R085: Describing Messages That Refer to Other Web Services

W3C WSD WG F2F

Rennes, 2003-05-13

Arthur Ryman

Outline

- Introduction
- Example
- Details

R085

“The description language SHOULD allow describing Messages that include references (URIs) to strongly-typed referents, both values and Services.”

Assumptions

- Strongly-typed references
  - WSDL must specify binding, and hence interface, of references contained in messages
- Reference formats
  - Required simple URI, i.e. xsd:anyURI
  - Optional other complex formats, e.g. WS-Addressings

Applications

- REST
  - shared information hyperspace
  - XML analog of HTML hrefs
- Composition by aggregation
  - Reference component Web services
- Factory services
  - Dynamically create new service instances, c.f. Grid
- Callbacks
  - e.g. simple event notification

REST Example

- Taken from “Building Web Services the REST Way” by Roger Costello
- Request part list
- Request part detail
- Place part order, etc. – not covered

GET http://www.parts-depot.com/parts XML Response using XLink

```
<?xml version="1.0"?>
<p:Parts xmlns:p="http://www.parts-depot.com"
        xmlns:xlink="http://www.w3.org/1999/xlink">
    <Part id="00345"
      xlink:href="http://www.parts-depot.com/parts/00345"/>
    <Part id="00346"
      xlink:href="http://www.parts-depot.com/parts/00346"/>
    <Part id="00347"
      xlink:href="http://www.parts-depot.com/parts/00347"/>
    <Part id="00348"
      xlink:href="http://www.parts-depot.com/parts/00348"/>
</p:Parts>
```

GET http://www.parts-depot.com/parts/00345 XML Response

```
<?xml version="1.0"?>
<p:Part xmlns:p="http://www.parts-depot.com"
        xmlns:xlink="http://www.w3.org/1999/xlink">
    <Part-ID>00345</Part-ID>
    <Name>Widget-A</Name>
    <Description>This part is used within the frap assembly</Description>
    <Specification
        xlink:href="http://www.parts-depot.com/parts/00345/specification"/>
    <UnitCost currency="USD">0.10</UnitCost>
    <Quantity>10</Quantity>
</p:Part>
```

Part Interface WSDL
No change

```
<wsdl:message name="getPartInput"/>
<wsdl:message name="getPartOutput">
   <wsdl:part name="return" element="p:Part“/>
</wsdl:message>

<wsdl:interface name="partInterface">
   <wsdl:operation name="GET">
      <wsdl:input message="tns:getPartInput"/>
      <wsdl:output message="tns:getPartOutput"/>
   </wsdl:operation>
</wsdl:interface>
```

Part Binding WSDL
No change

```
<wsdl:binding name="partHTTPBinding" type="tns:partInterface">
    <http:binding verb="GET"/>
        <wsdl:operation name="GET">
            <http:operation location=""/>
            <wsdl:input>
                <http:urlEncoded/>
            </wsdl:input>
            <wsdl:output>
                <mime:mimeXml part="return"/>
            </wsdl:output>
        </wsdl:operation>
</wsdl:binding>
```

Part List Interface WSDL
Describe endpoint reference interface

```
<wsdl:message name="getPartListInput"/>
<wsdl:message name="getPartListOutput">
   <wsdl:part name="return" element="p:Parts"/>
</wsdl:message>

<wsdl:interface name="partListInterface">
   <wsdl:operation name="GET">
      <wsdl:input message="tns:getPartListInput">
      <wsdl:output message="tns:getPartListOutput">
[1]       <wsdl:endpoint name="partURI" part="return"
            xpath="/p:Parts/Part/@xlink:href"
            interface="tns:partInterface"/>
      </wsdl:output>
   </wsdl:operation>
</wsdl:interface>
```

Part List Binding WSDL
Describe endpoint reference binding

```
<wsdl:binding name="PartListHTTPBinding" type="tns:partListInterface">
   <http:binding verb="GET"/>
   <wsdl:operation name="GET">
      <http:operation location=""/>
      <wsdl:input>
         <http:urlEncoded/>
      </wsdl:input>
      <wsdl:output>
[2]      <wsdl:endpoint name="partURI" binding="tns:partHTTPBinding"/>
         <mime:mimeXml part="return"/>
      </wsdl:output>
   </wsdl:operation>
</wsdl:binding>
```

Part List Service WSDL
No change

```
<wsdl:service name="PartListService">
    <wsdl:port name="PartListHTTPPort" binding="tns:PartListHTTPBinding">
        <http:address location="http://www.parts-depot.com/parts"/>
    </wsdl:port>
</wsdl:service>
```

GET http://www.parts-depot.com/parts XML Reponse with WS-Addressing

```
<?xml version="1.0"?>
<p:Parts xmlns:p="http://www.parts-depot.com"
        xmlns:wsa="http://schemas.xmlsoap.org/ws/2003/03/addressing">
    <Part id="00345">
      <wsa:EndpointReference>
        <wsa:Address>http://www.parts-depot.com/parts/00345</wsa:Address>
      </wsa:EndpointReference>
    </Part>
    …
    <Part id="00348">
      <wsa:EndpointReference>
        <wsa:Address>http://www.parts-depot.com/parts/00348</wsa:Address>
      </wsa:EndpointReference>
    </Part>
</p:Parts>
```

