Web Service Standards

Peter Tröger
Operating Systems and Middleware Group
Hasso-Plattner-Institute
University of Potsdam

Agenda

- **Introduction**
- Base specifications
- Metadata specifications
- Notification specifications
- Other standards
- Conclusion

WS-* and Standardization

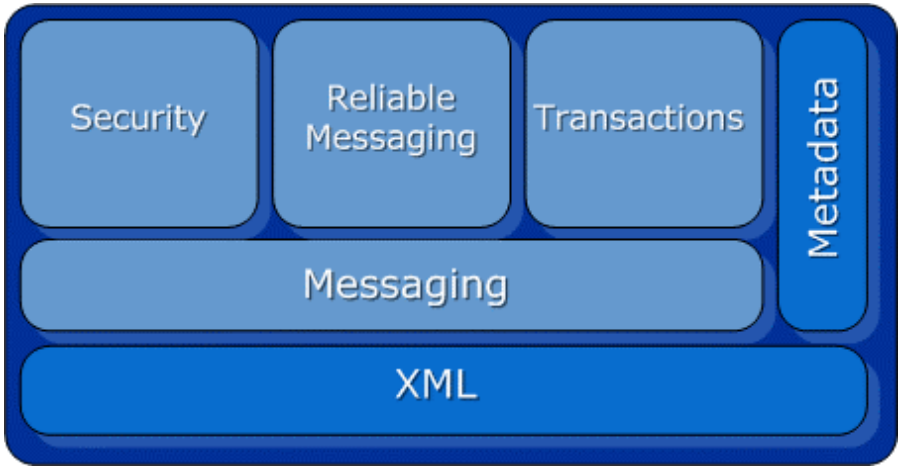
- Huge number of specifications
- Most documents are maintained by industry, without any standardization body
- Some documents are handled as W3C / OASIS member submission
- Small number as result of a W3C working group
- In some areas concurrent developments
  (WS-Eventing vs. WS-Notification)
- Often joint work, especially for the fundamental specs (IBM, Microsoft, BEA; also Sun, SAP)

Web Service Toolkits

- Apache Axis / WSFX / Muse
- IBM Emerging Technologies Toolkit (ETTK)
- Microsoft Web Services Enhancements toolkit (WSE)
- Sun Java Java Web Services Developer Pack (WSDP)
- Verisign Trust Service Integration Kit (TSIK)

Microsoft & Web Services

- Security:
  WS-Security, WS-Trust, WS-Federation
- Reliable Messaging:
  WS-ReliableMessaging
- Transaction:
  WS-Coordination,
  WS-AtomicTransaction
- Messaging:
  SOAP, WS-Addressing,
  MTOM, WS-Eventing,
  WS-Transfer
- Metadata:
  WSDL, UDDI, WS-Policy



Don Box and WS-Why (XML DevCon 2004)

- **WS-DesertIsland** – “a must have” for core XML Web Services (XML, SOAP, WS-Addressing, WS-MetadataExchange & XSD+WSDL)
- **WS-IslandResort** - the next layer of important specs (WS-Security, WS-Trust, WS-ReliableMessaging & WS-Policy)
- **WS-NewZealand** - specs you'd probably need once in a lifetime (WS-Eventing, WS-Enumeration & WS-AtomicTransaction)

Don Box (contd.)

- **WS-IslandOfDoctorMoreau** - the ugly step children of the WS-* spec family (UDDI, WS-Transfer, WS-BusinessActivity, MTOM and BPEL4WS)
- **WS-FantasyIsland** - specs Don would love to see (XPath/SQL-like query for web services, SOAP over TCP, binary XML & WSDL based on RELAX NG)

Agenda

- Introduction
- **Base specifications**
- Metadata specifications
- Notification specifications
- Other standards
- Conclusion

WS-Addressing

- W3C Member Submission (Microsoft, IBM, Sun, Bea, SAP), August 2004
- First public Working Draft in December 04
  - Core
  - SOAP binding
  - WSDL binding
- Multiple implementations (Apache Axis, IBM ETTK, Microsoft WSE, Systinet Server)
- Based on SOAP 1.2 and WSDL 2.0
- Successor to WS-Routing

WS-Addressing Goals

- Communication of information that is typically provided by transport protocols
  - → Source and destination endpoint
  - → Reply endpoint
  - → Policy
- Transport-neutral message transmission through processing nodes (Firewall, Gateway)
- Identification of Web Service endpoints
- Enabling secure end-to-end identification
- Support for asynchronous and event-based interaction patterns

WS-Addressing Definitions

- Web Service endpoint
  - Referencable target for Web Service messages
- **Endpoint references**
  - convey the information for endpoint identification and referencing
  - Allow endpoint access
  - Support addresses for individual messages
  - makes it possible to pass additional endpoint metadata (contract details and policies)
- **Message information header**
  - End-to-end message characteristics
  - Includes source and destination endpoint and message identifiers

