 0) as described in [CAB 1.1 TS] section 5.2 “Contact Subscription Function”

- CAB APIs e.g. if 3<sup>rd</sup> party system exposes CAB APIs as described in [CAB APIs] sections 6.1.6.3, 6.1.7.3 and H.1.

Note: the approaches outlined the preceding bullets are out of scope of the CAB Enabler.

Step 5: The CAB Server stores the resulting data and the meta information of updated PCC Document containing the update type set as 'import-profile' in the PCC Document of the PCC Application Usage [CAB 1.1 XDMS] subject to CAB User's preferences. This step assumes automatic update because the current PCC is overwritten.

Step 6: CAB Server performs an XCAP PUT using XDM-4i containing the result of import profile operation in the <response> element of the CAB Feature Handler Document in the CAB XDMS.

Step 7: CAB Feature Handler Application Usage from the CAB XDMS responds with 200 OK back to CAB Server using XDM-4i interface.

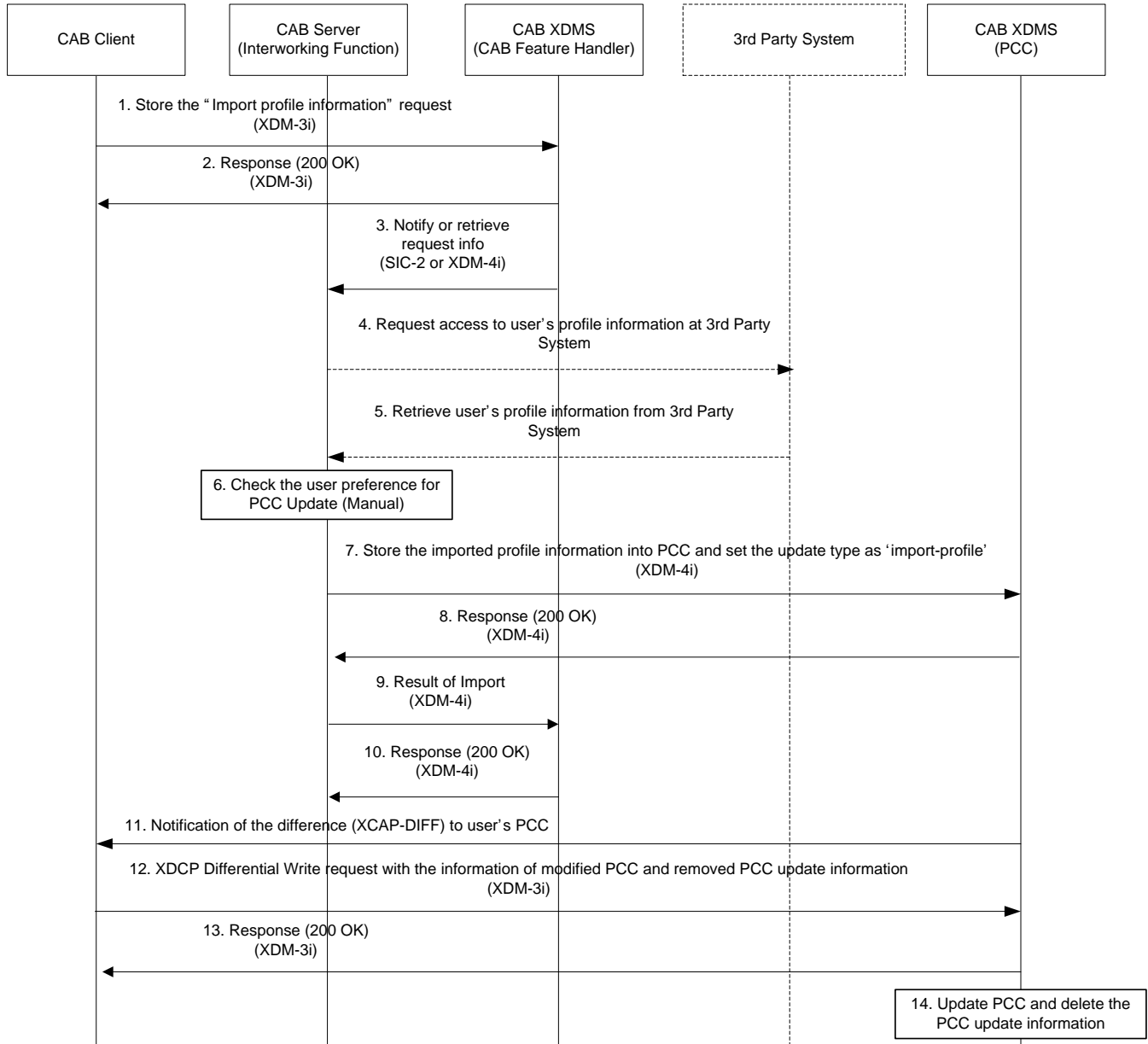
Step 8: In case the CAB User's CAB Client is subscribed to changes in the PCC Application Usage then it is notified of the change in the PCC by either of the following methods:

Step 8a: CAB Client receives a SIP NOTIFY assuming a prior subscription to CAB User's own PCC Document changes using SIP-based subscription.

Step 8b: CAB Client receives a Push OTA Message assuming a prior subscription to CAB User's own PCC Document changes using XDCCP-based subscription.

## C.8.2 Manual approval of External Profile Information Import

The following provides basic message flows for the import of External Profile information from 3rd parties in case the CAB User sets his/her preference as 'manual' i.e. manual approval of PCC changes.



Step 1: The CAB Client makes an “import profile” request by writing/storing the request information formatted to the <import-profile> element as described in the CAB Feature Handler Application Usage [CAB 1.1 XDMS].

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

Step 3: The CAB Server retrieves the “import profile” request information from the CAB Feature Handler Application Usage by either of the following methods:

Step 3a: CAB Feature Handler Application Usage notifies a change to the CAB Feature Handler Document to the CAB Server assuming a prior subscription to the changes is in place by the CAB Server.

Step 3b: CAB Feature Handler Application Usage is polled by the CAB Server for changes in the CAB Feature Handler Document based on a pre-determined time interval using XCAP.

Step 4: The CAB Server (i.e. Interworking Function) on behalf of the CAB User requests the 3rd party system access to the CAB User’s External Profile information, by supplying the necessary access parameters.

Step 5: Upon obtaining the access, the CAB Server (i.e. Interworking Function) retrieves/receives the CAB User’s External Profile information from the 3rd party system and transforms the imported data into the CAB Format.



Different approaches are possible to implement steps 4 and 5, namely using:

- Proprietary mechanisms e.g. via proprietary APIs exposed by the 3<sup>rd</sup> party systems.
- Native CAB Enabler procedures e.g. if 3<sup>rd</sup> party systems support receiving native CAB cross-domain PCC subscription requests (one-time subscriptions with expiry set to 0) as described in [CAB 1.1 TS] section 5.2 “*Contact Subscription Function*”
- CAB APIs e.g. if 3<sup>rd</sup> party system exposes CAB APIs as described in [CAB APIs] sections 6.1.6.3, 6.1.7.3 and H.1.

Note: the approaches outlined the preceding bullets are out of scope of the CAB Enabler.

Step 6; The CAB Server checks the user preference for the PCC update by External Profile import.

Step 7: The CAB Server stores the resulting changes in the PCC Update Document, and its reference and the update type set as ‘import-profile’ are stored in the PCC Document of the PCC Application Usage as described in [CAB 1.1 XDMS].

Step 8: PCC Application Usage sends the response back to CAB Server using XDM-4i interface.

Step 9: CAB Server performs an XCAP PUT using XDM-4i containing the result of import profile operation in the <response> element of the CAB Feature Handler Document in the CAB XDMS.

Step 10: CAB Feature Handler Application Usage from the CAB XDMS responds with 200 OK back to CAB Server using XDM-4i interface.

Step 11: In case the CAB User’s CAB Client is subscribed to changes in the PCC Application Usage then it is notified of the change in the PCC by either of the following methods:

Step 11a: CAB Client receives a SIP NOTIFY assuming a prior subscription to CAB User’s own PCC Document changes using SIP-based subscription.

Step 11b: CAB Client receives a Push OTA Message assuming a prior subscription to CAB User’s own PCC Document changes using XDCP-based subscription.

