Diploma Project

Manage Telecommunication equipment using Web Services

Web Service

Version V 1.2 (WS)

Professeurs:
Philippe Joye
François Buntschu

Mandatory:
Daniel Gachet

Expert:
Nicolas Mayencourt

Students:

Thierry Kiki
David Schneider

Table of Contents

1 Definitions

SOA:	Service Oriented Architecture
Web Services:	W3C recommendation
XML:	Extensible Markup Language
WSDL:	Web Service Description Language (W3C recommendation)
SOAP:	Simple Object Access Protocol (W3C recommendation)
UDDI:	Universal Description, Discovery and Integration (directory service)
W3C:	World Wide Web Consortium
OASIS:	Organization for the Advancement of Structured Information Standards
IETF:	Internet Engineering Task Force
WS-I:	Web Services Interoperability Organization

2 Initiation

2.1 Traditional web interaction

...

USER (Browser) ↔ HTML-file Web Application

2.2 Web Service interaction

...

Application ↔ XML Web Service

3 Service Oriented Architecture

Before we introduce the Web Service Architecture, we need to mention some basics of the Service Oriented Architecture. This is necessary because the Web Service Architecture extends the Service Oriented Architecture.

W3C provides the following equation which interconnects the two words:

World Wide Web (WWW) + Service Oriented Architecture (SOA)
= Web Service Architecture

3.1 Service Oriented Architecture

The name indicates the basic idea behind this architecture, it is service oriented. We will not describe the SOA in detail, more information can be found under the references mentioned.

Main advantages of the SOA are that it facilitates manageable growth of enterprise systems and can reduce the costs for cooperation between organizations.

As most of the IT infrastructures and its organization have grown with a pillars-like (säulen?) architecture, the changeover to a SOA will be very difficult and time-consuming.

The following graphic illustrate this problem very well:

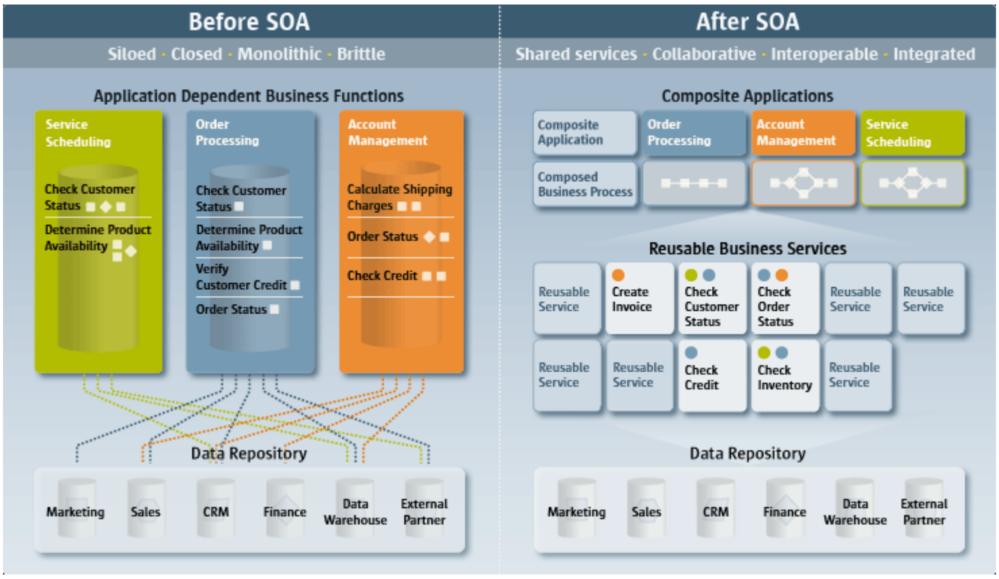


Illustration 1: Before & sfter SOA

The following illustration shows the famous triangle of Service Oriented Architectures. Roles are described briefly afterwards.

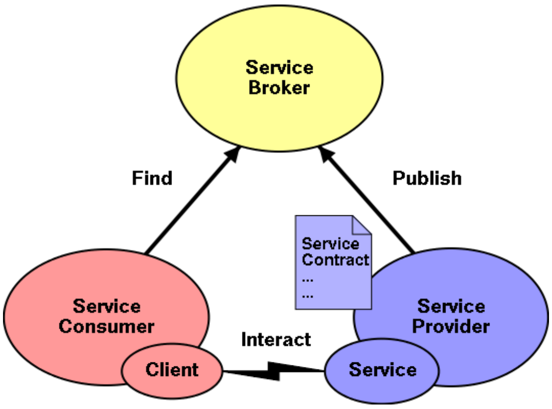


Illustration 2: Three roles in SOA

Service Provider

The service provider publishes the service. A description of the service is provided. The provider hosts and controls the access to the service.

Service Consumer

A service consumer interacts with the service via a service client. He can find services by querying the service broker. This role can be driven by an end user or by another service.

Service Broker (optional)

The service broker provides the directory service and allows service providers to publish and service costumer to find services. This role is optional, the service can also be found otherwise.

3.2 Basic characteristics of a SOA

A good summary of the basic characteristics of a SOA can be found in the technical library of IBM. The document is a recommendation to improve a Service Oriented Architecture and contains inter alia the following principles for a SOA.[1]

Guiding principles:

- *Reuse, granularity, modularity, composability*

[1] http://www.ibm.com/developerworks/webservices/library/ws-improvesoa/

- *Compliance to standards (both common and industry-specific)*
- *Services identification and categorization*

Specific architectural principles:

- *Separation of business logic from the underlying technology*
- *Single implementation and enterprise-view of components*
- *Life cycle management*
- *Efficient use of system resources*

4 Web Service Architecture

This chapter introduces the Web Service Architecture with its basic concept. We also want to introduce here the different organizations and task forces which standardize the concepts behind this architecture.
As we mentioned before, the Web Service Architecture extends a Service Oriented Architecture.

4.1 Definition

The definition of W3 published in the Web Services Architecture Requirements:[2]

« A Web service is a software system identified by a URI [RFC 2396], whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols. »

4.2 Basic Concept

The basic components of a Web Service Architecture are:

- Communication
- Service Description
- Directory Service

The W3 recommends for the communication of Web Services the use of SOAP, its specification defines the XML-based message format and how it is embedded into a transport protocol. SOAP is mostly transported over HTTP but is not at all dependent on this transport protocol.

