RESTful Network API for Anonymous Customer Reference Management

Approved Version 1.0 – 24 Dec 2013

Open Mobile Alliance

OMA-TS-REST_NetAPI_ACR-V1_0-20131224-A

Contents

Figures

Tables

1. Scope

This specification defines a RESTful API for Anonymous Customer Reference Management using HTTP protocol bindings.

2. References

2.1 Normative References

[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, January 2010, URL: http://tools.ietf.org/html/draft-uri-acr-extension-04

[RD-REST_NetAPI_ACR] "OMA RESTful Network API for Anonymous Customer Reference Management Requirements", Open Mobile Alliance™, OMA-RD-REST_NetAPI_ACR-V1_0, URL: http://www.openmobilealliance.org/

[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_SUP_ACR] "XML schema for the RESTful Network API for Anonymous Customer Reference Management", Open Mobile Alliance™, OMA-SUP-XSD_rest_netapi_acr-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.ietf.org/rfc/rfc3261.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

[RFC4627] "The application/json Media Type for JavaScript Object Notation (JSON)", D. Crockford, July 2006, URL: http://www.ietf.org/rfc/rfc4627.txt

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

[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/

[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

For the purpose of this TS, all definitions from the OMA Dictionary apply [OMADICT].

3.3 Abbreviations

ACR	Anonymous Customer Reference
API	Application Programming Interface
APP	Application
HTTP	HyperText Transfer Protocol
JSON	JavaScript Object Notation
MIME	Multipurpose Internet Mail Extensions
MSISDN	Mobile Subscriber ISDN Number
OMA	Open Mobile Alliance
REST	REpresentational State Transfer
SCR	Static Conformance Requirements
SIP	Session Initiation Protocol
TS	Technical Specification
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WP	White Paper
XML	eXtensible Markup Language
XSD	XML Schema Definition

4. Introduction

The Technical Specification of the RESTful Network API for Anonymous Customer Reference Management contains HTTP protocol bindings based on the requirements for Anonymous Customer Reference Management defined in [RD-REST_NetAPI_ACR], 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).

4.1 Version 1.0

Version 1.0 of this specification supports the following operations:

- Create an Anonymous Customer Reference (ACR)
- Query the status of an ACR
- Refresh an ACR

In addition, this specification provides:

- 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. Anonymous Customer Reference Management API definition

This section is organized to support a comprehensive understanding of the Anonymous Customer Reference (ACR) Management 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].

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 detailed specification for each of the resources. Each such subsection defines the resource, the request URL variables that are common for all HTTP methods, 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. 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 provides the operations mapping to a pre-existing baseline specification, 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 Anonymous Customer Reference Management.

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.

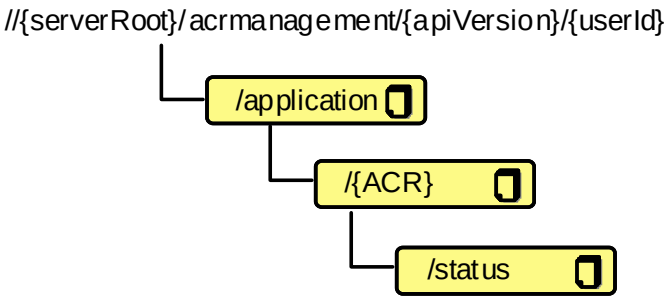

Figure 1 Resource structure defined by this specification

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: Operations on Anonymous Customer Reference

Resource	URL Base URL: http://{serverRoot}/acrmanagement/{apiVersion}/{userId}	Data Structures	HTTP verbs			
			GET	PUT	POST	DELETE
End user's application	/application	AcrList (used for GET response) Acr (used for POST)	Retrieves the issued ACR for the given end user identified by the {userId} in the context of a given application.	No	Creates an ACR for the user identified by the {userId}.	no
Anonymous customer reference	/application/{ACR}	Acr (used for GET response)	Retrieves the ACR data.	No	no	Remove an ACR.
ACR status	/application/{ACR}/status	Status	Retrieves the ACR status.	Refresh an "expired" ACR. (set the ACR status to "Valid")	no	no

5.2 Data Types

5.2.1 XML Namespaces

The XML namespace for the Anonymous Customer Reference Management data types is:

urn:oma:xml:rest:netapi:acrmanagement:1

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

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

5.2.2 Structures

The subsections of this section define the data structures used in the Anonymous Customer Reference Management API.

Some of the structures can be instantiated as so-called root elements.

5.2.2.1 Type: AcrList

This type represents a list of Anonymous Customer References.

Element	Type	Optional	Description
acr	Acr [0..unbounded]	Yes	A list of ACRs.
resourceURL	xsd:anyURI	No	Self referring URL.

A root element named acrList of type AcrList is allowed in response bodies.

5.2.2.2 Type: Acr

This type represents an individual Anonymous Customer Reference and associated metadata.

