The Web Service Challenge 2008

Steffen Bleul, Thomas Weise
University of Kassel
Wilhelmshöher Allee 73
34121 Kassel, Germany
weise|bleul|geihs@vs.uni-kassel.de

November 22, 2007

Abstract

In this description paper we outline the rules for the 2008 Web Service Challenge (WSC).

1 The 2008 Rules

Since 2005 the annual Web Service Challenge[1] (WS-Challenge, WSC) provides a platform for researchers in the area of web service composition to compare their systems and exchange experiences [1, 2, 3, 4]. It is co-located with the IEEE Conference on Electronic Commerce (CEC) and the IEEE International Conference on e-Technology, e-Commerce, and e-Service (EEE).

During the last years, the web service challenge focused on optimizing the service discovery and composition process solely. Therefore, abstractions from real-world situations were used. The taxonomy of semantic concepts was purely artificial as well as the involved data formats. Starting with this year's competition, the challenge will approach more practical scenarios. Therefore, the 2008 data formats as well as contest data itself will be based on real world schemas for service compositions, services, and ontologies.

2 The OWL Schema

Ontologies are usually expressed with OWL [5], an XML format [6]. OWL is the proprietary standard for the Semantic Web.

We use the OWL format in the 2008 challenge, but like in the previous years, we limit semantic evaluation strictly to taxonomies consisting of sub and super class relationship between semantic concepts only.

OWL is quite powerful. In addition to semantic concepts (OWL-Classes), OWL allows to specify instances of classes called individuals. Furthermore,

[1]see http://www.ws-challenge.org/ [accessed 2007-09-02]

one can define equivalence relations between individuals and classes. While we also distinguish between individuals and classes in the competition, the possibility to express equivalence relations between concepts is not used.

In OWL the semantic is specified with statements consisting of subject, predicate and object, e.g. `ISBN-10 is_a ISBN` (`ISBN` subsumes `ISBN-10`). Such statements can be specified with simple triplets but also with XML-Hierarchies and XML-References. The implementation of an OWL-Parser is hence not trivial. In order to ease the development of the competition contributions, we will stick to a fixed but valid OWL-Schema.

```
 1  <?xml version="1.0"?>
 2  <rdf:RDF
 3      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 4      xmlns:owl="http://www.w3.org/2002/07/owl#"
 5      xmlns="http://www.owl-ontologies.com/Ontology.owl#"
 6      xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
 7      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 8    xml:base="http://www.owl-ontologies.com/Ontology.owl">
 9    <owl:Ontology rdf:about=""/>
10  </rdf:RDF>
```

Listing 1: An example of a basic OWL-Document

Listing 1 illustrates an example of a basic OWL document. The single lines have the following meaning:

Line 1:	The XML-Document declaration.
Line 2:	The OWL-Document root. OWL is the ontology schema for the semantic description language RDF [7]. So some language elements are reused by OWL.
Line 3:	The RDF elements namespace.
Line 4:	The OWL elements namespace.
Line 5:	The namespace of this ontology (Syntax: `ontologyname`+`#`).
Line 6:	The XSD [8] elements namespace.
Line 7:	The RDFS [9] elements namespace. RDF Schema (RDFS) is a language extension of RDF.
Line 8:	The base namespace is also the ontology name in OWL.
Line 9:	The ontology declaration element `owl:Ontology`. The ontology name can be found in the XML-base namespace `xml:base`.

In general, the syntax for ontology namespaces is a valid URI [10, 11] followed directly by `#`. An ontology name must be a valid URI (e.g. `http://www.owl-ontologies.com/Ontology.owl`). Its namespace is then automatically the URI+`#`. (e.g. `http://www.owl-ontologies.com/Ontology.owl#`). In the WS-Challenge we stick to a fixed schema. We define a fixed URI and namespace for the taxonomy.

- The WSC-Ontology has the name `http://www.ws-challenge.org/wsc08.owl`.
- The WSC-Ontology namespace therefore is `http://www.ws-challenge.org/wsc08.owl#`.

