



RESTful Network API for
Presence
Candidate Version 1.0 - 12 Feb 2013

Open Mobile Alliance
OMA-TS-REST_NetAPI_Presence-V1_0-20130212-C

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

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

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

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

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

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

© 2013 Open Mobile Alliance Ltd. All Rights Reserved.

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

Contents

1.	SCOPE.....	15
2.	REFERENCES	16
2.1	NORMATIVE REFERENCES.....	16
2.2	INFORMATIVE REFERENCES.....	17
3.	TERMINOLOGY AND CONVENTIONS.....	18
3.1	CONVENTIONS.....	18
3.2	DEFINITIONS.....	18
3.3	ABBREVIATIONS	18
4.	INTRODUCTION	20
4.1	VERSION 1.0	20
5.	PRESENCE API DEFINITION	21
5.1	RESOURCES SUMMARY	21
5.2	DATA TYPES	33
5.2.1	XML Namespaces.....	33
5.2.2	Structures	33
5.2.2.1	Type: <i>PresenceSourceList</i>	33
5.2.2.2	Type: <i>PresenceSource</i>	34
5.2.2.3	Type: <i>Presence</i>	36
5.2.2.4	Type: <i>PersonAttributes</i>	37
5.2.2.5	Type: <i>ServiceAttributes</i>	39
5.2.2.6	Type: <i>DeviceAttributes</i>	40
5.2.2.7	Type: <i>ContentList</i>	40
5.2.2.8	Type: <i>ContentData</i>	41
5.2.2.9	Type: <i>WatcherList</i>	41
5.2.2.10	Type: <i>Watcher</i>	41
5.2.2.11	Type: <i>RuleList</i>	42
5.2.2.12	Type: <i>Rule</i>	43
5.2.2.13	Type: <i>PresenceList</i>	44
5.2.2.14	Type: <i>PresenceContact</i>	44
5.2.2.15	Type: <i>SubscriptionList</i>	45
5.2.2.16	Type: <i>WatchersSubscriptionList</i>	45
5.2.2.17	Type: <i>WatchersSubscription</i>	46
5.2.2.18	Type: <i>WatchersNotification</i>	47
5.2.2.19	Type: <i>PresenceSubscriptionList</i>	47
5.2.2.20	Type: <i>PresenceSubscription</i>	48
5.2.2.21	Type: <i>PresenceNotification</i>	49
5.2.2.22	Type: <i>PresenceListSubscriptionCollection</i>	49
5.2.2.23	Type: <i>PresenceListSubscription</i>	50
5.2.2.24	Type: <i>PresenceListNotification</i>	52
5.2.2.25	Type: <i>Activities</i>	52
5.2.2.26	Type: <i>PlaceType</i>	53
5.2.2.27	Type: <i>Privacy</i>	53
5.2.2.28	Type: <i>Sphere</i>	53
5.2.2.29	Type: <i>Mood</i>	54
5.2.2.30	Type: <i>PlaceIs</i>	54
5.2.2.31	Type: <i>TimeOffset</i>	54
5.2.2.32	Type: <i>StatusIcon</i>	54
5.2.2.33	Type: <i>NoteList</i>	55
5.2.2.34	Type: <i>Location</i>	55
5.2.2.35	Type: <i>CircleData</i>	55
5.2.2.36	Type: <i>CivicAddress</i>	56
5.2.2.37	Type: <i>OverridingWillingness</i>	56
5.2.2.38	Type: <i>LinkList</i>	57
5.2.2.39	Type: <i>Contact</i>	58
5.2.2.40	Type: <i>DeviceIdentityList</i>	58
5.2.2.41	Type: <i>NetworkAvailability</i>	58
5.2.2.42	Type: <i>Network</i>	58

5.2.2.43	Type: <i>ExtendedList</i>	58
5.2.2.44	Type: <i>AttributeValue</i>	59
5.2.3	Enumerations.....	60
5.2.3.1	Enumeration: <i>ActivityValue</i>	60
5.2.3.2	Enumeration: <i>PlaceTypeValue</i>	61
5.2.3.3	Enumeration: <i>PrivacyValue</i>	62
5.2.3.4	Enumeration: <i>SphereValue</i>	62
5.2.3.5	Enumeration: <i>MoodValue</i>	62
5.2.3.6	Enumeration: <i>PlaceIsAudio</i>	63
5.2.3.7	Enumeration: <i>PlaceIsVideo</i>	64
5.2.3.8	Enumeration: <i>PlaceIsText</i>	64
5.2.3.9	Enumeration: <i>OpenOrClosed</i>	64
5.2.3.10	Enumeration: <i>ActiveOrTerminated</i>	65
5.2.3.11	Enumeration: <i>AutomaticOrManual</i>	65
5.2.3.12	Enumeration: <i>HomeOrVisited</i>	65
5.2.3.13	Enumeration: <i>ResourceStatus</i>	65
5.2.3.14	Enumeration: <i>DefaultDecisionValue</i>	66
5.2.4	Values of the Link “rel” attribute.....	66
5.3	SEQUENCE DIAGRAMS.....	67
5.3.1	Application start-up; publish presence, fetch Watcher information, subscribe to Watcher information.....	67
5.3.2	Adding a Watcher; subscribe for presence and updating of presence information.....	70
5.3.3	Update of presence status.....	72
5.3.4	Shutdown; remove resources.....	74
6.	DETAILED SPECIFICATION OF THE RESOURCES.....	76
6.1	RESOURCE: PRESENCE SOURCES.....	76
6.1.1	Request URL variables.....	76
6.1.2	Response Codes and Error Handling.....	77
6.1.3	GET.....	77
6.1.3.1	Example: <i>retrieving all Presence Sources for user (Informative)</i>	77
6.1.3.1.1	Request.....	77
6.1.3.1.2	Response.....	77
6.1.3.2	Example: <i>retrieving of all Presence Sources metadata using a filter (Informative)</i>	78
6.1.3.2.1	Request.....	78
6.1.3.2.2	Response.....	78
6.1.4	PUT.....	79
6.1.5	POST.....	79
6.1.5.1	Example 1: <i>creating Presence Source for user (Informative)</i>	79
6.1.5.1.1	Request.....	79
6.1.5.1.2	Response.....	79
6.1.5.2	Example 2: <i>creating Presence Source for user fails (Informative)</i>	80
6.1.5.2.1	Request.....	80
6.1.5.2.2	Response.....	81
6.1.6	DELETE.....	81
6.2	RESOURCE: INDIVIDUAL PRESENCE SOURCE.....	81
6.2.1	Request URL variables.....	81
6.2.2	Response Codes and Error Handling.....	82
6.2.3	GET.....	82
6.2.3.1	Example 1: <i>retrieving Presence Source (Informative)</i>	82
6.2.3.1.1	Request.....	82
6.2.3.1.2	Response.....	82
6.2.3.2	Example 2: <i>retrieving Presence Source which does not exist (Informative)</i>	83
6.2.3.2.1	Request.....	83
6.2.3.2.2	Response.....	83
6.2.4	PUT.....	83
6.2.4.1	Example: <i>updating Presence Source (Informative)</i>	83
6.2.4.1.1	Request.....	83
6.2.4.1.2	Response.....	84
6.2.5	POST.....	85
6.2.6	DELETE.....	85
6.2.6.1	Example: <i>removing Presence Source (Informative)</i>	85

- 6.2.6.1.1 Request..... 85
- 6.2.6.1.2 Response..... 85
- 6.3 RESOURCE: INDIVIDUAL PRESENCE SOURCE ATTRIBUTE.....85**
 - 6.3.1 Request URL variables 85
 - 6.3.1.1 *Light-weight relative resource paths*..... 86
 - 6.3.2 Response Codes and Error Handling 86
 - 6.3.3 GET..... 86
 - 6.3.3.1 *Example: retrieving individual presence attribute (Informative)*..... 87
 - 6.3.3.1.1 Request..... 87
 - 6.3.3.1.2 Response..... 87
 - 6.3.4 PUT..... 87
 - 6.3.4.1 *Example: updating individual presence attribute (Informative)* 87
 - 6.3.4.1.1 Request..... 87
 - 6.3.4.1.2 Response..... 87
 - 6.3.5 POST..... 88
 - 6.3.6 DELETE 88
 - 6.3.6.1 *Example: removing individual presence attribute (Informative)* 88
 - 6.3.6.1.1 Request..... 88
 - 6.3.6.1.2 Response..... 88
- 6.4 RESOURCE: PERSISTENT PRESENCE SOURCE88**
 - 6.4.1 Request URL variables 88
 - 6.4.2 Response Codes and Error Handling 88
 - 6.4.3 GET..... 89
 - 6.4.3.1 *Example: retrieving persistent presence information (Informative)* 89
 - 6.4.3.1.1 Request..... 89
 - 6.4.3.1.2 Response..... 89
 - 6.4.4 PUT..... 89
 - 6.4.4.1 *Example: updating persistent presence information (Informative)*..... 89
 - 6.4.4.1.1 Request..... 89
 - 6.4.4.1.2 Response..... 90
 - 6.4.4.1.3 Request..... 90
 - 6.4.4.1.4 Response..... 91
 - 6.4.5 POST..... 91
 - 6.4.6 DELETE 91
 - 6.4.6.1 *Example: removing persistent presence information (Informative)* 91
 - 6.4.6.1.1 Request..... 91
 - 6.4.6.1.2 Response..... 91
- 6.5 RESOURCE: INDIVIDUAL PERSISTENT PRESENCE SOURCE ATTRIBUTE91**
 - 6.5.1 Request URL variables 92
 - 6.5.1.1 *Light-weight relative resource paths*..... 92
 - 6.5.2 Response Codes and Error Handling 93
 - 6.5.3 GET..... 93
 - 6.5.3.1 *Example: retrieving individual persistent presence attribute (Informative)*..... 93
 - 6.5.3.1.1 Request..... 93
 - 6.5.3.1.2 Response..... 93
 - 6.5.4 PUT..... 93
 - 6.5.4.1 *Example: updating individual persistent presence attribute (Informative)*..... 93
 - 6.5.4.1.1 Request..... 93
 - 6.5.4.1.2 Response..... 94
 - 6.5.5 POST..... 94
 - 6.5.6 DELETE 94
 - 6.5.6.1 *Example: removing individual persistent presence attribute (Informative)* 94
 - 6.5.6.1.1 Request..... 94
 - 6.5.6.1.2 Response..... 94
- 6.6 RESOURCE: PRESENTITY CONTENT LIST.....94**
 - 6.6.1 Request URL variables 95
 - 6.6.2 Response Codes and Error Handling 95
 - 6.6.3 GET..... 95
 - 6.6.3.1 *Example: retrieving list of available contents (Informative)*..... 95
 - 6.6.3.1.1 Request..... 95
 - 6.6.3.1.2 Response..... 95

6.6.4	PUT.....	96
6.6.5	POST.....	96
6.6.6	DELETE	96
6.7	RESOURCE: INDIVIDUAL PRESENTITY CONTENT.....	96
6.7.1	Request URL variables	96
6.7.2	Response Codes and Error Handling	96
6.7.3	GET.....	96
6.7.3.1	<i>Example: retrieving individual content by Presentity (Informative).....</i>	97
6.7.3.1.1	Request.....	97
6.7.3.1.2	Response.....	97
6.7.4	PUT.....	97
6.7.4.1	<i>Example: uploading/updating individual content by Presentity (Informative).....</i>	97
6.7.4.1.1	Request.....	97
6.7.4.1.2	Response.....	97
6.7.5	POST.....	97
6.7.6	DELETE	97
6.7.6.1	<i>Example: removing individual content by Presentity (Informative).....</i>	98
6.7.6.1.1	Request.....	98
6.7.6.1.2	Response.....	98
6.8	RESOURCE: WATCHERS LIST.....	98
6.8.1	Request URL variables	98
6.8.2	Response Codes and Error Handling	98
6.8.3	GET.....	98
6.8.3.1	<i>Example: retrieving list of Watchers (Informative).....</i>	99
6.8.3.1.1	Request.....	99
6.8.3.1.2	Response.....	99
6.8.4	PUT.....	99
6.8.5	POST.....	99
6.8.6	DELETE	99
6.9	RESOURCE: INDIVIDUAL WATCHER.....	100
6.9.1	Request URL variables	100
6.9.2	Response Codes and Error Handling	100
6.9.3	GET.....	100
6.9.3.1	<i>Example: retrieving individual Watcher (Informative).....</i>	100
6.9.3.1.1	Request.....	100
6.9.3.1.2	Response.....	100
6.9.4	PUT.....	101
6.9.5	POST.....	101
6.9.6	DELETE	101
6.10	RESOURCE: AUTHORIZATION RULES	101
6.10.1	Request URL variables	101
6.10.2	Response Codes and Error Handling	101
6.10.3	GET.....	102
6.10.3.1	<i>Example: retrieving all authorization rules (Informative).....</i>	102
6.10.3.1.1	Request.....	102
6.10.3.1.2	Response.....	102
6.10.4	PUT.....	102
6.10.5	POST.....	102
6.10.5.1	<i>Example: creating an authorization rule (Informative).....</i>	102
6.10.5.1.1	Request.....	102
6.10.5.1.2	Response.....	103
6.10.5.2	<i>Example 2: creating an authorization rule, response with resourceReference (Informative).....</i>	103
6.10.5.2.1	Request.....	103
6.10.5.2.2	Response.....	103
6.10.6	DELETE	104
6.11	RESOURCE: INDIVIDUAL AUTHORIZATION RULE.....	104
6.11.1	Request URL variables	104
6.11.2	Response Codes and Error Handling	104
6.11.3	GET.....	104
6.11.3.1	<i>Example: retrieving an authorization rule (Informative)</i>	104

- 6.11.3.1.1 Request 104
- 6.11.3.1.2 Response..... 105
- 6.11.4 PUT 105
 - 6.11.4.1 Example: updating an authorization rule (Informative)..... 105
 - 6.11.4.1.1 Request 105
 - 6.11.4.1.2 Response..... 105
- 6.11.5 POST..... 106
- 6.11.6 DELETE 106
 - 6.11.6.1 Example: removing an authorization rule (Informative)..... 106
 - 6.11.6.1.1 Request 106
 - 6.11.6.1.2 Response..... 106
- 6.12 RESOURCE: INDIVIDUAL AUTHORIZATION RULE DATA 106**
 - 6.12.1 Request URL variables 106
 - 6.12.1.1 Light-weight relative resource paths 107
 - 6.12.2 Response Codes and Error Handling 107
 - 6.12.3 GET 107
 - 6.12.3.1 Example: retrieving individual authorization rule data (Informative) 107
 - 6.12.3.1.1 Request 107
 - 6.12.3.1.2 Response..... 107
 - 6.12.4 PUT 108
 - 6.12.4.1 Example: updating individual authorization rule data (Informative)..... 108
 - 6.12.4.1.1 Request 108
 - 6.12.4.1.2 Response..... 108
 - 6.12.5 POST..... 108
 - 6.12.6 DELETE 108
 - 6.12.6.1 Example: removing individual authorization rule data (Informative) 108
 - 6.12.6.1.1 Request 108
 - 6.12.6.1.2 Response..... 109
- 6.13 RESOURCE: PRESENCE INFORMATION BY WATCHER..... 109**
 - 6.13.1 Request URL variables 109
 - 6.13.2 Response Codes and Error Handling 109
 - 6.13.3 GET 109
 - 6.13.3.1 Example 1: retrieving all presence information for Presentity (Informative)..... 110
 - 6.13.3.1.1 Request 110
 - 6.13.3.1.2 Response..... 110
 - 6.13.3.2 Example 2: retrieving presence for Presentity by using filter (Informative)..... 111
 - 6.13.3.2.1 Request 111
 - 6.13.3.2.2 Response..... 111
 - 6.13.4 PUT 111
 - 6.13.5 POST..... 111
 - 6.13.6 DELETE 111
- 6.14 RESOURCE: INDIVIDUAL PRESENCE ATTRIBUTE BY WATCHER 112**
 - 6.14.1 Request URL variables 112
 - 6.14.1.1 Light-weight relative resource paths 112
 - 6.14.2 Response Codes and Error Handling 113
 - 6.14.3 GET 113
 - 6.14.3.1 Example: retrieving individual presence attribute for Presentity (Informative)..... 113
 - 6.14.3.1.1 Request 113
 - 6.14.3.1.2 Response..... 113
 - 6.14.4 PUT 114
 - 6.14.5 POST..... 114
 - 6.14.6 DELETE 114
- 6.15 RESOURCE: PRESENCE INFORMATION BY WATCHER FOR A PRESENCE LIST 114**
 - 6.15.1 Request URL variables 114
 - 6.15.2 Response Codes and Error Handling 114
 - 6.15.3 GET 114
 - 6.15.3.1 Example: retrieving presence information for all Presentities in a Presence List (Informative)..... 115
 - 6.15.3.1.1 Request 115
 - 6.15.3.1.2 Response..... 115
 - 6.15.4 PUT 116

6.15.5	POST.....	116
6.15.6	DELETE	116
6.16	RESOURCE: CONTENT BY WATCHER.....	116
6.16.1	Request URL variables	117
6.16.2	Response Codes and Error Handling	117
6.16.3	GET.....	117
6.16.3.1	<i>Example: retrieving content by Watcher (Informative)</i>	117
6.16.3.1.1	Request.....	117
6.16.3.1.2	Response.....	117
6.16.4	PUT.....	117
6.16.5	POST.....	117
6.16.6	DELETE	118
6.17	RESOURCE: ALL SUBSCRIPTIONS	118
6.17.1	Request URL variables	118
6.17.2	Response Codes and Error Handling	118
6.17.3	GET.....	118
6.17.3.1	<i>Example: retrieving all active subscriptions for user (Informative)</i>	118
6.17.3.1.1	Request.....	118
6.17.3.1.2	Response.....	118
6.17.4	PUT.....	120
6.17.5	POST.....	120
6.17.6	DELETE	120
6.18	RESOURCE: ALL WATCHERS SUBSCRIPTIONS	120
6.18.1	Request URL variables	120
6.18.2	Response Codes and Error Handling	120
6.18.3	GET.....	120
6.18.3.1	<i>Example: retrieving all Watchers subscriptions (Informative)</i>	121
6.18.3.1.1	Request.....	121
6.18.3.1.2	Response.....	121
6.18.4	PUT.....	121
6.18.5	POST.....	122
6.18.5.1	<i>Example: creating new Watchers subscription, using tel URI (Informative)</i>	122
6.18.5.1.1	Request.....	122
6.18.5.1.2	Response.....	122
6.18.5.2	<i>Example: creating new Watchers subscription, using ACR (Informative)</i>	123
6.18.5.2.1	Request.....	123
6.18.5.2.2	Response.....	123
6.18.6	DELETE	123
6.19	RESOURCE: INDIVIDUAL WATCHERS SUBSCRIPTION	124
6.19.1	Request URL variables	124
6.19.2	Response Codes and Error Handling	124
6.19.3	GET.....	124
6.19.3.1	<i>Example: retrieving individual Watchers subscription (Informative)</i>	124
6.19.3.1.1	Request.....	124
6.19.3.1.2	Response.....	124
6.19.4	PUT.....	125
6.19.4.1	<i>Example: updating individual Watchers subscription (Informative)</i>	125
6.19.4.1.1	Request.....	125
6.19.4.1.2	Response.....	125
6.19.5	POST.....	126
6.19.6	DELETE	126
6.19.6.1	<i>Example: terminating individual Watchers subscription (Informative)</i>	126
6.19.6.1.1	Request.....	126
6.19.6.1.2	Response.....	126
6.20	RESOURCE: WATCHERS NOTIFICATION.....	126
6.20.1	Request URL variables	127
6.20.2	Response Codes and Error Handling	127
6.20.3	GET.....	127
6.20.4	PUT.....	127
6.20.5	POST.....	127

6.20.5.1	<i>Example 1: notifying Presentity about change in Watchers status (Informative)</i>	127
6.20.5.1.1	Request.....	127
6.20.5.1.2	Response.....	128
6.20.5.2	<i>Example2: notifying Presentity about subscription time out (Informative)</i>	128
6.20.5.2.1	Request.....	128
6.20.5.2.2	Response.....	128
6.20.5.3	<i>Example3: notifying Presentity about termination of Watchers subscription (reason unknown) (Informative)</i>	128
6.20.5.3.1	Request.....	128
6.20.5.3.2	Response.....	129
6.20.6	DELETE	129
6.21	RESOURCE: ALL PRESENCE SUBSCRIPTIONS.....	129
6.21.1	Request URL variables	129
6.21.2	Response Codes and Error Handling	129
6.21.3	GET.....	130
6.21.3.1	<i>Example: retrieving all presence subscriptions for all Presentities (Informative)</i>	130
6.21.3.1.1	Request.....	130
6.21.3.1.2	Response.....	130
6.21.4	PUT.....	131
6.21.5	POST.....	131
6.21.6	DELETE	131
6.22	RESOURCE: PRESENCE SUBSCRIPTIONS FOR A SINGLE PRESENTITY	131
6.22.1	Request URL variables	131
6.22.2	Response Codes and Error Handling	131
6.22.3	GET.....	131
6.22.3.1	<i>Example: retrieving all presence subscriptions for Presentity (Informative)</i>	132
6.22.3.1.1	Request.....	132
6.22.3.1.2	Response.....	132
6.22.4	PUT.....	132
6.22.5	POST.....	133
6.22.5.1	<i>Example 1: creating new presence subscription for Presentity (Informative)</i>	133
6.22.5.1.1	Request.....	133
6.22.5.1.2	Response.....	133
6.22.5.2	<i>Example 2: creating new presence subscription for unknown Presentity (Informative)</i>	134
6.22.5.2.1	Request.....	134
6.22.5.2.2	Response.....	134
6.22.6	DELETE	134
6.23	RESOURCE: INDIVIDUAL PRESENCE SUBSCRIPTION.....	134
6.23.1	Request URL variables	135
6.23.2	Response Codes and Error Handling	135
6.23.3	GET.....	135
6.23.3.1	<i>Example: retrieving individual presence subscription (Informative)</i>	135
6.23.3.1.1	Request.....	135
6.23.3.1.2	Response.....	135
6.23.4	PUT.....	136
6.23.4.1	<i>Example: updating individual presence subscription (Informative)</i>	136
6.23.4.1.1	Request.....	136
6.23.4.1.2	Response.....	136
6.23.5	POST.....	137
6.23.6	DELETE	137
6.23.6.1	<i>Example: terminating individual presence subscription (Informative)</i>	137
6.23.6.1.1	Request.....	137
6.23.6.1.2	Response.....	137
6.24	RESOURCE: PRESENCE NOTIFICATION.....	137
6.24.1	Request URL variables	138
6.24.2	Response Codes and Error Handling	138
6.24.3	GET.....	138
6.24.4	PUT.....	138
6.24.5	POST.....	138
6.24.5.1	<i>Example 1: notifying Watcher about presence information updates from an active subscription (Informative)</i>	139
6.24.5.1.1	Request.....	139

6.24.5.1.2	Response.....	139
6.24.5.2	<i>Example 2: notifying Watcher about presence information updates from pending subscription (Informative).....</i>	<i>139</i>
6.24.5.2.1	Request.....	139
6.24.5.2.2	Response.....	140
6.24.5.3	<i>Example 3: notifying Watcher about termination of presence subscription (reason unknown) (Informative).....</i>	<i>140</i>
6.24.5.3.1	Request.....	140
6.24.5.3.2	Response.....	140
6.24.5.4	<i>Example 4: notifying Watcher about termination of presence subscription (Watcher blocked) (Informative).....</i>	<i>140</i>
6.24.5.4.1	Request.....	140
6.24.5.4.2	Response.....	141
6.24.6	DELETE.....	141
6.25	RESOURCE: ALL PRESENCE LIST SUBSCRIPTIONS.....	141
6.25.1	Request URL variables.....	141
6.25.2	Response Codes and Error Handling.....	141
6.25.3	GET.....	141
6.25.3.1	<i>Example: retrieving all Presence List subscriptions towards all Presence Lists (Informative).....</i>	<i>142</i>
6.25.3.1.1	Request.....	142
6.25.3.1.2	Response.....	142
6.25.4	PUT.....	142
6.25.5	POST.....	143
6.25.6	DELETE.....	143
6.26	RESOURCE: PRESENCE LIST SUBSCRIPTIONS FOR A SINGLE PRESENCE LIST.....	143
6.26.1	Request URL variables.....	143
6.26.2	Response Codes and Error Handling.....	143
6.26.3	GET.....	143
6.26.3.1	<i>Example: retrieving all Presence List subscriptions towards a single Presence List (Informative).....</i>	<i>144</i>
6.26.3.1.1	Request.....	144
6.26.3.1.2	Response.....	144
6.26.4	PUT.....	144
6.26.5	POST.....	144
6.26.5.1	<i>Example: creating new Presence List subscription towards a single Presence List (Informative).....</i>	<i>144</i>
6.26.5.1.1	Request.....	144
6.26.5.1.2	Response.....	145
6.26.6	DELETE.....	145
6.27	RESOURCE: INDIVIDUAL PRESENCE LIST SUBSCRIPTION.....	146
6.27.1	Request URL variables.....	146
6.27.2	Response Codes and Error Handling.....	146
6.27.3	GET.....	146
6.27.3.1	<i>Example: retrieving individual Presence List subscription (Informative).....</i>	<i>146</i>
6.27.3.1.1	Request.....	146
6.27.3.1.2	Response.....	146
6.27.4	PUT.....	147
6.27.4.1	<i>Example: updating individual Presence List subscription (Informative).....</i>	<i>147</i>
6.27.4.1.1	Request.....	147
6.27.4.1.2	Response.....	148
6.27.5	POST.....	148
6.27.6	DELETE.....	148
6.27.6.1	<i>Example: terminating individual Presence List subscription (Informative).....</i>	<i>148</i>
6.27.6.1.1	Request.....	148
6.27.6.1.2	Response.....	148
6.28	RESOURCE: PRESENCE LIST NOTIFICATION.....	148
6.28.1	Request URL variables.....	149
6.28.2	Response Codes and Error Handling.....	149
6.28.3	GET.....	149
6.28.4	PUT.....	149
6.28.5	POST.....	149
6.28.5.1	<i>Example 1: notifying Watcher about presence information updates relating to Presence List (Informative).....</i>	<i>149</i>
6.28.5.1.1	Request.....	149
6.28.5.1.2	Response.....	150
6.28.5.2	<i>Example 2: notifying Watcher about termination of Presence List subscription (No resource) (Informative).....</i>	<i>150</i>

- 6.28.5.2.1 Request 150
- 6.28.5.2.2 Response..... 151
- 6.28.5.3 *Example 3: notifying Watcher about termination of presence subscription (reason unknown) (Informative)*..... 151
 - 6.28.5.3.1 Request 151
 - 6.28.5.3.2 Response..... 151
- 6.28.5.4 *Example 4: notifying Watcher about subscription time out (Informative)*..... 151
 - 6.28.5.4.1 Request 151
 - 6.28.5.4.2 Response..... 152
- 6.28.6 DELETE 152
- 6.29 RESOURCE: PRESENTITY PORTRAIT ICON152**
 - 6.29.1 Request URL variables152
 - 6.29.2 Response Codes and Error Handling152
 - 6.29.3 GET.....153
 - 6.29.3.1 *Example: retrieving portrait icon by Presentity (Informative)* 153
 - 6.29.3.1.1 Request 153
 - 6.29.3.1.2 Response..... 153
 - 6.29.4 PUT153
 - 6.29.4.1 *Example: uploading/updating of portrait icon and setting the link to the icon as presence information (Informative)*..... 153
 - 6.29.4.1.1 Request 153
 - 6.29.4.1.2 Response..... 153
 - 6.29.5 POST.....153
 - 6.29.6 DELETE153
 - 6.29.6.1 *Example: removing portrait icon by Presentity (Informative)*..... 154
 - 6.29.6.1.1 Request 154
 - 6.29.6.1.2 Response..... 154
- 7. FAULT DEFINITIONS155**
 - 7.1 SERVICE EXCEPTIONS.....155**
 - 7.1.1 SVC0220: No subscription request155
 - 7.1.2 SVC0221: Not a Watcher155
 - 7.1.3 SVC0222: Key property changes not allowed155
 - 7.1.4 SVC1001: Presence source does not exist155
 - 7.2 POLICY EXCEPTIONS156**
 - 7.2.1 POL0260: Maximum number of presence sources exceeded156
- APPENDIX A. CHANGE HISTORY (INFORMATIVE).....157**
 - A.1 APPROVED VERSION HISTORY157**
 - A.2 DRAFT/CANDIDATE VERSION 1.0 HISTORY157**
- APPENDIX B. STATIC CONFORMANCE REQUIREMENTS (NORMATIVE).....161**
 - B.1 SCR FOR REST.PRESENCE SERVER.....161**
 - B.1.1 SCR for REST.Presence.Presentity.PresenceSource Server 161
 - B.1.2 SCR for REST.Presence.Presentity.Individual.PresenceSource Server 161
 - B.1.3 SCR for REST.Presence.Presentity.Individual.PresenceSource.Attribute Server 162
 - B.1.4 SCR for REST.Presence.Presentity.PresenceSource.Persistent Server 162
 - B.1.5 SCR for REST.Presence.Presentity.PresenceSource.Persistent.Attribute Server 163
 - B.1.6 SCR for REST.Presence.Presentity.ContentList Server 163
 - B.1.7 SCR for REST.Presence.Presentity.Individual.Content Server 163
 - B.1.8 SCR for REST.Presence.Presentity.WatcherList Server..... 163
 - B.1.9 SCR for REST.Presence.Presentity.Individual.Watcher Server 164
 - B.1.10 SCR for REST.Presence.Presentity.Authorization.Rules Server 164
 - B.1.11 SCR for REST.Presence.Presentity.Individual.Authorization.Rule Server 164
 - B.1.12 SCR for REST.Presence.Presentity.Individual.Authorization.Rule.Data Server 165
 - B.1.13 SCR for REST.Presence.Watcher.PresenceContact Server 165
 - B.1.14 SCR for REST.Presence.Watcher.Individual.PresenceContact.Attribute Server..... 165
 - B.1.15 SCR for REST.Presence.Watcher.PresenceList Server 166
 - B.1.16 SCR for REST.Presence.Watcher.PresenceContactContent Server..... 166
 - B.1.17 SCR for REST.Presence.Subscriptions Server 166
 - B.1.18 SCR for REST.Presence.Presentity.Subscriptions.WatchersSubscriptions Server 166

B.1.19	SCR for REST.Presence.Presentity.Individual.Subscriptions.WatchersSubscriptions Server.....	167
B.1.20	SCR for REST.Presence.WatchersSubscriptions.Notifications Server.....	167
B.1.21	SCR for REST.Presence.Watcher.Subscriptions.PresenceSubscriptions Server	167
B.1.22	SCR for REST.Presence.Watcher.Subscriptions.PresenceSubscriptions.SinglePresentity Server	167
B.1.23	SCR for REST.Presence.Watcher.Individual.Subscriptions.PresenceSubscriptions.SinglePresentity Server	168
B.1.24	SCR for REST.Presence.PresenceSubscriptions.Notifications Server.....	168
B.1.25	SCR for REST.Presence.Watcher.Subscriptions.PresenceListSubscriptions Server	169
B.1.26	SCR for REST.Presence.Watcher.Subscriptions.PresenceListSubscriptions.SinglePresenceList Server	169
B.1.27	SCR for REST.Presence.Watcher.Individual.Subscriptions.PresenceListSubscriptions.SinglePresenceList Server	170
B.1.28	SCR for REST.Presence.PresenceListSubscriptions.Notifications Server	170
B.1.29	SCR for REST.Presence.Presentity.Portrait.Icon Server	170
APPENDIX C. APPLICATION/X-WWW-FORM-URLENCODED REQUEST FORMAT FOR POST OPERATIONS (NORMATIVE)172		
C.1	CREATE PRESENCE SOURCE.....	172
C.1.1	Example: creating Presence Source for user (Informative).....	175
C.1.1.1	Request.....	175
C.1.1.2	Response.....	176
C.2	CREATE AUTHORIZATION RULE	176
C.2.1	Example: creating an authorization rule (Informative)	177
C.2.1.1	Request.....	177
C.2.1.2	Response.....	177
C.3	CREATE WATCHERS SUBSCRIPTION.....	178
C.3.1	Example: creating new Watchers subscription, using tel URI (Informative).....	179
C.3.1.1	Request.....	179
C.3.1.2	Response	179
C.3.2	Example: creating new Watchers subscription, using ACR (Informative)	180
C.3.2.1	Request.....	180
C.3.2.2	Response.....	180
C.4	CREATE PRESENCE SUBSCRIPTION	181
C.4.1	Example: creating new presence subscription for Presentity (Informative).....	182
C.4.1.1	Request.....	182
C.4.1.2	Response.....	183
C.5	CREATE PRESENCE LIST SUBSCRIPTION	183
C.5.1	Example: creating new Presence List subscription towards a single Presence List (Informative).....	185
C.5.1.1	Request.....	185
C.5.1.2	Response	185
APPENDIX D. JSON EXAMPLES (INFORMATIVE)186		
D.1	RETRIEVING ALL PRESENCE SOURCES FOR USER (SECTION 6.1.3.1)	186
D.2	RETRIEVING OF ALL PRESENCE SOURCES METADATA USING A FILTER (SECTION 6.1.3.2).....	187
D.3	CREATING PRESENCE SOURCE FOR USER (SECTION 6.1.5.1).....	187
D.4	CREATING PRESENCE SOURCE FOR USER FAILS (SECTION 6.1.5.2).....	188
D.5	RETRIEVING PRESENCE SOURCE (SECTION 6.2.3.1)	189
D.6	RETRIEVING PRESENCE SOURCE WHICH DOES NOT EXIST (SECTION 6.2.3.2)	190
D.7	UPDATING PRESENCE SOURCE (SECTION 6.2.4.1).....	190
D.8	REMOVING PRESENCE SOURCE (SECTION 6.2.6.1)	192
D.9	RETRIEVING INDIVIDUAL PRESENCE ATTRIBUTE (SECTION 6.3.3.1)	192
D.10	UPDATING INDIVIDUAL PRESENCE ATTRIBUTE (SECTION 6.3.4.1).....	192
D.11	REMOVING INDIVIDUAL PRESENCE ATTRIBUTE (SECTION 6.3.6.1)	193
D.12	RETRIEVING PERSISTENT PRESENCE INFORMATION (SECTION 6.4.3.1)	193
D.13	UPDATING PERSISTENT PRESENCE INFORMATION (SECTION 6.4.4.1).....	193
D.14	REMOVING PERSISTENT PRESENCE INFORMATION (SECTION 6.4.6.1)	195
D.15	RETRIEVING INDIVIDUAL PERSISTENT PRESENCE ATTRIBUTE (SECTION 6.5.3.1).....	195
D.16	UPDATING INDIVIDUAL PERSISTENT PRESENCE ATTRIBUTE (SECTION 6.5.4.1)	196
D.17	REMOVING INDIVIDUAL PERSISTENT PRESENCE ATTRIBUTE (SECTION 6.5.6.1).....	196
D.18	RETRIEVING LIST OF AVAILABLE CONTENTS (SECTION 6.6.3.1).....	196
D.19	RETRIEVING INDIVIDUAL CONTENT BY PRESENTITY (SECTION 6.7.3.1)	197

D.20	UPLOADING/UPDATING INDIVIDUAL CONTENT BY PRESENTITY (SECTION 6.7.4.1)	197
D.21	REMOVING INDIVIDUAL CONTENT BY PRESENTITY (SECTION 6.7.6.1)	198
D.22	RETRIEVING LIST OF WATCHERS (SECTION 6.8.3.1)	198
D.23	RETRIEVING INDIVIDUAL WATCHER (SECTION 6.9.3.1)	199
D.24	RETRIEVING ALL AUTHORIZATION RULES (SECTION 6.10.3.1)	199
D.25	CREATING AN AUTHORIZATION RULE (SECTION 6.10.5.1)	200
D.26	CREATING AN AUTHORIZATION RULE, RESPONSE WITH RESOURCE REFERENCE (SECTION 6.10.5.2)	200
D.27	RETRIEVING AN AUTHORIZATION RULE (SECTION 6.11.3.1)	201
D.28	UPDATING AN AUTHORIZATION RULE (SECTION 6.11.4.1)	201
D.29	REMOVING AN AUTHORIZATION RULE (SECTION 6.11.6.1)	202
D.30	RETRIEVING INDIVIDUAL AUTHORIZATION RULE DATA (SECTION 6.12.3.1)	202
D.31	UPDATING INDIVIDUAL AUTHORIZATION RULE DATA (SECTION 6.12.4.1)	203
D.32	REMOVING INDIVIDUAL AUTHORIZATION RULE DATA (SECTION 6.12.6.1)	203
D.33	RETRIEVING ALL PRESENCE INFORMATION FOR PRESENTITY (SECTION 6.13.3.1)	203
D.34	RETRIEVING PRESENCE INFORMATION FOR PRESENTITY BY USING FILTER (SECTION 6.13.3.2)	204
D.35	RETRIEVING INDIVIDUAL PRESENCE ATTRIBUTE FOR PRESENTITY (SECTION 6.14.3.1)	205
D.36	RETRIEVING PRESENCE INFORMATION FOR ALL PRESENTITIES IN A PRESENCE LIST (SECTION 6.15.3.1)	205
D.37	RETRIEVING CONTENT BY WATCHER (SECTION 6.16.3.1)	206
D.38	RETRIEVING ALL ACTIVE SUBSCRIPTIONS FOR USER (SECTION 6.17.3.1)	207
D.39	RETRIEVING ALL WATCHERS SUBSCRIPTIONS (SECTION 6.18.3.1)	208
D.40	CREATING NEW WATCHERS SUBSCRIPTION, USING TEL URI (SECTION 6.18.5.1)	209
D.41	CREATING NEW WATCHERS SUBSCRIPTION, USING ACR (SECTION 6.18.5.1)	210
D.42	RETRIEVING INDIVIDUAL WATCHERS SUBSCRIPTION (SECTION 6.19.3.1)	210
D.43	UPDATING INDIVIDUAL WATCHERS SUBSCRIPTION (SECTION 6.19.4.1)	211
D.44	TERMINATING INDIVIDUAL WATCHERS SUBSCRIPTION (SECTION 6.19.6.1)	212
D.45	NOTIFYING PRESENTITY ABOUT CHANGE IN WATCHERS STATUS (SECTION 6.20.5.1)	212
D.46	NOTIFYING PRESENTITY ABOUT SUBSCRIPTION TIME OUT (SECTION 6.20.5.2)	213
D.47	NOTIFYING PRESENTITY ABOUT TERMINATION OF WATCHERS SUBSCRIPTION (REASON UNKNOWN) (SECTION 6.20.5.3)	213
D.48	RETRIEVING ALL PRESENCE SUBSCRIPTIONS FOR ALL PRESENTITIES (SECTION 6.21.3.1)	214
D.49	RETRIEVING ALL PRESENCE SUBSCRIPTIONS FOR PRESENTITY (SECTION 6.22.3.1)	215
D.50	CREATING NEW PRESENCE SUBSCRIPTION FOR PRESENTITY (SECTION 6.22.5.1)	216
D.51	CREATING NEW PRESENCE SUBSCRIPTION FOR UNKNOWN PRESENTITY (SECTION 6.22.5.2)	217
D.52	RETRIEVING INDIVIDUAL PRESENCE SUBSCRIPTION (SECTION 6.23.3.1)	217
D.53	UPDATING INDIVIDUAL PRESENCE SUBSCRIPTION (SECTION 6.23.4.1)	218
D.54	TERMINATING INDIVIDUAL PRESENCE SUBSCRIPTION (SECTION 6.23.6.1)	219
D.55	NOTIFYING WATCHER ABOUT PRESENCE INFORMATION UPDATES FROM AN ACTIVE SUBSCRIPTION (SECTION 6.24.5.1)	219
D.56	NOTIFYING WATCHER ABOUT PRESENCE INFORMATION UPDATES FROM PENDING SUBSCRIPTION (SECTION 6.24.5.2)	220
D.57	NOTIFYING WATCHER ABOUT TERMINATION OF PRESENCE SUBSCRIPTION (REASON UNKNOWN) (SECTION 6.24.5.3)	220
D.58	NOTIFYING WATCHER ABOUT TERMINATION OF PRESENCE SUBSCRIPTION (WATCHER BLOCKED) (SECTION 6.24.5.4)	221
D.59	RETRIEVING ALL PRESENCE LIST SUBSCRIPTIONS TOWARDS ALL PRESENCE LISTS (SECTION 6.25.3.1)	221
D.60	RETRIEVING ALL PRESENCE LIST SUBSCRIPTIONS TOWARDS A SINGLE PRESENCE LIST (SECTION 6.26.3.1)	222
D.61	CREATING NEW PRESENCE LIST SUBSCRIPTION TOWARDS A SINGLE PRESENCE LIST (SECTION 6.26.5.1)	223
D.62	RETRIEVING INDIVIDUAL PRESENCE LIST SUBSCRIPTION (SECTION 6.27.3.1)	224
D.63	UPDATING INDIVIDUAL PRESENCE LIST SUBSCRIPTION (SECTION 6.27.4.1)	225
D.64	TERMINATING INDIVIDUAL PRESENCE LIST SUBSCRIPTION (SECTION 6.27.6.1)	226
D.65	NOTIFYING WATCHER ABOUT PRESENCE INFORMATION UPDATES RELATING TO PRESENCE LIST (SECTION 6.28.5.1)	226
D.66	NOTIFYING WATCHER ABOUT TERMINATION OF PRESENCE LIST SUBSCRIPTION (NO RESOURCE) (SECTION 6.28.5.2)	227
D.67	NOTIFYING WATCHER ABOUT TERMINATION OF PRESENCE SUBSCRIPTION (REASON UNKNOWN) (SECTION 6.28.5.3)	227
D.68	NOTIFYING WATCHER ABOUT SUBSCRIPTION TIME OUT (SECTION 6.28.5.4)	228

D.69	RETRIEVING PORTRAIT ICON BY PRESENTITY (SECTION 6.29.3.1)	228
D.70	UPLOADING/UPDATING OF PORTRAIT ICON AND SETTING THE LINK TO THE ICON AS PRESENCE INFORMATION (SECTION 6.29.4.1)	229
D.71	REMOVING PORTRAIT ICON BY PRESENTITY (SECTION 6.29.6.1)	229
APPENDIX E.	PARLAY X OPERATIONS MAPPING (INFORMATIVE)	230
APPENDIX F.	LIGHT-WEIGHT RESOURCES (INFORMATIVE)	231
APPENDIX G.	AUTHORIZATION ASPECTS (NORMATIVE)	234
G.1	USE WITH OMA AUTHORIZATION FRAMEWORK FOR NETWORK API	234
G.1.1	Scope values	234
G.1.1.1	Definitions	234
G.1.1.2	Downscoping	234
G.1.1.3	Mapping with resources and methods	235
G.1.2	Use of 'acr:auth'	240

Figures

Figure 1	Resource structure defined by this specification	22
Figure 2	Creation of Presence Source, and subscription to Watchers information	68
Figure 3	Subscription for presence information, and Watcher authorization	71
Figure 4	Update of presence information	73
Figure 5	Termination of subscriptions for Watchers, and presence information	74

Tables

Table 1:	Parlay X operations mapping	230
Table 2:	Light-weight Resources for Presence	233
Table 3:	Scope values for RESTful Presence API	234
Table 4:	Required scope values for: Management of presence information on behalf of Presentity	236
Table 5:	Required scope values for: Retrieval of Watchers information by Presentity	237
Table 6:	Required scope values for: Management of subscriptions to notifications for Watchers information	237
Table 7:	Required scope values for: Management of authorization rules for accessing presence information	238
Table 8:	Required scope values for: Retrieval of presence information by Watcher	238
Table 9:	Required scope values for: Management of subscriptions to notifications for presence information	239

1. Scope

This specification defines a RESTful Presence API using an HTTP protocol binding, based on similar API defined in [3GPP 29.199-14].