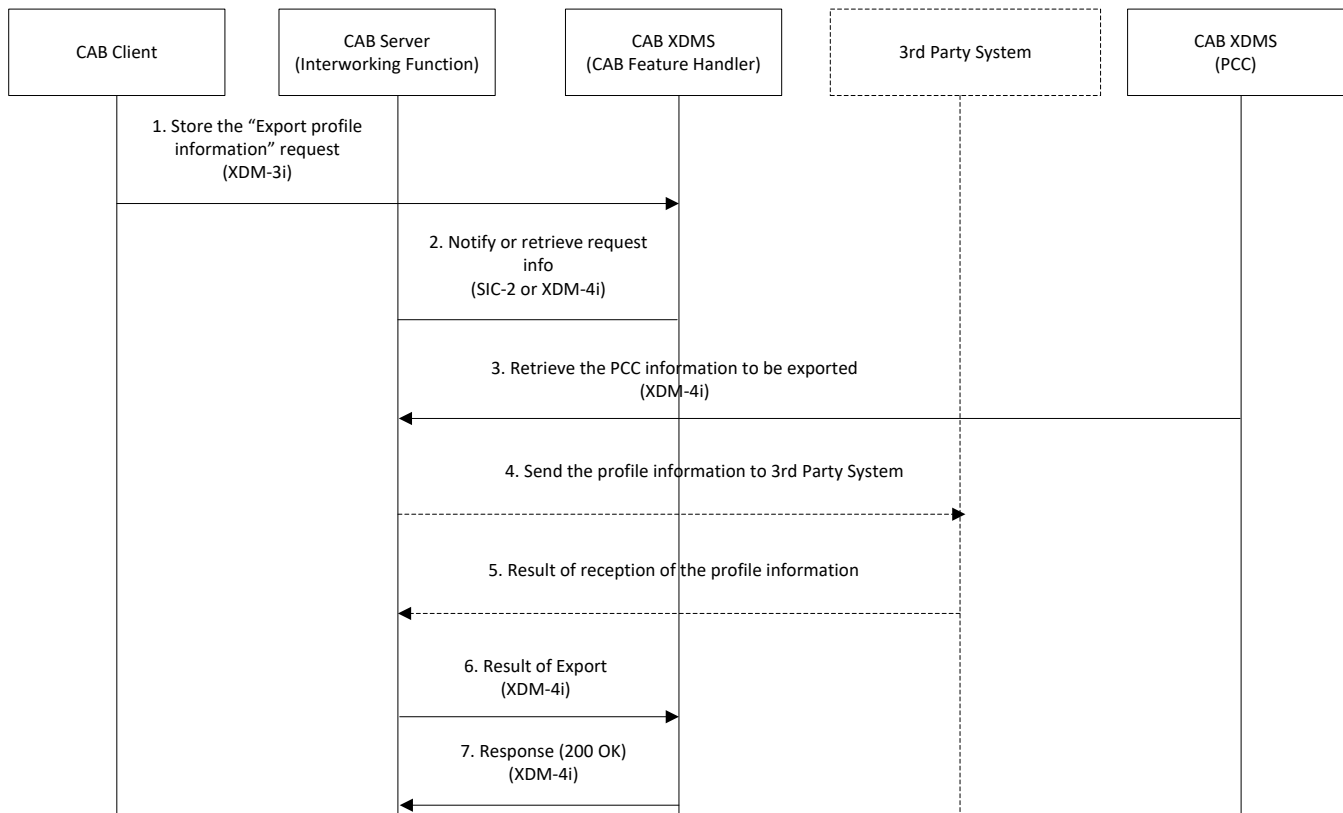
Step 12: CAB Client presents the proposed changes to the CAB User. The CAB User selects which changes to PCC he/she wants to approve. The CAB Client removes unwanted changes to the PCC by removing corresponding patches and sends a Differential Write request with the information of modified PCC and PCC update reference to be removed.

Step 13: PCC Application Usage sends the response back to CAB Client using XDM-3i interface.

Step 14: PCC Application Usage updates PCC Document and delete the PCC Update Document by using PCC update reference.

## C.9 PCC Information Export Flows

The following provides basic message flows for the export of PCC information to 3rd parties.



**Figure 22 : PCC Information Export Flows**

Step 1: The CAB Client makes an “export profile” request by writing/storing the request information formatted to the <export-profile> element as described in the CAB Feature Handler Application Usage [CAB 1.1 XDMS].

Step 2: The CAB Server retrieves the “export profile” request information from the CAB Feature Handler Application Usage by either of the following methods:

Step 2a: CAB Feature Handler Application Usage notifies a change to the CAB Feature Handler Document to the CAB Server assuming a prior subscription to the changes is in place by the CAB Server.

Step 2b: CAB Feature Handler Application Usage is polled by the CAB Server for changes in the CAB Feature Handler Document based on a pre-determined time interval using XCAP.

Step 3: The CAB Server retrieves the PCC information to be exported from the PCC Application Usage [CAB 1.1 XDMS] and convert the format of the PCC to be exported into External Profile.

Step 4: The CAB Server (i.e. Interworking Function) on behalf of the CAB User sends the formatted profile information to requests the 3rd party system, by supplying any necessary identification/ access parameters.

Step 5: 3rd party system responds with the result of the export profile operation

Different approaches are possible to implement steps 4 and 5, namely using:

- Proprietary mechanisms e.g. via proprietary APIs exposed by the 3<sup>rd</sup> party systems
- Native CAB Enabler procedures e.g. if 3<sup>rd</sup> party systems support receiving Contact Share requests for PCC as described in [CAB 1.1 TS] section 5.3.2 “Procedures at the Receiving Side”.
- CAB APIs e.g. if 3<sup>rd</sup> party system exposes CAB APIs as described in [CAB APIs] sections 6.1.7.4 and H2.

Note: the approaches outlined in the preceding bullets are out of scope of the CAB Enabler

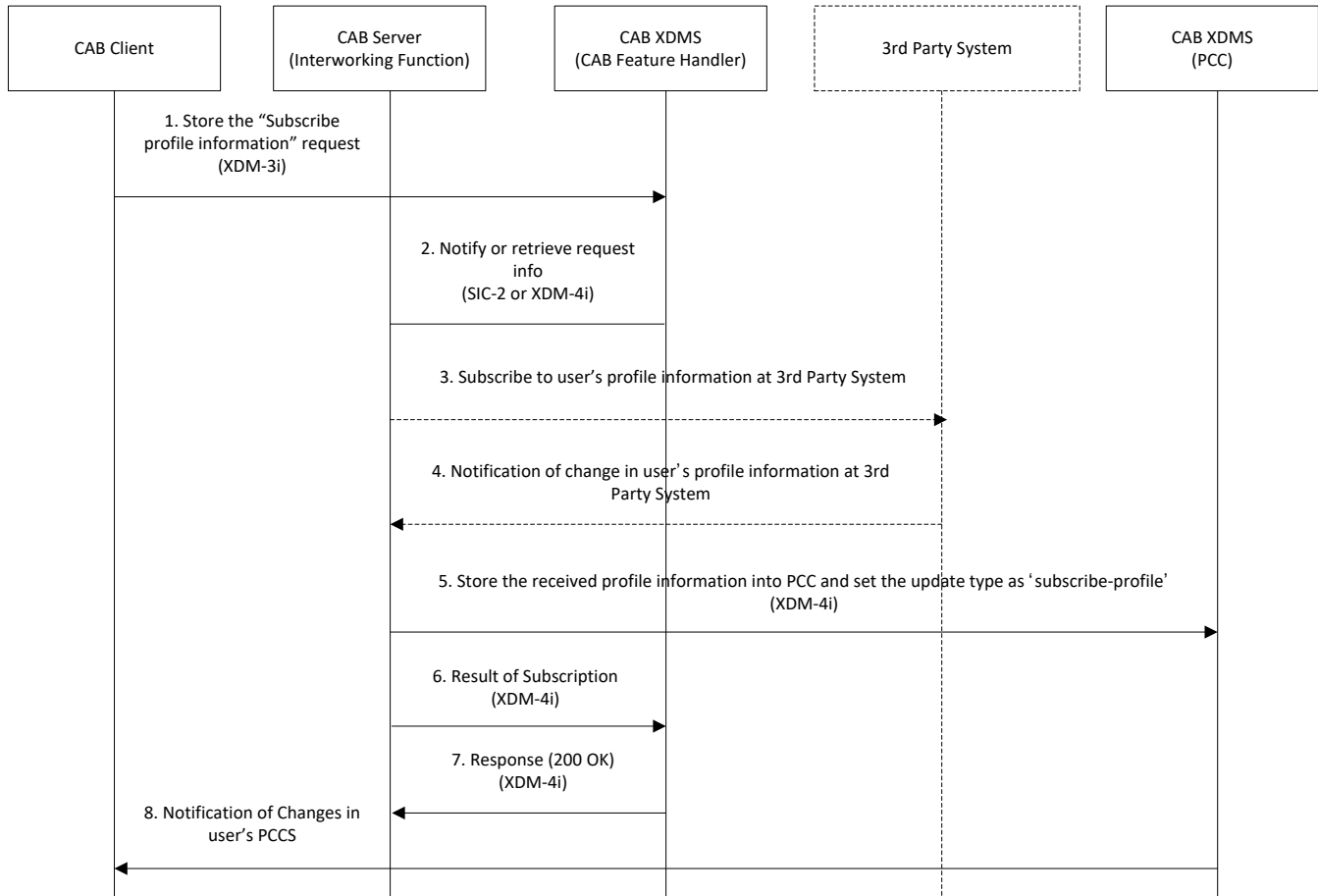
Step 6: CAB Server performs an XCAP PUT using XDM-4i containing the result of export profile operation in the <response> element of the CAB Feature Handler Document in the CAB XDMS.