WSDL, also XML-based, is used to describe the Web Service.

[2] Source: http://www.w3.org/TR/wsa-reqs/

Directory service specifies a standardized structure for the management of Web Service metadata. A possible directory service is UDDI. This service, which corresponds to the Service Broker of the SOA, is optional.

4.3 Standardization

W3C[3]:
Founded in 1994 by Tim Bernes-Lee at the Massachusetts Institute of Technology, Laboratory for Computer Science (MIT/LCS) with support of the CERN in Geneva, the DARPA (Defense Advanced Research Project Agency) and the EU (European Union).
Multiple task forces are engaged in standards for HTML, XML, SOAP and WSDL. Interesting for the future will be standards as RDF (Resource Description Framework) and OWL (Web Ontology Language) concerning the semantic web.

OASIS[4]:
The Organization for the Advancement of Structured Information Standards, originally founded in 1993 as a cooperation of commercial enterprises, has its focus on standards of the topic e-business. Beside Web Services they provide techniques as UDDI, ebXML(electronic business using XML) and WS-BPEL(Business Process Execution Language).

IETF[5]:
The Internet Engineering Task Force defines more technique oriented standards and is therefore less conspicuous on Web Service design tasks. The most important standards by IETF are TLS (Transport Layer Security), LDAP (Lightweight Directory Access Protocol) and IPv6 (Internet Protocol version 6).

WS-I[6]:
Web Service Interoperability Organization does not publish any standards. The focus lies on the examination of concrete specifications and the implementation of different producers and guarantee the interoperability of them.
Profiles were defined to describe how to use the implementation of the different producers.

[3] http://www.w3.org
[4] http://www.oasis-open.org
[5] http://www.ietf.org
[6] http://www.ws-i.org

5 Web Service Description

In the following paragraphs we will introduce the Web Service Description Language WSDL and link it to the interface of the MileGate we want to describe.

5.1 Introduction

WSDL is an XML language for describing Web Service interfaces. The language is standardized by the W3C.
The specification of the version 1.1 exists since 2001. The follower version (2.0) reached the status of a 'W3C Recommendation' in march 2006 but most of the current Web Services still use the previous version.

The description of the service is spit into two parties, we have an abstract and a concrete description. The abstract view focuses the functionality and the concrete enters more into the technical detail. Thus we have a separation between the details and the manner our service is offered.

The components of the description are:

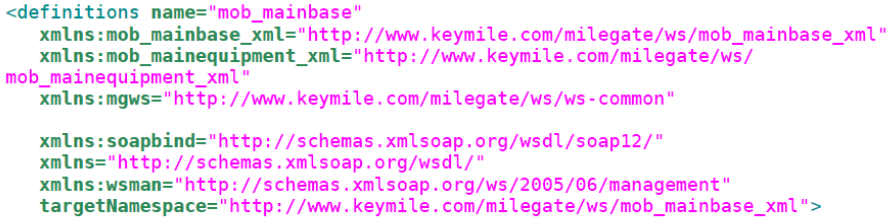
Abstract:	Operation Messages Exchange Pattern Interface
Concrete:	Binding Endpoint Service

The main difference of WSDL according to other description languages for interfaces (e.g. IDL, Interface Description Language) is that everything is concentrated in one file. We are able to communicate with the service just on the base of the WSDL file. Of course we have also the possibility to write the description modular (include, import) to provide better legibility and maintainability.

5.2 Structure of the description

We want to introduce briefly the elements used to describe the Web Service and show afterwards a few more details using the description of our interface.
If two elements are used to describe one single element, this is due to the different versions of WSDL. The first element belongs to the version 1.1 and the second to the standard 2.0.

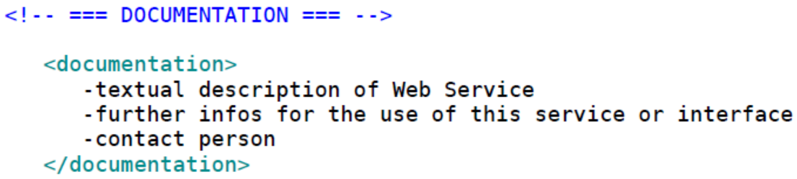
definitions / description (root element)
This XMLelement represents the root element of the WSDL file and defines the different name spaces.



```
<definitions name="mob_mainbase"
   xmlns:mob_mainbase_xml="http://www.keymile.com/milegate/ws/mob_mainbase_xml"
   xmlns:mob_mainequipment_xml="http://www.keymile.com/milegate/ws/
mob_mainequipment_xml"
   xmlns:mgws="http://www.keymile.com/milegate/ws/ws-common"

   xmlns:soapbind="http://schemas.xmlsoap.org/wsdl/soap12/"
   xmlns="http://schemas.xmlsoap.org/wsdl/"
   xmlns:wsman="http://schemas.xmlsoap.org/ws/2005/06/management"
   targetNamespace="http://www.keymile.com/milegate/ws/mob_mainbase_xml">
```