2. References

2.1 Normative References

- [3GPP 29.199-14] 3GPP Technical Specification, “Open Service Access (OSA); Parlay X Web Services; Part X: Presence (Release 8)”,
URL:<http://www.3gpp.org/>
- [Autho4API_10] “Authorization Framework for Network APIs”, Open Mobile Alliance™, OMA-ER-Autho4API-V1_0, URL: <http://www.openmobilealliance.org/>
- [IETF_ACR_draft] “The acr URI for anonymous users”, S.Jakobsson, K.Smith, July 2011, URL:
<http://tools.ietf.org/html/draft-uri-acr-extension-03>
- [ISO.3166-2] “Codes for the representation of names of countries and their subdivisions – Part 2: Country subdivision code”, International Organisation for Standardization, ISO Standard 3166-2:2007
- [OMA_DDS] “Presence SIMPLE Data Specification”, Open Mobile Alliance™, OMA-DDS-Presence_Data_Ext_V2_1,
URL:<http://www.openmobilealliance.org/>
- [OMNA] Open Mobile Naming Authority, URL: <http://www.openmobilealliance.org/Tech/OMNA.aspx>
- [REST_NetAPI_Common] “Common definitions for RESTful Network APIs”, Open Mobile Alliance™, OMA-TS-REST_NetAPI_Common-V1_0,
URL:<http://www.openmobilealliance.org/>
- [REST_NetAPI_Notification Channel] “RESTful Network API for Notification Channel”, Open Mobile Alliance™, OMA-TS-REST_NetAPI_NotificationChannel-V1_0, URL: <http://www.openmobilealliance.org/>
- [REST_SUP_Presence] “XML schema for the RESTful Network API for Presence”, Open Mobile Alliance™, OMA-SUP-XSD_rest_netapi_presence-V1_0, URL:<http://www.openmobilealliance.org/>
- [RFC2119] “Key words for use in RFCs to Indicate Requirement Levels”, S. Bradner, March 1997,
URL:<http://www.ietf.org/rfc/rfc2119.txt>
- [RFC2616] “Hypertext Transfer Protocol -- HTTP/1.1”, R. Fielding et. al, January 1999,
URL:<http://www.ietf.org/rfc/rfc2616.txt>
- [RFC3261] “SIP: Session Initiation Protocol”, J. Rosenberg et al., June 2002,
URL: <http://www.rfc-editor.org/rfc/rfc3261.txt>
- [RFC3265] “Session Initiation Protocol (SIP)-Specific Event Notification”, A.B. Roach, Jun 2002,
URL: <http://www.ietf.org/rfc/rfc3265.txt>
- [RFC3857] “A Watcher Information Event Template-Package for the Session Initiation Protocol (SIP)”, J. Rosenberg, Aug 2004,
URL: <http://www.ietf.org/rfc/rfc3857.txt>
- [RFC3863] “Presence Information Data Format (PIDF)”, H.Sugano et al., Aug 2004,
URL: <http://www.ietf.org/rfc/rfc3863.txt>
- [RFC3966] “The tel URI for Telephone Numbers”, H.Schulzrinne, December 2004, URL:
<http://www.ietf.org/rfc/rfc3966.txt>
- [RFC3986] “Uniform Resource Identifier (URI): Generic Syntax”, R. Fielding et. al, January 2005,
URL:<http://www.ietf.org/rfc/rfc3986.txt>
- [RFC4119] “A Presence-based GEOPRIV Location Object Format”, J.Peterson, December 2005,
URL: <http://www.ietf.org/rfc/rfc4119.txt>
- [RFC4234] “Augmented BNF for Syntax Specifications: ABNF”. D. Crocker, Ed., P. Overell. October 2005,
URL:<http://www.ietf.org/rfc/rfc4234.txt>
- [RFC4627] “The application/json Media Type for JavaScript Object Notation (JSON)”, D. Crockford, July 2006,
URL: <http://www.ietf.org/rfc/rfc4627.txt>
- [RFC4479] “A data model for Presence”, J.Rosenberg, July 2006,
URL: <http://www.ietf.org/rfc/rfc4479.txt>

- [RFC4480] “RPID: Rich Presence Extensions to the Presence Information Data Format (PIDF)”, H.Schulzrinne, V.Gurbani, P.Kyzivat, J.rosenberg, July 2006, URL: <http://www.ietf.org/rfc/rfc4480.txt>
- [RFC4482] “CIPID: Contact Information for the Presence Information Data Format”, H. Schulzrinne. July 2006, URL: <http://www.ietf.org/rfc/rfc4482.txt>
- [RFC5139] “Revised Civic Location Format for Presence Information Data Format Location Object (PIDF-LO)”. M. Thomson, J. Winterbottom, February 2008, URL: <http://www.ietf.org/rfc/rfc5139.txt>
- [SCRRULES] “SCR Rules and Procedures”, Open Mobile Alliance™, OMA-ORG-SCR_Rules_and_Procedures, URL:<http://www.openmobilealliance.org/>
- [W3C_URLENC] HTML 4.01 Specification, Section 17.13.4 Form content types, The World Wide Web Consortium, URL: <http://www.w3.org/TR/html401/interact/forms.html#h-17.13.4.1>
- [W3C_XML11] W3C XML 1.1 Specification, URL: <http://www.w3.org/TR/xml11/>
- [XMLSchema1] W3C Recommendation, XML Schema Part 1: Structures Second Edition, URL: <http://www.w3.org/TR/xmlschema-1/>
- [XMLSchema2] W3C Recommendation, XML Schema Part 2: Datatypes Second Edition, URL: <http://www.w3.org/TR/xmlschema-2/>

2.2 Informative References

- [OMADICT] “Dictionary for OMA Specifications”, Version 2.9, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_9, URL:<http://www.openmobilealliance.org/>
- [OMA_PRS_RD] “Presence SIMPLE Requirements“, Version 2.0, Open Mobile Alliance™, OMA-RD-Presence_SIMPLE-V2_0, URL: <http://www.openmobilealliance.org/>
- [OMA_PRS_AD] “Presence SIMPLE Architecture” Version 2.0, Open Mobile Alliance™, OMA-AD-Presence-V2_0, URL: <http://www.openmobilealliance.org/>
- [ParlayREST_Presence] “RESTful bindings for Parlay X Web Services – Presence”, Version 1.0, Open Mobile Alliance™, OMA-TS-ParlayREST_Presence-V1_0, URL:<http://www.openmobilealliance.org/>
- [REST_WP] “Guidelines for RESTful Network APIs”, Open Mobile Alliance™, OMA-WP-Guidelines_for_RESTful_Network_APIs, 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

Client-side Notification URL	An HTTP URL exposed by a client, on which it is capable of receiving notifications and that can be used by the client when subscribing to notifications.
Heavy-weight Resource	A resource which is identified by a resource URL which is then used by HTTP methods to operate on the entire data structure representing the resource
Light-weight Resource	A subordinate resource of a Heavy-weight Resource which is identified by its own resource URL which is then used by HTTP methods to operate on a part of the data structure representing the Heavy-weight Resource. The Light-weight Resource URL can be seen as an extension of the Heavy-weight Resource URL. There could be several levels of Light-weight Resources below the ancestor Heavy-weight Resource, depending on the data structure
Light-weight resource key property	A child element of an element that can be accessed as a Light-weight Resource which uniquely identifies its parent element among its siblings in the XML element tree.
Notification Channel	A channel created on the request of the client and used to deliver notifications from a server to a client. The channel is represented as a resource and provides means for the server to post notifications and for the client to receive them via specified delivery mechanisms.
Notification Server	A server that is capable of creating and maintaining Notification Channels.
Presence List	A pre-defined list of Presentities, e.g. stored on a server via [REST_NetAPI_AddressBook], which enables a Watcher to subscribe for or retrieve presence information of multiple Presentities using a single request.
Presence Source	An entity that on behalf of a Presentity is publishing presence information that is valid only a certain time unless it is not refreshed by the Presence Source. In the context of this specification a Presence Source refers to an instance of a Presentity's presence information on the server. There may be zero or more Presence Sources related to a given Presentity at a given time.
Presentity	A logical entity that has presence information associated with it. A Presentity is most commonly a reference for a person, although it may represent a role such as “help desk” or a resource such as “conference room #27”.
Server-side Notification URL	An HTTP URL exposed by a Notification Server, that identifies a Notification Channel and that can be used by a client when subscribing to notifications.
Watcher	Any uniquely identifiable entity that requests presence information about a Presentity.

Additionally, all definitions from the OMA Dictionary [OMADICT] and Presence specific definitions from [OMA_PRS_RD] and [OMA_PRS_AD] apply.

3.3 Abbreviations

ACR	Anonymous Customer Reference
API	Application Programming Interface
GPRS	General Packet Radio Service
GSM	Global System for Mobile communication

HTTP	HyperText Transfer Protocol
IMS	IP Multimedia Subsystem
IP	Internet Protocol
ISDN	Integrated Services Digital Network
JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation
MIME	Multipurpose Internet Mail Extensions
MSISDN	Mobile Subscriber ISDN Number
OMA	Open Mobile Alliance
OMNA	Open Mobile Naming Authority
REST	REpresentational State Transfer
SCR	Static Conformance Requirements
SIP	Session Initiation Protocol
TS	Technical Specification
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTC	Universal Time Coordinated
WP	White Paper
XML	eXtensible Markup Language
XSD	XML Schema Definition

4. Introduction

The Technical Specification for the RESTful Network API for Presence contains HTTP protocol bindings based on the Parlay X Presence Web Services [3GPP 29.199-14] specification, using the REST architectural style. The specification provides resource definitions, the HTTP verbs applicable for each of these resources, and the element data structures, as well as support material including flow diagrams and examples using the various supported message body formats (i.e. XML, JSON, and application/x-www-form-urlencoded).

4.1 Version 1.0

The RESTful Network API for Presence V1.0 is a republication of the ParlayREST Presence API V1.0 [ParlayREST_Presence] as part of the suite of OMA RESTful Network APIs.

Bug fixes and structural changes to fit that suite, but also functional changes have been applied.

Version 1.0 of the RESTful Network API for Presence keeps supporting the following operations:

- Presentity manages presence information with a certain time-to-live
- Presentity manages persistent presence information
- Presentity manages own content
- Presentity retrieves Watchers to its presence information
- Presentity manages authorization rules
- Watcher retrieves presence information for a single Presentity
- Watcher retrieves presence information for Presentities in a Presence List
- Watcher retrieves content from a Presentity
- Presentity retrieves all active subscriptions
- Watcher retrieves all active subscriptions
- Presentity manages subscriptions for Watchers
- Watcher manages presence subscriptions for single Presentities
- Watcher manages presence subscriptions for Presence Lists

The following new functionality has been introduced:

- Presentity manages own portrait icon using a dedicated resource
- Support for scope values used with authorization framework defined in [Autho4API_10]
- Support for Anonymous Customer Reference (ACR) as an end user identifier
- Support for “acr:auth” as a reserved keyword in a resource URL variable that identifies an end user

5. Presence API definition

This section is organized to support a comprehensive understanding of the Presence API design. It specifies the definition of all resources, definition of all data structures, and definitions of all operations permitted on the specified resources.

Common data types, naming conventions, fault definitions and namespaces are defined in [REST_NetAPI_Common], Presence specific terminology and conventions are defined in [OMA_PRS_RD] and [OMA_PRS_AD].

The remainder of this document is structured as follows:

Section 5 starts with a diagram representing the resources hierarchy, followed by a table listing all the resources (and their URL) used by this API, along with the data structure and the supported HTTP verbs (section 5.1). What follows are the data structures (section 5.2). A sample of typical use cases is included in section 5.3, described as high level flow diagrams.

Section 6 contains the detailed specification for each of the resources. Each such subsection defines the resource, the request URL variables that are common for all HTTP commands, the possible HTTP response codes, and the supported HTTP verbs. For each supported HTTP verb, a description of the functionality is provided, along with an example of a request and an example of a response. For each unsupported HTTP verb, the returned HTTP error status is specified, as well as what should be returned in the Allow header.

All examples in section 6 use XML as the format for the message body. Application/x-www-form-urlencoded examples are provided in Appendix C, while JSON examples are provided in Appendix D.

Section 7 contains fault definition details such as Service Exceptions and Policy Exceptions. Appendix B provides the Static Conformance Requirements (SCR).

Appendix E lists the Parlay X equivalent method for each supported REST resource and method combination, where applicable.

Appendix F provides a list of all Light-weight Resources, where applicable.

Appendix G defines authorization aspects to control access to the resources defined in this specification.

Note: Throughout this document client and application can be used interchangeably.

5.1 Resources Summary

This section summarizes all the resources used by the RESTful Network API for Presence.

The "apiVersion" URL variable SHALL have the value "v1" to indicate that the API corresponds to this version of the specification. See [REST_NetAPI_Common] which specifies the semantics of this variable.

The figure below visualizes the resource structure defined by this specification. Note that those nodes in the resource tree which have associated HTTP methods defined in this specification are depicted by solid boxes.

The following tables give a detailed overview of the resources defined in this specification, the data type of their representation and the allowed HTTP methods.

Purpose: To allow Presentity to manage own presence information

Resource	URL Base URL: http://{serverRoot}/presence/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Presence Sources	/{userId}/presenceSources	PresenceSourceList (Used for GET) PresenceSource (Used for POST) common:ResourceReference (optional alternative for POST response)	Retrieves all Presence Sources related to a Presentity	no	Creates a new Presence Source for a Presentity	no
Individual Presence Source	/{userId}/presenceSources/{presenceSourceId}	PresenceSource (Used for PUT/GET)	Retrieves presence information for a Presentity for a specified Presence Source NOTE: Watcher SHALL NOT use this operation	Updates presence information for a Presentity for a specified Presence Source	no	Removes presence information for a Presentity for a specified Presence Source
Individual Presence Source attribute	/{userId}/presenceSources/{presenceSourceId}/[ResourceRelPath]	<i>The data structure corresponds to an element within the PresenceSource structure pointed out by the request-URL.</i> (Used for PUT/GET)	Retrieves the value of a specified presence attribute for a specified Presence Source	Creates or updates a presence attribute for a specified Presence Source	no	Removes a presence attribute for a specified Presence Source
Persistent Presence Source	/{userId}/presenceSources/persistent	PresenceSource	Retrieves persistent presence	Creates or updates persistent	no	Removes persistent presence

Resource	URL Base URL: http://{serverRoot}/presence/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
			information for a Presentity	presence information for a Presentity		information for a Presentity.
Individual persistent Presence Source attribute	/{userId}/presenceSources/persistent/[ResourceRelPath]	<i>The data structure corresponds to an element within the PresenceSource structure pointed out by the request-URL.</i> (Used for PUT/GET)	Retrieves the value of a specified persistent presence attribute	Creates or updates a persistent presence attribute	no	Removes a persistent presence attribute.

Purpose: To allow Presentity to manage own content (e.g. pictures/avatars/icons)

Resource	URL Base URL: http://{serverRoot}/presence/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Presentity content list	/userId/content	ContentList	Retrieves all content identities related to a Presentity	no	no	no
Individual Presentity content	/userId/content/{contentId}	<i>Any MIME content</i>	Retrieves a specified content (e.g. a picture) for a Presentity NOTE: Watcher SHALL NOT use this operation	Creates or replaces a specified content for a Presentity on the server	no	Removes a specified content from the server
Presentity portrait icon	/userId/content/portraitIcon	<i>Any MIME content that represents an image</i>	Retrieves a portrait icon for a Presentity NOTE: Watcher SHALL NOT use this operation	Creates or replaces a portrait icon for a Presentity on the server and at the same time the server sets/updates the link to the icon	no	Removes a portrait icon for a Presentity from the server

Purpose: To allow Presentity to retrieve information about Watchers interested in the Presentity's presence information

Resource	URL Base URL: http://{serverRoot}/presence/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Watchers list	/{userId}/watchers	WatcherList	Retrieves a list of Watchers interested in the Presentity's presence information, including the current subscription status	no	no	no
Individual Watcher	/{userId}/watchers/{watcherUserId}	Watcher	Retrieves the current subscription status and the subscribed attributes for a specified Watcher	no	no	no

Purpose: To allow Presentity to control access to presence information for Watchers, member lists or domains.

Resource	URL Base URL: http://{serverRoot}/presence/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Authorization rules	/{userId}/authorization/rules	RuleList (Used for GET) Rule (Used for POST) common:ResourceReference (optional alternative for	Retrieves all authorization rules	no	Creates a new authorization rule	no

		POST response)				
Individual authorization rule	/ {userId}/authorization/rules / {ruleId}	Rule (Used for PUT/GET)	Retrieves a specified authorization rule	Updates a specified authorization rule	no	Removes a specified authorization rule
Individual authorization rule data	/ {userId}/authorization/rules / {ruleId}/[ResourceRelPath]	<i>The data structure corresponds to the element pointed out by the request-URL.</i> (Used for PUT/GET)	Retrieves a specified rule data for a specified authorization rule	Creates or updates a specified rule data for a specified authorization rule	no	Removes a specified rule data from a specified authorization rule

Purpose: To allow Watcher to retrieve presence information from a single Presentity or a Presence List

Resource	URL Base URL: http://{serverRoot}/presenc e/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Presence information by Watcher for a single Presentity	/{userId}/presenceContacts/{presentityUserId}	PresenceContact	Retrieves a composite presence information from a Presentity Note: Retrieved presence information MAY include presence information from several Presence Sources	no	no	no
Individual presence attribute by Watcher	/{userId}/presenceContacts/{presentityUserId}/[ResourceRelPath]	<i>The data structure corresponds to an element within the PresenceContact structure pointed out by the request-URL.</i>	Retrieves a specified presence attribute for a Presentity	no	no	no
Presence information by Watcher for a Presence List	/{userId}/presenceLists/{presenceListId}	PresenceList	Retrieves presence information for all Presentities in a specified Presence (member) List	no	no	no

Purpose: To allow Watcher to retrieve content from a Presentity

Resource	URL Base URL: http://{serverRoot}/presence/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Content by Watcher	/{userId}/PresenceContactsContent/{presentityUserId}/{contentId}	Any MIME content	Retrieves a specified content (e.g. picture) for a specified Presentity	no	no	no

Purpose: To allow a user (in both Watcher and Presentity role) to retrieve own subscriptions

Resource	URL Base URL: http://{serverRoot}/presence/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All subscriptions	/{userId}/subscriptions	SubscriptionList	Retrieves all active subscriptions for a user	no	no	no

Purpose: To allow Presentity to manage subscriptions for notifications on changes in Watchers information

Resource	URL Base URL: http://{serverRoot}/presence/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All Watchers subscriptions	/{userId}/subscriptions/watchersSubscriptions	WatchersSubscriptionList (Used for GET) WatchersSubscription (Used for POST) common:ResourceReference (optional alternative for	Retrieves all subscriptions related to the Watchers list	no	Creates a subscription for notifications on changes in the Watchers list	no

		POST response)				
Individual Watchers subscription	/ {userId}/subscriptions/watchersSubscriptions/{subscriptionId}	WatchersSubscription	Retrieves a specified subscription to changes in the Watchers list	Updates a specified subscription to the Watchers list	no	Terminates a specified subscription to the Watchers list

Purpose: To allow the server to inform Presentity about changes in Watchers subscription status

Resource	URL: <Specified by the client>	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Watchers notification	<Specified by the client when the subscription for notifications on changes in the Watchers information list is created, or during provisioning process>	WatchersNotification	no	no	Notifies the client about changes in the Watcher's subscription status	no

Purpose: To allow Watcher to manage own subscriptions for notifications on changes in presence information for a Presentity

Resource	URL Base URL: http://{serverRoot}/presence/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All presence subscriptions	/ {userId}/subscriptions/presenceSubscriptions	PresenceSubscriptionList	Retrieves all active subscriptions for presence information for all Presentities	no	no	no
Presence subscriptions for a single Presentity	/ {userId}/subscriptions/presenceSubscriptions/{presentityUserId}	PresenceSubscriptionList (Used for GET) PresenceSubscription (Used for POST)	Retrieves all active subscriptions for presence information for a specified	no	Creates a subscription for notification on changes in presence information for a	no

		common:ResourceReference (optional alternative for POST response)	Presentity		specified Presentity	
Individual presence subscription	/{userId}/subscriptions/presenceSubscriptions/{presentityUserId}/{subscriptionId}	PresenceSubscription (Used for GET/PUT)	Retrieves a specified active subscription for presence information	Updates a specified subscription for presence information	no	Terminates a specified subscription for presence information

Purpose: To allow the server to inform Watcher about changes in presence information for a Presentity

Resource	URL: <Specified by the client>	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Presence notification	<Specified by the client when the subscription for notification on changes in presence information for a single Presentity is created, or during provisioning process>	PresenceNotification	no	no	Notifies the client about changes in presence information for a single Presentity	no

Purpose: To allow Watcher to manage own subscriptions for notifications on changes in presence information for a Presence List

Resource	URL Base URL: http://{serverRoot}/presence/{apiVersion}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
All Presence List subscriptions	/{userId}/subscriptions/presenceListSubscriptions	PresenceListSubscriptionCollection	Retrieves all active Presence List subscriptions for all Presence Lists	no	no	no
Presence List subscriptions for a	/{userId}/subscriptions/presenceListSubscriptions/{pres	PresenceListSubscriptionCollection	Retrieves all active Presence	no	Creates a subscription for	no

single Presence List	enceListId}	(Used for GET) PresenceListSubscription (Used for POST) common:ResourceReference (optional alternative for POST response)	List subscriptions for a specified Presence List		notifications on changes in presence information for a specified Presence list	
Individual Presence List subscription	/{userId}/subscriptions/presenceListSubscriptions/{presenceListId}/{subscriptionId}	PresenceListSubscription (Used for GET/PUT)	Retrieves a specified Presence list subscription	Updates a specified Presence List subscription	no	Terminates a specified Presence List subscription

Purpose: To allow the server to inform Watcher about changes in presence information for Presentities in a Presence List

Resource	URL: <Specified by the client>	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
Presence List notification	<Specified by the client when the subscription for notifications on changes in presence information for a Presence List is created, or during provisioning process>	PresenceListNotification	no	no	Notifies the client about changes in presence information for a Presence List	no

5.2 Data Types

5.2.1 XML Namespaces

The namespace for the Presence data types is:

```
urn:oma:xml:rest:netapi:presence:1
```

The 'xsd' namespace is used in the present document to refer to the XML Schema data types defined in XML Schema [XMLSchema1, XMLSchema2]. The 'common' namespace is used in the present document to refer to the data types defined in [REST_NetAPI_Common]. The use of the names 'xsd' and 'common' is not semantically significant.

The XML schema for the data structures defined in the section below is given in [REST_SUP_Presence].

Applications following the RESTful Network API for Presence V 1.0 specification SHALL use the namespace urn:oma:xml:rest:netapi:presence:1.

Note: Server implementations can choose to also support the legacy namespace urn:oma:xml:rest:presence:1 for the Presence data types, in order to allow backwards-compatibility with [ParlayREST_Presence] applications. Use of this legacy namespace is deprecated and support is foreseen to be withdrawn in future versions of this specification. In messages sent from the server to the application, the legacy namespace is suggested to be used by the server if it was used by a legacy application in the corresponding request or subscription message

5.2.2 Structures

The subsections of this section define the data structures used in the Presence API.

Some of the structures can be instantiated as so-called root elements, i.e. they define the type of a representation of a so-called Heavy-weight Resource.

The column [ResourceRelPath] in the tables below, if used, includes relative resource paths for Light-weight Resource URLs that are used to access individual elements in the data structure (so-called Light-weight Resources). A string from this column needs to be appended to the corresponding Heavy-weight Resource URL in order to create Light-weight Resource URL for that particular element in the data structure. "Not applicable" means that individual access to that element is not supported. The root element and data type of the resource associated with the [ResourceRelPath] are defined by the Element and Type columns in the row that defines the [ResourceRelPath].

For structures that contain elements which describe a user identifier, the statements in section 6 regarding 'tel', 'sip' and 'acr' URI schemes apply.

5.2.2.1 Type: PresenceSourceList

This type describes a list of Presence Sources.

Element	Type	Optional	Description
presenceSource	PresenceSource [0..unbounded]	Yes	A list of Presence Sources
resourceURL	xsd:anyURI	No	Self referring URL

A root element named presenceSourceList of type PresenceSourceList is allowed in response bodies.

5.2.2.2 Type: PresenceSource

This type defines a set of parameters for the Presence Source.

Element	Type	Optional	[ResourceRelPath]	Description
clientCorrelator	xsd:string	Yes	Not applicable	<p>A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server.</p> <p>This element SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate Presence Source creation in such situations.</p> <p>In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.</p>
applicationTag	xsd:string	Yes	Not applicable	<p>A tag that the client MAY use to tag this particular resource on the server. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.</p> <p>This attribute SHALL NOT be present in case of persistent Presence Source.</p>
duration	xsd:int	Yes	duration	<p>Specifies the duration of the publication life time in seconds. When this time has elapsed the Presence Source will expire unless it has been refreshed.</p> <p>If the parameter is omitted, a default value specified by the server policy will be used for the publication life time.</p> <p>A too low value (including "0") will result in an error response. What is too low is defined by server policy. A too high requested value may be reduced by the server according to the service policy.</p> <p>This element SHALL NOT be present in case of persistent Presence Source.</p>

presence	Presence	Yes	Not applicable	Contains the actual presence attributes. This element SHALL be present in all requests and responses except in the response to GET request with a filter suppressing the element.
resourceURL	xsd:anyURI	Yes	Not applicable	Self referring URL. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named presenceSource of type PresenceSource is allowed in request and/or response bodies.

Note that the clientCorrelator is used for purposes of error recovery as specified in [REST_NetAPI_Common], and internal client purposes. The server is NOT REQUIRED to use the clientCorrelator value in any form in the creation of the URL of the resource. [REST_NetAPI_Common] provides a recommendation regarding the generation of the value of this field.

Please refer to section 5.2.2 for an explanation of the column [ResourceRelPath].

Note that applicationTag is used to enable a particular application instance to pick up a previously created resource (if it exists) and continue to operate on it. A typical usage is that a client will perform a GET on the parent resource and in the response receive a list of previously created resources from where the application is able to find its previously created resource. It is up to the client application how to construct the application tag. Please note that a typical usage of the client correlator is not enough for a stateless application to identify a previously created resource since it is uniquely generated every time a new resource is created.

5.2.2.3 Type: Presence

This type defines a set of presence attributes for a Presence Source.

Element	Type	Optional	[ResourceRelPath]	Description
person	PersonAttributes	Yes	person	The presence attributes related to a person
service	ServiceAttributes [0..unbounded]	Yes	service/{serviceId}/{version}	The presence attributes related to services. For description of 'serviceId' and 'version' see 5.2.2.5. The sub-elements 'serviceId' and 'version' of the type ServiceAttributes are key properties of service element and SHALL NOT be altered when this element is accessed as a Light-weight Resource.
device	DeviceAttributes [0..unbounded]	Yes	device/{deviceId}	The presence attributes related to devices. For description of 'deviceId' see 5.2.2.6. The sub-element 'deviceId' of the type DeviceAttribute is a key property for device element and SHALL NOT be altered when this element is accessed as a Light-weight Resource.

Please refer to section 5.2.2 for an explanation of the column [ResourceRelPath].

5.2.2.4 Type: PersonAttributes

This type defines a set of presence attributes that relate to a person.

Element	Type	Optional	[ResourceRelPath]	Description
activities	Activities	Yes	person/activities	The Presentity's activity (e.g. available, busy, lunch, etc.). See [RFC4480].
placeType	PlaceType	Yes	person/placeType	At what kind of place the Presentity is (e.g. home, office, etc.). See [RFC4480].
privacy	Privacy	Yes	person/privacy	The amount of privacy the user wants (e.g. public, quiet, etc.). See [RFC4480].
sphere	Sphere	Yes	person/sphere	The user's current environment (e.g. work, home). See [RFC4480].
mood	Mood	Yes	person/mood	The user's mood (e.g. angry, confused, happy, etc.). See [RFC4480].
placels	Placels	Yes	person/placels	Describes the properties of the place the user is currently at. See [RFC4480].
timeOffset	TimeOffset	Yes	person/timeOffset	Describes the number of minutes of offset from UTC that the user is currently at. See [RFC4480].
statusIcon	StatusIcon	Yes	person/statusIcon	Contains a link to an icon of the user. See [RFC4480].
class	xsd:token	Yes	person/class	Defines the particular class. See [RFC4480].
noteList	NoteList	Yes	person/noteList	Contains taglines of the user. See [RFC4479].
location	Location	Yes	person/location	Location of a person. See [RFC5491] and [RFC5139].
overridingWillingness	OverridingWillingness	Yes	person/overridingWillingness	The overriding willingness for a person. See [OMA_DDS].
linkList	LinkList	Yes	person/linkList	Defines labeled links for a person. See [OMA_DDS].
card	xsd:anyURI	Yes	person/card	URI to a business card. See [RFC4482].
displayName	xsd:string	Yes	person/displayName	A display name of a person. See [RFC4482].
homePage	xsd:anyURI	Yes	person/homePage	URI pointing to general information about a person. See [RFC4482].

icon	xsd:anyURI	Yes	person/icon	URI pointing to an image/icon of the person. See [RFC4482]. Note: It is recommended to use the StatusIcon for sharing icons/avatars between users.
map	xsd:anyURI	Yes	person/map	URI pointing to a map related to the person. See [RFC4482].
sound	xsd:anyURI	Yes	person/sound	URI pointing to a sound related to the person. See [RFC4482].
timestamp	xsd:dateTime	Yes	person/timestamp	Timestamp of the latest update. Mandatory in responses. See [RFC3863].
extended	ExtendedList	Yes	person/extended	Attributes for extended presence information

Please refer to section 5.2.2 for an explanation of the column [ResourceRelPath].

5.2.2.5 Type: ServiceAttributes

This type defines a set of presence attributes that relate to a service.

Element	Type	Optional	[ResourceRelPath]	Description
serviceld	xsd:token	No	Not applicable	Identifier of the service. It is a key property of the service and SHALL NOT be altered when included in the Light-weight Resource URL. NOTE: It is recommended that standards developing organizations making use of the serviceld element register it within OMNA Presence Service-Description Registry [OMNA].
version	xsd:token	No	Not applicable	The version of the specified service. It is a key property of the service and SHALL NOT be altered when included in the Light-weight Resource URL.
statusIcon	StatusIcon	Yes	service/{serviceld}/{version}/statusIcon	Contains a link to an icon of the user. See [RFC4480].
class	xsd:token	Yes	service/{serviceld}/{version}/class	Defines the particular class. See [RFC4480].
displayName	xsd:string	Yes	service/{serviceld}/{version}/displayName	A display name of a Service. See [RFC4482].
homePage	xsd:anyURI	Yes	service/{serviceld}/{version}/homePage	URI pointing to general information about a Service. See [RFC4482].
Icon	xsd:anyURI	Yes	service/{serviceld}/{version}/icon	URI pointing to an image/icon of the Service. See [RFC4482]. Note: It is recommended to use the StatusIcon for sharing icons/avatars between users.
map	xsd:anyURI	Yes	service/{serviceld}/{version}/map	URI pointing to a map related to the Service. See [RFC4482].
sound	xsd:anyURI	Yes	service/{serviceld}/{version}/sound	URI pointing to a sound related to the Service. See [RFC4482].
linkList	LinkList	Yes	service/{serviceld}/{version}/linkList	Defines labeled links for a Service. See [OMA_DDS].
serviceAvailability	OpenOrClosed	Yes	service/{serviceld}/{version}/serviceAvailability	Service specific availability. See [OMA_DDS].
serviceWillingness	OpenOrClosed	Yes	service/{serviceld}/{version}/serviceWillingness	Service specific willingness. See [OMA_DDS].
contact	Contact	Yes	service/{serviceld}/{version}/contact	A contact address for a Service. See [RFC3863].
sessionParticipation	OpenOrClosed	Yes	service/{serviceld}/{version}/sessionParticipation	Indicates a participation in a session. See [OMA_DDS].
registrationState	ActiveOrTerminated	Yes	service/{serviceld}/{version}/registrationState	The registration state for a Service. See [OMA_DDS].
barringState	ActiveOrTerminated	Yes	service/{serviceld}/{version}/barringState	The barring state for a Service. See [OMA_DDS].
sessionAnswerMode	AutomaticOrManual	Yes	service/{serviceld}/{version}/sessionAnswerMode	Indicates answer mode for a session. See [OMA_DDS].

devices	DeviceIdentityList	Yes	service/{serviceld}/{version}/devices	Identify devices which this particular Service is related to. See [RFC4479].
timestamp	xsd:dateTime	Yes	service/{serviceld}/{version}/timestamp	Timestamp of the latest update. Mandatory in responses. See [RFC3863].
extended	ExtendedList	Yes	service/{serviceld}/{version}/extended	Attributes for extended presence information

Please refer to section 5.2.2 for an explanation of the column [ResourceRelPath].

5.2.2.6 Type: DeviceAttributes

This type defines a set of presence attributes that relate to a device.

Element	Type	Optional	[ResourceRelPath]	Description
deviceld	xsd:anyURI	No	Not applicable	Identifier of the device (e.g. 'sip' URI, 'tel' URI, 'acr' URI). See [RFC4479]. It is a key property of the device and SHALL NOT be altered when included in the Light-weight Resource URL.
class	xsd:token	Yes	device/{deviceld}/class	Defines the particular class. See [RFC4480].
location	Location	Yes	device/{deviceld}/location	Location of a device. See [RFC5491] and [RFC5139].
networkAvailability	NetworkAvailability	Yes	device/{deviceld}/networkAvailability	The network availability for a device. See [OMA_DDS].
timestamp	xsd:dateTime	Yes	device/{deviceld}/timestamp	Timestamp of the latest update. Mandatory in responses. See [RFC3863].
extended	ExtendedList	Yes	device/{deviceld}/extended	Attributes for extended presence information

Please refer to section 5.2.2 for an explanation of the column [ResourceRelPath].

5.2.2.7 Type: ContentList

This type describes a list of content stored on the server.

Element	Type	Optional	Description
content	ContentData [0..unbounded]	Yes	The list of content stored on the server.
resourceURL	xsd:anyURI	No	Self referring URL

A root element named contentList of type ContentList is allowed in response bodies.