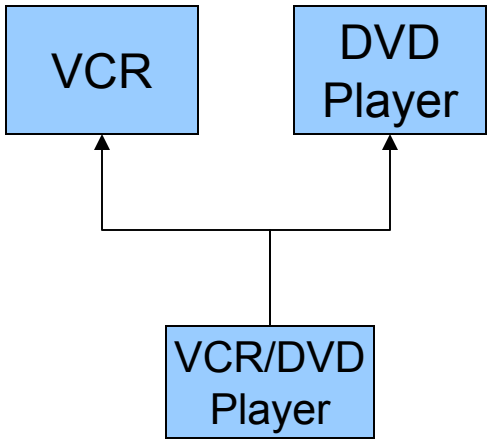
Part List Interface WSDL With WS-Addressing

```
<wsdl:interface name="partListInterface">
   <wsdl:operation name="GET">
      <wsdl:input message="tns:getPartListInput">
      <wsdl:output message="tns:getPartListOutput">
[1]       <wsdl:endpoint name="partURI" part="return"
              xpath="/p:Parts/Part/wsa:EndpointReference"
              interface="tns:partInterface"/>
      </wsdl:output>
   </wsdl:operation>
</wsdl:interface>
```

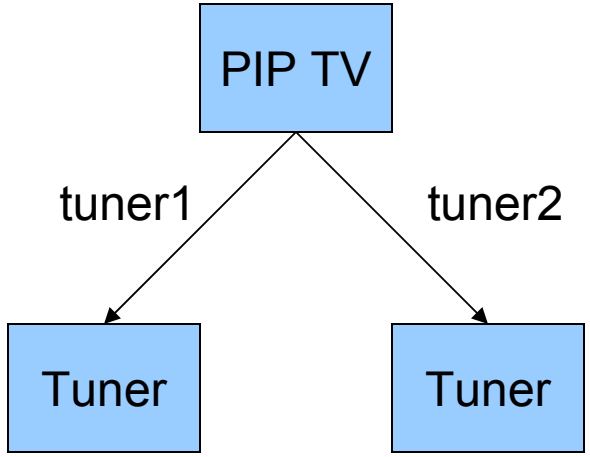
Aggregation Example

- Inheritance enables some types of composition, e.g. a combination VCR/DVD player can inherit from VCR and DVD player
- Aggregation is required for others, e.g. a Picture In Picture (PIP) TV has two tuners

Inheritance vs Aggregation



Inheritance



Aggregation

PIPTVInterface
GET XML Response

```
<?xml version="1.0"?>
<TV xmlns="http://xml.sony.com/tv>
  <Manufacturer>SONY</Manufacturer>
  <Model>29" PIP STEREO WEGA MULTISYSTEM TV</Model>
  <SN>746-ABG-554-XVC</SN>
  <Tuner1>http://192.168.0.208/tuner1</Tuner1>
  <Tuner2>http://192.168.0.208/tuner2</Tuner2>
</TV>
```

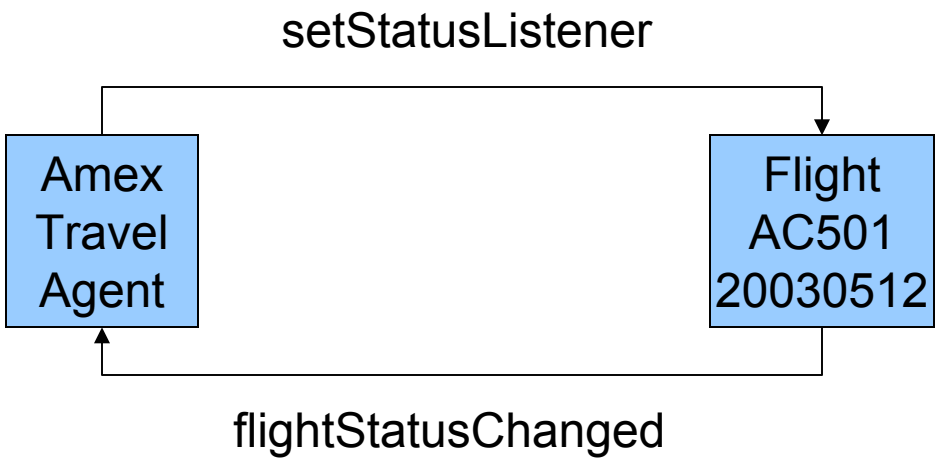
PIPTVInterface WSDL

```
<wsdl:interface name="PIPTVInterface">
  <wsdl:operation name="GET">
    <wsdl:input message="tns:getInput"/>
    <wsdl:output message="tns:getOutput"
      xmlns:tv="http://xml.sony.com/tv>
      <wsdl:endpoint name="tuner1Uri" part="return"
        xpath="/tv:TV/tv:Tuner1" interface="tns:TunerInterface"/>
      <wsdl:endpoint name="tuner2Uri" part="return"
        xpath="/tv:TV/tv:Tuner2" interface="tns:TunerInterface"/>
    </wsdl:output>
  </wsdl:operation>
</wsdl:interface>
```