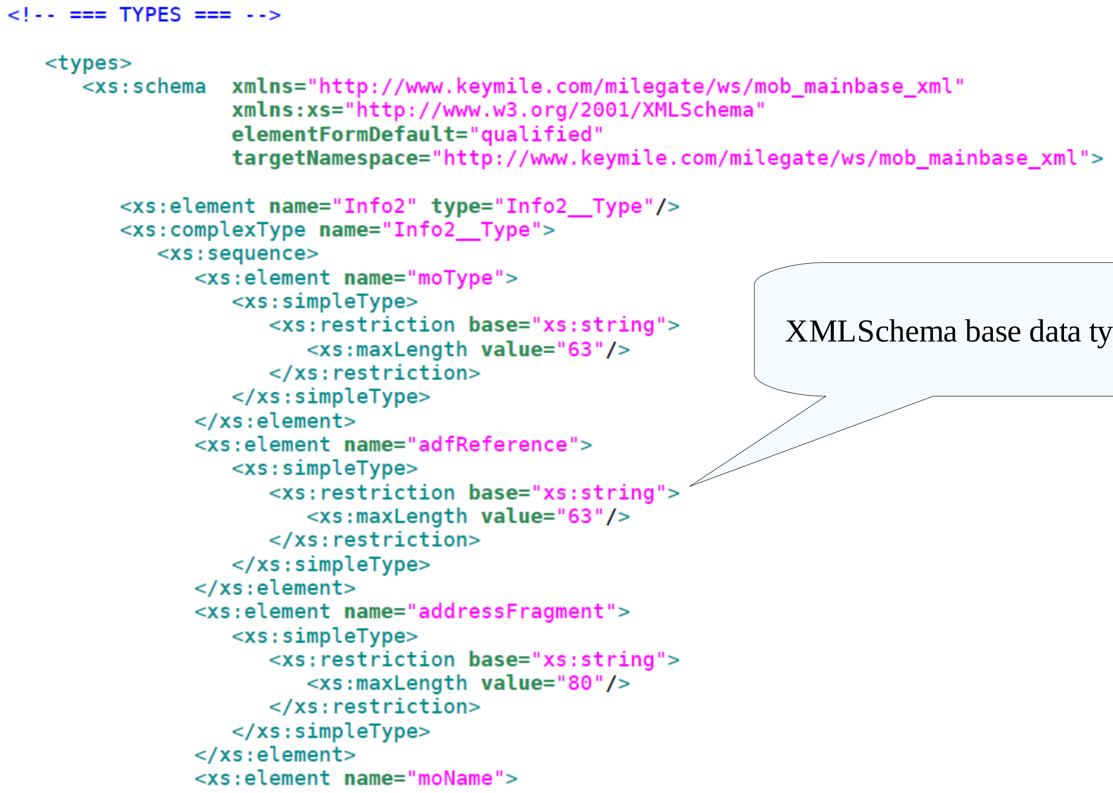
documentation
The section documentation contains a textual annotation to the service.



```
<!-- === DOCUMENTATION === -->

   <documentation>
      -textual description of Web Service
      -further infos for the use of this service or interface
      -contact person
   </documentation>
```

types
Defines the usable data types.



```
<!-- === TYPES === -->

   <types>
      <xs:schema  xmlns="http://www.keymile.com/milegate/ws/mob_mainbase_xml"
                  xmlns:xs="http://www.w3.org/2001/XMLSchema"
                  elementFormDefault="qualified"
                  targetNamespace="http://www.keymile.com/milegate/ws/mob_mainbase_xml">

         <xs:element name="Info2" type="Info2__Type"/>
         <xs:complexType name="Info2__Type">
            <xs:sequence>
               <xs:element name="moType">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="63"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="adfReference">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="63"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="addressFragment">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="80"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="moName">
```

XMLSchema base data types

.
.

```
         <xs:element name="Label" type="Label__Type"/>
         <xs:complexType name="Label__Type">
            <xs:sequence>
               <xs:element name="user">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="63"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="service">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="63"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
               <xs:element name="description">
                  <xs:simpleType>
                     <xs:restriction base="xs:string">
                        <xs:maxLength value="127"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
            </xs:sequence>
         </xs:complexType>

         <xs:element name="AlarmSeverity" type="AlarmSeverity__Type"/>
         <xs:complexType name="AlarmSeverity__Type">
            <xs:sequence>
               <xs:element name="maxAlarmSeverity" type="severity__Type"/>
               <xs:element name="maxPropagatedAlarmSeverity" type="severity__Type"/>
            </xs:sequence>
         </xs:complexType>

      </xs:schema>
   </types>
```

message

This element contains the possible messages and the types which are allowed to use.

```
<!-- === MESSAGES === -->

   <!-- WS-Management headers  -->
   <message name="ResourceURIMessage">
      <part name="Header" element="wsman:ResourceURI"/>
   </message>
   <message name="SelectorSetMessage">
      <part name="Header"  element="wsman:SelectorSet"/>
   </message>

   <!-- WS-Addressing headers  -->
   <message name="ToMessage">
      <part name="Header"  element="wsa:To"/>
   </message>
   <message name="ReplyToMessage">
      <part name="Header"  element="wsa:ReplyTo"/>
   </message>
   <message name="ActionMessage">
      <part name="Header"  element="wsa:Action"/>
   </message>
   <message name="MessageIDMessage">
```

```
      <part name="Header"  element="wsa:MessageID"/>
   </message>

   <!-- bodys  -->
   <message name="Info2__Message">
      <part name="Body" element="mob_mainbase_xml:Info2"/>
   </message>
   <message name="Discover__Message">
      <part name="Body" element="mob_mainbase_xml:Discover"/>
   </message>
   <message name="Label__Message">
      <part name="Body" element="mob_mainbase_xml:Label"/>
   </message>
   <message name="AlarmSeverity__Message">
      <part name="Body" element="mob_mainbase_xml:AlarmSeverity"/>
   </message>
```

Reference to data type (TYPES)

port type / interface
Describes the interfaces and the provided operations on this interface. For each operation the corresponding input and output messages are listed.

```
<!-- === PORT TYPES === -->

   <portType name="main_base__PortType">
      <operation name="GetLabel__Operation">
         <input message="mgws:EmptyMessage"/>
         <output message="mob_mainbase_xml:Label__Message"/>
      </operation>
      <operation name="SetLabel__Operation">
         <input message="mob_mainbase_xml:Label__Message"/>
         <output message="mob_mainbase_xml:Label__Message"/>
      </operation>
      <operation name="GetAlarmSeverity__Operation">
         <input message="mgws:EmptyMessage"/>
         <output message="mob_mainbase_xml:AlarmSeverity__Message"/>
      </operation>
      <operation name="GetInfo2__Operation">
         <input message="mgws:EmptyMessage"/>
         <output message="mob_mainbase_xml:Info2__Message"/>
      </operation>
      <operation name="GetDiscover__Operation">
         <input message="mgws:EmptyMessage"/>
         <output message="mob_mainbase_xml:Discover__Message"/>
      </operation>
   </portType>
```

The input message
(view of Service)
doesn't have any body!