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://www.ws-challenge.org/wsc08.owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.ws-challenge.org/wsc08.owl">
  <owl:Ontology rdf:about=""/>
</rdf:RDF>
```

Listing 2: An example of a WSC-08 OWL-based taxonomy document.

```
1  <?xml version="1.0"?>
2  <rdf:RDF
3      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
4      ...>
5    <owl:Ontology rdf:about=""/>
6
7    <owl:Class rdf:ID="Class1"/>
8    <owl:Class rdf:ID="Class2"/>
9
10   <owl:Class rdf:ID="Class1.1">
11     <rdfs:subClassOf rdf:resource="#Class1"/>
12   </owl:Class>
13 </rdf:RDF>
```

Listing 3: An example for the specification of class and subclasses.

Listing 3 outlines how classes and subclasses can be specified like in OWL. The lines have the following meaning:

Line 7:	The declaration of the concept/class with the name `Class1`.
Line 8:	The declaration of the concept/class with the name `Class2`.
Line 10:	The declaration of the concept/class with the name `Class1.1`.
Line 11:	The definition that `Class1.1` is a subclass of `Class1`.

In OWL, classes are defined with the `owl:Class` tag. Subclasses can be specified with `rdfs:subClassOf` tags. The attribute `rdf:ID` declares a new class name, whereas `rdf:resource` is a reference to a class. In OWL, a reference has the syntax `classname`+`#`, as for instance used in line 11.

In the WSC-08 competition, we use semantic individuals to annotate input and output parameters of services. Individuals are instances of classes and can be defined like in the following example.

Line 13: Specification of an individual with the name `Individual1` which is an instance of class `Class1`.

Line 14: Specification of an individual with the name `Individual1.1` which is an instance of class `Class1.1`

```
 1  <?xml version="1.0"?>
 2  <rdf:RDF
 3      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 4      ...>
 5    <owl:Ontology rdf:about=""/>
 6
 7    <owl:Class rdf:ID="Class1"/>
 8
 9    <owl:Class rdf:ID="Class1.1">
10      <rdfs:subClassOf rdf:resource="#Class1"/>
11    </owl:Class>
12
13    <Class1 rdf:ID="Individual1"/>
14    <Class1.1 rdf:ID="Individual1.1"/>
15  </rdf:RDF>
```

Listing 4: An example for the specification of semantic individuals.

The tag name of an individual declaration is also the name of the class. Since `Class1.1` is a subclass of `Class1`, `Individual1.1` also is a specialization of `Individual1`.

In short, the 2008 WSC will use the following basic document structure for the definition of semantic concepts:[2]

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    ...>
  <owl:Ontology rdf:about=""/>
  ...
  ClassDeclarations
    SubClassRelationships
    ...
  ...
  IndividualDeclarations
  ...
</rdf:RDF>
```

Listing 5: The basic document structure.

3 Service Descriptions

As already mentioned, we will use semantics specified in OWL to annotate the service descriptions. Furthermore, this year's competition service descriptions offer:

[2]XSD-definition will follow

- The service descriptions will be provided in a single WSDL-Document.
- The annotation with semantic individuals will not only be used for message parts, but for whole message structures specified with XSD.
- This message structures can consist of simple elements, SOAP-Arrays [12], Lists, Structures, and Enumerations.
- The matching will be based on documents in an extended WSDL format for semantic annotation. In contrast to existing solutions like WSDL-S [13] and SAWSDL [14], we also take matching of parameter groupings into consideration and (omitable) optional parameters.
- Also here we provide a fixed schema for easier parsing of the final document and will provide a schema definition.

