# Client-Server Protocol Plain Text Syntax

Candidate Version 1.3 – 11 Oct 2005

**Open Mobile Alliance**

OMA-TS-IMPS-CSP_PTS-V1_3-20051011-C

# Contents

# Tables

# 1. Scope

The Instant Messaging and Presence Service (IMPS) includes four primary features:

- Presence
- Instant Messaging
- Groups
- Shared Content

Presence is the key enabling technology for IMPS. It includes client device availability (my phone is on/off, in a call), user status (available, unavailable, in a meeting), location, client device capabilities (voice, text, GPRS, multimedia) and searchable personal statuses such as mood (happy, angry) and hobbies (football, fishing, computing, dancing). Since presence information is personal, it is only made available according to the user's wishes - access control features put the control of the user presence information in the users' hands.

Instant Messaging (IM) is a familiar concept in both the mobile and desktop worlds. Desktop IM clients, two-way SMS and two-way paging are all forms of Instant Messaging. Wireless Village IM will enable interoperable mobile IM in concert with other innovative features to provide an enhanced user experience.

Groups or chat are a fun and familiar concept on the Internet. Both operators and end-users are able to create and manage groups. Users can invite their friends and family to chat in group discussions. Operators can build common interest groups where end-users can meet each other online.

Shared Content allows users and operators to setup their own storage area where they can post pictures, music and other multimedia content while enabling the sharing with other individuals and groups in an IM or chat session.

These features, taken in part or as a whole, provide the basis for innovative new services that build upon a common interoperable framework.

# 2. References

## 2.1 Normative References


**[CSP]** "Client-Server Protocol Session and Transactions Version 1.3", OMA-TS-IMPS-CSP-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[CSP Trans]** "Client-Server Protocol Transport Bindings Version 1.3", OMA-TS-IMPS-CSP_Transport-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[IOPPROC]** "OMA Interoperability Policy and Process Version 1.1", OMA-IOP-Process-V1_1. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

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


## 2.2 Informative References


**[Arch]** "IMPS Architecture Version 1.3", OMA-AD-IMPS-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[CSP]** "Client-Server Protocol Session and Transactions Version 1.3", OMA-TS-IMPS-CSP-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[CSP XMLS]** "Client-Server Protocol XML Syntax Version 1.3". OMA-TS-IMPS-CSP-XMLS-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[CSP Trans]** "Client-Server Protocol Transport Bindings Version 1.3", OMA-TS-IMPS-CSP_Transport-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[CSP DataType]** "Client-Server Protocol Data Types Version 1.3", OMA-TS-IMPS-CSP_Data_Types-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[CSP PTS]** "Client-Server Protocol Plain Text Syntax Version 1.3", OMA-TS-IMPS-CSP_PTS-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[CSP WBXML]** "Client-Server Protocol Binary XML Definition and Examples Version 1.3", OMA-TS-IMPS-CSP_WBXML-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[PA]** "Presence Attributes Version 1.3", OMA-TS-IMPS-PA-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[PA XMLS]** "Presence Attribute XML Syntax Version 1.3", OMA-TS-IMPS-PA_XMLS-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[SSP]** "Server-Server Protocol Semantics Version 1.3", OMA-TS-IMPS-SSP-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[SSP Syntax]** "Server-Server Protocol XML Syntax Version 1.3", OMA-TS-IMPS-SSP_XMLS. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

**[SSP Trans]** "Server-Server Protocol Transport Binding Version 1.3", OMA-TS-IMPS-SSP_Transport-V1_3. Open Mobile Alliance™. URL: http://www.openmobilealliance.org

# 3. Terminology and Conventions

## 3.1 Conventions

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

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

## 3.2 Definitions

None.

## 3.3 Abbreviations

| | |
|---|---|
| **OMA** | Open Mobile Alliance |
| **WV** | Wireless Village |

# 4. Introduction

This document describes the encoding of primitives, information elements and the contained values in plain text format. Plain text format MAY be used over WSP, HTTP(S) and SMS transports. Please refer to [CSP Trans] for transport - related issues.

The message flows are defined in [CSP].

The enumerated values are as specified in [CSP].

# 5. Plain Text Syntax

The general plain text message format consists of preamble and parameters as follows:

*WVaaBBcccDD* <parameters> [&] where:

- **WV** indicates that this is a Wireless Village or IMPS message. It is case sensitive.
- **aa** is the version number of the OMA IMPS specification being used. Major version and minor version numbers without the dot in the middle, both are single digits from 0 to 9. This specification uses version number 1.3 (13 in messages).
  There are two exceptions – the VersionDiscovery request and response primitives. These two primitives are independent of the specification versions. The 'aa' part in these primitives is always 'XX' (not case sensitive).
- **BB** is the message type, identified by a two-letter code. It is not case sensitive.
- The **ccc** is the Transaction-ID in range 0-999 without preceding zero. The server initiated Disconnect primitive MUST NOT contain it.
- The **DD** is an SMS transport-specific value called concatenation identifier. It identifies multiple short messages within a single transaction: it is a two-letter identifier within the range of a-z ('a' being the first). The first letter indicates the number (position) of the message; the second letter indicates the total number of parts. (This gives the possibility to split a message into 26 SMSes, which limits the size of the message to 26*160=4160 characters (in case of 7bits/character). Note that it includes every additional information elements such as tags, commas and so on, so not the whole capacity is available to the user.) It is not case sensitive. Please refer to "Transaction over Multiple Short Messages" in [CSP Transport] for the syntax.
  The **DD** part is omitted when the Plain Text Syntax is used over HTTP transport, as splitting does not take place – the primitives according to Plain Text Syntax are sent over as a whole since the 160 character boundary does not exist.
- The ampersand (&) parameter separates multiple IMPS messages within plain text message(s). The separating ampersand does not follow the last IMPS message in the plain text message.

The general format of parameters with value SHALL be:

<name>=<value>

Example: CI=http://123.123.123.123:80/IMPSAPP

The general format of parameters without value SHALL be:

<name> Note the missing equal sign and value.

Example: PS

After the *aaBBcccDD* preamble, there is one space character. Each parameter is separated from other parameters by single space character. Every parameter is defined and used with its own unique two-letter code. Unlike in the DTD, the parameters MAY be in any order in a primitive. However with the exception of the ExtendedRequest/Response transactions, which are described in 8.2 - Introducing New Primitives on page 40.

If the value of the parameter contains spaces ( ), quotes (”), commas (,),parentheses (()),equal (=) or ampersand (&) characters, it SHALL be wrapped with quotes (“).

If the value contains quotes (“), all quote characters SHALL be doubled. For example if the value is:

John “Johnnie” Smith

then it SHALL be encoded as:

“John “”Johnnie”” Smith”

If the value is only one single quote, the encoding would be four quotes: “””””

Each parameter MAY be present only once in a primitive, so a parameter MAY also contain a list of values. In this case, the syntax SHALL be (example with three values):

<name>=(<value1>,<value2>,<value3>)

Single values MUST NOT be wrapped with parentheses, unless the syntax explicitly specifies that (screen name for example). Please refer to chapter 7.11 Information Element-specific Encoding on page 28 for exceptions.

A parameter MAY also contain a group of values. These groups MAY be also nested. Examples:

one value having three values (a parent having 3 children):

<name>=(<value1>,(<value2>,<value3>,<value4>))

nested values (root node has a single child, which has two children):

<name>=(<value1>,(<value2>,(<value3>,<value4>)))

Definitions SHALL be case-sensitive.

## 5.1 Transport Bindings

As defined in the [CSP Trans] document the CSP Plain Text Syntax can be used over SMS transport. The textual SMS messages (not encoded using a User Data Header) MAY also be used over a HTTP(S) and WSP based transport as defined in [CSP Trans].

# 6. Availability of Transactions

The SMS protocol itself provides some features and lacks others, which limits or disables certain transactions due to content type limitation (only text/plain) and delivery method (only "Push"). The list of supported transactions is listed in the table below. Transactions that are marked:

- "Full" can be utilized completely.
- "Limited" cannot be completely utilized due to the limitations originating from the SMS transport.
- "N/A" cannot be used at all.

WSP and HTTP(S) transports are not limited in such a manner – all of the transactions can be utilized, however all content types except text/plain MUST be encoded using an agreed transfer-encoding method ,such as 'BASE64'.

All instant messages that contain plain text message content and fit into the limits specified in 5 - Plain Text Syntax without truncation SHOULD be pushed to the terminal.

The following table describes the availability of each primitive defined in [CSP]:

| Transaction | SMS transport support | Code |
|---|---|---|
| AddGroupMembersRequest | Full | AM |
| BlockEntityRequest | Full | BE |
| CancelInviteRequest | Full | CI |
| CancelInviteUserRequest | Full | CU |
| ClientCapabilityRequest | Limited | CP |
| ClientCapabilityResponse | Limited | PC |
| CreateAttributeListRequest | Full | CA |
| CreateGroupRequest | Full | CG |
| CreateListRequest | Full | CL |
| CreateListResponse | Full | LC |
| DeleteAttributeListRequest | Full | DA |
| DeleteGroupRequest | Full | DG |
| DeleteListRequest | Full | DL |
| DeliveryReportRequest | Full | DR |
| Disconnect | Full | DI |
| DropSegmentRequest | Full | DS |
| ExtendConversionRequest | Full | EC |
| ExtendConversionResponse | Full | CE |
| ExtendedRequest | Full | XR |
| ExtendedResponse | Full | RX |
| ForwardMessageRequest | N/A | FW |
| ForwardMessageResponse | N/A | WF |
| GetAttributeListRequest | Full | GA |
| GetAttributeListResponse | Full | AG |
| GetBlockedListRequest | Full | GB |
| GetBlockedListResponse | Full | BG |
| GetGroupMembersRequest | Full | GM |
| GetGroupMembersResponse | Full | MG |
| GetGroupPropsRequest | Full | GR |
| GetGroupPropsResponse | Full | RG |
| GetJoinedUsersRequest | Full | JU |
| GetJoinedUsersResponse | Full | UJ |
| GetListRequest | Full | GL |
| GetListResponse | Full | LG |

| Transaction | SMS transport support | Code |
|---|---|---|
| GetMapRequest | N/A | GD |
| GetMapResponse | N/A | DG |
| GetMessageListRequest | Limited | MR |
| GetMessageListResponse | Limited | RM |
| GetMessageRequest | Limited | GX |
| GetMessageResponse | Limited | MX |
| GetPresenceRequest | Limited | GP |
| GetPresenceResponse | Limited | PG |
| GetPublicProfileRequest | Full | GU |
| GetPublicProfileResponse | Limited | UG |
| GetSegmentRequest | Full | GE |
| GetSegmentResponse | Full | EG |
| GetSPInfoRequest | Full | GS |
| GetSPInfoResponse | Limited | SG |
| GetWatcherListRequest | Full | GW |
| GetWatcherListResponse | Full | WG |
| GroupChangeNotice | Full | GG |
| InviteRequest | Full | IR |
| InviteResponse | Full | RI |
| InviteUserRequest | Full | IU |
| InviteUserResponse | Full | UI |
| JoinGroupRequest | Full | JG |
| JoinGroupResponse | Full | GJ |
| KeepAliveRequest | Full | KA |
| KeepAliveResponse | Full | AK |
| LeaveGroupRequest | Full | LU |
| LeaveGroupResponse | Full | UL |
| ListManageRequest | Full | LM |
| ListManageResponse | Full | ML |
| LoginRequest | Full | LR |
| LoginResponse | Full | RL |
| LogoutRequest | Full | OR |
| MemberAccessRequest | Full | ME |
| MessageDelivered | Full | MD |
| MessageNotification | N/A | MN |
| NewMessage | Limited | NM |
| NotificationRequest | Full | NR |
| PollingRequest | Full | PO |
| PresenceNotificationRequest | Limited | PN |
| RemoveGroupMembersRequest | Full | RM |
| RejectListRequest | Full | RE |
| RejectListResponse | Full | ER |
| RejectMessageRequest | N/A | RR |
| SearchRequest | Full | SR |
| SearchResponse | Full | RS |
| SendMessageRequest | Limited | SM |
| SendMessageResponse | Full | MS |
| ServiceRequest | Limited | SQ |
| ServiceResponse | Limited | QS |

| Transaction | SMS transport support | Code |
|---|---|---|
| SetDeliveryMethodRequest | Limited | SD |
| SetGroupPropsRequest | Full | SP |
| Status | Full | ST |
| StopSearchRequest | Full | SS |
| SubscribeGroupNoticeRequest | Full | SU |
| SubscribeGroupNoticeResponse | Full | US |
| SubscribeNotificationRequest | Full | SN |
| SubscribePresenceRequest | Full | SB |
| SystemMessage-Request | Limited | SY |
| SystemMessage-User | Full | YS |
| UnsubscribeNotificationRequest | Full | UN |
| UnsubscribePresenceRequest | Full | PS |
| UpdatePresence | Limited | UP |
| UpdatePublicProfileRequest | Limited | UR |
| VerifyIDRequest | Full | VR |
| WV-CSP-VersionDiscoveryRequest | Full | VD |
| WV-CSP-VersionDiscoveryResponse | Full | DV |

**Table 1: Transaction availability and codes**

The codes MUST NOT becase sensitive.