Element	Type	Optional	Description
value	xsd:anyURI	Yes	The string comprising the ACR, as per [IETF_ACR_draft]. It MUST be included in responses. “Dyna” and “Stat” are the values of the type field in the ACR value as per [IETF_ACR_draft]. A static ACR once created would never expire. It stays valid until it is revoked by either the user or the server. On the other hand a dynamic ACR is valid only for a certain period of time (as authorized by the user, requested by the application or determined by the server policy), after which it goes into an “expired” status. An expired ACR would need to be refreshed before it can be used again. An “expired” ACR after being dormant for a period of time may be removed by the server. Once a dormant-expired dynamic ACR is removed by the server, the application would need to request the creation of a brand new ACR (hence the name Dynamic as the ACR value may change if not used/refreshed in a

			timely manner). The value of expiry parameter as explained below determines whether the ACR is a dynamic or static ACR.
acrStatus	AcrStatus	Yes	Status of the ACR. It MUST be included in responses, if the ACR exists.
expiry	xsd:dateTime	Yes	The time at which the ACR will expire. This parameter MUST be included in responses if ACR is dynamic. However, this parameter may be excluded from responses for static (non-temporary) ACRs which means that the ACR shall not expire. If expiry parameter for static ACR is included in the responses, it shall have the value of “0001-01-01T00:00:00”. For Dynamic ACRs expiry date/time may be requested or determined in the following ways: • user authorizing a expiry date/time through OAuth access token which may be present in the request header (see [Autho4API_10]). • application indicating a expiry date/time in the API request via this (“expiry”) parameter. • server’s policy. In API requests if OAuth access token specifies an expiry date/time for the ACR then it has precedence over the application’s requested expiry date/time (i.e. “expiry” parameter in the request is ignored). Requested expiry date/time may be overridden by the server. As a result, the actual ACR expiry date/time is returned in the responses to the ACR creation requests. Expiry date/time value of “0001-01-01T00:00:00” in the requests (either through OAuth access token or the “expiry” parameter), would mean the request is for a Static ACR (i.e. the ACR should never expire). An expired ACR needs to be refreshed before it can be used again. How long an expired ACR would be kept around on the server’s side before it is removed is determined by server’s policy and out side of the scope of this document.
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 acr of type Acr is allowed in request and/or response bodies.

5.2.2.3 Type: Status

This type represents the status of an Anonymous Customer Reference.

Element	Type	Optional	Description
acrStatus	AcrStatus	Yes	Status of the ACR.It MUST be included in responses, if the ACR exists.
resourceURL	xsd:anyURI	No	Self referring URL

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

5.2.3 Enumerations

The subsections of this section define the enumerations used in the Anonymous Customer Reference Management API.

5.2.3.1 Enumeration: AcrStatus

This enumeration defines possible values to describe the status of the validity of an ACR.

Enumeration	Description
Valid	Indicates that the ACR is recognised by the server and may be used as the userId part within other requests managed by that server.
Revoked	The ACR has been revoked by the server, and may not be used as the userId part of other requests. A new ACR must be obtained for that MSISDN/User.
Expired	The ACR has expired, and a request can be made to refresh it. An expired ACR will have to be refreshed (see section 6.3.4) before it can be used as part of any other API request.

5.2.4 Values of the Link “rel” attribute

This specification does not define any elements of type Link.

5.3 Sequence Diagrams

The following subsections describe the resources, methods and steps involved in typical scenarios.

5.3.1 Operations on Anonymous Customer References

This figure below shows a scenario for retrieving, issuing and managing ACRs and ACR data.

The resources:

- To retrieve the issued ACR for the given end user identified by the {userId} in the context of a given application, read resource under

  http://{serverRoot}/acrmanagement/{apiVersion}/{userId}/application

- To issue (create) an ACR for the user identified by the {userId}, create resource under

  http://{serverRoot}/acrmanagement/{apiVersion}/{userId}/application

- To retrieve ACR data , read resource under

  http://{serverRoot}/acrmanagement/{apiVersion}/{userId}/application/{ACR}

- To remove an ACR, delete resource under

http://{serverRoot}/acrmanagement/{apiVersion}/{userId}/application/{ACR}

- To retrieve the ACR status, read resource under

  http://{serverRoot}/acrmanagement/{apiVersion}/{userId}/application/{ACR}/status

- To refresh an "expired" ACR (i.e. set the ACR status to "valid"), update resource under