Step 7: CAB Feature Handler Application Usage from the CAB XDMS responds with 200 OK back to CAB Server using XDM-4i interface.

## C.10 Subscription to External Profile Information Flows

### C.10.1 Automatic approval of Subscription to External Profile Information

The following provides basic message flows for the subscription to changes in the user’s External Profile at a 3rd party system in case the CAB User sets his/her preference as ‘automatic’ i.e. automatic approval of PCC changes.



**Figure 23 : Subscription to External Profile Information Flows**

Step 1: The CAB Client makes a “subscribe profile” request by writing/storing the request information formatted to the <subscribe-profile> element as described in the CAB Feature Handler Application Usage [CAB 1.1 XDMS].

Step 2: The CAB Server retrieves the “subscribe profile” request information from the CAB Feature Handler Application Usage by either of the following methods:

Step 2a: CAB Feature Handler Application Usage notifies a change to the CAB Feature Handler Document to the CAB Server assuming a prior subscription to the changes is in place by the CAB Server.

Step 2b: CAB Feature Handler Application Usage is polled by the CAB Server for changes in the CAB Feature Handler 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 subscribes to the CAB User’s External Profile information at the 3rd party system, supplying any necessary identification/ access parameters.

Step 4: The CAB Server gets notified a change in CAB User’s External Profile information at the 3rd party system and transforms the received data into the CAB Format.

Different approaches are possible to implement steps 3 and 4, namely using:

- Proprietary mechanisms e.g. via proprietary APIs exposed by the 3<sup>rd</sup> party systems
- Native CAB Enabler procedures e.g. if 3<sup>rd</sup> party systems support receiving native CAB-cross domain PCC subscription requests as described in [CAB 1.1 TS] section 5.2 “*Contact Subscription Function*”.
- CAB APIs e.g. if 3<sup>rd</sup> party system exposes CAB APIs as described in [CAB APIs] sections 6.13, 6.14, 6.15 and H.3

Note: the approaches outlined in the preceding bullets are out of scope of the CAB Enabler.

Step 5: The CAB Server stores the updates and the meta-information of updated PCC Document containing the update type set as ‘subscribe-profile’ in the PCC Document of the PCC Application Usage [CAB 1.1 XDMS] subject to CAB User’s preferences. This step assumes automatic update.

Step 6: CAB Server performs an XCAP PUT using XDM-4i containing the result of subscribe profile operation in the <response> element of the CAB Feature Handler Document in the CAB XDMS.

Step 7: CAB Feature Handler Application Usage from the CAB XDMS responds with 200 OK back to CAB Server using XDM-4i interface.

Step 8: In case the CAB User’s CAB Client is subscribed to changes in the PCC Application Usage then it is notified of the change(s) to the PCC by either of the following methods:

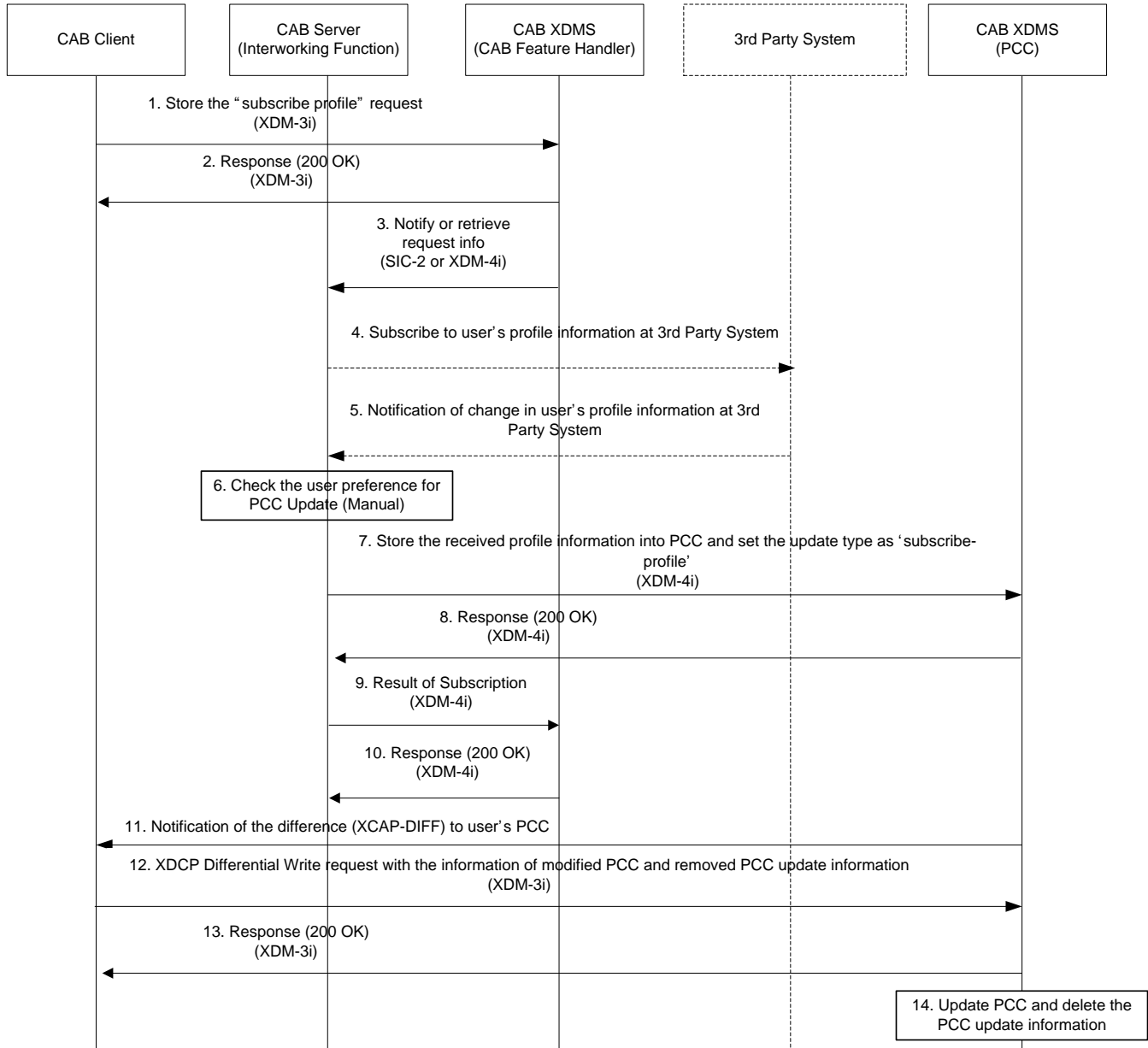
Step 8a: CAB Client receives a SIP NOTIFY assuming a prior subscription to CAB User’s own PCC Document changes using SIP-based subscription.

Step 8b: CAB Client receives a Push OTA Message assuming a prior subscription to CAB User’s own PCC Document changes using XDCP-based subscription.

Note: Step 4, 5 and 8 are repeated every time there is a subsequent change to be notified in the CAB User’s External Profile at the 3<sup>rd</sup> party system.

## C.10.2 Manual approval of Subscription to External Profile Information

The following provides basic message flows for the subscription to changes in the user’s External Profile at a 3<sup>rd</sup> party system in case the CAB User sets his/her preference as ‘manual’ i.e. manual approval of PCC changes .



Step 1: The CAB Client makes a “subscribe profile” request by writing/storing the request information formatted to the <subscribe-profile> element as described in the CAB Feature Handler Application Usage [CAB 1.1 XDMs].

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

Step 3: The CAB Server retrieves the “subscribe profile” request information from the CAB Feature Handler Application Usage by either of the following methods:

Step 3a: CAB Feature Handler Application Usage notifies a change to the CAB Feature Handler Document to the CAB Server assuming a prior subscription to the changes is in place by the CAB Server.

Step 3b: CAB Feature Handler Application Usage is polled by the CAB Server for changes in the CAB Feature Handler Document based on a pre-determined time interval using XCAP.

Step 4: The CAB Server (i.e. Interworking Function) on behalf of the CAB User subscribes to the CAB User’s External Profile information at the 3rd party system, supplying any necessary identification/ access parameters.

Step 5: The CAB Server gets notified a change in CAB User’s External Profile information at the 3rd party system and transforms the received data into the CAB Format.

Different approaches are possible to implement steps 4 and 5, namely using:

- Proprietary mechanisms e.g. via proprietary APIs exposed by the 3<sup>rd</sup> party systems
- Native CAB Enabler procedures e.g. if 3<sup>rd</sup> party systems support receiving native CAB-cross domain PCC subscription requests as described in [CAB 1.1 TS] section 5.2 “*Contact Subscription Function*”.
- CAB APIs e.g. if 3<sup>rd</sup> party system exposes CAB APIs as described in [CAB APIs] sections 6.13, 6.14, 6.15 and H.3

Note: the approaches outlined in the preceding bullets are out of scope of the CAB Enabler.