5.2.2.8 Type: ContentData

This type describes a content stored on the server.

Element	Type	Optional	Description
link	Common:Link	No	Link to the content instance where the actual content is stored.
contentType	xsd:string	Yes	The content type of the stored content (e.g. MIME type: image/jpeg).
eTag	xsd:string	Yes	HTTP ETag identifier that includes version information related to the stored content. It can be used to detect if the content has been updated compared with a previous retrieval of the content.
fSize	xsd:int	Yes	The file size of the content in bytes (e.g. 102400)
resolution	xsd:string	Yes	The resolution of the content (used for instance if the content is an image). The value of the string is of the type "width x height" (e.g. 640x480) where width and height are specified in number of pixels.

5.2.2.9 Type: WatcherList

This type describes a list of Watchers for presence information.

Element	Type	Optional	Description
watcher	Watcher [0..unbounded]	Yes	Contains an array of Watchers subscribing for presence information for Presentity.
resourceURL	xsd:anyURI	No	Self referring URL.

A root element named watcherList of type WatcherList is allowed in response bodies.

5.2.2.10 Type: Watcher

This type defines a set of parameters for a Watcher.

Element	Type	Optional	Description
watcherUserId	xsd:anyURI	No	The Watcher subscribing for the data (e.g. 'sip' URI, 'tel' URI, 'acr' URI). In case that the Watcher has requested that its user identity is not revealed to the Presentity, watcherUserId could be, for example, specified as "sip:anonymous@anonymous.invalid"
displayName	xsd:string	Yes	An optional display name of the Watcher
resourceStatus	ResourceStatus	No	Describes the state of the Watcher subscription.
subscribedAttribute	xsd:anyURI [0..unbounded]	Yes	Contains a list of relative paths according to the [ResourceRelPath] in sections 5.2.2.3, 5.2.2.4, 5.2.2.5 and 5.2.2.6.
resourceURL	xsd:anyURI	No	Self referring URL

A root element named watcher of type Watcher is allowed in response bodies.

5.2.2.11 Type: RuleList

This type describes a list of authorization rules.

Element	Type	Optional	Description
rule	Rule [0..unbounded]	Yes	Contains a list of all authorization rules.
resourceURL	xsd:anyURI	No	Self referring URL

A root element named ruleList of type RuleList is allowed in response bodies.

5.2.2.12 Type: Rule

This type defines a set of parameters for an authorization rule.

Element	Type	Optional	[ResourceRelPath]	Description
ruleName	xsd:ID	No	Not applicable	A name associated with the rule. It is a key property of the rule and SHALL NOT be altered when included in the Light-weight Resource URL.
watcherUserId	xsd:anyURI [1..unbounded]	Choice	watchers/{watcherUserId}	Contains a list of Watcher identities (e.g. 'sip' URI, 'tel' URI, 'acr' URI).
memberListId	xsd:string [1..unbounded]	Choice	memberLists/{memberListId}	Contains a list of member list identities.
domainName	xsd:string [1..unbounded]	Choice	domains/{domainName}	Contains a list of domain names.
anonymous	(empty)	Choice	Not applicable	Indicates that this rule applies for requests from anonymous.
otherUser	(empty)	Choice	Not applicable	Allows the client to specify a default behavior for unknown users.
decision	DefaultDecisionValue	No	Not applicable	The authorization decision for the rule
presenceFilter	xsd:anyURI [0..unbounded]	Yes	Not applicable	Contains filter indicating which presence attributes the Watchers are allowed to see. Please refer to the column [ResourceRelPath] in sections 5.2.2.3, 5.2.2.4, 5.2.2.5 and 5.2.2.6 for possible values of the presenceFilter with the following clarifications: The 'servicelId' MAY be specified using a "*" meaning that the rule applies to all services. The 'version' MUST always be specified using "*". The 'deviceId' MAY be specified using a "*" meaning that the rule applies to all devices. An empty or no-existing filter means that the Watchers have access to all presence attributes.

resourceURL	xsd:anyURI	Yes	Not applicable	Self referring URL. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.
-------------	------------	-----	----------------	---

A root element named rule of type Rule is allowed in request and/or response bodies.

XSD modeling use a “choice” to select either watcherUserId, memberListId, domainName, anonymous or otherUser.

Please refer to section 5.2.2 for an explanation of the column [ResourceRelPath].

5.2.2.13 Type: PresenceList

This type describes a list of presence contacts.

Element	Type	Optional	Description
presenceContact	PresenceContact [0..unbounded]	Yes	Contains presence information structure for each Presentity in the Presence List.
resourceURL	xsd:anyURI	No	Self referring URL

A root element named presenceList of type PresenceList is allowed in response bodies.

5.2.2.14 Type: PresenceContact

This type defines a set of parameters for a presence contact.

Element	Type	Optional	Description
presentityUserId	xsd:anyURI	No	Represents the owner of the presence information (e.g. 'sip' URI, 'tel' URI, 'acr' URI).
resourceStatus	ResourceStatus	Yes	Indicates the status of the Watcher in relation to the Presentity. This element MUST only be included when PresenceContact is used within the 'PresenceList' data type.
presence	Presence	Yes	The actual presence information for the Presentity
resourceURL	xsd:anyURI	No	Self referring URL

A root element named presenceContact of type PresenceContact is allowed in response bodies.

5.2.2.15 Type: SubscriptionList

This type describes a list of subscriptions.

Element	Type	Optional	Description
presenceSubscriptionList	PresenceSubscriptionList	Yes	Contains an array of presence subscriptions for individual users.
presenceListSubscriptionCollection	PresenceListSubscriptionCollection	Yes	Contains an array of Presence List subscriptions for Presence Lists.
watchersSubscriptionList	WatchersSubscriptionList	Yes	Contains a list of Watchers subscriptions.
resourceURL	xsd:anyURI	No	Self referring URL

A root element named subscriptionList of type SubscriptionList is allowed in response bodies.

5.2.2.16 Type: WatchersSubscriptionList

This type describes a list of subscriptions for Watchers.

Element	Type	Optional	Description
watchersSubscription	WatchersSubscription [0..unbounded]	Yes	Contains an array of Watchers subscriptions.
resourceURL	xsd:anyURI	No	Self referring URL

A root element named watchersSubscriptionList of type WatchersSubscriptionList is allowed in response bodies.

5.2.2.17 Type: WatchersSubscription

This type defines a set of parameters for a subscription to Watchers.

Element	Type	Optional	Description
presentityUserId	xsd:anyURI	Yes	Identifies the Presentity for which the subscription is created towards (e.g. 'sip' URI, 'tel' URI, 'acr' URI). Mandatory in responses. The client SHALL NOT be allowed to update the presentityUserId in a PUT request.
callbackReference	common:CallbackReference	No	Client's notification endpoint and parameters. Contains the callback URL on which notifications will be sent to for the duration of the subscription.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate subscription creation in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it..
applicationTag	xsd:string	Yes	A tag that the client MAY use to tag this particular resource on the server. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
duration	xsd:int	Yes	Specifies the duration of the subscription in seconds. When this time has elapsed, the subscription will expire unless it has been refreshed. The server SHALL always include the remaining duration of the subscription in the response. A too high requested value MAY be reduced by the server according to the service policy. If the parameter is omitted, a default value specified by the server policy will be used as the subscription life time.
resourceStatusFilter	ResourceStatus [0..unbounded]	Yes	Indicates the desired Watchers subscription statuses that the Presentity is interested to get notifications about. If the parameter is omitted or there is an empty filter it means monitoring all states.

frequency	xsd:int	Yes	Maximum frequency of notifications, expressed as minimum time in seconds between notifications.
resourceURL	xsd:anyURI	Yes	Self referring URL. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named watchersSubscription of type WatchersSubscription is allowed in request and/or response bodies. Regarding the clientCorrelator and applicationTag elements, the note in section 5.2.2.2 applies.

5.2.2.18 Type: WatchersNotification

This type defines a set of parameters for the notifications about Watchers.

Element	Type	Optional	Description
presentityUserId	xsd:anyURI	No	Identifies the Presentity for which the notification is related to (e.g. 'sip' URI, 'tel' URI, 'acr' URI). Normally it is the same user who created the subscription.
callbackData	xsd:string	Yes	The 'callbackData' element as passed by the application in the 'callbackReference' element when creating a subscription to notifications on changes in Watchers list. See [REST_NetAPI_Common] for details.
resourceStatus	ResourceStatus	No	Describes the state for the subscription for Watchers.
watcherList	WatcherList	Yes	Contains a list of Watchers including corresponding subscription status. This element is only present if the resourceStatus=Active.
link	common:Link [0..unbounded]	Yes	Link to other resources that are in relationship with the resource.

A root element named watchersNotification of type WatchersNotification is allowed in watcher notification request.

5.2.2.19 Type: PresenceSubscriptionList

This type describes a list of presence subscriptions.

Element	Type	Optional	Description
presenceSubscription	PresenceSubscription [0..unbounded]	Yes	Can contain an array of presence subscriptions.
resourceURL	xsd:anyURI	No	Self referring URL

A root element named presenceSubscriptionList of type PresenceSubscriptionList is allowed in response bodies.

5.2.2.20 Type: PresenceSubscription

This type defines a set of parameters for the subscription for presence information.

Element	Type	Optional	Description
presentityUserId	xsd:anyURI	Yes	Identifies the Presentity for which the subscription is created towards (e.g. 'sip' URI, 'tel' URI, 'acr' URI). Mandatory in responses. The client SHALL NOT be allowed to update the presentityUserId in a PUT request. If presentityUserId is also part of the request URL, the two MUST have the same value.
callbackReference	common:CallbackReference	No	Client's notification endpoint and parameters. Contains the callback URL on which notifications will be sent to for the duration of the subscription.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate subscription creation in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
applicationTag	xsd:string	Yes	A tag that the client MAY use to tag this particular resource on the server. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
anonymous	(empty)	Yes	Allows the Watcher to request that its user identity is not revealed to the Presentity.
duration	xsd:int	Yes	Specifies the duration of the subscription in seconds. When this time has elapsed the subscription will expire unless it has been refreshed. The server SHALL always include the remaining duration of the subscription in the response. A too high requested value may be reduced by the server according to the service policy If the parameter is omitted, a default value specified by the server policy will be used for the subscription life time.

presenceFilter	xsd:anyURI [0..unbounded]	Yes	Allows the Watcher to indicate what presence information he/she is interested in. The desired attributes are indicated with relative paths according to the [ResourceRelPath] in sections 5.2.2.3, 5.2.2.4, 5.2.2.5 and 5.2.2.6 with the following clarifications: The 'serviceld', 'version' and 'deviceid' MAY be specified using a "*" meaning that the filter applies to several services and devices respectively. If the parameter is omitted or there is an empty filter it means monitoring of all attributes.
frequency	xsd:int	Yes	Maximum frequency of notifications (expressed as minimum time in seconds between notifications).
resourceURL	xsd:anyURI	Yes	Self referring URL. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named presenceSubscription of type PresenceSubscription is allowed in request and/or response bodies. Regarding the clientCorrelator and applicationTag elements, the note in section 5.2.2.2 applies.

5.2.2.21 Type: PresenceNotification

This type defines a set of parameters for the presence notifications.

Element	Type	Optional	Description
presentityUserId	xsd:anyURI	No	Identifies the Presentity for which the notification is related to (e.g. 'sip' URI, 'tel' URI, 'acr' URI).
callbackData	xsd:string	Yes	The 'callbackData' element as passed by the application in the 'callbackReference' element when creating a subscription to notifications on changes in presence information for the Presentity. See [REST_NetAPI_Common] for details.
resourceStatus	ResourceStatus	No	Indicates the status of the subscription for the Presentity.
presence	Presence	Yes	The actual presence information for the Presentity. This element is only present if the resourceStatus=Active.
link	common:Link [0..unbounded]	Yes	Link to other resources that are in relationship with the resource.

A root element named presenceNotification of type PresenceNotification is allowed in presence notification request.

5.2.2.22 Type: PresenceListSubscriptionCollection

This type describes a collection of Presence List subscriptions.

Element	Type	Optional	Description
presenceListSubscription	PresenceListSubscription [0..unbounded]	Yes	Can contain an array of Presence List subscriptions.
resourceURL	xsd:anyURI	No	Self referring URL

A root element named presenceListSubscriptionCollection of type PresenceListSubscriptionCollection is allowed in response bodies.

5.2.2.23 Type: PresenceListSubscription

This type defines a set of parameters for the Presence List subscription.

Element	Type	Optional	Description
presenceListId	xsd:anyURI	Yes	Identifies the Presence List for which the subscription is created towards (e.g. 'sip' URI, 'tel' URI, 'acr' URI). Mandatory in responses. The client SHALL NOT be allowed to update the presenceListId in a PUT request. If presenceListId is also part of the request URL, the two MUST have the same value.
callbackReference	common:CallbackReference	No	Client's notification endpoint and parameters. Contains the callback URL on which notifications will be sent to for the duration of the subscription.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate subscription creation in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
applicationTag	xsd:string	Yes	A tag that the client MAY use to tag this particular resource on the server. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
anonymous	(empty)	Yes	Allows the Watcher to request that its user identity is not revealed to the Presentity.
duration	xsd:int	Yes	Specifies the duration of the subscription in seconds. When this time has elapsed the subscription will expire unless it has been refreshed. The server SHALL always include the remaining duration of the subscription in the response. A too high requested value may be reduced by the server according to service policy. If the parameter is omitted, a default value specified by the server policy will be used for the subscription life.

presenceFilter	xsd:anyURI [0..unbounded]	Yes	Allows the Watcher to indicate what type of presence information he/she is interested in. The desired attributes are indicated with relative paths according to the [ResourceRelPath] in sections 5.2.2.3, 5.2.2.4, 5.2.2.5 and 5.2.2.6 with the following clarifications: The 'serviceld', 'version' and 'deviceld' MAY be specified using a "*" meaning that the filter applies to several services and devices respectively. If the parameter is omitted or there is an empty filter it means monitoring of all attributes.
frequency	xsd:int	Yes	Maximum frequency of notifications (expressed as minimum time in seconds between notifications).
resourceURL	xsd:anyURI	Yes	Self referring URL. The resourceURL SHALL NOT be included in POST requests by the client, but MUST be included in POST requests representing notifications by the server to the client, when a complete representation of the resource is embedded in the notification. The resourceURL MUST also be included in responses to any HTTP method that returns an entity body, and in PUT requests.

A root element named presenceListSubscription of type PresenceListSubscription is allowed in request and/or response bodies.

Regarding the clientCorrelator and applicationTag elements, the note in section 5.2.2.2 applies.

5.2.2.24 Type: PresenceListNotification

This type defines a set of parameters for the Presence List notifications.

Element	Type	Optional	Description
presenceListId	xsd:anyURI	No	Identifies the Presence List for which the notification is related to.
callbackData	xsd:string	Yes	The 'callbackData' element as passed by the application in the 'callbackReference' element when creating a subscription to notifications on changes in presence information for a Presence List. See [REST_NetAPI_Common] for details
resourceStatus	ResourceStatus	No	Indicates the state of the subscription.
presenceList	PresenceList	Yes	Contains data for each Presententity in the Presence List. This element is only present if the resourceStatus value is set to "Active".
link	common:Link [0..unbounded]	Yes	Link to other resources that are in relationship with the resource.

A root element named presenceListNotification of type PresenceListNotification is allowed in presence notification request.

5.2.2.25 Type: Activities

The type defines a set of parameters for activities. It is inherited from [RFC4480].

Element	Type	Optional	Description
activityValue	ActivityValue [1..unbounded]	No	The value of the attribute as specified in the URI.
note	common:Language	Yes	A textual description of what the user is currently

	String		doing. The language of the text SHOULD be defined by populating the attribute 'xml:lang' of this element.
other	xsd:string	Yes	Only applicable in case activityValue is set to "ActivitiesOther"
from	xsd:dateTime	Yes	Indicates an absolute time from which time the attribute is expected to be valid.
until	xsd:dateTime	Yes	Indicates an absolute time until which time the attribute is expected to be valid.

5.2.2.26 Type: PlaceType

The type defines a set of parameters for the type of place. It is inherited from [RFC4480].

Element	Type	Optional	Description
placeTypeValue	PlaceTypeValue [1..unbounded]	No	Indicates the type of place the person is currently at.
note	common:Language String	Yes	A comment about the current place the person is located at. The language of the text SHOULD be defined by populating the attribute 'xml:lang' of this element.
other	xsd:string	Yes	A textual description of what type of place the person is located in. Only applicable when the placeTypeValue element is set to "PlaceOther".
until	xsd:dateTime	Yes	Indicates an absolute time the attribute is expected to be valid.

5.2.2.27 Type: Privacy

The type defines a set of parameters for privacy. It is inherited from [RFC4480].

Element	Type	Optional	Description
privacyValue	PrivacyValue [1..unbounded]	No	Contains the value(s) of the privacy attribute.
note	common:Language eString	Yes	A textual description of the privacy. The language of the text SHOULD be defined by populating the attribute 'xml:lang' of this element.

5.2.2.28 Type: Sphere

The type defines a set of parameters for the sphere. It is inherited from [RFC4480].

Element	Type	Optional	Description
sphereValue	SphereValue	No	Contains the value of the sphere attribute.
<any element>	< type is defined by the schema which implements the element >	Yes	Optional element which is applicable in case sphereValue is set to "SphereOther" only. Note that element <any element> can be any element from any other namespace (schema) than the target namespace, which defines the value of the attribute. Type of such element is defined by the schema implementing the element. In XML implementations, the element must be qualified with the namespace prefix.

5.2.2.29 Type: Mood

The type defines a set of parameters for mood. It is inherited from [RFC4480].

Element	Type	Optional	Description
moodValue	MoodValue [1..unbounded]	No	Contains the value(s) of the mood attribute.
note	common:Language String	Yes	A textual description of the mood for a person. The language of the text SHOULD be defined by populating the attribute 'xml:lang' of this element.
other	xsd:string	Yes	Only applicable in case moodValue is set to "MoodOther"
until	xsd:dateTime	Yes	Indicates an absolute time the attribute is expected to be valid.

5.2.2.30 Type: Placels

This type defines the properties of the place the Presentity is currently at, such as the levels of light and noise. This information can be used by a Watcher to determine the type of communication that is likely to be successful.

Element	Type	Optional	Description
placelsAudio	PlacelsAudio	Yes	Describes place conditions for audio communication.
placelsVideo	PlacelsVideo	Yes	Describes place conditions for video communication.
placelsText	PlacelsText	Yes	Describes place conditions for real-time and instant-messaging communication.

5.2.2.31 Type: TimeOffset

This type defines a set of parameters for the time offset. It describes the number of minutes of offset from UTC that the user is currently at.

Element	Type	Optional	Description
timeOffset	xsd:int	No	Number of minutes of offset from UTC that the user is currently at.
until	xsd:dateTime	Yes	Indicates an absolute time the attribute is expected to be valid.

5.2.2.32 Type: StatusIcon

This type defines a set of parameters for the status or portrait icon. It includes a URI pointing to an image that represents the current status or portrait/avatar of the user.

Element	Type	Optional	Description
statusIconAddress	xsd:anyURI	No	URL pointing to the content (icon)
contentType	xsd:string	Yes	The content-type related to the content
eTag	xsd:string	Yes	HTTP ETag identifier for the addressed content. The Presentity MAY specify an eTag (i.e. version) of the content allowing the Watcher to detect when the content has been updated.

fSize	xsd:int	Yes	The size of the content in bytes (e.g. 102400)
resolution	xsd:string	Yes	The resolution of the content (used for instance if the content points to an image). The value of the string is of the type "width x height" (e.g. 640x480) where width and height are specified in number of pixels.
until	xsd:dateTime	Yes	Indicates an absolute time the content is expected to be valid.

5.2.2.33 Type: NoteList

This type describes a list of notes. The note parameter is inherited from [RFC4479].

Element	Type	Optional	Description
note	common:Language String [1..unbounded]	No	Contains a list of taglines. The language of the text SHOULD be defined by populating the attribute 'xml:lang' of this element.

5.2.2.34 Type: Location

This defines a set of parameters for the location. It is inherited from [RFC5491], [RFC4119], and [RFC5139].

Element	Type	Optional	Description
circle	CircleData	Choice	Contains parameters for definition of location in a form of a circle (e.g. latitude, longitude, and radius)
civicAddress	CivicAddress	Choice	Contains parameters for definition of location in a form of a civic address (e.g. country, city, street, post code etc.)
retentionExpiry	xsd:dateTime	No	Specifies an absolute date at which time the location information is no longer valid..

XSD modelling use a "choice" to select either circle or civicAddress.

5.2.2.35 Type: CircleData

This defines a set of parameters that describe a circle.

Element	Type	Optional	Description
latitude	xsd:float	No	Latitude of center point
longitude	xsd:float	No	Longitude of center point
radius	xsd:float	Yes	Radius of circle around center point in meters

5.2.2.36 Type: CivicAddress

This type defines a set of parameters for the civic address. The parameter names are inherited from [RFC5139].

Element	Type	Optional	Description
country	xsd:token	Yes	Two-letter according to [ISO.3166-2].
A1	xsd:string	Yes	National subdivisions (state, region, province, prefecture)
A2	xsd:string	Yes	County, parish, gun (JP), district (IN)
A3	xsd:string	Yes	City, township, shi (JP)
A4	xsd:string	Yes	City division, borough, city district, ward, chou (JP)
A5	xsd:string	Yes	Neighborhood, block
A6	xsd:string	Yes	Group of streets below the neighborhood level
PRM	xsd:string	Yes	Road pre-modifier
PRD	xsd:string	Yes	Leading street direction
RD	xsd:string	Yes	Primary road or street
STS	xsd:string	Yes	Street suffix
POD	xsd:string	Yes	Trailing street suffix
POM	xsd:string	Yes	Road post-modifier
RDSEC	xsd:string	Yes	Road section
RDBR	xsd:string	Yes	Road branch
RDSUBBR	xsd:string	Yes	Road sub-branch
HNO	xsd:string	Yes	House number, numeric part only.
HNS	xsd:string	Yes	House number suffix
LMK	xsd:string	Yes	Landmark or vanity address
LOC	xsd:string	Yes	Additional location information
FLR	xsd:string	Yes	Floor
NAM	xsd:string	Yes	Name (residence, business or office occupant)
PC	xsd:string	Yes	Postal code
BLD	xsd:string	Yes	Building (structure)
UNIT	xsd:string	Yes	Unit (apartment, suite)
ROOM	xsd:string	Yes	Room
SEAT	xsd:string	Yes	Seat (desk, cubicle, workstation)
PLC	xsd:string	Yes	Place-type
PCN	xsd:string	Yes	Postal community name
POBOX	xsd:string	Yes	Post office box (P.O. box)
ADDCODE	xsd:string	Yes	Additional Code

5.2.2.37 Type: OverridingWillingness

This type defines a set of parameters for the overriding willingness.

Element/Attribute	Type	Optional	Description
overridingWillingnessValue	OpenOrClosed	No	Value of the presence attribute describing user's general willingness to accept or not to accept any type of communication service, thus overriding individual settings for serviceWillingness described in 5.2.2.5.
until	xsd:dateTime	Yes	Specifies validity for the attribute. It is defined as an attribute when used in XML format.

5.2.2.38 Type: LinkList

This type defines a set of parameters for the link list. It enables the client to set one or more links to different type of content and distribute them to its Watchers. It is inherited from [OMA_DDS].

Element/Attribute	Type	Optional	Description
link	xsd:anyURI [0..unbounded]	Yes	The address for the link. Contains a list of links. A link can contain an URI pointing to any type of resource. When used to address a REST resource, the link element corresponds to the 'href' attribute from 'Link' data type described in [REST_NetAPI_Common].
label	xsd:string	Yes	Label for the link. The Presentity can provide a description of the link. It is defined as an attribute when used in XML format.
priority	xsd:decimal	Yes	Priority for the link. The Presentity can provide a priority used to indicate to the Watcher which link to select first. It is defined as an attribute when used in XML format
contentType	xsd:string	Yes	MIME content type for the link. The Presentity can, if known, specify the content type related to the addressed content allowing the Watcher to detect e.g. if it can render the addressed content. It is defined as an attribute when used in XML format.
rel	xsd:string	Yes	Describes relation between the URI and the resource. If the link is to a REST resource, it corresponds to the 'rel' attribute from the 'Link' data type as described in [REST_NetAPI_Common]. It is defined as an attribute when used in XML format.
eTag	xsd:string	Yes	HTTP ETag identifier of the addressed content. The Presentity MAY specify an eTag (i.e. version) of the addressed content allowing the Watcher to detect that the content has been updated in case it is e.g. caching the content. It is defined as an attribute when used in XML format.
fSize	xsd:int	Yes	The file size of the addressed content in bytes (e.g. 102400). The Presentity MAY specify the size of the addressed content allowing the Watcher to detect e.g. how much bandwidth an upload of the addressed content requires. It is defined as an attribute when used in XML format.
resolution	xsd:string	Yes	The resolution of the addressed content. The value of the string is of the type "width x height" (e.g. 640x480) where width and height are specified in number of pixels. The Presentity can specify the resolution of the addressed content (used for instance if the link points to an image). It is defined as an attribute when used in XML format.

5.2.2.39 Type: Contact

This type defines a set of parameters for the contact. It enables the client to set a contact address for the service.

Element/Attribute	Type	Optional	Description
contactAddress	xsd:anyURI	No	A contact address for the service.
priority	xsd:decimal	Yes	Decimal number between 0 and 1 inclusive with at most 3 digits after the decimal point. Higher values indicate higher priority. It is defined as an attribute when used in XML format.

5.2.2.40 Type: DeviceIdentityList

This type describes a list of device identities. It enables the client to specify a number of device identities related to the particular service.

Element	Type	Optional	Description
deviceId	xsd:anyURI [1..unbounded]	No	A list of device identities related to the service (e.g. 'sip' URI, 'tel' URI, 'acr' URI).

5.2.2.41 Type: NetworkAvailability

This type describes a list of network availabilities. It enables the client to set the network availability for a device.

Element	Type	Optional	Description
network	Network [0..unbounded]	Yes	Represents the availability for a particular network.

5.2.2.42 Type: Network

This type defines a set of parameters that describe the network and its availability. The design of the data structure is aligned with [OMA-DDS-V2.1].

Element/Attribute	Type	Optional	Description
connectionStatus	ActiveOrTerminated	No	Indicates the current status of the connection for the corresponding network.
networkMode	HomeOrVisited	Yes	Indicates the current mode of the client connection.
id	xsd:token	No	The identity of the network (e.g. IMS, GSM, GPRS, 802.11x etc). It is defined as an attribute in XML format.

5.2.2.43 Type: ExtendedList

This type describes a list of extended presence attributes.

Element	Type	Optional	Description
attribute	AttributeValue [1..unbounded]	No	Contains one or more extended attributes.

5.2.2.44 Type: AttributeValue

This type defines a set of parameters for the presence attribute value. It enables the client to define a name and value for the extended presence attribute.

Element	Type	Optional	Description
name	xsd:string	No	Contains the name of the extended attribute.
value	xsd:string	Choice	Optional element; if present it provides the value of the extended attribute.
<any element>	< type is defined by the schema which implements the element>	Choice	Optional element; if present it provides the value of the extended attribute. Note that element <any element> can be any element from any other namespace (schema) than the target namespace, which defines the value of the extended attribute. Type of such element is defined by the schema implementing the element. In XML implementations, the element must be qualified with the namespace prefix.

XSD modelling uses an optional “choice” to select either a value or <any element>, or none of them.

5.2.3 Enumerations

The subsections of this section define the enumerations used in the Presence API.

5.2.3.1 Enumeration: ActivityValue

This enumeration defines possible values to describe the type of user activity. It is inherited from [RFC4480].

Enumeration	Description
Appointment	The user has an appointment.
Available	The user is available for communication.
Busy	The user is busy and is only available for urgent matters.
OnThePhone	The user is on the phone.
Steering	The user is driving a car / train / airplane, etc.
Meeting	The user is in a meeting.
Away	No idea what the user is doing, but he is away.
Meal	The user is eating.
Breakfast	The user is having breakfast.
Lunch	The user is having lunch.
Dinner	The user is having dinner.
PermanentAbsence	The user is away and will not return for an extended period.
Vacation	The user is on vacation.
Holiday	A scheduled national or local holiday.
Performance	The user is in a theatre / concert.
InTransit	The user is in the transit area of an (air) port.
Travel	The user is traveling.
Sleeping	The user is sleeping.
LookingForWork	The user is looking for (paid) work.
Playing	The user is occupying him- or her in amusement, sport, or other recreation.
Presentation	The user is giving a presentation, lecture, or participating in a formal round-table discussion.
Shopping	The user is visiting stores in search of goods or Services.
Spectator	The user is observing an event, such as a sports event.
TV	The user is watching television.
Working	The user is engaged in, typically paid, labor, as part of a profession or job.
Worship	The user is participating in religious rites.
ActivitiesUnknown	The activity of the user is unknown.
ActivitiesOther	The user is doing something not in this list.

5.2.3.2 Enumeration: PlaceTypeValue

This enumeration defines possible values for the type of a place the user is currently at. It is inherited from [RFC4480].

Enumeration	Description
Arena	The user is at an enclosed area used for sports events.
Home	The user is at home.
Office	The user is in an office.
PublicTransport	The user is on public transport.
Street	Walking on the street.
PublicPlace	The user is in a public place.
Hotel	The user is in a hotel.
Theatre	The user is in a theatre or concert.
Restaurant	The user is in a restaurant, coffee shop or, other public dining establishment.
School	The user is at school.
Industrial	The user is in an industrial building.
Quiet	The user is in a quiet area.
Noisy	The user is in a noisy area.
Aircraft	The user is on an aircraft.
Watercraft	The user is on a vessel for travel on water such as a boat or ship.
Automobile	The user is in a car.
Bus	The user is in a bus.
BusStation	The user is in a bus- station.
TrainStation	The user is in a train-station.
ShoppingArea	The user is in a shopping mall or shopping area.
Airport	The user is in an airport.
Train	The user is in a train.
Bank	The user is in a bank.
Bar	The user is in a bar.
Bicycle	The user is on a bicycle.
Cafe	The user is in a café; usually a small and informal establishment that serves various refreshments (such as coffee); coffee shop.
Classroom	The user is in an academic classroom or lecture hall.
Club	The user is in a dance club, nightclub, or discotheque.
Construction	The user is at a construction site.
ConventionCenter	The user is in a convention center or exhibition hall.
Government	The user is in a government building, such as those used by the legislative, executive, or judicial branches of governments, including court houses, police stations, and military installations.
Hospital	The user is in a hospital, hospice, medical clinic, mental institution, or doctor's office.
Library	The user is in a library.
Motorcycle	The user is on a motorcycle.
Outdoors	The user outside a building, in or into the open air, such as a park or city streets.
Parking	The user is in a parking lot or parking garage.
PlaceOfWorship	The user is at a religious site where congregations gather for religious observances, such as a church, chapel, meetinghouse, mosque, shrine, synagogue, or temple.
Prison	The user is in a prison, penitentiary, jail or a brig.
Residence	The user is in a private or residential setting.
Stadium	The user is in a stadium.
Store	The user is in a shop or store.
Truck	The user is in a truck.
Underway	The user is in a land-, water-, or aircraft that is underway (in motion).

Warehouse	The user is in a warehouse.
Water	The user is in, on, or above bodies of water, such as an ocean, lake, river, canal, or other waterway.
PlaceOther	The user is in a kind of place not listed here.

5.2.3.3 Enumeration: PrivacyValue

This enumeration defines possible values for privacy. It is inherited from [RFC4480].

Enumeration	Description
Audio	Inappropriate individuals are not likely to overhear audio communications.
Text	Inappropriate individuals are not likely to see text communications.
Video	Inappropriate individuals are not likely to see video communications.
Other	None of the other values applies.

5.2.3.4 Enumeration: SphereValue

This enumeration describes possible values for sphere. It is inherited from [RFC4480].

Enumeration	Description
Work	The user is acting within his work sphere, i.e. as a member of his company.
Home	The user is acting within his home sphere, i.e. as a private person.
Unknown	The current sphere is unknown.
Other	The user is acting neither within his work nor within his home sphere.

5.2.3.5 Enumeration: MoodValue

This enumeration describes possible values for mood. It is inherited from [RFC4480].

Enumeration	Description
Afraid	The user is afraid.
Amazed	The user is amazed.
Angry	The user is angry.
Annoyed	The user is annoyed.
Anxious	The user is anxious.
Ashamed	The user is ashamed.
Bored	The user is bored.
Brave	The user is brave.
Calm	The user is calm.
Cold	The user is cold.
Confused	The user is confused.
Contented	The user is contented.
Cranky	The user is cranky.
Curious	The user is curious.
Depressed	The user is depressed.
Disappointed	The user is disappointed.
Disgusted	The user is disgusted.
Distracted	The user is distracted.
Embarrassed	The user is embarrassed.
Excited	The user is excited.
Flirtatious	The user is flirtatious.
Frustrated	The user is frustrated.
Grumpy	The user is grumpy.

Guilty	The user is guilty.
Happy	The user is happy.
Hot	The user is hot.
Humbled	The user is humbled.
Humiliated	The user is humiliated.
Hungry	The user is hungry.
Hurt	The user is hurt.
Impressed	The user is impressed.
InAwe	The user is in awe.
InLove	The user is in love.
Indignant	The user is indignant.
Interested	The user is interested.
Invincible	The user is invincible.
Jealous	The user is jealous.
Lonely	The user is lonely.
Mean	The user is mean.
MoodUnknown	The user's mood is unknown.
Moody	The user is moody.
Nervous	The user is nervous.
Neutral	The user is neutral.
Offended	The user is offended.
Playful	The user is playful.
Proud	The user is proud.
Relieved	The user is relieved.
Remorseful	The user is remorseful.
Restless	The user is restless.
Sad	The user is sad.
Sarcastic	The user is sarcastic.
Serious	The user is serious.
Shocked	The user is shocked.
Shy	The user is shy.
Sick	The user is sick.
Sleepy	The user is sleepy.
Stressed	The user is stressed.
Surprised	The user is surprised.
Thirsty	The user is thirsty.
Worried	The user is worried.
MoodOther	The user's current mood is not listed here.

5.2.3.6 Enumeration: PlacelsAudio

This enumeration defines possible values to describe the place the Presentity is currently at with respect to audio communication. It is inherited from [RFC4480].

Enumeration	Description
Noisy	The user is in a place with a level of background noise that makes audio communications difficult.
Ok	The environmental conditions are suitable.
Quiet	The user is in a place such as a library, restaurant, place of worship, or theatre that discourages noise, conversation, and other distractions.
Unknown	The place attributes are not known.

5.2.3.7 Enumeration: PlacelsVideo

This enumeration defines possible values to describe the place the Presentity is currently at with respect to video communication. It is inherited from [RFC4480].

Enumeration	Description
TooBright	The place is too bright for video communication.
Ok	The environmental conditions for video communication are acceptable.
Dark	The place is too dark for video communication.
Unknown	The environmental conditions for video communication are not known.

5.2.3.8 Enumeration: PlacelsText

This enumeration defines possible values to describe the place the Presentity is currently at with respect to real-time text and instant messaging. It is inherited from [RFC4480].

Enumeration	Description
Uncomfortable	The place is uncomfortable for typing or other text entry.
Inappropriate	The place is inappropriate for typing or other text entry.
Ok	The environmental conditions are suitable for typing or other text entry.
Unknown	The place attributes for text communication is not known.

5.2.3.9 Enumeration: OpenOrClosed

This enumeration defines possible values to describe the state of a presence attribute related to a service. It is inherited from [OMA_DDS].

Element	Description
Open	Depending on the attribute type the value indicates: <ul style="list-style-type: none"> - the service is available for use (for serviceAvailability attribute), or - user desires to use that particular service for communication (for serviceWillingness attribute), or - user is willing to use any service for communication (for overridingWillingnessValue attribute), or - a user is participating in at least one session of that particular service (for sessionParticipation attribute).
Closed	Depending on the attribute type the value indicates: <ul style="list-style-type: none"> - the service is not available for use (for serviceAvailability attribute), or - a user is not willing to use that particular service for communication (for serviceWillingness attribute), or - a user is not willing to use any service for communication (for overridingWillingnessValue attribute), or - a user is not participating in any session of that particular service (for sessionParticipation attribute).

5.2.3.10 Enumeration: ActiveOrTerminated

This enumeration defines possible values to describe the state of a presence attribute related to a service, or network connection. It is inherited from [OMA_DDS].

Element	Description
Active	Depending on the attribute type the value indicates: <ul style="list-style-type: none"> - a user has an active registration with that particular service (for registrationState attribute), or - a user has activated communication barring for that particular service (for barringState attribute), or - a device is connected to that particular network (for connectionStatus attribute).
Terminated	Depending on the attribute type the value indicates: <ul style="list-style-type: none"> - a user does not have an active registration with that particular service (for registrationState attribute), or - a user has deactivated communication barring for that particular service (for barringState attribute), or - a device is not connected to that particular network (for connectionStatus attribute).

5.2.3.11 Enumeration: AutomaticOrManual

This enumeration defines possible values to describe the mode of a presence attribute. It is inherited from [OMA_DDS].

Element	Description
Automatic	Indicates that a user will automatically accept an incoming session for that particular service.
Manual	Indicates that a user must make decision, and manually accept/reject the incoming session for that particular service.

5.2.3.12 Enumeration: HomeOrVisited

This enumeration defines possible values to describe client connection mode to the network. It is inherited from [OMA_DDS].

Element	Description
Home	Indicates that a device of a user is in user's home network.
Visited	Indicates that a device of a user is in user's visiting network.

5.2.3.13 Enumeration: ResourceStatus

This enumeration defines possible values to describe the status of the subscription.

Enumeration	Description
Active	Indicates that the subscription is active and authorized. (corresponds to 'active' state in [RFC3265 and RFC3857]).
Pending	Indicates that the subscription is awaiting an authorization decision. ('pending', 'waiting' and 'terminated/giveup' state in [RFC3265] and [RFC3857]).
TerminatedBlocked	Indicates that the subscription has been terminated. The subscription was blocked. ('terminated/rejected' state in [RFC3265] and [RFC3857]).

TerminatedTimeout	Indicates that the subscription has been terminated. The subscription was not refreshed in time before it expired. ('terminated/timeout' state in [RFC3265] and [RFC3857]).
TerminatedNoResource	Indicates that the subscription has been terminated. The intended resource does not exist. ('terminated/noresource' state in [RFC3265] and [RFC3857]).
TerminatedOther	Indicates that the subscription has been terminated of an unknown reason. ('terminated/probation and terminated/deactivated' state in [RFC3265] and [RFC3857]).

5.2.3.14 Enumeration: DefaultDecisionValue

This enumeration defines possible values for the default authorization decision.

Enumeration	Description
Allow	New Watchers are automatically granted to access presence information about the Presentity.
Block	New Watchers are automatically blocked from seeing any presence information.
PolitelyBlock	New Watchers are automatically politely blocked
Confirm	New Watchers have to be manually authorized before being able to get access to the presence information.