3.1 The Basic WSDL-Document

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
  xmlns:service="http://www.ws-challenge.org/WSC08Services/"
  targetNamespace="http://www.ws-challenge.org/WSC08Services/">

  <service name="BookShopA">
    <port binding="searchBookSOAP" name="searchBook">
      <soap:address
          location="http://www.unknownexamplehost.ukn/"/>
    </port>
  </service>
  <binding name="searchBookSOAP"
      type="service:searchBookPortType"/>
  <portType name="searchBookPortType"/>
  <message name="BookShopRequestMessage"/>
  <message name="BookShopResponseMessage"/>

  .....

  <service name="BookShopZ">
    <port binding="searchBookSOAPZ" name="searchBookZ">
      <soap:address
          location="http://www.unknownexamplehost.ukn"/>
    </port>
  </service>
  <binding name="searchBookSOAPZ"
      type="service:searchBookPortTypeZ"/>
  <portType name="searchBookPortTypeZ"/>
  <message name="BookShopZRequestMessage"/>
  <message name="BookShopZResponseMessage"/>
```

```
33
34    ......
35
36    <types>
37      <xs:schema/>
38    </types>
39  </definitions>
```

Listing 6: Example of the basic WSDL-Document.

Line 1:	XML-Document declaration.
Line 2-10:	Namespacedeclaration (fixed for any composition challenge)
Line 12-16:	Servicedeclaration for a service named ”BookShopA”
Line 17:	The binding for BookShopA.
Line 18:	The `portType` for binding of `BookShopA`.
Line 19:	The request message for the `BookShopA`.
Line 20:	The response message for the `BookShopA`.
Line 22:	Marks an arbitrary amount of services.
Line 24:	Starting the declaration of `BookshopZ`.
Line 34:	Marks an arbitrary amount of services.
Line 36-38:	Here we declare all message structures for all services.
Line 39:	The end of the WSDL document.

In the WSC scenarios, we use the simplification that each service just has one unique service binding, `portType`, request and response message. The declaration of the binding of the service, `portType`, request, and response message is a fixed sequence. The elements related to one service are followed one after the other. Thus, parsing gets a lot easier.

This sequence in the declaration of a service will be used for the whole set of services for the composition challenge. The schema definition of all messages comes at the end. The overall WSDL document structure is valid. Challengers may or may not adjust their parsers for better performance by making use of the restrictions defined above.

3.2 The Binding Section for the Services

```
<service name="BookShopA">
  <port binding="searchBookSOAP" name="searchBook">
    <soap:address location="http://www.unknownexamplehost.ukn/"/>
  </port>
</service>
<binding name="searchBookSOAPA"
    type="service:searchBookPortTypeA">
  <soap:binding style="rpc"
      transport="http://schemas.xmlsoap.org/soap/http"/>
```

```
8    <operation name="searchBookA">
9      <soap:operation
10       soapAction="http://www.ws-challenge.org/BookShopA/searchBook"
           />
11     <input>
12       <soap:body use="literal" />
13     </input>
14     <output>
15       <soap:body use="literal" />
16     </output>
17   </operation>
18 </binding>
19 <portType name="searchBookPortType"/>
20 <message name="BookShopARequestMessage"/>
21 <message name="BookShopAResponseMessage"/>
```

Listing 7: An example for the binding section for each service

Line 6:	Declaration of the binding of `BookShopA` (see reference in line 2) and reference to its `portType` "`searchBookPortType`".
Line 7-18:	Template generated WSDL-Binding with no specific further semantics.

3.3 The `portType` Section

```
1  <service name="BookShopA"/>
2  <binding name="searchBookSOAPA"
       type="service:searchBookPortTypeA"/>
3  <portType name="searchBookPortTypeA">
4    <operation name="searchBookAOperation">
5      <input message="service:BookShopARequestMessage"/>
6      <output message="service:BookShopAResponseMessage"/>
7    </operation>
8  </portType>
9  <message name="BookShopARequestMessage"/>
10 <message name="BookShopAResponseMessage"/>
```

Listing 8: An example for the `portType` section for each service

Line 3:	Declaration of the service's BookShopA portType (see reference in line 2).
Line 4-8:	Automatically generated references to the service's input message (see `ID` in line 9) and response message (see `ID` in line 10).

3.4 The `message` Section

