WS-I Basic Profile

Sang Shin
Michèle Garoche
www.javapassion.com
"Learning is fun!"

Agenda

- Scope of Basic Profile 1.0a
- Messaging
  - XML Representations of SOAP Message
  - Use of SOAP in HTTP
- Service Description
  - Document Structure
  - Types
  - Messages
  - PortTypes
  - Bindings
  - SOAP Binding
  - Use of XML Schema
- Service Publication and Discovery
- Security

Scope of Basic Profile 1.0a

Scope of Basic Profile 1.0a (page1)

- Messaging (SOAP)
  - XML representation of SOAP messages
  - SOAP processing model
  - Use of SOAP in HTTP
- Service description (WSDL)
  - Document structure
  - Types
  - Messages
  - Port types
  - Bindings and SOAP bindings

Scope of Basic Profile 1.0a (page2)

- Service publication and discovery (UDDI)
  - bindingTemplate
  - tModel
- Security
  - Use of HTTPS

Messaging

XML Representation of SOAP Message

Messaging

Areas Clarified

- Fault messages
- SOAP encodingStyle attribute
- DTDs and PIs
- SOAP trailers
- SOAPAction field

SOAP encodingStyle Attribute

- Background: The soap:encodingStyle attribute is used to indicate the use of a particular scheme in the encoding of data into XML.
- However, this introduces complexity, as this function can also be served by the use of XML Namespaces.
- As a result, the Profile prefers the use of literal, non-encoded XML.

SOAP encodingStyle Attribute

- R1005: A MESSAGE MUST NOT contain soap:encodingStyle attributes on any of the elements whose namespace name is "http://schemas.xmlsoap.org/soap/envelope/"
- R1006: A MESSAGE MUST NOT contain soap:encodingStyle attributes on any element that is a child of soap:Body
- R1007: A MESSAGE described in an rpc-literal binding MUST NOT contain soap:encodingStyle attribute on any elements are grandchildren of soap:Body

SOAP's use of XML: DTDs and PIs

- Background: XML DTDs and PIs may introduce security vulnerabilities, processing overhead and ambiguity in message semantics when used in SOAP messages. As a result, these XML constructs are disallowed by section 3 of SOAP 1.1.
- R1008: A MESSAGE MUST NOT contain a Document Type Declaration (DTD)
- R1009: A MESSAGE MUST NOT contain Processing Instructions

SOAP Trailers

- Background: The interpretation of sibling elements following the soap:Body element is unclear. Therefore, such elements are disallowed.
- R1011: A MESSAGE MUST NOT have any element children of soap:Envelope following the soap:Body element.

SOAP Trailers: Incorrect Usage

```
<soap:Envelope xmlns:soap='http://schemas.xmlsoap.org/soap/envelope/' >
  <soap:Body>
    <p:Process xmlns:p='http://example.org/Operations' />
  </soap:Body>
  <m:Data xmlns:m='http://example.org/information' >
  Here is some data with the message
  </m:Data>
</soap:Envelope>
```

SOAP Trailers: Correct Usage

```
<soap:Envelope xmlns:soap='http://schemas.xmlsoap.org/soap/envelope/' >
  <soap:Body>
     <p:Process xmlns:p='http://example.org/Operations' >
       <m:Data xmlns:m='http://example.org/information' >
         Here is some data with the message
       </m:Data>
     </p:Process>
  </soap:Body>
</soap:Envelope>
```

SOAPAction

- Background: Interoperability testing has demonstrated that requiring the SOAPAction HTTP header field-value to be quoted increases interoperability of implementations
- Background: Even though HTTP allows for header field-values to be unquoted, some implementations require that the value be quoted.
- Background: The SOAPAction header is purely a hint to processors
  - All vital information regarding the intent of a message is carried in the Envelope

SOAPAction

- R2744: A HTTP request MESSAGE MUST contain a SOAPAction HTTP header field with a quoted value equal to the value of the soapAction attribute of soapbind:operation, if present in the corresponding WSDL description.
- R2745: A HTTP request MESSAGE MUST contain a SOAPAction HTTP header field with a quoted empty string value, if in the corresponding WSDL description, the soapAction attribute of soapbind:operation is either not present, or present with an empty string as its value.