  http://{serverRoot}/acrmanagement/{apiVersion}/{userId}/application/{ACR}/status

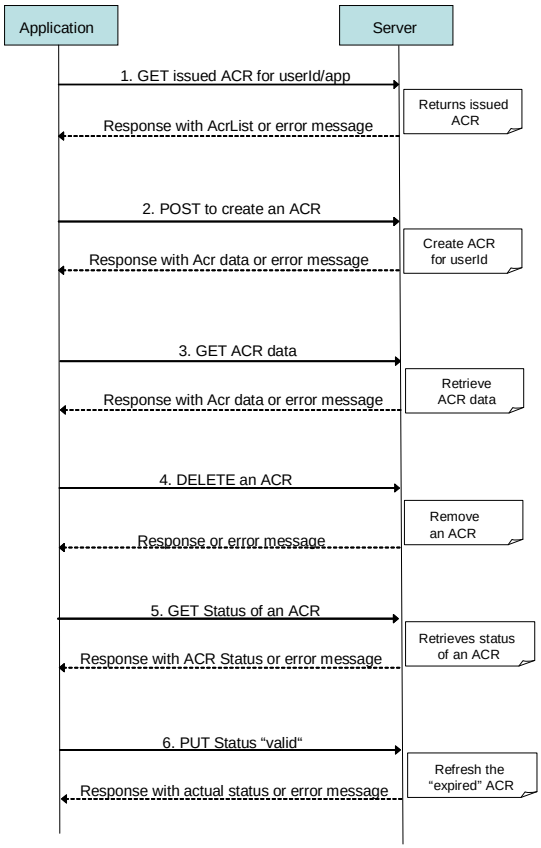

Figure 2 Management of own service capabilities

1. An application requests the ACR for the given user using GET method and receives the already issued application-specific ACR.
2. An application requests creation of an application-specific ACR for the given user identified by the userId using POST method and receives the resulting ACR.
3. An application requests the ACR data (value, status, expiry) using GET method and receives the ACR data.
4. An application deletes (removes) an ACR with data by using DELETE method and receives response with the result of operation.
5. An application requests the status of an ACR using GET method and receives the actual status of the ACR.
6. An application refreshes an "expired" ACR (i.e. sets the ACR status to "valid") by using PUT method and receives response with updated ACR status.

6. Detailed specification of the resources

The ACR Management API amongst other operations enables an application to create an ACR resource which in fact is an application-specific user identity of the user on whose behalf the application is acting. To create an ACR, the user's identity has to be known to the server through {userId} part of the resource URL. The following values are possible for the {userId}:

1) an identifier such as MSISDN
2) the "acr:auth" keyword.

When the {userId} is set to "acr:auth" keyword, it implies user's identity is known either through the OAuth access token (see [Autho4API_10]) available in the HTTP Authorization header of the API request or if the user's device is connected to the mobile network then MSISDN is known by the network and it is not necessary to be passed explicitly by the application in the API request.

The following applies to all resources defined in this specification regardless of the representation format (i.e. XML and JSON):

- 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 Appendix A.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. 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: End User's Application

The resource used is:

http://{serverRoot}/acrmanagement/{apiVersion}/{userId}/application

This resource is used for retrieving an already issued application-specific ACR and for creating an application-specific ACR.

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	User identifier. Examples: tel:+19585550100, acr:auth

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 Anonymous Customer Reference Management, see section 7.

6.1.3 GET

This operation is used for retrieving an already issued application-specific ACR. List returned may contain more than one ACR. For instance the list may contain a revoked ACR and an active ACR. It is assumed that the server knows how to determine which ACRs belong to a given application and appropriately list them in the response, through methods which are beyond the scope of this document (e.g. using the access token provided in the HTTP request header – see [Autho4API_10]).

As shown in the example below, GET operation enables a 3rd-party application the retrieval of an application-specific ACR over the RESTful API. Please note that, to support certain use cases, it might also be possible for the service provider to push an already created and valid ACR to the 3rd-party application, e.g. via HTTP header enrichment techniques (i.e. by injecting additional HTTP header fields carrying an ACR into the user's traffic). HTTP header enrichment with ACR is beyond the scope of this document.

6.1.3.1 Example 1: Retrieve the issued ACR using "auth" keyword (Informative)

6.1.3.1.1 Request

```
GET /exampleAPI/acrmanagement/v1/acr%3Aauth/application HTTP/1.1
Host: example.com
Accept: application/xml
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
```

6.1.3.1.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Oct 2012 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<cr:acrList xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <acr>
    <value>acr:abc123;ncc=23415;type=Stat</value>
    <acrStatus>Valid</acrStatus>

    <resourceURL>http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth
/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DStat</resourceURL>
  </acr>
  <resourceURL>http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application</resourceURL>
</cr:acrList>
```

6.1.3.2 Example 2: Retrieve the issued ACR using MSISDN (Informative)

6.1.3.2.1 Request

```
GET /exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application HTTP/1.1
Host: example.com
Accept: application/xml
```

6.1.3.2.2 Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Oct 2012 02:51:59 GMT
Content-Type: application/xml
Content-Length: nnnn

<?xml version="1.0" encoding="UTF-8"?>
<cr:acrList xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <acr>
    <value>acr:abc123;ncc=23415;type=Dyna</value>
    <acrStatus>Valid</acrStatus>
    <expiry>2013-10-26T21:32:52</expiry>

<resourceURL>http://example.com/exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application/acr%3Aabc123%3Bncc%3D
23415%3Btype%3DDyna</resourceURL>
  </acr>
  <resourceURL>http://example.com/exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application</resourceURL>
</cr:acrList>
```

6.1.3.3 Example 3: Retrieve a non-existent ACR using MSISDN (Informative)

6.1.3.3.1 Request

```
GET /exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application HTTP/1.1
Host: example.com
Accept: application/xml
```

6.1.3.3.2 Response

```
HTTP/1.1 404 Not Found
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Oct 2012 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
 <serviceException>
  <messageId>SVC1006</messageId>
  <text>ACR not found</text>
 </serviceException>
</common:requestError>
```

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 an ACR. The “expiry” value which may be provided by the OAuth access token in the HTTP header (see [Autho4API_10]) or the body of the POST, would determine the type of the ACR (Dynamic or Static) requested.

