Telecom SOA Use Cases and Issues Version 1.0

Committee Draft 02

23 February 2010

Specification URIs:
This Version:
http://docs.oasis-open.org/soa-tel/t-soa-uci/v1.0/cd02/t-soa-uc-cd-02.html
http://docs.oasis-open.org/soa-tel/t-soa-uci/v1.0/cd02/t-soa-uc-cd-02.pdf (Authoritative)
http://docs.oasis-open.org/soa-tel/t-soa-uci/v1.0/cd02/t-soa-uc-cd-02.doc

Previous Version:
http://docs.oasis-open.org/soa-tel/t-soa-uci/v1.0/cd01-pr01/t-soa-uc-pr-01.html
http://docs.oasis-open.org/soa-tel/t-soa-uci/v1.0/cd01-pr01/t-soa-uc-pr-01.pdf (Authoritative)
http://docs.oasis-open.org/soa-tel/t-soa-uci/v1.0/cd01-pr01/t-soa-uc-pr-01.doc

Latest Version:
http://docs.oasis-open.org/soa-tel/t-soa-uci/v1.0/t-soa-uc.html
http://docs.oasis-open.org/soa-tel/t-soa-uci/v1.0/t-soa-uc.pdf (Authoritative)
http://docs.oasis-open.org/soa-tel/t-soa-uci/v1.0/t-soa-uc.doc

Technical Committee:
OASIS SOA for Telecom (SOA-Tel) TC
Chair(s):
Mike Giordano, giordano@avaya.com

Editor(s):
Enrico Ronco, enrico.ronco@telecomitalia.it
Related work:
This specification replaces or supersedes:

- Not Applicable

This specification is related to:

- Not Applicable

Declared XML Namespace(s):
Not Applicable

Abstract:
This document is the first deliverable produced within the OASIS SOA for Telecom (SOA-TEL) TC and has the objective of collecting potential technical issues and gaps of SOA standards (specified by OASIS and other SDOs) utilized within the context of Telecoms.

All perceived technical issues on SOA standards contained in this document are structured with a description of the context, a use case, and a rationalization of the possible gap within the standard.

Amongst future deliverables of the SOA-TEL TC there is a Requirements specification, which will aim to extend the current core SOA enabling stack (Web Services and/or REST, etc.) in support of Telecom needs on the basis of the issues identified within the present document.

Status:

This document was last revised or approved by the OASIS SOA for Telecom (SOA-Tel) TC on the above date. The level of approval is also listed above. Check the "Latest Version" or "Latest Approved Version" location noted above for possible later revisions of this document.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at http://www.oasis-open.org/committees/soa-tel/.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (http://www.oasis-open.org/committees/soa-tel/ipr.php).

The non-normative errata page for this specification is located at http://www.oasis-open.org/committees/soa-tel/.

Notices

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The names "OASIS", “SOA-TEL”, are trademarks of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see http://www.oasis-open.org/who/trademark.php for above guidance.

Table of Contents

Table of Figures

1 Introduction

Service-Oriented Architecture, SOA, is a design approach that divides everyday business applications into individual processes and functions, otherwise termed “service components”. These service components can then be deployed and integrated among any supporting applications and run on any computing platform. SOA enables a business to drive its application architecture by aligning the business processes with the information technology infrastructure. In effect the composite application becomes a collection of services communicating over a message bus via standard interfaces and allowing each component to be incorporated into the business process flow creating loosely coupled reusable component architecture.

The use of SOA architectural concepts allows the developer to create complex and dynamically changing applications reaching out to other component providers, who may be inside the organization or an external third party component provider.

From the perspective of an application developer, SOA is a set of programming models and tools for creating, locating, and building services that implement business processes. SOA presents a programming model to build complex composite services, and at this time the current industry approach uses web service technologies to implement SOA.

The next generation of applications are adopting a composite model where the components that are involved in the application execution path may be obtained from the efforts of multiple providers, each specializing in certain core competencies. These components will need to provide an open standards based interface to the application plane that is consumable by the tooling that the business community is comfortable with using. This makes it easier to combine components into applications to meet the needs of customers, suppliers and business partners.

This approach allows the application service provider to offer complex services, whose behavior can be dynamically managed to offer the optimal experience for the end user. As well as providing a mechanism to develop rapid applications there are also various management and deployment areas that need to be handled in this multi-component multi-vendor model as each component may have specific deployment or management considerations.

The use of SOA technology within the telecommunications area is expanding as by using a standardized interface to the network the telecommunications enablers can be exposed for consumption by the IT applications running in the business plane. These interfaces can be based upon various aspects of SOA, WSDL, Web Services Description Language, a REST, REpresentational State Transfer, model or other technology. In any case the consuming application can use the relevant IT tool set to bring these enablers into the business process to supply a real time communications service component.

Part of the work being undertaken by the OASIS SOA-TEL TC is to understand how SOA-related specifications and standards are used within the scope of the telecommunications environment and determine if there are any issues when used in this manner.

The objective of this deliverable is to identify possible technical issues related to the utilization of current SOA standards and specifications in the context of telecommunications. Such issues or gaps are illustrated by means of specific use cases.

Amongst future deliverables of the SOA-TEL TC there is a Requirements specification, which will aim to extend the current core SOA enabling stack (Web Services and/or REST, etc.) in support of Telecom needs on the basis of the issues identified within the present document.

The next steps related to this activity after these two deliverables will be finalized, will possibly be taken within the OASIS Telecom Member Section. Most likely, issues and related requirements will be grouped according to categories, and sent and presented to the TCs or Working Groups considered as “owners” of the affected specifications, in order to verify if such groups will want to analyze them and provide their solution. Other alternatives may also be evaluated on a case by case approach. Nevertheless the solution of identified issues and the addressing of the related requirements are not to be considered as part of SOA-TEL’s TC Charter.

1.1 Terminology

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]**.

1.2 Normative References

[RFC2119]	S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt, IETF RFC 2119, March 1997.
[WS-I Basic Profile]	WS-I Basic Profile Version 1.0: "Final Material", available at http://www.ws-i.org/Profiles/BasicProfile-1.0-2004-04-16.html.
[WSDL 1.1]	W3C Note (15 March 2001): "Web Services Description Language (WSDL) 1.1". http://www.w3.org/TR/2001/NOTE-wsdl-20010315.
[SOAP 1.2]	W3C SOAP v.1.2, available at http://www.w3.org/TR/soap12-part1/
[WS-N 1.3]	OASIS Standard, “Web Services Base Notification 1.3 (WS-BaseNotification)”, version 1.3, 1 October 2006. http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.htm
[WS-A 1.0]	W3C Web Services Addressing 1.0 – Core W3C Recommendation 9 May 2006, http://www.w3.org/TR/2006/REC-ws-addr-core-20060509
[WS-S 1.1]	OASIS Standard, “Web Services Security specification, version1.1”, 1 February 2006. http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.0.pdf and http://docs.oasis-open.org/wss/oasis-wss-rel-token-profile-1.0.pdf
[SOA RM 1.0]	OASIS Standard, “OASIS Reference Model for Service Oriented Architecture 1.0”, Oct. 12, 2006. http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf
[SCA Assembly 1.1]	OASIS Committee Draft 03, “Service Component Architecture Assembly Model Specification Version 1.1”, Mar. 09, http://docs.oasis-open.org/opencsa/sca-assembly/sca-assembly-1.1-spec-cd03.html
[SOA RA 1.0]	OASIS Public Review Draft 01, “ Reference Architecture for Service Oriented Architecture 1.0”, Apr. 2008, http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra-pr-01.pdf
[WSDM-MOWS]	OASIS Standard - Web Services Distributed Management: Management of Web Services (WSDM-MOWS) 1.1, 1 August 2006, http://docs.oasis-open.org/wsdm/wsdm-mows-1.1-spec-os-01.htm
[WSDL 2.0]	W3C Web Services Description Language (WSDL) Version 2.0 Part 0: Primer, http://www.w3.org/TR/2007/REC-wsdl20-primer-20070626/Recommendation, June 2007
[SAML 2.0]	OASIS Standard, “Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0”, March. 2005, http://docs.oasis-open.org/security/saml/v2.0/saml-2.0-os.zip

1.3 Non-Normative References

[WS Landscape]	Possible representation of web services specification landscape, available at http://www.innoq.com.

2 Context setting

This section provides a classification of the issues presented in the document.

The list of received contributions is presented hereafter.

1. **Transaction Endpoints Specification**, related to a possible issue on the W3C WS-Addressing specification; the necessity to specify the endpoint of a final result of a “process/transaction" (i.e. asynchronous response) result should be sent.
2. **Notification**, related to a possible issue on the OASIS WS-Notification specification; the necessity to specify for the Provider of a notifications service to specify the endpoint to which the Notification should be sent.
3. **SOAP Protocol** issue, related on a possible issue on the W3C SOAP specification; the necessity for an “intermediate SOAP node” to also cover the role of “SOAP ultimate receiver node”.
4. **SAML Token Correlation**, related to a possible issue on the OASIS WS-Security specification; the necessity of enabling “correlation” of a security token to another.
5. **SAML Name Identifier Request**, related to a possible issue on the OASIS SAML specification: the possibility to extend the SAML protocol to enable a Service provider (SP) to register single Users with an Identity Provider (IdP) “on-the-fly”, as the need arises.
6. **SAML Attribute Management**, related to a possible issue on the OASIS SAML specification: the possibility to extend the SAML protocol to enable a SP (Service Provider) to transmit user attributes to be stored within an IdP (Identity Providers).
7. **User-ID Forwarding**, related to a possible issue in the OASIS WS-Security specification; the necessity to define a common means to add two (or more) credentials in one message.
8. **Services exposing Management Interface**, related to possible issues on the OASIS SOA Reference Model (SOA RM) and SOA Service Component Architecture (SCA) Assembly Model; the necessity to specify more than one service interface for a single SOA service.
9. **Metadata in support of Service Lifecycle Management**, related to the possibility to enrich the OASIS SOA Reference Architecture (SOA RA) with metadata necessary for Service Lifecycle Management identified within the TM Forum SDF program.
10. **Universal Communications Profile**, related to the specification of a possible common profile for universal interoperability across domains.

The document is organized in the following sections:

- Section 3, Issues on Addressing and Notification;
- Section 4, Issues on Communication Protocols;
- Section 5, Issues on Security;
- Section 6, Issues on Management;
- Section 7, Issues on SOA collective standards usage.

All perceived technical issues on SOA standards contained in this document are structured with a description of the context, a use case, and a rationalization of the possible gap within the standard.

3 Issues on Addressing and Notification

3.1 Transaction Endpoints Specification

3.1.1 Scenario/context

The issue presented in this section derives from a concrete case, implemented within an operator's SOA Middleware.

The operator is in the process of deploying a SOA infrastructure, of which some of the constituting elements are an ESB (Enterprise Service Bus), a BPM (Business Process Manager), some "Service Consumers (systems or applications), some "Service Providers" (systems or applications).

An aspect to be considered is that to satisfy performance criteria it has been decided that the ESB must be intrinsically "stateless" (i.e. it must not store any persistence information on destination of incoming service requests).

Moreover, the "number" of ESB can vary, i.e. there can be interconnected trunks of different vendors' ESB.

3.1.2 Use Case

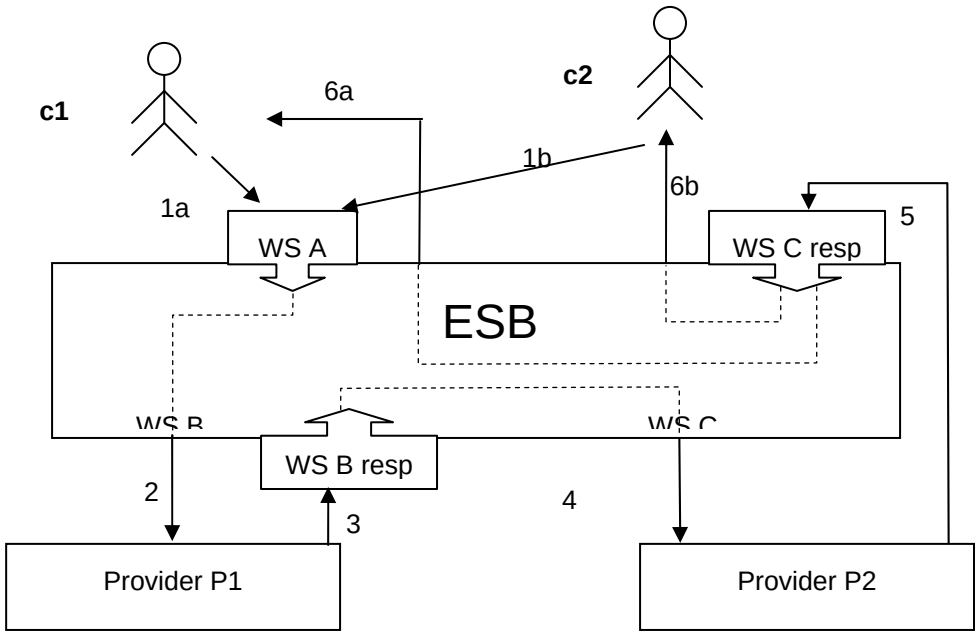
The following Use Case describes the technical problem (Figure 1 and Figure 2). To improve readability the depicted use case presents only one instance of ESB, but the possible solution to the problem must satisfy also the cases of multiple instances of ESB.

A Service Consumer (C1 or C2) invokes a Service, implemented as a Web Service (Web Service A).

Such WSA is achieved as an "itinerary" with the composition of more elementary services, provided by Provider P1 and Provider P2.

The ESB provides intermediary services for final exposition, enrichment and Data reconciliation and routing.