# 7. Plain Text Syntax-specific Encoding

## 7.1 Information Elements

All information elements have been encoded into shorter codes. Note that in order to be able to differentiate different types of data, some of the information elements have multiple codes based on the type of the data carried within the information element. Examples for this kind of codes are: AdminMapList, Detailed-Result, Recipient, Sender, Block/Unblock Lists, Grant/Ungrant lists, etc.

| Element | Code |
|---|---|
| Acceptance | AC |
| Add-Users-List | AU |
| Add-Nick-List | AN |
| Administrator | AD |
| AdminMapList – AdminMapping | AA |
| AdminMapList – ModMapping | AM |
| AdminMapList – UserMapping | AE |
| Advanced-Criteria | AI |
| Agreed-CapabilityList | AP |
| Application-ID | AT |
| All-Functions | AF |
| All-Functions-Request | AR |
| AuthorizeAndGrant | AH |
| Block-List (Add) | BA |
| Block-List (Entity) | BL |
| Block-List (Remove) | BR |
| Blocked-List-InUse | BU |
| Client-ID | CI |
| Code | RC |
| CapabilityList | CA |
| CapabilityRequest | CR |
| ClearPublicProfile | CE |
| CompletionFlag | CF |
| Contact-List-ID | CL |
| Contact-List-ID-List | CO |
| ContactList-Notify | CY |
| Contact-List-Props | CP |
| DateTime | DT |
| Default-Association-List | DA |
| Default-CList-ID | DC |
| Default-List | DL |
| Default-Notify | DY |
| Delivery-Report-Request | DE |
| Description | RT |
| Detailed-Result – Application-ID | DJ |
| Detailed-Result – Contact-List-ID | DK |
| Detailed-Result – Domain | DD |
| Detailed-Result – Group | DG |
| Detailed-Result – Message-ID | DM |
| Detailed-Result – Screenname | DS |
| Detailed-Result – SearchElement | DH |

| Element | Code |
|---|---|
| Detailed-Result – TryAgainTimeout | DN |
| Detailed-Result – User | DU |
| Delivery-Time | DX |
| Digest-Bytes | DB |
| Digest-Schema | DI |
| ExtendConversationID | EI |
| ExtendConversationUserID | EU |
| ExtBlock | EB |
| ExtBlockETEM | ET |
| Font-Color | FC |
| Font-Size | FZ |
| Font-Style | FS |
| Grant-List (Add) | GA |
| Grant-List (Entity) | GL |
| Grant-List (Remove) | GR |
| Granted-List-InUse | GU |
| Group-Content-Limit | GC |
| Group-ID | GI |
| Group-Props | GP |
| HistoryPeriod | HP |
| ID-List (Contact list ID) | IC |
| ID-List (Domain) | ID |
| ID-List (Group-ID) | IG |
| ID-List (Screen name) | IS |
| ID-List (User ID) | IU |
| Invite-ID | II |
| Invite-Reason | IR |
| Invite-Response | IX |
| Invite-Type | IT |
| JoinGroup | JG |
| Joined-Request | JR |
| Joined | JU |
| JoinedBlocked | JB |
| Keep-Alive-Time | KA |
| Left-Users-List | LU |
| LeftBlocked | LB |
| MaxWatcherList | MW |
| Message-Content | MC |
| Message-Count | MN |
| Message-ID | MI |
| Message-Info | MF |
| Message-Info-List | ML |
| Message-Total-Count | MT |
| Moderator | MO |
| Name | NA |
| Namespace | NS |
| Nonce | NO |
| Not-Available-Functions | NF |
| Notification-Type | NT |

| Element | Code |
|---|---|
| Notification-Type-List | NL |
| Other-Server | OS |
| Own-Props | OP |
| Own-Screen-Name | ON |
| Password-String | PW |
| Presence | PR |
| PresenceList (ContactList) | PC |
| PresenceList (User) | PU |
| PresenceSubList | PS |
| PublicProfile | PP |
| Recalled-Content | RC |
| Recall-Reason | RR |
| Receive-List | RL |
| Recipient – Contact-ListID | RI |
| Recipient – GroupID | RG |
| Recipient – ScreenName | RM |
| Recipient – UserID | RE |
| Remove-Nick-List | RN |
| Remove-Users-List | RU |
| Requested-Functions | RF |
| Result (Status code and description) | ST |
| Screen-Name | SN |
| Search-Findings | SF |
| Search-ID | SD |
| Search-Index | SX |
| Search-Limit | SL |
| Search-Pair-List | SP |
| Search-Results | SR |
| Segment-Content | SJ |
| Segment-ID | SK |
| Segment-Info | SO |
| Sender – GroupID | SG |
| Sender – ScreenName | SM |
| Sender – UserID | SE |
| Session-Cookie | SC |
| Session-ID | SI |
| Session-Priority | SW |
| SubscribeNotification | SA |
| Subscribe-Type | SU |
| Subscription-State | SS |
| Supported-Digest-Schema | SH |
| SystemMessageList | SQ |
| SystemMessageResponseList | SV |
| Text | TX |
| Time-To-Live | TL |
| URL | UR |
| URLList | UL |
| User-ID | UI |
| User-ID-List | UE |

| Element | Code |
|---|---|
| UserIDPair | VX |
| User-List | US |
| User-Nick-List | UN |
| User-Prop-List | UP |
| UserMapList | UM |
| UserNotify | UY |
| Validity | VA |
| Version-List | VL |
| Watcher | WA |
| WatcherCount | WC |
| Welcome-Text | WT |

**Table 2: Information element codes**

The codes MUST NOT be case sensitive.

## 7.2 Service Tree Elements

To shorten messages, the elements of the service tree have been encoded into shorter codes. The tree is different as from what is specified in [CSP]. For further clarification please refer to chapter 7.11 - Information Element-specific Encoding on page 28. Note that the elements marked unsupported in the table below cannot be used in the SMS transport-specific service list.

| Name | SMS transport support | Code |
|---|---|---|
| ADDGM | Yes | AG |
| ADVSR | Yes | AS |
| BLENT | Yes | BL |
| CAINV | Yes | CI |
| CCLI | Yes | CC |
| ContListFunc | Yes | FC |
| CREAG | Yes | CG |
| DCLI | Yes | DC |
| DELGR | Yes | DG |
| EXCON | Yes | EC |
| FundamentalFeat | Yes | FF |
| FWMSG | No | FW |
| GCLI | Yes | GC |
| GETGM | Yes | GG |
| GETGP | Yes | GR |
| GETJU | Yes | GJ |
| GETLM | Yes | GL |
| GETM | Yes | GM |
| GETMAP | No | GA |
| GETPR | Yes | GP |
| GETSPI | Yes | GS |
| GETWL | Yes | GW |
| GLBLU | Yes | GB |
| GRCHN | Yes | GN |
| GroupAuthFunc | Yes | GF |
| GroupFeat | Yes | GE |
| GroupMgmtFunc | Yes | GT |
| GroupUseFunc | Yes | GU |

| Name | SMS transport support | Code |
|---|---|---|
| IMAuthFunc | Yes | IA |
| IMFeat | Yes | IF |
| IMReceiveFunc | Yes | IR |
| IMSendFunc | Yes | IS |
| MF | Yes | MF |
| MG | Yes | MG |
| MM | Yes | MM |
| MP | Yes | MP |
| INVIT | Yes | IV |
| InviteFunc | Yes | IN |
| MBRAC | Yes | MA |
| MCLS | Yes | MC |
| MDELIV | Yes | MD |
| NEWM | Yes | NM |
| NOTIF | No | NO |
| OFFNOTIF | Yes | ON |
| PresenceAuthFunc | Yes | PA |
| PresenceDeliverFunc | Yes | PD |
| PresenceFeat | Yes | PF |
| REJCM | No | RM |
| REJEC | Yes | RE |
| RMVGM | Yes | RG |
| SearchFunc | Yes | SF |
| ServiceFunc | Yes | SE |
| SETD | Yes | SD |
| SETGP | Yes | SG |
| SGMNT | Yes | SM |
| SRCH | Yes | SR |
| STSRC | Yes | ST |
| SUBGCN | Yes | SU |
| UPDPR | Yes | UP |
| VerifyIDFunc | Yes | VD |
| VRID | Yes | VI |
| WVCSPFeat | Yes | WV |

**Table 3: Service tree element codes**

The codes MUST NOT be case sensitive.

## 7.3 Client Capability Negotiation Elements

In order to shorten messages, the client capabilities have been encoded into a shorter code.

| Name | Code |
|---|---|
| AcceptedPullLength | AL |
| AcceptedPushLength | AU |
| AcceptedTextContentLength | AT |
| AcceptedTransferEncoding | AE |
| AnyContent | AY |
| ClientType | CT |
| CIRHTTPAddress | CI |

| Name | Code |
|---|---|
| CIRSMSAddress | CS |
| DefaultLanguage | DL |
| InitialDeliveryMethod | ID |
| MultiTrans | MT |
| MultiTransPerMessage | MP |
| OfflineETEMHandling | OE |
| OnlineETEMHandling | ON |
| ParserSize | PS |
| PlainTextCharSet | PT |
| SAPSessionLimit | SL |
| ServerPollMin | PM |
| SessionPriority | SP |
| SupportedBearer | SB |
| SupportedCIRMethod | SC |
| SupportedOfflineBearer | SO |
| TCPAddress | TA |
| TCPPort | TP |
| UDPAddress | UA |
| UDPPort | UP |
| UserSessionLimit | UL |

**Table 4: Client capability element codes**

The codes MUST NOT be case sensitive.

## 7.4 Client Capability Values

In order to shorten messages, the client capabilities have been encoded into a shorter code.

| Name | Code |
|---|---|
| DETECT | DE |
| FORKALL | FO |
| PRIORITYREJECT | PR |
| PRIORITYSTORE | PS |
| REJECT | RE |
| SENDREJECT | SR |
| SENDSTORE | SO |
| SERVERLOGIC | SL |
| SSMS | SS |
| STCP | ST |
| SUDP | SU |
| WAPSMS | WS |
| WAPUDP | WU |

**Table 5: Client capability value codes**

The codes MUST NOT be case sensitive.

The ClientType values are encoded in the presence value table.

## 7.5 Presence Attributes

The SMS protocol itself provides some features and lacks others, which limits or disables certain transactions due to content type limitation (only text/plain). The list of supported presence attreibutes is listed in the table below. Presence attributes that are marked:

- "Full" can be utilized completely.
- "Limited" cannot be completely utilized due to the limitations originating from the SMS transport.
- "N/A" cannot be used at all.

WSP and HTTP(S) transports are not limited in such a manner – all of the presence attributes can be utilized, however all content types except text/plain MUST be encoded using an agreed transfer-encoding method ,such as 'BASE64'.

Extension presence attributes are not supported, only presence attributes described in the [PA].

Only plain text-based presence attributes are supported.

The qualifier is not present.

To shorten messages, the presence attributes have been encoded in shorter codes.

| Name | SMS transport support | Code |
|---|---|---|
| AcceptedContentType | Full | AR |
| AcceptedTextContentLength | Full | AX |
| AcceptedTransferEncoding | Full | AE |
| Accuracy (GeoLocation) | Full | AL |
| Accuracy (Address) | Full | AA |
| Address | Full | AD |
| AddrPref | Full | AP |
| Alias | Full | AI |
| AnyContent | Full | AY |
| Altitude | Full | AT |
| ApplicationID | Full | AC |
| Building | Full | BU |
| Caddr | Full | CD |
| Cap | Full | CA |
| City | Full | CI |
| ClientContentLimit | Full | CL |
| ClientID | Full | CH |
| ClientIMPriority | Full | CG |
| ClientInfo | Limited | CF |
| ClientProducer | Full | CP |
| ClientType | Full | CT |
| ClientVersion | Full | CV |
| CommC | Full | CM |
| CommCap | Full | CC |
| Contact | Full | CB |
| ContactInfo | Simplified | CE |
| ContainedvCard | N/A | CJ |
| ContentType | Full | CY |
| Country | Full | CO |
| Crossing1 | Full | C1 |
| Crossing2 | Full | C2 |
| Cname | Full | CN |

| Name | SMS transport support | Code |
|---|---|---|
| Cpriority | Full | CR |
| Cstatus | Full | CS |
| DevManufacturer | Full | DM |
| DirectContent | N/A | DC |
| Extended Presence Info | N/A | |
| FreeTextLocation | Full | FT |
| GeoLocation | Full | GL |
| Inf_Link | Full | IK |
| InfoLink | Full | IL |
| Language | Full | LN |
| Latitude | Full | LA |
| Link | Full | LI |
| Longitude | Full | LO |
| MaxPullLength | Full | ML |
| MaxPushLength | Full | MS |
| Model | Full | MO |
| NamedArea | Full | NA |
| Note | Full | NT |
| OnlineStatus | Full | OS |
| PlainTextCharset | Full | PT |
| PLMN | Full | PM |
| PrefC | Full | PF |
| PreferredContacts | Full | PC |
| PreferredLanguage | Full | PL |
| PresenceValue | Limited | PV |
| ReferredContent | Full | RC |
| ReferredvCard | Full | RV |
| Registration | Full | RG |
| Status | Full | SA |
| StatusContent | Limited | SC |
| StatusMood | Full | SM |
| StatusText | Full | ST |
| Street | Full | SR |
| Text | Full | TE |
| TimeZone | Full | TZ |
| UserAvailability | Full | UA |
| Zone | Full | ZN |

**Table 6: Presence attribute element codes**

The codes MUST NOT be case sensitive.