5.2.4 Values of the Link “rel” attribute

The “rel” attribute of the Link element is a free string set by the server implementation, to indicate a relationship between the current resource and an external resource. The following are possible strings (list is non-exhaustive, and can be extended):

- PresenceSourceList
- PresenceSource
- PresenceList
- PresenceContact
- Content
- ContentList
- PresenceSubscriptionList
- PresenceListSubscriptionCollection
- SubscriptionList
- PresenceSubscription
- PresenceNotification
- PresenceListSubscription
- PresenceListNotification
- WatcherList
- Watcher

- WatchersSubscriptionList
- WatchersSubscription
- WatchersNotification
- RuleList
- Rule

These values indicate the kind of resource that the link points to.

5.3 Sequence Diagrams

The following subsections describe the resources, methods and steps involved in typical scenarios for the usage of the Presence API. In the scenarios described below, there are two applications involved with different roles.

- Application 1 acts on behalf of Alice and has the pesentity role.
- Application 2 acts on behalf of Bob and has a Watcher role.

The sequences also try to show the interaction between these different roles.

In a sequence diagram, a step which involves delivering a notification is labeled with “POST or NOTIFY”, where “POST” refers to delivery via the HTTP POST method, and “NOTIFY” refers to delivery using the Notification Channel [REST_NetAPI_NotificationChannel].

5.3.1 Application start-up; publish presence, fetch Watcher information, subscribe to Watcher information

This figure below shows a scenario for starting or restarting an application instance of Application 1 on terminal 1 of Alice. Application 1 is a multi-terminal application and can publish different presence status from each of the terminals the application is running on. The sequence shows the following steps.

- Publishing information by application 1 on terminal 1 on behalf of Alice (step 1 - 2)
- Retrieving information about the Watchers of Alice (step 3 - 4)
- Subscribing to Watcher information for Alice, including the corresponding notification (step 5 - 6)

The notification URL (included in callbackReference) passed by the client during the subscription step can be a Client-side Notification URL, or a Server-side Notification URL. Refer to [REST_NetAPI_NotificationChannel] for sequence flows illustrating the creation of a Notification Channel and obtaining a Server-side Notification URL on the server-side, and the use of that Notification Channel by the client.

The resources:

- To fetch the list of presenceSources the following resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/presenceSources
- To create a new presenceSource the following resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/presenceSources
- To fetch the current Watchers this resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/watchers
- To fetch the list of subscriptions the following resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/watchersSubscriptions
- To subscribe to changes in the Watcher information the following resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/watchersSubscriptions

- The notification of the Watcher information is done on the notification URL provided by the application.

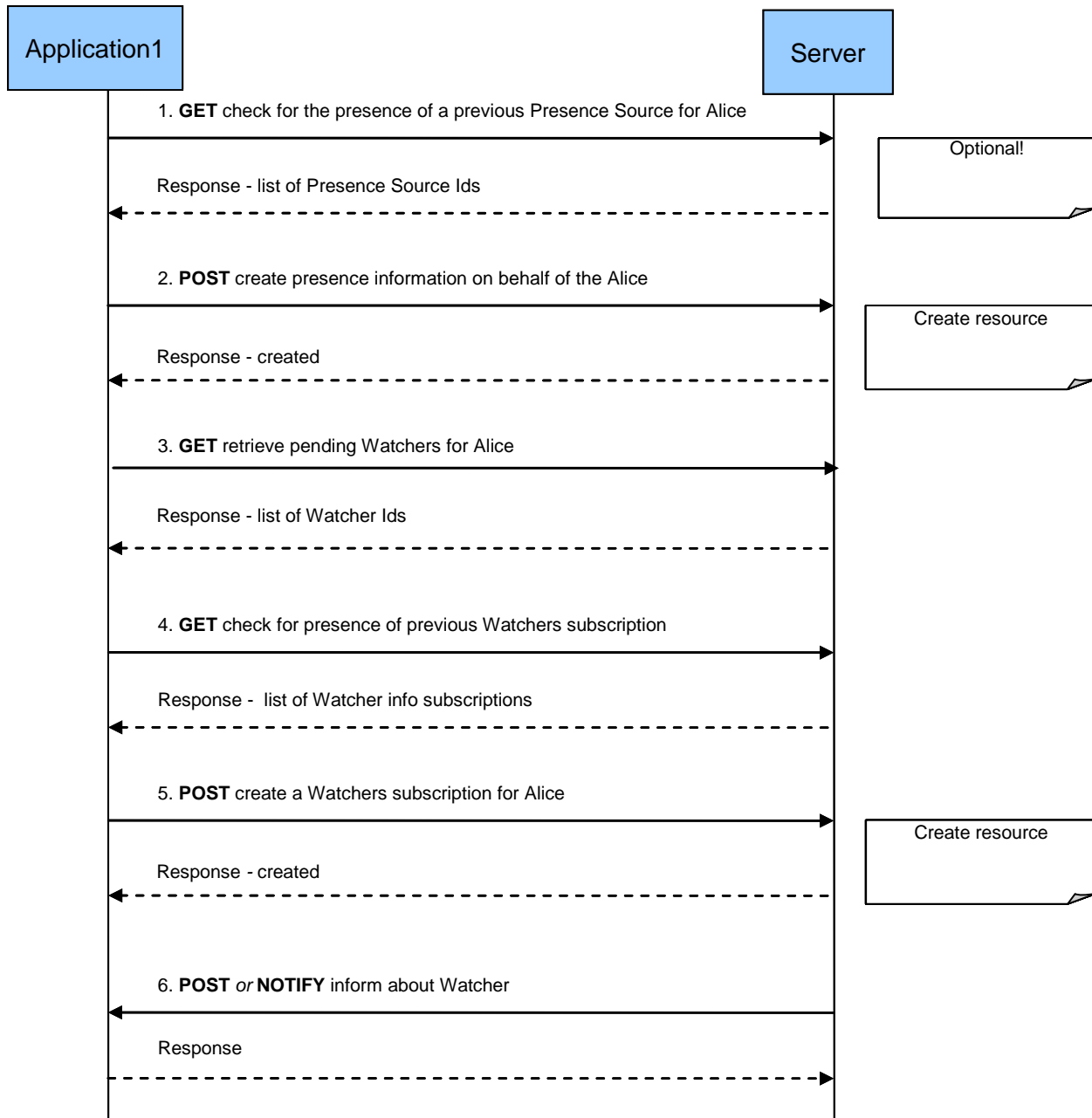


Figure 2 Creation of Presence Source, and subscription to Watchers information

Outline of the flows:

The idea is that the application 1 is stateless. i.e., it does not store any data between restarts. So in fact it does not know if the current situation is a start or a restart. The applicationTag is created by the client, and in this case they are created based on the application id and the terminal id (i.e.. app1_term1), to create a unique identifier per terminal per application. The (optional) applicationTag is used to retrieve resources that were created before the restart and can be reused after the restart.

1. Application 1 retrieves the list of presenceSources by using a GET method. The response returns a list of presenceSources. Each presenceSource will have a clientCorrelator and an applicationTag. The application tries to

find the resource that matches its applicationTag. This way it can find out the resourceURL of that resource.

In this scenario it is assumed that the resource was not found and the next step is to create a new resource. However, if the resource would have been found, the next step could be to do a GET on the resource, in order to synchronise the client with the server view of the resource (i.e., get the e-tag of the resource and get the current content). After that the client would be in a position to update the resource by using PUT (see later sequences on updating an existing presenceSourceId resource).

2. To create publication data (presenceSourceId) by application 1 on terminal 1 (2) the application uses POST method which includes applicationTag and a clientCorrelator that is generated to be unique.

In the response a 201 result is returned with the location of the resource.

3. Application 1 fetches the current Watchers by using a GET method.

A list of Watchers is returned. The result contains most data about the Watchers, except for some detailed information which is obtained in the following step.

4. Application 1 gets the list of Watcher information subscriptions by using GET method.. This is because in case of a restart it wants to reuse (and probably refresh) the same subscription that was used before the restart.

The response is a list of subscriptions which the application uses to find if there is a subscription matching its specific applicationTag (i.e. app1_term1). In this case (after a start) the resource is not found.

5. Application 1 subscribes to changes in the Watcher information by using a POST method. The application uses the same applicationTag as used in step 1 and 3 and a unique clientCorrelator.

The response contains a 201 with a location header pointing to the created resource.

6. The subscription in step 5 will result in a notification of the application with the current status of the Watcher info. The application provided notification URL is used in the notification. Alternatively, the application obtains the notifications using a Notification Channel [REST_NetAPI_NotificationChannel].

This makes step 3 superfluous, but it was included as an alternative way to fetch the same information by polling.

5.3.2 Adding a Watcher; subscribe for presence and updating of presence information.

This is a continuation of the sequence started in the previous section. More specifically the following preconditions apply:

- There is an active subscription for Watcher information by application 1 for the Presentity Alice.

This figure below shows the following scenario:

- Application 2 (a stateful application) subscribes to Alice's presence on behalf of Bob (and corresponding notify) (step 1 - 2)
- Watcher information notification since Bob becomes a pending Watcher (step 3)
- Adding Bob to the allowed list (step 4)
- Presence notification to Bob's application since Bob is now allowed to see the status of Alice (step 5)
- Watcher information notification to Alice's application since the status of the Watcher Bob changed to active (step 6)

The notification URL (included in callbackReference) passed by the client during the subscription step can be a Client-side Notification URL, or a Server-side Notification URL. Refer to [REST_NetAPI_NotificationChannel] for sequence flows illustrating the creation of a Notification Channel and obtaining a Server-side Notification URL on the server-side, and the use of that Notification Channel by the client.

The resources:

- To create a subscription for presence notifications for a single entity the following resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/presenceSubscriptions/{presentityUserId}
- The initial notification of the presence information is done on the notification URL provided by the application 2.
- The notification of the Watchers list is done on the notification URL provided by the application 1.
- To add a Watcher to the allowed list the following Light-weight Resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/authorization/rules/{ruleId}/[ResourceRelPath]
Where [ResourceRelPath] is a light-weight relative resource URL, and in this case it shall be replaced with "watchers/{watcherUserId}"
- The notification of the presence information is done on the notification URL provided by the application 2.
- The notification of the Watcher information is done on the notification URL provided by the application 1.

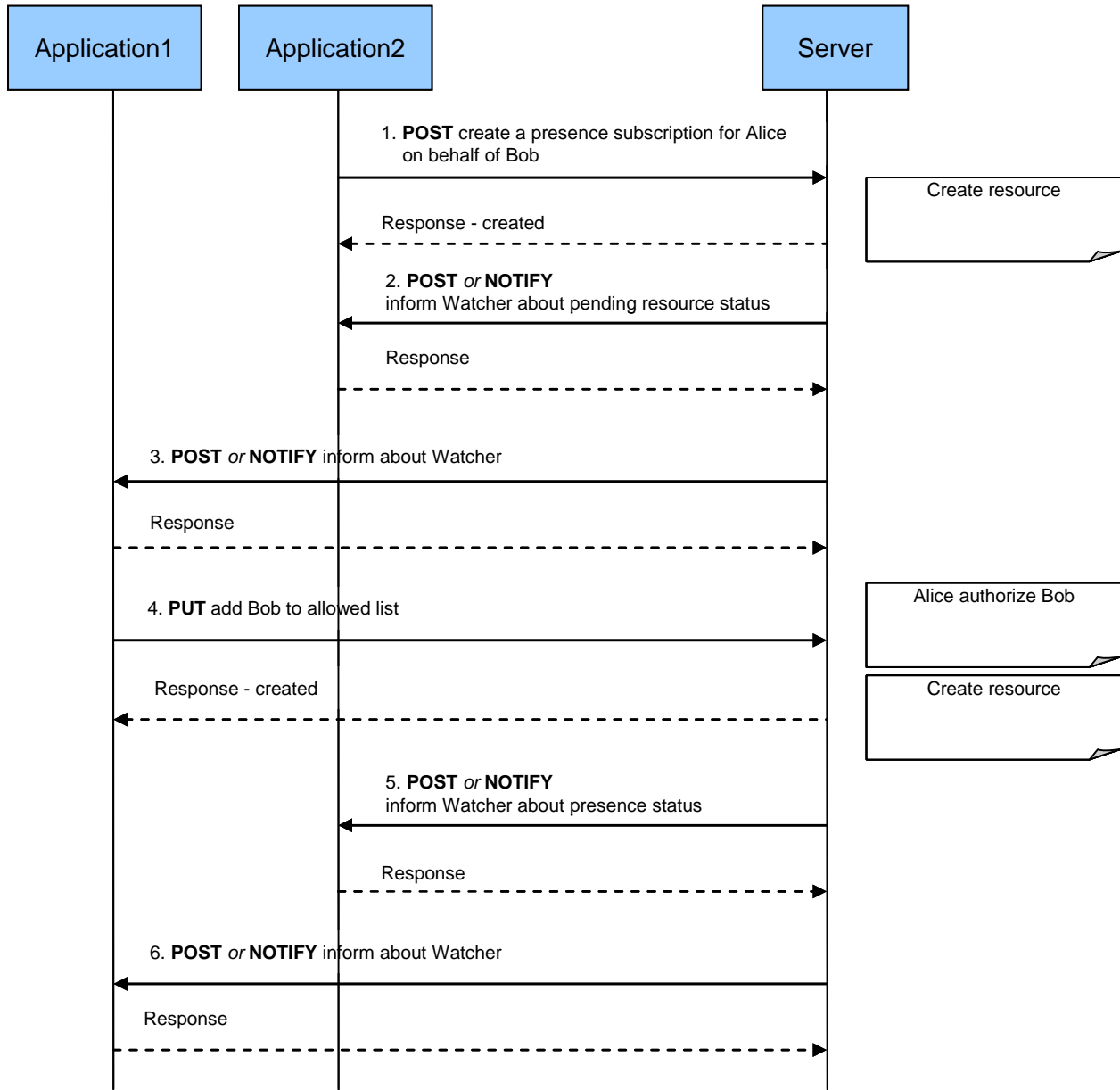


Figure 3 Subscription for presence information, and Watcher authorization

Outline of the flows:

Application 2 is a stateful application, i.e., it stored information between restarts. Therefore, it will remember the resources that were used in the previous session, and does not have to fetch any resources from the server to find if there is any resource that matches its applicationTag.

1. Application 2 creates a subscription to the presence information of Alice. Application 2 acts on behalf of Bob (the Watcher). The subscription is created by using a POST method with a client generated unique correlator. No applicationTag is included.

As a result a 201 created is returned. The location header is pointing to the created resource.

2. The server notifies application 2 about the current status of the subscription. Alternatively, the application obtains the notifications using a Notification Channel [REST_NetAPI_NotificationChannel].
In this case the subscription status is notified as being pending, since Bob is not yet authorized by Alice to view the presence status of Alice.
3. The server notifies application 1 about a new Watcher called Bob, whose status is unauthorized. Alternatively, the application obtains the notifications using a Notification Channel [REST_NetAPI_NotificationChannel].
4. Application 2 prompts Alice to request authorization of Bob. Alice allows Bob, so application 1 adds Bob to the allowed list of Alice, meaning that Bob is authorized to view the status of Alice. This is done by performing a PUT on the Light-weight Resource that includes Watcher Bob's identifier.

In this case Bob was not yet authorized, so the result is 201 created.

5. The server notifies application 2 about the current status of the subscription. Alternatively, the application obtains the notifications using a Notification Channel [REST_NetAPI_NotificationChannel].
In this case the subscription status is notified as being active, since Bob is now authorized by Alice to view the presence status of Alice. The notification will also contain the all of the current presence information of Alice that Bob is allowed to see according to the rules.
6. The server notifies application 1 about a new Watcher called Bob, whose status is now changed to active. Alternatively, the application obtains the notifications using a Notification Channel [REST_NetAPI_NotificationChannel].

5.3.3 Update of presence status

This is a continuation of the sequence started in the previous sections. More specifically the following preconditions apply:

- There is an active subscription for Watcher information by application 1 for the Presentity Alice.
- There is an active subscription for the presence of Presentity Alice by application 2 on behalf of Watcher Bob
- There is an active publication resource for Presentity Alice created by application 1.

This figure below shows the following scenario

- Application 1 uploads a new status-icon for Alice (step 1)
- Application 1 updates the presence information of Alice to with a link to the uploaded status-icon (step 2)
- Application 2 is notified about the changed presence information (step 3)
- Application 2 retrieves the content status-icon (step 4)

The resources:

- To put the content of the status icon the following resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/content/{contentId}
- To modify the published presence status the following Light-weight Resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/presenceSources/{presenceSourceId}/[ResourceRelPath]
Where [ResourceRelPath] is a light-weight relative resource URL, and in this case it shall be replaced with "person/statusIcon"
- The notification of the presence information is done on the notification URL provided by the application.

- To get the content of the status-icon the following resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/presenceContactsContent/{presentityUserId}/{contentId}

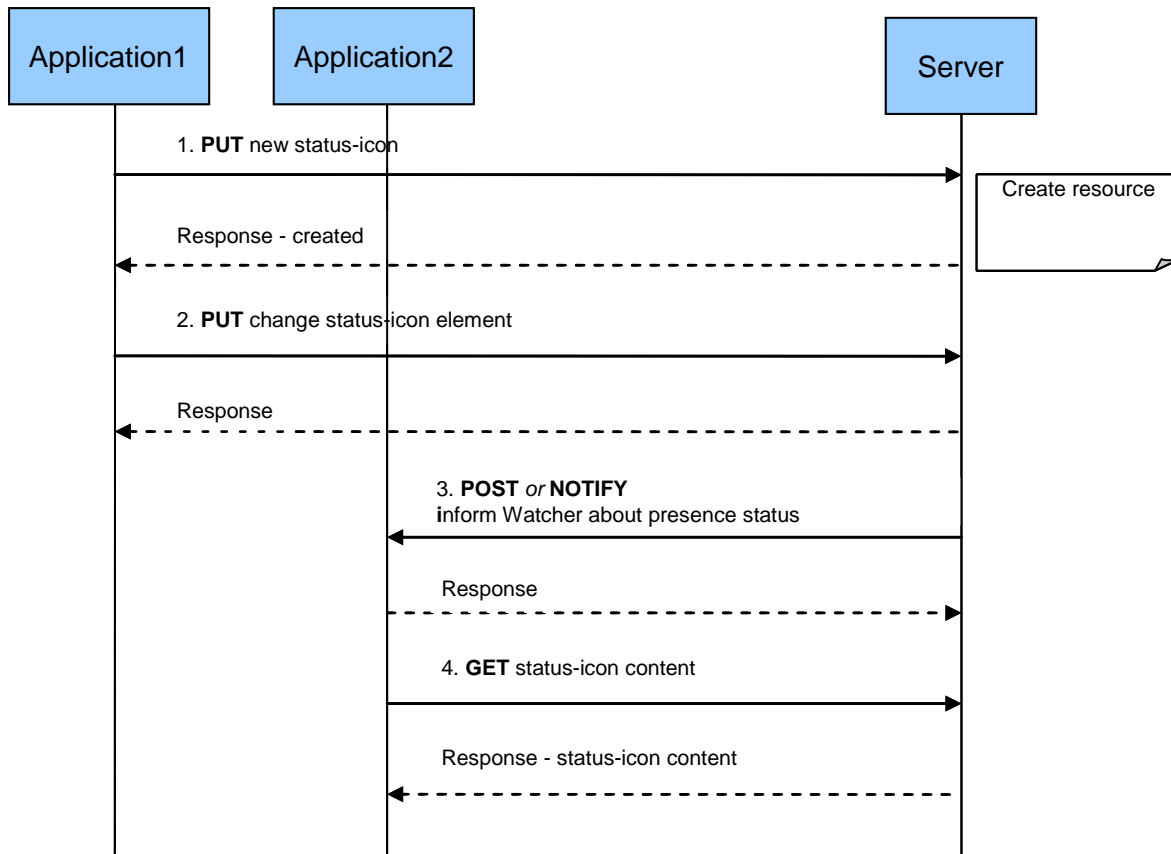


Figure 4 Update of presence information

Outline of the flows:

1. Application 1 uploads a new status-icon for the Alice. It includes the content of the icon as the body in the PUT request.
 The result depends on whether the content with that id already exists. In this case it is assumed that it did not yet exist, so a 201 created is returned.
2. Application 1 updates the status of the Alice, by only updating the status-icon part. It does by using a PUT method on the Light-weight Resource for status-icon.
 The result depends on whether the old presence information already contained a status-icon.
3. The server notifies Application 2 with the Watcher Bob about the status change of the Presentity Alice.
 Alternatively, the application obtains the notifications using a Notification Channel [REST_NetAPI_NotificationChannel]. The provided presence information contains the status-icon with a link to the location of the icon.
4. Application 2 fetches the content of the status-icon by using a GET method on the resource with the specified content id.
 The response contains the status icon content in the body.

5.3.4 Shutdown; remove resources

This is a continuation of the sequence started in the previous sections. More specifically the following preconditions apply:

- There is an active subscription for Watcher information by application 1 for the Presentity Alice.
- There is an active subscription for the presence of Presentity Alice by application 2 on behalf of Watcher Bob
- There is an active publication resource for Presentity Alice created by application 1.

This figure below shows the following scenario

- All the created subscriptions and the publications are terminated (but not the status-icon content)

The resources:

- To delete the presence subscription the following resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/presenceSubscriptions/{presentityUserId}/{subscriptionId}
- To delete the Watcher information subscription the following resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/watchersSubscriptions/{subscriptionId}
- To delete the publication of presence information the following resource is used:
http://{serverRoot}/presence/{apiVersion}/{userId}/presenceSources/{presenceSourceId}

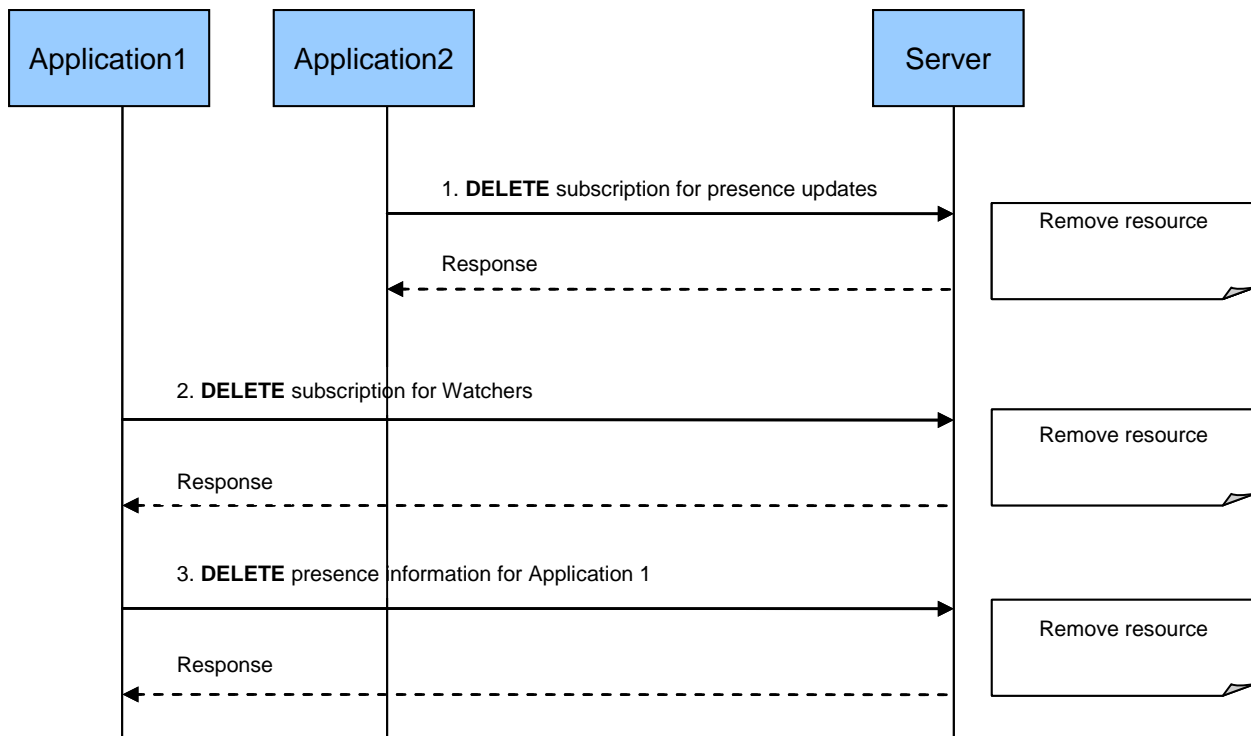


Figure 5 Termination of subscriptions for Watchers, and presence information

Outline of the flows:

1. Application 2 deletes the subscription resource for presence information by using a DELETE method on the resource with the specified subscription id.

Note that a DELETE on a subscription resource will NOT trigger any notifications!

2. Application 1 deletes the Watcher information subscription by using a DELETE method on the resource with specified subscription id.
3. Application 2 deletes the publication of presence information by using a DELETE method on the resource with the specified Presence Source id.

6. Detailed specification of the resources

The following applies to all resources defined in this specification regardless of the representation format (i.e. XML, JSON, application/x-www-form-urlencoded):

- Reserved characters in URL variables (parts of a URL denoted below by a name in curly brackets) **MUST** be percent-encoded according to [RFC3986]. Note that this always applies, no matter whether the URL is used as a Request URL or inside the representation of a resource (such as in “resourceURL” and “link” elements).
- If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of an MSISDN, it **MUST** be defined as a global number according to [RFC3966] (e.g. tel:+19585550100). The use of characters other than digits and the leading “+” sign **SHOULD** be avoided in order to ensure uniqueness of the resource URL. This applies regardless of whether the user identifier appears in a URL variable or in a parameter in the body of an HTTP message.
- If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of a SIP URI, it **MUST** be defined according to [RFC3261].
- If a user identifier (e.g. address, userId, etc) of type anyURI is in the form of an Anonymous Customer Reference (ACR), it **MUST** be defined according to [IETF_ACR_draft], i.e. it **MUST** include the protocol prefix 'acr:' followed by the ACR.
 - The ACR ‘auth’ is a supported reserved keyword, and **MUST NOT** be assigned as an ACR to any particular end user. See G.1.2 for details regarding the use of this reserved keyword.
- For requests and responses that have a body, the following applies: in the requests received, the server **SHALL** support JSON and XML encoding of the parameters in the body, and **MAY** support application/x-www-form-urlencoded parameters in the body. The Server **SHALL** return either JSON or XML encoded parameters in the response body, according to the result of the content type negotiation as specified in [REST_NetAPI_Common]. In notifications to the Client, the server **SHALL** use either XML or JSON encoding, depending on which format the client has specified in the related subscription. The generation and handling of the JSON representations **SHALL** follow the rules for JSON encoding in HTTP Requests/Responses as specified in [REST_NetAPI_Common].

6.1 Resource: Presence Sources

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/presenceSources

This resource is used to create a Presence Source with a lifetime. The Presence Source will expire if it is not refreshed in time. The resource is also used to retrieve all Presence Sources including the persistent presence document.

6.1.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the Presence Source is created for. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.1.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common]

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.1.3 GET

This operation is used to retrieve all Presence Sources for the specified user.

Supported parameters in the query string of the request URL are:

Name	Type/value	Optional	Description
PresenceSourceFilter	presenceSourceMetaData	Yes	Allows the Presentity to request that only metadata about its Presence Sources are returned in the response i.e. all presence information is filtered out. Example: “?presenceSourceFilter=presenceSourceMetaDat a”

6.1.3.1 Example: retrieving all Presence Sources for user (Informative)

6.1.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources HTTP/1.1
Host: example.com
Accept: application/xml
```

6.1.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSourceList xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceSource>
    <clientCorrelator>123</clientCorrelator>
    <applicationTag>myApp</applicationTag>
    <duration>3575</duration>
    <presence>
      <person>
        <mood>
          <moodValue>Happy</moodValue>
        </mood>
      </person>
    </service>
    <serviceId>org.openmobilealliance:IM-Session</serviceId>
    <version>1.0</version>
    <serviceAvailability>Open</serviceAvailability>
    <devices>
      <deviceId>mac:321</deviceId>
    </devices>
  </service>
</pr:presenceSourceList>
```

```

<device>
  <deviceId>mac:321</deviceId>
  <networkAvailability>
    <network id="GPRS">
      <connectionStatus>Active</connectionStatus>
    </network>
  </networkAvailability>
</device>
</presence>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123</resourceURL>
</presenceSource>
<presenceSource>
  <presence>
    <person>
      <noteList>
        <note xml:lang="en">I am on vacation!</note>
      </noteList>
    </person>
  </presence>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent</resourceURL>
</presenceSource>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources</resourceURL>
</pr:presenceSourceList>

```

6.1.3.2 Example: retrieving of all Presence Sources metadata using a filter (Informative)

6.1.3.2.1 Request

```

GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources?presenceSourceFilter=presenceSourceMetaData HTTP/1.1
Host: example.com
Accept: application/xml

```

6.1.3.2.2 Response

```

HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSourceList xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceSource>
    <clientCorrelator>123</clientCorrelator>
    <applicationTag>myApp</applicationTag>
    <duration>3575</duration>
    <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123</resourceURL>
  </presenceSource>
  <presenceSource>
    <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent</resourceURL>
  </presenceSource>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources</resourceURL>
</pr:presenceSourceList>

```

6.1.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC2616].

6.1.5 POST

This operation is used for creating a Presence Source with a specified time-to-live. The server can modify the client requested duration to a lower value according to the server policy if the requested value is too high,. If a too low value was requested an error code will be returned.

6.1.5.1 Example 1: creating Presence Source for user (Informative)

6.1.5.1.1 Request

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources HTTP/1.1
```

```
Host: example.com
```

```
Accept: application/xml
```

```
Content-Type: application/xml
```

```
Content-Length: nnnn
```

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSource xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <clientCorrelator>123</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <presence>
    <person>
      <mood>
        <moodValue>Happy</moodValue>
      </mood>
    </person>
    <service>
      <serviceId>org.openmobilealliance:IM-Session</serviceId>
      <version>1.0</version>
      <serviceAvailability>Open</serviceAvailability>
      <devices>
        <deviceId>mac:321</deviceId>
      </devices>
    </service>
    <device>
      <deviceId>mac:321</deviceId>
      <networkAvailability>
        <network id="GPRS">
          <connectionStatus>Active</connectionStatus>
        </network>
      </networkAvailability>
    </device>
  </presence>
</pr:presenceSource>
```

6.1.5.1.2 Response

```
HTTP/1.1 201 Created
```

```
Location: http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123
```

Date: Thu, 04 Jun 2009 02:51:59 GMT

Content-Type: application/xml

Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSource xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <clientCorrelator>123</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <presence>
    <person>
      <mood>
        <moodValue>Happy</moodValue>
      </mood>
    </person>
    <service>
      <serviceld>org.openmobilealliance:IM-Session</serviceld>
      <version>1.0</version>
      <serviceAvailability>Open</serviceAvailability>
      <devices>
        <deviceld>mac:321</deviceld>
      </devices>
    </service>
    <device>
      <deviceld>mac:321</deviceld>
      <networkAvailability>
        <network id="GPRS">
          <connectionStatus>Active</connectionStatus>
        </network>
      </networkAvailability>
    </device>
  </presence>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123</resourceURL>
</pr:presenceSource>
```

6.1.5.2 Example 2: creating Presence Source for user fails (Informative)

This example shows a policy exception where the maximum number of publication for Presentity has been reached.

6.1.5.2.1 Request

POST /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources HTTP/1.1

Host: example.com

Accept: application/xml

Content-Type: application/xml

Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSource xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <clientCorrelator>123</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <presence>
    <person>
      <mood>
```



```

    <moodValue>Happy</moodValue>
  </mood>
</person>
</presence>
</pr:presenceSource>

```

6.1.5.2.2 Response

```

HTTP/1.1 409 Conflict
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

```

```

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <link rel="PresenceSourceList"
    href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources"/>
  <policyException>
    <messageId>POL0260</messageId>
    <text>Maximum number of presence sources exceeded.</text>
  </policyException>
</common:requestError>

```

6.1.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC2616].

6.2 Resource: Individual Presence Source

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/presenceSources/{presenceSourceId}

This resource is used for managing of an existing Presence Source which includes: retrieval of a previously created Presence Source, update an existing Presence Source, or to remove a Presence Source. Only the creator of the Presence Source SHOULD be allowed to manage it.

6.2.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the Presence Source is created for. Examples: tel:+19585550100, acr:pseudonym123
presenceSourceId	Identifier of the Presence Source

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.2.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common]

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.2.3 GET

This operation is used to retrieve a particular Presence Source for the specified user.

6.2.3.1 Example 1: retrieving Presence Source

(Informative)

6.2.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123 HTTP/1.1
Host: example.com
Accept: application/xml
```

6.2.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSource xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <clientCorrelator>123</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>5237</duration>
  <presence>
    <person>
      <mood>
        <moodValue>Happy</moodValue>
      </mood>
    </person>
    <service>
      <serviceId>org.openmobilealliance:IM-Session</serviceId>
      <version>1.0</version>
      <serviceAvailability>Open</serviceAvailability>
      <devices>
        <deviceId>mac:321</deviceId>
      </devices>
    </service>
    <device>
      <deviceId>mac:321</deviceId>
      <networkAvailability>
        <network id="GPRS">
          <connectionStatus>Active</connectionStatus>
        </network>
      </networkAvailability>
    </device>
  </presence>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123</resourceURL>
```

```
</pr:presenceSource>
```

6.2.3.2 Example 2: retrieving Presence Source which does not exist(Informative)

6.2.3.2.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123 HTTP/1.1
Host: example.com
Accept: application/xml
```

6.2.3.2.2 Response

```
HTTP/1.1 404 Not Found
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <link rel="PresenceSourceList"
    href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123"/>
  <serviceException>
    <messageId>SVC1001</messageId>
    <text>Presence source does not exist</text>
  </serviceException>
</common:requestError>
```

6.2.4 PUT

This operation is used for updating of all presence attributes for a Presence Source.

6.2.4.1 Example: updating Presence Source (Informative)

6.2.4.1.1 Request

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123 HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSource xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <duration>7200</duration>
  <presence>
    <person>
      <mood>
        <moodValue>Invincible</moodValue>
      </mood>
    </person>
  </presence>
  <service>
```

```

<serviceId>org.openmobilealliance:IM-Session</serviceId>
<version>1.0</version>
<serviceAvailability>Closed</serviceAvailability>
<devices>
  <deviceId>mac:321</deviceId>
</devices>
</service>
<device>
  <deviceId>mac:321</deviceId>
  <networkAvailability>
    <network id="GPRS">
      <connectionStatus>Active</connectionStatus>
    </network>
  </networkAvailability>
</device>
</presence>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123</resourceURL>
</pr:presenceSource>

```

6.2.4.1.2 Response

```

HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSource xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <duration>7200</duration>
  <presence>
    <person>
      <mood>
        <moodValue>Invincible</moodValue>
      </mood>
    </person>
  <service>
    <serviceId>org.openmobilealliance:IM-Session</serviceId>
    <version>1.0</version>
    <serviceAvailability>Closed</serviceAvailability>
    <devices>
      <deviceId>mac:321</deviceId>
    </devices>
  </service>
  <device>
    <deviceId>mac:321</deviceId>
    <networkAvailability>
      <network id="GPRS">
        <connectionStatus>Active</connectionStatus>
      </network>
    </networkAvailability>
  </device>
</presence>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123</resourceURL>
</pr:presenceSource>

```

6.2.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT, DELETE' field in the response as per section 14.7 of [RFC2616].

6.2.6 DELETE

This operation is used for removing of all attributes for a Presence Source.

6.2.6.1 Example: removing Presence Source (Informative)

6.2.6.1.1 Request

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123 HTTP/1.1
Host: example.com
```

6.2.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.3 Resource: Individual Presence Source attribute

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/presenceSources/{presenceSourceId}/[ResourceRelPath]

This resource is used to update a particular presence attribute as well as extending the duration of the Presence Source.

This resource type can be used to access and manage parts of presence attributes. The resource URL consists of Heavy-weight Resource path, **http://{serverRoot}/presence/{apiVersion}/{userId}/presenceSources/{presenceSourceId}/**, and an extension of the resource URL path, which is relative resource path for a Light-weight Resource and it is represented by **[ResourceRelPath]**.

6.3.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the Presence Source is created for. Example: tel:+19585550100
presenceSourceId	Identifier of the Presence Source
[ResourceRelPath]	Relative resource path for a Light-weight Resource, consisting of a relative path down

	to an element in the data structure. For more information about the applicable values (strings) for this variable see 6.3.1.1.
--	--

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.3.1.1 Light-weight relative resource paths

The following table describes the types of Light-weight Resources that can be accessed by using this resource, applicable methods, and links to data structures that contain values (strings) for those relative resource paths.

Light-weight Resource type	Method supported	Description
Presence attribute groups	GET, PUT, DELETE	Enables access to presence attributes related to Person, Service or Device. See data structure 5.2.2.3 for possible values for the light-weight relative resource path.
Person attributes	GET, PUT, DELETE	Enables access to a single presence attribute related to a person. See data structure 5.2.2.4 for possible values for the light-weight relative resource path.
Service attributes	GET, PUT, DELETE	Enables access to a single presence attribute related to a service. See data structure 5.2.2.5 for possible values for the light-weight relative resource path.
Device attributes	GET, PUT, DELETE	Enables access to a single presence attribute related to a device. See data structure 5.2.2.6 for possible values for the light-weight relative resource path.
Duration of Presence Source	GET, PUT	Used to update the duration or retrieve the remaining life time of the Presence Source. See data structure 5.2.2.2 for the light-weight relative resource path.

6.3.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.3.3 GET

This operation is used to retrieve a particular presence attribute in the specified Presence Source. It may also be used to retrieve the remaining duration of the life time. If the Presence Source is not refreshed in time it will expire.

6.3.3.1 Example: retrieving individual presence attribute (Informative)

6.3.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123/person/mood HTTP/1.1
Host: example.com
Accept: application/xml
```

6.3.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:mood xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <moodValue>Happy</moodValue>
</pr:mood>
```

6.3.4 PUT

This operation is used to update (or create if it does not exist already) an individual presence attribute in the specified Presence Source. It may also be used to extend the duration of a Presence Source.

6.3.4.1 Example: updating individual presence attribute (Informative)

6.3.4.1.1 Request

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123/person/mood HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn
```

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:mood xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <moodValue>Excited</moodValue>
</pr:mood>
```

6.3.4.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:mood xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <moodValue>Excited</moodValue>
</pr:mood>
```

6.3.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT, DELETE' field in the response as per section 14.7 of [RFC2616].

6.3.6 DELETE

This operation is used to remove a particular presence attribute from a Presence Source.