```
1  <wsdl:message name="BookShopARequestMessage">
```

```
2     <wsdl:part element="service:Book"
          name="BookShopAParameters"/>
3  </wsdl:message>
4  <wsdl:message name="BookShopAResponseMessage">
5     <wsdl:part element="service:Books"
          name="BookShopAParameters"/>
6  </wsdl:message>
```

Listing 9: An example for the `message` section for each service

Line 2+5: : Reference to the respective message structures inside the `<types/>`-Section of this WSDL-Document. This reference has no further semantic meaning.

3.5 The Message Structures and Semantic Annotations

```
<types>
  <xsd:schema
      targetNamespace="http://www.ws-challenge.org/WSC08Services/"
    xmlns:wsdl-ext="http://www.vs.uni-kassel.de/wsdl_extensions">
  <!-- Simple message with one element:-->
  <xsd:element name="BasicBook" name="title" type="xsd:string"
      wsdl-ext:id="Semtitle"/>

  <!-- Message structure of a struct "book":-->
  <xsd:element name="Book">
    <xsd:complexType>
      <xsd:sequence>
            <xsd:element name="isbn"   type="xsd:string"
                wsdl-ext:id="buchISBN"/>
            <xsd:element name="title"  type="xsd:string"
                wsdl-ext:id="buchTitel"/>
            <xsd:element name="author" type="xsd:string"
                wsdl-ext:id="buchAutor"/>
            <xsd:element name="year"   type="xsd:string"
                wsdl-ext:id="buchVerlag"/>
            <xsd:element name="price"  type="xsd:float"
                wsdl-ext:id="buchPreis"/>
          </xsd:sequence>
        <xsd:complexType>
     </xsd:element>

     <!-- Message structure of an array of books:-->
     <xsd:element name="Books">
      <xsd:complexType>
        <xsd:sequence minOccurs="0" maxOccurs="unbounded">
        <xsd:sequence>
              <xsd:element name="isbn"   type="xsd:string"
                  wsdl-ext:id="buchISBN"/>
              <xsd:element name="title"  type="xsd:string"
                  wsdl-ext:id="buchTitel"/>
              <xsd:element name="author" type="xsd:string"
                  wsdl-ext:id="buchAutor"/>
```