## 7.6 Presence Values

To shorten messages, the enumerated presence values have been encoded in short codes.

| Name | Code |
|---|---|
| ANGRY | AG |
| ANXIOUS | AX |
| ASHAMED | AS |
| AUDIO_CALL | AU |

| Name | Code |
|---|---|
| AVAILABLE | AV |
| BORED | BO |
| CALL | CA |
| CLI | CL |
| CLOSED | CS |
| COMPUTER | CO |
| DISCREET | DI |
| EMAIL | EM |
| EXCITED | EX |
| HAPPY | HA |
| IM | IM |
| IM_OFFLINE | OF |
| IM_ONLINE | ON |
| IN_LOVE | IL |
| INVINCIBLE | IN |
| JEALOUS | JE |
| MMS | MS |
| MOBILE_PHONE | MP |
| NOT_AVAILABLE | NA |
| OPEN | OP |
| OTHER | OT |
| PDA | PD |
| SAD | SA |
| SLEEPY | SL |
| SMS | SM |
| VIDEO_CALL | VC |
| VIDEO_STREAM | VS |

**Table 7: Presence attribute value codes**

The codes MUST NOT be case sensitive.

## 7.7 Group Properties

To shorten messages, the group properties have been encoded in shorter codes.

| Name | Code |
|---|---|
| Accesstype | AT |
| ActiveUsers | AU |
| AutoDelete | AD |
| AutoJoin | AJ |
| History | HT |
| IsMember | IM |
| MaxActiveUsers | MU |
| Name | NM |
| PrivateMessaging | PM |
| PrivilegeLevel | PL |
| RequireInvitation | RI |
| Searchable | SE |
| ShowID | SI |
| Topic | TO |

| Name | Code |
|---|---|
| Type | TY |
| Validity | VL |
| WelcomeNote | WN |
| AutoJoin | AJ |

**Table 8: Group property codes**

The codes MUST NOT be sensitive.

## 7.8 Contact List Properties

To shorten messages, the contact list properties have been encoded in shorter codes.

| Name | Code |
|---|---|
| DisplayName | DN |
| DoNotNotify | DO |
| Default | DE |

**Table 9: Contact list property codes**

The codes MUST NOT be case sensitive.

## 7.9 Search and Profile Elements

To shorten messages, the enumerated search elements have been encoded in shorter codes.

| Name | Code |
|---|---|
| GROUP_ID | GI |
| GROUP_NAME | GN |
| GROUP_TOPIC | GT |
| GROUP_USER_ID_AUTOJOIN | UJ |
| GROUP_USER_ID_JOINED | GJ |
| GROUP_USER_ID_OWNER | GO |
| PP_AGE | UG |
| PP_CITY | UC |
| PP_COUNTRY | UO |
| PP_FREE_TEXT | UX |
| PP_FRIENDLY_NAME | UN |
| PP_GENDER | UR |
| PP_INTENTION | UT |
| PP_INTERESTS | UH |
| PP_MARITAL_STATUS | US |
| USER_AGE_MAX | AA |
| USER_AGE_MIN | AI |
| USER_ALIAS | UA |
| USER_EMAIL_ADDRESS | UE |
| USER_FIRST_NAME | UF |
| USER_ID | UI |
| USER_LAST_NAME | UL |
| USER_MOBILE_NUMBER | UM |
| USER_ONLINE_STATUS | UO |
| GROUP_USER_ID_OWNER | GO |

| Name | Code |
|---|---|
| GROUP_USER_ID_AUTOJOIN | UJ |

**Table 10: Search element codes**

The codes MUST NOT be are not case sensitive.

# 7.10 Watcher State Values

To shorten messages, the enumerated watcher state values have been encoded in shorter codes.

| Name | Code |
|---|---|
| CURRENT_SUBSCRIBER | CS |
| FORMER_SUBSCRIBER | FS |
| PRESENCE_ACCESS | PA |

**Table 11: Watcher state value codes**

The codes MUST NOT be case sensitive.

# 7.11 Information Element-specific Encoding

There are a number of information elements where the syntax is not self-explanatory, and requires additional clarification. These information elements are collected and explained in this chapter.

## 7.11.1 Version-List

The version list – unlike in XML – is not expressed using namespaces, as the Plain Text Syntax itself does not use namespaces. Version numbers are expressed as it is defined in section 5 - Plain Text Syntax on page 12 (see "aa" in preamble), thus the list of different version numbers is expressed as a list of comma-separated numbers. For example the following list is valid for a client/server that supports 1.0, 1.1, 1.2 and 1.3 versions of WV specifications:

```
VL=(10,11,12,13)
```

The version numbers MAY be in any order.

## 7.11.2 Result parameter (ST)

In XML the Result element is carried within one single structure: Result (ST). In PTS the result details are divided into separate parameters so that it is possible to distinguish the referenced ContactList-IDs (DI), Domains (DD), Group-IDs (DG), Message-IDs (DM), ScreenNames (DS) and User-IDs (DU).

```
The syntax for the Result parameter is as follows:
ST=<StatusCode>
ST=(<StatusCode>,<Description>)
```

The syntax for the Detailed Result parameters is as follows:

```
DJ=(<StatusCode1>,[<Description1>],<ApplicationIDn>)
or
DJ=((<StatusCode1>,[<Description1>],<ApplicationIDn>)\
,(<StatusCodem>,[<Descriptionm>],<ApplicationIDo>))

DK=(<StatusCode1>,[<Description1>],<ContactListIDn>)
or
DK=((<StatusCode1>,[<Description1>],<ContactListIDn>)\
,(<StatusCodem>,[<Descriptionm>],<ContactListIDo>))

DD=(<StatusCode1>,[<Description1>],<Domainn>)
or
DD=((<StatusCode1>,[<Description1>],<Domainn>)\
,(<StatusCodem>,[<Descriptionm>],<Domaino>))

DG=(<StatusCode1>,[<Description1>],<GroupIDn>)
or
DG=((<StatusCode1>,[<Description1>],<GroupIDn>)\
```

```
,(<StatusCode_m>,[<Description_m>],<GroupID_o>))

DM=(<StatusCode_1>,[<Description_1>],<MessageID_n>)
or
DM=((<StatusCode_1>,[<Description_1>],<MessageID_n>)\
,(<StatusCode_m>,[<Description_m>],<MessageID_o>))

DS=(<StatusCode_1>,[<Description_1>],((<ScreenName_n>,<GroupID_n>)))
or
DS=((<StatusCode_1>,[<Description_1>],((<ScreenName_n>,<GroupID_n>)))\
,(<StatusCode_m>,[<Description_m>],((<ScreenName_o>,<GroupID_o>))))

DU=(<StatusCode_1>,[<Description_1>],<UserID_n>)
or
DU=((<StatusCode_1>,[<Description_1>],<UserID_n>)\
,(<StatusCode_m>,[<Description_m>],<UserID_o>))

DH=(<StatusCode_1>,[<Description_1>],<SearchElement_n>)
or
DH=((<StatusCode_1>,[<Description_1>],<SearchElement_n>)\
,(<StatusCode_m>,[<Description_m>],<SearchElement_o>))

DN=<TryAgainTimeOut>)
```

where

<StatusCode> is the status code. Please refer to [CSP] for the codes.
<Description> is the description of the result.
<ApplicationID> is an Application-ID to which the error code in the detailed result refers.
<ContactListID> is a Contact-List-ID to which the error code in the detailed result refers.
<Domain> is a Domain to which the error code in the detailed result refers.
<GroupID> is a Group-ID to which the error code in the detailed result refers.
<MessageID> is a Message-ID to which the error code in the detailed result refers.
<ScreenName> is a Screen-Name (always with Group-ID) to which the error code in the detailed result refers.
<UserID> is a User-ID to which the error code in the detailed result refers.
<SearchElement> is a Search-Element to which the error code in the detailed result refers.
<TryAgainTimeout> is the TryAgainTimeout of the related transaction.

The followings are valid examples:

```
ST=201 DU=(531,"Unknown user.",wv:bad_user1@im.com)

ST=(201,"Partially completed.") DU=((531,"Unknown
user",wv:bad_user1@im.com),(532,,wv:bad_user2@im.com))

ST=(560,"Unsupported search-element was requested.") DH=((562,,(UT,UH,UE)))

DN=3600
```

## 7.11.3 Service tree

The service tree – unlike in XML – is not transferred as a tree. It is simply a list of short codes, which indicate the requested or not agreed services. The list MUST NOT contain one specific element more than once. If a child element is specified then the parent is not present in the list. The short codes MAY be in any order.

### 7.11.3.1 Explanation using an example

The following step-by-step build-up demonstrates the following case: the client requests all available features and functions, but the server does not agree to provide FundamentalFeat (FF), ContListFunc (FC), PresenceAuthFunc (PA), and IMAuthFunc (IA). Note that the examples below are indicating the service list only, all other parameters are missing.

The client requests every features and functions so it sends the root element only:

```
RF=WV
```

The server sends the negated list to the client. It indicates only those elements that the client did request, but the server does not agree to provide. In our example FF, FC, PA, IA SHALL be indicated. Let us build up this list (top-down).

WV is the root, that's surely needed, and we put the parentheses there, since there will be more than one element.

```
NF=(WV)
```

Let us go one level down. The server does not provide the whole FF sub-tree. FF is a sub-tree; it is a child of WV, so the parent (WV) is removed and FF is added to the list.

```
NF=(FF)
```

The server does not provide FC and PA. These are child element of the PF, so PF needs to be added to the list as well.

```
NF=(FF,PF)
```

The server does not provide IA. These are child elements of the IF, so IF needs to be added to the list as well.

```
NF=(FF,PF,IF)
```

Let us go one level down. FC, and PA SHALL be indicated in PF, so we add these two and remove the parent PF.

```
NF=(FF,FC,PA,IF)
```

Finally, IA SHALL be indicated in IF, so we add IA, and remove the parent IF.

```
NF=(FF,FC,PA,IA)
```

The servers sends this tree in the response:

```
NF=(FF,FC,PA,IA)
```

The list MAY be constructed many different ways, depends on the actual implementation.

## 7.11.4 PresenceSubList parameter

Unlike in XML, SMS does not support any extended presence information.

The PresenceSubList element is very similar to the service tree. Each presence information element and enumerated value has its own short code, which MUST be used at all times.

For clarification the basic syntax is:

Attributes that do not have child attributes:
<attribute>[,<qualifier$_m$>][,<value$_n$>]
Attributes that do have child attributes:
<attribute>[,<qualifier$_o$>,<sub-attribute$_o$>]

<sub-attribute> has the same structure as <attribute>, <qualifier> and <value> carries the actual qualifier and presence value. The qualifier is always present in attributes that do have qualifier; the value MAY be missing (if value is meant to be empty). If there is more than one presence element present in under the same parent, then these MUST be in parentheses.

The order of the <attribute> element within the PresenceSubList does not matter, but every single <attribute> element MAY be present only once.

### 7.11.4.1 Explanation using an example

The following step-by-step build-up demonstrates how to build up an empty Presence SubList parameter. (Empty PresenceSubList is used for referencing: attribute lists, authorization, etc.) It will contain all presence attributes that are available for referencing (except ContainedvCard, DirectContent, extended presence information: these are not supported presence attributes).

Let us begin with the easy ones that do not have child attributes: OnlineStatus (OS), Registration (RG), FreeTextLocation (FT), PLMN (PM), UserAvailability (UA), PreferredLanguage (PL), StatusText (ST), StatusMood (SM), Alias (AI), TimeZone (TZ):

```
PS=(OS,RG,FT,UA,PL,ST,SM,AI)
```

Let us add now the ones that do have child attributes (let us not add child attributes yet): ClientInfo (CF), GeoLocation (GL), Address (AD), CommCap (CC), PreferredContacts (PC), StatusContent (SC), ContactInfo (CE):

```
PS=(OS,RG,FT,UA,PL,ST,SM,AI,TZ,CF,GL,AD,CC,PC,SC,CE,IL)
```

If used in the proper primitive, the above PresenceSubList element would authorize all presence information to the specified users. Note that some of these attributes do have child attribute(s), but these child attributes are not usable in empty attribute lists, as those cannot be authorized separately.

Now let us see how the PresenceSubList is filled up with qualifiers and values (and the possible child attributes). The following example demonstrates how to build up a PresenceSubList parameter that carries all supported presence attributes. In order to demonstrate empty values; attributes without qualifier and empty values - the TimeZone (TZ) and the DevManufacturer attributes will have empty values. In order to demonstrate how to include more than one child attribute (where allowed of course), two CommC (CM) and two AddrPref (AP) attributes will be included.