Or, we could have simply getTuner1 and getTuner2 operations …

Callback Example

http://www.amex.com/travelagent



http://www.aircanada.com/ac501/20030512

POST http://www.aircanada.com/ac501/20030512

The travel agent sends the following message to register interest in a flight:

```
<?xml version="1.0"?>
<setStatusListener>
  <callback>http://www.amex.com/travelagent</callback>
</setStatusListener>
```

POST http://www.amex.com/travelagent

The Air Canada flight Web service sends the following when the status of flight AC501 on 2003-05-12 changes:

```
<?xml version="1.0"?>
<flightStatusChanged>
  <flight>http://www.aircanada.com/ac501/20030512</flight>
  <status>delayed</status>
</flightStatusChanged>
```

FlightInterface WSDL

```
<wsdl:operation name="setStatusListener">
  <wsdl:input message="tns:setStatusListenerInput>
    <wsdl:endpoint name="agentUri" part="callback"
      interface="tns:AgentInterface"/>
  </wsdl:input>
  <wsdl:output message="tns:setStatusListenerOutput"/>
</wsdl:operation>
```

AgentInterface WSDL

```
<wsdl:operation name="flightStatusChanged">
  <wsdl:input message="tns:flightStatusChangedInput">
     <wsdl:endpoint name="flightUri" part="flight"
        interface="tns:FlightInterface"/>
 </wsdl:input>
  <wsdl:output message="tns:flightStatusChangedOutput"/>
</wsdl:operation>
```

Endpoint Reference Interface Description

- An endpoint reference interface description may appear as an immediate child of <wsdl:input>, <wsdl:output>, or <wsdl:fault> within a <wsdl:interface> element. The parent of the <wsdl:endpoint> element associates a <wsdl:message> element with it. The <wsdl:enpoint> element has the following attributes:
- @name
  - a required NCName which must be unique among all endpoint reference interface descriptions within the scope of the enclosing <wsdl:input>, <wsdl:output>, or <wsdl:fault>. The name attribute identifies the endpoint reference interface description.
- @part
  - a required NCName which must match a <wsdl:part> element in the associated <wsdl:message>.

Endpoint Reference Interface Description cont'd

- @xpath
  - an optional XPath expression that is evaluated on the content of the specified <wsdl:part>. The XPath expression should evaluate to one or more nodes of type xsd:anyURI or some other type such as xsa:EndpointReferenceType. The default value of this attribute is ".". A conforming WSDL processor MUST understand xsd:anyURI and MAY understand other types such as wsa:EndpointReferenceType.
- @interface
  - a required QName which must match some <wsdl:interface> element either in the current WSDL document or in an imported or included WSDL document.

Endpoint Reference Binding Description

- An endpoint reference binding description may appear as an immediate child of <wsdl:input>, <wsdl:output>, or <wsdl:fault> within a <wsdl:binding> element. The <wsdl:endpoint> element has the following attributes:
- @name
  - a required NCName which must match a corresponding <wsdl:endpoint> element in the <wsdl:interface> element of the interface that is bound by the enclosing <wsdl:binding> element.

Endpoint Reference Binding Description cont'd

- @binding
  - a required QName which must match some <wsdl:binding> element either in the current WSDL document or in an imported or included WSDL document, and that binds the <wsdl:interface> identified by the endpoint reference interface description.

Design Rationale

- Why not annotate XSD with interface and binding?
  - Makes message formats non-reusable, e.g. xsd:anyURI must be extended for every binding
  - Couples interface with binding, e.g. suppose protocol used by component services is the same as the protocol of the aggregate service
  - Couples XSD with WSDL, e.g. messages may be defined before services
  - Inconsistent with current WSDL message/interface/binding layering

Design Rationale cont'd

- Why not use XML Schema Component Designators (SCD) instead of XPath?
  - XPath allows used of instance data, e.g.
    - <endpoint … xpath="Person[@jobcode='01']" interface="EmployeeInterface"/>
    - <endpoint … xpath="Person[@jobcode='02']", interface="ManagerInterface"/>
  - XPath is well-understood, mature, and supported by processors and tools
  - SCD is under current development
  - SCD may expose incidental structure of schema, e.g. use of <group>

Open Questions

- Should we restrict @xpath to a subset of XPath? (no)
- Do we need to add anything to WSDL to describe the use of XML Base with XLink? (no)
- Should we allow binding to a derived interface? (yes)

Conclusion

- Endpoint references have many valid uses in Web services
- Proposal satisfies R085
- Proposal is consistent with WSDL layering of interface and binding
- Proposal adds a small number of new concepts to WSDL:
  - endpoint reference interface description
  - endpoint reference binding description