```
28                <xsd:element name="year"   type="xsd:string"
                      wsdl-ext:id="buchVerlag"/>
29                <xsd:element name="price"  type="xsd:float"
                      wsdl-ext:id="buchPreis"/>
30              </xsd:sequence>
31              </xsd:sequence>
32          <xsd:complexType>
33       </xsd:element>
34
35       <!-- Message structure of an array of books with an
             enumeration on genre:-->
36       <xsd:element name="BooksGenre">
37        <xsd:complexType>
38          <xsd:sequence minOccurs="0" maxOccurs="unbounded">
39          <xsd:sequence>
40                <xsd:element name="isbn"   type="xsd:string"
                      wsdl-ext:id="buchISBN"/>
41                <xsd:element name="title"  type="xsd:string"
                      wsdl-ext:id="buchTitel"/>
42                <xsd:element name="author" type="xsd:string"
                      wsdl-ext:id="buchAutor"/>
43                <xsd:element name="year"   type="xsd:string"
                      wsdl-ext:id="buchVerlag"/>
44                <xsd:element name="price"  type="xsd:float"
                      wsdl-ext:id="buchPreis"/>
45                <xsd:element name="genre">
46              <xsd:simpleType>
47              <xsd:restriction base="xsd:string">
48              <xsd:enumeration value="romantic"
                    wsdl-ext:id="SemRomantic"/>
49              <xsd:enumeration value="adventure"
                    wsdl-ext:id="SemAdventure"/>
50              <xsd:enumeration value="science-fiction"
                    wsdl-ext:id="SemScienceFiction"/>
51              </xsd:restriction>
52            </xsd:simpleType>
53            </xsd:element>
54              </xsd:sequence>
55              </xsd:sequence>
56          <xsd:complexType>
57       </xsd:element>
58    </xsd:schema>
59  </types>
```

Listing 10: A simple message with one element

Line 1:	Start of the WSDL-Types section. This section specifies the structure of the SOAP messages [12] with XSD.
Line 20:	Start of the XSD section.
Line 4:	Definition of a single message element `BasicBook` consisting of the title of the book as a string.
Line 7-17:	Definition of a book message structure `Book` consisting of single structure elements `isbn`, `title`, `author`, `year` and `price`.

Line 9-15: Structure is a sequence of elements.

Line 21-33: A definition like the ones above, but this one is an array of books. (see `xsd-attributes minOccurs=""maxOccurs=""`).

Line 36-58: Like the examples before, this one defines an array of books. Additionally, there is an enumeration representing the book genre (see `xsd:enumeration`).

Line 45-53: Enumeration is a `simpleType` restricted on the data type `string` and consists of an arbitrary numbers of enumeration elements.

Note that:

- There are no references on XSD elements. The message definition is strictly hierarchical. We introduce this simplification to ease parsing.
- We have added the `wsdl-ext` attribute that references the semantic extension in our approach. It represents the annotation of message elements.
- Only simple elements like `xsd:element` with types, e.g. `string` are annotated. Complex elements defined by a `xsd:complexType` must be reasoned by its sub-elements. This is the semantic challenge.

4 WSDL Semantic Extension

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  ...
  xmlns:wsdl-ext="http://www.vs.uni-kassel.de/wsdl_extensions">

  <service .../>
  <binding .../>
  <portType .../>
  <message .../>
  ...
  <message .../>
  <service/>
  ...

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

  <!-- WSC-08 Semantic Annotation Section -->
  <wsdl-ext:semExtension>

    <!-- Semantic Annotation Element-->
    <wsdl-ext:semExtension/>
    ...
    <wsdl-ext:semExtension/>

    <!-- Semantic Parameter Grouping -->
    <wsdl-ext:semMessageExt/>
    ...
    <wsdl-ext:semMessageExt/>
  </wsdl-ext:semExtension>
</definitions>
```

Listing 11: A WSDL Document for the WSC-08

```
<wsdl-ext:semExtension>

  <!-- Semantic Extension for the message with ID
      "getPriceRequest" -->
  <wsdl-ext:semMessageExt idref="getPriceRequest">

    <!-- Semantic Annotation for the xsd:element with ID "price"
        -->
    <wsdl-ext:semExt id="price">

      <!-- Ontology reference for the semantic individual for
          this element -->
      <wsdl-ext:ontologyRef>
         http://www.owl-ontologies.com/Ontology.owl#Bookprice
      </wsdl-ext:ontologyRef>

      <!-- Ontology reference for the semantic individual for
          the unit of thos element -->
      <wsdl-ext:unit>
        http://www.owl-ontologies.com/Ontology.owl#Dollar
      </wsdl-ext:unit>

      <!-- This is a flag indicating if this element is required
          in this message -->
      <wsdl-ext:optional>true</wsdl-ext:optional>
    </wsdl-ext:semExt>

  </wsdl-ext:semMessageExt>

  <!-- Arbitrary amount of message annotations -->
  <wsdl-ext:semMessageExt idref="getPriceResponse"/>
  ...
  <wsdl-ext:semMessageExt .../>
  ...
</wsdl-ext:semExtension>
```