binding
With the element binding we declare which transport protocol is used for which interface. For inputs or outputs of operations we need to assign the messages to the elements of the transport protocol (for the example SOAP, this will be SOAP:body or SOAP:header)

```
<!-- === BINDING === -->

   <binding name="main_base__Interface" type="mob_mainbase_xml:main_base__PortType">
      <soapbind:binding style="document" transport="http://schemas.xmlsoap.org/soap/
http"/>

      <operation name="GetLabel__Operation">
         <soapbind:operation soapAction="http://schemas.xmlsoap.org/ws/2004/09/transfer/
Get"/>
         <input>
            <soapbind:header message="tns:ResourceURIMessage" part="Header"
use="literal" />
            <soapbind:header message="tns:SelectorSetMessage" part="Header"
use="literal" />
            <soapbind:header message="tns:ToMessage" part="Header" use="literal" />
            <soapbind:header message="tns:ReplyToMessage" part="Header" use="literal" />
            <soapbind:header message="tns:ActionMessage" part="Header" use="literal" />
            <soapbind:header message="tns:MessageIDMessage" part="Header"
```

Definition of SOAP transport protocol and style

```
use="literal" />
            <soapbind:body use="literal"/>
         </input>
         <output>
            <soapbind:body use="literal"/>
         </output>
      </operation>

      <operation name="SetLabel__Operation">
         <soapbind:operation soapAction="http://schemas.xmlsoap.org/ws/2004/09/transfer/
Set"/>
         <input>
            <soapbind:header message="tns:ResourceURIMessage" part="Header"
use="literal" />
            <soapbind:header message="tns:SelectorSetMessage" part="Header"
use="literal" />
            <soapbind:header message="tns:ToMessage" part="Header" use="literal" />
            <soapbind:header message="tns:ReplyToMessage" part="Header" use="literal" />
            <soapbind:header message="tns:ActionMessage" part="Header" use="literal" />
            <soapbind:header message="tns:MessageIDMessage" part="Header"
use="literal" />
            <soapbind:body use="literal"/>
         </input>
         <output>
            <soapbind:body use="literal"/>
         </output>
      </operation>
      <operation name="GetDiscover__Operation">
         <soapbind:operation soapAction="http://schemas.xmlsoap.org/ws/2004/09/transfer/
Get"/>
         <input>
            <soapbind:header message="tns:ResourceURIMessage" part="Header"
use="literal" />
            <soapbind:header message="tns:SelectorSetMessage" part="Header"
use="literal" />
            <soapbind:header message="tns:ToMessage" part="Header" use="literal" />
            <soapbind:header message="tns:ReplyToMessage" part="Header" use="literal" />
            <soapbind:header message="tns:ActionMessage" part="Header" use="literal" />
            <soapbind:header message="tns:MessageIDMessage" part="Header"
use="literal" />
            <soapbind:body use="literal"/>
         </input>
         <output>
            <soapbind:body use="literal"/>
         </output>
      </operation>
   </binding>
```

Each operation has its transfer function (soapAction)

Each operation has its SOAP header and body

No need for interpretation. Passes to application as a full XML

service

Describes where the service is located. 'Services' can be subdivided into 'port/endpoint' with different addressing parameters

```
<!-- === SERVICE === -->

   <service name="MileGateService">

      <port name="LabelPort" binding="main_base__Interface">
         <soapbind:address location="http://localhost:9357/wsman" />
         <wsa:EndpointReference
         xmlns:wsa="http://www.w3.org/2005/08/addressing"
         xmlns:wsaw="http://www.w3.org/2006/02/addressing/wsdl"
         xmlns:wsdli="http://www.w3.org/2006/01/wsdl-instance">
            <wsa:Address>http://localhost:9357/wsman</wsa:Address>
            <wsa:Metadata>
               <wsaw:InterfaceName>main_base__PortType</wsaw:InterfaceName>
            </wsa:Metadata>
            <wsa:ReferenceParameters>

<wsman:SelectorSet>
                  <wsman:Selector Name="mf">main</
wsman:Selector>
                  <wsman:Selector Name="property">www.keymile.com/mg/2008/06/MoInfo/
Lable</wsman:Selector>
               </wsman:SelectorSet>
               <wsman:ResourceURI>/unit-11</wsman:ResourceURI>
            </wsa:ReferenceParameters>
         </wsa:EndpointReference>
      </port>
```

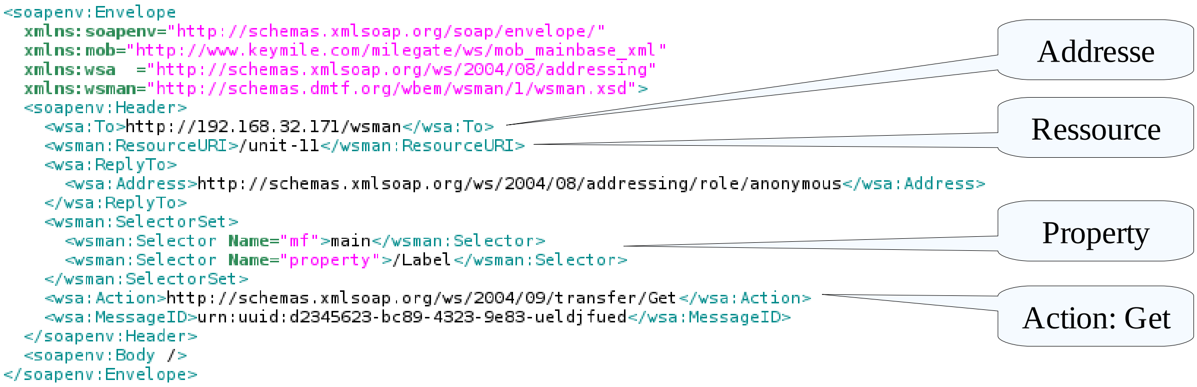
Definition of Service Endpoint with the addressing parameters

.

5.3 SOAP Message

The SOAP message defined in the Web Service Description File helps a lot to understand the descrtiption.
We are going to represent the SOAP messages for the Set- and GetLabel operation.