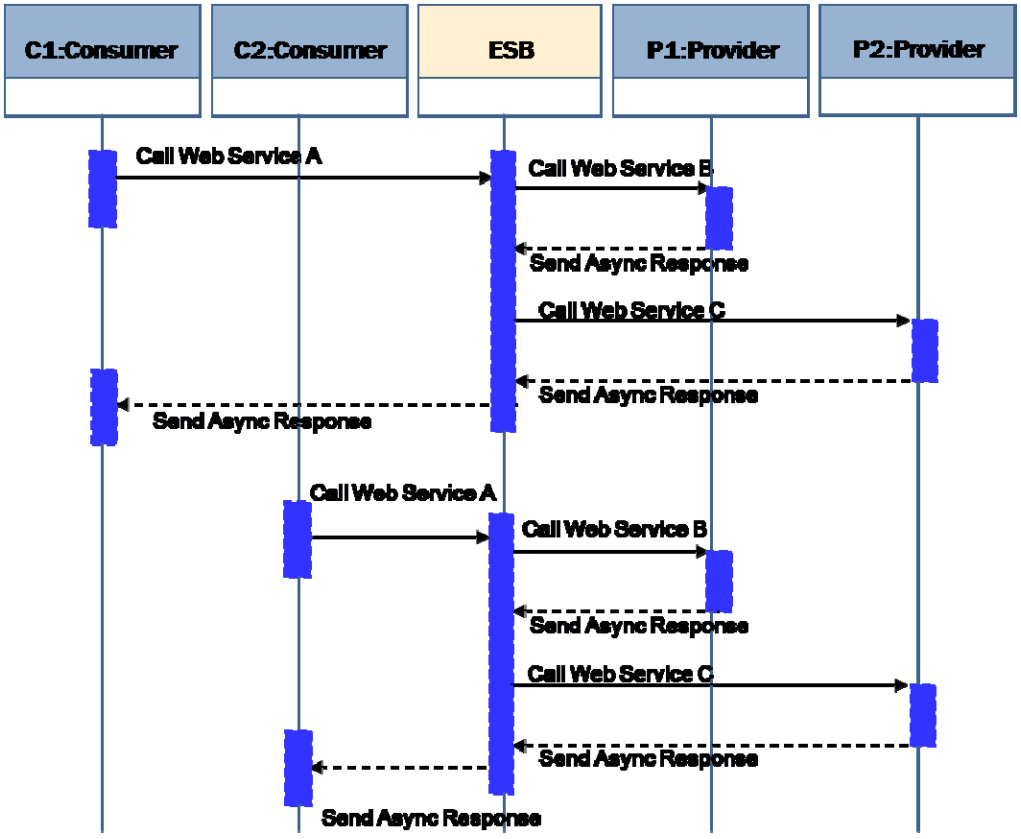
- Case **A**: C1 is the originator and final receiver.
- Case **B**: C2 is the originator and final receiver.



171

172 Figure 1: Transaction endpoints scenario

173



174

175

176 Figure 2: Transaction endpoints scenario flow

Use Case Steps:

Case **A**

- C1 invokes WSA**,** exposed by ESB**.**
- WSA is executed with the internal composition (transparent to C1) and with intermediary services provided by the ESB.
- At the end of the internal interactions, the ESB forwards the response to C1.

Case **B**

- C2 invokes WSA**,** exposed by ESB**.**
- WSA is executed with the internal composition (transparent to C2) and with intermediary services provided by the ESB.
- At the end of the internal interactions, the ESB forwards the response to C2.

3.1.3 Perceived Technical Issue

With the current knowledge and expertise, in presence of an ESB offering intermediary services, there is no formal way to specify the endpoint (e.g. C1 or C2) to which the final result of a “process/transaction" (i.e. asynchronous response) result should be sent.

Affected specification is W3C **[WS-A]**.

3.2 WS-Notification

3.2.1 Scenario/context

Event-Driven Architectures are extremely important in environments, like Telecoms, where it is necessary to handle massive network events that have a business value to registered subscribers.

Often these solutions rely on proprietary protocols that work against the implementation of SOA principles.

There’s a strong technical and business need for a Notify/Subscribe protocol which could be widely adopted and used by Vendors and Telecom Operators. Moreover the protocol should support the presence of intermediaries between the Subscriber and the Notifier.

In the following, 2 use cases and related issues are presented, one related to a lack of acceptance of an existing standard by the vendor community, and one on a specific technical issue on existing standards.

Specifications addressed within this section are:

- OASIS Web Services Base Notification 1.3 (WS-BaseNotification) **[WS-N]**, OASIS Standard, 1 October 2006, http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.htm
- W3C Web Services Addressing 1.0 **[WS-A]** – Core W3C Recommendation 9 May 2006, http://www.w3.org/TR/2006/REC-ws-addr-core-20060509.

3.2.2 Use Case (A)

The following Use Case describes a technical problem which is common for a Telecom Operator (ref. Figure 3).

An Application wants to be notified when a specific “Large Account Mobile Number” receives an SMS with a specific keyword in the message content.

Use Case Steps:

1. The Application informs the Provider that it wants to be notified when the specified Large Account Number “33536821686” receives an SMS containing the word “poll”.

2. The Provider notifies the Application when an incoming event from the underlying network responds to the Subscribing criteria.

3. The Application informs the Provider that it does not want to be notified anymore when the specified Large Account Number “33536821686” receives an SMS containing the word “poll”.

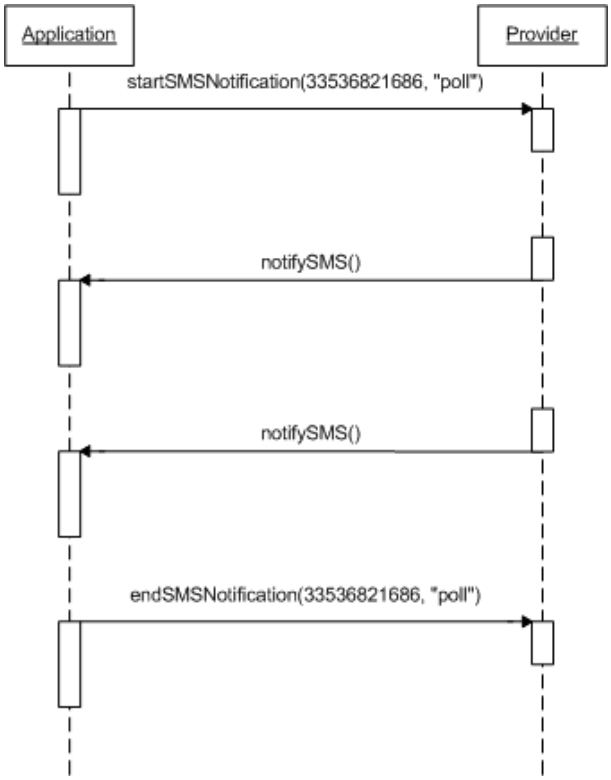


Figure 3: Notification Use Case (a) flow

3.2.3 Perceived technical issue (A)

Currently a commonly used interoperable standard does not exist to address “Notify/Subscribe message exchanges”.

The last approved specification, OASIS WS-Notification **[WS-N]**, has been very poorly adopted by the vendors community and consequently has no interoperability value.

The need is that such specification gets endorsed/adopted by the vendor community in order for it to add value in this specific context.

Such lack is perceived as a strong market gap with negative impacts for both Telecom Operators and Third Parties involved in the development of new services:

1) Operators are limited in their business development since they must rely on costly proprietary solutions and customizations implemented by vendors;
2) Third Parties, who are typically involved in developing new services for their customers, can not fully exploit in their services development the open network infrastructures provided by Telco Operators.

3.2.4 Use Case (B)

The following Use Case describes a second technical problem which is common for Telecom Operators (ref. Figure 4).

An Application must be notified when a specific “Large Account Mobile Number” receives an SMS with a specific keyword in the message content. There are one or more intermediaries between the Application and the Provider.

Use Case Steps:

1. The Application informs the Intermediary that it wants to be notified when the specified Large Account Number “33536821686” receives an SMS containing the word “poll”.

2. The Intermediary sends the subscription request to the Provider.

3. The Provider notifies the Intermediary when an incoming event from the underlying network responds to the Subscribing criteria.

4. The Intermediary sends the notification to the Application.

5. The Application informs the Intermediary that it does not want to be notified anymore when the specified Large Account Number “33536821686” receives an SMS containing the word “poll”.

6. The Intermediary sends the “unsubscribe” request to the Provider.

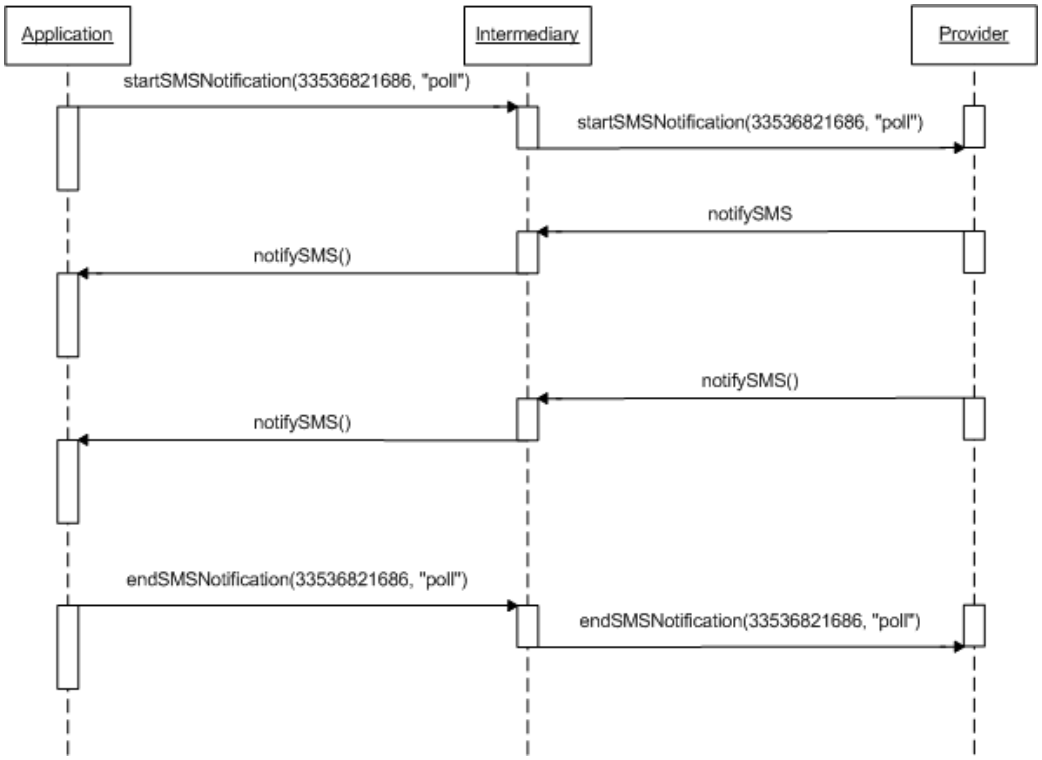


Figure 4: Notification use case (b) flow

3.2.5 Perceived Technical issue (B)

The last approved specification to support Notify/Subscribe patterns, WS-Notification **[WS-N]**, relies on W3C WS-Addressing **[WS-A]** for the asynchronous delivery of notifications, which means that there is no formal way for the Provider to specify the endpoint to which the Notification should be sent.

As an example, in the case illustrated above there is no standard way for the Provider to indicate the original Application as destination of the notification, due to the presence of intermediary (ies) in the path.

The issue on WS-A impacts thus also the WS-N specification. Refer to Section 3.1 within this document for the technical issues with the WS-A specification.

> "in presence of intermediary, there is no formal way to specify the endpoint to which the final result of a "process/transaction" (i.e. asynch. response) result should be sent."

The technical problem here exposed prevents Telecom Operators to develop standardized solutions for the management of "multiple notify/subscribe patterns", and forces to rely on costly customizations and proprietary solutions.

4 Issues on communications protocols

4.1 SOAP

4.1.1 Scenario/context

The issue presented in this section derives from a concrete case, occurred within the context of the development of a platform for Mobile Virtual Network Operators (MVNOs).

This section is related to a possible technical issue within the SOAP 1.2 **[SOAP 1.2]** specification, in particular on the "SOAP Intermediary" and "Ultimate SOAP receiver" concepts.

The specification defines the following (within its section 1.5.3):

- **Initial SOAP sender**
  - The SOAP sender that originates a SOAP message at the starting point of a SOAP message path.
- **SOAP intermediary**
  - A SOAP intermediary is both a SOAP receiver and a SOAP sender and is targetable from within a SOAP message. It processes the SOAP header blocks targeted at it and acts to forward a SOAP message towards an ultimate SOAP receiver.
- **Ultimate SOAP receiver**
  - The SOAP receiver that is a final destination of a SOAP message. It is responsible for processing the contents of the SOAP body and any SOAP header blocks targeted at it. In some circumstances, a SOAP message might not reach an ultimate SOAP receiver, for example because of a problem at a SOAP intermediary. An ultimate SOAP receiver cannot also be a SOAP intermediary for the same SOAP message (see **2. SOAP Processing Model**).

In particular it is stated that

- A **SOAP Intermediary** processes the header of a SOAP message.
- An **Ultimate SOAP receiver** processes the body of a SOAP message and can not also be a SOAP intermediary for the same SOAP message.

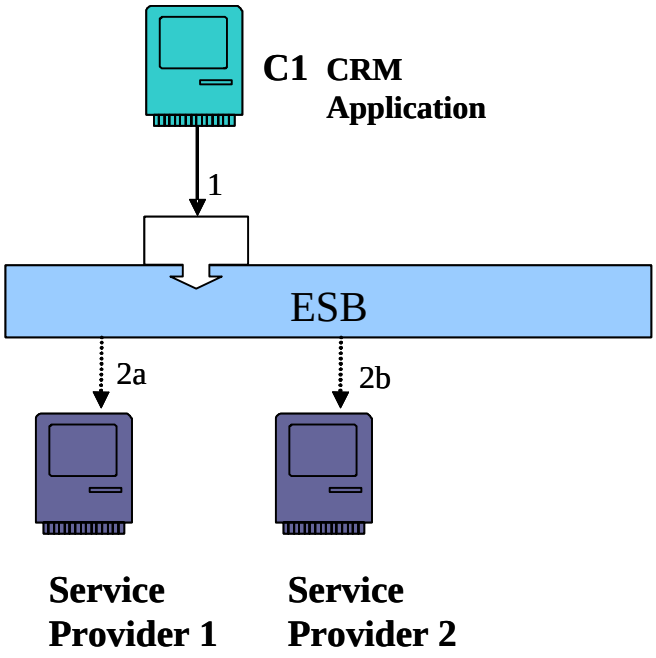
The issue presented in the following Use Case illustrates the need to have a SOAP Intermediary which must process the body of a SOAP message in addition to its "canonical" role of processing the SOAP message header.

The case is included within the activities of deployment of a company-ware SOA infrastructure, of which some of the constituting elements are an ESB (Enterprise Service Bus), some "Service Consumers (systems or applications), some "Service Providers" (systems or applications), a BPM (Business Process Manager), etc.