Endpoint References (ER)

- Identification & reference for a Web Service endpoint
- Extension of the WSDL 1.1 description model
- Complement WSDL elements:
  - When statefull service instances need to be identified – no more HTTP session keys !
  - For communication of dynamic endpoint configuration changes
- Usage scenarios
  - Dynamic generation and customization of service endpoint descriptions
  - Identification and description of specific service instances
  - Dynamic exchange of endpoint information

Endpoint Reference Properties

- Address (mandatory)
  - URI; identifies the endpoint; may reference network or logical address
- Selected port type (optional)
  - QName of the primary WSDL portType
- Service name (optional)
  - QName identifying the WSDL service element that contains the WSDL definition of the endpoint
  - optional non-qualified name for specific port that corresponds to the endpoint

```
<wsa:EndpointReference>
 <wsa:Address>xs:anyURI</wsa:Address>
 <wsa:PortType>xs:QName</wsa:PortType>
 <wsa:ServiceName PortName="xs:ns"?>xs:QName</wsa:ServiceName>
</wsa:EndpointReference>
```

Endpoint Reference Properties

- Reference properties / parameters (optional)
  - Required to identify the entity or resource being conveyed
  - Provided by the issuer of the reference, opaque to the consuming application
  - Endpoints with different *reference properties* may accept different sets of messages or may follow different policies
  - Differing *reference parameters* do not imply differing XML Schema, WSDL or policies for the endpoints

```
<wsa:EndpointReference>
 <wsa:Address>xs:anyURI</wsa:Address>
 <wsa:ReferenceProperties>... </wsa:ReferenceProperties>
 <wsa:ReferenceParameters>... </wsa:ReferenceParameters>
</wsa:EndpointReference>
```

Endpoint Reference Example

```
<wsa:EndpointReference
  xmlns:c="http://example.org/claims"
  xmlns:p="http://schemas.xmlsoap.org/ws/2002/12/policy" > <wsa:Address
     http://claimserver/ins/p.asmx
  </wsa:Address>
  <wsa:ReferenceProperties>
     <c:PatientProfile>123456</c:PatientProfile>
     <c:CarrierID>987654</c:CarrierID> </wsa:ReferenceProperties>
  <wsa:PortType>c:ClaimsPortType</wsa:PortType> <wsa:ServiceName
  PortName="c:ClaimsSoapPort">
     c:ClaimsService
  </wsa:ServiceName>
  <p:Policy> ... <!-- policy statement -->
  </p:Policy>
</wsa:EndpointReference>
```

Message Information Headers

- Augmentation of messages with abstract properties
- Independence from transport protocol
- Enable the identification and location of endpoints in an interaction
  - One Way
  - Request-Reply
- Allows multiple response messages to different endpoints (e.g. WS-ReliableMessaging: reply, fault, acknowledgement)

Message Information Properties

- Destination (mandatory)
  - Address of the intended receiver, independent from transport destination information (HTTP request URL)
- Source endpoint (optional)
  - Endpoint where the message was originated
- Message identifier (optional)
  - Identifies the message uniquely in time and space
  - Retransmissions should occur with the same message ID
  - Must be set when a reply is expected

```
<wsa:To>xs:anyURI</wsa:To>
<wsa:From>endpoint-reference</wsa:From>
<wsa:MessageID>xs:anyURI</wsa:MessageID>
```

Message Information Properties

- Reply endpoint (optional)
  - Intended receiver for replies to this message
  - Source endpoint MAY be used when absent
  - Client must consider it, requires also message ID
- Fault endpoint (optional)
  - Intended receiver for faults related to this message
  - Reply or source endpoint MAY be used when absent
  - Client must consider it, requires also message ID

```
<wsa:To>xs:anyURI</wsa:To>
<wsa:ReplyTo>endpoint-reference</wsa:ReplyTo>
<wsa:FaultTo>xs:anyURI</wsa:FaultTo>
```

Message Information Properties

- Action (mandatory)
  - In-envelope version of the HTTP SoapAction header element
  - Both URI's must be the same when specified
- Relationship (optional)
  - Type of relationship as QName
  - Predefined: „wsa:Reply“ for replies
  - Message ID of the related message
  - Must be present if the message is a reply
  - URI for unspecified message: *http://schemas.xmlsoap.org/ws/2004/08/addressing/id/unspecified*

```
<wsa:To>xs:anyURI</wsa:To>
<wsa:Action>xs:anyURI</wsa:Action>
<wsa:RelatesTo RelationshipType=„...“>xs:anyURI</wsa:RelatesTo>
```

Anonymous Endpoints

- Resolvable global URI for endpoints is usually difficult (NAT, DHCP, firewall)