6.3.6.1 Example: removing individual presence attribute (Informative)

6.3.6.1.1 Request

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123/person/mood HTTP/1.1
Host: example.com
```

6.3.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.4 Resource: Persistent Presence Source

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/presenceSources/persistent

This resource is used by the Presentity to manage persistent presence information. Persistent presence information is normally used for more static kind of presence information and does not have a time-to-live and hence will never expires. There is only one instance of the persistent Presence Source on the server.

6.4.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the persistent Presence Source is created for. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.4.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.4.3 GET

This operation is used to retrieve the persistent Presence Source for the specified user.

6.4.3.1 Example: retrieving persistent presence information (Informative)

6.4.3.1.1 Request

This example shows also an alternative way to indicate desired content type in response from the server, by using URL query parameter “?resFormat” which is described in [REST_NetAPI_Common].

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent?resFormat=XML HTTP/1.1
Host: example.com
```

6.4.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
ETag:"11"
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSource xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presence>
    <person>
      <noteList>
        <note xml:lang="en">Im on vacation!</note>
      </noteList>
    </person>
  </presence>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent</resourceURL>
</pr:presenceSource>
```

6.4.4 PUT

This operation is used to update (or create if it does not exist already) the persistent Presence Source for the specified user.

6.4.4.1 Example: updating persistent presence information (Informative)

This example illustrates a scenario where two clients operate on the persistent presence information and are using conditional headers to prevent one client overwriting the data created by another client.

6.4.4.1.1 Request

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent HTTP/1.1
Host: example.com
Accept: application/xml
If-Match: "10"
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
```

```
<pr:presenceSource xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presence>
    <person>
      <statusIcon>
        <statusIconAddress>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg</statusIconAddress>
        <contentType>image/jpeg</contentType>
        <eTag>123</eTag>
      </statusIcon>
      <noteList>
        <note xml:lang="en">My picture is updated!</note>
      </noteList>
    </person>
  </presence>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent</resourceURL>
</pr:presenceSource>
```

6.4.4.1.2 Response

```
HTTP/1.1 412 Precondition Failed
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

The request above failed because the other client has modified the resource since the last operation on it, which has led to the server updating the ETag. The client has to retrieve the resource (see the example in 6.4.3) in order to synchronize the ETag again. The response is examined (in order to possibly retain other data), and a new PUT request with the latest ETag value is initiated.

6.4.4.1.3 Request

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent HTTP/1.1
Host: example.com
Accept: application/xml
If-Match: "11"
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSource xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presence>
    <person>
      <statusIcon>
        <statusIconAddress>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg</statusIconAddress>
        <contentType>image/jpeg</contentType>
        <eTag>123</eTag>
      </statusIcon>
      <noteList>
        <note xml:lang="en">My picture is updated!</note>
      </noteList>
    </person>
  </presence>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent</resourceURL>
</pr:presenceSource>
```

6.4.4.1.4 Response

```

HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
ETag: "12"
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSource xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presence>
    <person>
      <statusIcon>
        <statusIconAddress>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg</statusIconAddress>
        <contentType>image/jpeg</contentType>
        <eTag>123</eTag>
      </statusIcon>
      <noteList>
        <note xml:lang="en">My picture is updated!</note>
      </noteList>
    </person>
  </presence>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent</resourceURL>
</pr:presenceSource>

```

6.4.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT, DELETE' field in the response as per section 14.7 of [RFC2616].

6.4.6 DELETE

This operation is used to remove the persistent Presence Source.

6.4.6.1 Example: removing persistent presence information (Informative)

6.4.6.1.1 Request

```

DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent HTTP/1.1
Host: example.com

```

6.4.6.1.2 Response

```

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

6.5 Resource: Individual persistent Presence Source attribute

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/presenceSources/persistent/[ResourceRelPath]

This resource is used to manage individual persistent presence attributes. Persistent presence information is normally used for more static kind of data and does not have a time-to-live and hence will never expire.

This resource type can be used to access and manage parts of presence attributes. The resource URL consists of Heavy-weight Resource path, **http://{serverRoot}/presence/{apiVersion}/{userId}/presenceSources/persistent/**, and an extension of the resource URL path, which is relative resource path for a Light-weight Resource, and it is represented by **[ResourceRelPath]**.

6.5.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the persistent presence attributes is managed for. Examples: tel:+19585550100, acr:pseudonym123
[ResourceRelPath]	Relative resource path for a Light-weight Resource, consisting of a relative path down to an element in the data structure. For more information about the applicable values (strings) for this variable, see 6.5.1.1.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.5.1.1 Light-weight relative resource paths

The following table describes the types of Light-weight Resources that can be accessed by using this resource, applicable methods, and the links to data structures that contain values (strings) for those relative resource paths.

Light-weight Resource type	Method supported	Description
Presence attribute groups	GET, PUT, DELETE	Enables access to presence attributes related to Person, Service or Device. See data structure 5.2.2.3 for possible values for the light-weight relative resource path.
Person attributes	GET, PUT, DELETE	Enables access to a single presence attribute related to a person. See data structure 5.2.2.4 for possible values for the light-weight relative resource path.
Service attributes	GET, PUT, DELETE	Enables access to a single presence attribute related to a service. See data structure 5.2.2.5 for possible values for the light-weight relative resource path.
Device attributes	GET, PUT, DELETE	Enables access to a single presence attribute related to a device. See data structure 5.2.2.6 for possible values for the light-weight relative resource path.

6.5.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.5.3 GET

This operation is used to retrieve a particular persistent presence attribute for the specified user.

6.5.3.1 Example: retrieving individual persistent presence attribute (Informative)

6.5.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent/person/statusIcon HTTP/1.1
Host: example.com
Accept: application/xml
```

6.5.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:statusIcon xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <statusIconAddress>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg</statusIconAddress>
  <contentType>image/jpeg</contentType>
  <eTag>123</eTag>
</pr:statusIcon>
```

6.5.4 PUT

This operation is used to update (or create if it does not exist already) a particular persistent presence attribute for the specified user.

6.5.4.1 Example: updating individual persistent presence attribute (Informative)

6.5.4.1.1 Request

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent/person/statusIcon HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:statusIcon xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <statusIconAddress>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg</statusIconAddress>
  <contentType>image/jpeg</contentType>
```

```
<eTag>456</eTag>
</pr:statusIcon>
```

6.5.4.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:statusIcon xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <statusIconAddress>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg</statusIconAddress>
  <contentType>image/jpeg</contentType>
  <eTag>456</eTag>
</pr:statusIcon>
```

6.5.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT, DELETE' field in the response as per section 14.7 of [RFC2616].

6.5.6 DELETE

This operation is used to remove a particular persistent presence attribute.

6.5.6.1 Example: removing individual persistent presence attribute (Informative)

6.5.6.1.1 Request

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent/person/mood HTTP/1.1
Host: example.com
```

6.5.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.6 Resource: Presentity content list

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/content

This resource is used to retrieve information about contents stored on the server. The result contains for example the URL for each uploaded content file. The file may consist of an icon/picture etc.

6.6.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the content is retrieved for. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.6.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.6.3 GET

This operation is used to retrieve a list of content for the specified user.

6.6.3.1 Example: retrieving list of available contents (Informative)

6.6.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/content HTTP/1.1
Host: example.com
Accept: application/xml
```

6.6.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<pr:contentList xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <content>
    <link rel="content" href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic003.jpg"/>
    <contentType>image/jpeg</contentType>
  </content>
  <content>
    <link rel="content" href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic004.jpg"/>
    <contentType>image/jpeg</contentType>
  </content>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content</resourceURL>
</pr:contentList>
```

6.6.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.6.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.6.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.7 Resource: Individual Presentity content

The resource used is:

`http://{serverRoot}/presence/{apiVersion}/{userId}/content/{contentId}`

This resource is used by Presentity to retrieve, upload, and remove an individual content (e.g. an icon). The uploaded content is related to the Presentity by using the "person/status-icon" attribute where the link to the content is stored.

6.7.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the content is managed for. Examples: tel:+19585550100, acr:pseudonym123
contentId	Contains an identifier of the content. The identifier may be structured as a relative path consisting of a directory and filename. Example: oma_status-icon/myPicture.jpg

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.7.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.7.3 GET

This operation is used by the Presentity to retrieve the content as specified in the URL.

6.7.3.1 Example: retrieving individual content by Presentity (Informative)

6.7.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg HTTP/1.1
Host: example.com
Accept: image/jpeg
```

6.7.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: image/jpeg
Content-Length: nnnn
```

data

6.7.4 PUT

This operation is used to upload new or modify the existing content on the server.

6.7.4.1 Example: uploading/updating individual content by Presentity (Informative)

6.7.4.1.1 Request

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: image/jpeg
Content-Length: nnnn
```

data

6.7.4.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.7.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT, DELETE' field in the response as per section 14.7 of [RFC2616].

6.7.6 DELETE

This operation is used to remove the content specified by the URL.

6.7.6.1 Example: removing individual content by Presentity (Informative)

6.7.6.1.1 Request

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg HTTP/1.1
Host: example.com
```

6.7.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.8 Resource: Watchers list

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/watchers

This resource is used by the Presentity to retrieve the list of Watchers that are interested in the Presentity's presence information including the current subscription status. The desired state is provided in a query parameter (e.g. state=pending or state=active).

A typical usage is to retrieve unauthorized users in order to decide whether to allow, block or politely block them.

6.8.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the Watcher list is retrieved for. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.8.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.8.3 GET

This operation is used to retrieve a list of Watchers (including corresponding subscription status) interested in the Presentity's presence information.

Supported parameters in the query string of the request URL are:

Name	Type/value	Optional	Description
resourceStatusFilter	ResourceStatus [0..unbounded]	Yes	Allows the Presentity to indicate which resource status for Watchers he/she is interested in. Possible values for Watchers resource status are specified in section 5.2.3.13. Example: "?resourceStatusFilter=Pending"

6.8.3.1 Example: retrieving list of Watchers

(Informative)

6.8.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/watchers?resourceStatusFilter=Pending HTTP/1.1
Host: example.com
Accept: application/xml
```

6.8.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<pr:watcherList xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <watcher>
    <watcherUserId>tel:+19585550101</watcherUserId>
    <displayName>Bob</displayName>
    <resourceStatus>Pending</resourceStatus>
    <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/watchers/
tel%3A%2B19585550101</resourceURL>
  </watcher>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/watchers</resourceURL>
</pr:watcherList>
```

6.8.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.8.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.8.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.9 Resource: Individual Watcher

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/watchers/{watcherUserId}

This resource is used to retrieve subscription status about an individual Watcher.

6.9.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the Watcher is retrieved for. Examples: tel:+19585550100, acr:pseudonym123
watcherUserId	Identity of the Watcher to retrieve information about. Examples: tel:+19585550101, acr:pseudonym124

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.9.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.9.3 GET

This operation is used by a Presentity to retrieve subscription status about an individual Watcher.

6.9.3.1 Example: retrieving individual Watcher

(Informative)

6.9.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/watchers/{tel%3A%2B19585550101 HTTP/1.1
Host: example.com
Accept: application/xml
```

6.9.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:watcher xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
```

```
<watcherUserId>tel:+19585550101</watcherUserId>
<displayName>Bob</displayName>
<resourceStatus>Pending</resourceStatus>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/watchers/tel%3A%2B19585550101</resourceURL>
</pr:watcher>
```

6.9.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.9.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.9.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.10 Resource: Authorization rules

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/authorization/rules

This resource is used by a Presentity to create and retrieve authorization rules. The authorization rules controls who will have access to Presentity's presence information. A Watcher may be authorized to all or a subset of the available presence attributes.

6.10.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the authorization rules are managed for. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.10.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.10.3 GET

This operation is used by a Presentity to retrieve all authorization rules.

6.10.3.1 Example: retrieving all authorization rules (Informative)

6.10.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules HTTP/1.1
Host: example.com
Accept: application/xml
```

6.10.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:ruleList xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <rule>
    <ruleName>allowList</ruleName>
    <watcherUserId>tel:+19585550102</watcherUserId>
    <watcherUserId>tel:+19585550104</watcherUserId>
    <decision>Allow</decision>
  </rule>
  <rule>
    <ruleName>blockList</ruleName>
    <memberListId>myBlockList</memberListId>
    <decision>Block</decision>
  </rule>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules</resourceURL>
</pr:ruleList>
```

6.10.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC2616].

6.10.5 POST

This operation is used by a Presentity to create a new authorization rule.

6.10.5.1 Example: creating an authorization rule (Informative)

6.10.5.1.1 Request

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn
```

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:rule xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <ruleName>otherUsers</ruleName>
  <otherUser/>
  <decision>Confirm</decision>
</pr:rule>
```

6.10.5.1.2 Response

HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule003
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:rule xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <ruleName>otherUsers</ruleName>
  <otherUser/>
  <decision>Confirm</decision>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule003</resourceURL>
</pr:rule>
```

6.10.5.2 Example 2: creating an authorization rule, response with resourceReference (Informative)

6.10.5.2.1 Request

POST /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:rule xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <ruleName>otherUsers</ruleName>
  <otherUser/>
  <decision>Confirm</decision>
</pr:rule>
```

6.10.5.2.2 Response

HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule003
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule003</resourceURL>
</common:resourceReference>
```

6.10.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC2616].

6.11 Resource: Individual authorization rule

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/authorization/rules/{ruleId}

This resource is used by a Presentity to manage an individual authorization rule.

6.11.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the individual authorization rule is managed for. Examples: tel:+19585550100, acr:pseudonym123
ruleId	Identity of the rule generated by the server

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.11.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.11.3 GET

This operation is used by a Presentity to retrieve an authorization rule.

6.11.3.1 Example: retrieving an authorization rule

(Informative)

6.11.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule001 HTTP/1.1
Host: example.com
Accept: application/xml
```


6.11.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:rule xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <ruleName>allowList</ruleName>
  <watcherUserId>tel:+19585550102</watcherUserId>
  <watcherUserId>tel:+19585550104</watcherUserId>
  <decision>Allow</decision>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule001</resourceURL>
</pr:rule>
```

6.11.4 PUT

This operation is used by a Presentity to update an authorization rule.

6.11.4.1 Example: updating an authorization rule

(Informative)

6.11.4.1.1 Request

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule001 HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:rule xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <ruleName>allowList</ruleName>
  <watcherUserId>tel:+19585550102</watcherUserId>
  <watcherUserId>tel:+19585550104</watcherUserId>
  <watcherUserId>tel:+19585550105</watcherUserId>
  <decision>Allow</decision>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule001</resourceURL>
</pr:rule>
```

6.11.4.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
```

```
<pr:rule xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <ruleName>allowList</ruleName>
  <watcherUserId>tel:+19585550102</watcherUserId>
  <watcherUserId>tel:+19585550104</watcherUserId>
  <watcherUserId>tel:+19585550105</watcherUserId>
  <decision>Allow</decision>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule001</resourceURL>
</pr:rule>
```

6.11.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT, DELETE' field in the response as per section 14.7 of [RFC2616].

6.11.6 DELETE

This operation is used by a Presentity to remove an authorization rule.

6.11.6.1 Example: removing an authorization rule (Informative)

6.11.6.1.1 Request

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule002 HTTP/1.1
Host: example.com
```

6.11.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.12 Resource: Individual authorization rule data

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/authorization/rules/{ruleId}/[ResourceRelPath]

This resource is used by Presentity to update an authorization rule by specifying the identity to authorize in the URL. Users, member lists or domains may be authorized using this operation. The resource URL consists of Heavy-weight Resource path, **http://{serverRoot}/presence/{apiVersion}/{userId}/authorization/rules/{ruleId}/**, and an extension of the resource URL path, which is relative resource path for a Light-weight Resource, and it is represented by **[ResourceRelPath]**.

6.12.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI

apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the authorization rule data is managed for. Examples: tel:+19585550100, acr:pseudonym123
ruleId	Identifier of the rule generated by the server
[ResourceRelPath]	Relative resource path for a Light-weight Resource, consisting of a relative path down to an element in the data structure. For more information about the applicable values (strings) for this variable, see 6.12.1.1.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.12.1.1 Light-weight relative resource paths

The following table describes the type of Light-weight Resources that can be accessed by using this resource, applicable methods, and the link to a data structure that contains values (strings) for those relative resource paths.

Light-weight Resource type	Method supported	Description
Watcher identities	GET, PUT, DELETE	Enables access to authorization data related to a specific authorization rule. See data structure 5.2.2.12 for possible values for the light-weight relative resource path.

6.12.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.12.3 GET

This operation is used by a Presentity to retrieve a user, member list or domain from an authorization rule.

6.12.3.1 Example: retrieving individual authorization rule data (Informative)

6.12.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule002/watchers/tel%3A%2B19585550102 HTTP/1.1
Host: example.com
Accept: application/xml
```

6.12.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<pr:watcherUserId xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
```

```
tel:+19585550102
</pr:watcherUserId>
```

6.12.4 PUT

This operation is used by a Presentity to authorize a user, member list or domain by including its identity in the request.

6.12.4.1 Example: updating individual authorization rule data (Informative)

6.12.4.1.1 Request

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule002/watchers/tel%3A%2B19585550103 HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:watcherUserId xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  tel:+19585550103
</pr:watcherUserId>
```

6.12.4.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:watcherUserId xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  tel:+19585550103
</pr:watcherUserId>
```

6.12.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT, DELETE' field in the response as per section 14.7 of [RFC2616].

6.12.6 DELETE

This operation is used by a Presentity to remove a user, member list or domain from an authorization rule.

6.12.6.1 Example: removing individual authorization rule data (Informative)

6.12.6.1.1 Request

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule002/watchers/tel%3A%2B19585550103
Host: example.com
```

6.12.6.1.2 Response

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

6.13 Resource: Presence information by Watcher

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/presenceContacts/{presentityUserId}

This resource is used by a Watcher to retrieve presence information about a single Presentity.

6.13.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Watcher retrieving the presence information from the Presentity. Examples: tel:+19585550101, acr:pseudonym124
presentityUserId	Identity of the Presentity owning the presence information. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.13.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.13.3 GET

This operation is used by a Watcher to retrieve presence information about Presentity.

Supported parameters in the query string of the Request URL are:

Name	Type/value	Optional	Description
presenceFilter	xsd:anyURI [0..unbounded]	Yes	Allows the Watcher to indicate what type of presence information he/she is interested in. The desired attributes are indicated with relative paths according to the [ResourceRelPath] in sections 5.2.2.3, 5.2.2.4, 5.2.2.5 and 5.2.2.6 with the following clarifications: The 'serviceld', 'version' and 'deviceld' MAY be specified using a "*" meaning that the filter applies to several services and devices respectively Example: "?presenceFilter=person/mood&presenceFilter=service/*/*/icon"
anonymous	empty	Yes	Allows the Watcher to request that its user identity is not revealed to the Presentity.

			Example: ?anonymous
--	--	--	---------------------

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.13.3.1 Example 1: retrieving all presence information for Presentity(Informative)

6.13.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100 HTTP/1.1
Host: example.com
Accept: application/xml
```

6.13.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceContact xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <presence>
    <person>
      <mood>
        <moodValue>Happy</moodValue>
      </mood>
      <noteList>
        <note xml:lang="en">Im on vacation!</note>
      </noteList>
    </person>
    <service>
      <serviceId>org.openmobilealliance:IM-Session</serviceId>
      <version>1.0</version>
      <serviceAvailability>Open</serviceAvailability>
      <devices>
        <deviceId>mac:321</deviceId>
      </devices>
    </service>
  </device>
  <deviceId>mac:321</deviceId>
  <networkAvailability>
    <network id="GPRS">
      <connectionStatus>Active</connectionStatus>
    </network>
  </networkAvailability>
</device>
</presence>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100</resourceURL>
</pr:presenceContact>
```

6.13.3.2 Example 2: retrieving presence for Presentity by using filter (Informative)

6.13.3.2.1 Request

```
GET
/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100?presenceFilter=person/mood&presen
ceFilter=service/org.openmobilealliance:IM-Session/1.0/serviceAvailability HTTP/1.1
Host: example.com
Accept: application/xml
```

6.13.3.2.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceContact xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <presence>
    <person>
      <mood>
        <moodValue>Happy</moodValue>
      </mood>
    </person>
    <service>
      <serviceId>org.openmobilealliance:IM-Session</serviceId>
      <version>1.0</version>
      <serviceAvailability>Open</serviceAvailability>
      <devices>
        <deviceId>mac:321</deviceId>
      </devices>
    </service>
  </presence>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100</res
  ourceURL>
</pr:presenceContact>
```

6.13.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.13.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.13.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.14 Resource: Individual presence attribute by Watcher

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/presenceContacts/{presentityUserId}/[ResourceRelPath]

This resource is used by a Watcher to retrieve an individual presence attribute from a single Presentity. The resource URL consists of Heavy-weight Resource path,

http://{serverRoot}/presence/{apiVersion}/{userId}/presenceContacts/{presentityUserId}/, and an extension of the resource URL path, which is relative resource path for a Light-weight Resource, and it is represented by **[ResourceRelPath]**.

6.14.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Watcher retrieving the presence information from the Presentity. Examples: tel:+19585550101, acr:pseudonym124
presentityUserId	Identity of the Presentity owning the presence information. Examples: tel:+19585550100, acr:pseudonym123
[ResourceRelPath]	Relative resource path for a Light-weight Resource, consisting of a relative path down to an element in the data structure. For more information about the applicable values (strings) for this variable, see 6.14.1.1.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.14.1.1 Light-weight relative resource paths

The following table describes the types of Light-weight Resources that can be accessed using this resource, applicable methods, and the links to data structures that contain values (strings) for those relative resource paths.

Light-weight Resource type	Method supported	Description
Presence attribute groups	GET, PUT, DELETE	Enables access to presence attributes related to Person, Service or Device. See data structure 5.2.2.3 for possible values for the light-weight relative resource path.
Person attributes	GET, PUT, DELETE	Enables access to a single presence attribute related to a person. See data structure 5.2.2.4 for possible values for the light-weight relative resource path.
Service attributes	GET, PUT, DELETE	Enables access to a single presence attribute related to a service.

		See data structure 5.2.2.5 for possible values for the light-weight relative resource path.
Device attributes	GET, PUT, DELETE	Enables access to a single presence attribute related to a device. See data structure 5.2.2.6 for possible values for the light-weight relative resource path.

6.14.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.14.3 GET

This operation is used by a Watcher to retrieve presence information about Presentity.

Supported parameters in the query string of the Request URL are:

Name	Type/value	Optional	Description
anonymous	empty	Yes	Allows the Watcher to request that its user identity is not revealed to the Presentity. Example: ?anonymous

6.14.3.1 Example: retrieving individual presence attribute for Presentity (Informative)

6.14.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100/person/noteList HTTP/1.1
Host: example.com
Accept: application/xml
```

6.14.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:noteList xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <note xml:lang="en">Im on vacation!</note>
</pr:noteList>
```

6.14.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.14.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.14.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.15 Resource: Presence information by Watcher for a Presence List

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/presenceLists/{presenceListId}

This resource is used by a Watcher to retrieve presence information about Presentities in a Presence List.

6.15.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Watcher retrieving the presence information from the Presence List. Examples: tel:+19585550101, acr:pseudonym124
presenceListId	Identity of the Presence List. Example: myFriends

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.15.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.15.3 GET

This operation is used by a Watcher to retrieve presence information about Presentities in a Presence List.

Supported parameters in the query string of the Request URL are:

Name	Type/value	Optional	Description
presenceFilter	xsd:anyURI [0..unbounded]	Yes	Allows the Watcher to indicate what type of presence information he/she is interested in. The desired attributes are indicated with resource relative paths according to the [ResourceRelPath] in sections 5.2.2.3, 5.2.2.4, 5.2.2.5 and 5.2.2.6 with the following clarifications: The 'serviceld', 'version' and 'deviceld' MAY be specified using a "*" meaning that the filter applies to several services and devices respectively. Example: "?presenceFilter=person/mood&presenceFilter=service/*/*/icon"
anonymous	empty	Yes	Allows the Watcher to request that its user identity is not revealed to the Presentity. Example: ?anonymous

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.15.3.1 Example: retrieving presence information for all Presentities in a Presence List (Informative)

6.15.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/presenceLists/myFriends HTTP/1.1
Host: example.com
Accept: application/xml
```

6.15.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceList xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceContact>
    <presentityUserId>tel:+19585550100</presentityUserId>
    <resourceStatus>Active</resourceStatus>
    <presence>
      <person>
        <mood>
          <moodValue>Happy</moodValue>
        </mood>
        <noteList>
          <note xml:lang="en">Im on vacation!</note>
        </noteList>
      </person>
      <service>
        <serviceld>org.openmobilealliance:IM-Session</serviceld>
        <version>1.0</version>
        <serviceAvailability>Open</serviceAvailability>
        <devices>
          <deviceld>mac:321</deviceld>
        </devices>
      </service>
    </presence>
  </presenceContact>
</pr:presenceList>
```

```

    </devices>
  </service>
  <device>
    <deviceId>mac:321</deviceId>
    <networkAvailability>
      <network id="GPRS">
        <connectionStatus>Active</connectionStatus>
      </network>
    </networkAvailability>
  </device>
</presence>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100</resourceURL>
</presenceContact>
<presenceContact>
  <presentityUserId>tel:+19585550102</presentityUserId>
  <resourceStatus>Pending</resourceStatus>
</presenceContact>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550102</resourceURL>
</presenceContact>
<presenceContact>
  <presentityUserId>tel:+19585550104</presentityUserId>
  <resourceStatus>TerminatedNoResource</resourceStatus>
</presenceContact>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550104</resourceURL>
</presenceContact>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceLists/myFriends</resourceURL>
</pr:presenceList>

```

6.15.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.15.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.15.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.16 Resource: Content by Watcher

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/PresenceContactsContent/{presentityUserId}/{contentId}

This resource is used by a Watcher to retrieve content from Presentity.

The Watcher is only allowed to retrieve it if has been authorized by the Presentity.

6.16.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Watcher retrieving the content data from the Presentity. Examples: tel:+19585550101, acr:pseudonym124
presentityUserId	Identity of the Presentity owning the content data. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.16.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.16.3 GET

This operation is used by a Watcher to retrieve content, such as a picture/icon from Presentity.

6.16.3.1 Example: retrieving content by Watcher (Informative)

6.16.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContactsContent/tel%3A%2B19585550100/pic001.jpg HTTP/1.1
Host: example.com
Accept: image/jpeg
```

6.16.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: image/jpeg
Content-Length: nnnn
```

data

6.16.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.16.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.16.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.17 Resource: All subscriptions

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions

This resource is used to retrieve all subscriptions that the user has created. It includes all active presence, Presence List- and watchers subscriptions.

6.17.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the user retrieving the subscriptions. Examples: tel:+19585550101, acr:pseudonym124

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.17.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.17.3 GET

This operation is used by a user to retrieve all active subscriptions.

6.17.3.1 Example: retrieving all active subscriptions for user (Informative)

6.17.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions HTTP/1.1
Host: example.com
Accept: application/xml
```

6.17.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn
```

```

<?xml version="1.0" encoding="UTF-8"?>
<pr:subscriptionList xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceSubscriptionList>
    <presenceSubscription>
      <presentityUserId>tel:+19585550100</presentityUserId>
      <callbackReference>
        <notifyURL>http://application.example.com/notifications/presenceNotification</notifyURL>
        <callbackData>1234</callbackData>
      </callbackReference>
      <clientCorrelator>321</clientCorrelator>
      <applicationTag>myApp</applicationTag>
      <duration>5246</duration>
      <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/su
        </resourceURL>
      </presenceSubscription>
      <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions</resourceURL>
    </presenceSubscriptionList>
    <presenceListSubscriptionCollection>
      <presenceListSubscription>
        <presenceListId>myFriends</presenceListId>
        <callbackReference>
          <notifyURL>http://application.example.com/notifications/presenceListNotification</notifyURL>
          <callbackData>2345</callbackData>
        </callbackReference>
        <clientCorrelator>432</clientCorrelator>
        <applicationTag>myApp</applicationTag>
        <duration>4629</duration>
        <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/
          sub002</resourceURL>
        </presenceListSubscription>
        <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions</resourceURL>
      </presenceListSubscriptionCollection>
      <watchersSubscriptionList>
        <watchersSubscription>
          <presentityUserId>tel:+19585550100</presentityUserId>
          <callbackReference>
            <notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
            <callbackData>3456</callbackData>
          </callbackReference>
          <clientCorrelator>543</clientCorrelator>
          <applicationTag>myApp</applicationTag>
          <duration>2413</duration>
          <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/watchersSubscriptions/sub003
            </resourceURL>
          </watchersSubscription>
          <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/watchersSubscriptions</resourceURL>
        </watchersSubscriptionList>
      <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions</resourceURL>
    </pr:subscriptionList>

```

6.17.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.17.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.17.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.18 Resource: All Watchers subscriptions

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/watchersSubscriptions

This resource is used by a Presentity to manage Watchers subscriptions, i.e. subscriptions for changes in the Watcher list. The list contains Watchers that are subscribing for presence information about the Presentity.

For instance, a notification will be generated when there is a new Watcher for Presentity and the authorization decision evaluates to 'Confirm'.

This resource can be used in conjunction with a Client-side Notification URL, or in conjunction with a Server-side Notification URL. In this latter case, the application MUST first create a Notification Channel (see [REST_NetAPI_NotificationChannel]) before creating a subscription.

6.18.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity retrieving the subscriptions. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.18.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.18.3 GET

This operation is used by a Presentity to retrieve all its active Watchers subscriptions.

6.18.3.1 Example: retrieving all Watchers subscriptions (Informative)

6.18.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions HTTP/1.1
Host: example.com
Accept: application/xml
```

6.18.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersSubscriptionList xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <watchersSubscription>
    <presentityUserId>tel:+19585550100</presentityUserId>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
      <callbackData>1234</callbackData>
    </callbackReference>
    <clientCorrelator>321</clientCorrelator>
    <applicationTag>myApp</applicationTag>
    <duration>5246</duration>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/tel%3A%2B19585550100/sub001
  </resourceURL>
</watchersSubscription>
  <watchersSubscription>
    <presentityUserId>tel:+19585550100</presentityUserId>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
      <callbackData>4321</callbackData>
    </callbackReference>
    <clientCorrelator>123</clientCorrelator>
    <applicationTag>myApp</applicationTag>
    <duration>5237</duration>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/tel%3A%2B19585550100/sub002
  </resourceURL>
</watchersSubscription>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions</resourceURL>
</pr:watchersSubscriptionList>
```

6.18.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: GET, POST’ field in the response as per section 14.7 of [RFC2616].

6.18.5 POST

This operation is used by a Presentity to create a new Watchers subscription.

The notifyURL in the callbackReference either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

6.18.5.1 Example: creating new Watchers subscription, using tel URI (Informative)

6.18.5.1.1 Request

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
</pr:watchersSubscription>
```

6.18.5.1.2 Response

```
HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>3600</duration>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001</resourceURL>
</pr:watchersSubscription>
```

In the above example, the Presentity requested the duration of the subscription to be 7200 seconds. However the server policy for that particular server allows a maximum of 3600 seconds for the duration of a subscription, which is reflected in the response.

6.18.5.2 Example: creating new Watchers subscription, using ACR (Informative)

6.18.5.2.1 Request

POST /exampleAPI/presence/v1/acr%3A pseudonym123/subscriptions/watchersSubscriptions HTTP/1.1

Host: example.com

Accept: application/xml

Content-Type: application/xml

Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>acr%3A pseudonym123</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
</pr:watchersSubscription>
```

6.18.5.2.2 Response

HTTP/1.1 201 Created

Location: http://example.com/exampleAPI/presence/v1/acr%3A pseudonym123/subscriptions/watchersSubscriptions/sub001

Date: Thu, 04 Jun 2009 02:51:59 GMT

Content-Type: application/xml

Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>acr%3A pseudonym123</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>3600</duration>
  <resourceURL>http://example.com/exampleAPI/presence/v1/acr%3A pseudonym123/subscriptions/watchersSubscriptions/sub001
  </resourceURL>
</pr:watchersSubscription>
```

In the above example, the Presentity requested the duration of the subscription to be 7200 seconds. However the server policy for that particular server allows a maximum of 3600 seconds for the duration of a subscription, which is reflected in the response.

6.18.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC2616].

6.19 Resource: Individual Watchers subscription

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/watchersSubscriptions/{subscriptionId}

This resource is used by a Presentity to manage an individual Watchers subscription.

This resource can be used in conjunction with a Client-side Notification URL, or in conjunction with a Server-side Notification URL. In this latter case, the application **MUST** first create a Notification Channel (see [REST_NetAPI_NotificationChannel]) before creating a subscription.

6.19.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity managing the subscription. Examples: tel:+19585550100, acr:pseudonym123
subscriptionId	Identifier of the subscription generated by the server.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.19.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.19.3 GET

This operation is used by a Presentity to retrieve an individual Watchers subscription.

6.19.3.1 Example: retrieving individual Watchers subscription (Informative)

6.19.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001 HTTP/1.1
Host: example.com
Accept: application/xml
```

6.19.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
```

```

<pr:watchersSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>5246</duration>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001
  </resourceURL>
</pr:watchersSubscription>

```

6.19.4 PUT

This operation is used by a Presentity to update subscription parameters (e.g. duration, filter, frequency etc) for an ongoing Watchers subscription.

The notifyURL in the callbackReference either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

6.19.4.1 Example: updating individual Watchers subscription (Informative)

6.19.4.1.1 Request

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001 HTTP/1.1
```

```
Host: example.com
```

```
Accept: application/xml
```

```
Content-Type: application/xml
```

```
Content-Length: nnnn
```

```

<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001</resourceURL>
</pr:watchersSubscription>

```

6.19.4.1.2 Response

```
HTTP/1.1 200 OK
```

```
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

```
Content-Type: application/xml
```

Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001
  </resourceURL>
</pr:watchersSubscription>
```

6.19.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT, DELETE' field in the response as per section 14.7 of [RFC2616].

6.19.6 DELETE

This operation is used by a Presentity to terminate an active Watchers subscription.

6.19.6.1 Example: terminating individual Watchers subscription (Informative)

6.19.6.1.1 Request

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001 HTTP/1.1
Host: example.com
```

6.19.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.20 Resource: Watchers notification

The resource URL is provided by the Presentity client when the subscription was created. The RESTful Presence API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel to obtain the notifications, the client needs to use the mechanisms described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.20.5.

A notification SHALL be generated by the server in the following occasions:

Type of notification	Value of ResourceStatus	Generated in the following occasions:
Subsequent notification	Active	An entry in the Watcher list has been updated.

		Note: A request to extend the duration of a subscription does not generate a new notification.
Final notification	TerminatedTimeout	The subscription has expired because it was not refreshed in time. Note: A request to end the subscription does not generate a new notification.
	TerminatedNoResource	The Presentity has been removed from the server.
	TerminatedOther	The subscription has been terminated for an unknown reason.

6.20.1 Request URL variables

Provided by the Presentity client

6.20.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.20.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.20.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.20.5 POST

This operation is used by the server when a new notification is generated.

6.20.5.1 Example 1: notifying Presentity about change in Watchers status (Informative)

6.20.5.1.1 Request

```
POST /notifications/watchersNotification HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersNotification xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackData>1234</callbackData>
  <resourceStatus>Active</resourceStatus>
  <watcherList>
    <watcher>
      <watcherUserId>tel:+19585550101</watcherUserId>
```

```

    <displayName>Bob</displayName>
    <resourceStatus>Pending</resourceStatus>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001</resourceURL>
</watcher>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001</resourceURL>
</watcherList>
  <link rel="WatchersSubscription"
    href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001"/>
</pr:watchersNotification>

```

6.20.5.1.2 Response

```

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

6.20.5.2 Example2: notifying Presentity about subscription time out (Informative)

6.20.5.2.1 Request

```

POST /notifications/watchersNotification HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersNotification xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackData>1234</callbackData>
  <resourceStatus>TerminatedTimeout</resourceStatus>
  <link rel="WatchersSubscription"
    href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001"/>
</pr:watchersNotification>

```

6.20.5.2.2 Response

```

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

6.20.5.3 Example3: notifying Presentity about termination of Watchers subscription (reason unknown) (Informative)

6.20.5.3.1 Request

```

POST /notifications/watchersNotification HTTP/1.1
Host: example.com
Accept: application/xml

```


Content-Type: application/xml
Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersNotification xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackData>1234</callbackData>
  <resourceStatus>TerminatedOther</resourceStatus>
  <link rel="WatchersSubscription"
  href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001"/>
</pr:watchersNotification>
```

6.20.5.3.2 Response

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

6.20.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.21 Resource: All presence subscriptions

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/presenceSubscriptions

This resource is used to retrieve all active presence subscriptions for all Presentities.

6.21.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Watcher retrieving the subscriptions. Examples: tel:+19585550101, acr:pseudonym124

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.21.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.21.3 GET

This operation is used by a Watcher to retrieve all active presence subscriptions for all Presentities.

6.21.3.1 Example: retrieving all presence subscriptions for all Presentities (Informative)

6.21.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions HTTP/1.1
Host: example.com
Accept: application/xml
```

6.21.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSubscriptionList xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceSubscription>
    <presentityUserId>tel:+19585550100</presentityUserId>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/presenceNotification</notifyURL>
      <callbackData>1234</callbackData>
    </callbackReference>
    <clientCorrelator>321</clientCorrelator>
    <applicationTag>myApp</applicationTag>
    <duration>5246</duration>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001
    </resourceURL>
  </presenceSubscription>
  <presenceSubscription>
    <presentityUserId>tel:+19585550100</presentityUserId>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/presenceNotification</notifyURL>
      <callbackData>4321</callbackData>
    </callbackReference>
    <clientCorrelator>123</clientCorrelator>
    <applicationTag>myApp</applicationTag>
    <duration>5237</duration>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550101/sub004
    </resourceURL>
  </presenceSubscription>
</resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions</resourceURL>
</pr:presenceSubscriptionList>
```

6.21.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.21.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.21.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.22 Resource: Presence subscriptions for a single Presentity

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/presenceSubscriptions/{presentityUserId}

This resource is used by a Watcher to manage presence subscriptions.

This resource can be used in conjunction with a Client-side Notification URL, or in conjunction with a Server-side Notification URL. In this latter case, the application MUST first create a Notification Channel (see [REST_NetAPI_NotificationChannel]) before creating a subscription.

6.22.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Watcher managing the subscriptions. Examples: tel:+19585550101, acr:pseudonym124
presentityUserId	Identity of the Presentity owning the presence information. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.22.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.22.3 GET

This operation is used by a Watcher to retrieve all active presence subscriptions for the specified Presentity.

6.22.3.1 Example: retrieving all presence subscriptions for Presentity(Informative)

6.22.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100 HTTP/1.1
Host: example.com
Accept: application/xml
```

6.22.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSubscriptionList xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceSubscription>
    <presentityUserId>tel:+19585550100</presentityUserId>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/presenceNotification</notifyURL>
      <callbackData>1234</callbackData>
    </callbackReference>
    <clientCorrelator>321</clientCorrelator>
    <applicationTag>myApp</applicationTag>
    <duration>5246</duration>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001
  </resourceURL>
</presenceSubscription>
  <presenceSubscription>
    <presentityUserId>tel:+19585550100</presentityUserId>
    <callbackReference>
      <notifyURL>http://application2.example.com/notifications/presenceNotification</notifyURL>
      <callbackData>6789</callbackData>
    </callbackReference>
    <clientCorrelator>987</clientCorrelator>
    <applicationTag>myApp</applicationTag>
    <duration>4132</duration>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub005
  </resourceURL>
</presenceSubscription>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100</resourceURL>
</pr:presenceSubscriptionList>
```

6.22.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC2616].