Let us create the whole structure first without any qualifiers, or values (note that this example will be invalid, because a reference (empty) list MUST NOT contain child attributes). The child attributes to be added to the empty PresenceSubList above are:

| | |
|---|---|
| OnlineStatus (OS) | PresenceValue (PV), ClientID (CH) |
| Registration (RG) | PresenceValue (PV), ClientID (CH) |
| ClientInfo (CF) | ClientContentLimit (CL), ClientType (CT), DevManufacturer (DM), ClientProducer (CP), Model (MO), ClientVersion (CV), Language (LN), ClientIMPriority (CG), ApplicationID (AC), ClientID (CH) |
| TimeZone (TZ) | PresenceValue (PV), ClientID (CH) |
| GeoLocation (GL) | Longitude (LO), Latitude (LA), Altitude (AT), Accuracy (AL), ClientID (CH) |
| Address (AD) | Country (CO), City (CI), Street (SR), Crossing1 (C1), Crossing2 (C2), Building (BU), NamedArea (NA), Accuracy (AA), ClientID (CH) |
| FreeTextLocation (FT) | PresenceValue (PV), ClientID (CH) |
| PLMN (PM) | PresenceValue (PV), ClientID (CH) |
| CommCap (CC) | CommC (CM) , ClientID (CH) |
| PreferredContacts (PC) | AddrPref (AP) |
| StatusContent (SC) | ReferredContent (RC), ContentType (CY) - DirectContent is not supported. |
| ContactInfo (CE) | ReferredvCard (RV) - ContainedvCard is not supported. |
| InfoLink (IL) | Inf_Link (IK) |

```
PS=((OS,(PV,CH)),(RG,(PV,CH)),(FT,(PV,CH)),(PM,(PV,CH)),UA,PL,ST,SM,AI,(TZ,(PV,CH)),(CF,(CL,CT,DM,
CP,MO,CV,LN,CG,AC,CH)),(GL,(LO,LA,AT,AL,CH)),(AD,(CO,CI,SR,C1,C2,BU,NA,AA,CH)),(CC,(CM,CM,CH)),(PC
,(AP,AP)),(SC,RC),(CE,RV),(IL,IK))
```

Let us create the lowest level of the structure now (note that this example will be invalid as well). The child attributes to be added to the empty PresenceSubList above are:

| | |
|---|---|
| ClientContentLimit (CL) | AcceptedContentType (AR), AcceptedTextContentLength (AX), AcceptedTransferEncoding (AE), MaxPullLength (ML), MaxPushLength (MS), PlainTextCharset (PT) |
| CommC (CM) | Cap (CA), Status (SA), Contact (CB), Note (NT) |
| AddrPref (AP) | PrefC (PF), Caddr (CD), Cstatus (CS), Cname (CN), Cpriority (CR) |
| Inf_Link (IK) | Link (LI), Text (TE), ContentType (CY) |

```
PS=((OS,(PV,CH)),(RG,(PV,CH)),(FT,(PV,CH)),(PM,(PV,CH))UA,PL,ST,SM,AI,(TZ,(PV,CH)),(CF,((CL,((AR,A
R),AX,AE,ML,MS,(PT,PT,PT))),CT,DM,CP,MO,CV,LN,CG,AC,CH)),(GL,(LO,LA,AT,AL,CH)),(AD,(CO,CI,SR,C1,C2
,BU,NA,AA,CH)),(CC,((CM,(CA,SA,CB,NT)),(CM,(CA,SA,CB,NT)),CH)),(PC,((AP,(PF,CD,CS,CN,CR)),(AP,(PF,
CD,CS,CN,CR)))),(SC,(RC,CY)),(CE,RV),(IL,(IK,(LI,TE,CY))))
```

The structure is complete, let us fill it up now with qualifiers (those that have), and values (those that have).

```
PS=((OS,T,((PV,T),(CH,http://123.123.123.123:80/IMPSAPP))),(RG,T,((PV,T),(CH,http://123.123.123.12
3:80/IMPSAPP))),(FT,T,((PV,"At
```

```
home”),(CH,http://123.123.123.123:80/IMPSAPP))),(PM,T,((PV,”NETWORK01”),(CH,http://123.123.123.123
:80/IMPSAPP))),(UA,T,AV),(PL,T,fin),(ST,T,”Busy editing a
document”),(SM,T,SL),(AI,T,ASa),(TZ,T,((PV,+02),(CH,http://123.123.123.123:80/IMPSAPP))),(CF,T,(CL
,(((AR,(image/jpeg,307200,R,307201)),(AR,(multipart/mixed,1024000,C,1024001))),(AX,10240),(AE,BASE
64),(ML,5242880),(MS,102400),((PT,4),(PT,62),(PT,106)))),((CT,MP),(DM,”ABC company"),(CP,”DEF
Company”),(MO,xyz200),(CV,1.1b),(LN,fin),(CG,32767),(AC,MyChessGamev1.1b123),(CH,http://123.123.12
3.123:80/IMPSAPP))),(GL,T,((LO,”35 24 15.652W”),(LA,”12 36
22.5N”),(AT,250),(AL,50),(CH,http://123.123.123.123:80/IMPSAPP))),(AD,T,((CO,FR),(CI,Paris),(SR,”B
ig street”),(C1,”A street”),(C2,”B street”),(BU,”Eiffel tower”),(NA,”Eiffel
tower”),(AA,300),(CH,http://123.123.123.123:80/IMPSAPP))),(CC,T,((CM,((CA,CA),(SA,CS),(CB,+3589912
3123),(NT,” I am using this phone outside office
hours”))),(CM,((CA,IM),(SA,OP),(CB,wv:user@im.com),(NT,”My IM-application is now
online”))),(CI,http://123.123.123.123:80/IMPSAPP))),(PC,T,((AP,((PF,CA),(CD,+35899123123),(CS,OP),
(CN,”Home Phone”),(CR,10))),(AP,((PF,SM),(CD,+35899123123),(CS,CS),(CN,”Home
SMS”),(CR,20))))),(SC,T,((RC,http://www.foo.com/MMS/Pictures/MyLogo),(CY,image/gif))),(CE,T,(RV,ht
tp://www.foo.com/Contacts/vCards/MyCard)),(IL,T,(IK,((LI,http://www.myserviceprovider.com/myHomePa
ge),(TE,”This is my homepage”),(CY,text/html)))))
```

The PresenceSubList above is a complete and valid example.

## 7.11.5 Presence parameter (PR)

The syntax for the Presence parameter is as follows:

```
PR=(<UserID1>[,<PresenceSubList1>])
PR=((<UserID1>[,<PresenceSubList1>]),(<UserIDn>[,<PresenceSubListn>]))
```

where

<UserID> is the User-ID to whom the presence information belongs.

<PresenceSubList> is the list of presence attributes and values. See chapter 7.11.4 - PresenceSubList parameter on page 30.

The followings are valid examples:

```
PR=(wv:matthias@salamander.com)
PR=((wv:matthias@salamander.com),(wv:Francisco))
PR=((wv:matthias@salamander.com),(wv:Francisco,((OS,T,T))))

PR=(wv:matthias@salamander.com,((OS,,T)))
PR=(wv:matthias@salamander.com,((OS,T,T)))
PR=(wv:matthias@salamander.com,((OS,T,T),(FT,T,”In the office”)))
PR=((wv:matthias@salamander.com,((OS,T,T),(FT,T,”In the office”))),(wv:francisco,((OS,T,T))))
```

## 7.11.6 PresenceList

The PresenceList element syntax according to [CSP XMLS] would be:

```
Parm=((<UserIDn>,UserNotifyn) | (<ContactListIDn>,ContactListNotifyn)[,PresenceSubListn])
```

However in order to be able to distinguish between User-ID(s) and ContactList-ID(s), the PresenceList element has been split up to separate PresenceList elements (PU and PC). The syntax of the PresenceList parameter MUST be:

```
PC=(<CListID1>,ContactListNotify1[,<PresenceSubList1>])
PU=(<UserID1>,UserNotify1[,<PresenceSubList1>])
```

or

```
PC=((<CListID1>,ContactListNotify1[,<PresenceSubList1>]),(<ClistIDn>,ContactListNotifyn[,<PresenceSu
bListn>]))
PU=((<UserID1>,UserNotify1[,<PresenceSubList1>]),(<UserIDn>,UserNotifyn[,<PresenceSubListn>]))
```

where

<CListID> is the ContactList-ID to which the association belongs.

<UserID> is the User-ID to whom the association belongs.

<UserNotify> is the notification flag for the User-ID to whom the association belongs.

<ContactListNotify> is the notification flag for the ContactList-ID to whom the association belongs.

<PresenceSubList> is the list of associated presence attributes. See chapter 7.11.4 PresenceSubList parameter on page 30.

The followings are valid examples:

```
PC=(wv:john/colleagues,T)
PC=((wv:john/colleagues,T),(wv:john/family,T))
PC=((wv:john/colleagues,T),(wv:john/family,T,UA))

PU=(wv:john@smith.com,F,UA)
PU=(wv:john@smith.com,F,(UA,OS))
PU=((wv:john@smith.com,F,UA),(wv:matthias@salamander.com,F,(OS,PL)))
```

## 7.11.7 Message-Info (MF) parameter

The Message-Info element syntax according to [CSP XMLS] would be:

```
MF=([<MessageID>],[<MessageURI>],[<ContentType>],[<ContentEncoding>],<ContentSize>,[<ContentName>]
,<Recipient>,<Sender>,[<DateTime>],[<Font>],[<Validity>])
```

However in order to be able to distinguish between Recipient/Sender User-IDs, Contact-List-IDs, Group-IDs, screen names, as well as Font color, size and style, these parameters have been split up:

- The Recipient and Sender elements MUST be:

```
([<UserID(s)>],[<ContactListID(s)>],[<GroupID(s)>],[<ScreenName(s)>])
```

- The Font element MUST be:

```
([<FontColor>],[<FontSize>],[<FontStyle>])
```

Thus the encoding of the Message-Info element in PTS MUST be:

```
MF=([<MessageID>],[<MessageURI>],[<ContentType>],[<ContentEncoding>],<ContentSize>,[<ContentName>]
,([<Recipient UserID(s)>],[<Recipient ContactListID(s)>],[<Recipient GroupID(s)>],[<Recipient
ScreenName(s)>]),([<Sender UserID(s)>],[<Sender ContactListID(s)>],[<Sender GroupID(s)>],[<Sender
ScreenName(s)>]),[<DateTime>],[<Font>],[<Validity>])
```

The followings are valid examples:

- In SendMessageRequest primitive the Message-ID, MessageURI, ContentType, ContentEncoding, ContentName, DateTime, Font, Validity MAY/MUST be missing:
  - Recipient using User-ID:

```
MF=(,,,,60,,(wv:matthias@salamander.com),(wv:me@home.com))
```

  - Recipient using User-ID with Client-ID:

```
MF=(,,,,60,,(((wv:matthias@salamander.com,,http://123.123.123.123:80/IMPSAPP),wv:Francisco)),(wv:m
e@home.com,” Me, myself and I”))
```

  - Recipient using User-ID with Application-ID:

```
MF=(,,,,60,,(((wv:matthias@salamander.com,,,MyChessGamev1.0),wv:Francisco)),(wv:me@home.com,” Me,
myself and I”))
```

  - Recipients using Contact-List-ID:

```
MF=(,,,,60,,(,wv:john/colleagues),(wv:me@home.com))
MF=(,,,,60,,(,(wv:john/colleagues,wv:john/friends)),(wv:me@home.com))
```

  - Recipients using Group-ID:

```
MF=(,,,,60,,(,,wv:/chatgroup@wv.com),(,,,((Some1,wv:/chatgroup@wv.com))))
MI=(,,,,60,,(,,(wv:/chatgroup@wv.com,wv:/othergroup@wv.com)),(,,,((Some1,wv:/chatgroup@wv.com))))
```

  - Recipients using ScreenNames:

```
MF=(,,,,60,,(,,,((“The boss”,wv:/chatgroup@wv.com))),(,,,((Some1,wv:/chatgroup@wv.com))))
MF=(,,,,60,,(,,,((“Gary”,wv:/chatgroup@wv.com),(“Andy”,wv:/chatgroup@wv.com))),(,,,((Some1,wv:/cha
tgroup@wv.com))))
```

- In NewMessage primitive the MessageURI, ContentType, ContentEncoding, ContentName, Font might be missing:
  - Sender using User-ID:

```
MF=(11235,,,,60,,(wv:me@home.com),((wv:matthias@salamander.com,"Matthias")),20011118T1203Z,,3600)
```

- Sender using User-ID with Client-ID:

```
MF=(11235,,,,60,,(wv:me@home.com),((wv:matthias@salamander.com,"Matthias",http://123.123.123.123:80/IMPSAPP),20011118T1203Z,,3600)
```

- Sender using User-ID with Application-ID:

```
MF=(11235,,,,60,,(wv:me@home.com),((wv:matthias@salamander.com,"Matthias",,MyChessGamev1.0),20011118T1203Z,,3600)
```

- Sender using Group-ID:

```
MF=(11235,,,,60,,(,,,((Some1,wv:/chatgroup@wv.com))),(,,wv:/chatgroup@wv.com),20011118T1203Z,,3600)
```

- Sender using ScreenName:

```
MF=(11235,,,,60,,(,,,((Some1,wv:/chatgroup@wv.com))),(,,,(("Gary",wv:/chatgroup@wv.com))),20011118T1203Z,,3600)
```

- Various font formattings:

```
MF=(,,,,60,,(wv:matthias@salamander.com),(wv:me@home.com),,(Red,Big,Bold))
MF=(,,,,60,,(wv:matthias@salamander.com),(wv:me@home.com),,(Red,Big,Italic))
MF=(,,,,60,,(wv:matthias@salamander.com),(wv:me@home.com),,(Red,Big,(Bold,Italic)))
```

### 7.11.8 Block-List and Grant-List parameters

The Block-List and Grant-List element syntax according to [CSP XMLS] would be:

```
Parm=(<EntityList> | ([<AddList>],[<RemoveList>]))
```

However in order to be able to distinguish between EntityList, AddList and RemoveList, the Block-List and Grant-List elements have been split up to separate EntityList (BL and GL), AddList (BA and GA) and Remove-List (BR and GR) parameters. These parameters have to be split further in order to be able to distinguish between User-IDs, Contact-List-IDs, Group-IDs, screen names and Application-IDs. The syntax of the EntityList, AddList and RemoveList parameters MUST be:

```
[<UserID(s)>],[<ContactListID(s)>],[<GroupID(s)>],[<ScreenName(s)>],[<ApplicationID(s)]
```