GetLabel SOAP message:

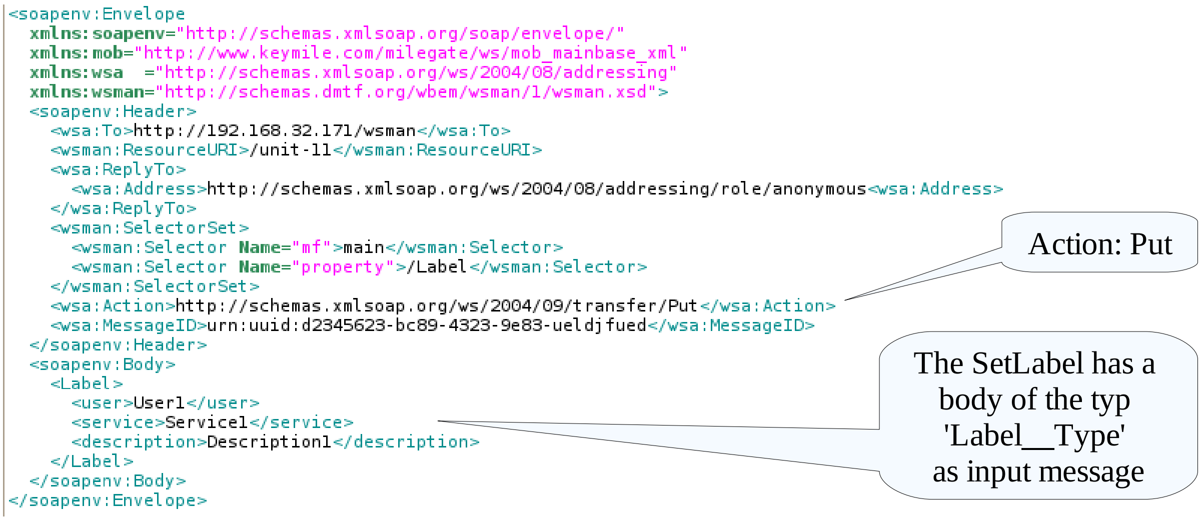


```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:mob="http://www.keymile.com/milegate/ws/mob_mainbase_xml"
  xmlns:wsa  ="http://schemas.xmlsoap.org/ws/2004/08/addressing"
  xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
  <soapenv:Header>
    <wsa:To>http://192.168.32.171/wsman</wsa:To>
    <wsman:ResourceURI>/unit-11</wsman:ResourceURI>
    <wsa:ReplyTo>
      <wsa:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous</wsa:Address>
    </wsa:ReplyTo>
    <wsman:SelectorSet>
      <wsman:Selector Name="mf">main</wsman:Selector>
      <wsman:Selector Name="property">/Label</wsman:Selector>
    </wsman:SelectorSet>
    <wsa:Action>http://schemas.xmlsoap.org/ws/2004/09/transfer/Get</wsa:Action>
    <wsa:MessageID>urn:uuid:d2345623-bc89-4323-9e83-ueldjfued</wsa:MessageID>
  </soapenv:Header>
  <soapenv:Body />
</soapenv:Envelope>
```

The input type (view of service) of the GetLabel message (defined in PortTypes):

```
<input message="mgws:EmptyMessage"/>        <!-- NO BODY -->
```

SetLabel SOAP message:



```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:mob="http://www.keymile.com/milegate/ws/mob_mainbase_xml"
  xmlns:wsa  ="http://schemas.xmlsoap.org/ws/2004/08/addressing"
  xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
  <soapenv:Header>
    <wsa:To>http://192.168.32.171/wsman</wsa:To>
    <wsman:ResourceURI>/unit-11</wsman:ResourceURI>
    <wsa:ReplyTo>
      <wsa:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous<wsa:Address>
    </wsa:ReplyTo>
    <wsman:SelectorSet>
      <wsman:Selector Name="mf">main</wsman:Selector>
      <wsman:Selector Name="property">/Label</wsman:Selector>
    </wsman:SelectorSet>
    <wsa:Action>http://schemas.xmlsoap.org/ws/2004/09/transfer/Put</wsa:Action>
    <wsa:MessageID>urn:uuid:d2345623-bc89-4323-9e83-ueldjfued</wsa:MessageID>
  </soapenv:Header>
  <soapenv:Body>
    <Label>
      <user>User1</user>
      <service>Service1</service>
      <description>Description1</description>
    </Label>
  </soapenv:Body>
</soapenv:Envelope>
```

Action: Put
The SetLabel has a body of the typ 'Label__Type' as input message

The input type (view of service) of the SetLabel message (defined in PortTypes):

```
<input message="mob_mainbase_xml:Label__Message"/>
```

6 Web Service Concepts

There is a huge variety of concepts and standards for Web Services. Concepts are provided by the World Wide Web Consortium W3C[7], OASIS[8], Microsoft[9], IBM[10] and even more. Some of this concepts overlap.
This chapter discusses the used concepts for our Web Service and provides an selection of other concepts which has been defined during this project as interesting for the future development of the MileGate Web Service.
Most of these concepts had not been included in the actual Web Service Description (WSDL) for reasons of time constraints. For these concepts interesting points for KEYMILE are emphasized and commented.

6.1 Addressing

The W3C recommendation Web Service Addressing 1.0 – Core[11] of the 9 May 2006 defines the construct of the message addressing properties and the endpoint references.
Other recommendation describes the Web Service Addressing 1.0 – SOAP Binding[12] (9 May 2006), the Web Service Addressing 1.0 – Metadata[13] (4 September 2007) and the candidate recommendation Web Service Addressing 1.0 – WSDL Binding[14] (29 May 2006).

6.1.1 WS-Addressing

This recommendation provides a mechanisms for end-to-end addressing of messages independent of the transport protocol used.
Addressing properties are, with the use of SOAP, contained in the header block.

The use of WS-Addressing allows us to address the source and destination endpoint of the system and to provide a identity for the message. Additional we specifies an action URI which defines the expected semantics.

With the concept of SOAP binding we assign the exchange structure defined by SOAP and a set of predefined faults.

The WSDL Metadata and WSDL binding indicate if the service is using WS-Addressing and provides the possibility for different message exchange patterns

[7] http://www.w3.org/2002/ws/
[8] http://www.oasis-open.org/specs/
[9] http://msdn.microsoft.com/en-us/library/ms951274.aspx
[10] http://www.ibm.com/developerworks/webservices/standards/
[11] http://www.w3.org/TR/ws-addr-core/
[12] http://www.w3.org/TR/ws-addr-soap/
[13] http://www.w3.org/TR/ws-addr-metadata/
[14] http://www.w3.org/TR/ws-addr-wsdl/

such as one-way, request-response, notification and solicit-response for WSDL 1.1 and some more for WSDL 2.0.

6.1.1.1 Endpoint Reference EPR

Endpoint Reference is a concept introduced by WS-Addressing and is used for the dynamic generation and customization of service endpoints.