SOAPAction Correct Usage

- Correct usage example 1
  - WSDL: <soapbind:operation soapAction="foo" />
  - HTTP header field: SOAPAction: "foo"
- Correct usage example 2
  - WSDL: <soapbind:operation /> or <soapbind:operation soapAction="" />
  - HTTP header field: SOAPAction: ""

Use of SOAP in HTTP

Messaging

Areas Clarified

- Identifying SOAP faults
- HTTP methods and extensions
- HTTP and TCP ports
- HTTP success status codes
- HTTP redirect status codes
- HTTP server error status codes

Identifying SOAP Faults

- Background: Some consumer implementations use only the HTTP status code to determine the presence of a SOAP Fault. Because there are situations where the Web infrastructure changes the HTTP status code, and for general reliability, the Profile requires that they examine the envelope.
- R1107: A RECEIVER MUST interpret SOAP messages containing only a soap:Fault element as a Fault.

HTTP Methods and Extensions

- Background: The SOAP1.1 specification defined its HTTP binding such that two possible methods could be used, the HTTP POST method and the HTTP Extension Framework's M-POST method. The Profile requires that only the HTTP POST method be used and precludes use of the HTTP Extension Framework.
- R1132 A HTTP request MESSAGE MUST use the HTTP POST method.
- R1108 A MESSAGE MUST NOT use the HTTP Extension Framework.

HTTP and TCP ports

- Background: SOAP is designed to take advantage of the HTTP infrastructure. However, there are some situations (e.g., involving proxies, firewalls and other intermediaries) where there may be harmful side effects. As a result, instances may find it advisable to use ports other than the default for HTTP (port 80).
- R1110 An INSTANCE MAY accept connections on TCP port 80 (HTTP).

HTTP Success Status Codes (1)

- Background: HTTP uses the 2xx series of status codes to communicate success. In particular, 200 is the default for successful messages, but 202 can be used to indicate that a message has been submitted for processing. Additionally, other 2xx status codes may be appropriate, depending on the nature of the HTTP interaction.

HTTP Success Status Codes (2)

- R1124 An INSTANCE MUST use a 2xx HTTP status code for responses that indicate a successful outcome of a request.
- R1111 An INSTANCE SHOULD use a "200 OK" HTTP status code for responses that contain a SOAP message that is not a SOAP fault.
- R1112 An INSTANCE SHOULD use either a "200 OK" or "202 Accepted" HTTP status code for a response that does not contain a SOAP message but indicates successful HTTP outcome of a request.

HTTP Redirect Status Codes (1)

- Background: There are interoperability problems with using many of the HTTP redirect status codes, generally surrounding whether to use the original method, or GET
- Background: The Profile mandates "307 Temporary Redirect", which has the semantic of redirection with the same HTTP method, as the correct status code for redirection

HTTP Redirect Status Codes (2)

- R1130 An INSTANCE MUST use HTTP status code "307 Temporary Redirect" when redirecting a request to a different endpoint.
- R1131 A CONSUMER MAY automatically redirect a request when it encounters a "307 Temporary Redirect" HTTP status code in a response.

HTTP Server Error Status Codes

- Background: HTTP uses the 5xx series of status codes to indicate failure due to a server error.
- R1126: An INSTANCE MUST use a "500 Internal Server Error" HTTP status code if the response message is a SOAP Fault.

Service Description

Specifications Referenced

- WSDL 1.1
- XML Schema Part 1: Structures
- XML Schema Part 2: Datatypes

Document Structure

Service Description

WSDL Schema Definitions (1)

- Background: The normative schemas for WSDL appearing in Appendix 4 of the WSDL 1.1 specification have inconsistencies with the normative text of the specification. The Profile references new schema documents that have incorporated fixes for known errors.
- Background: Although the Profile requires WSDL descriptions to be Schema valid, it does not require consumers to validate WSDL documents. It is the responsibility of a WSDL document's author to assure that it is Schema valid.

WSDL Schema Definitions (2)

- R2028 A DESCRIPTION using the WSDL namespace (prefixed "wsdl" in this Profile) MUST be valid according to the XML Schema found at "http://schemas.xmlsoap.org/wsdl/2003-02-11.xsd".
- R2029 A DESCRIPTION using the WSDL SOAP binding namespace (prefixed "soapbind" in this Profile) MUST be valid according to the XML Schema found at "http://schemas.xmlsoap.org/wsdl/soap/2003-02-11.xsd".

Placement of WSDL Import Element

- R2022 When they appear in a DESCRIPTION, wsdl:import elements MUST precede all other elements from the WSDL namespace except wsdl:documentation.
- R2023 When they appear in a DESCRIPTION, wsdl:types elements MUST precede all other elements from the WSDL namespace except wsdl:documentation and wsdl:import.