Step 6; The CAB Server checks the user preference for the PCC update by External Profile import.

Step 7: The CAB Server stores the resulting changes in the PCC Update Document, and its reference and the update type set as ‘subscribe-profile’ are stored in the PCC Document of the PCC Application Usage as described in [CAB 1.1 XDMS].

Step 8: PCC Application Usage sends the response back to CAB Server using XDM-4i interface.

Step 9: CAB Server performs an XCAP PUT using XDM-4i containing the result of subscribe profile operation in the <response> element of the CAB Feature Handler Document in the CAB XDMS.

Step 10: CAB Feature Handler Application Usage from the CAB XDMS responds with 200 OK back to CAB Server using XDM-4i interface.

Step 11: In case the CAB User’s CAB Client is subscribed to changes in the PCC Application Usage then it is notified of the change in the PCC by either of the following methods:

Step 11a: CAB Client receives a SIP NOTIFY assuming a prior subscription to CAB User’s own PCC Document changes using SIP-based subscription.

Step 11b: CAB Client receives a Push OTA Message assuming a prior subscription to CAB User’s own PCC Document changes using XDCCP-based subscription.

Step 12: The CAB Client presents the proposed changes to the CAB User. The CAB User selects which changes to PCC he/she wants to approve. The CAB Client removes unwanted changes to the PCC by removing corresponding patches and sends a Differential Write request with the information of modified PCC and PCC update reference to be removed.

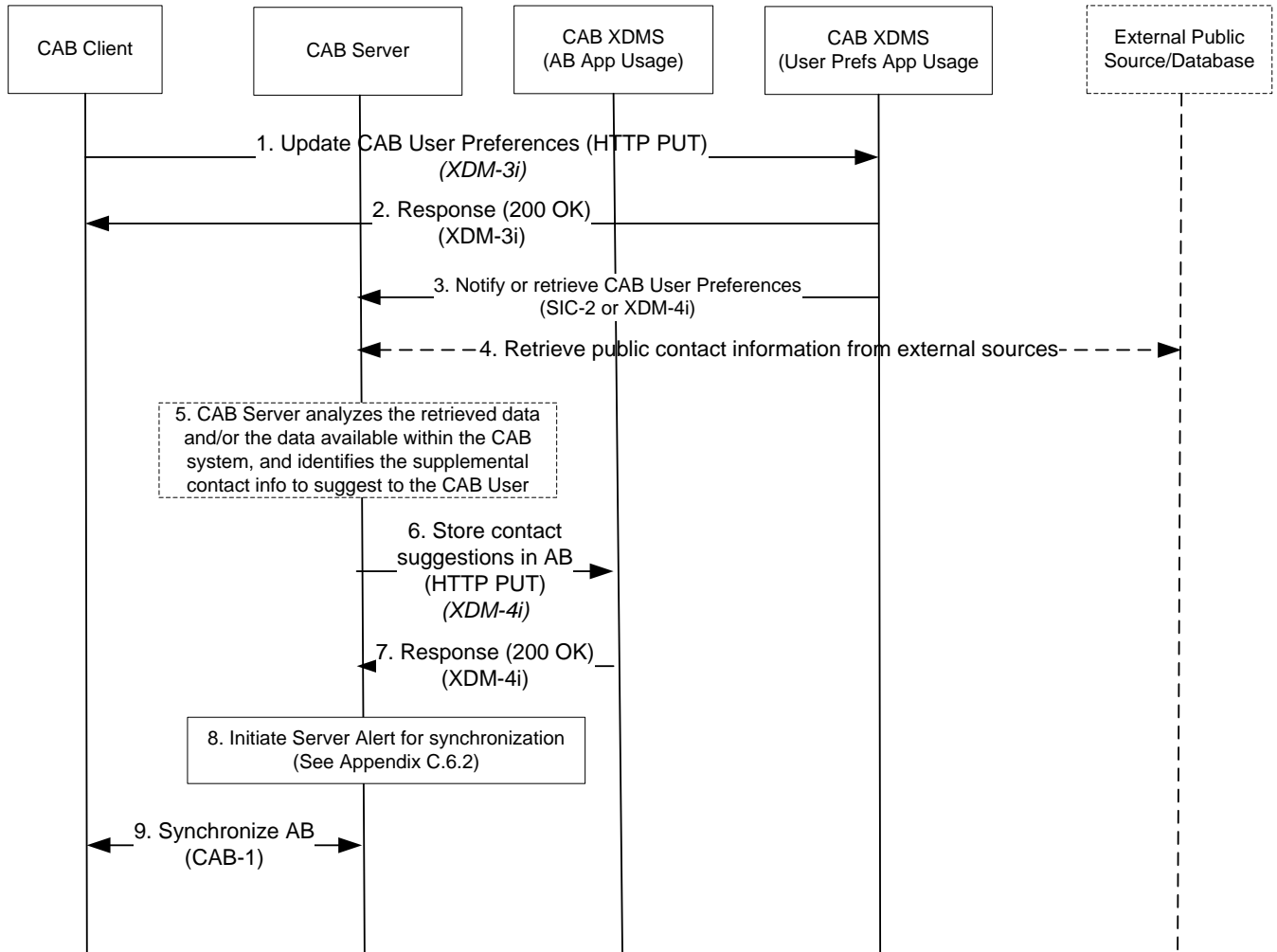
Step 13: PCC Application Usage sends the response back to CAB Client using XDM-3i interface.

Step 14: PCC Application Usage updates PCC Document and delete the PCC Update Document by using PCC update reference.

## C.11 Suggestion of contact information

### C.11.1 Suggestion of contact information (via User Preferences)

The following illustrates a message flow and interactions for suggestion of contact information to the CAB user based on enabling the CAB User Preferences i.e. <allow-suggested-contact-info> element.



**Figure 24 : Suggestion of contact information (via User Preferences)**

Step 1: The CAB Client sends a HTTP PUT request over XDM-3i to update the CAB User Preferences XML document to enable the <allow-suggested-contact-info> preference.

```

PUT /org.openmobilealliance.cab-user-prefs/users/sip:joebloggs@example.com/index /~/cab-user-prefs
/cap-upp HTTP/1.1
Host: xcab.example.com
...
Content-Type: application/xcap-el+xml; charset="utf-8"
Content-Length: (...)

<cab-upp>
<cab-upp-set>
  <profile id='1234'>
    <display-name> CAB UPP of Joe </display-name>
    <allow-suggested-contact-info>true</allow-suggested-contact-info>
  </profile>
</cab-upp-set>
    
```

```
</cab-upp>
```

Step 2: The HTTP “200 OK” message is received by the CAB Client.

Step 3: Assuming the CAB Server has established a subscription to changes to CAB User Preferences XML document, a SIP NOTIFY message is received by the CAB Server (Contact Status Function) using SIC-2 interface.

Note: Alternatively, the CAB Server can poll for changes to the CAB User Preferences XML document via XDM-4i

Steps 4 & 5: Upon successfully receiving the CAB User Preferences for <allow-suggested-contact-info>, the CAB Server retrieves external public contact information and/or analyzes the contact information available within the CAB system, to identify the contact suggestions to be provided to the CAB Client/User.

Note: These steps (i.e. 4 and 5) are out of the scope of this specification

Step 6: The CAB Server sends a HTTP PUT message via XDM-4i to store the contact suggestions in the CAB User’s AB XML document, by creating a new <contact> element and setting the value of <temporary> element under <entry-status> element to “suggested”, and populating the available contact information.

```
PUT /org.openmobilealliance.cab-address-book/users/sip:joebloggs@example.com/index /~/cab-address-
book/address-book HTTP/1.1
Host: xcap.example.com
...
Content-Type: application/xcap-el+xml; charset="utf-8"
Content-Length: (...)

<address-book>
  <contact>
    <contact-status>
      <contact-source>Yellow Pages</contact-source>
      <entry-status>
        <temporary>suggested</temporary>
      <entry-status>
    <contact-status>
    <person-details index="gt4fd890bu8">
      <name>
        <name-entry index="riuetut1" pref="2" xml:lang="en" name-type="KnownAs">
          <given>John</given>
          <family>Doe</family>
          <display-name>John Doe</display-name>
        </name-entry>
      </name>
      <address>
      <comm-addr xml:lang="en">
        <tel index="nbvfjf" pref="2" tel-type="Mobile">
          <tel-nb>
            <tel-str>1-800-555-1111</tel-str>
          </tel-nb>
          <label>Personal Phone</label>
        </tel>
      </comm-addr>
      <person-details index="gt4fd890bu8">
    </contact>
  </address-book>
```

Step 7: The HTTP “200 OK” message is received by the CAB Server.

Step 8: 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 “Address Book Modifications from Network”.



Step 9: Following this, the CAB Client may initiate synchronization as described in [CAB TS] Appendix C.6.1 “CAB Client Address Book Modifications and Synchronization”. The CAB Client may either accept or reject the suggested contact information. In the case of the CAB User accepts the suggested contact, the CAB Client removes the <temporary> element associated with the contact.