As we have in our system endpoints that can change with the modification of the configuration or with the insertion of new hardware, we need a mechanism to indicate the new endpoint.
Possibilities are an additional Web Service (Endpoint Manager) which provides information about the addressable endpoints. Such a Web Service is described on the apache website
(http://svn.apache.org/repos/asf/cxf/trunk/testutils/src/main/resources/wsdl/locator.wsdl).
Other approaches are described later in the chapter 'WS-Distributed Management' under 'Advertisement' and 'Discovery'.

6.1.2 WS-Management

The final specification WS-Management was published by the Distributed Management Task Force DMTF the 02 December 2008. It provides a common way for systems to access and exchange management information.

The default addressing model uses a representation of an EPR that is a tuple of the following SOAP headers:[15]

- ***wsa:To*** *(required): the transport address of the service*
- ***wsman:ResourceURI*** *(required if the default addressing model is used): the URI of the resource class representation or instance representation*
- ***wsman:SelectorSet*** *(optional): identifies or "selects" the resource instance to be accessed if more than one instance of a resource class exists*

The ResourceURI is in our case used to address the Managed Object (e.g. /unit-11) and the SelectorSet specifies the management function (mf, e.g. Main) and the property (e.g. Label).

6.1.3 WS-Transfer

WS-Management has the status of W3C Member Submission (27 September 2006). The latest working draft is dated the 25 June 2009.

[15] http://www.dmtf.org/standards/published_documents/DSP0226_1.0.0.pdf (5.1.2 Default Addressing Model)

We use just the defined resource operations such as get and put with the URI:

http://schemas.xmlsoap.org/ws/2004/09/transfer/Get
http://schemas.xmlsoap.org/ws/2004/09/transfer/Put

REMARK: In the latest working draft the URI changed to:

http://www.w3.org/2009/06/ws-tra/Get
http://www.w3.org/2009/06/ws-tra/Put

Additionally the resource operations delete and create are possible.

6.2 Resource

The concepts in this chapter describe the handling of resources with Web Services. Following specifications are published by OASIS, please pay attention on the status of the recommendation which is indicated at the beginning of each description.

6.2.1 WS-Discovery

WS-Discovery is not standardized yet and has the state of an OASIS Committee Specification 01 since 14 May 2009.[16]

It defines a discovery protocol to locate services. It is often used to discover structures like LDAP (Lightweight Directory Access Protocol) or similar directories.

As our system contains one single service per MileGate, we have no need of a discovery at this level. Discovery could be used to figure out the complete infrastructure (ensemble of MileGates). Actually, this function is not needed because the system architecture and its addressing is designed in advance and won't change over the time.

CAN'T BE USED TO DISCOVER THE MANAGED OBJECTS (RESOURCE) OF THE MILEGATE!

6.2.2 WS-Resource

WS-Resource became a OASIS Standard the 1 April 2006.

The goal of WS-Resource is to standardize the terminology and concepts needed to express the relationship between Web services and resources.[17]

[16] http://docs.oasis-open.org/ws-dd/discovery/1.1/wsdd-discovery-1.1-spec.pdf
[17] http://docs.oasis-open.org/wsrf/wsrf-ws_resource-1.2-spec-os.pdf (1.1 Goals and Requirements)

A resource is represented by an endpoint reference (EPR) and addressed using the WS-Addressing concept:

```
<wsa:EndpointReference>
  <wsa:Address>http://192.168.0.1?res=RessourceName</wsa:Address>
  …
</wsa:EndpointReference>
```

The SOAP binding would look as followed:

```
<wsa:To>http://192.168.0.1?res=RessourceName</wsa:To>
```

6.2.2.1 WS-Resource Properties[18]

WS-Resource Properties also became a OASIS Standard the 1 April 2006.

The goal of WS-ResourceProperties is to standardize the terminology, concepts, operations, WSDL and XML needed to express the resource properties projection, its association with the Web service interface, and the messages defining the query and update capability against the properties of a WS-Resource.

Resource Property:
A resource property is a piece of information defined as part of the state model of a WS-Resource.

Resource Properties Document:
The XML document representing a logical composition of resource property elements. The resource properties document defines a particular view or projection of the state data implemented by the WS-Resource.

6.2.2.2 Comment

This concepts offer another manner for addressing the MILEGATE property (e.g. Label) and its parameters (e.g. Label1).

- With GetMultipleResourceProperties we can get a selection of Resource Properties. This mechanism offers the possibility of a customized request according to the preferences of the client. The advantage is that we do not have to request multiple operations and filter the content afterwards.
- With QueryResourceProperties we are able to query a Resource Properties document of a WS-Resource using a query expression such as XPath.
- The manageability of the system could be improved due to the dynamic add/delete of Resource Properties into the Resource Property document. (InsertResourceProperties, UpdateResourceProperties, DeleteResourceProperties)

[18] http://docs.oasis-open.org/wsrf/wsrf-ws_resource_properties-1.2-spec-os.pdf (1.1 and 2)

DOES NOT HELP TO FIGURE OUT WHICH ENDPOINT IS SUPPORTED BY WHICH OPERATION!

6.2.3 WS-Notification

WS-Notification contains the standard WS-Base Notification, WS-Brokered Notification and WS-Topics.

6.2.3.1 WS-Base Notification

WS-Base Notification became a OASIS Standard the 1 October 2006.

The goal of WS-BaseNotification is to standardize the terminology, concepts, operations, WSDL and XML needed to express the basic roles involved in Web services publish and subscribe for notification message exchange.[19]

A notify message containing one or more notifications should look as followed:[20]

```
…
<wsnt:Notify>
  <wsnt:NotificationMessage>
    <wsnt:SubscriptionReference>
      wsa:EndpointReferenceType
    </wsnt:SubscriptionReference> ?
    <wsnt:Topic Dialect="xsd:anyURI">
      {any} ?
    </wsnt:Topic>?
    <wsnt:ProducerReference>
      wsa:EndpointReferenceType
    </wsnt:ProducerReference> ?
    <wsnt:Message>
      {any}
    </wsnt:Message>
  </wsnt:NotificationMessage> +
    {any} *
</wsnt:Notify>
…
```

The notify message just before is transported as content of the SOAP body. Addressing for the notification (in SOAP header) by definition is following WS-Addressing action.