4.1.2 Use Case

A Service Consumer C1 (e.g. a CRM application) invokes a Web Service to execute a transaction within a specific business process for the management of Mobile Virtual Network Operators (ref. Figure 5).

The access point for the Consumer C1 is the ESB, which exposes such Web Service and moreover executes some of its typical functions such as Data Enrichment and Content Based Routing (CBR).



313
314
315 Figure 5: "SOAP" use case representation
316
317 Figure 6 contains the SOAP message which is the request formulated by the Service Consumer (e.g. the
318 CRM application) to the ESB.
319 The request contains:
320 • A SOAP Envelope (in **black** color). This is enclosed for completeness but is not subject of
321 discussion within this contribution;
322 • the SOAP Header, in **red** color;
323 • The SOAP message Body, in **blue** (and **green)** color.
324
325 With reference to the SOAP 1.2 specification, the ESB is a "SOAP Node" (ref. Section 1.5 in the [SOAP
326 1.2] specification).
327

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:m0="http://operator/BSS/MVNO/NetProvisioningCustomTypes">
<SOAP-ENV:Header>
<m:Header xmlns:m="http://operator/BSS/MVNO/NetProvisioningHeaderTypes">
        <m:sourceSystem>String</m:sourceSystem>
        <m:businessID>String</m:businessID>
</m:Header>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
        <m:ActivateLineMessage xmlns:m="http://telecomitalia.it/BSS/MVNO/NetProvisioning">
                <m:Command>
                        <m0:description>String</m0:description>
                </m:Command>
                <m:MobilePhoneAccount>
                        <m0:telephoneNumber>String</m0:telephoneNumber>
                        <m0:ManagedOn>
                                <m0:ICCID>String</m0:ICCID>
                        </m0:ManagedOn>
                </m:MobilePhoneAccount>
                <m:NetworkProfile>
                        <m0:ID>String</m0:ID>
                        <m0:TDS>String</m0:TDS>
                </m:NetworkProfile>
                <m:Context>
                        <m0:value>String</m0:value>
                </m:Context>
        </m:ActivateLineMessage>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figure 6: SOAP message, request formulated by the Service Consumer

The ESB for this use case must process the body of the SOAP message in order to perform 2 operations:

1. "Data Enrichment",

   The ESB queries a provisioning system to obtain the IMSI of the asset (mobile phone number) in order to add such data to the message: it invokes a Web Service, exposed by that system, which takes in input the ICCD, present in the message, and returns the IMSI.

2. CBR (Content Based Routing)

   The ESB decides on the final receiver of the SOAP message on the basis of the content of the "Context" field (in **green** in Figure 6).

   Once such tasks are performed, the ESB deletes the "Context" field from the message and subsequently forwards the SOAP message to the selected Service Provider.

Note:

The Data Enrichment task is executed with the collaboration of other “Service Providers” (different than SP1 or SP2), but it is not a subject to be discussed within this contribution: for this reason details are omitted.

After such tasks are complete, the ESB must forward the SOAP message to the selected Service Provider, which is the “real” Ultimate SOAP receiver. The message that must be finally sent to the SP by the ESB is the one depicted in Figure 7.

It is fundamental to state that the Service Provider needs the header present in the SOAP message, e.g. because the content of the “business ID” field can not be associated to the body of the SOAP message.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:m0="http://operator/BSS/MVNO/NetProvisioningCustomTypes">
<SOAP-ENV:Header>
        <m:Header xmlns:m="http://operator/BSS/MVNO/NetProvisioningHeaderTypes">
                <m:sourceSystem>String</m:sourceSystem>
                <m:businessID>String</m:businessID>
        </m:Header>
</SOAP-ENV:Header>
<SOAP-ENV:Body>
        <m:ActivateLineMessage xmlns:m="http://operator/BSS/MVNO/NetProvisioning">
                <m:Command>
                        <m0:description>String</m0:description>
                </m:Command>
                <m:MobilePhoneAccount>
                        <m0:telephoneNumber>String</m0:telephoneNumber>
                        <m0:ManagedOn>
                                <m0:ICCID>String</m0:ICCID>
                                <m0:IMSI>String</m0:IMSI>
                        </m0:ManagedOn>
                </m:MobilePhoneAccount>
                <m:NetworkProfile>
                        <m0:ID>String</m0:ID>
                        <m0:TDS>String</m0:TDS>
                </m:NetworkProfile>
        </m:ActivateLineMessage>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figure 7: Message needed by the Service Provider (Ultimate SOAP receiver)

Nevertheless, given the initial definitions (section 1.5.3 of the SOAP Specification), since the ESB needs to elaborate the body of the message, it becomes an “Ultimate SOAP receiver” and thus can not be simultaneously classified as “SOAP Intermediary”.

The consequence of this is that the ESB can not forward the header of the SOAP message to the selected Service Provider (i.e. to the “real” Ultimate SOAP receiver).

Thus the message really forwarded by the ESB is depicted in Figure 8.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:m0="http://operator/BSS/MVNO/NetProvisioningCustomTypes">
<SOAP-ENV:Body>
        <m:ActivateLineMessage xmlns:m="http://operator/BSS/MVNO/NetProvisioning">
                <m:Command>
                        <m0:description>String</m0:description>
                </m:Command>
                <m:MobilePhoneAccount>
                        <m0:telephoneNumber>String</m0:telephoneNumber>
                        <m0:ManagedOn>
                                <m0:ICCID>String</m0:ICCID>
                                <m0:IMSI>String</m0:IMSI>
                        </m0:ManagedOn>
                </m:MobilePhoneAccount>
                <m:NetworkProfile>
                        <m0:ID>String</m0:ID>
                        <m0:TDS>String</m0:TDS>
                </m:NetworkProfile>
        </m:ActivateLineMessage>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figure 8: Message effectively forwarded by the ESB to the appropriate Service Provider

This is a real case faced by the operator, and to overcome the problem some costly ad-hoc developments-customizations were necessary to **re-build / reinsert** the necessary header within the message before the ESB could forward the “complete” message to the final Service Provider.

4.1.3 Perceived Technical issue

In the SOAP specification the following is stated.

2.1 SOAP Nodes

A SOAP node can be the initial **SOAP sender**, an **ultimate SOAP receiver**, or a **SOAP intermediary**. A SOAP node receiving a SOAP message MUST perform processing according to the SOAP processing model as described in this section and in the remainder of this specification, etc.

2.2 SOAP Roles and SOAP Nodes

In processing a SOAP message, a SOAP node is said to act in one or more SOAP roles, each of which is identified by a URI known as the SOAP role name. The roles assumed by a node MUST be invariant during the processing of an individual SOAP message. This specification deals only with the processing

of individual SOAP messages. No statement is made regarding the possibility that a given SOAP node might or might not act in varying roles when processing more than one SOAP message.

Table 2 defines three role names which have special significance in a SOAP message (see **2.6 Processing SOAP Messages**).

Table 2: SOAP Roles defined by this specification

Short-name	Name	Description
`Next`	"http://www.w3.org/2003/05/soap-envelope/role/next"	Each SOAP intermediary and the ultimate SOAP receiver MUST act in this role.
`None`	"http://www.w3.org/2003/05/soap-envelope/role/none"	SOAP nodes MUST NOT act in this role.
`ultimateReceiver`	"http://www.w3.org/2003/05/soap-envelope/role/ultimateReceiver"	The ultimate receiver MUST act in this role.

In addition to the SOAP role names defined in **Table 2**, other role names MAY be used as necessary to meet the needs of SOAP applications.

Due to the fact that the ESB (as a SOAP Node) processes the body of the message, it is classified as “ultimateReceiver”.

As a consequence, the ESB can not “Forward” the SOAP Header to the appropriate Service Provider (ref. Sections 2.7.1 of the SOAP specification) since it has value “`ultimateReceiver`”. The following table depicts the behavior of the ESB being an `ultimateReceiver`.

Role		**Header block**	
Short-name	**Assumed**	**Understood & Processed**	**Forwarded**
next	Yes	Yes	No, unless reinserted
		No	No, unless relay ="true"
user-defined	Yes	Yes	No, unless reinserted
		No	No, unless relay ="true"
	No	n/a	Yes
ultimateReceiver	Yes	Yes	n/a
		No	n/a
none	No	n/a	Yes

The case presented shows that a SOAP Intermediary (the ESB), which is clearly not the “ultimate receiver” of the SOAP message, is forced to assume the role of “`ultimateReceiver`” since it processes

403 the body of the message. This prevents the ESB to correctly perform it s “proper” intermediary role, since
404 “An ultimate SOAP receiver cannot also be a SOAP intermediary for the same SOAP message”.

405 The perceived technical gap suggested by the operator is that the SOAP specification should be modified
406 in order to enable a SOAP Intermediary node to “forward” the SOAP Header in automatic mode (thus
407 without the Header reinsertion) even if such node performs some processing operation over the body of
408 the SOAP message.

409 Another way of expressing this perceived gap is to state that currently only 3 roles are allowed for a
410 SOAP Node (i.e. initial SOAP Sender, SOAP intermediary, SOAP ultimate receiver – section 2.1 of the
411 SOAP 1.2 specification), while a probable fourth role enabling the simultaneous body processing and
412 header forwarding of a specific SOAP message may be needed.

413 Should the specification already enable this, OASIS SOA-TEL TC suggests to modify them in order to
414 avoid possible ambiguities and misinterpretations.

415

5 Issues on Security

5.1 SAML Token Correlation

5.1.1 Scenario/context

The issue presented in this section derives from a concrete case of telecommunications services' sales and post-sales: in particular the activation and provisioning of ADSL service to residential customers.

The business process under analysis is complex and necessitates to be orchestrated by a BPM (Business Process Management) application.

Such process is a "long-running" type process: in fact one of its tasks requires a human intervention within the central office, which can be executed within hours (or days).

This implies that the process must be handled in a different mode from the "security management" perspective. This section addresses potential issues within the OASIS Web Services Security specification, **[WS-S 1.1]**.

5.1.2 Use Case

A consumer, e.g. a CRM application invokes a service to execute a specific business process, the activation of ADSL services for a residential customer.

The BPM application gets in charge of the orchestration/execution of such processes.

Given the fact that the process is "long-running", the BPM shall, at a given point, suspend the orchestration/execution of the process until it will receive a specific "activity closure" event from a back office system once the appropriate technician will have terminated his manual tasks.

The following schema Figure 9 depicts a simplified transaction diagram, while Figure 10 provides a pictorial representation of the Use Case.

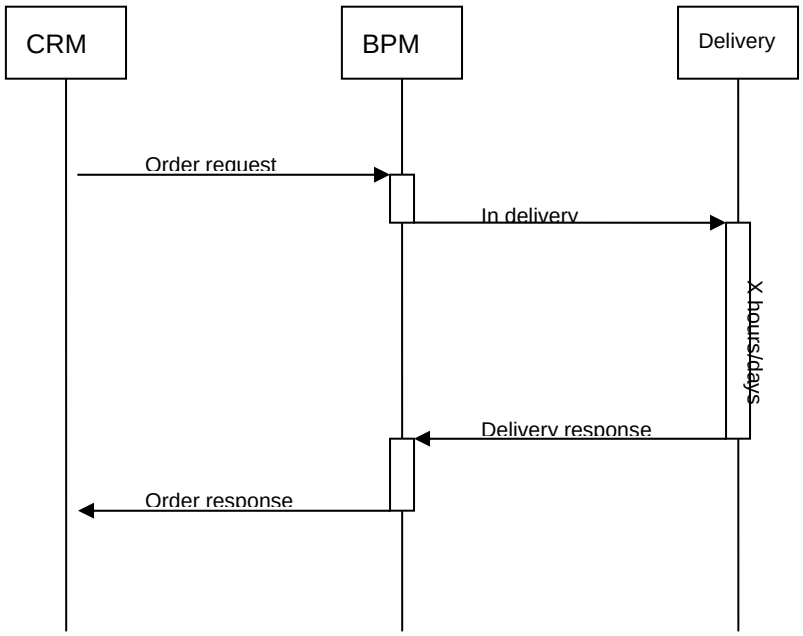


Figure 9: Simplified transaction diagram for the "SAML token correlation" use case

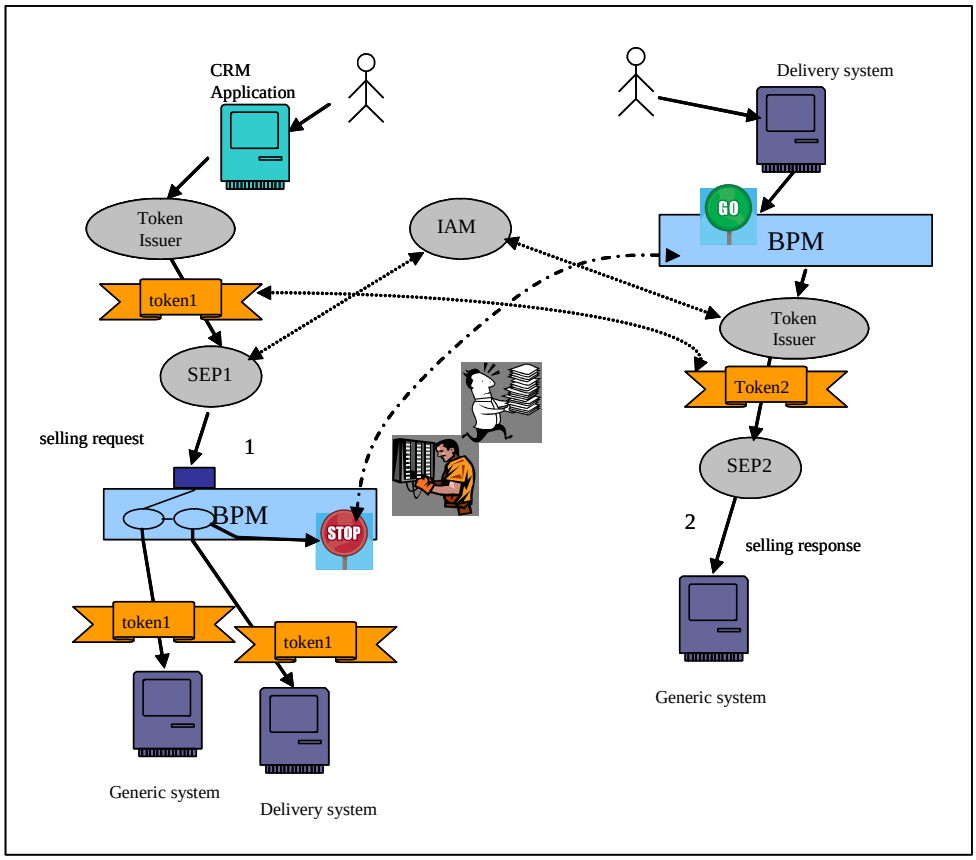


Figure 10: "SAML token correlation" use case: pictorial representation

Use Case steps.