Correct Usage Example 1

```
<definitions name="StockQuote"
targetNamespace="http://example.com/stockquote/definitions">
    <import namespace="http://example.com/stockquote/base"
location="http://example.com/stockquote/stockquote.wsdl"/>
    <message name="GetLastTradePriceInput">
      <part name="body" element="..."/>
    </message>
          ...
</definitions>
```

Correct Usage Example 2

```
<definitions name="StockQuote"
                ...
  xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <schema targetNamespace="http://example.com/stockquote/schemas"
        xmlns="http://www.w3.org/2001/XMLSchema">
          .......
    </schema>
  </types>
  <message name="GetLastTradePriceInput">
      <part name="body" element="tns:TradePriceRequest"/>
  </message>              ...
  <service name="StockQuoteService">
    <port name="StockQuotePort" binding="tns:StockQuoteSoap">
          ....
    </port>
  </service>
</definitions>
```

WSDL Extensions

- Background: Requiring support for WSDL extensions that are not explicitly specified by this or another WS-I Profile can lead to interoperability problems with development tools that have not been instrumented to understand those extensions

Types

Service Description

soapenc:Array (1)

- Background: The recommendations in WSDL 1.1 Section 2.2 for declaration of array types have been interpreted in various ways, leading to interoperability problems. Further, there are other clearer ways to declare arrays.

soapenc:Array (2)

- R2110 In a DESCRIPTION, array declarations MUST NOT extend or restrict the soapenc:Array type.
- R2111 In a DESCRIPTION, array declarations MUST NOT use wsdl:arrayType attribute in the type declaration.
- R2112 In a DESCRIPTION, array declaration wrapper elements SHOULD NOT be named using the convention ArrayOfXXX.
- R2113 A MESSAGE containing serialized arrays MUST NOT include the soapenc:arrayType attribute.

soapenc:Array (3)

- Given the WSDL Description:

```
<xsd:element name="MyArray1" type="tns:MyArray1Type"/>
<xsd:complexType name="MyArray1Type">
 <xsd:sequence>
  <xsd:element name="x" type="xsd:string"
   minOccurs="0" maxOccurs="unbounded"/>
 </xsd:sequence>
</xsd:complexType>
```

- The SOAP message would serialize as (omitting namespace declarations for clarity):

```
<MyArray1>
 <x>abcd</x>
 <x>efgh</x>
</MyArray1>
```

Messages

Service Description

Binding and Parts (1)

- Background: There are various interpretations about how many wsdl:part elements are permitted or required for document-literal and rpc-literal bindings and how they must be defined.
- Background: Use of wsdl:message elements with zero parts is permitted in Document styles to permit operations that can send or receive messages with empty soap:Body. Use of wsdl:message elements with zero parts is permitted in RPC styles to permit operations that have no (zero) parameters and/or a return value.

Binding and Parts (2)

- For document-literal bindings, the Profile requires that at most one part, abstractly defined with the element attribute, be serialized into the soap:Body element.
- When a wsdl:part element is defined using the type attribute, the wire representation of that part is equivalent to an implicit (XML Schema) qualification of a minOccurs attribute with the value "1", a maxOccurs attribute with the value "1" and a nillable attribute with the value "false".

Declaration of part elements

- Background: Examples 4 and 5 in WSDL 1.1 Section 3.1 incorrectly show the use of XML Schema types (e.g. "xsd:string") as a valid value for the element attribute of a wsdl:part element.
- R2206 A wsdl:message in a DESCRIPTION containing a wsdl:part that uses the element attribute MUST refer, in that attribute, to a global element declaration.

Incorrect Usage Examples

- INCORRECT:

```
<message name="GetTradePriceInput">
   <part name="tickerSymbol" element="xsd:string"/>
   <part name="time" element="xsd:timeInstant"/>
</message>
```

- INCORRECT:

```
<message name="GetTradePriceInput">
   <part name="tickerSymbol" element="xsd:string"/>
</message>
```

Correct Usage Examples

- CORRECT:

```
<message name="GetTradePriceInput">
   <part name="body" element="tns:SubscribeToQuotes"/>
</message>
```

PortTypes

Service Description

Order of part elements

- Background: Permitting the use of parameterOrder helps code generators in mapping between method signatures and messages on the wire.
- R2301 The order of the elements in the soap:body of a MESSAGE MUST be the same as that of the wsdl:parts in the wsdl:message that describes it.
- R2302 A DESCRIPTION MAY use the parameterOrder attribute of an wsdl:operation element to indicate the return value and method signatures as a hint to code generators.

Bindings

Service Description

Use of SOAP Binding

- Background: The Profile limits the choice of bindings to the well defined and most commonly used SOAP binding. MIME and HTTP GET/POST bindings are not permitted by the Profile.
- R2401 A wsdl:binding element in a DESCRIPTION MUST use WSDL SOAP Binding as defined in WSDL 1.1 Section 3.