### C.11.2 Suggestion of contact information (via FH App Usage)

The following illustrates a message flow and interactions for suggestion of contact information to the CAB user based on CAB Client request to the CAB Server i.e. using the <contact-suggest> element under the FH App Usage.

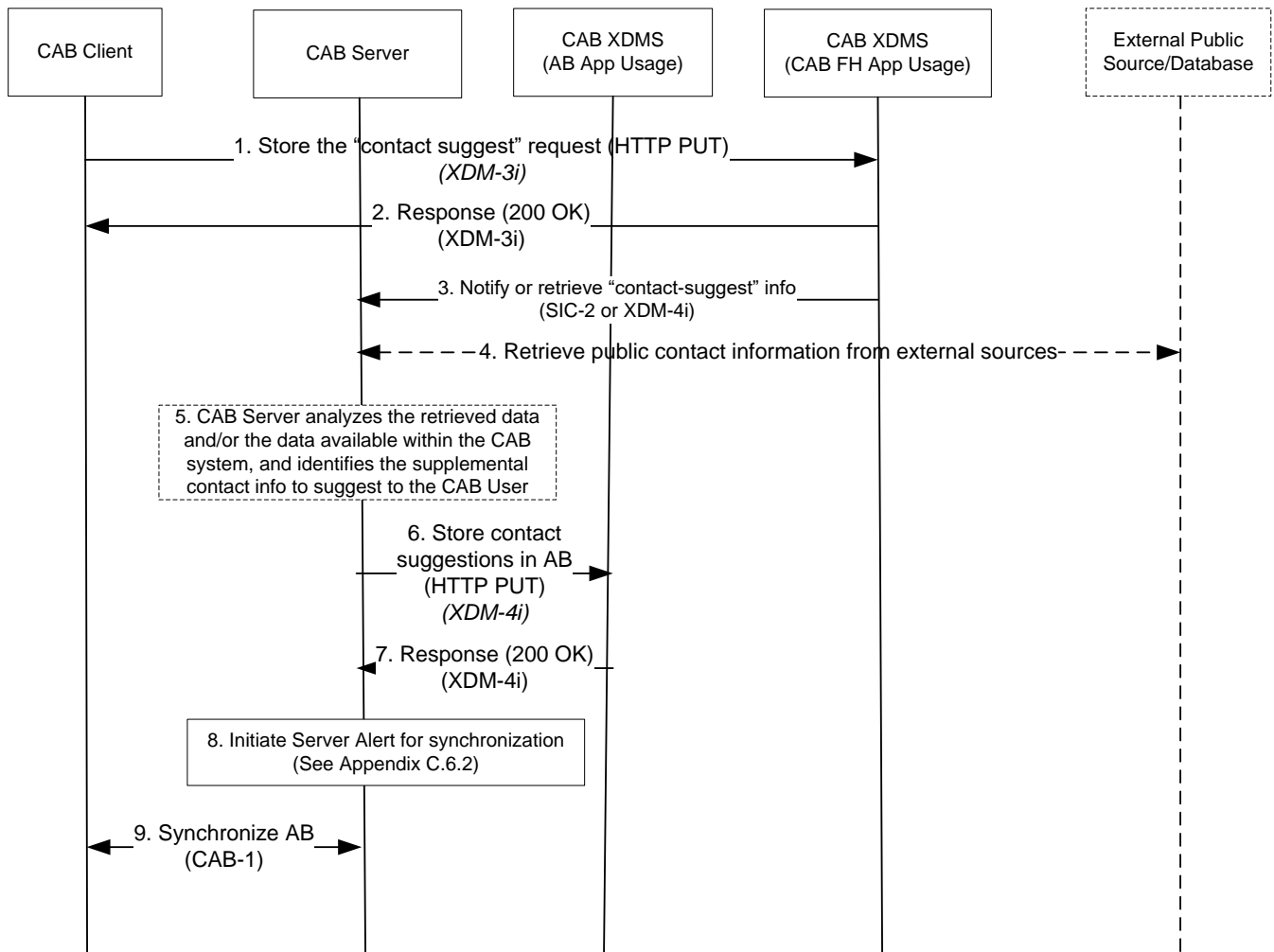


Figure 25 : Suggestion of contact information (via FH App Usage)

Step 1: The CAB Client stores the ‘contact suggest’ request in the CAB FH Document via HTTP PUT/XDM-3i

```

PUT /org.openmobilealliance.cab-feature-handler/users/sip:joebloggs@example.com/index/~/cab-
feature-handler/feature-handler HTTP/1.1
Host: xcap.example.com
    
```

```

...
Content-Type: application/xcap-el+xml; charset="utf-8"
Content-Length: (...)

<feature-handler>
  <cab-feature id="3456">
    <contact-suggest>
      <non-CAB-source>suggestions.example.com</non-CAB-source>
      <scheduled-interval>0</scheduled-interval>
      <criteria>soccer</criteria>
      <credentials>
        <username>username</username>
        <password>*****</password>
      </credentials>
    </contact-suggest>
  </cab-feature>
</feature-handler>

```

Step 2: The HTTP “200 OK” message is received by the CAB Client.

Step 3: Assuming the CAB Server has established a subscription to changes to CAB FH App Usage, a SIP NOTIFY message is received by the CAB Server (Contact Status Function) using SIC-2 interface.

Note: Alternatively, the CAB Server can poll for changes to the CAB User Preferences XML document via XDM-4i

Step 4: The CAB Server requests/retrieves external contact information based upon the contact suggest request information.

Step 5 : The CAB Server analyzes the contact information available within the CAB system, to identify the contact suggestions to be provided to the CAB Client/User.

Note: These steps (i.e. 4 and 5) are out of the scope of this specification

Step 6: The CAB Server sends a HTTP PUT message via XDM-4i to store the contact suggestions in the CAB User’s AB XML document, by creating a new <contact> element and setting the value of <temporary> element under <entry-status> element to “suggested”, and populating the available contact information.

```

PUT /org.openmobilealliance.cab-address-book/users/sip:jobloggs@example.com/index /~/cab-address-
  book/address-book HTTP/1.1
Host: xcap.example.com
...
Content-Type: application/xcap-el+xml; charset="utf-8"
Content-Length: (...)

<address-book>
  <contact>
    <contact-status>
      <contact-source>Yellow Pages</contact-source>
      <entry-status>
        <temporary>suggested</temporary>
      <entry-status>
    </contact-status>
    <person-details index="gt4fd890bu8">
      <name>
        <name-entry index="riuetut1" pref="2" xml:lang="en" name-type="KnownAs">
          <given>John</given>
          <family>Doe</family>
          <display-name>John Doe</display-name>
        </name-entry>
      </name>
      <address>
        <comm-addr xml:lang="en">
          <tel index="nbvfjf" pref="2" tel-type="Mobile">
            <tel-nb>
              <tel-str>1-800-555-1111</tel-str>
            </tel-nb>
          </comm-addr>
        </address>
      </person-details>
    </contact>
  </address-book>

```

```
        </tel-nb>
        <label>Personal Phone</label>
    </tel>
</comm-addr>
<person-details index="gt4fd890bu8">
</contact>
</address-book>
```

Step 7: The HTTP “200 OK” message is received by the CAB Server.

Step 8: 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 “*Address Book Modifications from Network*”.

Step 9: Following this, the CAB Client may initiate synchronization as described in [CAB TS] Appendix C.6.1 “*CAB Client Address Book Modifications and Synchronization*”. The CAB Client may either accept or reject the suggested contact information. In the case of the CAB User accepts the suggested contact, the CAB Client removes the <temporary> element associated with the contact.