- The CRM sends an ADSL activation request.
- The consumer (CRM) provides its credentials to a Token Issuer and requires the generation of a security token, *"token1"*. The token is associated to the initial message and has limited duration, since extending it would mean to have a weaker security policy.
- The Security Enforcement Point, interacting with the policy decision point (IAM) (Identity Access Manager), applies the authentication and authorization policies.
- The BPM orchestrates the process interacting with the various services exposed by the involved systems within the company SOA infrastructure. All interactions are executed with the *"token1"* as security token.
- When appropriate, the BPM invokes a service exposed by a Delivery system to obtain a physical configuration within the central office. At this stage the BPM suspends the execution of the business process (the duration of the task may require hours or days), awaiting for the reception of a specific "activity closure" event.
- The Delivery System activates the technical configuration task.
- A human intervention is performed within the central office.
- Once this task is terminated, the technician reports the "activity closure" on the Delivery system, which generates the "activity closure" event for the BPM.
- The BPM resumes the suspended process, invoking the "next step" in the ADSL activation process.
- If the security token *"token1"* is expired, the BPM requests the Token Issuer to generate a new security token, *"token2"*, since the previous is not valid any more.
- The remaining portion of the process is executed utilizing the new security token, *"token2"*.

5.1.3 Perceived Technical issue

In the described scenario the issue is related to which credentials (capabilities) must be utilized to generate the security token *"token2"*.

The BPM is responsible for the orchestration/execution of the process, and is the entity which is entitled to request the generation of the new security token *"token2"*, which is of course different from *"token1"*.

This is a weakening factor for the "security architecture", since an element of the middleware infrastructure (the BPM) would need to request the generation of security tokens which are not "correlated" (or "directly coupled") to the real entity which requires the initiation of the business process (i.e. the CRM application, thus the CRM sales representative) and to the business process itself. It is a requirement for the Telecom Operator to reduce such potential security threats.

It should be possible for the BPM to request the Token Issuer to generate a new token "associated" to the *"token1",* and to maintain evidence of that correlation, in order to authorize the BPM itself, once security checks are validated by the IAM, to invoke all pending services within the second part of the process because such invocations are "really" part of a "security authorized" business process.

The WS-Sec specification **[WS-S 1.1]**, in Section 7 - row 824, states that mechanisms for referencing security tokens are defined.

In row 870 the following is stated:

---

870 */wsse:SecurityTokenReference/@wsse:Usage*

871 This optional attribute is used to type the usage of the

872 `<wsse:SecurityTokenReference>.` Usages are specified using URIs and multiple

873 usages MAY be specified using XML list semantics. **No usages are defined by this**

874 **specification.**

Thus, form a syntactical perspective, the specification enables the "correlation" of a security token to another one, but it does not prescribe how such correlation should be formalized.

Moreover, within non-normative Appendix D "SecurityTokenReference Model", specific examples of security token referencing are provided, with emphasis of the "signature referencing".

Within this appendix, Row 2413 to 2432 do provide an example of "non-signature references", but the specification states that

2430 This may be an expensive task and in the

2431 general case impossible as there is no way to know the "schema location" for a specific

2432 namespace URI.

In conclusion, the lack of normative guidelines on how to address this problem is perceived as a strong issue for a Telecom Operator because the "correlation" problem must anyhow be solved, but adopted solutions result to inevitably be proprietary, costly, non-standard, vendor/platform dependent customizations.

5.2 SAML Name Identifier Request

5.2.1 Scenario/context

The context of this section is that of a SP (Service Provider) being newly added to the circle of trust of an IdP (identity Provider).

Currently, as soon as a SP becomes a member of the circle of trust of an IdP, the SP is forced to import all of the SP's Users into the IdP's databases.

The objective of this contribution is to propose a modification to the current SAML V2.0 specification (saml-core-2.0-os.pdf) so that the SP can be enabled to register single Users with the IdP "on-the-fly", as the need arises. Such goal can be achieved with the introduction of a new SAML protocol, named "SAML Name Identifier Request" within the SAML specification.

SAML supports SPs to get attributes about Users from an IdP. Regarding name identifiers, the SP usually sends an AuthnRequest to the IdP. Then, the IdP sends an AuthnResponse containing a NameIdentifier ("Subject") back to the SP. However, if a SP is newly added to the circle of trust of an IdP, the IdP will not know of the User identifiers of the SP, which is required in order for the IdP to authenticate the Users of a SP.

The issue highlighted in this section aims at possibly extending the SAML specifications.

5.2.2 Use Case

A user device, a SP and an IdP are the actors of this use case of the SAML Name Identifier Request mechanism. The SP is new to the circle of trust of the IdP. The IdP does not know a name identifier of the user device. The IdP requests a name identifier from the SP, who sends the desired name identifier to the IdP.

Figure 11 provides a high-level message flow illustrating this SAML Name Identifier Request use case. Messages 4 and 6 belong to the SAML Name Identifier Request protocol this contribution is aiming at. These messages are interlaced into the SAML Authentication Request and Response exchange between SP and IdP and are not specified in SAML V2.0 yet (therefore, marked in red):

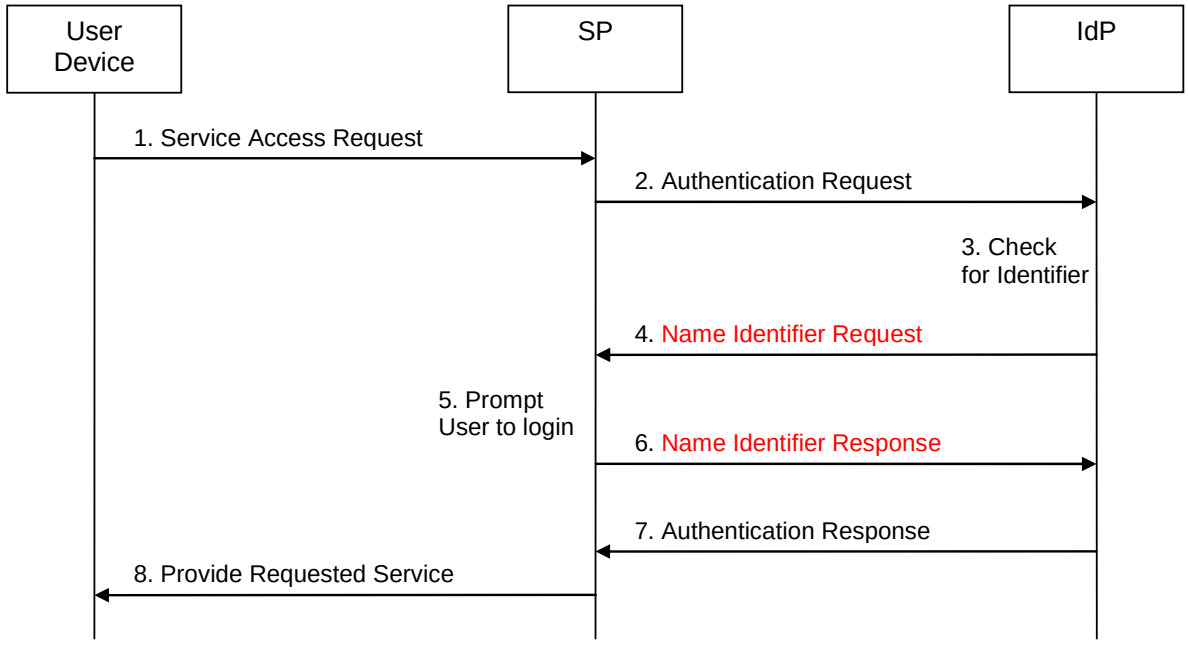


Figure 11: "SAML name Identifier request" use case: pictorial representation

The single steps of this use case are as follows:

1) The user requests access to a service offered by a SP. The user device does not include any authentication credentials.
2) Since access to this service requires the User to be authenticated but the request in step 1 does not include any authentication credentials, the SP sends an Authentication Request to the IdP. This Authentication Request may be passed to the IdP via the user device using redirection.
3) The IdP checks the Authentication Request received in step 2, and - as the SP is new to the IdP's circle of trust - the IdP determines that it does not have an identifier stored in its database for the User for the given SP.

   Conventionally, the IdP would respond to the Authentication Request by issuing an error message or a randomly generated identifier. This, however, is problematic: In the former case, the service access request in step 1 breaks down. In the latter case, the SP has to ask the user for his credentials and then send (usually via a backchannel) a message to the IdP indicating that from now on the IdP should use the "real identifier" instead of the random one for the given user (this could be done via the NameIdentifier Management Protocol).
4) This step is not defined in SAML V2.0: Since the IdP has realized in step 3 that it does not have an identifier for the combination of the User and the SP, the IdP generates a message called Name Identifier Request and sends it to the SP.
5) Upon receipt of the Name Identifier Request, the SP recognises that the IdP does not have an identifier for the combination of SP and User. Therefore, the SP prompts the User to log in to the SP.
6) This step is also not defined in SAML V2.0: The SP sends a message called Name Identifier Response to the IdP. This response message includes the identifier for the combination of User and SP that the IdP is to use in any further communication and authentication processes.
7) On receipt of the Name Identifier Response, the IdP stores the identifier contained in the Name Identifier Response in its database. The IdP sends an Authentication Response to the SP, which uses the identifier received in step 6.
8) The SP grants the User access to the requested service.

5.2.3 Perceived Technical issue

This contribution aims at introducing a new SAML protocol called SAML Name Identifier Request protocol into the SAML 2.0 specifications.

5.3 SAML Attribute Management Request

5.3.1 Scenario/context

More and more services and applications are becoming available on the Internet, and many of these services and applications require authentication. With the convergence of telco and Internet domain, the telco has added functionality, namely IDM functions. The telco operator will collaborate with several SPs, that in return depend on the telco's profile and attribute store. This causes a scenario where not the SP manages the attributes, but the telco operated IDM.

One approach that has been developed to assist users to access multiple services and applications, each requiring separate authentication procedures, involves the use of identity federation.

Security Assertion Markup Language (SAML) is an XML standard for exchanging authentication and authorisation data between security domains. For example, SAML is used for exchanging assertion data between an identity provider (a producer of assertions) and a service provider (a consumer of assertions).

The issue highlighted in this section aims at possibly extending the SAML specifications.

5.3.2 Use Case

A user wishes to use his attribute information across multiple service providers, such attribute information can be layout, preferred email address, etc. Today, these attributes are stored locally at each of service provider. Thus, user will have to enter and changes the same attributes multiple times in order to ensure they are consistent for each of the different service providers the user has an account with, resulting in a bad user experience.

The user creates a temporary or transient account. The service provider allows the user to set specific settings like coloring, text size, etc. But he/she does not want to set these setting again each time the user logs in because the service provider will not be able to link the attributes for a user's temporary account with the user's permanent account. This is because by the very nature of a temporary or transient account the next time the user logs on to the service provider the user will have a different username and so the service provider will not be able to link the attributes for a user's temporary account with the user's permanent account.

Figure 12 provides a high-level message flow outlining the proposed SAML Attribute Management protocol:

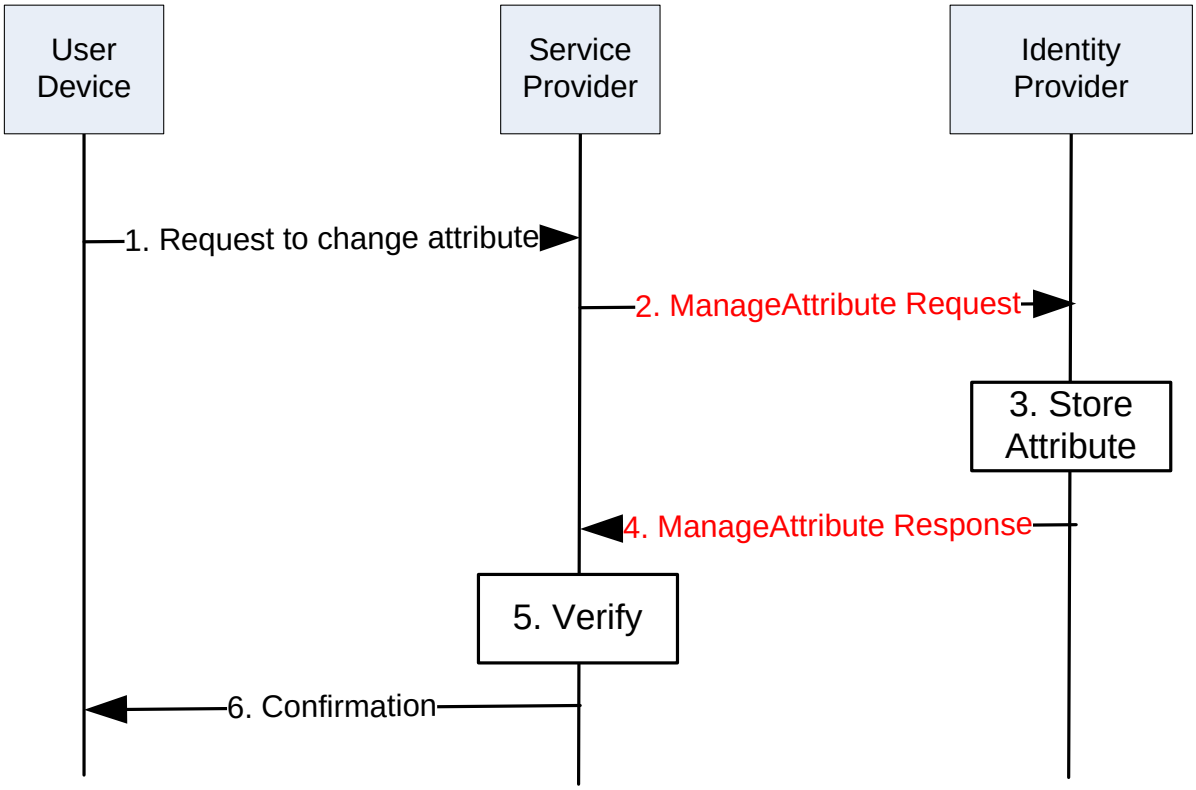


Figure 12: "SAML Attribute Management request" use case: pictorial representation