Thus the encoding of the Block-List and Grant-List in PTS MUST be:

```
BL=[<Entity-UserID(s)>],[<Entity-ContactListID(s)>],[<Entity-GroupID(s)>],[<Entity-ScreenName(s)>],[<Entity-ApplicationID(s)]
GL=[<Entity-UserID(s)>],[<Entity-ContactListID(s)>],[<Entity-GroupID(s)>],[<Entity-ScreenName(s)>],[<Entity-ApplicationID(s)]
```

or

```
BA=[<Add-UserID(s)>],[<Add-ContactListID(s)>],[<Add-GroupID(s)>],[<Add-ScreenName(s)>],[<Add-ApplicationID(s)]
BR=[<Remove-UserID(s)>],[<Remove-ContactListID(s)>],[<Remove-GroupID(s)>],[<Remove-ScreenName(s)>],[<Remove-ApplicationID(s)]

GA=[<Add-UserID(s)>],[<Add-ContactListID(s)>],[<Add-GroupID(s)>],[<Add-ScreenName(s)>],[<Add-ApplicationID(s)]
GR=[<Remove-UserID(s)>],[<Remove-ContactListID(s)>],[<Remove-GroupID(s)>],[<Remove-ScreenName(s)>],[<Remove-ApplicationID(s)]
```

The followings are valid examples:

- Grant-List:
  - Using User-ID(s) in EntityList:

```
GL=wv:matthias@salamander.com
GL=((wv:matthias@salamander.com,wv:me@home.com))
```

  - Using Contact-List-ID(s) in AddList:

```
GA=(,wv:john/colleagues)
```

```
GA=(,(wv:john/colleagues,wv:john/friends))
```

- Using Group-ID(s) in RemoveList:

```
GR=(,,wv:/chatgroup@wv.com)
GR=(,,(wv:/chatgroup@wv.com,wv:/othergroup@wv.com))
```

- Block-List:
  - Using screen name(s) in EntityList:

```
BL=(,,,(("Gary",wv:/chatgroup@wv.com)))
BL=(,,,(("Gary",wv:/chatgroup@wv.com),("Andy",wv:/chatgroup@wv.com)))
```

- Using Application-ID(s) in AddList:

```
BA=(,,,,MyChessGame*)
BA=(,,,,(MyChessGame*,YourChessGame*))
```

- Using Application-ID(s) in RemoveList:

```
BR=(,,,,MyChessGame*)
BR=(,,,,(MyChessGame*,YourChessGame*))
```

## 7.11.9 AdminMapList, UserMapList parameters

These information elements are defined as structures in XML, and cannot be included in PTS without specific syntax. In order to be able to distinguish between recipient Admin, Mod, and User mappings within the AdminMapList element are separated to `AA`, `AM`, `AS`.

The syntax is:

```
AA=<MappingList>
AM=<MappingList>
AE=<MappingList>
UM=<MappingList>
```

Where <MappingList> is comma-separated <Mapping> elements: $<Mapping_1>,<Mapping_n>$

One single <Mapping> element consists of a Screen-Name (note that it is only the Name part without Group-ID), and an optional User-ID: <SName>[,<UserID>]. If the <Mapping> element contains the optional User-ID, it is wrapped with parentheses: (<SName>,<UserID>) .

Examples:

```
AA=He
AM=((He,wv:he@there.com))
AE=(He,She)
UM=((He,wv:he@there.com),(She,wv:she@there.com))
```

Note that the parentheses MUST be doubled when there is only one screen name with user-id. Please refer to chapter 7.12.5 Single nickname on page 37, as the problem described there is identical.

## 7.11.10 ExtBlock and ExtBlockETEM parameter

The syntax for the ExtBlock parameter is as follows:

```
EB=((<namespace indication>,([param values])))
ET=((<namespace indication>,([param values])))
```

, or to include several namespaces and parameter values:

$EB=((<namespace_1>,([paramvalues_1])),(<namespace_n>,([paramvalues_n])))$
$ET=((<namespace_1>,([paramvalues_1])),(<namespace_n>,([paramvalues_n])))$

The contents and syntax of "Param values" is outside the scope of IMPS specifications.

The followings are valid examples:

```
EB=((http://www.foo.com/1.2,(T,Blue,(15,12),"No way")))
```

```
EB=((http://www.foo.com/1.2,(T,Blue,(15,12),"No way")),(http://www.oof.com/1.2,(16)))
```

# 7.12 Single Parameters

## 7.12.1 Single presence attribute with qualifier or value

Case: the PresenceSubList parameter is a single presence attribute with qualifier and/or value.

A single presence attribute consists an attribute, a qualifier, and/or a value:

Parm=<attribute>[,<qualifier>,<sub-attribute>]

Parm=<attribute>[,<qualifier>][,<value>]

The problem is that this looks like a list for the parser, so it is not possible to include a single presence attribute with qualifier and/or value in any parameter like this:

```
PS=UA,T,VA
```

Nor like this (this would be recognized as a reference (empty) attribute list):

```
PS=(UA,T,VA)
```

In order to be able to include a single presence attribute with qualifier and/or value in a parameter the parentheses MUST be doubled:

```
PS=((UA,T,VA))
```

Note that this problem does not occur when there is a list of presence attributes (with qualifier and/or value) assigned to a parameter, since then the double parentheses are already there:

```
PS=((OS,T,((PV,T),(CH,http://123.123.123.123:80/IMPSAPP))),(UA,T,VA))
```

## 7.12.2 Single User-ID with FriendlyName

Case: any user-ID parameter is a single User-ID with FriendlyName.

A single User-ID with FriendlyName consists of a User-ID, and a FriendlyName:

```
Parm=<User-ID>,<FriendlyName>
```

The problem is that this looks like a list for the parser, so it is not possible to include a single User-ID with FriendlyNamein any parameter like this:

```
SE=wv:john@smith.com,Johnnie
```

Nor like this (this would be recognized as two User-IDs):

```
SE=(wv:john@smith.com,Johnnie)
```

In order to be able to include a single User-ID with FriendlyName in a parameter the parentheses MUST be doubled:

```
SE=((wv:john@smith.com,Johnnie))
```

Note that this problem does not occur when there is a list of User-IDs (with FriendlyName) assigned to a parameter, since then the double parentheses are already there:

```
SE=((wv:john@smith.com,Johnnie),(wv:me@home.com," Me, myself and I"))
```

Note that this problem does not occur when there is a Client-ID specified for the user, since then the double parentheses are already there:

```
SE=((wv:john@smith.com,,http://123.123.123.123:80/IMPSAPP))
SE=((wv:me@home.com,"Me, myself and I",http://100.100.100.100:80/IMApp))
```

## 7.12.3 Single screen name

Case: any screen name parameter is a single screen name.

A single screen name consists of a name, and a group-ID part:

```
Parm=<SName>,<Group-ID>
```

The problem is that this looks like a list for the parser, so it is not possible to include a single screen name in any parameter like this:

```
SN=matthias,wv:/chatgroup@wv.com
```

Nor like this:

```
SN=(matthias,wv:/chatgroup@wv.com)
```

In order to be able to include a single screen name in a parameter the parentheses MUST be doubled:

```
SN=((matthias,wv:/chatgroup@wv.com))
```

Note that this problem does not occur when there is a list of screen names assigned to a parameter, since then the double parentheses are already there:

```
SN=(("The boss",wv:/othergroup@somewhere.com),(matthias,wv:/chatgroup@wv.com))
```

### 7.12.4 Single search-pair

Case: a search-pair parameter is a single search-pair.

A single search-pair consists of a type, and a substring part:

```
Parm=<Type>,<Substring>[,PairID]
```

The problem is that this looks like a list for the parser, so it is not possible to include a single search-pair in a search-pair parameter like this:

```
SP=UL,Smith
```

Nor like this:

```
SP=(UL,Smith)
```

In order to be able to include a single search-pair in a parameter the parentheses MUST be doubled:

```
SP=((UL,Smith))
```

Note that this problem does not occur when there is a list of search-pairs assigned to a parameter, since then the double parentheses are already there:

```
SP=((UL,Smith),(UO,T))
```

### 7.12.5 Single nickname

Case: any nickname parameter is a single nickname.

A single nickname list is either a User-ID or a NickName that consists of a name and a user-ID part:

```
Parm=<User-ID>
Parm=<Name>,<User-ID>
```

The latter definition looks problematic as it looks like a list for the parser, so it is not possible to include a single nickname in any parameter like this:

```
AN="Randall the Vandal",wv:randall@fairlane.com
```

Nor like this:

```
AN=("Randall the Vandal",wv:randall@fairlane.com)
```

In order to be able to include a single nickname in a parameter the parentheses MUST be doubled:

```
AN=(("Randall the Vandal",wv:randall@fairlane.com))
```

Note that this problem does not occur when there is a list of nicknames assigned to a parameter, since then the double parentheses are already there:

```
AN=(("Randall the
Vandal",wv:randall@fairlane.com),wv:no.nick@name.com,(Brainstrom,wv:bright@dark.com))
```

Note that this problem does not occur when User-IDs are used.:

```
AN=wv:randall@fairlane.com
AN=(wv:randall@fairlane.com,wv:no.nick@name.com)
```

## 7.12.6 Single contact-list, group and own group properties

Case: a contact-list (CP) or group (GP) or own group (OP) property parameter is a single property.

A single property consists of a property name, and a property value part:

```
Parm=<Property>,<Value>
```

The problem is that this looks like a list for the parser, so it is not possible to include a single property in a CP, GP, or OP property parameter like this:

```
CP=DN,"My enemies"
```

Nor like this:

```
CP=(DN,"My enemies")
```

In order to be able to include a single contact-list property in a parameter the parentheses MUST be doubled:

```
CP=((DN,"My enemies"))
```

Note that this problem does not occur when there is a list of contact-list properties assigned to a parameter, since then the double parentheses are already there:

```
CP=((DN,"My enemies"),(DE,T),(DO,F))
```

## 7.12.7 Single watcher

Case: any watcher parameter includes only a single watcher.

A single watcher consists of a user-ID with FriendlyName, a client-ID, and the watcher status (client-ID and FriendlyName are optional):

```
Parm=<User-ID>,<WatcherStatus>
Parm=(<User-ID>,<FriendlyName>),<WatcherStatus>
Parm=(<User-ID>,<Client-ID>),<WatcherStatus>
Parm=((<User-ID>,<FriendlyName>),<Client-ID>),<WatcherStatus>
```

The problem is that these look like a list for the parser and in some cases the Client-ID might be confused with FriendlyName, so it is not possible to include a single watcher in any parameter like this:

```
WA=wv:noinfo@there.com,PA
WA=(wv:he@there.com,He),FS
WA=(wv:friend@there.com,http://124.124.124.124:80/IMPSAPP),FS
WA=((wv:she@there.com,She),http://123.123.123.123:80/IMPSAPP),FS
```

Nor like this (this would be recognized as two separate watchers):

```
WA=(wv:noinfo@there.com,PA)
WA=((wv:he@there.com,He),FS)
WA=((wv:friend@there.com,http://124.124.124.124:80/IMPSAPP),FS)
WA=(((wv:she@there.com,She),http://123.123.123.123:80/IMPSAPP),FS)
```

In order to be able to include a single watcher (with or without client-ID) in a parameter the parentheses MUST be doubled and the User-ID MUST be included in parentheses even though the FriendlyName is not included:

```
WA=(((wv:noinfo@there.com),PA))
WA=(((wv:he@there.com,He),FS))
WA=((((wv:friend@there.com),http://124.124.124.124:80/IMPSAPP),FS))
WA=((((wv:she@there.com,She),http://123.123.123.123:80/IMPSAPP),FS))
```

Note that this problem does not occur when there is a list of user-IDs (with client-IDs) assigned to a parameter, since then the double parentheses are already there:

```
WA=(((wv:noinfo@there.com),PA),((wv:he@there.com,He),CS),(((wv:friend@there.com),http://124.124.12
4.124:80/IMPSAPP),FS),(((wv:she@there.com,She),http://123.123.123.123:80/IMPSAPP),FS))
```

## 7.12.8 OtherServer parameter

The syntax of the OtherServer parameter differs from the syntax defined in the DTD. The DTD allows having single URL, single MSISDN, or coupled MSISDNs and URLs in the OtherServer element. In order to keep the coupling and eliminate the need for three different OtherServer elements in PTS encoding, both URL and MSISDN are considered MANDATORY parameters meaning that URL and MSISDN always come in pairs, and when a URL or an MSISDN is missing from the pair, its empty element is indicated.

The syntax is:

`OS=(<OtherServerList>)` ,where <OtherServerList> is comma separated list of one or more <OtherServer> elements.

An <OtherServer> element is <URL>, <MSISDN> pair in parentheses, separated by comma: (<URL>,<MSISDN>)

Either <URL> or <MSISDN> can be empty (but not both), in which case the <OtherServer> is either (,<MSISDN>) or (<URL>,).

`OS=((<URL`$_1$`>,),(,<MSISDN`$_2$`>),(<URL`$_n$`>,<MSISDN`$_n$`>))`

# 8. Extension Framework

## 8.1 Extending Existing Primitives

Extended blocks MAY be added to any primitive using the ExtBlock - "EB" - parameter. This parameter includes a namespace element and the actual extension data. The namespace element MAY contain a value that points to a non-existing content.

For the syntax of the ExtBlock parameter please refer to chapter 7.11.10 - ExtBlock and ExtBlockETEM parameter on page 35.

## 8.2 Introducing New Primitives

New primitives MAY be introduced using the "XR" and "RX" primitives as follows:

```
<Preamble> <ExtNamespace> <ExtParameters>          , where
```

- The <Preamble> is defined in chapter 5 - Plain Text Syntax on page 12.
- The <ExtNamespace> MUST be the very first parameter within the primitive. The namespace element MAY contain a value that points to a non-existing content.
- The contents and syntax of <ExtParameters> is outside the scope of OMA IMPS.