## C.12 Subscription Invitation Flow

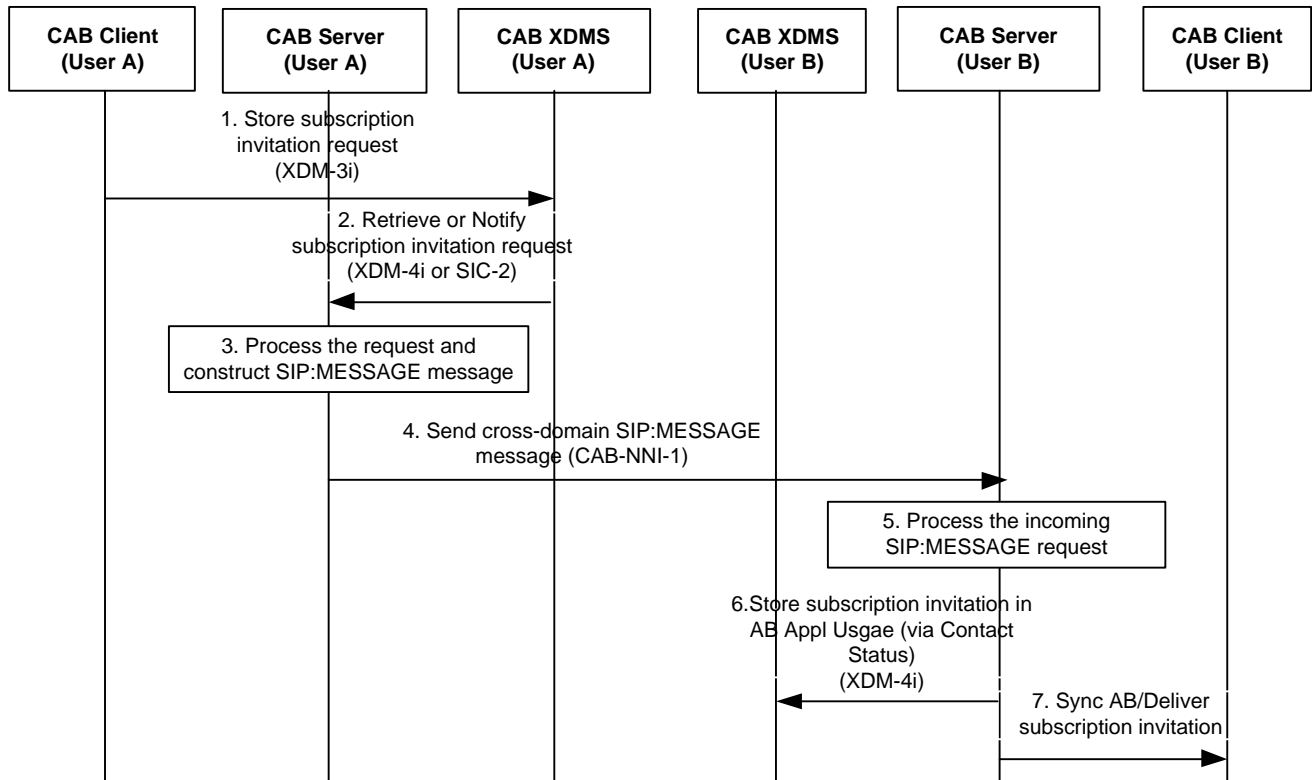
The following provides Subscription Invitation message flows through two options:

- 1) Subscription Invitation using [SIP:MESSAGE](#)
- 2) Subscription Invitation using XDM Forward

The flows demonstrate how a CAB User (A) invites another CAB User (B) to subscribe to his/her Published Contact Card, and assumes that CAB User A and B belong to different domains

### C.12.1 Subscription Invitation using SIP:Message

This flow describes the Subscription Invitation functionality using SIP:MESSAGE.



**Figure 26 : Subscription Invitation using SIP:Message**

Step 1: The CAB Client of User A stores the subscription invitation request (e.g. details may include address of CAB User B, access permissions, Contact View to be sent with the invitation, invitation text, etc.) in the CAB Feature Handler App Usage [CAB 1.1 XDMs] using XDM-3i interface.

Step2: The CAB Server retrieves the invitation request from the CAB Feature Handler App Usage either through receiving a notification (SIC-2) or accessing the request information through XCAP (XDM-4i).

Step 3: Upon receiving the request, the CAB Server validates and processes the invitation request (e.g. validate the recipient, modify access permissions), and then constructs a SIP:MESSAGE message with the appropriate routing headers and invitation content (e.g. to include CAB User A’s published contact card, the invitation message).

Step 4: CAB Server (of CAB User A) sends the cross-domain SIP:MESSAGE message to CAB Server (of CAB User B), with the subscription invitation information, using NNI-1 interface.

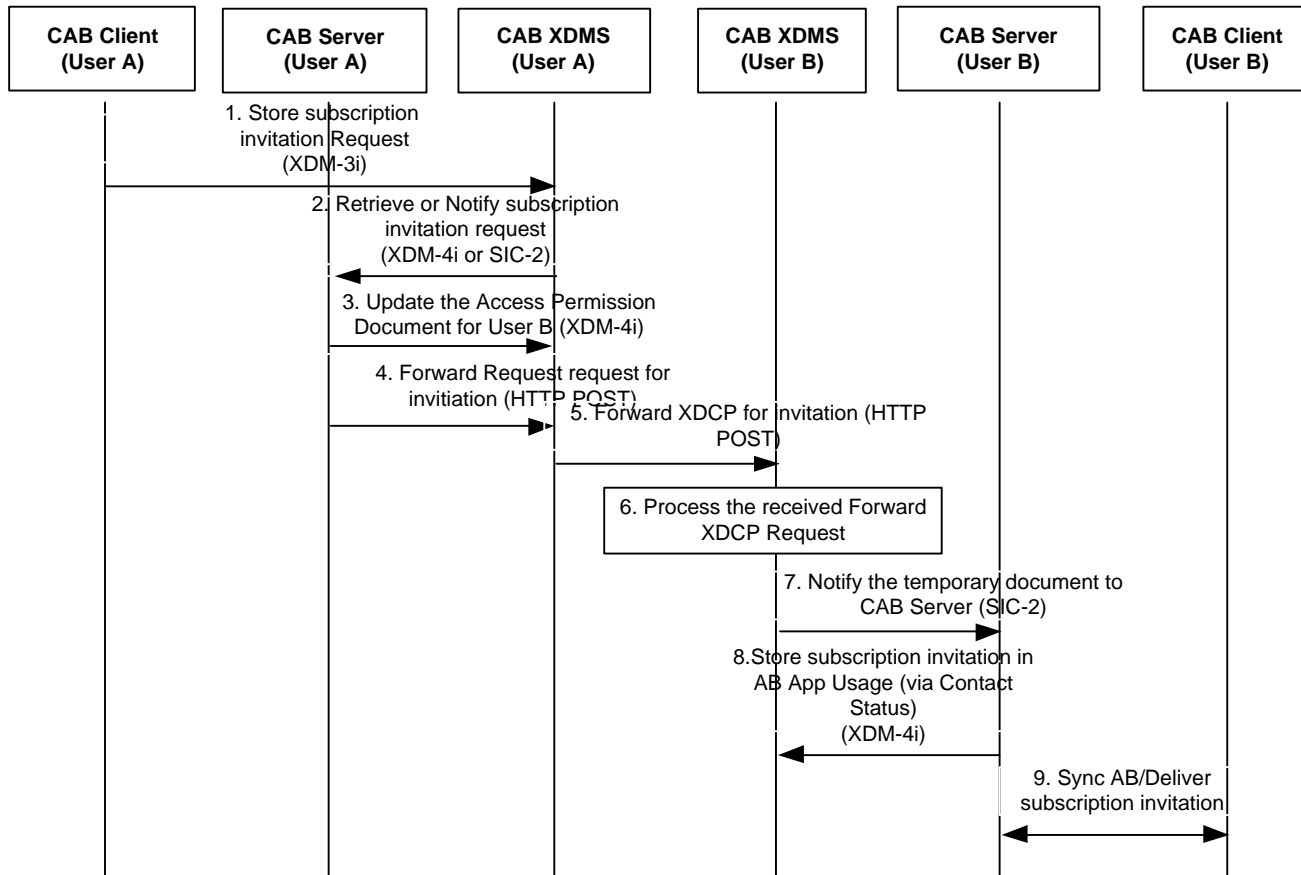
Step 5: The CAB Server (of CAB User B) processes the incoming SIP:MESSAGE message e.g. parse the invitation content, and check the User Preferences of CAB User B on whether or not to notify invitations to Contact Subscription).

Step 6: If CAB User B sets the preferences to receive Contact Subscription invitation notifications, the CAB Server creates a Contact card (either a temporary card with a reference to an existing card or no reference to an existing card), and sets the Contact Status of the contact entry to type “Subscription Invitation” to indicate that CAB User A has invited CAB User B to subscribe to his/her PCC. The temporary contact card includes the available or published information (or contact view) of CAB User A.

Step 7: The CAB User B’s address book with the Contact Status updates are delivered to the CAB Client of CAB User B through synchronization process (i.e. either through OMA DS or OMA XDM), by which the invitation gets delivered to CAB User B.

### C.12.2 Subscription Invitation using XDM Forward

This flow describes the Subscription Invitation functionality using XDM Forward.



**Figure 27 : Subscription Invitation using XDM Forward**

Step 1: The CAB Client of User A stores the subscription invitation request (e.g. details may include address of CAB User B, access permissions, Contact View to be sent with the invitation, invitation text, etc) in the CAB Feature Handler App Usage [CAB 1.1 XDMS] using XDM-3i interface.

Step2: The CAB Server retrieves the invitation request from the CAB Feature Handler App Usage either through receiving a notification (SIC-2) or accessing the request information through XCAP (XDM-4i).

Step 3: The CAB Server updates access permission for CAB User B’s subscription.

Step 4: The CAB Server sends XDCP Forward request as described in sub-clause 5.3.1 “*Procedueres at the Originating Side*” of section 5.3 “*Contact Share Function*” for invitation.

Step 5: The CAB XDMS generates Forward XDCP request for invitation and sends it to the receiving side.