The ManageAttribute Request and Response messages are marked in red since the SAML 2.0 does not support such messages yet. The ManageAttribute Request allows the Service Provider to manage attributes stored on the Identity Provider side. As an example, the following XML instance of a ManageAttribut Request asks the Identity Provider to set the value of the "mail" attribute to "trscavo@gmail.com":

609 The following example shows what such a change in the specification would enable to do:

```
610 <samlp:ManageAttributeRequest
611   xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
612   xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
613   ID="aaf23196-1773-2113-474a-fe114412ab72"
614   Version="2.0"
615   IssueInstant="2006-07-17T20:31:40Z">
616   <saml:Issuer
617     Format="urn:oasis:names:tc:SAML:1.1:nameid-
618 format:X509SubjectName">
619     C=US, O=NCSA-TEST, OU=User, CN=trscavo@uiuc.edu
620   </saml:Issuer>
621   <saml:Subject>
622     <saml:NameID
623       Format="urn:oasis:names:tc:SAML:1.1:nameid-
624 format:X509SubjectName">
625       C=US, O=NCSA-TEST, OU=User, CN=trscavo@uiuc.edu
626     </saml:NameID>
627   </saml:Subject>
628   <saml:AttributeStatement>
629     <saml:Attribute
630       xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
631       x500:Encoding="LDAP"
632       NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
633       Name="urn:oid:1.3.6.1.4.1.1466.115.121.1.26"
634       FriendlyName="mail">
635       <saml:AttributeValue
636         xsi:type="xs:string">trscavo@gmail.com</saml:AttributeValue>
637     </saml:Attribute>
638   </saml:AttributeStatement>
639 </samlp:ManageAttributeRequest>
```

640 5.3.3 Perceived Technical issue

641 The SAML protocol currently provides two methods that enable *a service provider to retrieve attributes*
642 relating to a user *from identity provider*.:

643 • The first method is an attribute push method in which the identity provider can send attribute
644 information within the SAML assertion provided in response to the service provider's user
645 authentication request.
646 • The second method is an attribute pull method in which the service provider can use an
647 AttributeAuthority message or an AttributeQuery message to retrieve information regarding user
648 attributes from the identity provider once the user has been authenticated by the identity provider.

649

→ In both methods described, the service provider can only obtain information relating to the attributes of the user logged into the service provider.

→ There currently exists no mechanism to enable *a service provider to transmit user attributes to be stored at the identity provider*. This contribution identifies the use case of such mechanism.

The issue highlighted in this section aims at possibly extending the SAML specifications.

5.4 User ID Forwarding

5.4.1 Scenario/context

The issue presented in this section derives from a concrete case of activities performed by an operator in order to define and implement a “security architecture” for its SOA middleware infrastructure.

This section addresses potential issues within the OASIS Web Services Security specification (**[WS-S 1.1]**.

Specifically such issues/limitations are related to the necessity of forwarding the User ID across the SOA Infrastructure.

5.4.2 Use Cases

In order to better describe the potential technical issues, hereafter a use case is presented (ref. Figure 13), with two possible different example scenarios. The use case is that of a Web Service exposed by an Application Provider, and the scenarios are:

- Customer Care portal accessed by both operator customers and personnel (Call Center Operators), each of them having different “rights” on accessed data.
- Telco Messenger Service accessed by different MVNOs (Mobile Virtual Network Operators), each of them having different “rights” on accessed data.

Use case Description

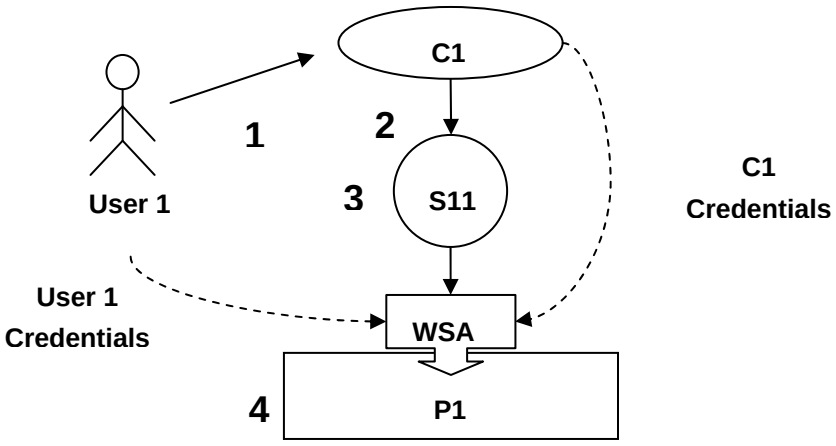


Figure 13: User ID Forwarding use case

1. User 1 accesses a front-end application (C1) using his Credentials (i.e. SSO Token).
2. C1 invokes a Web Service (WS-A) exposed by P1 and passes the User’s credentials (i.e. SAML Assertion) and its credentials (i.e. X.509 Certificate) for XML Encryption and XML Signature (WS-Security 1.1).

3. S1 (Security Enforcement Point) handles the invocation message and enforces the AAA policies:
    a. It validates C1 X.509 Certificate.
    b. It verifies the XML Encryption and Signature using the public key of C1.
    c. It verifies if C1 is authenticated & authorized to access the WS-A (C1 X.509 Certificate).
    d. It verifies if the SAML Assertion and User's token are still valid.
    e. It verifies if User 1 is authenticated & authorized to access WS-A.
4. P1 (Provider) runs the business logic.

5.4.2.1 Customer Care portal accessed by both operator customers and personnel (Call Center Operators)

C1 is a Portal for Customer Caring that consumes a Web Service (WS-A) for retrieving profile information. It is used by both Customers (for Self Caring) and Call Center Operators (ref. Figure 14).

Some of the available information such as: incoming and outgoing calls, personal information or credit cards details are ruled by privacy policies.

Obviously WS-A and all its operations are accessible by C1 but information provided as result or specific details depend on the original requester: a Customer could have full access on all information and details available on its profile while a Call Center Operator could be granted to view only a subset such data (i.e. partial call numbers, filtered credit cards details, etc.).

In the following scenarios C1 invokes WS-A for retrieving the list of incoming call numbers for specific customers:

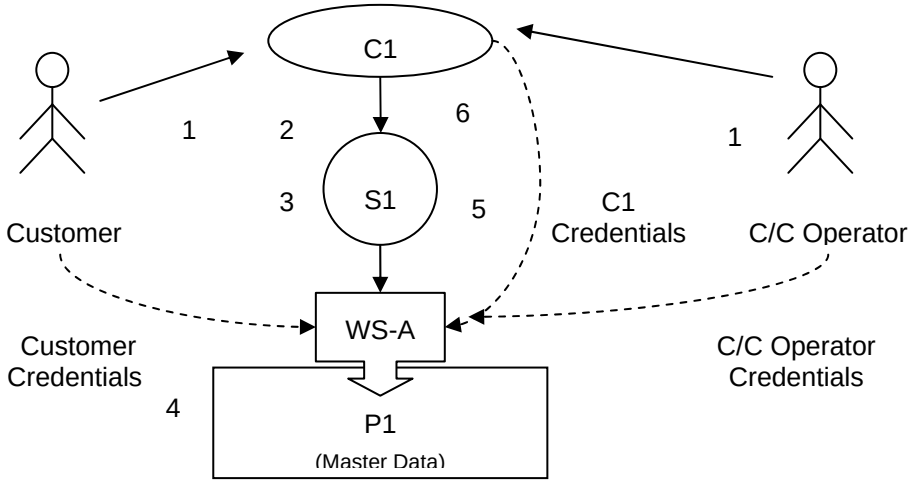


Figure 14: User ID Forwarding – "Customer care" use case

Scenario 1 (Operator's Customers)

1) A Customer accesses C1 to view the list of outcoming calls by using his Credentials (i.e. SSO Token).
2) C1 invokes a Web Service (WS-A) exposed by P1 passing the Customer's credentials in a SAML Assertion and using its X.509 Certificate for XML Encryption and XML Signature (WS-Security 1.1).
3) S1 (Security Enforcement Point) handles the invocation message and enforces the AAA policies:
    a. It validates C1 X.509 Certificate,
    b. It verifies the XML Encryption and Signature using the public key of C1,
    c. It verifies if C1 is authenticated & authorized to access the WS-A (C1 X.509 Certificate),
    d. It verifies if the SAML Assertion and User's token are still valid,

e. It verifies if operator Customers is authenticated & authorized to invoke WS-A and what level of information could access.

4) P1 (Provider) runs the business logic.
5) S1 receives the result from P1 and applies all the privacy policies in order to then return the data to C1
6) C1 shows the entire results to Customers such as:

03/27/09	11:39	3355799553	05:37
03/27/09	12:03	3359955125	10:57.

Scenario 2 (Call Center Operator)

1) A Call Center Operator accesses to view the list of incoming call numbers for a specific customer by using his Credentials (i.e. SSO Token).
2) C1 invokes a Web Service (WS-A) exposed by P1 passing the Operator's credentials in a SAML Assertion and using its X.509 Certificate for XML Encryption and XML Signature (WS-Security 1.1).
3) S1 (Security Enforcement Point) handles the invocation message and enforces the AAA policies:
   a. It validates C1 X.509 Certificate,
   b. It verifies the XML Encryption and Signature using the public key of C1,
   c. It verifies if C1 is authenticated & authorized to access the WS-A (C1 X.509 Certificate),
   d. It verifies if the SAML Assertion and User's token are still valid,
   e. It verifies if C/C Operator is authenticated & authorized to invoke WS-A and what level of information could access.
4) P1 (Provider) runs the business logic.
5) S1 receives the result from P1 and applies all the privacy policies in order to then return the data to C1.
6) C1 shows the entire results to C/C Operator such as:

03/27/09	11:39	3355799XXX	05:37
03/27/09	12:03	3359955XXX	10:57

5.4.2.2 Telco Messenger Service accessed by different MVNOs (Mobile Virtual Network Operators)

An operator has released a new integration layer called "Services Exposure" (SE) dedicated to supply all possible services (Telco, OSS and BSS) needed to any MVNO. At the moment the operator has 2 MVNO customers which consume more or less the same services, but with different policies and SLAs ruled by specific service contacts (ref. Figure 15).

The possibility to uniquely identify the NVNO that is using a service and enforce ad-hoc policies becomes essential to enable the operator to guarantee those contracts.

In addition to that all services exposed by the Service Exposure are potentially consumable by any other operator application. Therefore the possibility to identify also the application consumer is strong requirement for an operator.

In the following scenario MVNO1 and MVNO2 invoke WS-A to send messages to their customers, but while MVNO1 can send all types of messages (i.e. SMS, Reliable SMS, MMS, email, etc.), MVNO1 can send only SMS and MMS:

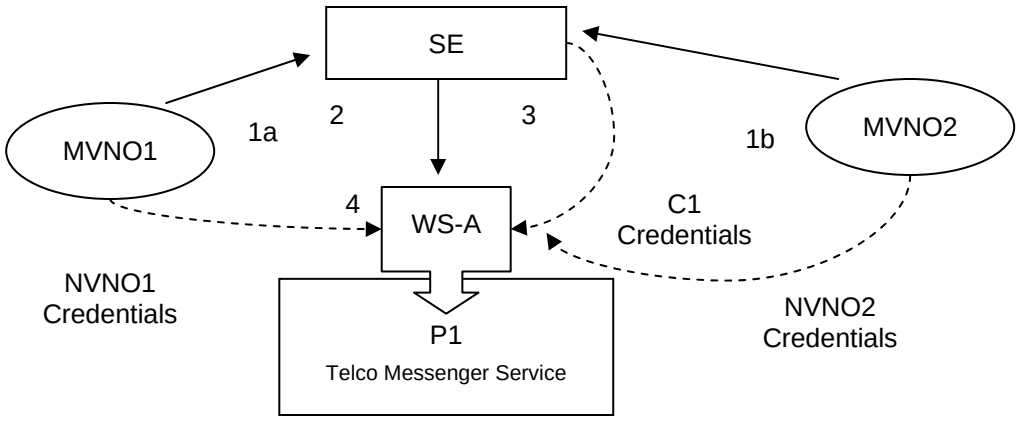


Figure 15: User ID Forwarding – "MVNO" use case

1) MVNO1 and MVNO2 invoke a service exposed by SE for sending messages.
2) SE enforce the AAA policies based on services contracts specific for each MVNOs.
3) SE verifies which types of messages MVNO1 and MVNO2 can send.
4) SE forwards the invocations to WS-A using its credentials (i.e. X.509 Certificate) and including the MVNO credentials (i.e. SAML Assertion).

5.4.3 Perceived Technical issue

At the moment it seems to be impossible to add two (or more) credentials in one message.

OASIS WS-Sec specifications **[WS-S 1.1]**, Section 6, "Security Tokens" rows 717 and 719, may offer a possibility to address the issue.

In row 717 and following it is stated:

717 */wsse:UsernameToken/wsse:Username/@{any}*

718 This is an extensibility mechanism to allow additional attributes, based on schemas, to be

719 added to the `<wsse:Username>` element.

While in row 791 and following it is stated:

791 */wsse:BinarySecurityToken/@{any}*

792 This is an extensibility mechanism to allow additional attributes, based on schemas, to be

793 added.

In any case, the solution proposed by specifications is not sufficient because, even allowing the addition of an attribute, e.g. an "Original Requester" in the specific use case, such addition would not solve the issue because it would be anyway necessary to agree the schema (protocol) amongst all actors involved in the SOA infrastructure (provided by different vendors, etc.).

This would inevitably lead to the necessity of a high customization (and consequent expenditure) of the security models.

In order to avoid costly, non-standard, vendor/platform dependent customizations and ad-hoc agreements, the operator considers that it is opportune to standardize such “protocol”.

6 Issues on Management

6.1 Introduction

The purpose of this section is to introduce to OASIS SOA-Tel TC requirements related to Service Interface cardinality and definition of metadata for Service Lifecycle Management as they emerge from the specification work in TeleManagement Forum Service Delivery Framework (SDF) program (http://www.tmforum.org/ServiceDeliveryFramework/4664/home.html).

This section addresses:

- potential limitations in the OASIS specifications that have been considered when analyzing the architectural patterns and possible implementations (such as SOA) for SDF's distributed capabilities, specifically OASIS SOA-Reference Model **[SOA RM 1.0]** and SCA Assembly Model **[SCA Assembly 1.1]**.
- potential updates to OASIS SOA Reference Architecture **[SOA RA 1.0]** as a result of the specification work developed in TM Forum SDF team, specifically:
  - additional Service Management Interface,
  - additional metadata for the support of Service Lifecycle Management.