Listing 12: The Semantic Extension

```
<!-- Semantic Extension for the message with ID
    "getPriceRequest" -->
<wsdl-ext:semMessageExt idref="getPriceRequest">

  <!-- Semantic Annotation for the xsd:element with ID "author"
      and "price" -->
  <wsdl-ext:semExt id="author">
    <wsdl-ext:ontologyRef>
       http://www.owl-ontologies.com/Ontology.owl#Bookauthor
    </wsdl-ext:ontologyRef>
    <wsdl-ext:optional>false</wsdl-ext:optional>
  </wsdl-ext:semExt>
  <wsdl-ext:semExt id="price">
    <wsdl-ext:ontologyRef>
       http://www.owl-ontologies.com/Ontology.owl#Bookprice
    </wsdl-ext:ontologyRef>
    <wsdl-ext:unit>
      http://www.owl-ontologies.com/Ontology.owl#Dollar
    </wsdl-ext:unit>
    <wsdl-ext:optional>true</wsdl-ext:optional>
  </wsdl-ext:semExt>

  <!-- Parameter Group Definition: Operation may be invoked with the following message elements -->
  <wsdl-ext:group id="group1">
    <!-- Message MUST contain a title, author and publisher to be able to invoke the operation -->
    <wsdl-ext:group-element id="title"/>
    <wsdl-ext:group-element id="author"/>
    <wsdl-ext:group-element id="publisher"/>
  </wsdl-ext:group>

  <!-- Optional the operation may be invoked with the following message elements -->
  <wsdl-ext:group id="group2">
    <!-- Message MUST contain at least an isbn number
    <wsdl-ext:group-element id="isbn"/>
  </wsdl-ext:group>

  <!-- Priority definitions for the parameter groups defined before -->
  <wsdl-ext:group-choice id="groupChoice1">

    <!-- Group with parameters title, author and publisher has low priority "1" -->
    <wsdl-ext:groupChoiceMember ref="group1" priority="1"/>

    <!-- Group with parameters isbn is preferred with higher priority "2" -->
    <wsdl-ext:groupChoiceMember ref="group2" priority="2"/>
  </wsdl-ext:group-choice>
<wsdl-ext:semMessageExt/>
```

Listing 13: Parameter Groupings

5 The Solution Format

```
<!-- Document root -->
<WSChallenge type="solutions">

  <!-- First solution -->
  <case name="SolutionA">
    <service name="servicep36a4117307"/>
    <serviceSpec name="2.1">
      <service name="servicep71a5006237"/>
        <serviceSpec name="2.2">
        <service name="servicep08a5642735"/>
          <service name="servicep81a0118335"/>
        </serviceSpec>
        <service name="servicep63a9740674"/>
      </serviceSpec>
  </case>

  <!-- Second alternative solution -->
  <case name="SolutionB">
    ...
  </case>
</WSChallenge>
```

Listing 14: Example Solution

```
<!-- Document Root -->
<process name="MoreCreditsBP"
         targetNamespace="http://xmlns.oracle.com/MoreCreditsBP">

  <!-- Main sequence -->
  <sequence name="main">

    <!-- Starting BPEL invocation (Input: Challenge Query) -->
    <receive name="receiveQuery"
                 portType="SolutionProcess" variable="query"/>

        <!-- Switch/Case operator for alternative solutions -->
        <switch name="SolutionAlternatives-SolutionA-SolutionB">

          <!-- First solution -->
          <case name="Alternative-SolutionA">

            <!-- Sequence for serviceSpec 2.0 -->
            <sequence>
              <!-- Firstly invoking Service: servicep36a4117307
                  -->
              <invoke name="servicep36a4117307"
                            portType="seeWSDLFile"
                            operation="seeWSDLFile"/>

                    <!-- Parallel execution of serviceSpec 2.1
                        and 2.2 -->
                    <flow>
                      <!-- serviceSpec 2.1 -->
                      <sequence>
                        <invoke name="servicep71a5006237"
                              portType="seeWSDLFile"
                              operation="seeWSDLFile"/>
                      </sequence>

                      <!-- serviceSpec 2.2 -->
                      <sequence>
                        <!-- Two alternatives either
                            servicep08a5642735
                          or servicep81a0118335 -->

                        <switch name="Alternative-Services">
                          <case
                              name="Execute-servicep08a5642735">
                            <sequence>
```

```

                              <!-- Trigger Service
                                  servicep08a5642735 -->
                              <invoke name="servicep08a5642735"
                                      portType="seeWSDLFile"
                                      operation="seeWSDLFile"/>
                            </sequence>
                          </case>
                          <case
                              name="Execute-servicep81a0118335">
                            <sequence>
                              <!-- OR Trigger Service
                                  servicep81a0118335 -->
                              <invoke name="servicep81a0118335"
                                      portType="seeWSDLFile"
                                      operation="seeWSDLFile"/>
                            </sequence>
                          </case>
                        </switch>
                      </sequence>
                    </flow>
            </sequence>
          </case>

          <!-- Second solution -->
          <case name="Alternative-SolutionB">
            ...
          </case>
        </switch>
    </sequence>
</process>
```