Please note that, the inclusion of “expiry” value as part of the OAuth access token in order to ensure user’s authorization (as shown in Example 1 below) would require the usage of “subscoping” technique as defined in [Autho4API_10]). Please also note that, expiry value may also be present in the request body (as shown in example 1 below) as per aforementioned

"subscoping" technique however, as stated in section 5.2.2.2 the value of expiry value in the OAuth access token takes precedence (if one is available).

6.1.5.1 Example 1: Create an ACR using "auth" keyword with expiry value included in OAuth access token (Informative)

This example illustrates the case of creating an ACR using "auth" keyword, where expiry value "0001-01-01T00:00:00" is included in OAuth access token

6.1.5.1.1 Request

```
POST /exampleAPI/acrmanagement/v1/acr%3Aauth/application HTTP/1.1
Accept: application/xml
Content-Length: nnnn
Content-Type: application/xml
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903

<?xml version="1.0" encoding="UTF-8"?>
<cr:acr xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <expiry>0001-01-01T00:00:00</expiry>
</cr:acr>
```

6.1.5.1.2 Response

```
HTTP/1.1 201 Created
Location:
http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DStat
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

<?xml version="1.0" encoding="UTF-8"?>
<cr:acr xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <value>acr:abc123;ncc=23415;type=Stat</value>
  <acrStatus>Valid</acrStatus>

<resourceURL>http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%
3DStat</resourceURL>
</cr:acr>
```

6.1.5.2 Example 2: Create an ACR using MSISDN (Informative)

6.1.5.2.1 Request

```
POST /exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application HTTP/1.1
Accept: application/xml
Content-Length: nnnn
Content-Type: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<cr:acr xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <expiry>2013-10-26T21:32:52</expiry>
</cr:acr>
```

6.1.5.2.2 Response

```
HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application/acr%3Aabc123%3B
ncc%3D23415%3Btype%3DDyna
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

<?xml version="1.0" encoding="UTF-8"?>
<cr:acr xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <value>acr:abc123;ncc=23415;type=Dyna</value>
  <acrStatus>Valid</acrStatus>
  <expiry>2013-10-26T21:32:52</expiry>

<resourceURL>http://example.com/exampleAPI/acrmanagement/v1/tel%2B%3A4479901234567/application/acr%3Aabc123%3Bncc%3D23415%
3Btype%3DDyna</resourceURL>
</cr:acr>
```

6.1.5.3 Example 3: Create an ACR over mobile network authenticated connection, response with a location of created resource (Informative)

6.1.5.3.1 Request

```
POST /exampleAPI/acrmanagement/v1/acr%3Aauth/application HTTP/1.1
Accept: application/xml
Content-Length: nnnn
Content-Type: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<cr:acr xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <expiry>2013-10-26T21:32:52</expiry>
</cr:acr>
```

6.1.5.3.2 Response

```
HTTP/1.1 201 Created
Location:
http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:resourceReference xmlns:common="urn:oma:xml:rest:netapi:common:1">
<resourceURL>http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%
3DDyna</resourceURL>
</common:resourceReference>
```

6.1.5.4 Example 4: Create ACR over mobile network authenticated connection, response with creation failure due to an existing expired ACR (Informative)

6.1.5.4.1 Request

```
POST /exampleAPI/acrmanagement/v1/acr%3Aauth/application HTTP/1.1
```

```
Accept: application/xml
Content-Length: nnnn
Content-Type: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<cr:acr xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <expiry>2013-10-26T21:32:52</expiry>
</cr:acr>
```

6.1.5.4.2 Response

```
HTTP/1.1 403 Forbidden
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Oct 2012 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <policyException>
   <messageId>POL1025</messageId>
   <text>An expired ACR, %1, already exists which needs to be refreshed prior to usage</text>
   <variables>abc123;ncc=23415;type=Dyna</variables>
 </policyException>
</common:requestError>
```

6.1.5.5 Example 5: Create an ACR over mobile network authenticated connection, response with a copy of created resource (Informative)

6.1.5.5.1 Request

```
POST /exampleAPI/acrmanagement/v1/acr%3Aauth/application HTTP/1.1
Accept: application/xml
Content-Length: nnnn
Content-Type: application/xml
Host: example.com

<?xml version="1.0" encoding="UTF-8"?>
<cr:acr xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <expiry>2013-10-26T21:32:52</expiry>
</cr:acr>
```

6.1.5.5.2 Response

```
HTTP/1.1 201 Created
Location:
http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

<?xml version="1.0" encoding="UTF-8"?>
<cr:acr xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <value>acr:abc123;ncc=23415;type=Dyna</value>
  <acrStatus>Valid</acrStatus>
```

```
  <expiry>2013-10-26T21:32:52</expiry>

<resourceURL>http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%
3DDyna</resourceURL>
</cr:acr>
```

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: Anonymous Customer Reference

The resource used is:

http://{serverRoot}/acrmanagement/{apiVersion}/{userId}/application/{ACR}

This resource is used for retrieving ACR data and for removing an ACR.