6.2 Scenario/context

The context from which this proposal originates is the modeling and specification activities that TeleManagement Forum is performing in order to define a Service Delivery Framework. The results are published in TM Forum's SDF Reference Model (TR139v2) and SDF Reference Architecture (TMF061) documents, available to TM Forum's Members.

The TM Forum SDF objective is to manage end to end the lifecycle of services including cases where services have dependencies they can not manage and cases where services are the result of dynamic and static composition across service ownership/governance domains.

A Service Delivery Framework must respond to most actual management needs of Service Providers while Services increasingly diversify:

- manage a Service the same way, whether it comes from network, web or IT resources,
- manage a Service the same way, whether it is retailed, wholesale or operated in-house,
- manage compositions of Services when each Service may be owned by separate entities (organizations, Service or Content Providers), including the relationship that must exist among these entities,
- manage multiple versions of a Service.

6.3 Services exposing Management Interface

The complexity of Service Providers business and operations requires a Service to be managed close to the context in which it is used in order to understand who is using the service, eventually change service parameters to adapt to its usage, measure in real-time the quality of each interaction with the service, check on service status, etc.

A Service may have multiple capabilities, some of which may be used for functional purposes some for management purposes, depending on the context in which the service is used.

To fulfill TM Forum SDF's goal of E2E service lifecycle management, the TM Forum SDF team considers as Service model one where the Service exposes its manageability capabilities by means of a specific Interface, following the pattern in Figure 16.

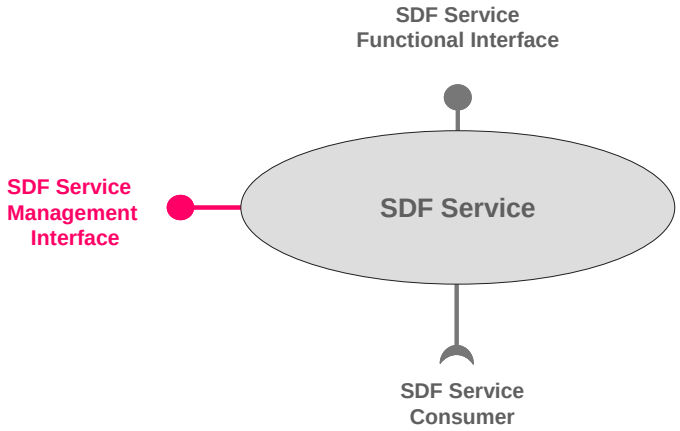


Figure 16: TM Forum "SDF Service"

In this model, the SDF Service capabilities are exposed and consumed through the SDF Functional Interfaces (SDF FI) while the management capabilities/operations of the SDF Service are available through the SDF Service Management Interface (SMI). SDF Service may consume other Services through yet another, consumer type, interface (ref. Figure 17).

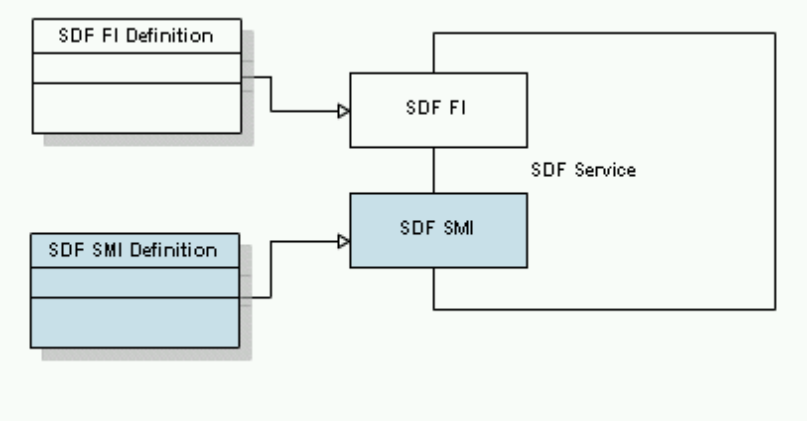


Figure 17: Including management capabilities definition in the SDF Service description

The reasons for the separation and exposure of manageability capabilities at another interface (SMI) are:

- Management capabilities are consumed by other type of (specialized) consumers (e.g. support services) with different policy/security rules than consumers of functional capabilities
- Some higher level operations and business around services can be simplified by ignoring "layers/levels" at which functional capabilities of services may be embedded, and access directly their management capabilities.

6.3.1 Perceived Technical Issues

The OASIS documentation defines Services in SOA RM and Service Components in SCA as if the cardinality of Service Interface is 1 and only one.

> **]SOA-RM 1.0]**: (Section 3.1) “A service is accessed by means of **a** service interface (see Section 3.3.1.4), where the interface comprises the specifics of how to access the underlying capabilities.”
> **[SOA-RM 1.0]**: (Subsection 3.3.1.4) “**The** service interface is the means for interacting with a service.”
> **[SCA Assembly 1.1]**: “A Service represents **an** addressable interface of the implementation.”
> Note – SCA definition for Service may be a consequence of the SOA-RM definition, we do not know

---

Moreover, for those implementers who use WSDL to describe services, the W3C **[WSDL 2.0]** primer document, (section 5.4) states that, “wsdl:service specifies only one wsdl:interface ()”.

We are aware of the solutions presented by W3C but these solutions are not standardized.

Following these documents it seems to be impossible to have two or more interfaces for a SOA Service. At the same time, SOA RA document acknowledges that “In fact, managing a service has quite a few similarities to using a service” hinting that a management of a service should happen at an interface. The same document offers though another solution (separation between management services and non-management services) which we will discuss in the next use case.

> **[SOA-RA 1.0]** (3137 – 3140) “In fact, managing a service has quite a few similarities to using a service: suggesting that we can use the service oriented model to manage SOA-based systems as well as provide them. A management service would be distinguished from a non-management service more by the nature of the capabilities involved (i.e., capabilities that relate to managing services) than by any intrinsic difference. “

Today many management capabilities are bundled with the functional interface of the service description which makes management of services very hard. This situation poses a problem for suppliers who would like to follow a SOA path for their SDF solutions. For example,

- how can they take already existing SOA Services and make them SDF Services?
- Can a SOA Service work with a Management Interface and a Functional Interface?

In TM Forum, the MTOSI team created multiple (coarse and fine grain) web services as alternative to multiple interfaces (http://www.tmforum.org/BestPracticesStandards/mTOPMTOSI/2319/Home.html). There is a need to specify that all these WS-es are related (e.g. allow access and interaction with the same Inventory and its elements).

TM Forum SDF team is seeking reconciliation on this matter and asks about possibilities to express the SDF Service and its SMI using SOA Service model.

TM Forum SDF team is also seeking alignment of its SMI addition to a Service model with the work developed in OASIS WSDM – MOWs.

6.4 Metadata in support of Service Lifecycle Management

In TM Forum’s SDF Reference Model (ref. Figure 18) (ref. TM Forum TR 139 v 2) the lifecycle management of an SDF Service is supported by other services created to fulfill the needs of business and operational processes.

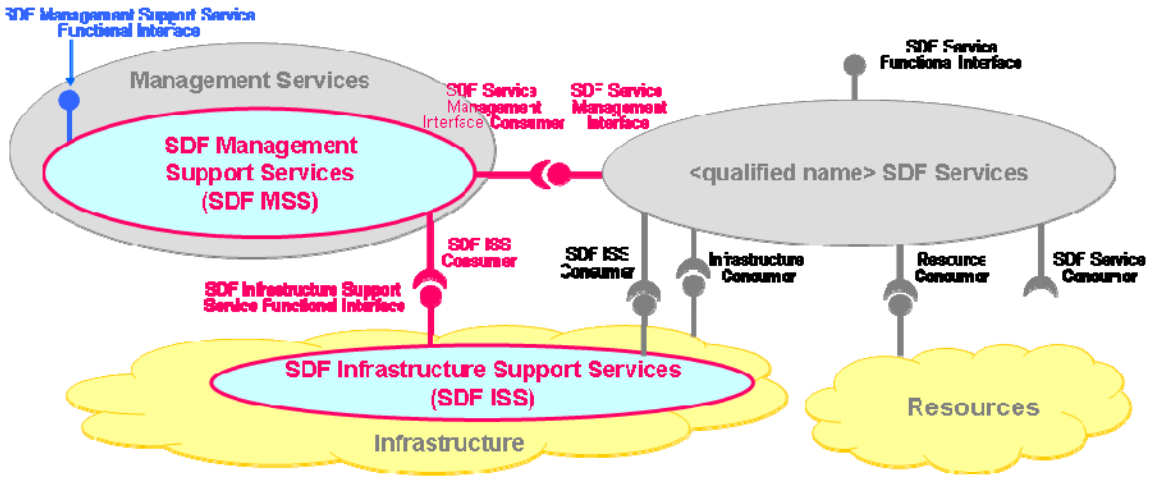


Figure 18: SDF Reference Model

- **SDF Management Support Service** (SDF MSS): An SDF Management Support Service (SDF MSS) consumes the SDF SMI of a SDF Service to manage the SDF Service. Examples of SDF MSS-es are Activation/Configuration, Problem management, Service Quality Management.

- **SDF Infrastructure Support Service** (SDF ISS): An SDF ISS provides reusable functionalities, exposed via functional interface(s), to support the SDF. Examples of possible SDF ISS are: Catalogues, Metadata repository, User Profile.

In agreement with the OASIS **[SOA RA 1.0]** (3137 – 3140) paragraph mentioned in section 6.3.1, SDF RM shows that these supporting services are of the same nature as the SDF Service itself, the only difference is that they "manage" or help in managing the SDF service (e.g. helping is the role of ISS Services). But these services need to be managed at their turn. For this reason, SDF Support Services follow the same pattern as the SDF Service: they have both **a functional and a management interface**.

Specialization in supporting and managing a service during its whole lifecycle requires finer granularity knowledge about that service: properties, supported actions or operations, possible states as well as contracts that may govern interactions with the service (including pre and post conditions for these interactions), what is the "architectural" style for service "composability", what are its dependencies or what is the level of exposure for its functional capabilities.

The proposed model for the TMF SDF SDF Service is complemented by additional data representation (metadata) in support of SDF Service lifecycle management (ref. Figure 19 and Figure 20). This new data representation containing information about the service in various phases of its lifecycle, aims at covering current gaps in the information available for the purpose of service management (e.g. what is already covered by the SOA Service description) in the overall context of Service Provider's business and operations. Moreover, this metadata is dynamic: it may change from one phase to another of the SDF Service lifecycle.

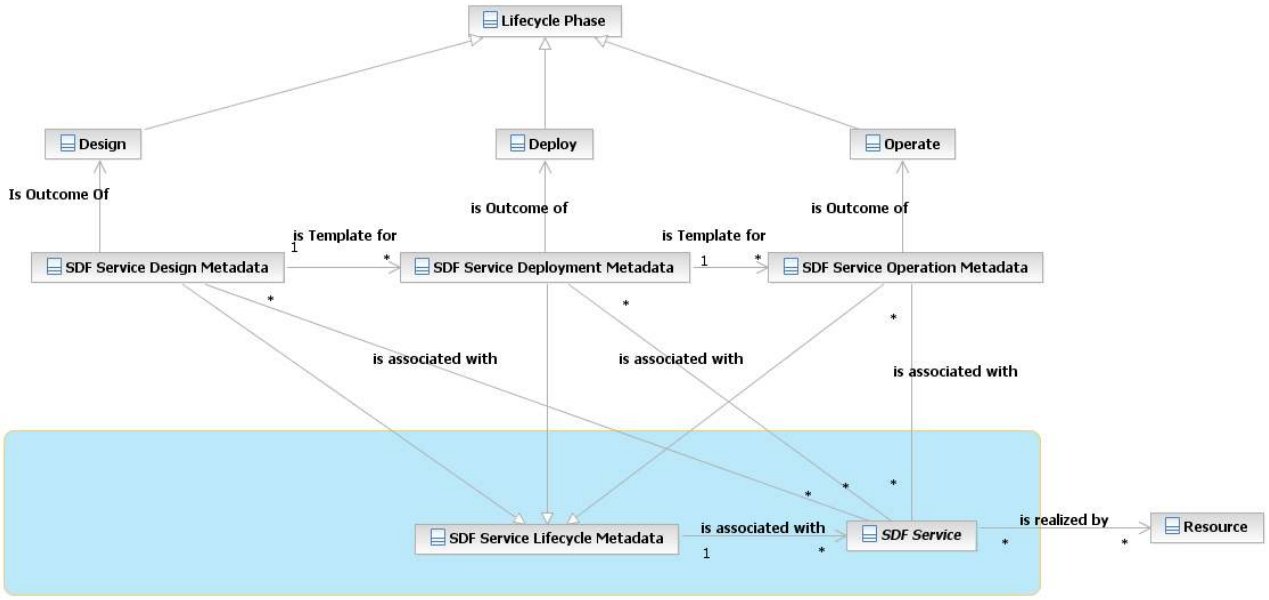


Figure 19: SDF Service lifecycle phases and associated metadata

The SDF Service Lifecycle Metadata consists at least of:

1. **Additional information about the SMI of a SDF Service** (properties, actions);
2. **Management Dependencies of the SDF Service**, including cross-domains dependencies;
3. **Management State** of the SDF Service.