6.22.5 POST

This operation is used by a Watcher to create a new presence subscription towards the specified Presentity.

The notifyURL in the callbackReference either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

6.22.5.1 Example 1: creating new presence subscription for Presentity(Informative)

6.22.5.1.1 Request

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100 HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <frequency>600</frequency>
</pr:presenceSubscription>
```

6.22.5.1.2 Response

```
HTTP/1.1 201 Created
Location:
http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceSubscriptions/tel%3A%2B19585550100/sub001
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <frequency>600</frequency>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001
  </resourceURL>
</pr:presenceSubscription>
```

6.22.5.2 Example 2: creating new presence subscription for unknown Presentity (Informative)

6.22.5.2.1 Request

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100 HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <frequency>600</frequency>
</pr:presenceSubscription>
```

6.22.5.2.2 Response

```
HTTP/1.1 404 Not Found
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <link rel="PresenceSourceList"
    href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceSources"/>
  <serviceException>
    <messageId>SVC0004</messageId>
    <text>No valid addresses provided in message part %1</text>
    <variables>The specified identity does not exists.</variables>
  </serviceException>
</common:requestError>
```

6.22.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: GET, POST’ field in the response as per section 14.7 of [RFC2616].

6.23 Resource: Individual presence subscription

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/presenceSubscriptions/{presentityUserId}/{subscriptionId}

This resource is used by a Watcher to manage an individual presence subscription.

This resource can be used in conjunction with a Client-side Notification URL, or in conjunction with a Server-side Notification URL. In this latter case, the application MUST first create a Notification Channel (see [REST_NetAPI_NotificationChannel]) before creating a subscription.

6.23.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Watcher managing the subscription. Examples: tel:+19585550101, acr:pseudonym124
presentityUserId	Identity of the Presentity owning the presence information. Examples: tel:+19585550100, acr:pseudonym123
subscriptionId	Identifier of the subscription generated by the server.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.23.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.23.3 GET

This operation is used by a Watcher to retrieve an individual presence subscription.

6.23.3.1 Example: retrieving individual presence subscription (Informative)

6.23.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001
HTTP/1.1
Host: example.com
Accept: application/xml
```

6.23.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
```

```

<callbackReference>
  <notifyURL>http://application.example.com/notifications/presenceNotification</notifyURL>
  <callbackData>1234</callbackData>
</callbackReference>
<clientCorrelator>321</clientCorrelator>
<applicationTag>myApp</applicationTag>
<duration>5246</duration>
<frequency>600</frequency>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001
  </resourceURL>
</pr:presenceSubscription>

```

6.23.4 PUT

This operation is used by a Watcher to update subscription parameters (e.g. duration, filter, frequency etc) for an ongoing presence subscription.

The notifyURL in the callbackReference either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

6.23.4.1 Example: updating individual presence subscription (Informative)

6.23.4.1.1 Request

```

PUT /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001
HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <frequency>600</frequency>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001</resourceURL>
</pr:presenceSubscription>

```

6.23.4.1.2 Response

```
HTTP/1.1 200 OK
```


Date: Thu, 04 Jun 2009 02:51:59 GMT

Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <frequency>600</frequency>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001
  </resourceURL>
</pr:presenceSubscription>
```

6.23.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT, DELETE' field in the response as per section 14.7 of [RFC2616].

6.23.6 DELETE

This operation is used by a Watcher to terminate an active presence subscription.

6.23.6.1 Example: terminating individual presence subscription (Informative)

6.23.6.1.1 Request

```
DELETE
/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001HTTP/1.1
Host: example.com
```

6.23.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.24 Resource: Presence notification

The resource URL is provided by the Watcher client when the subscription was created. The RESTful Presence API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel to obtain the notifications, the client needs to use the mechanisms described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.24.5.

A notification SHALL be generated by the server in the following occasions:

Type of notification	Value of ResourceStatus	Generated in the following occasions:
Subsequent notification	Active	The Presentity's presence information has been updated. A pending subscription has been authorized (allowed) by the Presentity. Note: A request to extend the duration of a subscription does not generate a new notification.
	Pending	The subscription has not yet been authorized. Note: A request to extend the duration of a subscription does not generate a new notification.
Final notification	TerminatedBlocked	The subscription that has been blocked by the Presentity.
	TerminatedTimeout	The subscription has expired because it was not refreshed in time. Note: A request to end the subscription does not generate a new notification.
	TerminatedNoResource	The Presentity or Watcher has been removed from the server.
	TerminatedOther	The subscription has been terminated for an unknown reason.

6.24.1 Request URL variables

Provided by the Watcher client

6.24.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.24.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.24.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.24.5 POST

This operation is used by the server when a new notification is generated.

6.24.5.1 Example 1: notifying Watcher about presence information updates from an active subscription (Informative)

6.24.5.1.1 Request

POST /notifications/presenceNotification HTTP/1.1

Host: example.com

Accept: application/xml

Content-Type: application/xml

Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<pr:presenceNotification xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
```

```
<presentityUserId>tel:+19585550100</presentityUserId>
```

```
<callbackData>1234</callbackData>
```

```
<resourceStatus>Active</resourceStatus>
```

```
<presence>
```

```
<person>
```

```
<mood>
```

```
<moodValue>Happy</moodValue>
```

```
</mood>
```

```
</person>
```

```
</presence>
```

```
<link rel="PresenceSubscription"
```

```
href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001"/>
```

```
</pr:presenceNotification>
```

6.24.5.1.2 Response

HTTP/1.1 204 No Content

Date: Thu, 04 Jun 2009 02:51:59 GMT

6.24.5.2 Example 2: notifying Watcher about presence information updates from pending subscription (Informative)

6.24.5.2.1 Request

POST /notifications/presenceNotification HTTP/1.1

Host: example.com

Accept: application/xml

Content-Type: application/xml

Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<pr:presenceNotification xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
```

```
<presentityUserId>tel:+19585550100</presentityUserId>
```

```
<callbackData>2345</callbackData>
```

```
<resourceStatus>Pending</resourceStatus>
```

```
<link rel="PresenceSubscription"
```

```
href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550101/sub002"/>
```

```
</pr:presenceNotification>
```

6.24.5.2.2 Response

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

6.24.5.3 Example 3: notifying Watcher about termination of presence subscription (reason unknown) (Informative)

6.24.5.3.1 Request

```
POST /notifications/presenceNotification HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceNotification xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackData>1234</callbackData>
  <resourceStatus>TerminatedOther</resourceStatus>
  <link rel="PresenceSubscription"
href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001"/>
</pr:presenceNotification>
```

6.24.5.3.2 Response

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

6.24.5.4 Example 4: notifying Watcher about termination of presence subscription (Watcher blocked) (Informative)

6.24.5.4.1 Request

```
POST /notifications/presenceNotification HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceNotification xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackData>1234</callbackData>
  <resourceStatus>TerminatedBlocked</resourceStatus>
  <link rel="PresenceSubscription"
href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001"/>
</pr:presenceNotification>
```

```
</pr:presenceNotification>
```

6.24.5.4.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.24.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.25 Resource: All Presence List subscriptions

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/presenceListSubscriptions

This resource is used to retrieve all active Presence List subscriptions towards all Presence Lists.

6.25.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Watcher retrieving the subscriptions. Examples: tel:+19585550101, acr:pseudonym124

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.25.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.25.3 GET

This operation is used by a Watcher to retrieve all active Presence List subscriptions towards all Presence Lists.

6.25.3.1 Example: retrieving all Presence List subscriptions towards all Presence Lists (Informative)

6.25.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions HTTP/1.1
Host: example.com
Accept: application/xml
```

6.25.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceListSubscriptionCollection xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceListSubscription>
    <presenceListId>myFriends</presenceListId>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/presenceListNotification</notifyURL>
      <callbackData>1234</callbackData>
    </callbackReference>
    <clientCorrelator>321</clientCorrelator>
    <applicationTag>myApp</applicationTag>
    <duration>5246</duration>
    <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001</resourceURL>
  </presenceListSubscription>
  <presenceListSubscription>
    <presenceListId>myColleagues</presenceListId>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/presenceListNotification</notifyURL>
      <callbackData>4321</callbackData>
    </callbackReference>
    <clientCorrelator>123</clientCorrelator>
    <applicationTag>myApp</applicationTag>
    <duration>5237</duration>
    <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myColleagues/sub002</resourceURL>
  </presenceListSubscription>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions</resourceURL>
</pr:presenceListSubscriptionCollection>
```

6.25.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.25.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.25.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET' field in the response as per section 14.7 of [RFC2616].

6.26 Resource: Presence List subscriptions for a single Presence List

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/presenceListSubscriptions/{presenceListId}

This resource is used by a Watcher to manage Presence List subscriptions.

This resource can be used in conjunction with a Client-side Notification URL, or in conjunction with a Server-side Notification URL. In this latter case, the application MUST first create a Notification Channel (see [REST_NetAPI_NotificationChannel]) before creating a subscription.

6.26.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Watcher managing the subscriptions. Examples: tel:+19585550101, acr:pseudonym124
presenceListId	Identity of the Presence List. Example: myFriends

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.26.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.26.3 GET

This operation is used by a Watcher to retrieve all active Presence List subscriptions for the specified Presence List.

6.26.3.1 Example: retrieving all Presence List subscriptions towards a single Presence List (Informative)

6.26.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends HTTP/1.1
Host: example.com
Accept: application/xml
```

6.26.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceListSubscriptionCollection xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceListSubscription>
    <presenceListId>myFriends</presenceListId>
    <callbackReference>
      <notifyURL>http://application.example.com/notifications/presenceListNotification</notifyURL>
      <callbackData>1234</callbackData>
    </callbackReference>
    <clientCorrelator>321</clientCorrelator>
    <applicationTag>myApp</applicationTag>
    <duration>5246</duration>
    <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001</resourceURL>
  </presenceListSubscription>
</pr:presenceListSubscriptionCollection>
```

6.26.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, POST' field in the response as per section 14.7 of [RFC2616].

6.26.5 POST

This operation is used by a Watcher to create a new Presence List subscription towards the specified Presence List.

The notifyURL in the callbackReference either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

6.26.5.1 Example: creating new Presence List subscription towards a single Presence List (Informative)

6.26.5.1.1 Request

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends HTTP/1.1
```


Host: example.com
 Accept: application/xml
 Content-Type: application/xml
 Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceListSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceListNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <presenceFilter>person/mood</presenceFilter>
  <presenceFilter>service/org.openmobilealliance:IM-Session/1.0</presenceFilter>
  <frequency>600</frequency>
</pr:presenceListSubscription>
```

6.26.5.1.2 Response

HTTP/1.1 201 Created
 Location:
<http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001>
 Date: Thu, 04 Jun 2009 02:51:59 GMT
 Content-Type: application/xml
 Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceListSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceListId>myFriends</presenceListId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceListNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <presenceFilter>person/mood</presenceFilter>
  <presenceFilter>service/org.openmobilealliance:IM-Session/1.0</presenceFilter>
  <frequency>600</frequency>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001</resourceURL>
</pr:presenceListSubscription>
```

6.26.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: GET, POST’ field in the response as per section 14.7 of [RFC2616].

6.27 Resource: Individual Presence List subscription

The resource used is:

http://{serverRoot}/presence/{apiVersion}/{userId}/subscriptions/presenceListSubscriptions/{presenceListId}/{subscriptionId}

This resource is used by a Watcher to manage an individual Presence List subscription.

This resource can be used in conjunction with a Client-side Notification URL, or in conjunction with a Server-side Notification URL. In this latter case, the application MUST first create a Notification Channel (see [REST_NetAPI_NotificationChannel]) before creating a subscription.

6.27.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Watcher managing the subscription. Examples: tel:+19585550101, acr:pseudonym124
presenceListId	Identity of the Presence List. Example: myFriends
subscriptionId	Identifier of the subscription generated by the server.

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.27.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.27.3 GET

This operation is used by a Watcher to retrieve an individual Presence List subscription.

6.27.3.1 Example: retrieving individual Presence List subscription (Informative)

6.27.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001 HTTP/1.1
Host: example.com
Accept: application/xml
```

6.27.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
```

Content-Length: nnnn

Date: Thu, 04 Jun 2009 02:51:59 GMT

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceListSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceListId>myFriends</presenceListId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceListNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>5274</duration>
  <presenceFilter>person/mood</presenceFilter>
  <presenceFilter>service/org.openmobilealliance:IM-Session/1.0</presenceFilter>
  <frequency>600</frequency>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001</resourceURL>
</pr:presenceListSubscription>
```

6.27.4 PUT

This operation is used by a Watcher to update subscription parameters (e.g. duration, filter, frequency etc) for an ongoing Presence List subscription.

The notifyURL in the callbackReference either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

6.27.4.1 Example: updating individual Presence List subscription (Informative)

6.27.4.1.1 Request

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001 HTTP/1.1
```

Host: example.com

Accept: application/xml

Content-Type: application/xml

Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceListSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceListNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <presenceFilter>person/mood</presenceFilter>
  <presenceFilter>service/org.openmobilealliance:IM-Session/1.0</presenceFilter>
  <frequency>600</frequency>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001</resourceURL>
</pr:presenceListSubscription>
```

6.27.4.1.2 Response

```

HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceListSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceListId>myFriends</presenceListId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceListNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <presenceFilter>person/mood</presenceFilter>
  <presenceFilter>service/org.openmobilealliance:IM-Session/1.0</presenceFilter>
  <frequency>600</frequency>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001</resourceURL>
</pr:presenceListSubscription>

```

6.27.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT, DELETE' field in the response as per section 14.7 of [RFC2616].

6.27.6 DELETE

This operation is used by a Watcher to terminate an active Presence List subscription.

6.27.6.1 Example: terminating individual Presence List subscription (Informative)

6.27.6.1.1 Request

```

DELETE /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001 HTTP/1.1
Host: example.com

```

6.27.6.1.2 Response

```

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

6.28 Resource: Presence List notification

The resource URL is provided by the Watcher client when the subscription was created. The RESTful Presence API does not make any assumption about the structure of this URL. If this URL is a Client-side Notification URL, the server will POST notifications directly to it. If this URL is a Server-side Notification URL, the server uses it to determine the address of the

Notification Server to which the notifications will subsequently be POSTed. The way the server determines the address of the Notification Server is out of scope of this specification.

Note: In the case when the client has set up a Notification Channel to obtain the notifications, the client needs to use the mechanisms described in [REST_NetAPI_NotificationChannel], instead of the mechanism described below in section 6.28.5.

A notification SHALL be generated by the server in the following occasions:

Type of notification	Value of ResourceStatus	Generated in the following occasions:
Subsequent notification	Active	An entry in the Presence List has been updated. Note: A request to extend the duration of a subscription does not generate a new notification.
Final notification	TerminatedTimeout	The subscription has expired because it was not refreshed in time. Note: A request to end the subscription does not generate a new notification.
	TerminatedNoResource	The Watcher has been removed from the server.
	TerminatedOther	The subscription has been terminated for an unknown reason.

6.28.1 Request URL variables

Provided by the Watcher client

6.28.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.28.3 GET

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.28.4 PUT

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: POST' field in the response as per section 14.7 of [RFC2616].

6.28.5 POST

This operation is used by the server when a new notification is generated.

6.28.5.1 Example 1: notifying Watcher about presence information updates relating to Presence List (Informative)

6.28.5.1.1 Request

```
POST /notifications/presenceListNotification HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
```

Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceListNotification xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceListId>myFriends</presenceListId>
  <callbackData>1234</callbackData>
  <resourceStatus>Active</resourceStatus>
  <presenceList>
    <presenceContact>
      <presentityUserId>tel:+19585550101</presentityUserId>
      <resourceStatus>Active</resourceStatus>
      <presence>
        <person>
          <mood>
            <moodValue>Happy</moodValue>
          </mood>
        </person>
      </presence>
    </presenceContact>
    <presenceContact>
      <presentityUserId>tel:+19585550102</presentityUserId>
      <resourceStatus>Pending</resourceStatus>
    </presenceContact>
  </presenceList>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/presenceListSubscription/myFriends/sub001</resourceURL>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/presenceListSubscription/myFriends/sub001</resourceURL>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/presenceListSubscription/myFriends/sub001</resourceURL>
  <link rel="PresenceListSubscription" href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/presenceListSubscriptions/myFriends/sub001" />
</pr:presenceListNotification>
```

6.28.5.1.2 Response

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

6.28.5.2 Example 2: notifying Watcher about termination of Presence List subscription (No resource) (Informative)

6.28.5.2.1 Request

```
POST /notifications/presenceListNotification HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn
```

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<pr:presenceListNotification xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceListId>myFriends</presenceListId>
  <callbackData>1234</callbackData>
  <resourceStatus>TerminatedNoResource</resourceStatus>
</pr:presenceListNotification>
```

6.28.5.2.2 Response

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

6.28.5.3 Example 3: notifying Watcher about termination of presence subscription (reason unknown) (Informative)

6.28.5.3.1 Request

```
POST /notifications/presenceListNotification HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceListNotification xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceListId>myFriends</presenceListId>
  <callbackData>1234</callbackData>
  <resourceStatus>TerminatedOther</resourceStatus>
  <link rel="PresenceListSubscription"
href="http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001"/>
</pr:presenceListNotification>
```

6.28.5.3.2 Response

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

6.28.5.4 Example 4: notifying Watcher about subscription time out (Informative)

6.28.5.4.1 Request

```
POST /notifications/presenceListNotification HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceListNotification xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceListId>myFriends</presenceListId>
```

```
<callbackData>1234</callbackData>
<resourceStatus>TerminatedTimeout</resourceStatus>
<link rel="PresenceListSubscription"
  href="http://example.com/exampleAPI/presence/v1/{userId}/subscriptions/presenceListSubscriptions/myFriends/sub001"/>
</pr:presenceListNotification>
```

6.28.5.4.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.28.6 DELETE

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the ‘Allow: POST’ field in the response as per section 14.7 of [RFC2616].

6.29 Resource: Presentity portrait icon

The resource used is:

`http://{serverRoot}/presence/{apiVersion}/{userId}/content/portraitIcon`

This resource is used by Presentity to retrieve, upload, and remove its own portrait icon. Simultaneously with upload/update of the icon, the server SHALL set/update the link to the icon as presence information in order to distribute the link to the Watcher of the Presentity’s presence information. The uploaded portrait icon is related to the Presentity by using the “person/status-icon” attribute where the link to the icon is stored.

6.29.1 Request URL variables

The following request URL variables are common for all HTTP commands:

Name	Description
serverRoot	Server base url: hostname+port+base path. Port and base path are OPTIONAL. Example: example.com/exampleAPI
apiVersion	Version of the API client wants to use. The value of this variable is defined in section 5.1.
userId	Identity of the Presentity that the content is managed for. Examples: tel:+19585550100, acr:pseudonym123

See section 6 for a statement on the escaping of reserved characters in URL variables.

6.29.2 Response Codes and Error Handling

For HTTP response codes, see [REST_NetAPI_Common].

For Policy Exception and Service Exception fault codes applicable to Presence, see section 7.

6.29.3 GET

This operation is used by the Presentity to retrieve its own portrait icon.

6.29.3.1 Example: retrieving portrait icon by Presentity (Informative)

6.29.3.1.1 Request

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/content/portraitIcon HTTP/1.1
Host: example.com
Accept: image/jpeg
```

6.29.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: image/jpeg
Content-Length: nnnn
```

data

6.29.4 PUT

This operation is used by Presentity to upload a new or modify the existing own portrait icon on the server. Simultaneously the server SHALL set/update the link to the icon as presence information.

6.29.4.1 Example: uploading/updating of portrait icon and setting the link to the icon as presence information (Informative)

6.29.4.1.1 Request

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/content/portraitIcon HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: image/jpeg
Content-Length: nnnn
```

data

6.29.4.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

6.29.5 POST

Method not allowed by the resource. The returned HTTP error status is 405. The server should also include the 'Allow: GET, PUT, DELETE' field in the response as per section 14.7 of [RFC2616].

6.29.6 DELETE

This operation is used to by Presentity to remove its own portrait icon.

6.29.6.1 Example: removing portrait icon by Presentity (Informative)

6.29.6.1.1 Request

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/content/portraitIcon HTTP/1.1
Host: example.com
```

6.29.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

7. Fault definitions

7.1 Service Exceptions

For common Service Exceptions refer to [REST_NetAPI_Common]. The following additional Service Exception codes are defined for the RESTful Presence API.

7.1.1 SVC0220: No subscription request

Name	Description
MessageID	SVC0220
Text	No subscription request from Watcher %1 for attribute %2
Variables	%1 - Watcher URI %2 - type of attribute, The type of attribute is indicated with relative path according to the [ResourceRelPath] in sections 5.2.2.3, 5.2.2.4, 5.2.2.5, and 5.2.2.6.
HTTP status code(s)	403 Forbidden

7.1.2 SVC0221: Not a Watcher

Name	Description
MessageID	SVC0221
Text	%1 is not a Watcher
Variables	%1 – Watcher URI
HTTP status code(s)	403 Forbidden

7.1.3 SVC0222: Key property changes not allowed

Name	Description
MessageID	SVC0222
Text	Key property changes not allowed: key property %1
Variables	%1 – key property
HTTP status code(s)	403 Forbidden

7.1.4 SVC1001: Presence source does not exist

Name	Description
MessageID	SVC1001
Text	Presence source does not exist.

Variables	None
HTTP status code(s)	404 Not found

7.2 Policy Exceptions

For common Policy Exceptions refer to [REST_NetAPI_Common]. The following additional Policy Exception codes are defined for the RESTful Presence API.

7.2.1 POL0260: Maximum number of presence sources exceeded

Name	Description
MessageID	POL0260
Text	Maximum number of presence sources exceeded.
Variables	None
HTTP status code(s)	403 Forbidden

Appendix A. Change History (Informative)

A.1 Approved Version History

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

A.2 Draft/Candidate Version 1.0 History

Document Identifier	Date	Sections	Description
Draft Versions OMA-TS-REST_NetAPI_Presence-V1_0	02 May 2011	Many	Structural changes to fit the OMA RESTful Network API release. This version inherits the technical content of OMA-TS-ParlayREST_Presence-V1_0-20110111-C and applies changes according to ARC INP 30R01, 98R02, 155R01,156R01, 0186, 0187, and 165R02.
	10 May 2011	Many	Implemented CR: OMA--ARC-REST-NetAPI-2011-0010-CR_TS_changes_for_Presence
	15 Jun 2011	Many	Implemented CRs: OMA-ARC-REST-NetAPI-2011-0061-CR_Fixing_Sphere_AttributeValue_for_Presence_TS, OMA-ARC-REST-NetAPI-2011-0063-CR_Bug_fixes_for_Presence, OMA-ARC-REST-NetAPI-2011-0064-CR_Generation_of_notifications_for_Presence, OMA-ARC-REST-NetAPI-2011-0065R01-CR_Presence_TS_with_new_template,
	15 Sep 2011	Many	Implemented CRs: OMA-ARC-REST-NetAPI-2011-0165-CR_LW_resource_keys_as_elements_Presence_NetAPI_mirror_of_RES T_M_0015 OMA-ARC-REST-NetAPI-2011-0207-CR_Presence_telURI_changes OMA-ARC-REST-NetAPI-2011-0208R02-CR_Presence_Notif_Channel_changes OMA-ARC-REST-NetAPI-2011-0209-CR_Presence_Appendix_C_link_to_section_6 OMA-ARC-REST-NetAPI-2011-0211-CR_Presence_resourceURL_changes Fixed validation errors in XML and JSON
	16 Nov 2011	Many	Implemented CR: OMA--ARC-REST-NetAPI-2011-0263-CR_Presence_TS_ACR_changes
	23 Nov 2011	Many	Implemented CR: OMA-ARC-REST-NetAPI-2011-0389R01-CR_Presence_TS_api_version_changes, OMA-ARC-REST-NetAPI-2011-0390R01-CR_Presence_TS_resourceURL_and_other_bugfixes, OMA-ARC-REST-NetAPI-2011-0391R02-CR_Presence_TS_support_for_portrait_icon_management, OMA-ARC-REST-NetAPI-2011-0400R01-CR_Presence_TS_updates_for_Appendix_G Editorial changes for headings in D.71 and Appendix F.
	02 Feb 2012	Many	Implemented CR: OMA-ARC-REST-NetAPI-2012-0040-CR_Presence_CONRR_editorial_comments_changes
	13 Feb 2012	Many	Implemented CR: OMA-ARC-REST-NetAPI-2012-0048-CR_Presence_TS_CONR_editorial_comments_changes_2
	15 Feb 2012	Many	Implemented CR: OMA-ARC-REST-NetAPI-2012-0058-

Document Identifier	Date	Sections	Description
			CR_Presence_TS_CONR_comments_section_5_1
	16 Feb 2012	Many	Implemented CR: OMA-ARC-REST-NetAPI-2012-0062- CR_Presence_TS_CONR_comments_resolution_3
	20 Feb 2012	Many	Implemented CRs: OMA-ARC-REST-NetAPI-2012-0066-CR_Presence_TS_CONR_comments_resolution_4, OMA-ARC-REST-NetAPI-2012-0068-CR_Presence_TS_SCR_subscriptions_optional_to_mandatory_changes
	22 Feb 2012	Many	Implemented CRs: OMA-ARC-REST-NetAPI-2012-0064-CR_Presence_TS_Adding_Service_Policy_Exceptions, OMA-ARC-REST-NetAPI-2012-0065-CR_Presence_TS_adding_new_definitions. Small editorial changes to align letter case for definitions through the document.
Candidate Version OMA-TS-REST_NetAPI_Presence-V1_0	13 Mar 2012	n/a	Status changed to Candidate by TP TP Ref # OMA-TP-2012-0108- INP_REST_NetAPI_Presence_1_0_ERP_and_ETR_for_Candidate_Approval
Draft Versions OMA-TS-REST_NetAPI_Presence-V1_0	15 June 2012	5.3	Incorporated CR: OMA-ARC-REST-NetAPI-2012-0164- CR_Presence_POST_or_NOTIFY_changes Updated template. Editorial changes
	23 Jul 2012	3.2, 5, 5.2.2, 5.2.2.2, 5.2.2.3, 5.2.2.5, 5.2.2.6, 5.2.2.12, 5.2.2.18, 5.2.2.21, 5.2.2.24, 5.2.2.44, 5.2.3.2, 5.2.3.14, 5.3.2, 5.3.3, 6, 6.1.3, 6.1.3.1.1, 6.1.3.2.1, 6.1.5.1.1, 6.1.5.2.1, 6.2.3.1.1, 6.2.3.2.1, 6.2.4.1.1, 6.2.6.1.1, 6.3, 6.3.1, 6.3.1.1, 6.3.3.1.1, 6.3.4.1.1, 6.3.6.1.1, 6.4, 6.4.3.1.1, 6.4.4.1.1, 6.4.4.1.3, 6.4.6.1.1, 6.5, 6.5.1, 6.5.1.1, 6.5.3.1.1, 6.5.4.1.1, 6.5.6.1.1, 6.6.3.1.1, 6.7.3.1.1, 6.7.4.1.1, 6.7.6, 6.7.6.1.1, 6.7.6.1.2, 6.8.3.1.1, 6.9.3.1.1,	Incorporated CRs: OMA-ARC-REST-NetAPI-2012-0174R01- CR_Presence_TS_callbackData_and_other_bug_fixes OMA-ARC-REST-NetAPI-2012-0205- CR_Presence_TS_missing_resourceURL_for_PUT Editorial changes

Document Identifier	Date	Sections	Description
		6.10.3.1.1, 6.10.5.1.1, 6.10.5.2.1, 6.11.1, 6.11.3.1.1, 6.11.4.1.1, 6.11.6.1.1, 6.12, 6.12.1, 6.12.1.1, 6.12.3.1.1, 6.12.4.1.1, 6.12.6.1.1, 6.13.3.1.1, 6.13.3.2.1, 6.14, 6.14.1, 6.14.1.1, 6.14.3.1.1, 6.15.3.1.1, 6.16.3.1.1, 6.17.3.1.1, 6.18.3.1.1, 6.18.5.1.1, 6.18.5.1.2, 6.18.5.2.1, 6.19.1, 6.19.3.1.1, 6.19.4.1.1, 6.19.6.1.1, 6.20, 6.20.5, 6.20.5.1.1, 6.20.5.2.1, 6.20.5.3.1, 6.21.3.1.1, 6.22.3.1.1, 6.22.5.1.1, 6.22.5.2.1, 6.23.1, 6.23.3.1.1, 6.23.4.1.1, 6.23.6.1.1, 6.24, 6.24.5, 6.24.5.1.1, 6.24.5.2.1, 6.24.5.3.1, 6.24.5.4.1, 6.25.3.1.1, 6.26.3.1.1, 6.26.5.1.1, 6.27.1, 6.27.3.1.1, 6.27.4.1.1, 6.27.6.1.1, 6.28, 6.28.5, 6.28.5.1.1, 6.28.5.2.1, 6.28.5.3.1, 6.28.5.4.1, 6.29.3.1.1, 6.29.4.1.1, 6.29.6.1.1, B.1.4, B.1.8, B.1.20, C.1, C.1.1.1, C.2.1.1, C.3.2.1, C.4.1.1, C.5.1.1, D, F	
	06 Aug 2012	6.1.5.2.2, 6.2.3.2.2, 7.1.4, 7.2.1,	Incorporated CRs: OMA-ARC-REST-NetAPI-2012-0214-

Document Identifier	Date	Sections	Description
		D.4, D.6	CR_Followup_for_INP_200_TS_Presence Editorial changes
	24 Aug 2012	5.2.2.2, 5.2.2.17, 5.2.2.20, 5.2.2.23, C.1, C.3, C.4, C.5	Incorporated CR: OMA-ARC-REST-NetAPI-2012-0231- CR_Presence_TS_issue_20_clientCorrelator_resolution Editorial changes
	13 Dec 2012	3.2, 4.1, 5.2.2.4, 5.3.1, 5.3.2, 6, 6.1.1, 6.1.3, 6.2.1, 6.3.1, 6.3.1.1, 6.4.1, 6.5.1, 6.5.1.1, 6.6.1, 6.7.1, 6.8.1, 6.8.3, 6.9.1, 6.10.1, 6.11.1, 6.12.1, 6.12.1.1, 6.13.1, 6.13.3, 6.13.6, 6.14.1, 6.14.1.1, 6.14.3, 6.15.1, 6.15.3, 6.16.1, 6.17.1, 6.18.1, 6.19.1, 6.20, 6.21.1, 6.22.1, 6.23.1, 6.24, 6.25.1, 6.26.1, 6.27.1, 6.28, 6.29.1, 7.1.4, B, C.1, C.2, C.3, C.4, C.5, E, G.1.1.1, G.1.2	Incorporated CR: OMA-ARC-REST-NetAPI-2012-0290- CR_Presence_TS_blueprint_for_longpolling_and_authorization Template changed to OMA-TEMPLATE- TS_RESTful_Network_API-20120813-I Editorial changes
	08 Feb 2013	2.2	Reference to OMA Dictionary updated to version 2.9 Template updated
Candidate Version OMA-TS-REST_NetAPI_Presence-V1_0	12 Feb 2013	n/a	Status changed to Candidate by TP TP Ref # OMA-TP-2013-0048- INP_REST_NetAPI_Presence_V1_0_Candidate_ERP_for_Notificati on

Appendix B. Static Conformance Requirements (Normative)

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

B.1 SCR for REST.Presence Server

Item	Function	Reference	Requirement
REST-PRESENCE-SUPPORT-S-001-M	Support for the RESTful Presence API	5, 6	
REST-PRESENCE-SUPPORT-S-002-M	Support for the XML request & response format	6	
REST-PRESENCE-SUPPORT-S-003-M	Support for the JSON request & response format	6	
REST-PRESENCE-SUPPORT-S-004-O	Support for the application/x-www-form-urlencoded format	Appendix C	

B.1.1 SCR for REST.Presence.Presentity.PresenceSource Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-PS-S-001-M	Support for creation and retrieval of Presence Source data	6.1	
REST-PRESENCE-PRES-PS-S-002-M	Retrieve all Presence Sources related to Presentity - GET	6.1.3	
REST-PRESENCE-PRES-PS-S-003-M	Create Presence Source data – POST (XMLor JSON)	6.1.5	
REST-PRESENCE-PRES-PS-S-004-O	Create Presence Source data – POST (application/x-www-form-urlencoded)	C.1	

B.1.2 SCR for REST.Presence.Presentity.Individual.PresenceSource Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-IND-PS-S-001-M	Support for the management of presence information for a Presentity from an individual Presence Source	6.2	
REST-PRESENCE-PRES-IND-PS-S-002-M	Retrieve presence information for a Presentity - GET	6.2.3	
REST-PRESENCE-PRES-IND-PS-S-	Update presence	6.2.4	

Item	Function	Reference	Requirement
003-M	information for a Presentity - PUT		
REST-PRESENCE-PRES-IND-PS-S-004-M	Delete presence information for a Presentity- DELETE	6.2.6	

B.1.3 SCR for REST.Presence.Presentity.Individual.PresenceSource.Attribute Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-IND-PS-ATTR-S-001-O	Support for the management of an individual presence attribute	6.3	REST-PRESENCE-PRES-IND-PS-ATTR-S-003-O REST-PRESENCE-PRES-IND-PS-ATTR-S-004-O
REST-PRESENCE-PRES-IND-PS-ATTR-S-002-O	Retrieve value of an individual presence attribute - GET	6.3.3	
REST-PRESENCE-PRES-IND-PS-ATTR-S-003-O	Update value of an individual presence attribute - PUT	6.3.4	
REST-PRESENCE-PRES-IND-PS-ATTR-S-004-O	Delete value of an individual presence attribute - DELETE	6.3.6	

B.1.4 SCR for REST.Presence.Presentity.PresenceSource.Persistent Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-PS-PERS-S-001-M	Support for the management of persistent presence information for a Presentity	6.4	
REST-PRESENCE-PRES-PS-PERS-S-002-M	Retrieve persistent presence information - GET	6.4.3	
REST-PRESENCE-PRES-PS-PERS-S-003-M	Update persistent presence information - PUT	6.4.4	
REST-PRESENCE-PRES-PS-PERS-S-004-M	Delete persistent presence information - DELETE	6.4.6	

B.1.5 SCR for REST.Presence.Presentity.PresenceSource.Persistent.Attribute Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-PS-PERS-ATTR-S-001-O	Support for the management of an individual persistent presence attribute	6.5	REST-PRESENCE-PRES-PS-PERS-ATTR-S-003-O REST-PRESENCE-PRES-PS-PERS-ATTR-S-004-O
REST-PRESENCE-PRES-PS-PERS-ATTR-S-002-O	Retrieve individual persistent presence attribute - GET	6.5.3	
REST-PRESENCE-PRES-PS-PERS-ATTR-S-003-O	Update individual persistent presence attribute - PUT	6.5.4	
REST-PRESENCE-PRES-PS-PERS-ATTR-S-004-O	Delete individual persistent presence attribute - DELETE	6.5.6	

B.1.6 SCR for REST.Presence.Presentity.ContentList Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-CONTL-S-001-M	Support for a read access to a content list	6.6	
REST-PRESENCE-PRES-CONTL-S-002-M	Retrieve a content list -GET	6.6.3	

B.1.7 SCR for REST.Presence.Presentity.Individual.Content Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-IND-CONT-S-001-M	Support for the management of an individual content by Presentity	6.7	
REST-PRESENCE-PRES-IND-CONT-S-002-M	Retrieve individual content - GET	6.7.3	
REST-PRESENCE-PRES-IND-CONT-S-003-M	Create/replace individual content - PUT	6.7.4	
REST-PRESENCE-PRES-IND-CONT-S-004-M	Delete (remove) individual content - DELETE	6.7.6	

B.1.8 SCR for REST.Presence.Presentity.WatcherList Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-WL-S-001-M	Support for a read access to Watcher list	6.8	
REST-PRESENCE-PRES-WL-S-002-M	Retrieve Watcher list -	6.8.33	

Item	Function	Reference	Requirement
	GET		

B.1.9 SCR for REST.Presence.Presentity.Individual.Watcher Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-IND-WATCHER-S-001-M	Support for a read access to an individual Watcher information	6.9	
REST-PRESENCE-PRES-IND-WATCHER-S-002-M	Retrieve individual Watcher information - GET	6.9.3	

B.1.10 SCR for REST.Presence.Presentity.Authorization.Rules Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-AUTH-RULES-S-001-M	Support for creation and read access for authorization rules	6.10	
REST-PRESENCE-PRES-AUTH-RULES-S-002-M	Retrieve list of authorization - GET	6.10.3	
REST-PRESENCE-PRES-AUTH-RULES-S-003-M	Create a new authorization rule – POST (XML or JSON)	6.10.5	
REST-PRESENCE-PRES-AUTH-RULES-S-004-O	Create a new authorization rule – POST (application/x-www-form-urlencoded)	C.2	

B.1.11 SCR for REST.Presence.Presentity.Individual.Authorization.Rule Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-IND-AUTH-RULE-S-001-M	Support for the management of an individual authorization rule	6.11	
REST-PRESENCE-PRES-IND-AUTH-RULE-S-002-M	Retrieve an individual authorization rule - GET	6.11.3	
REST-PRESENCE-PRES-IND-AUTH-RULE-S-003-M	Update an individual authorization rule - PUT	6.11.4	
REST-PRESENCE-PRES-IND-AUTH-RULE-S-004-M	Delete an individual authorization rule - DELETE	6.11.6	

B.1.12 SCR for REST.Presence.Presentity.Individual.Authorization.Rule.Data Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-IND-AUTH-RULE-DATA-S-001-O	Support for the management of individual data from an authorization rule	6.12	REST-PRESENCE-PRES-IND-AUTH-RULE-DATA-S-003-O REST-PRESENCE-PRES-IND-AUTH-RULE-DATA -S-004-O
REST-PRESENCE-PRES-IND-AUTH-RULE-DATA-S-002-O	Retrieve individual data from an authorization rule - GET	6.12.3	
REST-PRESENCE-PRES-IND-AUTH-RULE-DATA-S-003-O	Create/Update individual data for an authorization rule - PUT	6.12.4	
REST-PRESENCE-PRES-IND-AUTH-RULE-DATA -S-004-O	Delete individual data for an authorization rule - DELETE	6.12.6	

B.1.13 SCR for REST.Presence.Watcher.PresenceContact Server

Item	Function	Reference	Requirement
REST-PRESENCE-WATCH-PC-S-001-M	Support for a read access for a composite presence information for a single Presence contact (Presentity)	6.13	
REST-PRESENCE-WATCH-PC-S-002-M	Retrieve composite presence information for a Presentity - GET	6.13.3	

B.1.14 SCR for REST.Presence.Watcher.Individual.PresenceContact.Attribute Server

Item	Function	Reference	Requirement
REST-PRESENCE-WATCH-IND-PC-ATTR-S-001-O	Support for a read access for an individual presence attribute for a Presentity	6.14	REST-PRESENCE-WATCH-IND-PC-ATTR-S-002-O
REST-PRESENCE-WATCH-IND-PC-ATTR-S-002-O	Retrieve individual presence attribute for a Presentity - GET	6.14.3	