Step 6: The CAB XDMS of User B process the received Forward XDCP request. (e.g. checking the XDM Preferences and fetching the corresponding document)

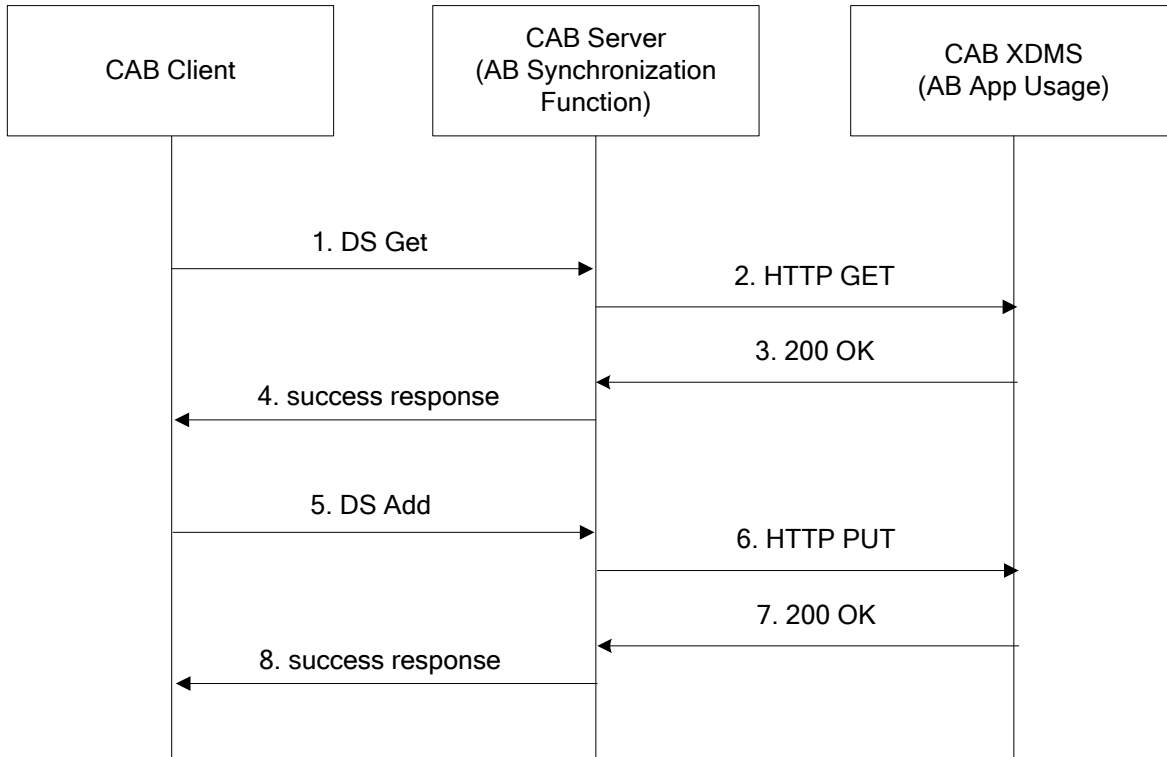
Step 7: The CAB XDMS notifies the temporary document to CAB Server.

Step 8: The CAB Server process it as described in sub-clause 5.10.2.2 “*Using XDCP Forward*” to store the contact entry with the entry status if it regards as the invitation request for subscription invitation.

Step 9: The CAB User B’s address book with the Contact Status updates are delivered to the CAB Client of CAB User B through synchronization process (i.e. either through OMA DS or OMA XDM), by which the invitation gets delivered to CAB User B.

## C.13 Retrieval of deleted contact information

The following provides the flow for retrieval of deleted contact information from the network AB repository. In this section of the flow, it shows only the way of using DS mechanism. It is assumed that the authentication between the CAB Client and the CAB Server has been performed successfully.



**Figure 28 : Retrieval of deleted contact information**

Step 1: With the request from the CAB User to retrieve the deleted contact information, the CAB Client sends DS Get request to the AB Synchronization Function of the CAB Server with the filter criteria indicating the deleted contact information.

Step 2: The AB Synchronization Function of the CAB Server sends HTTP GET request towards the AB App Usage via the XDM Agent to receive the deleted contact information.

Step 3: The AB App Usage returns one or more deleted contact information via the reverse path of the corresponding request.

Step 4: The AB Synchronization Function of the CAB Server returns one or more deleted contact information to the CAB Client.

Step 5: With the request from the CAB User to restore the deleted contact information, the CAB Client sends DS Add request to the AB Synchronization Function of the CAB Server with the contact information to be added.

Step 6: The AB Synchronization Function of the CAB Server sends HTTP PUT request towards the AB App Usage via the XDM Agent to restore the deleted contact information in the AB App Usage.

Step 7: The AB App Usage responds with the successful response.

Step 8: The AB Synchronization Function responds with the successful response.

NOTE: In case the local device maintains the deleted contact information, the CAB Client may not need to fetch the deleted contact information from AB App Usage based on the service provider policy. The mechanism to view the deleted contact list and to restore one or more contact information within the local device is out of scope of this specification.

## C.14 Common Connections

The following illustrates a message flow and interactions for identifying Common Connections in CAB User's AB, based on enabling the CAB User Preferences i.e. <notify-common-connections> element.

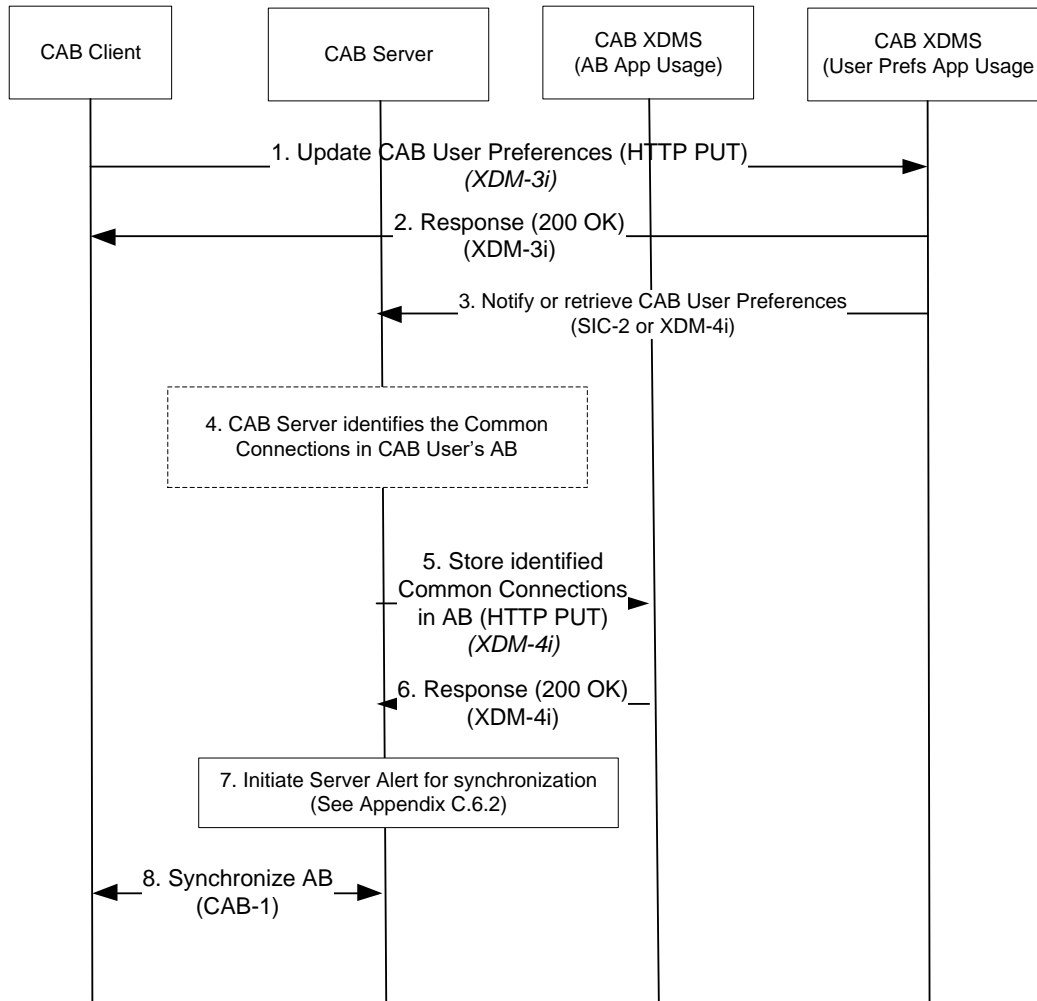


Figure 29: Common Connections

Step 1: The CAB Client sends a HTTP PUT request over XDM-3i to update the CAB User Preferences XML document to enable the <notify-common-connections> preference.

```

PUT /org.openmobilealliance.cab-user-prefs/users/sip:joebloggs@example.com/index/~/cab-user-prefs/cap-upp HTTP/1.1
Host: xcab.example.com
...
Content-Type: application/xcap-el+xml; charset="utf-8"
Content-Length: (...)

<cab-upp>
<cab-upp-set>
  <profile id='1234'>

```

```

        <display-name> CAB UPP of Joe </display-name>
        <notify-common-connections>true</notify-common-connections>
    </profile>
</cab-upp-set>
</cab-upp>

```

Step 2: The HTTP “200 OK” message is received by the CAB Client.