Example:

```
WV13XR761 NS=http://www.foo.com/1.3AA=Some_Data BB=More_Data
```

# Appendix A. Change History (Informative)

## A.1 Approved Version History

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

## A.2 Draft/Candidate Version 1.3 History

| Document Identifier | Date | Sections | Description |
|---|---|---|---|
| Draft Versions<br>OMA-IMPS-WV-CSP_PTS-V1_3 | 31 Dec 2004 | All | - Taken OMA-IMPS-Enabler-Package-V1_2-20041217-C package as baseline.<br>- Updated references to 1.3 specs - also removed SCR references as those documents are going to be discontinued. Move the CSP and CSP Transport references from informative to normative section.<br>- Removed WV tags from document IDs.<br>- Replaced SMS references from 1.2 to 1.3, and updated the examples as well.<br>- Document renamed from SMS Binding (SMS) to Plain Text Syntax (PTS).<br>- - Added CR OMA-IMPS-2003-0106-COMVERSE-NAT-UDP_CIR. |
| | 7 Jan 2005 | All | - Changed the SCR section to OMA preferred. |
| | 28 Jan 2005 | All | - Updated the references.<br>- Added approved CRs:<br>OMA-IM-2005-0014R01-MoveStoS |
| Draft Versions<br>OMA-IMPS-CSP_PTS-V1_3 | 04 Feb 2005 | All | - Applied the OMA 2005 template. |
| | 29 Mar 2005 | All | Added approved CRs:<br>OMA-IM-2005-0039-FriendlyName<br>OMA-IM-2005-0066R02-IMPS-1.3-CSP-Plain-Text-Syntax-Changes-for-Content-Font-Formatting<br>OMA-IM-2005-0116-IMPS13_PrAuth-CSP_PTS |
| | 04 Apr 2005 | All | Added approved CRs:<br>- OMA-IM-2005-0142-IMPS13_ShowMap-CSP_PTS<br>- OMA-IM-2005-0159-IMPS13-Add-Contact-By-User-ID-PTS<br>Accepted deletion of old TOCs to shorten the document.<br>Updated reference to the approved 1.2 enabler package. |
| | 15 Apr 2005 | All | Added approved CRs:<br>- OMA-IM-2005-0173-IMPS13_GenNoticeAddUser_CSP_PTS<br>- OMA-IM-2005-0185-IMPS13_AdvSrc_CSP_PTS<br>- OMA-IM-2005-0202-IMPS13_PublicProfile_CSP_DTYPE<br>- OMA-IM-2005-0215-IMPS13_LowMem_CSP_PTS<br>Accepted deletion of old TOCs and examples to shorten the document. |
| | 24 Apr 2005 | All | Added approved CRs:<br>- OMA-IM-2005-0263-IMPS13_SysMsgOZContinue_CSP_PTS<br>- OMA-IM-2005-0273-IMPS13_OffMsgSiemensContinue_CSP_PTS<br>- OMA-IM-2005-0278-IMPS13-Extend-IM-and-PGC-PTS<br>- OMA-IM-2005-0247-IMPS13_MsgCapBound_CSP_PTS<br>- OMA-IM-2005-0296-IMPS13_MultiSessionApp_CSP_PTS (Changed ClientID presence code to CH due to conflict.)<br>- OMA-IM-2005-0255-PTS-ADC2 (Changed AuthorizeAndGrant code to AH due to conflict.)<br>Accepted deletion of old TOCs to shorten the document. |
| | 24 Apr 2005 | All | Added approved CRs:<br>- OMA-IM-2005-0305-IMPS13_PrAuth_CSP_PTS<br>Accepted deletion of old TOCs to shorten the document. |

| Document Identifier | Date | Sections | Description |
|---|---|---|---|
| Draft Versions<br>OMA-TS-IMPS-CSP_PTS-V1_3 | 1 Jun 2005 | All | Fixed copyright date on cover page.<br>Changed “C0 Code0” font size from 10 to 8.<br>Removed unused references from normative references.<br>Added -TS- to document name. |
| | 11 Jun 2005 | All | Added agreed change requests:<br>- OMA-IM-2005-0338-CONRR_Correction_CSP_PTS<br>- OMA-IM-2005-0361-IMPS13-ArchRef-CSP_PTS<br>Removed references to CLP and MD6. |
| | 20 Jun 2005 | All | Addressed all editorial changes as AIs instructed in OMA-IM-2005-0419-MeetingMinutes-SanDiego-13June2005<br>Accepted TOC update change to shorten the document. |
| | 12 Jul 2005 | All | Added agreed change requests:<br>- OMA-IM-2005-0455-IMPS13_CIRSMSAddress_CSP_PTS<br>- OMA-IM-2005-0456-IMPS13_MSISDNFix-CSP_PTS<br>- OMA-IM-2005-0478-IMPS-13-PTS-remove-RA<br>Added table references.<br>Accepted field code updates to shorten the document. |
| | 27 Jul 2005 | All | Added agreed change requests:<br>- OMA-IM-2005-0524-IMPS13_DocUpdateAI-CSP_PTS<br>- OMA-IM-2005-0531-IMPS-13-PTS-MulitTransPerMessage<br>- OMA-IM-2005-0543-IMPS13_AIBasedUp-CSP_PTS<br>Accepted field code updates to shorten the document. |
| | 08 August 2005 | All | Added agreed change requests:<br>- OMA-IM-2005-0539-IMPS-13-PTS-DataTypes-MaxWatcherList<br>- OMA-IM-2005-0558-MPS13_AIs_CSP_PTS<br>- OMA-IM-2005-0576-IMPS13_SysMsgReqResp_CSP-PTS<br>- OMA-IM-2005-0585-IMPS-1_3-CSP-PTS-autosub<br>Accepted field code updates to shorten the document. |
| | 25 Aug 2005 | All | Fixed additional inconsistencies reported by e-mail (editorials):<br>- OMA document references,<br>Added agreed change requests:<br>- OMA-IM-2005-0628-IMPS13_SearchOnAge-CSP_PTS<br>Accepted field code updates to shorten the document. |
| | 09 Sep 2005 | All | Added dash to SystemMessage primitive names.<br>Added FriendlyName example to Search-Results.<br>Added FriendlyName to Watcher (also in examples).<br>Corrected GetMessageRequest and GetMessageResponse codes in examples.<br>Changed User-ID-List to User-List in NotificationRequest, GetGroupMembersResponse and RejectListResponse. |
| | 16 Sep 2005 | All | Added agreed change requests:<br>- OMA-IM-2005-0653-IMPS-13-ExtendConversationFixes-PTS<br>- OMA-IM-2005-0657R01-IMPS-13-JoinedBlockedUsersFix-PTS |
| Candidate Versions<br>OMA-TS-IMPS-CSP_PTS-V1_3 | 11 Oct 2005 | n/a | Status changed to Candidate by TP<br>TP ref # OMA-TP-2005-0279R01-IMPS-V1_3-for-Candidate-approval |

# Appendix B. Static Conformance Requirements (Normative)

The notation used in this section is specified in [IOPPROC].

## B.1 SMS Encoding Requirements

### B.1.1 Clients

| Item | Function | Reference | Status | Requirement |
|---|---|---|---|---|
| IMPS-CSP-PTS-C-001 | All primitive names, parameter names, service tree elements, presence attributes, presence values, group properties, contact list properties and search elements in a primitive are encoded with the short code(s). | 6 and 7 | M | |

### B.1.2 Servers

| Item | Function | Reference | Status | Requirement |
|---|---|---|---|---|
| IMPS-CSP-PTS-S-001 | All primitive names, parameter names, service tree elements, presence attributes, presence values, group properties, contact list properties and search elements in a primitive are encoded with the short code(s). | 6 and 7 | M | |

# Appendix C. Plain Text Syntax Examples (Informative)

## C.1 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(201,”Partially completed.”) DU=((531,”Unknown
user.”,wv:bad_user1@im.com,wv:bad_user2@im.com),(532,Blocked.,wv:bad_user3@im.com,wv:bad_user4@im.
com)) DN=30
```

## C.2 PollingRequest primitive

```
WV13PO761 SI=im.user.com#48815@server.com
```

## C.3 Version discovery transaction

### C.3.1 VersionDiscoveryRequest primitive

```
WV13VD761
```

### C.3.2 VersionDiscoveryResponse primitive

```
WV13DV761 VL=(11,12)
OS=((http://www.otherserver.com,+123456789),(,+987654321),(http://206.226.10.25:80/IMPSAPP,))
```

## C.4 2-Way Login transaction

### C.4.1 LoginRequest primitive

```
WV13LR761 UI=wv:john@smith.com CI=http://123.123.123.123:80/IMPSAPP PW=this1is2my3pass
SC=im.user.com#20011224#328746293 TL=600
```

### C.4.2 LoginResponse primitive

```
WV13RL761 CI=http://123.123.123.123:80/IMPSAPP ST=(200,”Successfully completed.”)
SI=im.user.com#48815@server.com
KA=300 CR=T
```

## C.5 4-way Login transaction

### C.5.1 LoginRequest primitive

```
WV13LR761 UI=wv:john@smith.com CI=http://123.123.123.123:80/IMPSAPP SH=(PWD,SHA,MD4,MD5)
SC=im.user.com#20011224#328746293
```

### C.5.2 LoginResponse primitive

```
WV13RL761 CI=http://123.123.123.123:80/IMPSAPP ST=(401,”Further authorization required”)
NO=92387rhf934fho3fh9fkn309fn3pfun304ufn3 DI=MD5 CR=F
```

### C.5.3 LoginRequest primitive

```
WV13LR762 UI=wv:john@smith.com CI=http://123.123.123.123:80/IMPSAPP
DB=alkkuayfdsAKDSJfsdfjhksadhlkasdlkfgsal TL=600 SC=im.user.com#20011224#328746293
```

### C.5.4 LoginResponse primitive

```
WV13RL762 CI=http://123.123.123.123:80/IMPSAPP ST=(200,”Successfully logged in.”)
SI=im.user.com#48815@server.com KA=300 CR=T
```

## C.6 Client capability negotiation transaction

Note that the transaction is simplified when the underlying transport is SMS:

The request or the response cannot contain other elements than those described in [CSP].

### C.6.1 ClientCapabilityRequest primitive

```
WV13CP761 SI=im.user.com#48815@server.com
CA=((AU,1024),(AT,512),(CT,MP),(DL,fin),(MT,5),(MP,5),(PS,2048),(SB,(SMS,WSP,HTTP)),(SP=0),(OE,RE)
)
```

### C.6.2 ClientCapabilityResponse primitive

```
WV13PC761 SI=im.user.com#48815@server.com AP=((SB,SMS),(SP=0))
```

## C.7 Logout transaction

### C.7.1 LogoutRequest primitive

```
WV13OR761 SI=im.user.com#48815@server.com
```

### C.7.2 Status primitive

```
WV13DI761 SI=im.user.com#48815@server.com ST=200
```

## C.8 Server initiated logout transaction

### C.8.1 Disconnect primitive

```
WV13DI SI=im.user.com#48815@server.com ST=(601,”Updating server software. All services offline for
3 hours.”)
```

## C.9 Keep Alive transaction

### C.9.1 KeepAliveRequest primitive

**WV13KA761 SI=im.user.com#48815@server.com TL=600**

### C.9.2 KeepAliveResponse primitive

```
WV13AK761 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)KA=600
```

## C.10 Get service provider info transaction

### C.10.1 GetSPInfoRequest primitive

```
WV13GS761 SI=im.user.com#48815@server.com
```

### C.10.2 GetSPInfoResponse primitive

MMS messages are not allowed, thus the Logo element is not present in the primitive.