B.1.15 SCR for REST.Presence.Watcher.PresenceList Server

Item	Function	Reference	Requirement
REST-PRESENCE-WATCH-PL-S-001-M	Support for a read access to presence information for Presence List	6.15	
REST-PRESENCE-WATCH-PL-S-002-M	Retrieve presence information for all users (Presentities) in a Presence List - GET	6.15.3	

B.1.16 SCR for REST.Presence.Watcher.PresenceContactContent Server

Item	Function	Reference	Requirement
REST-PRESENCE-WATCH-PCC-S-001-M	Support for a read access for a content	6.16	
REST-PRESENCE-WATCH-PCC-S-002-M	Retrieve a content from a Presentity - GET	6.16.3	

B.1.17 SCR for REST.Presence.Subscriptions Server

Item	Function	Reference	Requirement
REST-PRESENCE-SUBSCR-S-001-O	Support for a read access for subscriptions for a particular user (could be either in Presentity or Watcher role)	6.17	REST-PRESENCE-SUBSCR-S-002-O
REST-PRESENCE-SUBSCR-S-002-O	Read all active subscriptions - GET	6.17.3	

B.1.18 SCR for REST.Presence.Presentity.Subscriptions.WatchersSubscriptions Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-SUBSCR-WS-S-001-M	Support for subscriptions for notifications about Watchers information	6.18	
REST-PRESENCE-PRES-SUBSCR-WS-S-002-O	Read all active subscriptions - GET	6.18.3	
REST-PRESENCE-PRES-SUBSCR-WS-S-003-M	Create subscription for a Watchers list – POST (XML or JSON)	6.18.5	
REST-PRESENCE-PRES-SUBSCR-WS-S-004-O	Create subscription for a Watchers list – POST (application/x-www-form-urlencoded)	C.3	

B.1.19 SCR for REST.Presence.Presentity.Individual.Subscriptions.WatchersSubscriptions Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-IND-SUBSCR-WS-S-001-M	Support for individual subscription for notifications about Watchers information	6.19	
REST-PRESENCE-PRES-IND-SUBSCR-WS-S-002-O	Read individual subscription - GET	6.19.3	
REST-PRESENCE-PRES-IND-SUBSCR-WS-S-003-M	Update subscription for Watcher Information - PUT	6.19.4	
REST-PRESENCE-PRES-IND-SUBSCR-WS-S-004-M	Delete (terminate) subscription - DELETE	6.19.6	

B.1.20 SCR for REST.Presence.WatchersSubscriptions.Notifications Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-WS-NOTIF-S-001-M	Support for notifying application (Presentity) about changes in Watcher's subscription status	6.20	
REST-PRESENCE-PRES-WS-NOTIF-S-002-M	Notify application about changes in Watcher's subscription status - POST	6.20.5	

B.1.21 SCR for REST.Presence.Watcher.Subscriptions.PresenceSubscriptions Server

Item	Function	Reference	Requirement
REST-PRESENCE-WATCH-SUBSCR-PS-S-001-O	Support for read access for presence subscriptions	6.21	REST-PRESENCE-WATCH-SUBSCR-PS-S-002-O
REST-PRESENCE-WATCH-SUBSCR-PS-S-002-O	Read all active presence subscriptions - GET	6.21.3	

B.1.22 SCR for REST.Presence.Watcher.Subscriptions.PresenceSubscriptions.SinglePresentity Server

Item	Function	Reference	Requirement
REST-PRESENCE-WATCH-SUBSCR-PS-SINGP-S-001-M	Support for subscriptions for notifications about	6.22	

Item	Function	Reference	Requirement
	presence information for a particular Presentity		
REST-PRESENCE-WATCH-SUBSCR-PS-SINGP-S-002-O	Read all active presence subscriptions for a particular Presentity - GET	6.22.3	
REST-PRESENCE-WATCH-SUBSCR-PS-SINGP-S-003-M	Create subscription for presence information for a particular Presentity – POST (XML or JSON)	6.22.5	
REST-PRESENCE-WATCH-SUBSCR-PS-SINGP-S-004-O	Create subscription for presence information for a particular Presentity – POST (application/x-www-form-urlencoded)	C.4	

B.1.23 SCR for REST.Presence.Watcher.Individual.Subscriptions.PresenceSubscriptions.SinglePresentity Server

Item	Function	Reference	Requirement
REST-PRESENCE-WATCH-IND-SUBSCR-PS-SINGP-S-001-M	Support for individual subscription for notifications about presence information for a particular Presentity	6.23	
REST-PRESENCE-WATCH-IND-SUBSCR-PS-SINGP-S-002-O	Read an individual presence subscription for a particular Presentity - GET	6.23.3	
REST-PRESENCE-WATCH-IND-SUBSCR-PS-SINGP-S-003-M	Update subscription for presence information for a particular Presentity - PUT	6.23.4	
REST-PRESENCE-WATCH-IND-SUBSCR-PS-SINGP-S-004-M	Delete (terminate) subscription for presence information for a particular Presentity - DELETE	6.23.6	

B.1.24 SCR for REST.Presence.PresenceSubscriptions.Notifications Server

Item	Function	Reference	Requirement
REST-PRESENCE-PS-NOTIF-S-001-	Support for notifying	6.24	

Item	Function	Reference	Requirement
M	application (Watcher) about changes in presence information		
REST-PRESENCE-PS-NOTIF-S-002-M	Notify application about changes in presence information - POST	6.24.5	

B.1.25 SCR for REST.Presence.Watcher.Subscriptions.PresenceListSubscriptions Server

Item	Function	Reference	Requirement
REST-PRESENCE-WATCH-SUBSCR-PLS-S-001-O	Support for read access for all presence subscriptions for all Presence Lists	6.25	REST-PRESENCE-WATCH-SUBSCR-PLS-S-002-O
REST-PRESENCE-WATCH-SUBSCR-PLS-S-002-O	Read all active Presence List subscriptions towards all Presence Lists - GET	6.25.3	

B.1.26 SCR for REST.Presence.Watcher.Subscriptions.PresenceListSubscriptions. SinglePresenceList Server

Item	Function	Reference	Requirement
REST-PRESENCE-WATCH-SUBSCR-PLS-SINGPL-S-001-M	Support for Presence List subscriptions for a single Presencelist	6.26	
REST-PRESENCE-WATCH-SUBSCR-PLS-SINGPL-S-002-O	Read all active Presence List subscriptions for a particular Presence List - GET	6.26.3	
REST-PRESENCE-WATCH-SUBSCR-PLS-SINGPL-S-003-M	Create Presence List subscription for a particular Presence List – POST (XML or JSON)	6.26.5	
REST-PRESENCE-WATCH-SUBSCR-PLS-SINGPL-S-004-O	Create Presence List subscription for a particular Presence List – POST (application/x-www-form-urlencoded)	C.5	

B.1.27 SCR for REST.Presence.Watcher.Individual.Subscriptions.PresenceListSubscriptions.SinglePresenceList Server

Item	Function	Reference	Requirement
REST-PRESENCE-WATCH-IND-SUBSCR-PLS-SINGPL-S-001-M	Support for individual Presence List subscription for a single Presence List	6.27	
REST-PRESENCE-WATCH-IND-SUBSCR-PLS-SINGPL-S-002-O	Read an individual Presence List subscription for a particular Presence List - GET	6.27.3	
REST-PRESENCE-WATCH-IND-SUBSCR-PLS-SINGPL-S-003-M	Update Presence List subscription for a particular Presence List - PUT	6.27.4	
REST-PRESENCE-WATCH-IND-SUBSCR-PLS-SINGPL-S-004-M	Delete (terminate) Presence List subscription for a particular Presence List - DELETE	6.27.6	

B.1.28 SCR for REST.Presence.PresenceListSubscriptions.Notifications Server

Item	Function	Reference	Requirement
REST-PRESENCE-PLS-NOTIF-S-001-M	Support for notifying application (Watcher) about changes in presence status for a Presentity from a particular Presence List	6.28	
REST-PRESENCE-PLS-NOTIF-S-002-M	Notify application about changes in presence information for a Presentity from a particular Presence List - POST	6.28.5	

B.1.29 SCR for REST.Presence.Presentity.Portrait.Icon Server

Item	Function	Reference	Requirement
REST-PRESENCE-PRES-PORTR-ICON-S-001-M	Support for the management of a portrait icon by Presentity	6.29	
REST-PRESENCE-PRES-PORTR-ICON-S-002-M	Retrieve portrait icon - GET	6.29.3	
REST-PRESENCE-PRES-PORTR-	Create/replace portrait	6.29.4	

Item	Function	Reference	Requirement
ICON-S-003-M	icon - PUT		
REST-PRESENCE-PRES-PORTR- ICON-S-004-M	Delete (remove) portrait icon - DELETE	6.29.6	

Appendix C. Application/x-www-form-urlencoded Request Format for POST Operations (Normative)

This section defines a format for the RESTful Presence API requests where the body of the request is encoded using the application/x-www-form-urlencoded MIME type.

Note: only the request body is encoded as application/x-www-form-urlencoded, the response is still encoded as XML or JSON depending on the preference of the client and the capabilities of the server. Names and values MUST follow the application/x-www-form-urlencoded character escaping rules from [W3C_URLENC].

The encoding is defined below for all Presence REST operations which are based on POST requests:

C.1 Create Presence Source

This operation is used for creating a Presence Source with a specified time-to-live, see section 6.1.5.

Creation of presence information using an application/x-www-form-urlencoded request is supported for “person” related presence information only. This specification does not support creation of Presence Sources that include “service” or “device” related presence information in application/x-www-form-urlencoded requests.

The request parameters for creation of Presence Source are as follows:

Name	Type/Values	Optional	Description
clientCorrelator	xsd:string	Yes	<p>A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server.</p> <p>This element SHOULD be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate Presence Source creation in such situations.</p> <p>In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.</p>
applicationTag	xsd:string	Yes	<p>A tag that the client MAY use to tag this particular resource on the server. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.</p>
duration	xsd:int	Yes	<p>Specifies the duration of the publication life time in seconds. When this time has elapsed the Presence Source will expire unless it has been refreshed (by using some other methods than application/x-www-form-urlencoded).</p> <p>If the parameter is omitted, a default value specified by the server policy will be used for the publication life time..</p> <p>A too low value (including “0”) will result in an</p>

			error response. What is too low is defined by service policy. A too high requested value may be reduced by the server according to the service policy.
person-activities	ActivityValue [0..unbounded]	Yes	The Presentity's activity (available, busy, lunch, etc.)
person-activities-note	xsd:string	Yes	A textual description of what the user is currently doing. MAY only be included if person-activities is specified.
person-activities-other	xsd:string	Yes	MAY only be included if person-activities is specified.
person-activities-until	xsd:dateTime	Yes	Indicates an absolute time the attribute is expected to be valid. MAY only be included if person-activities is specified.
person-placeType	PlaceTypeValue [0..unbounded]	Yes	Specifies the place type values.
person-placeType-note	xsd:string	Yes	A textual description of what type of place the person is located in. MAY only be included if person-placeType is specified.
person-placeType-other	xsd:string	Yes	MAY only be included if person-placeType is specified.
person-placeType-until	xsd:dateTime	Yes	Indicates an absolute time the attribute is expected to be valid. MAY only be included if person-placeType is specified.
person-privacy	PrivacyValue [0..unbounded]	Yes	Specifies the type of privacy expected at the user's current position.
person-privacy-note	xsd:string	Yes	A textual description of the privacy. MAY only be included if person-privacy is specified.
person-sphere	SphereValue	Yes	Specifies the sphere the user is in.
person-sphere-other	xsd:string	Yes	A textual description of the sphere. MAY only be included if person-sphere is specified.
person-mood	MoodValue [0..unbounded]	Yes	Specifies the mood of the user (happy, sad etc)
person-mood-note	xsd:string	Yes	A textual description of the mood. MAY only be included if person-mood is specified.
person-mood-other	xsd:string	Yes	MAY only be included if person-mood is specified.
person-mood-until	xsd:dateTime	Yes	Indicates an absolute time the attribute is expected to be valid. MAY only be included if person-mood is specified.
person-placels-Audio	PlacelsAudio	Yes	Describes place conditions for audio communication.

person-placels-Video	PlacelsVideo	Yes	Describes place conditions for video communication.
person-placels-Text	PlacelsText	Yes	Describes place conditions for real-time and instant-messaging communication.
person-timeOffset	xsd:int	Yes	Number of minutes of offset from UTC that the user is currently at.
person-timeOffset-until	xsd:dateTime	Yes	Indicates an absolute time the attribute is expected to be valid. MAY only be included if person-timeOffset is specified.
person-statusIcon	xsd:anyURI	Yes	The URL of the content.
person-statusIcon-contentType	xsd:string	Yes	The content-type related to the content. MAY only be included if person-statusIcon is specified.
person-statusIcon-eTag	xsd:string	Yes	HTTP ETag identifier of the addressed content. The Presentity MAY specify an eTag (i.e. version) of the content allowing the Watcher to detect when the content has been updated. MAY only be included if person-statusIcon is specified.
person-statusIcon-fSize	xsd:int	Yes	The size of the content in bytes (e.g. 102400). MAY be included if person-statusIcon is specified only.
person-statusIcon-resolution	xsd:string	Yes	The resolution of the content. The value of the string is of the type "width x height" (e.g. 640x480) where width and height are specified in number of pixels. MAY be included if person-statusIcon is defined only.
person-statusIcon-until	xsd:dateTime	Yes	Indicates an absolute time the attribute is expected to be valid. MAY only be included if person-statusIcon is defined.
person-class	xsd:token	Yes	Specifies a class of the presence information.
person-note	xsd:string	Yes	Contains a tagline.
person-note-lang	xsd:string	Yes	Language of the text for a note. The format of this parameter is aligned with that of the built-in XML attribute xml:lang [W3C_XML11]. It is RECOMMENDED to provide this parameter
person-location-circle-latitude	xsd:float	Yes	Latitude of center point. MUST be included if person-location-circle-longitude is included.
person-location-circle-longitude	xsd:float	Yes	Longitude of center point. MUST be included if person-location-circle-latitude is included.
person-location-circle-	xsd:float	Yes	Radius of circle around center point in meters.

radius			MAY be included if person-location-circle-latitude and person-location-circle-longitude is specified only.
person-overridingWillingness	OpenOrClosed	Yes	The overriding willingness for a person.
person-overridingWillingness-until	xsd:dateTime	Yes	Specifies validity for the attribute. MAY only be included if person-overridingWillingness is specified.
person-link	xsd:anyURI	Yes	The address for the link
person-link-label	xsd:string	Yes	Label for the link. MAY only be included if person-link is specified.
person-link-priority	xsd:decimal	Yes	Priority for the link. MAY only be included if person-link is specified.
person-card	xsd:anyURI	Yes	URI to a business card.
person-displayName	xsd:string	Yes	A display name of a person.
person-homePage	xsd:anyURI	Yes	URI pointing to general information about a person.
person-icon	xsd:anyURI	Yes	URI pointing to an image/icon of the person. Note: It is recommended to use the StatusIcon for sharing icons/avatars between users.
person-map	xsd:anyURI	Yes	URI pointing to a map related to the person
person-sound	xsd:anyURI	Yes	URI pointing to a sound related to the person.

If the operation was successful, it returns an HTTP Status of “201 Created”.

C.1.1 Example: creating Presence Source for user (Informative)

C.1.1.1 Request

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn

clientCorrelator=123&
applicationTag=myApp&
duration=3600&
person-activities=Busy&
person-activities-note=meeting%20until%20lunch&
person-placeType=Home&
person-placeType-note=At%20home!&
person-mood=happy
```

C.1.1.2 Response

HTTP/1.1 201 Created

Location: http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123

Date: Thu, 04 Jun 2009 02:51:59 GMT

Content-Type: application/xml

Content-Length: nnnn

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSource xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <clientCorrelator>123</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>3600</duration>
  <presence>
    <person>
      <activities>
        <activityValue>Busy</activityValue>
        <note>meeting until lunch</note>
      </activities>
      <placeType>
        <placeTypeValue>Home</placeTypeValue>
        <note>At home</note>
      </placeType>
      <mood>
        <moodValue>Happy</moodValue>
      </mood>
    </person>
  </presence>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123</resourceURL>
</pr:presenceSource>
```

C.2 Create authorization rule

This operation is used by a Presentity to create an authorization rule, see section 6.10.5. The authorization rules control who will have access to Presentity's presence information. A Watcher may be authorized to all or a subset of the available presence attributes.

The request parameters are as follows:

Name	Type/Values	Optional	Description
ruleName	xsd:ID	No	A name associated with the rule.
watcherUserId	xsd:anyURI [0..unbounded]	Choice	Contains a list of Watcher identities for which the rule will apply (e.g. 'sip' URI, 'tel' URI, 'acr' URI).
memberListId	xsd:string [0..unbounded]	Choice	Contains a list of member list identities for which the rule will apply.
domainName	xsd:string [0..unbounded]	Choice	Contains a list of domain names for which the rule will apply.
anonymous	xsd:boolean	Choice	Specifies that the rule will apply for anonymous requests. XSD modeling sets this parameter value to "true".

otherUser	xsd:boolean	Choice	Specifies that the rule will apply for unknown users. It allows the client to specify a default behavior for unknown users. XSD modeling sets this parameter value to "true".
decision	DefaultDecisionValue	No	The authorization decision for the rule.
presenceFilter	xsd:anyURI [0..unbounded]	Yes	Contains filter indicating which presence attributes the Watchers are allowed to see. An empty filter means that the Watchers have access to all presence attributes.

Application/x-www-form-urlencoded modelling use a "choice" to select either watcherUserId, memberListId, domainName, anonymous or otherUser.

If the operation was successful, it returns an HTTP Status of "201 Created".

C.2.1 Example: creating an authorization rule (Informative)

C.2.1.1 Request

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn

ruleName=allowList&
watcherUserId=tel%3A%2B19585550102&
watcherUserId=tel%3A%2B19585550104&
watcherUserId=tel%3A%2B19585550105&
decision=Allow
```

C.2.1.2 Response

```
HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule003
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:rule xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <ruleName>allowList</ruleName>
  <watcherUserId>tel:+19585550102</watcherUserId>
  <watcherUserId>tel:+19585550104</watcherUserId>
  <watcherUserId>tel:+19585550105</watcherUserId>
  <decision>Allow</decision>
</pr:rule>
```

C.3 Create Watchers subscription

This resource is used by a Presentity to create a Watchers subscription, i.e. subscription for changes in the Watchers list, see section 6.18.5. The list contains Watchers that are subscribing for presence information about the Presentity.

The notifyURL either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

The request parameters are as follows:

Name	Type/Value	Optional	Description
presentityUserId	xsd:anyURI	Yes	Identifies the Presentity for which the subscription is created towards (e.g. 'sip' URI, 'tel' URI, 'acr' URI). Mandatory in responses.
notifyURL	xsd:anyURI	No	Notification endpoint definition.
callbackData	xsd:string	Yes	Data the application can register with the server when subscribing to notifications, and that are passed back unchanged in each of the related notifications.
notificationFormat	common:NotificationFormat	Yes	Default: XML Application can specify format of the resource representation in notifications that are related to this subscription. The choice is between {XML, JSON}.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate subscription creation in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
applicationTag	xsd:string	Yes	A tag that the client MAY use to tag this particular resource on the server. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.
duration	xsd:int	Yes	Specifies the duration of the subscription in seconds. When this time has elapsed the subscription will expire unless it has been refreshed (by using some other methods than application/x-www-form-urlencoded).

			<p>The server SHALL always include the remaining duration of the subscription in the response.</p> <p>A too high requested value may be reduced by the server according to the service policy.</p> <p>If the parameter is omitted, a default value specified by the server policy will be used for the subscription life time.</p>
resourceStatusFilter	ResourceStatus [0..unbounded]	Yes	<p>Indicates the desired Watcher subscription statuses that the Presentity is interested to get notifications about.</p> <p>If the parameter is omitted or there is an empty filter it means monitoring all states.</p>
frequency	xsd:int	Yes	<p>Maximum frequency of notifications, expressed as minimum time between notifications in seconds.</p>

If the operation was successful, it returns an HTTP Status of “201 Created”.

C.3.1 Example: creating new Watchers subscription, using tel URI (Informative)

C.3.1.1 Request

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions HTTP/1.1
Accept: application/xml
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn

presentityUserId=tel%3A%2B19585550100&
notifyURL=http://application.example.com/notifications/watchersNotification&
callbackData=1234&
clientCorrelator=321&
applicationTag=myApp&
duration=7200&
resourceStatusFilter=Pending
```

C.3.1.2 Response

```
HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackReference>
```

```

<notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
<callbackData>1234</callbackData>
</callbackReference>
<clientCorrelator>321</clientCorrelator>
<applicationTag>myApp</applicationTag>
<duration>7200</duration>
<resourceStatusFilter>Pending</resourceStatusFilter>
<resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001
</resourceURL>
</pr:watchersSubscription>

```

C.3.2 Example: creating new Watchers subscription, using ACR (Informative)

C.3.2.1 Request

```

POST /exampleAPI/presence/v1/acr%3A%2B19585550100/subscriptions/watchersSubscriptions HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn

```

```

presentityUserId=acr%3A%2B19585550100&
notifyURL=http://application.example.com/notifications/watchersNotification&
callbackData=1234&
clientCorrelator=321&
applicationTag=myApp&
duration=7200&
resourceStatusFilter=Pending

```

C.3.2.2 Response

```

HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/presence/v1/acr%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

```

```

<?xml version="1.0" encoding="UTF-8"?>
<pr:watchersSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>acr%3A%2B19585550100</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/watchersNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <resourceStatusFilter>Pending</resourceStatusFilter>
  <resourceURL>http://example.com/exampleAPI/presence/v1/acr%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001
  </resourceURL>

```

```
</pr:watchersSubscription>
```

C.4 Create presence subscription

This operation is used by a Watcher to create a new presence subscription towards the specified Presentity, see section 6.22.5.

The notifyURL either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

The request parameters are as follows:

Name	Type/value	Optional	Description
presentityUserId	xsd:anyURI	Yes	Identifies the Presentity for which the subscription is created towards (e.g. 'sip' URI, 'tel' URI, 'acr' URI). Mandatory in responses. If presentityUserId is also part of the request URL, the two MUST have the same value
notifyURL	xsd:anyURI	No	Notification endpoint definition
callbackData	xsd:string	Yes	Data the application can register with the server when subscribing to notifications, and that are passed back unchanged in each of the related notifications.
notificationFormat	common:NotificationFormat	Yes	Default: XML Application can specify format of the resource representation in notifications that are related to this subscription. The choice is between {XML, JSON}.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a request to create a resource on the server. This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate subscription creation in such situations. In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it..
applicationTag	xsd:string	Yes	A tag that the client MAY use to tag this particular resource on the server. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the

			field is not present, the server SHALL NOT generate it.
anonymous	xsd:boolean	Yes	Allows the Watcher to request that its user identity is not revealed to the Presentity. XSD modeling sets this parameter value to "true"
duration	xsd:int	Yes	Specifies the duration of the subscription in seconds. When this time has elapsed the subscription will expire unless it has been refreshed (by using some other methods than application/x-www-form-urlencoded). The server SHALL always include the remaining duration of the subscription in the response. A too high requested value may be reduced by the server according to the service policy. If the parameter is omitted, a default value specified by the server policy will be used for the subscription life time.
presenceFilter	xsd:anyURI [0..unbounded]	Yes	Allows the Watcher to indicate what type of presence information he/she is interested in. The desired attributes are indicated with relative paths according to the [ResourceRelPath] in sections 5.2.2.3, 5.2.2.4, 5.2.2.5 and 5.2.2.6. If the parameter is omitted or there is an empty filter it means monitoring of all attribute types.
frequency	xsd:int	Yes	Maximum frequency of notifications, expressed as minimum time between notifications in seconds.

If the operation was successful, it returns an HTTP Status of "201 Created".

C.4.1 Example: creating new presence subscription for Presentity (Informative)

C.4.1.1 Request

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100 HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn

presentityUserId=tel%3A%2B19585550100&
notifyURL= http://application.example.com/notifications/presenceNotification&
callbackData=1234&
clientCorrelator=321&
applicationTag=myApp&
duration=7200
```

C.4.1.2 Response

HTTP/1.1 201 Created

Location:

http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001

Date: Thu, 04 Jun 2009 02:51:59 GMT

```
<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presentityUserId>tel:+19585550100</presentityUserId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>7200</duration>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001
  </resourceURL>
</pr:presenceSubscription>
```

C.5 Create Presence List subscription

This operation is used by a Watcher to create a new Presence List subscription towards the specified Presence List, see section 6.26.5.

The notifyURL either contains the Client-side Notification URL (as defined by the client) or the Server-side Notification URL (as obtained during the creation of the Notification Channel [REST_NetAPI_NotificationChannel]).

The request parameters are as follows:

Name	Type/values	Optional	Description
presenceListId	xsd:string	Yes	Identifies the Presence List for which the subscription is created towards. Mandatory in responses.
notifyURL	xsd:anyURI	No	Notification endpoint definition
callbackData	xsd:string	Yes	Data the application can register with the server when subscribing to notifications, and that are passed back unchanged in each of the related notifications.
notificationFormat	common:NotificationFormat	Yes	Default: XML Application can specify format of the resource representation in notifications that are related to this subscription. The choice is between {XML, JSON}.
clientCorrelator	xsd:string	Yes	A correlator that the client can use to tag this particular resource representation during a

			<p>request to create a resource on the server.</p> <p>This element MAY be present. Note: this allows the client to recover from communication failures during resource creation and therefore avoids duplicate subscription creation in such situations.</p> <p>In case the element is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it. .</p>
applicationTag	xsd:string	Yes	<p>A tag that the client MAY use to tag this particular resource on the server. In case the field is present, the server SHALL not alter its value, and SHALL provide it as part of the representation of this resource. In case the field is not present, the server SHALL NOT generate it.</p>
anonymous	xsd:boolean	Yes	<p>Allows the Watcher to request that its user identity is not revealed to the Presentity. XSD modeling sets this parameter value to "true"</p>
duration	xsd:int	Yes	<p>Specifies the duration of the subscription in seconds. When this time has elapsed the subscription will expire unless it has been refreshed (by using some other methods than application/x-www-form-urlencoded).</p> <p>The server SHALL always include the remaining duration of the subscription in the response.</p> <p>A too high requested value may be reduced by the server according to the service policy.</p> <p>If the parameter is omitted, a default value specified by the server policy will be used for the subscription life time.</p>
presenceFilter	xsd:anyURI [0..unbounded]	Yes	<p>Allows the Watcher to indicate what type of presence information he/she is interested in. The desired attributes are indicated with relative paths according to the [ResourceRelPath] in sections 5.2.2.3, 5.2.2.4, 5.2.2.5 and 5.2.2.6.</p> <p>If the parameter is omitted or there is an empty filter it means monitoring of all attribute types.</p>
frequency	xsd:int	Yes	<p>Maximum frequency of notifications, expressed as minimum time in seconds between notifications in seconds.</p>

If the operation was successful, it returns an HTTP Status of "201 Created".

C.5.1 Example: creating new Presence List subscription towards a single Presence List (Informative)

C.5.1.1 Request

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends HTTP/1.1
Host: example.com
Accept: application/xml
Content-Type: application/x-www-form-urlencoded
Content-Length: nnnn

presenceListId=myFriends&
notifyURL=http://application.example.com/notifications/presenceListNotification&
callbackData=1234&
clientCorrelator=321&
applicationTag=myApp&
duration=7200&
presenceFilter=person%2fmood&
frequency=600
```

C.5.1.2 Response

```
HTTP/1.1 201 Created
Location:
http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<pr:presenceListSubscription xmlns:pr="urn:oma:xml:rest:netapi:presence:1">
  <presenceListId>myFriends</presenceListId>
  <callbackReference>
    <notifyURL>http://application.example.com/notifications/presenceListNotification</notifyURL>
    <callbackData>1234</callbackData>
  </callbackReference>
  <clientCorrelator>321</clientCorrelator>
  <applicationTag>myApp</applicationTag>
  <duration>5200</duration>
  <presenceFilter>person/mood</presenceFilter>
  <frequency>600</frequency>
  <resourceURL>http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/
  sub001</resourceURL>
</pr:presenceListSubscription>
```

Appendix D. JSON examples (Informative)

JSON (JavaScript Object Notation) is a lightweight, text-based, language-independent data interchange format. It provides a simple means to represent basic name-value pairs, arrays and objects. JSON is relatively trivial to parse and evaluate using standard JavaScript libraries, and hence is suited for invocations from browsers or other processors with JavaScript engines. Further information on JSON can be found at [RFC4627].

The following examples show the request and response for various operations using a JSON binding. The examples follow the XML to JSON serialization rules in [REST_NetAPI_Common]. A JSON response can be obtained by using the content type negotiation mechanism specified in [REST_NetAPI_Common].

For full details on the operations themselves please refer to the section number indicated.

D.1 Retrieving all Presence Sources for user (section 6.1.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"presenceSourceList": {
  "presenceSource": [
    {
      "applicationTag": "myApp",
      "clientCorrelator": "123",
      "duration": "3575",
      "presence": {
        "device": {
          "deviceId": "mac:321",
          "networkAvailability": {"network": {
            "connectionStatus": "Active",
            "id": "GPRS"
          }}
        },
        "person": {"mood": {"moodValue": "Happy"}},
        "service": {
          "devices": {"deviceId": "mac:321"},
          "serviceAvailability": "Open",
          "serviceId": "org.openmobilealliance:IM-Session",
          "version": "1.0"
        }
      },
      "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123"
    },
    {
      "presence": {"person": {"noteList": {"note": {
        "$t": "I am on vacation!"
      }}}
    }
  ]
}
```

```

    "lang": "en"
  }},
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent"
}
],
"resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources"
}}

```

D.2 Retrieving of all Presence Sources metadata using a filter (section 6.1.3.2)

Request:

```

GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources?presenceSourceFilter=presenceSourceMetaData HTTP/1.1
Host: example.com
Accept: application/json

```

Response:

```

HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"presenceSourceList": {
  "presenceSource": [
    {
      "applicationTag": "myApp",
      "clientCorrelator": "123",
      "duration": "3575",
      "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123"
    },
    {"resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent"}
  ],
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources"
}}

```

D.3 Creating Presence Source for user (section 6.1.5.1)

Request:

```

POST /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceSource": {
  "applicationTag": "myApp",

```



```
POST /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources HTTP/1.1
Content-Type: application/json
Accept: application/json
Content-Length: nnnn
Host: example.com
```

```
{"presenceSource": {
  "applicationTag": "myApp",
  "clientCorrelator": "123",
  "duration": "7200",
  "presence": {"person": {"mood": {"moodValue": "Happy"}}}
}}
```

Response:

```
HTTP/1.1 409 Conflict
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn
```

```
{"requestError": {
  "link": {
    "href": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources",
    "rel": "PresenceSourceList"
  },
  "policyException": {
    "messageId": "POL02600",
    "text": "Maximum number of presence sources exceeded",
  }
}}
```

D.5 Retrieving Presence Source (section 6.2.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123 HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

```
{"presenceSource": {
  "applicationTag": "myApp",
  "clientCorrelator": "123",
  "duration": "5237",
  "presence": {
    "device": {
      "deviceId": "mac:321",
```

```
"networkAvailability": {"network": {
  "connectionStatus": "Active",
  "id": "GPRS"
}},
},
"person": {"mood": {"moodValue": "Happy"}},
"service": {
  "devices": {"deviceId": "mac:321"},
  "serviceAvailability": "Open",
  "serviceId": "org.openmobilealliance:IM-Session",
  "version": "1.0"
}
},
"resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123"
}}
```

D.6 Retrieving Presence Source which does not exist (section 6.2.3.2)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123 HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 404 Not Found
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"requestError": {
  "link": {
    "href": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123",
    "rel": "PresenceSourceList"
  },
  "serviceException": {
    "messageId": "SVC1001",
    "text": "Presence source does not exist"
  }
}}
```

D.7 Updating Presence Source (section 6.2.4.1)

Request:

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123 HTTP/1.1
Content-Type: application/json
Accept: application/json
```

```

Content-Length: nnnn
Host: example.com

{"presenceSource": {
  "duration": "7200",
  "presence": {
    "device": {
      "deviceId": "mac:321",
      "networkAvailability": {"network": {
        "connectionStatus": "Active",
        "id": "GPRS"
      }}
    }
  },
  "person": {"mood": {"moodValue": "Invincible"}},
  "service": {
    "devices": {"deviceId": "mac:321"},
    "serviceAvailability": "Closed",
    "serviceId": "org.openmobilealliance:IM-Session",
    "version": "1.0"
  }
},
"resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123"
}}

```

Response:

```

HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/{presenceSourceId}
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"presenceSource": {
  "duration": "7200",
  "presence": {
    "device": {
      "deviceId": "mac:321",
      "networkAvailability": {"network": {
        "connectionStatus": "Active",
        "id": "GPRS"
      }}
    }
  },
  "person": {"mood": {"moodValue": "Invincible"}},
  "service": {
    "devices": {"deviceId": "mac:321"},
    "serviceAvailability": "Closed",
    "serviceId": "org.openmobilealliance:IM-Session",
    "version": "1.0"
  }
},
"resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123"
}}

```

D.8 Removing Presence Source (section 6.2.6.1)

Request:

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123 HTTP/1.1
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.9 Retrieving individual presence attribute (section 6.3.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123/person/mood HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"mood": {"moodValue": "Happy"}}
```

D.10 Updating individual presence attribute (section 6.3.4.1)

Request:

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123/person/mood HTTP/1.1
Content-Type: application/json
Accept: application/json
Content-Length: nnnn
Host: example.com

{"mood": {"moodValue": "Excited"}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"mood": {"moodValue": "Excited"}}
```


D.11 Removing individual presence attribute (section 6.3.6.1)

Request:

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/prs123/person/mood HTTP/1.1
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.12 Retrieving persistent presence information (section 6.4.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent?resFormat=JSON HTTP/1.1
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"presenceSource": {
  "presence": {"person": {"noteList": {"note": {
    "$t": "Im on vacation!",
    "lang": "en"
  }}}},
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent"
}}
```

D.13 Updating persistent presence information (section 6.4.4.1)

Request:

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent HTTP/1.1
Host: example.com
If-Match: "10"
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceSource": {
  "presence": {"person": {
    "noteList": {"note": {
      "$t": "My picture is updated!",
      "lang": "en"
    }}
  }}
}
```

```

    }},
    "statusIcon": {
      "contentType": "image/jpeg",
      "eTag": "123",
      "statusIconAddress": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg"
    }
  }},
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent"
}}

```

Response:

HTTP/1.1 412 Precondition Failed
Date: Thu, 04 Jun 2009 02:51:59 GMT

Request:

```

PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent HTTP/1.1
Host: example.com
If-Match: "11"
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceSource": {
  "presence": {"person": {
    "noteList": {"note": {
      "$t": "My picture is updated!",
      "lang": "en"
    }},
    "statusIcon": {
      "contentType": "image/jpeg",
      "eTag": "123",
      "statusIconAddress": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg"
    }
  }},
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent"
}}

```

Response:

```

HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
ETag: "12"
Content-Type: application/json
Content-Length: nnnn

{"presenceSource": {
  "presence": {"person": {
    "noteList": {"note": {

```

```
    "$t": "My picture is updated!",
    "lang": "en"
  }},
  "statusIcon": {
    "contentType": "image/jpeg",
    "eTag": "123",
    "statusIconAddress": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg"
  }
},
"resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent"
}}
```

D.14 Removing persistent presence information (section 6.4.6.1)

Request:

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent HTTP/1.1
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.15 Retrieving individual persistent presence attribute (section 6.5.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent/person/statusIcon HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"statusIcon": {
  "contentType": "image/jpeg",
  "eTag": "123",
  "statusIconAddress": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg"
}}
```

D.16 Updating individual persistent presence attribute (section 6.5.4.1)

Request:

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent/person/statusIcon HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"statusIcon": {
  "contentType": "image/jpeg",
  "eTag": "456",
  "statusIconAddress": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg"
}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"statusIcon": {
  "contentType": "image/jpeg",
  "eTag": "456",
  "statusIconAddress": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg"
}}
```

D.17 Removing individual persistent presence attribute (section 6.5.6.1)

Request:

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/presenceSources/persistent/person/mood HTTP/1.1
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.18 Retrieving list of available contents (section 6.6.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/content HTTP/1.1
Host: example.com
```

Accept: application/json

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"contentList": {
  "content": [
    {
      "contentType": "image/jpeg",
      "link": {
        "href": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic003.jpg",
        "rel": "content"
      }
    },
    {
      "contentType": "image/jpeg",
      "link": {
        "href": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic004.jpg",
        "rel": "content"
      }
    }
  ],
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/content"
}}
```

D.19 Retrieving individual content by Presentity (section 6.7.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg HTTP/1.1
Host: example.com
Accept: image/jpeg
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: image/jpeg
Content-Length: nnnn
```

data

D.20 Uploading/updating individual content by Presentity (section 6.7.4.1)

Request:

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg HTTP/1.1
Host: example.com
Content-Type: image/jpeg
Content-Length: nnnn
```

data

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.21 Removing individual content by Presentity (section 6.7.6.1)

Request:

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/content/pic001.jpg HTTP/1.1
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.22 Retrieving list of Watchers (section 6.8.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/watchers?resourceStatusFilter=Pending HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

```
{"watcherList": {
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/watchers",
  "watcher": {
    "displayName": "Bob",
    "resourceStatus": "Pending",
    "watcherUserId": "tel:+19585550101"
  }
}}
```

D.23 Retrieving individual Watcher (section 6.9.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/watchers/{tel%3A%2B19585550101 HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"watcher": {
  "displayName": "Bob",
  "resourceStatus": "Pending",
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/watchers/tel%3A%2B19585550101",
  "watcherUserId": "tel:+19585550101"
}}
```

D.24 Retrieving all authorization rules (section 6.10.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"ruleList": {
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules",
  "rule": [
    {
      "decision": "Allow",
      "ruleName": "allowList",
      "watcherUserId": [
        "tel:+19585550102",
        "tel:+19585550104"
      ]
    },
    {
      "decision": "Block",
      "memberListId": "myBlockList",
      "ruleName": "blockList"
    }
  ]
}}
```

```
]
}}
```

D.25 Creating an authorization rule (section 6.10.5.1)

Request:

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"rule": {
  "decision": "Confirm",
  "otherUser": null,
  "ruleName": "otherUsers"
}}
```

Response:

```
HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule003
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"rule": {
  "decision": "Confirm",
  "otherUser": null,
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule003",
  "ruleName": "otherUsers"
}}
```

D.26 Creating an authorization rule, response with resourceReference (section 6.10.5.2)

Request:

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"rule": {
  "decision": "Confirm",
  "otherUser": null,
```



```
"ruleName": "otherUsers"  
}}
```

Response:

```
HTTP/1.1 201 Created  
Location: http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule003  
Date: Thu, 04 Jun 2009 02:51:59 GMT  
Content-Type: application/json  
Content-Length: nnnn  
  
{  
  "resourceReference": {  
    "resourceURL":  
      "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule003"}  
}
```

D.27 Retrieving an authorization rule (section 6.11.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule001 HTTP/1.1  
Host: example.com  
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK  
Date: Thu, 04 Jun 2009 02:51:59 GMT  
Content-Type: application/json  
Content-Length: nnnn  
  
{  
  "rule": {  
    "decision": "Allow",  
    "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule001",  
    "ruleName": "allowList",  
    "watcherUserId": [  
      "tel:+19585550102",  
      "tel:+19585550104"  
    ]  
  }  
}
```

D.28 Updating an authorization rule (section 6.11.4.1)

Request:

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule001 HTTP/1.1  
Host: example.com  
Content-Type: application/json  
Accept: application/json  
Content-Length: nnnn  
  
{  
  "rule": {  
    "decision": "Allow",
```

```
"resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule001",
"ruleName": "allowList",
"watcherUserId": [
  "tel:+19585550102",
  "tel:+19585550104",
  "tel:+19585550105"
]
}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"rule": {
  "decision": "Allow",
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule001",
  "ruleName": "allowList",
  "watcherUserId": [
    "tel:+19585550102",
    "tel:+19585550104",
    "tel:+19585550105"
  ]
}}
```

D.29 Removing an authorization rule (section 6.11.6.1)

Request:

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule002 HTTP/1.1
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.30 Retrieving individual authorization rule data (section 6.12.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule002/watchers/tel%3A%2B19585550102 HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"watcherUserId": "\n tel:+19585550102\n"}
```

D.31 Updating individual authorization rule data (section 6.12.4.1)

Request:

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule002/watchers/tel%3A%2B19585550103 HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"watcherUserId": "\n tel:+19585550103\n"}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"watcherUserId": "\n tel:+19585550103\n"}
```

D.32 Removing individual authorization rule data (section 6.12.6.1)

Request:

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/authorization/rules/rule002/watchers/tel%3A%2B19585550103
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.33 Retrieving all presence information for Presentity (section 6.13.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100 HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn
```

```
{
  "presenceContact": {
    "presence": {
      "device": {
        "deviceId": "mac:321",
        "networkAvailability": {
          "network": {
            "connectionStatus": "Active",
            "id": "GPRS"
          }
        }
      },
      "person": {
        "mood": {
          "moodValue": "Happy"
        },
        "noteList": {
          "note": {
            "$t": "Im on vacation!",
            "lang": "en"
          }
        }
      },
      "service": {
        "devices": {
          "deviceId": "mac:321"
        },
        "serviceAvailability": "Open",
        "serviceId": "org.openmobilealliance:IM-Session",
        "version": "1.0"
      }
    },
    "presentityUserId": "tel:+19585550100",
    "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100"
  }
}
```

D.34 Retrieving presence information for Presentity by using filter (section 6.13.3.2)

Request:

```
GET
/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100?presenceFilter=person/mood&presenceFilter=service/org.openmobilealliance:IM-Session/1.0/serviceAvailability HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
```

Date: Thu, 04 Jun 2009 02:51:59 GMT

Content-Type: application/json

Content-Length: nnnn

```
{"presenceContact": {
  "presence": {
    "person": {"mood": {"moodValue": "Happy"}},
    "service": {
      "devices": {"deviceId": "mac:321"},
      "serviceAvailability": "Open",
      "serviceId": "org.openmobilealliance:IM-Session",
      "version": "1.0"
    }
  }
},
"presentityUserId": "tel:+19585550100",
"resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100"
}}
```

D.35 Retrieving individual presence attribute for Presentity (section 6.14.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100/person/noteList HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn
```

```
{"noteList": {"note": {
  "$t": "Im on vacation!",
  "lang": "en"
}}}
```

D.36 Retrieving presence information for all Presentities in a Presence List (section 6.15.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/presenceLists/myFriends HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```

HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"presenceList": {
  "presenceContact": [
    {
      "presence": {
        "device": {
          "deviceId": "mac:321",
          "networkAvailability": {"network": {
            "connectionStatus": "Active",
            "id": "GPRS"
          }}
        },
        "person": {
          "mood": {"moodValue": "Happy"},
          "noteList": {"note": {
            "$t": "Im on vacation!",
            "lang": "en"
          }}
        },
        "service": {
          "devices": {"deviceId": "mac:321"},
          "serviceAvailability": "Open",
          "serviceId": "org.openmobilealliance:IM-Session",
          "version": "1.0"
        }
      },
      "presenceUserId": "tel:+19585550100",
      "resourceStatus": "Active",
      "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550100"
    },
    {
      "presenceUserId": "tel:+19585550102",
      "resourceStatus": "Pending",
      "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550102"
    },
    {
      "presenceUserId": "tel:+19585550104",
      "resourceStatus": "TerminatedNoResource",
      "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContacts/tel%3A%2B19585550104"
    }
  ],
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceLists/myFriends"
}}

```

D.37 Retrieving content by Watcher (section 6.16.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/presenceContactsContent/{tel%3A%2B19585550100}/pic001.jpg HTTP/1.1
Host: example.com
Accept: image/jpeg
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: image/jpeg
Content-Length: nnnn
```

data

D.38 Retrieving all active subscriptions for user (section 6.17.3.1)**Request:**

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn
```

```
{
  "subscriptionList": {
    "presenceListSubscriptionCollection": {
      "presenceListSubscription": {
        "applicationTag": "myApp",
        "callbackReference": {
          "callbackData": "2345",
          "notifyURL": "http://application.example.com/notifications/presenceListNotification"
        },
        "clientCorrelator": "432",
        "duration": "4629",
        "presenceListId": "myFriends",
        "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub002"
      },
      "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions"
    },
    "presenceSubscriptionList": {
      "presenceSubscription": {
        "applicationTag": "myApp",
        "callbackReference": {
          "callbackData": "1234",
          "notifyURL": "http://application.example.com/notifications/presenceNotification"
        },
        "clientCorrelator": "321",
        "duration": "5246",
        "presentityUserId": "tel:+19585550100",
```

```

    "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001"
  },
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions"
},
"resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions",
"watchersSubscriptionList": {
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/watchersSubscriptions",
  "watchersSubscription": {
    "applicationTag": "myApp",
    "callbackReference": {
      "callbackData": "3456",
      "notifyURL": "http://application.example.com/notifications/watchersNotification"
    },
    "clientCorrelator": "543",
    "duration": "2413",
    "presentityUserId": "tel:+19585550100",
    "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/watchersSubscriptions/sub003"
  }
}
}
}}

```

D.39 Retrieving all Watchers subscriptions (section 6.18.3.1)

Request:

```

GET /exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions HTTP/1.1
Host: example.com
Accept: application/json

```

Response:

```

HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"watchersSubscriptionList": {
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions",
  "watchersSubscription": [
    {
      "applicationTag": "myApp",
      "callbackReference": {
        "callbackData": "1234",
        "notifyURL": "http://application.example.com/notifications/watchersNotification"
      },
      "clientCorrelator": "321",
      "duration": "5246",
      "presentityUserId": "tel:+19585550100",
      "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/tel%3A%2B19585550100/sub001"
    },
    {

```



```

    "applicationTag": "myApp",
    "callbackReference": {
      "callbackData": "4321",
      "notifyURL": "http://application.example.com/notifications/watchersNotification"
    },
    "clientCorrelator": "123",
    "duration": "5237",
    "presentityUserId": "tel:+19585550100",
    "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/tel%3A%2B19585550100/sub002"
  }
}
}}

```

D.40 Creating new Watchers subscription, using tel URI (section 6.18.5.1)

Request:

```

POST /exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"watchersSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/watchersNotification"
  },
  "clientCorrelator": "321",
  "duration": "7200",
  "presentityUserId": "tel:+19585550100"
}}

```

Response:

```

HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"watchersSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/watchersNotification"
  },
  "clientCorrelator": "321",
  "duration": "3600",
  "presentityUserId": "tel:+19585550100",
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001"
}}

```

```
}}
```

D.41 Creating new Watchers subscription, using ACR (section 6.18.5.1)

Request:

```
POST /exampleAPI/presence/v1/acr%3A pseudonym123/subscriptions/watchersSubscriptions HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"watchersSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/watchersNotification"
  },
  "clientCorrelator": "321",
  "duration": "7200",
  "presentityUserId": "acr%3A pseudonym123"
}}
```

Response:

```
HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/presence/v1/acr%3A pseudonym123/subscriptions/watchersSubscriptions/sub001
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"watchersSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/watchersNotification"
  },
  "clientCorrelator": "321",
  "duration": "3600",
  "presentityUserId": "acr%3A pseudonym123",
  "resourceURL": "http://example.com/exampleAPI/presence/v1/acr%3A pseudonym123/subscriptions/watchersSubscriptions/sub001"
}}
```

D.42 Retrieving individual Watchers subscription (section 6.19.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001 HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"watchersSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/watchersNotification"
  },
  "clientCorrelator": "321",
  "duration": "5246",
  "presentityUserId": "tel:+19585550100",
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001"
}}
```

D.43 Updating individual Watchers subscription (section 6.19.4.1)

Request:

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001 HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"watchersSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/watchersNotification"
  },
  "clientCorrelator": "321",
  "duration": "7200",
  "presentityUserId": "tel:+19585550100"
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001"
}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"watchersSubscription": {
```

```

"applicationTag": "myApp",
"callbackReference": {
  "callbackData": "1234",
  "notifyURL": "http://application.example.com/notifications/watchersNotification"
},
"clientCorrelator": "321",
"duration": "7200",
"presentityUserId": "tel:+19585550100",
"resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001"
}}

```

D.44 Terminating individual Watchers subscription (section 6.19.6.1)

Request:

```

DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001 HTTP/1.1
Host: example.com

```

Response:

```

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

D.45 Notifying Presentity about change in Watchers status (section 6.20.5.1)

Request:

```

POST /notifications/watchersNotification HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"watchersNotification": {
  "callbackData": "1234",
  "link": {
    "href": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001",
    "rel": "WatchersSubscription"
  },
  "presentityUserId": "tel:+19585550100",
  "resourceStatus": "Active",
  "watcherList": {
    "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001",
    "watcher": {
      "displayName": "Bob",
      "resourceStatus": "Pending",

```

```
"resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001",
  "watcherUserId": "tel:+19585550101"
}
}
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.46 Notifying Presentity about subscription time out (section 6.20.5.2)

Request:

```
POST /notifications/watchersNotification HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"watchersNotification": {
  "callbackData": "1234",
  "link": {
    "href": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001",
    "rel": "WatchersSubscription"
  },
  "presentityUserId": "tel:+19585550100",
  "resourceStatus": "TerminatedTimeout"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.47 Notifying Presentity about termination of Watchers subscription (reason unknown) (section 6.20.5.3)

Request:

```
POST /notifications/watchersNotification HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn
```

```
{
  "watchersNotification": {
    "callbackData": "1234",
    "link": {
      "href": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/watchersSubscriptions/sub001",
      "rel": "WatchersSubscription"
    },
    "presentityUserId": "tel:+19585550100",
    "resourceStatus": "TerminatedOther"
  }
}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.48 Retrieving all presence subscriptions for all Presentities (section 6.21.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"presenceSubscriptionList": {
  "presenceSubscription": [
    {
      "applicationTag": "myApp",
      "callbackReference": {
        "callbackData": "1234",
        "notifyURL": "http://application.example.com/notifications/presenceNotification"
      },
      "clientCorrelator": "321",
      "duration": "5246",
      "presentityUserId": "tel:+19585550100",
      "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001"
    },
    {
      "applicationTag": "myApp",
      "callbackReference": {
        "callbackData": "4321",
        "notifyURL": "http://application.example.com/notifications/presenceNotification"
      },
      "clientCorrelator": "123",
      "duration": "5237",
      "presentityUserId": "tel:+19585550100",
```

```

    "resourceURL":
    "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550101/sub004"
  }
],
"resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions"
}}

```

D.49 Retrieving all presence subscriptions for Presentity (section 6.22.3.1)

Request:

```

GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100 HTTP/1.1
Host: example.com
Accept: application/json

```

Response:

```

HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"presenceSubscriptionList": {
  "presenceSubscription": [
    {
      "applicationTag": "myApp",
      "callbackReference": {
        "callbackData": "1234",
        "notifyURL": "http://application.example.com/notifications/presenceNotification"
      },
      "clientCorrelator": "321",
      "duration": "5246",
      "presentityUserId": "tel:+19585550100",
      "resourceURL":
      "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001"
    },
    {
      "applicationTag": "myApp",
      "callbackReference": {
        "callbackData": "6789",
        "notifyURL": "http://application2.example.com/notifications/presenceNotification"
      },
      "clientCorrelator": "987",
      "duration": "4132",
      "presentityUserId": "tel:+19585550100",
      "resourceURL":
      "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub005"
    }
  ],
  "resourceURL":
  "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100"
}

```

```
}}
```

D.50 Creating new presence subscription for Presentity (section 6.22.5.1)

Request:

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100 HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/presenceNotification"
  },
  "clientCorrelator": "321",
  "duration": "7200",
  "frequency": "600"
}}
```

Response:

```
HTTP/1.1 201 Created
Location:
http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceSubscriptions/tel%3A%2B19585550100/sub001
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"presenceSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/presenceNotification"
  },
  "clientCorrelator": "321",
  "duration": "7200",
  "frequency": "600",
  "presentityUserId": "tel:+19585550100",
  "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001"
}}
```


D.51 Creating new presence subscription for unknown Presentity (section 6.22.5.2)

Request:

```
POST /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100 HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/presenceNotification"
  },
  "clientCorrelator": "321",
  "duration": "7200",
  "frequency": "600"
}}
```

Response:

```
HTTP/1.1 201 Created
Location:
http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceSubscriptions/tel%3A%2B19585550100/sub001
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"requestError": {
  "link": {
    "href": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/presenceSources",
    "rel": "PresenceSourceList"
  },
  "serviceException": {
    "messageId": "SVC0004",
    "text": "No valid addresses provided in message part %1",
    "variables": "The specified identity does not exists."
  }
}}
```

D.52 Retrieving individual presence subscription (section 6.23.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001
HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"presenceSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/presenceNotification"
  },
  "clientCorrelator": "321",
  "duration": "5246",
  "frequency": "600",
  "presentityUserId": "tel:+19585550100",
  "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001"
}}
```

D.53 Updating individual presence subscription (section 6.23.4.1)

Request:

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001
HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/presenceNotification"
  },
  "clientCorrelator": "321",
  "duration": "7200",
  "frequency": "600",
  "presentityUserId": "tel:+19585550100",
  "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001"
}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"presenceSubscription": {
```

```

"applicationTag": "myApp",
"callbackReference": {
  "callbackData": "1234",
  "notifyURL": "http://application.example.com/notifications/presenceNotification"
},
"clientCorrelator": "321",
"duration": "7200",
"frequency": "600",
"presentityUserId": "tel:+19585550100",
"resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001"
}}

```

D.54 Terminating individual presence subscription (section 6.23.6.1)

Request:

```

DELETE
/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001HTTP/1.1
Host: example.com

```

Response:

```

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

D.55 Notifying Watcher about presence information updates from an active subscription (section 6.24.5.1)

Request:

```

POST /notifications/presenceNotification HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceNotification": {
  "callbackData": "1234",
  "link": {
    "href":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/sub001",
    "rel": "PresenceSubscription"
  },
  "presence": {"person": {"mood": {"moodValue": "Happy"}}},
  "presentityUserId": "tel:+19585550100",

```

```
"resourceStatus": "Active"  
}}
```

Response:

```
HTTP/1.1 204 No Content  
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.56 Notifying Watcher about presence information updates from pending subscription (section 6.24.5.2)

Request:

```
POST /notifications/presenceNotification HTTP/1.1  
Host: example.com  
Content-Type: application/json  
Accept: application/json  
Content-Length: nnnn  
  
{  
  "presenceNotification": {  
    "callbackData": "2345",  
    "link": {  
      "href":  
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550101/s  
ub002",  
      "rel": "PresenceSubscription"  
    },  
    "presentityUserId": "tel:+19585550100",  
    "resourceStatus": "Pending"  
  }  
}
```

Response:

```
HTTP/1.1 204 No Content  
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.57 Notifying Watcher about termination of presence subscription (reason unknown) (section 6.24.5.3)

Request:

```
POST /notifications/presenceNotification HTTP/1.1  
Host: example.com  
Content-Type: application/json  
Accept: application/json  
Content-Length: nnnn  
  
{  
  "presenceNotification": {  
    "callbackData": "1234",
```

```
"link": {
  "href":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/
sub001",
  "rel": "PresenceSubscription"
},
"presentityUserId": "tel:+19585550100",
"resourceStatus": "TerminatedOther"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.58 Notifying Watcher about termination of presence subscription (Watcher blocked) (section 6.24.5.4)

Request:

```
POST /notifications/presenceNotification HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceNotification": {
  "callbackData": "1234",
  "link": {
    "href":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceSubscriptions/tel%3A%2B19585550100/s
ub001",
    "rel": "PresenceSubscription"
  },
  "presentityUserId": "tel:+19585550100",
  "resourceStatus": "TerminatedBlocked"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.59 Retrieving all Presence List subscriptions towards all Presence Lists (section 6.25.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions HTTP/1.1
```

Host: example.com
Accept: application/json

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"presenceListSubscriptionCollection": {
  "presenceListSubscription": [
    {
      "applicationTag": "myApp",
      "callbackReference": {
        "callbackData": "1234",
        "notifyURL": "http://application.example.com/notifications/presenceListNotification"
      },
      "clientCorrelator": "321",
      "duration": "5246",
      "presenceListId": "myFriends",
      "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001"
    },
    {
      "applicationTag": "myApp",
      "callbackReference": {
        "callbackData": "4321",
        "notifyURL": "http://application.example.com/notifications/presenceListNotification"
      },
      "clientCorrelator": "123",
      "duration": "5237",
      "presenceListId": "myColleagues",
      "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myColleagues/sub002"
    }
  ],
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions"
}}
```

D.60 Retrieving all Presence List subscriptions towards a single Presence List (section 6.26.3.1)

Request:

GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends HTTP/1.1
Host: example.com
Accept: application/json

Response:

HTTP/1.1 200 OK

```

Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"presenceListSubscriptionCollection": {
  "presenceListSubscription": {
    "applicationTag": "myApp",
    "callbackReference": {
      "callbackData": "1234",
      "notifyURL": "http://application.example.com/notifications/presenceListNotification"
    },
    "clientCorrelator": "321",
    "duration": "5246",
    "presenceListId": "myFriends",
    "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001"
  },
  "resourceURL": "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions"
}}

```

D.61 Creating new Presence List subscription towards a single Presence List (section 6.26.5.1)

Request:

```

POST /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceListSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/presenceListNotification"
  },
  "clientCorrelator": "321",
  "duration": "7200",
  "frequency": "600"
  "presenceFilter": [
    "person/mood",
    "service/org.openmobilealliance:IM-Session/1.0"
  ],
}}

```

Response:

```

HTTP/1.1 201 Created
Location:
http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001
Date: Thu, 04 Jun 2009 02:51:59 GMT

```

```
Content-Type: application/json
Content-Length: nnnn

{"presenceListSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/presenceListNotification"
  },
  "clientCorrelator": "321",
  "duration": "7200",
  "frequency": "600",
  "presenceFilter": [
    "person/mood",
    "service/org.openmobilealliance:IM-Session/1.0"
  ],
  "presenceListId": "myFriends",
  "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001"
}}
```

D.62 Retrieving individual Presence List subscription (section 6.27.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001 HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"presenceListSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/presenceListNotification"
  },
  "clientCorrelator": "321",
  "duration": "5274",
  "frequency": "600",
  "presenceFilter": [
    "person/mood",
    "service/org.openmobilealliance:IM-Session/1.0"
  ],
  "presenceListId": "myFriends",
  "resourceURL":
```



```
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001"
}}
```

D.63 Updating individual Presence List subscription (section 6.27.4.1)

Request:

```
PUT /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001 HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceListSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/presenceListNotification"
  },
  "clientCorrelator": "321",
  "duration": "7200",
  "frequency": "600",
  "presenceFilter": [
    "person/mood",
    "service/org.openmobilealliance:IM-Session/1.0"
  ],
  "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001"
}}
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"presenceListSubscription": {
  "applicationTag": "myApp",
  "callbackReference": {
    "callbackData": "1234",
    "notifyURL": "http://application.example.com/notifications/presenceListNotification"
  },
  "clientCorrelator": "321",
  "duration": "7200",
  "frequency": "600",
  "presenceFilter": [
    "person/mood",
    "service/org.openmobilealliance:IM-Session/1.0"
  ],
  "presenceListId": "myFriends",
  "resourceURL":
```

```
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001"
}}
```

D.64 Terminating individual Presence List subscription (section 6.27.6.1)

Request:

```
DELETE /exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001 HTTP/1.1
Host: example.com
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.65 Notifying Watcher about presence information updates relating to Presence List (section 6.28.5.1)

Request:

```
POST /notifications/presenceListNotification HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceListNotification": {
  "callbackData": "1234",
  "link": {
    "href":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/presenceListSubscriptions/myFriends/sub001",
    "rel": "PresenceListSubscription"
  },
  "presenceList": {
    "presenceContact": [
      {
        "presence": {"person": {"mood": {"moodValue": "Happy"}}},
        "presentityUserId": "tel:+19585550101",
        "resourceStatus": "Active",
        "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/presenceListSubscription/myFriends/sub001"
      },
      {
        "presentityUserId": "tel:+19585550102",
        "resourceStatus": "Pending",
        "resourceURL":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/presenceListSubscription/myFriends/sub001"
      }
    ]
  }
}
```

```
    ],  
    "resourceURL":  
    "http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550100/subscriptions/presenceListSubscription/myFriends/sub001"  
  },  
  "presenceListId": "myFriends",  
  "resourceStatus": "Active"  
}}
```

Response:

```
HTTP/1.1 204 No Content  
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.66 Notifying Watcher about termination of Presence list subscription (No resource) (section 6.28.5.2)

Request:

```
POST /notifications/presenceListNotification HTTP/1.1  
Host: example.com  
Content-Type: application/json  
Accept: application/json  
Content-Length: nnnn  
  
{  
  "presenceListNotification": {  
    "callbackData": "1234",  
    "presenceListId": "myFriends",  
    "resourceStatus": "TerminatedNoResource"  
  }  
}
```

Response:

```
HTTP/1.1 204 No Content  
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.67 Notifying Watcher about termination of presence subscription (reason unknown) (section 6.28.5.3)

Request:

```
POST /notifications/presenceListNotification HTTP/1.1  
Host: example.com  
Content-Type: application/json  
Accept: application/json  
Content-Length: nnnn  
  
{  
  "presenceListNotification": {  
    "callbackData": "1234",  
    "link": {
```

```
"href":
"http://example.com/exampleAPI/presence/v1/tel%3A%2B19585550101/subscriptions/presenceListSubscriptions/myFriends/sub001",
"rel": "PresenceListSubscription"
},
"presenceListId": "myFriends",
"resourceStatus": "TerminatedOther"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.68 Notifying Watcher about subscription time out (section 6.28.5.4)

Request:

```
POST /notifications/presenceListNotification HTTP/1.1
Host: example.com
Content-Type: application/json
Accept: application/json
Content-Length: nnnn

{"presenceListNotification": {
  "callbackData": "1234",
  "link": {
    "href": "http://example.com/exampleAPI/presence/v1/{userId}/subscriptions/presenceListSubscriptions/myFriends/sub001",
    "rel": "PresenceListSubscription"
  },
  "presenceListId": "myFriends",
  "resourceStatus": "TerminatedTimeout"
}}
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

D.69 Retrieving portrait icon by Presentity (section 6.29.3.1)

Request:

```
GET /exampleAPI/presence/v1/tel%3A%2B19585550100/content/portraitIcon HTTP/1.1
Host: example.com
Accept: image/jpeg
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Jun 2009 02:51:59 GMT
```

Content-Type: image/jpeg
Content-Length: nnnn

data

D.70 Uploading/updating of portrait icon and setting the link to the icon as presence information (section 6.29.4.1)

Request:

PUT /exampleAPI/presence/v1/tel%3A%2B19585550100/content/portraitIcon HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: image/jpeg
Content-Length: nnnn

data

Response:

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

D.71 Removing portrait icon by Presentity (section 6.29.6.1)

Request:

DELETE /exampleAPI/presence/v1/tel%3A%2B19585550100/content/portraitIcon HTTP/1.1
Host: example.com

Response:

HTTP/1.1 204 No Content
Date: Thu, 04 Jun 2009 02:51:59 GMT

Appendix E. Parlay X operations mapping (Informative)

The table below illustrates the mapping between REST resources/methods and Parlay X [3GPP 29.199-14] equivalent operations.

REST Resource	REST Method	REST Section reference	Parlay X equivalent operation
Presence Sources	POST	6.1.5	publish
Individual Presence Source	PUT	6.2.4	publish
Individual Presence Source attribute	PUT	6.3.4	publish
Watchers list	GET	6.8.3	getMyWatchers
Individual Watcher	GET	6.9.3	getSubscribedAttributes
Authorization rules	POST	6.10.5	updateAuthorizationRule
Individual authorization rule	PUT	6.11.4	updateAuthorizationRule
	DELETE	6.11.6	deleteAuthorizationRule
Presence information by Watcher	GET	6.13.3	getUserPresence
Individual presence attribute by Watcher	GET	6.14.3	getUserPresence
Presence information by Watcher for Presence List	GET	6.15.3	getUserPresence
All Watchers subscriptions	POST	6.18.5	startMyWatcherNotification
Individual Watchers subscription	PUT	6.19.4	startMyWatcherNotification
	DELETE	6.19.6	endMyWatchersNotification
Watchers notification	POST	6.20.5	notifyMyWatchers, notifyMyWatchersEnd, NotifyError
Presence subscriptions for a single Presentity	POST	6.22.5	startPresenceNotification
Individual presence subscription	PUT	6.23.4	startPresenceNotification
	DELETE	6.23.6	endPresenceNotification
Presence notification	POST	6.24.5	statusNotified, statusEnd, subscriptionEnded
Presence List subscriptions for a single Presence List	POST	6.26.5	startPresenceNotification
Individual Presence List subscription	PUT	6.27.4	startPresenceNotification
	DELETE	6.27.6	endPresenceNotification
Presence List notification	POST	6.28.5	statusNotified, statusEnd, subscriptionEnded

Table 1: Parlay X operations mapping

Appendix F. Light-weight Resources (Informative)

The following table lists all presence information structure elements that can be accessed individually as Light-weight Resources. For each Light-weight Resource there are listed: corresponding root element name, root element type and [ResourceRelPath] string.

Type of Light-weight Resources (and references to data structures)	Element/attribute that can be accessed as Light-weight Resource	Root element name for the Light-weight Resource	Root element type for the Light-weight Resource	[ResourceRelPath] string that needs to be appended to the corresponding Heavy-weight Resource URL
Presence Source information (5.2.2.2)	duration	duration	xsd:int	duration
Presence information (5.2.2.3)	person	person	PersonAttributes	person
	service	service	ServiceAttributes	service/{servicelId}/{version}
	device	device	DeviceAttributes	device/{devicelId}
Person attributes (5.2.2.4)	activities	activities	Activities	person/activities
	placeType	placeType	PlaceType	person/placeType
	privacy	privacy	Privacy	person/privacy
	sphere	sphere	Sphere	person/sphere
	mood	mood	Mood	person/mood
	placels	placels	Placels	person/placels
	timeOffset	timeOffset	TimeOffset	person/timeOffset
	statusIcon	statusIcon	StatusIcon	person/statusIcon
	class	class	xsd:token	person/class
	noteList	noteList	NoteList	person/noteList
	location	location	Location	person/location
	overridingWillingness	overridingWillingness	OverridingWillingness	person/overridingWillingness
	linkList	linkList	LinkList	person/linkList
	card	card	xsd:anyURI	person/card
	displayName	displayName	xsd:string	person/displayName
homePage	homePage	xsd:anyURI	person/homePage	
icon	icon	xsd:anyURI	person/icon	
map	map	xsd:anyURI	person/map	

	sound	sound	xsd:anyURI	person/sound
	timestamp	timestamp	xsd:dateTime	person/timestamp
	extended	extended	ExtendedList	person/extended
Service attributes (5.2.2.5)	statusIcon	statusIcon	StatusIcon	service/{serviceld}/{version}/statusIcon
	class	class	xsd:token	service/{serviceld}/{version}/class
	displayName	displayName	xsd:string	service/{serviceld}/{version}/displayName
	homePage	homePage	xsd:anyURI	service/{serviceld}/{version}/homePage
	icon	icon	xsd:anyURI	service/{serviceld}/{version}/icon
	map	map	xsd:anyURI	service/{serviceld}/{version}/map
	sound	sound	xsd:anyURI	service/{serviceld}/{version}/sound
	linkList	linkList	LinkList	service/{serviceld}/{version}/linkList
	serviceAvailability	serviceAvailability	OpenOrClosed	service/{serviceld}/{version}/serviceAvailability
	serviceWillingness	serviceWillingness	OpenOrClosed	service/{serviceld}/{version}/serviceWillingness
	contact	contact	Contact	service/{serviceld}/{version}/contact
	sessionParticipation	sessionParticipation	OpenOrClosed	service/{serviceld}/{version}/sessionParticipation
	registrationState	registrationState	ActiveOrTerminated	service/{serviceld}/{version}/registrationState
	barringState	barringState	ActiveOrTerminated	service/{serviceld}/{version}/barringState
	sessionAnswerMode	sessionAnswerMode	AutomaticOrManual	service/{serviceld}/{version}/sessionAnswerMode
	devices	devices	DeviceIdentityList	service/{serviceld}/{version}/devices
	timestamp	timestamp	xsd:dateTime	service/{serviceld}/{version}/timestamp
extended	extended	ExtendedList	service/{serviceld}/{version}/extended	

Device attributes (5.2.2.6)	class	class	xsd:token	device/{deviceId}/class
	location	location	Location	device/{deviceId}/location
	networkAvailability	networkAvailability	NetworkAvailability	device/{deviceId}/networkAvailability
	timestamp	timestamp	xsd:dateTime	device/{deviceId}/timestamp
	extended	extended	ExtendedList	device/{deviceId}/extended
Rule data (5.2.2.12)	watcherUserId	watcherUserId	xsd:anyURI	watchers/{watcherUserId}
	memberListId	memberListId	xsd:anyURI	memberLists/{memberListId}
	domainName	domainName	xsd:string	domains/{domainName}

Table 2: Light-weight Resources for Presence

Note: When appending [ResourceRelPath] string to its Heavy-weight Resource URL, all variables within curly brackets “{}” such as: “serviceld”, “version”, “deviceld”, “watcherUserId”, “memberListId”, and “domainName” have to be replaced by their real values.

Appendix G. Authorization aspects (Normative)

This appendix specifies how to use the RESTful Presence API in combination with some authorization frameworks.

G.1 Use with OMA Authorization Framework for Network API

The RESTful Presence API MAY support the Autho4API authorization framework defined in [Autho4API_10].

A RESTful Presence API supporting Autho4API:

- SHALL conform to section D.1 of [REST_NetAPI_Common];
- SHALL conform to this section G.1.

G.1.1 Scope values

G.1.1.1 Definitions

In compliance with [Autho4API_10], an authorization server serving clients requests for getting authorized access to the resources exposed by the RESTful Presence API:

- SHALL support the scope values defined in Table 1 below;
- MAY support scope values not defined in this specification.

Scope value	Description	For one-time access token
oma_rest_presence.all_{apiVersion}	Provide access to all defined operations on the resources in this version of the API. The {apiVersion} part of this identifier SHALL have the same value as the “apiVersion” URL variable which is defined in section 5.1. This scope value is the union of the other scope values listed in next rows of this table.	No
oma_rest_presence.publish	Provide access to all defined operations for the owner of presence information (Presentity)	No
oma_rest_presence.auth	Provide access to all defined operations for access authorization to presence information	No
oma_rest_presence.watcher	Provide access to all defined operations for a Watcher	No

Table 3: Scope values for RESTful Presence API

G.1.1.2 Downscoping

In the case where the client requests authorization for “oma_rest_presence.all_{apiVersion}” scope, the authorization server and/or resource owner MAY restrict the granted scope to some of the following scope values:

- “oma_rest_presence.publish”
- “oma_rest_presence.auth”

- “oma_rest_presence.watcher”

G.1.1.3 Mapping with resources and methods

Tables in this section specify how the scope values defined in section G.1.1.1 for the RESTful Presence API map to the REST resources and methods of this API. In these tables, the root “oma_rest_presence.” of scope values is omitted for readability reasons.

Resource	URL Base URL: http://{serverRoot}/presence/{apiVersion}	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
Presence Sources	/ {userId}/presenceSources	6.1	all_{apiVersion} or publish	n/a	all_{apiVersion} or publish	n/a
Individual Presence Source	/ {userId}/presenceSources/{presenceSourceId}	6.2	all_{apiVersion} or publish	all_{apiVersion} or publish	n/a	all_{apiVersion} or publish
Individual Presence Source attribute	/ {userId}/presenceSources/{presenceSourceId}/[ResourceRelPath]	6.3	all_{apiVersion} or publish	all_{apiVersion} or publish	n/a	all_{apiVersion} or publish
Persistent Presence Source	/ {userId}/presenceSources/persistent	6.4	all_{apiVersion} or publish	all_{apiVersion} or publish	n/a	all_{apiVersion} or publish
Individual persistent Presence Source attribute	/ {userId}/presenceSources/persistent/[ResourceRelPath]	6.5	all_{apiVersion} or publish	all_{apiVersion} or publish	n/a	all_{apiVersion} or publish
Presentity content list	/ {userId}/content	6.6	all_{apiVersion} or publish	n/a	n/a	n/a
Individual Presentity content	/ {userId}/content/{contentId}	6.7	all_{apiVersion} or publish	all_{apiVersion} or publish	n/a	all_{apiVersion} or publish
Presentity portrait icon	/ {userId}/content/portrait/icon	6.29	all_{apiVersion} or publish	all_{apiVersion} or publish	n/a	all_{apiVersion} or publish

Table 4: Required scope values for: Management of presence information on behalf of Presentity

Resource	URL Base URL: http://{serverRoot}/{apiVersion}/ presence	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
Watchers list	/userId/watchers	6.8	all_{apiVersion} or publish	n/a	n/a	n/a
Individual Watcher	/userId/watchers/{watcherUserId}	6.9	all_{apiVersion} or publish	n/a	n/a	n/a

Table 5: Required scope values for: Retrieval of Watchers information by Presentity

Resource	URL Base URL: http://{serverRoot}/{apiVersion}/p resence	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
All subscriptions	/userId/subscriptions	6.17	all_{apiVersion} or publish	n/a	n/a	n/a
All Watchers subscriptions	/userId/subscriptions/watchersSub scriptions	6.18	all_{apiVersion} or publish	n/a	all_{apiVersion} or publish	n/a
Individual Watchers subscription	/userId/subscriptions/watchersSub scriptions/{subscriptionId}	6.19	all_{apiVersion} or publish	all_{apiVersion} or publish	n/a	all_{apiVersion} or publish

Table 6: Required scope values for: Management of subscriptions to notifications for Watchers information

Resource	URL Base URL: http://{serverRoot}/{apiVersion}/ presence	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
Authorization rules	/ {userId} / authorization / rules	6.10	all_{apiVersion} or auth	n/a	all_{apiVersion} or auth	n/a
Individual authorization rule	/ {userId} / authorization / rules / {ruleId}	6.11	all_{apiVersion} or auth	all_{apiVersion} or auth	n/a	all_{apiVersion} or auth
Individual authorization rule data	/ {userId} / authorization / rules / {ruleId} / [ResourceRelPath]	6.12	all_{apiVersion} or auth	all_{apiVersion} or auth	n/a	all_{apiVersion} or auth

Table 7: Required scope values for: Management of authorization rules for accessing presence information

Resource	URL Base URL: http://{serverRoot}/{apiVersion}/pr esence	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
Presence information by Watcher	/ {userId} / presenceContacts / {presentit yUserId}	6.13	all_{apiVersion} or watcher	n/a	n/a	n/a
Individual presence attribute by Watcher	/ {userId} / presenceContacts / {presentit yUserId} / [ResourceRelPath]	6.14	all_{apiVersion} or watcher	n/a	n/a	n/a
Presence information by Watcher for a Presence List	/ {userId} / presenceLists / {presenceList Id}	6.15	all_{apiVersion} or watcher	n/a	n/a	n/a
Content by Watcher	/ {userId} / PresenceContactsContent / { presentityUserId} / {contentId}	6.16	all_{apiVersion} or watcher	n/a	n/a	n/a

Table 8: Required scope values for: Retrieval of presence information by Watcher

Resource	URL Base URL: http://{serverRoot}/{apiVersion}/ presence	Section reference	HTTP verbs			
			GET	PUT	POST	DELETE
All subscriptions	/userId/subscriptions	6.17	all_{apiVersion} or watcher	n/a	n/a	n/a
All presence subscriptions	/userId/subscriptions/presenceSubscriptions	6.21	all_{apiVersion} or watcher	n/a	n/a	n/a
Presence subscriptions for a single Presence	/userId/subscriptions/presenceSubscriptions/{presenceUserId}	6.22	all_{apiVersion} or watcher	n/a	all_{apiVersion} or watcher	n/a
Individual presence subscription	/userId/subscriptions/presenceSubscriptions/{presenceUserId}/{subscriptionId}	6.23	all_{apiVersion} or watcher	all_{apiVersion} or watcher	n/a	all_{apiVersion} or watcher
All Presence List subscriptions	/userId/subscriptions/presenceListSubscriptions	6.25	all_{apiVersion} or watcher	n/a	n/a	n/a
Presence List subscriptions for a single Presence List	/userId/subscriptions/presenceListSubscriptions/{presenceListId}	6.26	all_{apiVersion} or watcher	n/a	all_{apiVersion} or watcher	n/a
Individual Presence List subscription	/userId/subscriptions/presenceListSubscriptions/{presenceListId}/{subscriptionId}	6.27	all_{apiVersion} or watcher	all_{apiVersion} or watcher	n/a	all_{apiVersion} or watcher

Table 9: Required scope values for: Management of subscriptions to notifications for presence information

G.1.2 Use of 'acr:auth'

This section specifies the use of 'acr:auth' in place of an end user identifier in a resource URL path.

An 'acr' URI of the form 'acr:auth', where 'auth' is a reserved keyword MAY be used to avoid exposing a real end user identifier in the resource URL path.

A client MAY use 'acr:auth' in a resource URL in place of the {userId} resource URL variable in the resource URL path, when the RESTful Presence API is used in combination with [Autho4API_10].

In the case the RESTful Presence API supports [Autho4API_10], the server:

- SHALL accept 'acr:auth' as a valid value for the resource URL variable {userId}.
- SHALL conform to [REST_Common_TS] section 5.8.1.1 regarding the processing of 'acr:auth'.