Step 3: Assuming the CAB Server has established a subscription to changes to CAB User Preferences XML document, a SIP NOTIFY message is received by the CAB Server (Contact Status Function) using SIC-2 interface.

Note: Alternatively, the CAB Server can poll for changes to the CAB User Preferences XML document via XDM-4i

Step 4: Upon successfully receiving the CAB User Preferences for <notify-common-connections >, the CAB Server identifies the Common Connections associated with contacts in the CAB User’s AB.

Note: Step 4 is out of the scope of this specification

Step 5: The CAB Server sends a HTTP PUT message via XDM-4i to store the Common Connections information in the CAB User’s AB XML document, by updating or creating a new <common-connections> element under <contact-status> element of the <contact> element and populating the child elements. For procedures, see sub-clause 5.8.5 “*Common Connections*”. In this example, it is shown that a user named “Joe Smith” is a Common Connection between the CAB User (Joe Bloggs) and a contact (with id=“skhuff”) in his/her AB. Joe Smith is a connection of type ‘address book’ and ‘watcher’ in the example shown. For more information refer to <type> element, as described in [CAB 1.1 XDMS] sub-clause 5.1.1.1 “*Structure*” and 5.1.1.7 “*Data Semantics*”.

Note: The <person-details> of contact (with id=“skhuff”) is omitted from this example for simplicity.

```

PUT /org.openmobilealliance.cab-address-book/users/sip:joebloggs@example.com/index/~/cab-address-
book/address-book HTTP/1.1
Host: xcap.example.com
...
Content-Type: application/xcap-el+xml; charset="utf-8"
Content-Length: (...)

<address-book>
  <contact id="skhuff">
    <contact-status>
      <common-connections>
        <connection>
          <display-name>Joe Smith</display-name>
          <XUI>sip:joe.smith@example.com</XUI>
          <type-list>
            <type>address book</type>
            <type>watcher</type>
          </type-list>
        </connection>
      </common-connections>
    </contact-status>
  </contact>
</address-book>

```

Step 6: The HTTP “200 OK” message is received by the CAB Server.

Step 7: 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 “*Address Book Modifications from Network*”.

Step 8: Following this, the CAB Client may initiate synchronization as described in [CAB TS] Appendix C.6.1 “*CAB Client Address Book Modifications and Synchronization*”.

## C.15 Public Content

The following illustrates a message flow and interactions for delivering public content associated with contacts in CAB User’s AB, based on enabling the CAB User Preferences i.e. <notify-public-content> element.



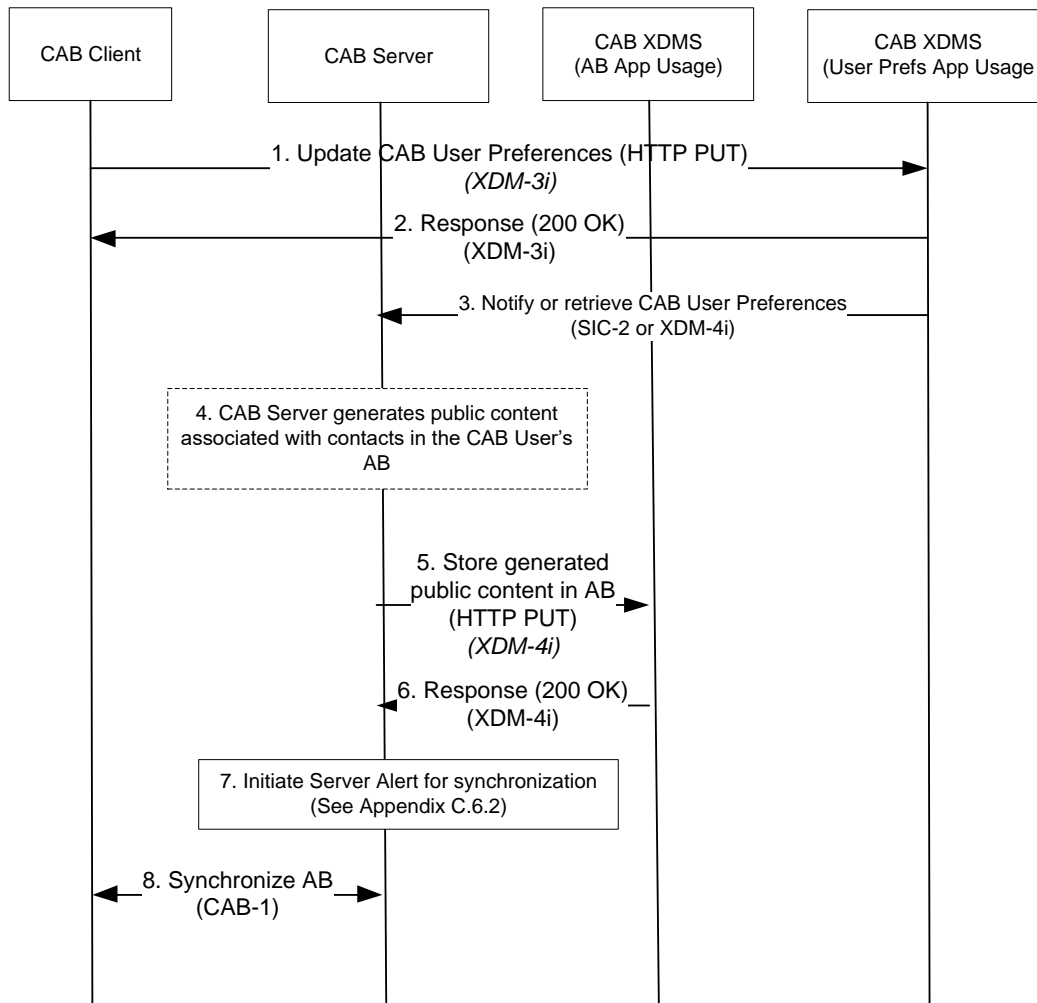


Figure 30: Public Content

Step 1: The CAB Client sends a HTTP PUT request over XDM-3i to update the CAB User Preferences XML document to enable the <notify-public-content> preference.

```

PUT /org.openmobilealliance.cab-user-prefs/users/sip:joebloggs@example.com/index/~/cab-user-prefs/cap-upp HTTP/1.1
Host: xcap.example.com
...
Content-Type: application/xcap-el+xml; charset="utf-8"
Content-Length: (...)

<cab-upp>
<cab-upp-set>
  <profile id='1234'>
    <display-name> CAB UPP of Joe </display-name>
    <notify-public-content>true</notify-public-content>
  </profile>
</cab-upp-set>
</cab-upp>
    
```

Step 2: The HTTP “200 OK” message is received by the CAB Client.

Step 3: Assuming the CAB Server has established a subscription to changes to CAB User Preferences XML document, a SIP NOTIFY message is received by the CAB Server (Contact Status Function) using SIC-2 interface.

Note: Alternatively, the CAB Server can poll for changes to the CAB User Preferences XML document via XDM-4i

Step 4: Upon successfully receiving the CAB User Preferences for <notify-public-content >, the CAB Server generates public content associated with contacts in the CAB User’s AB.

Note: Step 4 is out of the scope of this specification

Step 5: The CAB Server sends a HTTP PUT message via XDM-4i to store the generated public content in the CAB User’s AB XML document, by updating or creating a new <public-content> element under <contact-status> element of the <contact> element and populating the child elements. For procedures, see sub-clause 5.8.8 “*Public content*”. In this example, it is assumed that a user named “Joe Smith” is already present in the CAB User’s (Joe Bloggs) AB with contact id=“oidjfishfd”, and the following snippet adds public content associated with this contact.

```
PUT /org.openmobilealliance.cab-address-book/users/sip:joebloggs@example.com/index/~/cab-address-
book/address-book HTTP/1.1
Host: xcap.example.com
...
Content-Type: application/xcap-el+xml; charset="utf-8"
Content-Length: (...)

<address-book>
  <contact id="oidjfishfd">
    <contact-status>
      <public-content>
        <content-entry>
          <type>blog</type>
          <url>http://example.org/JoeSmith/personal/blog.html</url>
          <title>Joe Smith's Personal Blog</title>
          <time>2012-07-27T14:45:00</time>
        </content-entry>
      </public-content>
    </contact-status>
  </contact>
</address-book>
```

Step 6: The HTTP “200 OK” message is received by the CAB Server.

Step 7: 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 “*Address Book Modifications from Network*”.

Step 8: Following this, the CAB Client may initiate synchronization as described in [CAB TS] Appendix C.6.1 “*CAB Client Address Book Modifications and Synchronization*”.