```
WV13SG761 SI=im.user.com#48815@server.com NA="Wireless Village" TX="This is OMA's IMPS test
service" UR="http://www.openmobilealliance.org"
```

## C.11 Service Negotiation transaction

The following example illustrates service negotiation with the following parameters:

Client requests:

- All Fundamental Features,
- All IM Features,
- All Presence Features,
- None of the Group Features.

Server does not agree to provide:

- Fundamental features,
- GetWatcherList transaction,
- IM Authorization Functions.

### C.11.1 ServiceRequest primitive

```
WV13SQ761 SI=im.user.com#48815@server.com RF=(FF,IF,PF) AR=F
```

### C.11.2 ServiceResponse primitive

```
WV13QS761 SI=im.user.com#48815@server.com  NF=(FF,GW,IA)
```

## C.12 Segmentation mechanism example

The following example demonstrates how a client retrieves a long grant and block list in segments from the server – assuming a block list (40 users) in use and a grant list (10 users) not in use. Note that in order to make the example shorter, not all users are listed. Notice the difference between the SMS transport-specific concatenation and the segmentation mechanism.

### C.12.1 GetBlockedListRequest primitive

```
WV13GB761 SI=im.user.com#48815@server.com
```

### C.12.2 GetBlockedListResponse primitive

```
WV13BG761ab SL=(2,(761,0)) SI=im.user.com#48815@server.com
BL=((wv:blockedusr01@server.com,wv:blockedusr17@server.c

WV13BG761bb om,wv:blockedusr30@server.com)) BU=T GL=wv:grantedusr01@server.com GU=F
```

### C.12.3 GetSegmentRequest primitive

```
WV13GE762 SI=im.user.com#48815@server.com SK=(761,1)
```

### C.12.4 GetSegmentResponse primitive

```
WV13EG762ab SI=im.user.com#48815@server.com
BL=((wv:blockedusr31@server.com,wv:blockedusr40@server.com))
GL=((wv:grantedusr02@server.com,wv:grantedusr09@server

WV13EG762bb .com,wv:grantedusr10@server.com))
```

## C.13 System Message transactions

### C.13.1 SystemMessageRequest primitive

```
WV13SY761 SI=im.user.com#48815@server.com SQ=((0x1234,T,"These are the offered services, and MUST
You agree to these and these terms before use of the service.",,"To confirm, please enter the last
four digits of Your social security number."),(0x1235,T,"You must accept the costs of the service,
which are described in a table available on the following URL: http://www.operator-specific-
url.com To confirm, please visit the designated web page, and follow the instructions.",,
http://www.operator-specific-url.com/verifythis),(0x1236,T,"You have to choose a User-ID. Please
select one from the
list.",((0,john.smith@there.com),(1,johnsmith@there.com),(2,jsmith@there.com),(3,
john01@there.com)),"To confirm, please enter the last four digits of Your social security
number."))
```

### C.13.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,"Successfully completed.")
```

### C.13.3 SystemMessageUser primitive

```
WV13YS762 SI=im.user.com#48815@server.com SV=((0x1234,,0000),(0x1235,,aX9b8d6Q3),(0x1236,0,0000))
```

### C.13.4 Status primitive

```
WV13ST762 SI=im.user.com#48815@server.com ST=200
```

## C.14 General notification transactions

### C.14.1 SubscribeNotificationRequest primitive

```
WV13SN761 SI=im.user.com#48815@server.com NL=(ATCL,CLC,GR)
```

### C.14.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,"Successfully completed.")
```

### C.14.3 NotificationRequest primitive

```
WV13NR761 SI=im.user.com#48815@server.com NT=ATCL US=((wv:john@smith.com,Johnnie))
```

### C.14.4 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=200
```

### C.14.5 UnsubscribeNotificationRequest primitive

```
WV13UN761 SI=im.user.com#48815@server.com NL=GR
```

### C.14.6 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,"Ok.")
```

## C.15 Retrieve Public Profile transaction

### C.15.1 GetPublicProfileRequest primitive

```
WV13GU761 SI=im.user.com#48815@server.com UE=(wv:he@there.com,wv:non.exsiting-
user@server.com)
```

### C.15.2 GetPublicProfileResponse primitive

```
WV13UG761ac SI=im.user.com#48815@server.com ST=(201,"Partially completed.") DU=((531,"Unknown
user",wv:non.exsiting-user@server.com) PP=((wv:he@there.com,(UG,19

WV13UG761bc 8001),(UO,FI),(UN,"John The Great, II"),(UC,"Paradise City"),(UR,M),(UT,"Chat about
high-tech cars and fishing techniques"),(UH,"Online chat, cars,

WV13UG761cc fishing"),(US,M)))
```

## C.16 Update Public Profile transaction

### C.16.1 UpdatePublicProfileRequest primitive

```
WV13UR762 SI=im.user.com#48815@server.com CE=F PP=((,(UN,"John The Fisher")))
```

### C.16.2 Status primitive

```
WV13ST762 SI=im.user.com#48815@server.com ST=(200,"Successfully completed.")
```

## C.17 Search transaction

### C.17.1 SearchRequest primitive (1st)

```
WV13SR761 SI=im.user.com#48815@server.com SP=((UL,Smith),(UO,T)) SL=5
```

### C.17.2 SearchResponse primitive (1st)

```
WV13RS761 SI=im.user.com#48815@server.com SF=7 CF=F SX=6 SD=112233 SR=((wv:john,"John The Great,
II"),wv:smithy@village.com,wv:smith@wezzon.com,wv:locksmith@site.org,wv:smithereens@car.org)
```

### C.17.3 SearchRequest primitive (continued)

```
WV13SR762 SI=im.user.com#48815@server.com SD=112233 SX=6
```

### C.17.4 SearchResponse primitive (continued)

```
WV13RS762 SI=im.user.com#48815@server.com SF=7 CF=T SX=7
SR=(wv:tinsmith@home.se,wv:coppersmith@bigfish.com)
```

# C.18 Stop search transactions

## C.18.1 StopSearchRequest primitive

```
WV13SS763 SI=im.user.com#48815@server.com SD=112233
```

## C.18.2 Status primitive

```
WV13ST763 SI=im.user.com#48815@server.com ST=200
```

# C.19 Advanced search transaction

## C.19.1 SearchRequest primitive (1st)

```
WV13SR761 SI=im.user.com#48815@server.com SP=((UL,Smith,0),(UF,John,1),(UF,Johnny,2)) AI=0+[1|2]
SL=5
```

## C.19.2 SearchResponse primitive (1st)

```
WV13RS761 SI=im.user.com#48815@server.com SF=1 CF=T SX=1 SD=112233
SR=wv:john,wv:smithy@village.com
```

## C.19.3 StopSearchRequest primitive

```
WV13SS762 SI=im.user.com#48815@server.com SD=112233
```

## C.19.4 Status primitive

```
WV13ST762 SI=im.user.com#48815@server.com ST=200
```

# C.20 Invitation transactions

## C.20.1 InviteRequest primitive

```
WV13IR761 SI=im.user.com#48815@server.com II=11 IT=PR PS=(OS,TZ,FT)
SE=((wv:john@smith.com,,http://206.226.10.25:80/IMPSAPP))
RE=(wv:lara.naval@secret.gov,wv:francisco) IR=”Feel free to use my presence infos!”
```

## C.20.2 Status primitive

```
WV13ST761 ST=200
```

## C.20.3 InviteUserRequest primitive

```
WV13IU762 SI=54321 II=11 IT=PR SE=((wv:john@smith.com,Johnnie,http://206.226.10.25:80/IMPSAPP))
RE=((wv:Francisco,Francis)) PS=(OS,TZ,FT) IR=”Feel free to use my presence infos!”
```

## C.20.4 Status primitive

```
WV13ST762 SI=54321 ST=200
```

## C.20.5 InviteUserResponse primitive

```
WV13UI763 SI=54321 II=11 AC=T IX=”Thanks a lot!” SE=wv:francisco
RE=((wv:john@smith.com,,http://206.226.10.25:80/IMPSAPP)) SN=((john,wv:%2Fchatgroup@wv.com))
```

### C.20.6 Status primitive

```
WV13ST763 SI=54321 ST=200
```

### C.20.7 InviteResponse primitive

```
WV13RI764 SI=im.user.com#48815@server.com II=11 SE=((wv:francisco,Francis))
RE=((wv:john@smith.com,Johnnie,http://206.226.10.25:80/IMPSAPP)) AC=T IX="Thanks a lot!"
```

### C.20.8 Status primitive

```
WV13ST764 ST=200
```

## C.21 Canceling invitation transactions

### C.21.1 CancelInviteRequest primitive

```
WV13CI761 SI=im.user.com#48815@server.com II=11 SE=wv:john@smith.com
RE=(wv:lara.naval@secret.gov,wv:francisco) RR="I will be on vacation for 1 week."
```

### C.21.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=200
```

### C.21.3 CancelInviteUserRequest primitive

```
WV13CU762 SI=54321 II=11 SE=((wv:john@smith.com,Johnnie)) RE=((wv:Francisco,Francis)) RR="I will
be on vacation for a week."
```

### C.21.4 Status primitive

```
WV13ST762 SI=54321 ST=200
```

## C.22 Verify ID transaction

### C.22.1 VerifyIDRequest primitive

```
WV13VI761 SI=im.user.com#48815@server.com IU=(john,pam/friends,wv:pam/friends@outofmynet.com)
IC=/friends IG=/managers@outofmynet ID=baddomain.com
```

### C.22.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=201 DU=(531,"Unknown
user.",wv:john@mynet.com,wv:pam/friends@mynet.com,pam/friends@outofmynet.com) DG=(200,"Group
exists.",/managers@outofmynet.com) DK=(700,"Contact list does not exist.",/friends@mynet.com)
DD=(404,"Domain name not found.",baddomain.com)
```

## C.23 Get list of contact list IDs transaction

### C.23.1 GetListRequest primitive

```
WV13GL761 SI=im.user.com#48815@server.com
```

### C.23.2 GetListResponse primitive

```
WV13LG761 SI=im.user.com#48815@server.com CO=(wv:john/colleagues,wv:john/friends)
DC=wv:john/family
```

## C.24 Create contact list transaction

### C.24.1 CreateListRequest primitive

```
WV13CL761 SI=im.user.com#48815@server.com CL=wv:john/friends UN=((”New
friend”,wv:new@friend.org),(,wv:no.nick@name.com)) CP=((DN,”My friends”),(DE,T))
```

### C.24.2 CreateListResponse primitive

```
WV13LC761 SI=im.user.com#48815@server.com CL=wv:john/friends CP=((DN,”My friends”),(DE,T),(DO,F))
```

## C.25 Delete contact list transaction

### C.25.1 DeleteListRequest primitive

```
WV13DL761 SI=im.user.com#48815@server.com CL=wv:john/friends
```

### C.25.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

## C.26 Retrieve a contact list transaction

### C.26.1 ListManageRequest primitive

```
WV13LM761 SI=im.user.com#48815@server.com CL=wv:john/friends RL=T
```

### C.26.2 ListManageResponse primitive

```
WV13ML761 SI=im.user.com#48815@server.com ST=200 CP=((DN,”My friends”),(DE,T),(DO,F)) UN=((”New
friend”,wv:new@friend.org),(,wv:no.nick@name.com))
```

## C.27 Add users to a contact list transaction

### C.27.1 ListManageRequest primitive

```
WV13LM761 SI=im.user.com#48815@server.com CL=wv:john/friends AN=((“Randall the
Vandal”,wv:randall@fairlane.com),(,wv:no.nick@name.com),(Brainstrom,wv:bright@dark.com)) RL=T
```

### C.27.2 ListManageResponse primitive

```
WV13ML761 SI=im.user.com#48815@server.com ST=200 UN=((“Randall the
Vandal”,wv:randall@fairlane.com),(,wv:no.nick@name.com),(Brainstrom,wv:bright@dark.com),(”New
friend”,wv:new@friend.org))
```

# C.28 Remove users from a contact list

## C.28.1 ListManageRequest primitive

```
WV13LM761 SI=im.user.com#48815@server.com CL=wv:john/friends RN=((”New friend”,wv:new@friend.org))
RL=T
```

## C.28.2 ListManageResponse primitive

```
WV13ML761 SI=im.user.com#48815@server.com ST=200 UN=((“Randall the
Vandal”,wv:randall@fairlane.com),(,wv:no.nick@name.com),(Brainstrom,wv:bright@dark.com))
```

# C.29 Modify properties of contact list transaction

## C.29.1 ListManageRequest primitive

```
WV13LM761 SI=im.user.com#48815@server.com CL=wv:john/friends CP=((DN,”My enemies”),(DE,T),(DO,F))
RL=F
```

## C.29.2 ListManageResponse primitive

```
WV13ML761 SI=im.user.com#48815@server.com ST=200 CP=((DN,”My enemies”),(DE,T),(DO,F))
```

# C.30 Create attribute list transaction

## C.30.1 CreateAttributeListRequest primitive

```
WV13CA761 SI=im.user.com#48815@server.com PS=(OS,TZ,FT)
UE=(wv:matthias@salamander.com,wv:francisco) CO=wv:john/friends DL=T DY=F UY=T CY=T
```

## C.30.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

# C.31 Delete attribute list transaction

## C.31.1 DeleteAttributeListRequest primitive

```
WV13DA761 SI=im.user.com#48815@server.com UE=(wv:matthias@salamander.com,wv:francisco)
CO=wv:john/friends DL=F
```

## C.31.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=200
```

# C.32 Get attribute list(s) transaction

## C.32.1 GetAttributeListRequest primitive

```
WV13GA761 SI=im.user.com#48815@server.com DL=T
```

### C.32.2 GetAttributeListResponse primitive

```
WV13AG761 SI=im.user.com#48815@server.com ST=200
PC=((wv:john/colleagues,T,OS),(wv:john/family,F,(OS,FT)))
PU=((wv:john@smith.com,F,UA),(wv:matthias@salamander.com,F,(OS,PL))) DA=OS
```

## C.33 Subscribe/unsubscribe presence transaction

### C.33.1 SubscribePresenceRequest primitive

```
WV13SB761 SI=im.user.com#48815@server.com UE=(wv:matthias@salamander.com,wv:francisco)
CO=wv:john/family PS=OS
```

### C.33.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

### C.33.3 PresenceNotificationRequest primitive

```
WV13PN761 SI=im.user.com#48815@server.com
PR=((wv:matthias@salamander.com,((OS,T,((PV,T),(CH,http://123.123.123.123:80/IMPSAPP))),(FT,T,((PV
,”At
home”),(CI,http://123.123.123.123:80/IMPSAPP))),(wv:francisco,((OS,T,((PV,T),(CI,http://100.100.10
0.100:80/IMPSAPP))))))
```

### C.33.4 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=200
```

### C.33.5 UnsubscribePresenceRequest primitive

```
WV13PS761 SI=im.user.com#48815@server.com UE=(wv:matthias@salamander.com,wv:francisco)
CO=wv:john/family
```

### C.33.6 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

## C.34 Get watcher list transaction

### C.34.1 GetWatcherListRequest primitive

```
WV13GW761 SI=im.user.com#48815@server.com HP=0 MW=200
```

### C.34.2 GetWatcherListResponse primitive