[19] http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf (1.1 Goals and Requirements)

[20] http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf (3.2 Notify)

```
<wsa:Action>
  http://docs.oasis-open.org/wsn/bw-2/NotificationConsumer/Notify
</wsa:Action>
```

The concept for the management of the subscription is also defined in WS-Base Notification.

6.2.3.2 WS-Brokered Notification

WS-Topics became a OASIS Standard the 1 October 2006.

The goal of WS-BrokeredNotification is to standardize message exchanges involved in Web services publish and subscribe of a message broker.[21]

6.2.3.3 WS-Topics

WS-Topics became a OASIS Standard the 1 October 2006.

The goal of the WS-Topics specification is to define a mechanism to organize and categorize items of interest for subscription known as “topics”. It defines a set of topic expression dialects that can be used as subscription expressions in subscribe request messages and other parts of the WS-Notification system.[22]

Topic:
A Topic is the concept used to categorize Notifications and their related Notification schemas.

Topic Tree:
A hierarchical grouping of Topics.

6.2.3.4 Comment

The mechanism described in this standards is basically similar to the notification system used in the MileGate. The requirement for the notifications used for the logbook could be fulfilled with this technique without the need for a continuous polling. (Pull-style notifications also possible)
Information in the logbook subcategories alarm, configuration changes, session login, equipment changes and events can be made accessible in a more particular way for other purposes.
It is recommended to allow authorization policies for topics.

[21] http://docs.oasis-open.org/wsn/wsn-ws_brokered_notification-1.3-spec-os.pdf (1.1 Goals and Requirements)

[22] http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-os.pdf (1.1 Goals and Requirements)

- The hierarchical structure of the topics allows a very targeted subscription for notifications.
- Management of the topics stays handy, also for large topic sets.
- The client can regroup the readout of notification according to his belongings and anywhere in his system.

6.3 Management

We already saw the WS-Management specification in the chapter Addressing. The idea behind this separation is that we just used WS-Management for addressing purposes.
In this chapter we describe functionality that goes much further. A complex concept is represented which interconnects multiple standards described before.

6.3.1 WS-Distributed Management

The standard WS-Distributed Management contains two parties.
Management using Web Services (MUWS 1.0) became a OASIS Standard the 9 March 2005 and Management of Web Services (MOWS 1.0) on 1 August 2006.
We will discuss here just the first standard. The second standard (MOWS 1.0) will be more interesting for the implementation of the management interface and not for the definition of the interface.

6.3.1.1 Management Using Web Services

The following paragraph defines some necessary terminology defined in the MUWS specification.

Manageable resource:
A resource capable of supporting one or more standard manageability capabilities.

Capability:
A group of properties, operations, events and metadata, associated with identifiable semantics and information and exhibiting specific behaviors.

Manageability capability:
A capability associated with one or more management domains.

Manageability endpoint:
A Web service endpoint associated with and providing access to a manageable resource.

Management domain:
An area of knowledge relative to providing control over, and information about, the behavior, health, lifecycle, etc. of manageable resources.

Management Using Web Services (MUWS) enables management of distributed information technology (IT) resources using Web services. Many distributed IT resources use different management interfaces. By leveraging Web service technology, MUWS enables easier and more efficient management of IT resources.[23]

MUWS is based on number of other specifications such as WS-Addressing, Metadata, Endpoint Reference, WS-Notification, WS-Topics, WS-Discovery, WS-Resource Properties which have been introduced before.

6.3.1.1.1 Manageability capabilities

The following capabilities are summarized from the documents MUWS part 1[24] (Chapter 3) & 2[25] (Chapter 2 and 3) mentioned as reference. The capabilities describe how the service can be used.

Operations

The operations in the MUWS specification correspond to those used in WSDL (portType element containing operation element with a description and any relevant metadata).

Properties
The properties of a manageable resource use the mechanism defined in WS-Resource Properties and its resource properties document.

Events
Event types are defined by using 'topic' and 'message content' elements. The information in the second element is transmitted as a part of the notification message (defined by WS-Base Notification).
To support event classification, different SituationCategoryTypes (element) such as AvailabilitySituation, CapabilitySituation, ConfigurationSituation and so on were defined (full list on page 9 of MUWS part 2). The aim of this classification is that the event consumer can comprehend the situation according the ability of the event source.
For each capability, topics are defined to link the capability with the event.

Metadata
We can define metadata on properties and operations. The aim of this is to provide information available in WSDL and WS-Resource Properties to a tool or management application.

[23] http://docs.oasis-open.org/wsdm/wsdm-muws1-1.1-spec-os-01.pdf (1 Introduction)
[24] http://docs.oasis-open.org/wsdm/wsdm-muws1-1.1-spec-os-01.pdf
[25] http://docs.oasis-open.org/wsdm/wsdm-muws2-1.1-spec-os-01.pdf

With the metadata element 'ValidWhile', we are able to block the invocation of an operation if certain properties do not have certain values.

Operational Status
With the capability operational status we have can simply represent if a resource is 'Available', 'PartiallyAvailable', 'Unavailable' or 'Unknown'.
This function can be implemented using the notification on property value change provided by WS-Resource Properties.

6.3.1.1.2 Management-*related capabilities*

The function of a management-related capability is related to the management of a resource, but it is not necessarily offered directly by a manageability endpoint of a resource. For example, the capability to help a manageability consumer discover a new manageable resource can be provided by a registry instead of by a management representation of the resource. As another example, a manageable resource may provide information about relationships in which it participates.

The following capabilities are summarized from the documents MUWS 2 (Chapter 4 and 5) mentioned as reference.

Relationships
The relationship defines the association between resources and the role of each participant. Interesting point for the MileGate system is that we can define a common AccessEndpoint for the participants of a relationship. A relationship may have its own properties, operations, events, lifecycles or can provide information about the relationship.
Another good point is that with the definition of relationships we enable the discovery of Endpoint References for other resource that participates in the relationship.

Advertisement
This capability provides a mechanisms to notify the creation or destruction of manageable resources. The following four new event topics are defined by Advertisement:

- ManageabilityEndpointCreation
- ManageableResourceCreation
- ManageabilityEndpointDestruction
- ManageableResourceDestruction

On the creation of a new Endpoint, the most interesting case for the MileGate system, an associated 'CreationNotification' message (WS-Notification) delivers the new Endpoint Reference.