Listing 15: The WSBPEL document

References

[1] M. Brian Blake, Kwok Ching Tsui, and Andreas Wombacher. The eee-05 challenge: a new web service discovery and composition competition. In *Proceedings of the 2005 IEEE International Conference on e-Technology, e-Commerce, and e-Service, EEE'05*, pages 780–783, March 29-April 1, 2005. Online available at `http://ws-challenge.georgetown.edu/ws-challenge/The%20EEE.htm` [accessed 2007-09-02].

[2] Thomas Weise, Steffen Bleul, and Kurt Geihs. Web service composition systems for the web service challenge - a detailed review. Kasseler Informatikschriften (KIS) 2007, 7, University of Kassel, FB16, Distributed Systems Group, Wilhelmshöher Allee 73, 34121 Kassel, Germany, November 19, 2007. Persistent Identifier: urn:nbn:de:hebis:34-2007111919638. Online available at `http://kobra.bibliothek.uni-kassel.de/handle/urn:nbn:de:hebis:34-2007111919638` and `http://www.it-weise.de/documents/files/WBG2007WSCb.pdf` [accessed 2007-11-20]. See also [15] and [16].

[3] M. Brian Blake, William K.W. Cheung, Michael C. Jaeger, and Andreas Wombacher. Wsc-06: The web service challenge. In *Proceedings of 2006 IEEE Joint Conference on E-Commerce Technology (CEC'06) and Enterprise Computing, E-Commerce and E-Services (EEE'06)*, pages 505–508, 2006. See proceedings [17].

[4] M. Brian Blake, William K.W. Cheung, Michael C. Jaeger, and Andreas Wombacher. Wsc-07: Evolving the web service challenge. In *Proceedings of IEEE Joint Conference (CEC/EEE 2007) on E-Commerce Technology (9th CEC'07) and Enterprise Computing, E-Commerce and E-Services (4th EEE'07)*, pages 422–423, 2006. See proceedings [18].

[5] Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. *OWL Web Ontology Language Reference*. World Wide Web Consortium (W3C), February 10, 2004. W3C Recommendation. Online available at `http://www.w3.org/TR/2004/REC-owl-ref-20040210/` [accessed 2007-11-22].

[6] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, and François Yergeau. *Extensible Markup Language (XML) 1.0 (Fourth Edition)*. World Wide Web Consortium (W3C), September 29, 2007. W3C Recommendation. Online available at `http://www.w3.org/TR/2006/REC-xml-20060816` [accessed 2007-09-02].

[7] World Wide Web Consortium (W3C). *RDF Primer*, February 10, 2004. W3C Recommendation. Online available at `http://www.w3.org/TR/2004/REC-rdf-primer-20040210/` [accessed 2007-11-22].

[8] David C. Fallside and Priscilla Walmsley. *XML Schema Part 0: Primer Second Edition*. World Wide Web Consortium (W3C), second edition, October 28, 2004. W3C Recommendation. Online available at `http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/` [accessed 2007-09-02].