SOAP Binding

Service Description

HTTP Transport

- Background: The profile limits the underlying transport protocol to HTTP.
- R2702 A wsdl:binding element in a DESCRIPTION MUST specify the HTTP transport protocol with SOAP binding. Specifically, the transport attribute of its soapbind:binding child MUST have the value "http://schemas.xmlsoap.org/soap/http".

Consistency of style Attribute

- Background: The style, "document" or "rpc", of an interaction is specified at the wsdl:operation level, permitting wsdl:bindings whose wsdl:operations have different styles. This has led to interoperability problems.
- R2705 A wsdl:binding in a DESCRIPTION MUST use either be a rpc-literal binding or a document-literal binding.

Encodings and the use Attribute

- Background: The Profile prohibits the use of encodings, including the SOAP encoding.
- R2706 A wsdl:binding in a DESCRIPTION MUST use the value of "literal" for the use attribute in all soapbind:body, soapbind:fault, soapbind:header and soapbind:headerfault elements.

Default for use Attribute

- Background: There is an inconsistency between the WSDL 1.1 specification and the WSDL 1.1 schema regarding whether the use attribute is optional on soapbind:body, soapbind:header, and soapbind:headerfault, and if so, what omitting the attribute means.
- R2707 A wsdl:binding in a DESCRIPTION that contains one or more soapbind:body, soapbind:fault, soapbind:header or soapbind:headerfault elements that do not specify the use attribute MUST be interpreted as though the value "literal" had been specified in each case.

Child Element for Document-Literal Bindings

- Background: WSDL 1.1 is not completely clear what, in document-literal style bindings, the child element of soap:Body is.
- R2712 A document-literal binding MUST be represented on the wire as a MESSAGE with a soap:Body whose child element is an instance of the global element declaration referenced by the corresponding wsdl:message part.

One-way Operations (1)

- Background: There are differing interpretations of how HTTP is to be used when performing one-way operations.
- Background: One-way operations do not produce SOAP responses. Therefore, the Profile prohibits sending a SOAP envelope in response to a one-way operation. This means that transmission of one-way operations can not result in processing level responses or errors. For example, a "500 Internal Server Error" HTTP response that includes a SOAP message containing a SOAP Fault element can not be returned.

One-way Operations (2)

- Background: The HTTP response to a one-way operation indicates the success or failure of the transmission of the message. Based on the semantics of the different response status codes supported by the HTTP protocol, the Profile specifies that "200" and "202" are the preferred status codes that the sender should expect, signifying that the one-way message was received. A successful transmission does not indicate that the SOAP processing layer and the application logic has had a chance to validate the message or have committed to processing it.

One-way Operations (3)

- Background: Despite the fact that the HTTP 1.1 assigns different meanings to response status codes "200" and "202", in the context of the Profile they should be considered equivalent by the initiator of the request. The Profile accepts both status codes because some SOAP implementations have little control over the HTTP protocol implementation and cannot control which of these response status codes is sent.

One-way Operations (4)

- R2714 For one-way operations, an INSTANCE MUST NOT return a HTTP response that contains a SOAP envelope. Specifically, the HTTP response entity-body must be empty.
- R2750 A CONSUMER MUST ignore a SOAP response carried in a response from a one-way operation.
- R2727 For one-way operations, a CONSUMER MUST NOT interpret a successful HTTP response status code (i.e., 2xx) to mean the message is valid or that the receiver would process it.

Use of XML Schema

Service Description

Use of XML Schema

- Background: WSDL 1.1 uses XML Schema as one of its type systems. The Profile mandates the use of XML Schema as the type system for WSDL descriptions of Web Services.
- R2800 A DESCRIPTION MAY use any construct from XML Schema 1.0.
- R2801 A DESCRIPTION MUST use XML Schema 1.0 Recommendation as the basis of user defined datatypes and structures.

Technical Highlights – WSDL1.1

- Limited to rpc- and document-literal styles
- SOAP/HTTP binding required
  - Other bindings out of scope, but *may* be used
- WSDL Schema errata fixed
- Exclude wsdl:import of XSD files
- Exclude xs:import of schema in WSDL files
- Numerous spec clarifications

Service Publication & Discovery

Technical Highlights – UDDI2.0

- Require WSDL1.1 as description language
- Established category to identify WS-I conformant entities

Security

Technical Highlights – Security

- May use SSLv3 (HTTP/S)
- HTTP1.1 Basic Auth
- Identify risks and threats and countermeasures within Profile

Thank you!

Sang Shin
Michèle Garoche
www.javapassion.com
"Learning is fun!"