6.3.1.1.3 Discovery

The goal of discovery is to obtain the EPR of a manageability endpoint.[26]

The advertisement capability, just introduced before, provides one way to provide a discovery mechanisms via events.
Another possibility is the discovery mechanisms via relationships described in under 'Relationships'.

A last possibility, perhaps also interesting for the MileGate, is the discovery of manageable resource by invoking a query on a registry. It is recommended to use a registry of the type specified by the WS-Service Group[27] specification.
Therefore MileGate should provide such a registry.

6.3.1.2 Comment

This specification defines how the different concepts can be combined together and all the advantages from each of them can enhance the usability of the complete system. We have plenty of good mechanisms for the dissolving of the problems we get if we pass from a proprietary to a standardized solution using Web Services.
It follows a short recapitulation of the advantages.

Resource Properties

- customized requests
- query resource properties using XPath
- better manageability on changes of the resource properties

Notification/Topic

- similarity to actual notification system
- hierarchical structure of topics
- subscription

Metadata

- constraints for the invocation of operations
- machine readable

Operational status

- knowledge if resource is available

Relationships

- common AccessEndpoint in relationship
- discovery of Endpoint References in relationship

Advertisement

- discovery of Endpoint References with creation notifications

[26] http://docs.oasis-open.org/wsdm/wsdm-muws2-1.1-spec-os-01.pdf (5 Discovery)
[27] http://docs.oasis-open.org/wsrf/wsrf-ws_service_group-1.2-spec-os.pdf

7 Conclusion

The study of the SOA was very interesting and helped to understand the advancement of the Internet in direction of Web Services. Not all the ideas are implementable for the MileGate because we have existing constraints and what is even more important, an existing and functional system. As all the transformations towards an service oriented architecture, the process will be very time-consuming.

Web Services architectures provides some exceptional concepts which offers a mass of new possibilities. Here a careful study of the requirements and on the functionalities wanted to offer had to be performed.
Attention have to be paid on the level of complexity of the system to not set limits for the implementation on the client side but also for not defining it vague or ambiguous.

"Things should be made as simple as possible, but no simpler."
Quote Albert Einstein

The endpoints of the Web Service and its management pose some problems which could be solved with the different techniques described. The easiest way to manage them is to use the endpoint references EPR by programming on the client side. A adapted version of the 'discover' (MileGate operation) which furnishes just the required information for the Web Service would be more efficient and would additionally allow to hide the infrastructure from the client.

The actual MileGate notifications, which follows the same principles as WS-Notification, should be translated into Web Service notifications and described with meta description to make it machine-readable.

For purposes of flexibility, the direct access of the management functions (not over two parameters, e.g. main/label). It will be easier to define constraints for the invocation of operations which are related to the access address (EPR).
The further idea is that we need to ensure that just possible functions can be invoked. Possibilities therefore are the simple response with an error, the 'ValidWhile' provided by WS-A Metadata or the use of relation according to WS-Distributed Management.

8 Annexes

8.1 Revision history

Revision			Short description of the modification	Prepared by	Checked by	Approve d
Doc ID	Version	Date				
WS	1	24/06/09	First Version	DSCHN		
WS	1.1	26/06/09	- WebService Description & SOAP message - WS-Resource and WS-Notification concept - References	DSCHN		
WS	1.2	01/07/09	- WS-Distributed Management - W3C Concepts - Web/Service oriented architecture - Conclusion	DSCHN		

8.2 References

8.2.1 Service Oriented Architecture / Web Service Architecture

- W3C documents about Web Service Architecture
http://www.w3.org/2002/ws/arch/

- Reference Model for Service Oriented Architecture 1.0
http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf

- Article: What Is Service-Oriented Architecture on webservices.xml.com
http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html

- Book: Service-orientierte Architekturen mit Web Services, Konzepte – Standards – Praxis, Ingo Melzer et al., SPEKTRUM Akademischer Verlag

- Book: Web Services. Principles and Technology, Michael P. Papazoglou, PEARSON

8.2.2 Webservice description / concepts

- WS-Addressing, W3C Recommendation

http://www.w3.org/TR/ws-addr-core/

- WS-A: WSDL Binding, W3C Recommendation
  http://www.w3.org/TR/ws-addr-wsdl/

- WS-A: SOAP Binding, W3C Recommendation
  http://www.w3.org/TR/ws-addr-soap/

- WS-A: Metatdata, W3C Recommendation
  http://www.w3.org/TR/ws-addr-metadata/

- WS-Management, Distributed Management Task Force
  http://www.dmtf.org/standards/published_documents/DSP0226_1.0.0.pdf

- WS-Transfer, W3C Submission
  http://www.w3.org/Submission/WS-Transfer/

- WS-Discovery, OASIS Committee Specification
  http://docs.oasis-open.org/ws-dd/discovery/1.1/wsdd-discovery-1.1-spec.pdf

- WS-Base Notification, OASIS Standard
  http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf

- WS-Brokered Notification, OASIS Standard
  http://docs.oasis-open.org/wsn/wsn-ws_brokered_notification-1.3-spec-os.pdf

- WS-Topics, OASIS Standard
  http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-os.pdf

- WS-Resource, OASIS Standard
  http://docs.oasis-open.org/wsrf/wsrf-ws_resource-1.2-spec-os.pdf

- WS-Resource Properties, OASIS Standard
  http://docs.oasis-open.org/wsrf/wsrf-ws_resource_properties-1.2-spec-os.pdf

- WS-Distributed Management: Management Using Web Services MUWS Part 1, OASIS Standard
  http://docs.oasis-open.org/wsdm/wsdm-muws1-1.1-spec-os-01.pdf

- WS-Distributed Management: Management Using Web Services MUWS Part 2, OASIS Standard
  http://docs.oasis-open.org/wsdm/wsdm-muws2-1.1-spec-os-01.pdf

- UDDI (Universal Description, Discovery and Integration), OASIS Standard
  http://uddi.org/pubs/uddi-v3.0.2-20041019.pdf

8.2.3 Illustrations

Before & after SOA
Source: http://www.sun.com/products/soa/img/ig_soa_before_after.gif

Three roles in SOA
Source: http://www.w3.org/2003/Talks/0521-hh-wsa/soa.png