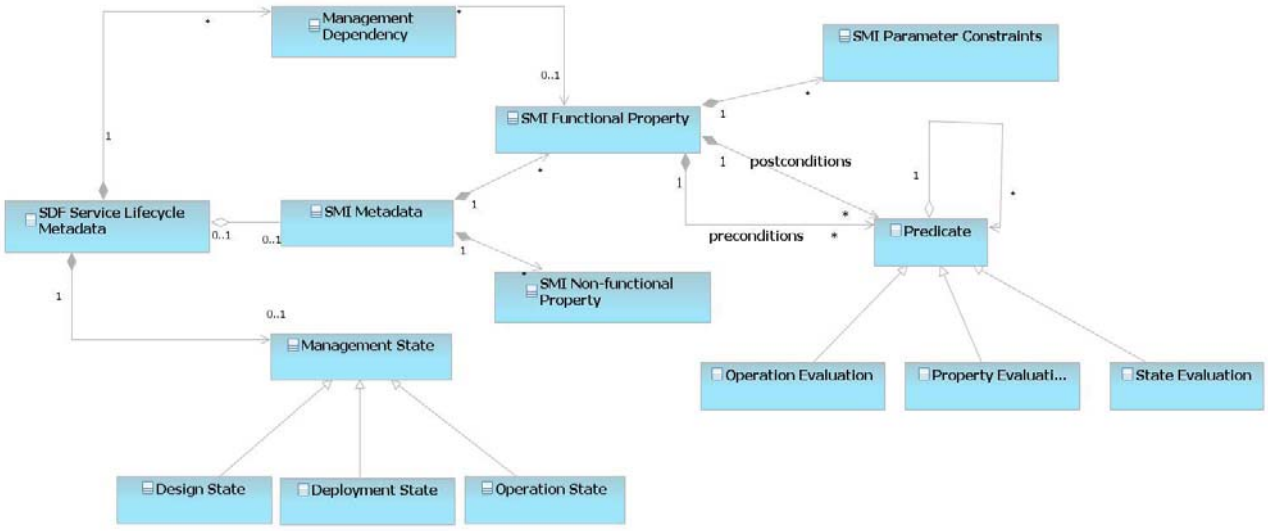


Figure 20: SDF Service Metadata (concepts)

The way this metadata is used by SDF Supporting Services to manage an SDF Service during its lifecycle is depicted below (ref. Figure 21).

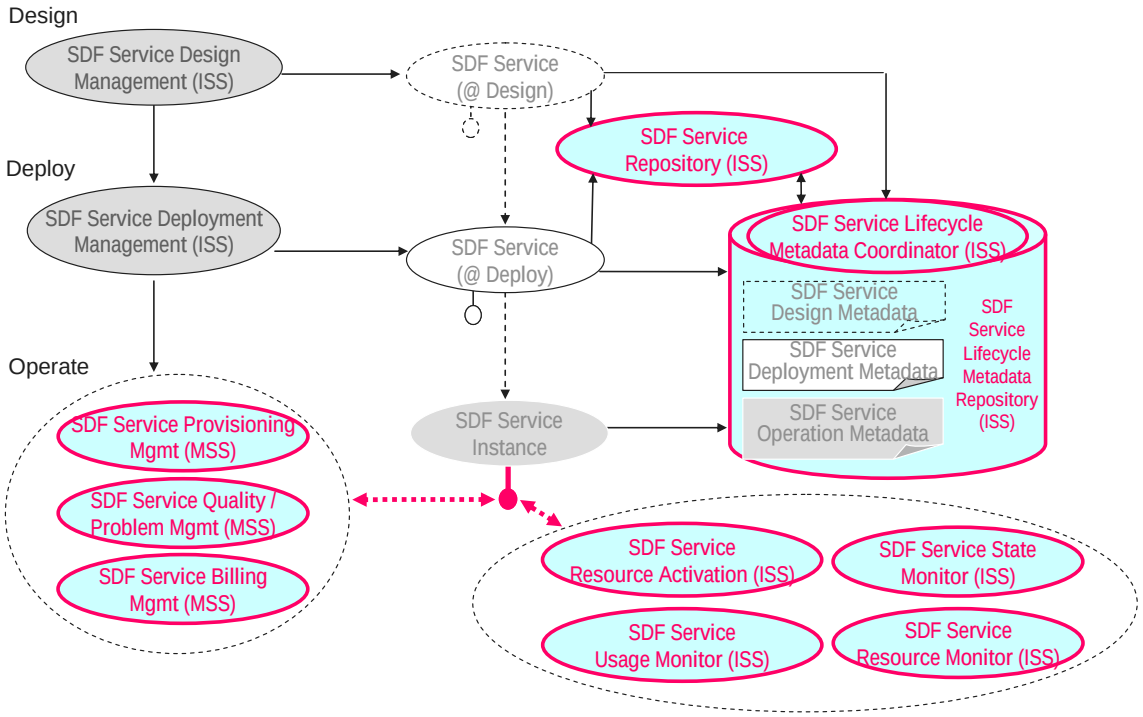


Figure 21: Service Lifecycle Management through SDF

6.4.1 Perceived Technical issues

The purpose of TM Forum work is not to duplicate existing work but to add to it that part that is necessary for service lifecycle management. The information representation (metadata) that TM Forum SDF team has identified as necessary for SDF Service Lifecycle Management, as well as its evolving nature, do not seem to be modeled in the current SOA Service Description Model and supported by the Management of Services approach described in **[SOA –RA 1.0]** document. TM Forum SDF Team believes that modeling service dependencies including dependencies across ownership/governance domains is important addition to the SOA RA.

TM Forum SDF team is seeking OASIS expert advice on what to do. Can the additional metadata it specifies for the purpose of SDF Service lifecycle management be added to the current **[SOA RA 1.0]**, in respect to the views and the models that are already part of this Reference Architecture?

TM Forum SDF team is also seeking OASIS expert advice on aspects such as supporting versioning and compatibility of this metadata, existing architectural patterns for data contribution from various applications/sources/systems and for assurance of cohesiveness across metadata elements and along the phases in the lifecycle of a service.

6.5 Recap of issues and considerations for OASIS SOA-Tel analysis

TM Forum SDF team is seeking reconciliation on the matter of the additional service management interface and asks about possibilities to express the SDF Service and its Service Management Interface (SMI) in the SOA Service model. TM Forum SDF Team believes that distinguishing the SMI from the Functional Interface of a Service is necessary for the reasons exposed in the use case.

What is OASIS's advice on this and how can SDF Service model be realized with current SOA Services Model?

TM Forum SDF team is also seeking OASIS expert advice on positioning of its SMI addition to a Service model within the work developed in OASIS **[WSDM-MOWS]**.

TM Forum SDF team is also seeking OASIS expert advice on what should be the relationship between the SDF Reference Model and the SOA Reference Architecture - Service as Managed Entities part.

TM Forum SDF team is seeking OASIS (namely the SOA-RM, SOA-RA and SCA TCs, and possibly the WSDM TC) expert advice on how to organize and integrate the additional metadata for the purpose of SDF Service lifecycle management in the current [**SOA RA 1.0]** and do so with respect to the views and the models which are already part of this RA.

TM Forum SDF team is also seeking OASIS expert advice on aspects such as supporting versioning and compatibility of metadata, existing architectural patterns for data contribution from various applications/sources/systems and for assurance of cohesiveness across metadata elements and along the phases in the lifecycle of a service.

7 Issues on SOA collective standards usage

7.1 Common Patterns for Interoperable Service Based Communications

7.1.1 Scenario/purpose

There is a growing set of application models that serve a general web and mobile market and consequently can only expect a web application pattern and can not make any assumptions of the protocol stack other then IP. These applications are no longer exclusive to the public domain. Applications in the enterprise are adopting these new computing models, seamlessly moving between internal and external clouds trying to leverage the elasticity that the model offers and blending application oriented communications across these boundaries. Such applications are typically designed to support highly functional virtual and often transient partner/ end user/ customer relationships.

Users in these models expect access to information anytime, anywhere and will expect the enablement of communications within that context of any application to be delivered in the same way. Ubiquity of communications as a part of this set of internet type applications, LAN attached or mobile, needs to allow for interoperation across a definable set of standards and device types in order for it to achieve the same universality as the supporting application models, bringing seamless communications utility across different communication domains and applications.

In such models, the application can only make general assumption about the device attributes and protocol stacks these devices support. Ubiquity of communication within the application model calls for device information and communications channel setup to be ascertained thru the process of user/ device connecting to the application. In some situations the application may not be directly involved in setting up media, in other cases it will either need to participate, at least in part or entirely. An application may even have to make decisions as to the best choice of path of delivery.

Achieving ubiquitous access to application resources irrespective of network domain is often a function a combined collection of standards working in unison (i.e. profile) providing consistent patterns to access applications resources. Consistency in approach across different media and control paths, client types and application domains is essential to foster larger a eco-system of co-operative applications for the user across different network and application domains. Hence, the patterns supporting the discovery, setup and delivery of communications within the context of a set of applications needs to be normalized in order to enable interoperable solutions across heterogeneous environments.

Enclosed is an example:

- An Independent collision appraisal company has independent collision agents that broker across separate suppliers on behalf of many insurance companies, auto suppliers and collision repair shops. The agents choose which suppliers to use based on their locale and relationships but these are under a lot of change.
  - No one company owns and controls the type of agent device.
  - Agents typically search a few supplier sites for any given situation. They expect to be able to quickly call and have the context of the part/order be available to any parts supplier, insurance company and collision shop they use. The agent may further use media (picture, video) to support and verify the parts needed with the supplier.
  - The applications from different companies support different service profiles (voice, video, picture, and data) to deliver the capability. Real Time communications is supported thru variable means including but not limited to, SIP, Jingle or simply an RTP stream controlled directly by the application.
  - A standard means application communications profile needs to be delivered in order to allow any agent and device to work in the context of a set of independent applications from different suppliers

The market in general needs a normalized means to establish communications to the endpoint without being prescriptive at the endpoint. Applications need greater control over the different choices to be made given multiple network paths and options. An application requesting a connection should be able to adapt seamlessly to the network environment and protocols used to set up the communications channels. In addition, external tools such as BPEL, BPM and ESB should be able to leverage this common foundation to incorporate communications processing. This is important for broader adoption of communication as a service using well known patterns and skills. Figure 22 depicts the case.

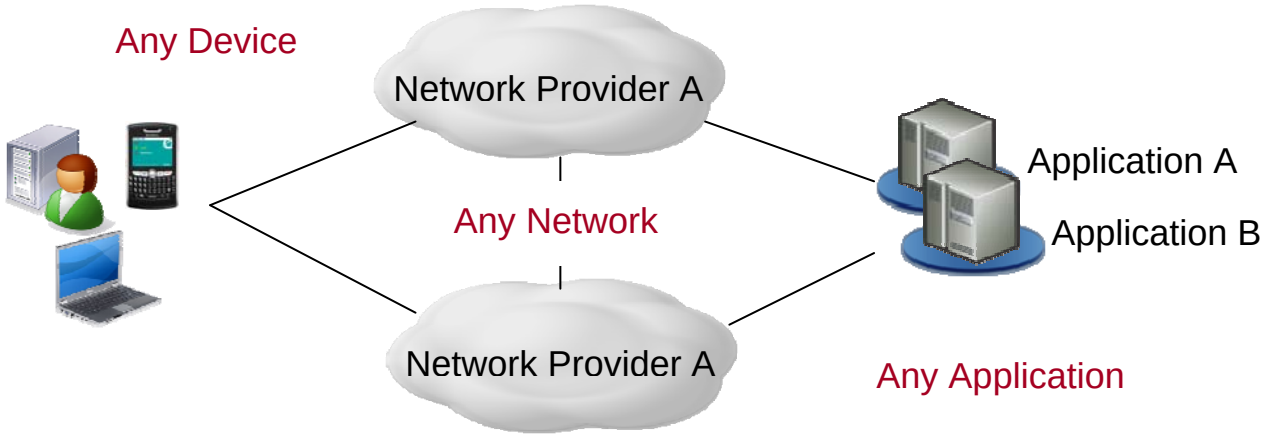


Figure 22: Real-time communications in the context of an “any” application seamlessly across any device and network

The following is a minimum set of requirements:

1. **Universal service discovery/ dynamic bindings**
2. **Bi-directional, full duplex control across different modes of communication thru web service interfaces**
3. **Common support for asynchronous interactions with event subscriptions and notifications**
4. **Means to associate application context with stateful communication interactions (i.e. session)**
5. **Common communication information model enabling connection negotiation.**
6. **Common patterns for client web services to work within a SIP and XMPP context.**
   - **Integrated control of media delivery (transport channels and their parameters)**
   - **Control of communications channel, events for that session**

Items 1, 2, 3 and 4 above target a common set of web service infrastructure requirements to generically set up communications. Items 5 and 6 are essential to handle differences (e.g., between a SIP or Jingle, etc based endpoints) thru the service interface.

7.1.2 Scenario/context

This use case involves a simple web application that connects to the site, pulls down a list of people to contact and allows the user to click-to-call. Assume a simple model where JavaScript is downloaded to the client and sets up the web service call to a communication service with the URI provided. The sequence diagram in Figure 23 depicts the case.

The use case defines a simple setup of a voice connection for one side of the connection. More complex types of communication scenarios (e.g. conferencing, video) and multi-modal interactions (e.g. voice with chat sessions) should be supported with the same pattern. All applications need a common means to set up different ports supporting different types (voice, pictures) or multiplex thru one port but can not assume one standard or protocol stack is at play as they do not know who and what type of device is going to connect. A server based model implies that communications is handled at the server (i.e. server connects client A to client B) where as the client model is more p2p. Each mode must be generally supported by the pattern.