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	User identifier. Examples: tel:+19585550100, acr:auth
ACR	The string comprising the ACR, as per [IETF_ACR_draft]

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 Anonymous Customer Reference Management, see section 7.

6.2.3 GET

This operation is used for retrieving ACR data.

6.2.3.1 Example 1: Retrieve ACR data (Informative)

6.2.3.1.1 Request

```
GET /exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna
HTTP/1.1
Accept: application/xml
Host: example.com
```

6.2.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
```

```
Date: Thu, 26 Oct 2012 21:32:52 GMT

<?xml version="1.0" encoding="UTF-8"?>
<cr:acr xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <value>acr:abc123;ncc=23415;type=Dyna</value>
  <acrStatus>Valid</acrStatus>
  <expiry>2013-10-26T21:32:52</expiry>
<resourceURL>http://example.com/exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna</resourceURL>
</cr:acr>
```

6.2.4 PUT

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

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, DELETE]' field in the response as per section 14.7 of [RFC2616].

6.2.6 DELETE

This operation is used for removing an ACR.

6.2.6.1 Example 1: Remove an ACR (Informative)

6.2.6.1.1 Request

```
DELETE /exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DStat HTTP/1.1
Host: example.com
Accept: application/xml
```

6.2.6.1.2 Response

```
HTTP/1.1 204 No Content
Date: Thu, 26 Oct 2012 21:32:52 GMT
```

6.3 Resource: ACR Status

The resource used is:

http://{serverRoot}/acrmanagement/{apiVersion}/{userId}/application/{ACR}/status

This resource is used for retrieving and refreshing the status of an ACR.

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	User identifier. Examples: tel:+19585550100, acr:auth
ACR	The string comprising the ACR, as per [IETF_ACR_draft]

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

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 Anonymous Customer Reference Management, see section 7.

6.3.3 GET

This operation is used for retrieving the status of an ACR.

6.3.3.1 Example 1: Read the Status of an ACR using “auth” keyword (Informative)

6.3.3.1.1 Request

```
GET /exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%Btype%3DStat/status HTTP/1.1
Accept: application/xml
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
```

6.3.3.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

<?xml version="1.0" encoding="UTF-8"?>
<cr:status xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
  <acrStatus>Valid</acrStatus>
<resourceURL>http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%
3DDyna/status</resourceURL>
</cr:status>
```

6.3.3.2 Example 2: Read the Status of a non-existent ACR using “auth” keyword (Informative)

6.3.3.2.1 Request

```
GET /exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%Btype%3DStat/status HTTP/1.1
Accept: application/xml
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
```

6.3.3.2.2 Response

```
HTTP/1.1 404 Not Found
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Oct 2012 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <serviceException>
    <messageId>SVC1006</messageId>
    <text>ACR not found</text>
  </serviceException>
</common:requestError>
```

6.3.4 PUT

This operation is used for refreshing an expired ACR.

6.3.4.1 Example 1: Refresh an expired ACR by using “auth” keyword (Informative)

6.3.4.1.1 Request

```
PUT /exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna/status HTTP/1.1
Accept: application/xml
Content-Length: nnnn
Content-Type: application/xml
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903

<?xml version="1.0" encoding="UTF-8"?>
<cr:status xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
     <acrStatus>Valid</acrStatus>

<resourceURL>http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%
3DDyna/status</resourceURL>
</cr:status>
```

6.3.4.1.2 Response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

<?xml version="1.0" encoding="UTF-8"?>
<cr:status xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">

   <acrStatus>Valid</acrStatus>

<resourceURL>http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%
3DDyna/status</resourceURL>
</cr:status>
```

6.3.4.2 Example 2: Refreshing a revoked ACR is rejected (Informative)

6.3.4.2.1 Request

```
PUT /exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna/status HTTP/1.1
Accept: application/xml
Content-Length: nnnn
Content-Type: application/xml
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903

<?xml version="1.0" encoding="UTF-8"?>
<cr:status xmlns:cr="urn:oma:xml:rest:netapi:acrmanagement:1">
     <acrStatus>Valid</acrStatus>

<resourceURL>http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%
3DDyna/status</resourceURL>
</cr:status>
```

6.3.4.2.2 Response

```
HTTP/1.1 403 Forbidden
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Oct 2012 02:51:59 GMT