[9] World Wide Web Consortium (W3C). *RDF Vocabulary Description Language 1.0: RDF Schema*, February 10, 2004. W3C Recommendation. Online available at `http://www.w3.org/TR/2004/REC-rdf-schema-20040210/` [accessed 2007-11-22].

[10] T. Berners-Lee. *Universal Resource Identifiers in WWW – A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web*. Network Working Group, June 1994. Request for Comments (RFC) 1630, Category: Informational. Online available at `http://tools.ietf.org/html/rfc1630` [accessed 2007-11-22]. See also [11].

[11] T. Berners-Lee, R. Fielding, and L. Masinter. *Uniform Resource Identifier (URI): Generic Syntax*. Network Working Group, January 2005. Request for Comments (RFC) 3986, Category: Standards Track, STD:

66, Updates: RFC 1738, Obsoletes: RFC 2732, RFC 2396, RFC 1808. Online available at http://tools.ietf.org/html/rfc3986 [accessed 2007-11-22]. See also [10].

[12] World Wide Web Consortium (W3C). *SOAP Version 1.2 Part 0: Primer (Second Edition)*, April 27, 2007. W3C Recommendation. Online available at http://www.w3.org/TR/2007/REC-soap12-part0-20070427/ [accessed 2007-09-02].

[13] Rama Akkiraju, Joel Farrell, John Miller, Meenakshi Nagarajan, Marc-Thomas Schmidt, Amit Sheth, and Kunal Verma. *Web Service Semantics - WSDL-S*. World Wide Web Consortium (W3C), November 7, 2005. W3C Member Submission, Version 1.0. Online available at http://www.w3.org/Submission/2005/SUBM-WSDL-S-20051107/ [accessed 2007-09-02].

[14] World Wide Web Consortium (W3C). *Semantic Annotations for WSDL and XML Schema*, August 28, 2007. (SAWSDL) W3C Recommendation. Online available at http://www.w3.org/TR/2007/REC-sawsdl-20070828/ [accessed 2007-09-02].

[15] Steffen Bleul, Thomas Weise, and Kurt Geihs. Large-scale service composition in semantic service discovery. In Ws-Challenge Part: M. Brian Blake, Andreas Wombacher, Michel C. Jaeger, and William K. Cheung, editors, *Proceedings of 2006 IEEE Joint Conference on E-Commerce Technology (CEC'06) and Enterprise Computing, E-Commerce and E-Services (EEE'06)*, pages 427–429, 2006. See proceedings [17]. 1st place in 2006 WSC. Online available at http://www.it-weise.de/documents/files/BWG2006WSC.pdf [accessed 2007-11-22]. See 2007 WSC [16] and [2].

[16] Steffen Bleul, Thomas Weise, and Kurt Geihs. Making a fast semantic service composition system faster. In *Proceedings of IEEE Joint Conference (CEC/EEE 2007) on E-Commerce Technology (9th CEC'07) and Enterprise Computing, E-Commerce and E-Services (4th EEE'07)*, pages 517–520, 2007. See proceedings [18]. 2nd place in 2007 WSC. Online available at http://www.it-weise.de/documents/files/BWG2007WSC.pdf [accessed 2007-11-22]. See 2006 WSC [15] and [2].

[17] *Proceedings of 2006 IEEE Joint Conference on E-Commerce Technology (CEC'06) and Enterprise Computing, E-Commerce and E-Services (EEE'06)*, ISBN: 978-0-7695-2511-2. IEEE Computer Society, Los Alamitos, California, Washington, Tokyo, The Westin San Francisco Airport, 1 Old Bayshore Highway, Millbrae, United States, June 26-29, 2006.

[18] IEEE Computer Society. *Proceedings of IEEE Joint Conference (CEC/EEE 2007) on E-Commerce Technology (9th CEC'07) and Enterprise Computing, E-Commerce and E-Services (4th EEE'07)*, ISBN: 978-0-7695-2913-4. IEEE Computer Society, National Center of Sciences, Tokyo, Japan, July 23-26, 2007.

Index