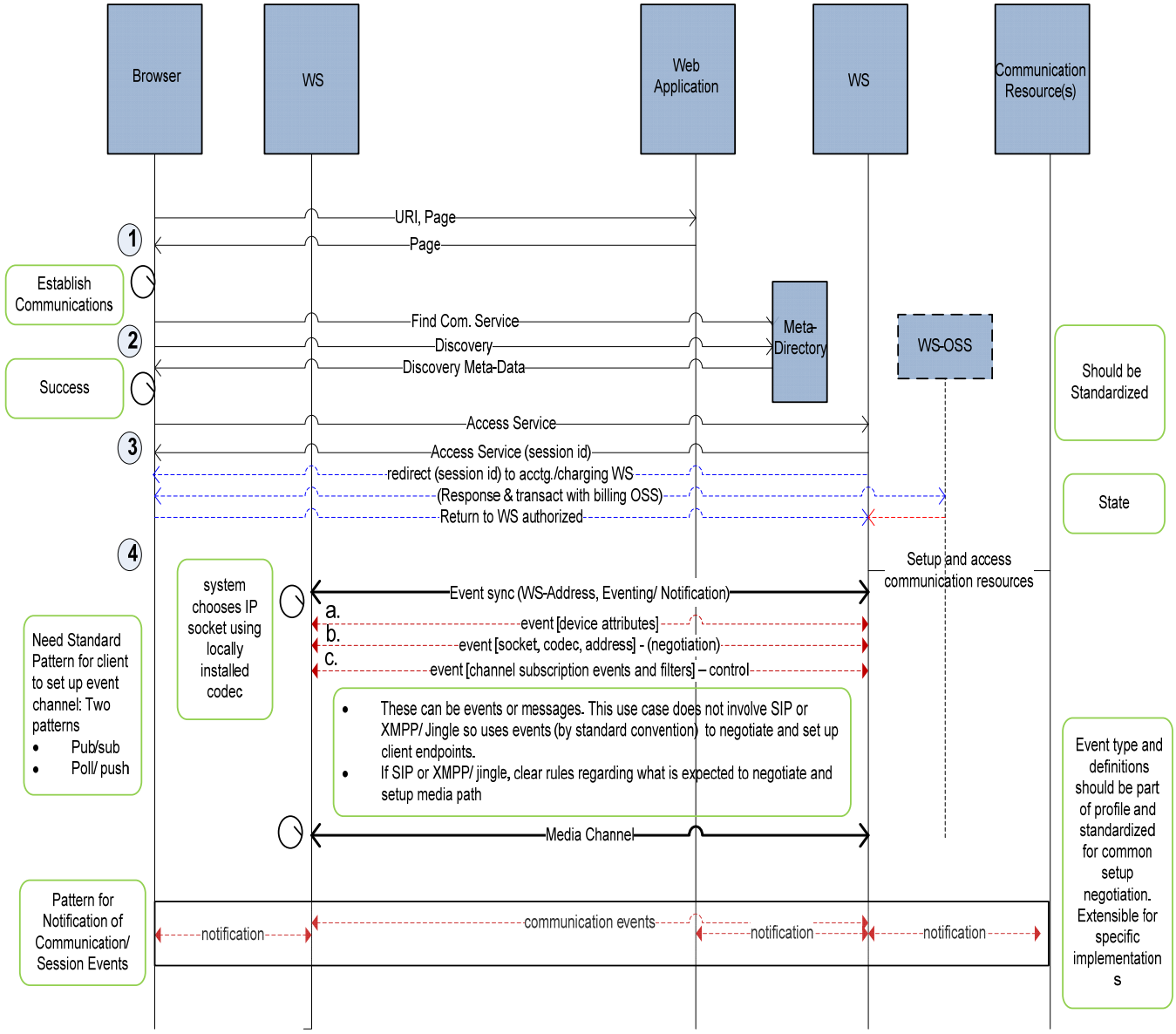
The pattern discussed in this use case can equally be applied to REST type models using Restful API mechanisms. This use case will confine itself to a web services client/ interaction model. It is important to understand that whichever programming model used for the application, for generally application interoperability across domain, the application model for communications needs to be consistent. Lastly, some of the interface discovery complexity could be handled thru a commonly defined interface used across vendors. Lack of such an agreed upon model, places more complexity in the meta-data needed to describe what services handle what type of communications (i.e. voice or video connection, conference, etc,) and more importantly describing the events types and data structures across the wire. This use case does not go into detail the interactions for device attribute and/or interface discovery.

The basic interaction in this use case involves a web service interchange enabling the setup of a communications channel exclusively. In this case we are selecting a communication channel that is a proprietary RTP enabled socket controlled by the application. Hence, events need to be exchanged to inform, negotiate and select the address on each side, the real time protocol used, the codec and other pertinent information. The same negotiation process can be used to select a SIP or XMPP/ Jingle based media channel when device attributes and condition warrant. In this latter case, these protocols would negotiate the information on their own, freeing the service itself from this activity.

Looking at this pattern we see that the set of requirements for the web services infrastructure (i.e. standards) within the context of communications is clarified. We need a standard means to establish a multimedia channel supporting real-time voice and video exclusively thru the web but also allow for variation to support other approaches. This allows a higher degree of inter-operability across different business and network domains. The standard pattern promotes common skills, behavior and tool integration. It fosters development consistency, simplicity driving wider adoption and most important, allows providers to offer solutions that work in the context of an inter-operable cloud.

1107 **Use Case Sequence Diagram:**

1108



1109
1110

1111 Figure 23: Sequence diagram example for the Universal Communication Profile case

1112

Use Case Steps:

1. The communication responds back with a session id for the context of the application within a communication channel.
2. A bi-directional web services interface is set up to receive events for this session id.
    a. Client looks up service meta-data and discovers interface, binding, events and capabilities of service. (i.e. WS- meta data and WS-policy)[1].
    b. If there is no clear interface specification (i.e. CSTA, Parlay-x, other) then a very robust meta-directory and policy infrastructure is needed to support the interface variations across vendors.
    c. Connection is attempted. This may trigger events such as subscription authorization or pay-as-you-go. This results in redirecting to a billing-OSS WS that engages the client over the event-channel for payment methods and payment completion – leading to a notification and return to the service-WS for further service delivery/denial[2].
3. Client connect to WS
    a. Event channel is set up.
    b. This event channel is overlaid with a subscription interface allowing each side to subscribe and filter as necessary specific events needed for the communications.
        i. Model needs to support timely and reliable delivery of events
        ii. Model needs to support events delivered in specific order
4. Client sends event indicating its device characteristics, communication modes (SIP, Jingle, etc.)[3].
    a. Connection is made using “proprietary” socket. Application has designed the separation of different types (i.e. picture, video, voice) and it manages the parsing and reformatting of each for the application.
        i. User is in voice session
        ii. User is in transmitting pictures
    b. Server sends event indicating the mode it wishes to use given the device attributes.
        i. If SIP or XMPP/ Jingle client, negotiation of codec and address via those standards but information (i.e. session description) is delivered to client application thru the web service. The application sets up and controls the media, creates SDP response and defines RTP port
    c. In this simple case we are using RTP with session description/ negotiation being handled thru WS event channel.
    d. Client sends event to WS indicating what connection processing events it is interested in. In this case it asks for connection, disconnect, hold/resume for picture and mute/un-mute for events.
    e. Remote user presses hold for picture. Event is propagated to device and picture transmission is held

---

[1] Note: IETF work and SIP media and session policies stds (xml-based; can be realized as derived schema of the ws-policy core). Same goes for security policy (though ws-security-policy as it is is restricted to only policies for ws-security standards.).

[2] This step is but an example interaction of several possible generic pre-communication events. In-communication and post-communication events are also conceivable.

[3] Note: Any WS-standards here or is it an area that the SOA-TEL TC can develop schema for?

Since service architectures are inherently transport neutral, we can not rely on any underlying means (i.e. TCP) to manage the session lifecycle. We do not imply any particular means in this example to establish statefulness at either point across the wire, just a means to set up and convey the information across any channel.

It is our intention to first look to see if this is a common pattern across all communications services and to identify the relevant standards that can be used and/or need to extend to support the need. Once explored for web services we can extrapolate this to a common set of patterns for a broader set of service interface types.

7.1.3 Technical Issues/ Solutions:

The purpose of the above uses case is not to prescribe a solution but what a solution may need to look like in the context of the problem. The problem is basically that in order to deliver ubiquitous mobility and interoperability to users, applications can not be bound by a single network provider nor underlying assumptions on the real-time protocols used. Access to real-time communications needs to be normalized across set of common access patterns in the context of any given application. The process is not disjoint; application and communications need to work in context to deliver full effectiveness. Access to the application resource requires the discovery the right pattern without any pre-defined assumptions about the underlying network. The application also needs to be able to make decisions as to the best path in multiple paths exist based on policy, cost, quality and device attributes.

Service orient architectures are in principle about decoupling the underlying transport form the delivery of the application resource. This principle needs to be hold for access to applications / services and real time communications used in the context of any application allowing for common access across a broad set of applications.

8 Conformance

The objective of this document is to collect potential technical issues and gaps of SOA standards utilized within the context of communications service providers, in order to enable subsequent development of requirements for the solution of such issues.

As such no conformance clauses apply to this document.

Appendix A. Acknowledgements

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

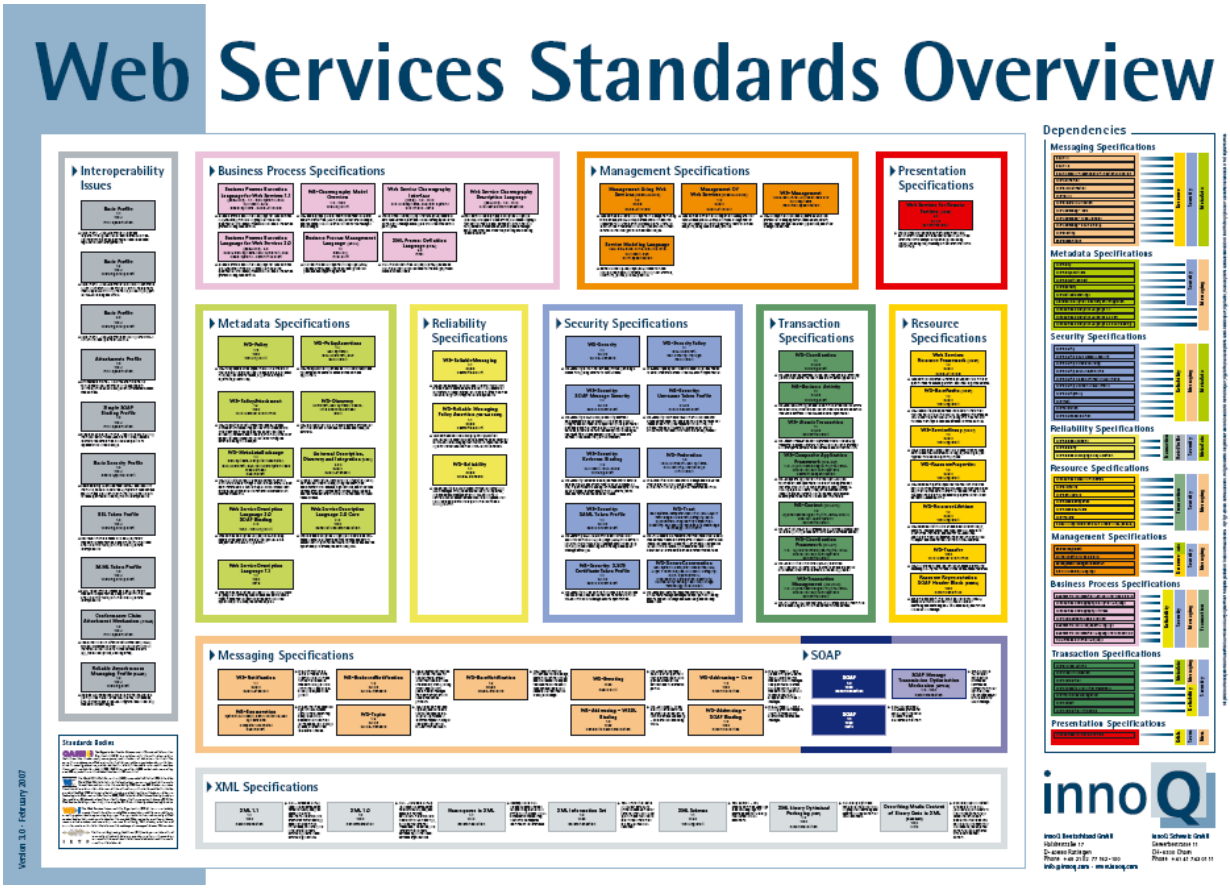
Participants:

Mike Giordano	Avaya
Liu Feng	Avaya
Mahalingam Mani	Avaya
Ian Jones	BT
Sami Bhiri	Digital Enterprise Research Institute (DERI)
Paul Knight	Individual
Lucia Gradinariu	LGG Solutions
Orit Levin	Microsoft
Joerg.Abendroth	Nokia Siemens Networks
Christian Guenter	Nokia Siemens Networks
Thinn Nguyenphu	Nokia Siemens Networks
Olaf Renner	Nokia Siemens Networks
Abbie Barbir	Nortel
John Storrie,	Individual
Vincenzo Amorino	Telecom Italia
Luca Galeani	Telecom Italia
Maria Jose Mollo	Telecom Italia
Vito Pistillo	Telecom Italia
Enrico Ronco	Telecom Italia
Federico Rossini	Telecom Italia
Luca Viale	Telecom Italia

Appendix B. Web Services Standards Landscape

This section is non-normative.

The following diagram shows a possible representation of web services specification landscape, and is available at http://www.innoq.com - **[WS Landscape]**.



Appendix C. Possible workaround related to issue in Section 3.1 “Transaction Endpoints Specification”

This section is non-normative.

This issue described within Section 3.1 could be solved with the following “workaround” solution, which in any case is not mandatory but exploits some “optional” features of WS-Addressing.

Note:

- This proposal does not require any “persistence” on any intermediary and is fully compliant with WS-Addressing specification.
- The TC asks if, apart from the proposed workaround, there is another standard reference solution for the highlighted problem.

Should there be no other solution apart from the proposed workaround; **the proposal is to extend the WS-Addressing specification in order that the “Message Properties” include a new tag (provisionally named “Final Destination”) to specify the process/transaction result.**

Moreover the proposal is to make the utilization of this new tag as Mandatory whenever it is necessary to specify a “final destination”, i.e. in presence of a non-direct “requester-consumer” situation.

Proposed Workaround:

CASE **A**:

1. **C1 invokes WS-A** and specifies in the *replyTo* section of the WS-Addressing header the *EPR (Endpoint Reference)* where it wants to receive the asynchronous response **(C1)**. (Example: http://service1.sc.local/response).

2. The **ESB invokes WSB** and specifies in the *replyTo* section of the WS-Addressing header the *EPR (Endpoint Reference)* where it wants to receive the asynchronous response (Example: http://service1.esb.local/response). By doing so it takes the *replyTo* section received by C1 and embeds it in the *referenceParameters* section of *replyTo*. P1 is obliged by WS-Addressing specification to return the *referenceParameters* in the *To* section when sending the asynchronous response.

3. **P1 returns the asynchronous response** to the replyTo address (Example: http://service1.esb.local/response) specified by the ESB, together with the *referenceParameters* section.

4. The **ESB invokes WSC** and specifies in the *replyTo* section of the WS-Addressing header the *EPR (Endpoint Reference)* where it wants to receive the asynchronous response (Example: http://service2.esb.local/response)**.** By doing so it takes the *referenceParameters* section received by WSB and embeds it in the *replyTo* section. P2 is obliged by WS-Addressing specification to return the *referenceParameters* in the *To* section when sending the asynchronous response.

5. **P2 returns the asynchronous response** to the ESB *replyTo* address (Example: http://service2.esb.local/response) specified by the ESB, which includes the *referenceParameters* section.

6. **The ESB gets the *replyTo* info**, embedded in the *referenceParameters* received from P2, to address the asynchronous response to **C1**.

CASE **B**:

Same as Case 1 with C2 originator and final destination.