```
WV13WG761 SI=im.user.com#48815@server.com HP=172800
WA=(((wv:noinfo@there.com),PA),((wv:he@there.com,He),CS),(((wv:friend@there.com),http://124.124.12
4.124:80/IMPSAPP),FS),(((wv:she@there.com,She),http://123.123.123.123:80/IMPSAPP),FS))
```

## C.35 Get presence transaction

### C.35.1 GetPresenceRequest primitive

```
WV13GP761 SI=im.user.com#48815@server.com UE=(wv:matthias,wv:francisco@don.com) PS=OS
```

### C.35.2 GetPresenceResponse primitive

```
WV13PG761 SI=im.user.com#48815@server.com ST=200
PR=((wv:matthias@salamander.com,((OS,T,((PV,T),(CI,http://123.123.123.123:80/IMPSAPP))))),(wv:fran
cisco,((OS,T,((PV,T),(CI,http://100.100.100.100:80/IMPSAPP))))))
```

## C.36 Update presence transaction

### C.36.1 UpdatePresenceRequest primitive

```
WV13UP761 SI=im.user.com#48815@server.com PS=((OS,T,T),(FT,T,"In the office"))
```

### C.36.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=200
```

## C.37 Send message transaction

### C.37.1 SendMessageRequest primitive

```
WV13SM761 SI=im.user.com#48815@server.com
MF=(,,,,36,(((wv:matthias@salamander.com,wv:francisco)),wv:john/colleagues,wv:/chatgroup@wv.com,((
"The boss",wv:/chatroup@wv.com))),(wv:me@home.com),,(Red,Big,(Bold,Italic))) DE=T MC="Hello
everybody! How You guys doing?"
```

### C.37.2 SendMessageResponse primitive

```
WV13MS761 SI=im.user.com#48815@server.com ST=(200,"Successfully completed.") MI=11235
```

## C.38 Pushing a message from the server transaction

### C.38.1 NewMessage primitive

```
WV13NM761 SI=im.user.com#48815@server.com
MF=(11235,,,,36,,(((wv:john@smith.com,Johnnie))),(wv:me@home.com),20011118T1203Z,(Red,Big,(Bold,It
alic)),3600) MC="Hello everybody! How You guys doing?"
```

### C.38.2 MessageDelivered primitive

```
WV13MD761 SI=im.user.com#48815@server.com MI=11235
```

## C.39 Get message list transaction

### C.39.1 GetMessageListRequest primitive

```
WV13MR761 SI=im.user.com#48815@server.com GI=wv:/chatgroup@wv.com MN=5
```

### C.39.2 GetMessageListResponse primitive

Only message-IDs are carried in the primitive.

```
WV13RM761 SI=im.user.com#48815@server.com
ML=((11236,,,,27,,(((wv:john@smith.com,Johnnie))),(wv:me@home.com),20011118T1203Z,(Red,Big,(Bold,I
talic)),3600),(11237,,,,70,,(((wv:john@smith.com,Johnnie))),(wv:me@home.com),2001118T1205Z,(Red,Bi
g,(Bold,Italic)),3600),(11238,,,,50,,(((wv:john@smith.com,Johnnie))),(wv:me@home.com),20011118T120
7Z,(Red,Big,(Bold,Italic)),3600))
```

# C.40 Retrieving a message from the server transaction

## C.40.1 GetMessageRequest primitive

```
WV13GX761 SI=im.user.com#48815@server.com MI=11236
```

## C.40.2 GetMessageResponse primitive

```
WV13MX761 SI=im.user.com#48815@server.com
MF=(11236,,,,27,,(((wv:john@smith.com,Johnnie))),(wv:me@home.com),20011118T1203Z,(Red,Big,(Bold,It
alic)),3600) MC="Thanks, everything is fine."
```

## C.40.3 MessageDelivered primitive

```
WV13NM761 SI=im.user.com#48815@server.com MI=11236
```

## C.40.4 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=200
```

# C.41 Delivery status report transaction

## C.41.1 DeliveryReportRequest primitive

```
WV13DR761 SI=im.user.com#48815@server.com ST=200 DX=20011118T1204Z
MF=(11235,,,,36,(((wv:matthias@salamander.com,wv:francisco)),wv:john/colleagues,wv:/chatgroup@wv.c
om,(("The boss",wv:/chatroup@wv.com))),(wv:me@home.com),,(Red,Big,(Bold,Italic)))
```

## C.41.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=200
```

# C.42 Extend conversation transaction

Note that the example does not include the additional signaling required.

## C.42.1 ExtendConversionRequest primitive

```
WV13EC761 SI=im.user.com#48815@server.com UE=(wv:bob@server.com,wv:alice@server.com) EI=0x42266335
EU=wv:tom@server.com SA=T WT="Welcome to my chat" ON=((Bobby,wv:/dummygroup@wv.com))
```

## C.42.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,"Successfully completed.")
```

## C.42.3 ExtendConversionResponse primitive

```
WV13CE762 SI=im.user.com#48815@server.com EI=0x42266335 GI=wv:/conversation684331@server.com
IX="Thanks a lot!"
```

## C.42.4 Status primitive

```
WV13ST762 SI=im.user.com#48815@server.com ST=(200,"Successfully completed.")
```

# C.43 Get blocked user list transaction

## C.43.1 GetBlockedListRequest primitive

```
WV13GB761 SI=im.user.com#48815@server.com
```

## C.43.2 GetBlockedListResponse primitive

```
WV13BG761ab SI=im.user.com#48815@server.com
BL=((wv:he@there.com,wv:she@there.com),wv:john/colleagues,wv:/chatgroup@nowhere.com,((“The
boss”,wv:/othergroup@somewhere.com)),*chessgame*) BU=T
GL=((wv:matthias@salamander.com,wv:francisco),,wv:/rock@roll.com,((Talkative,wv:/nowhere@there.org
))) GU=F
```

# C.44 Block entity transaction

## C.44.1 BlockEntityRequest primitive

```
WV13BE761 SI=im.user.com#48815@server.com GA=(,wv:john/colleagues) BU=F GU=T
```

## C.44.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

# C.45 Create group transaction

## C.45.1 CreateGroupRequest primitive

```
WV13CG761ab SI=im.user.com#48815@server.com GI=wv:john/private@there.com GP=((NM,”Chit chat
group”),(AT,Restricted),(PM,T),(SE,F),(TO,”Family, relation ships”),(MU

WV13CG761bb ,30),(WN,”Welcome to my group. Feel free to discuss about our current
topic.”),(AD,T),(VL,60)) JG=F SA=F
```

## C.45.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=200
```

# C.46 Delete group transaction

## C.46.1 DeleteGroupRequest primitive

```
WV13DG761 SI=im.user.com#48815@server.com GI=wv:john/private@there.com
```

## C.46.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=200
```

# C.47 Join Group transaction

## C.47.1 JoinGroupRequest primitive

```
WV13JG761 SI=im.user.com#48815@server.com GI=wv:/chatgroup@there.com SN=((”-=Bart Simpson=-
”,wv:/chatgroup@there.com)) JR=T SA=F
```

## C.47.2 JoinGroupResponse primitive

```
WV13GJ761 SI=im.user.com#48815@server.com JU=((Matthias,wv:mat@ny.net),”Francisco (of the
Dons)”,(Anonymous12,wv:anon@foo.com)) WT=”Welcome to WV!”
```

# C.48 User initiated leave group transaction

## C.48.1 LeaveGroupRequest primitive

```
WV13LU761 SI=im.user.com#48815@server.com GI=wv:/chatgroup@there.com
```

## C.48.2 LeaveGroupResponse primitive

```
WV13UL761 SI=im.user.com#48815@server.com ST=200 GI=wv:/chatgroup@there.com
```

# C.49 Server initiated leave group transaction

## C.49.1 LeaveGroupResponse primitive

```
WV13UL761 SI=im.user.com#48815@server.com ST=(809,”You have been rejected from the group.”)
GI=wv:/chatgroup@there.com
```

## C.49.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=200
```

# C.50 Get group members’ list transaction

## C.50.1 GetGroupMembersRequest primitive

```
WV13GM761 SI=im.user.com#48815@server.com GI=wv:john/private@there.com
```

## C.50.2 GetGroupMembersResponse primitive

```
WV13MG761 SI=im.user.com#48815@server.com AD=wv:john@smith.com
MO=(wv:matthias@salamander.com,wv:francisco) US=((wv:he@there.com,He),wv:she@there.com)
```

# C.51 Get Joined User’s list transaction

## C.51.1 GetJoinedUsersRequest primitive

```
WV09JU761 SI=im.user.com#48815@server.com GI=wv:/chatgroup@there.com
```

## C.51.2 GetJoinedUsersResponse primitive

```
WV09UJ761 SI=im.user.com#48815@server.com AA=((John,wv:john@smith.com))
AE=((He,wv:he@there.com),(She,wv:she@there.com)) JB=((She,wv:she@there.com))
```

# C.52 Add group member(s) transaction

## C.52.1 AddGroupMembersRequest primitive

```
WV13AM761 SI=im.user.com#48815@server.com GI=wv:john/private@there.com
UE=(wv:me@home.com,wv:you@there.com)
```

## C.52.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

# C.53 Remove group member(s) transaction

## C.53.1 RemoveGroupMembersRequest primitive

```
WV13RM761 SI=im.user.com#48815@server.com GI=wv:john/private@there.com
UE=(wv:me@home.com,wv:you@there.com)
```

## C.53.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=200
```

# C.54 Member access rights transaction

## C.54.1 MemberAccessRequest primitive

```
WV13ME761 SI=im.user.com#48815@server.com GI=wv:john/private@there.com
AD=(wv:matthias@salamander.com,wv:francisco) MO=(wv:he@there.com,wv:she@there.com)
UE=wv:john@smith.com
```

## C.54.2 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

# C.55 Modify group properties transactions

## C.55.1 GetGroupPropsRequest primitive

```
WV13GR761 SI=im.user.com#48815@server.com GI=wv:john/private@there.com
```

## C.55.2 GetGroupPropsResponse primitive

```
WV13RG761ab SI=im.user.com#48815@server.com GP=((NM,”Chit chat
group”),(AT,Restricted),(PM,T),(SE,F),(TO,”Family, relationships”),(MU,30),(WN,”Welcome to my
group. Feel free to discuss about our current topi

WV13RG761bb c.”),(AD,T),(VL,60)) OP=((PM,T),(PL,Admin),(IM,T),(AJ,F),(SI,F))
```

## C.55.3 SetGroupPropsRequest primitive

```
WV13SP762 SI=im.user.com#48815@server.com GI=wv:john/private@there.com OP=((PM,T))
```

## C.55.4 Status primitive

```
WV13ST762 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

# C.56 Rejected list transactions

## C.56.1 RejectListRequest primitive

```
WV13RE761 SI=im.user.com#48815@server.com GI=wv:john/private@there.com
AU=(wv:he@there.com,wv:she@there.com) RU=(wv:matthias@salamander.com,wv:francisco)
```

## C.56.2 RejectListResponse primitive

```
WV13ER761 SI=im.user.com#48815@server.com US=((wv:he@there.com,He),wv:she@there.com)
```

# C.57 Subscribe group change notification transaction

## C.57.1 SubscribeGroupNoticeRequest primitive (get)

```
WV13SU761 SI=im.user.com#48815@server.com GI=wv:/chatgroup@there.com SU=G
```

## C.57.2 SubscribeGroupNoticeResponse primitive

```
WV13US761 SI=im.user.com#48815@server.com SS=F
```

## C.57.3 SubscribeGroupNoticeRequest primitive (set)

```
WV13SG762 SI=im.user.com#48815@server.com GI=wv:/chatgroup@there.com SU=S
```

## C.57.4 Status primitive

```
WV13ST762 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

## C.57.5 Group change notification primitive

```
WV13GG761 SI=im.user.com#48815@server.com GI=wv:/chatgroup@there.com JU=(Matthias,Anonymous22)
LU=((Matthias,wv:/chatgroup@there.com),(”Francisco (of the Dons)”,wv:/chatgroup@there.com)
GP=((AU,8)) OP=((PL,Mod))
```

## C.57.6 Status primitive

```
WV13ST761 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

# C.58 Notification transaction

## C.58.1 SubscribeNotificationRequest primitive

```
WV13SN237 SI=im.user.com#48815@server.com NL=(AT,CC,GR)
```

## C.58.2 Status primitive

```
WV13ST237 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

## C.58.3 UnsubscribeNotificationRequest primitive

```
WV13UN284 SI=im.user.com#48815@server.com NL=(AT,CC,GR)
```

## C.58.4 Status primitive

```
WV13ST284 SI=im.user.com#48815@server.com ST=(200,”Successfully completed.”)
```

### C.58.5 NotificationRequest primitive

```
WV13NR387 SI=im.user.com#48815@server.com GI=wv:/somegroup@imps.com NT=GR
```

### C.58.6 Status primitive

```
WV13ST387 SI=im.user.com#48815@server.com ST=(200,"Successfully completed.")
```

## C.59 Example for multiple transactions

In the following example the client encapsulates 3 transactions into two Plain Text messages. The client:

- Responds to a request (Transaction-ID 700), and
- Requests to join a group (Transaction-ID 701). and
- Responds to another request (Transaction-ID 702).

1st SMS:

```
WV13ST700 SI=im.user.com#48815@server.com ST=(200,"Successfully completed.") & WV13JG701ab
SI=im.user.com#48815@server.com GI=wv:/chatgroup@there.com SN=(("-=
```

2nd SMS:

```
WV13JG701bb Bart Simpson=-",wv:/chatgroup@there.com)) JR=T SA=F & WV13ST702
SI=im.user.com#48815@server.com ST=200
```