<?xml version="1.0" encoding="UTF-8"?>
<common:requestError xmlns:common="urn:oma:xml:rest:netapi:common:1">
  <policyException>
   <messageId>POL1027</messageId>
   <text>ACR, %1, is revoked. A new ACR is required to be created</text>
   <variables>abc123;ncc=23415;type=Dyna</variables>
 </policyException>
</common:requestError>
```

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]' field in the response as per section 14.7 of [RFC2616].

6.3.6 DELETE

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

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 Anonymous Customer Reference Management API.

7.1.1 SVC1005: ACR creation failed. Unknown userId

Name	Description
MessageID	SVC1005
Text	ACR creation operation failed. Unknown userId
Variables	None
HTTP status code(s)	403 Forbidden

7.1.2 SVC1006: ACR not found

Name	Description
MessageID	SVC1006
Text	ACR not found
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 Anonymous Customer Reference Management API.

7.2.1 POL1024: ACR creation failed. An active ACR already exists

Name	Description
MessageID	POL1024
Text	An active ACR, %1, already exists
Variables	%1 – ACR value
HTTP status code(s)	403 Forbidden

7.2.2 POL1025: ACR creation failed. An expired ACR already exists

Name	Description
MessageID	POL1025
Text	An expired ACR, %1, already exists which needs to be refreshed prior to usage

Variables	%1 – ACR value
HTTP status code(s)	403 Forbidden

7.2.3 POL1026: Creation of Static ACR is not supported

Name	**Description**
MessageID	POL1026
Text	Creation of Static ACR is not supported
Variables	None
HTTP status code(s)	403 Forbidden

7.2.4 POL1027: Revoked ACR can no longer be used or refreshed

Name	**Description**
MessageID	POL1027
Text	ACR, %1, is revoked. A new ACR is required to be created.
Variables	%1 – ACR value
HTTP status code(s)	403 Forbidden

7.2.5 POL1028: Expired ACR may not be used unless refreshed

Name	**Description**
MessageID	POL1028
Text	ACR, %1, is expired. It is required to be refreshed before it is used.
Variables	%1 – ACR value
HTTP status code(s)	403 Forbidden

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
OMA-TS-REST_NetAPI_ACR-V1_0-20131224-A	24 Dec 2013	Status changed to Approved by TP TP Ref # OMA-TP-2013-0394-INP_REST_NetAPI_ACR_V1_0_ERP_for_final_Approval

Appendix B. Static Conformance Requirements (Normative)

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

B.1 SCR for REST.ACRMgnt Server

Item	Function	Reference	Requirement
REST-ACRMGNT-SUPPORT-S-001-M	Support for the RESTful Anonymous Customer Reference Management API	5, 6	
REST-ACRMGNT-SUPPORT-S-002-M	Support for the XML request & response format	6	
REST-ACRMGNT-SUPPORT-S-003-M	Support for the JSON request & response format	6	

B.1.1 SCR for REST. ACRMgt.UsersApp Server

Item	Function	Reference	Requirement
REST- ACRMGNT-USERSAPP-S-001-M	Support for creation of an ACR and retrieval of an issued ACR	6.1	
REST- ACRMGNT-USERSAPP-S-002-M	Retrieve the issued ACR - GET	6.1.3	
REST- ACRMGNT-USERSAPP-S-003-M	Create an ACR – POST (XML or JSON)	6.1.5	

B.1.2 SCR for REST. ACRMgt.ACR Server

Item	Function	Reference	Requirement
REST- ACRMGNT-ACR-S-001-M	Support for removal of an ACR and retrieval of ACR data	6.2	
REST- ACRMGNT-ACR-S-002-M	Retrieve ACR data - GET	6.2.3	
REST- ACRMGNT-ACR-S-003-M	Remove an ACR – DELETE	6.2.6	

B.1.3 SCR for REST. ACRMgt.ACR.Status Server

Item	Function	Reference	Requirement
REST- ACRMGNT-ACRSTATUS-S-001-M	Support for refreshing expired ACRs and reading the status of an ACR	6.3	
REST- ACRMGNT-ACRSTATUS-S-002-M	Read the status of an ACR – GET	6.3.3	
REST- ACRMGNT-ACRSTATUS-S-003-M	Refresh an expired ACR – PUT	6.3.4	

Appendix C. Application/x-www-form-urlencoded Request Format for POST Operations (Normative)

This specification does not define any API request based on application/x-www-form-urlencoded MIME type.

Appendix D. JSON examples (Informative)

JSON (JavaScript Object Notation) is a light-weight, 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 REST 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 the JSON data format. 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 Retrieve the issued ACR using “auth” keyword (section 6.1.3.1)

Request:

```
GET /exampleAPI/acrmanagement/v1/acr%3Aauth/application HTTP/1.1
Host: example.com
Accept: application/json
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Oct 2012 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"acrList": {
    "acr": {
       "acrStatus": "Valid",
       "resourceURL": "http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth
/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DStat",
       "value": "acr:abc123;ncc=23415;type=Stat"
    },
    "resourceURL": "http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application"
}}
```

D.2 Retrieve the issued ACR using MSISDN (section 6.1.3.2)

Request:

```
GET /exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application HTTP/1.1
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 200 OK
Date: Thu, 04 Oct 2012 02:51:59 GMT
Content-Type: application/json
Content-Length: nnnn

{"acrList": {
    "acr": {
        "acrStatus": "Valid",
        "expiry": "2013-10-26T21:32:52",
        "resourceURL":
"http://example.com/exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application/acr%3Aabc123%3Bncc%3D23415%3Btype
%3DDyna",
        "value": "acr:abc123;ncc=23415;type=Dyna"
    },
    "resourceURL": "http://example.com/exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application"
}}
```

D.3 Retrieve a non-existent ACR using MSISDN (section 6.1.3.3)

Request:

```
GET /exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application HTTP/1.1
Host: example.com
Accept: application/xml
```

Response:

```
HTTP/1.1 404 Not Found
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Oct 2012 02:51:59 GMT

{"requestError": {"serviceException": {
    "messageId": "SVC1006",
    "text": "ACR not found"
}}}
```

D.4 Create an ACR using “auth” keyword with expiry value included in OAuth access token (section 6.1.5.1)

Request:

```
POST /exampleAPI/acrmanagement/v1/acr%3Aauth/application HTTP/1.1
Accept: application/json
Content-Length: nnnn
Content-Type: application/json
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903

{"acr": {"expiry": "0001-01-01T00:00:00"}}
```

Response:

```
HTTP/1.1 201 Created
Location: http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DStat
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

{"acr": {
    "acrStatus": "Valid",
    "resourceURL":
"http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DStat",
    "value": "acr:abc123;ncc=23415;type=Stat"
}}
```

D.5 Create an ACR using MSISDN (section 6.1.5.2)

Request:

```
POST /exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application HTTP/1.1
Accept: application/json
Content-Length: nnnn
Content-Type: application/json
Host: example.com

{"acr": {"expiry": "2013-10-26T21:32:52"}}
```

Response:

```
HTTP/1.1 201 Created
Location:
http://example.com/exampleAPI/acrmanagement/v1/tel%2B%3A4479901234567/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyn
a
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

{"acr": {
    "acrStatus": "Valid",
    "expiry": "2013-10-26T21:32:52",
    "resourceURL":
"http://example.com/exampleAPI/acrmanagement/v1/tel%2B%3A4479901234567/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyn
a",
    "value": "acr:abc123;ncc=23415;type=Dyna"
}}
```

D.6 Create an ACR over mobile network authenticated connection, response with a location of created resource (section 6.1.5.3)

Request:

```
POST /exampleAPI/acrmanagement/v1/acr%3Aauth/application HTTP/1.1
Accept: application/json
```

```
Content-Length: nnnn
Content-Type: application/json
Host: example.com

{"acr": {"expiry": "2013-10-26T21:32:52"}}
```

Response:

```
HTTP/1.1 201 Created
Location:
http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

{"resourceReference": {"resourceURL":
"http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna"}}
```

D.7 Create ACR over mobile network authenticated connection, response with creation failure due to an existing expired ACR (section 6.1.5.4)

Request:

```
POST /exampleAPI/acrmanagement/v1/acr%3Aauth/application HTTP/1.1
Accept: application/xml
Content-Length: nnnn
Content-Type: application/xml
Host: example.com

{"acr": {"expiry": "2013-10-26T21:32:52"}}
```

Response:

```
HTTP/1.1 403 Forbidden
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Oct 2012 02:51:59 GMT

{"requestError": {"policyException": {
    "messageId": "POL1025",
    "text": "An expired ACR, %1, already exists which needs to be refreshed prior to usage",
    "variables": "abc123;ncc=23415;type=Dyna"
}}}
```

D.8 Create an ACR over mobile network authenticated connection, response with a copy of created resource (section 6.1.5.5)

Request:

```
POST /exampleAPI/acrmanagement/v1/acr%3Aauth/application HTTP/1.1
```

```
Accept: application/xml
Content-Length: nnnn
Content-Type: application/xml
Host: example.com

{"acr": {"expiry": "2013-10-26T21:32:52"}}
```

Response:

```
HTTP/1.1 201 Created
Location:
http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

{"acr": {
    "acrStatus": "Valid",
    "expiry": "2013-10-26T21:32:52",
    "resourceURL":
"http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna",
    "value": "acr:abc123;ncc=23415;type=Dyna"
}}
```

D.9 Retrieve ACR data (section 6.2.3.1)

Request:

```
GET /exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna
HTTP/1.1
Accept: application/json
Host: example.com
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

{"acr": {
    "acrStatus": "Valid",
    "expiry": "2013-10-26T21:32:52",
    "resourceURL":
"http://example.com/exampleAPI/acrmanagement/v1/tel%3A%2B4479901234567/application/acr%3Aabc123%3Bncc%3D23415%3Btyp
e%3DDyna",
    "value": "acr:abc123;ncc=23415;type=Dyna"
}}
```

D.10 Remove an ACR (section 6.2.6.1)

Request:

```
DELETE /exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DStat HTTP/1.1
```

```
Host: example.com
Accept: application/json
```

Response:

```
HTTP/1.1 204 No Content
Date: Thu, 26 Oct 2012 21:32:52 GMT
```

D.11 Read the Status of an ACR using “auth” keyword (section 6.3.3.1)

Request:

```
GET /exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%Btype%3DStat/status HTTP/1.1
Accept: application/json
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

{"status": {
    "acrStatus": "Valid",
    "resourceURL":
"http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna/status"
}}
```

D.12 Read the Status of a non-existent ACR using “auth” keyword (section 6.3.3.2)

Request:

```
GET /exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%Btype%3DStat/status HTTP/1.1
Accept: application/xml
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903
```

Response:

```
HTTP/1.1 404 Not Found
Content-Type: application/xml
Content-Length: nnnn
Date: Thu, 04 Oct 2012 02:51:59 GMT

{"requestError": {"serviceException": {
    "messageId": "SVC1006",
    "text": "ACR not found"
}}}
```

D.13 Refresh an expired ACR by using “auth” keyword (section 6.3.4.1)

Request:

```
PUT /exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna/status HTTP/1.1
Accept: application/json
Content-Length: nnnn
Content-Type: application/json
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903

{"status": {
    "acrStatus": "Valid",
    "resourceURL":
"http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna/status
"
}}
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: nnnn
Date: Thu, 26 Oct 2012 21:32:52 GMT

{"status": {
    "acrStatus": "Valid",
    "resourceURL":
"http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna/status
"
}}
```

D.14 Refreshing a revoked ACR is rejected (section 6.3.4.2)

Request:

```
PUT /exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna/status HTTP/1.1
Accept: application/xml
Content-Length: nnnn
Content-Type: application/xml
Host: example.com
Authorization: BEARER 08776724-6d0d-4aa6-a404-2bc19b5cf903

{"status": {
    "acrStatus": "Valid",
    "resourceURL":
"http://example.com/exampleAPI/acrmanagement/v1/acr%3Aauth/application/acr%3Aabc123%3Bncc%3D23415%3Btype%3DDyna/status
"
}}
```

Response:

```
HTTP/1.1 403 Forbidden
Content-Type: application/xml
```

```
Content-Length: nnnn
Date: Thu, 04 Oct 2012 02:51:59 GMT

{"requestError": {"policyException": {
    "messageId": "POL1027",
    "text": "ACR, %1, is revoked. A new ACR is required to be created",
    "variables": "abc123;ncc=23415;type=Dyna"
}}}
```

Appendix E. Operations mapping to a pre-existing baseline specification (Informative)

As this specification does not have a baseline specification, this appendix is empty.

Appendix F. Light-weight resources (Informative)

As this version of the specification does not define any light-weight resources, this appendix is empty.

Appendix G. Authorization aspects (Normative)

This appendix specifies how to use the RESTful Anonymous Customer Reference Management API in combination with some authorization frameworks.

G.1 Use with OMA Authorization Framework for Network APIs

The RESTful Anonymous Customer Reference Management API MAY support the authorization framework defined in [Autho4API_10].

A RESTful Anonymous Customer Reference Management API supporting [Autho4API_10]:

- 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 Anonymous Customer Reference Management API:

- SHALL support the scope values defined in the table below;
- MAY support scope values not defined in this specification.

Scope value	Description	For one-time access token
oma_rest_acrm.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

Table 1 Scope values for RESTful Anonymous Customer Reference Management API

G.1.1.2 Downscoping

This version of the specification does not define any scope values to be used for downscoping purposes by the server and/or the resource owner.

G.1.1.3 Mapping with resources and methods

Tables in this section specify how the scope values defined in section G.1.1.1for the RESTful Anonymous Customer Reference Management API map to the REST resources and methods of this API. In these tables, the root “oma_rest_acrm.” of scope values is omitted for readability reasons.

Resource	URL Base URL: http://{serverRoot}/acrmanagement/{apiVersion}{userId}	Section refe-rence	HTTP verbs			
			GET	PUT	POST	DELETE
End User's Application	/application	6.1	**all_{apiVersion}**	n/a	**all_{apiVersion}**	n/a
Anonymous Customer Reference	/application/{ACR}	6.2	**all_{apiVersion}**	n/a	n/a	**all_{apiVersion}**
ACR Status	/application/{ACR}/status	6.3	**all_{apiVersion}**	**all_{apiVersion}**	n/a	n/a

Table 2 **Required scope values for: Management of Anonymous Customer Reference**

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 Anonymous Customer Reference Management API is used in combination with [Autho4API_10].

In the case the RESTful Anonymous Customer Reference Management API supports [Autho4API_10], the server:

- SHALL accept ‘acr:auth’ as a valid value for the resource URL variable {userId}.

SHALL conform to [REST_NetAPI_Common] section 5.8.1.1 regarding the processing of ‘acr:auth’