→ *http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous*

- Requests with anonymous endpoints for reply / source / fault must provide out-of-band transport for messages
  - HTTP GET / POST request-reply interaction
- Allowed only as destination (“To”) for reply messages

```
<soap:Header>
  <wsa:To>http://host/WidgetService</wsa:To>
  <wsa:ReplyTo><wsa:Address>
     http://schemas.xmlsoap.org/ws/2003/03/addressing/role/anonymous
  </wsa:Address></wsa:ReplyTo>
  <wsa:FaultTo><wsa:Address>
     http://client/myReceiver
  </wsa:Address></wsa:FaultTo>
</soap:Header>
```

SOAP Binding for Endpoint References

- Conformant implementations must support SOAP binding
- WSA address property is copied to the message information header field for the destination („To“)
- Ref. properties / parameters become SOAP header fields
- Fault information maps to SOAP 1.2 constructs

```
<S:Body><S:Fault>
  <S:Code>
    <S:Value>[Code]</S:Value>
    <S:Subcode><S:Value>[Subcode]</S:Value></S:Subcode>
  </S:Code>
  <S:Reason><S:Text>[Reason]</S:Text></S:Reason>
  <S:Detail>[Detail]</S:Detail>
</S:Fault></S:Body>
```

SOAP 1.2 Binding Example

```
<wsa:EndpointReference xmlns:wsa="..." xmlns:fabrikam="...">
   <wsa:Address>http://www.fabrikam123.example/acct</wsa:Address>
   <wsa:ReferenceProperties>
     <fabrikam:CustomerKey>123456789</fabrikam:CustomerKey>
   </wsa:ReferenceProperties>
   <wsa:ReferenceParameters>
     <fabrikam:ShoppingCart>ABCDEFG</fabrikam:ShoppingCart>
   </wsa:ReferenceParameters> </wsa:EndpointReference>
```

```
<S:Envelope xmlns:S=http://www.w3.org/2003/05/soap-envelope
   xmlns:wsa="..." xmlns:fabrikam="... ">
<S:Header> ...
  <wsa:To>http://www.fabrikam123.example/acct</wsa:To>
  <fabrikam:CustomerKey>123456789</fabrikam:CustomerKey>
  <fabrikam:ShoppingCart>ABCDEFG</fabrikam:ShoppingCart> ...
</S:Header>
<S:Body> ... </S:Body>
</S:Envelope>
```

SOAP 1.2 Binding Example (contd.)

```
<S:Envelope xmlns:S=http://www.w3.org/2003/05/soap-envelope
     xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
  <S:Header>
    <wsa:MessageID>
      uuid:6B29FC40-CA47-1067-B31D-00DD010662DA
    </wsa:MessageID>
    <wsa:ReplyTo>
      <wsa:Address>http://bus456.example/client1</wsa:Address>
    </wsa:ReplyTo>
    <wsa:To>http://fabrikam123.example/Purchasing</wsa:To>
    <wsa:Action>http://fabrikam123.example/SubmitPO</wsa:Action>
  </S:Header>
  <S:Body>
  ...
  </S:Body>
</S:Envelope>
```

Formulating a Reply Message

Request Message

```
<s:Envelope xmlns:s="..." xmlns:wsa="..." xmlns:c="...">
  <s:Header>
    <wsa:Action>http://skonnard.com/SubmitClaim</wsa:Action>
    <wsa:To>http://skonnard.com/Claims/Submit.asmx</wsa:To>
    <wsa:ReplyTo>
      <wsa:Address>http://skonnard.com/resp/resp.asmx
      </wsa:Address>
      <wsa:ReferenceProperties>
        <c:PatientProfile>123456</c:PatientProfile>
        <c:CarrierID>987654</c:CarrierID>
      </wsa:ReferenceProperties>
    </wsa:ReplyTo>
  </s:Header>
  <s:Body xmlns:c="http://example.org/claims">
    <c:SubmitClaim> ... </c:SubmitClaim>
  </s:Body>
</s:Envelope>
```

Endpoint Reference

Response Message

```
<s:Envelope xmlns:s="..." xmlns:wsa="..." xmlns:c="...">
  <s:Header>
    <wsa:Action>http://skonnard.com/SubmitClaimResponse
    </wsa:Action>
    <wsa:To>http://skonnard.com/resp/resp.asmx</wsa:To>
    <c:PatientProfile>123456</c:PatientProfile>
    <c:CarrierID>987654</c:CarrierID>
  </s:Header>
  <s:Body xmlns:c="http://example.org/claims">
    <c:SubmitClaim> ... </c:SubmitClaim>
  </s:Body>
</s:Envelope>
```

WS-Addressing - Conclusion

- All relevant information for processing inside of the SOAP envelope
  → no problem with intermediaries or new transport protocols
- Client-side can be unaware of additional information needed by the provider
- Implicit asynchronous support for Web Services
- Standardized way to pass around references to Web Services

Message Transmission Optimization Mechanism (MTOM)

- W3C Proposed Recommendation
- Microsoft, IBM, BEA, Canon
- No implementing product
- Optimization of the transmission and/or wire format of SOAP messages
- Based on XOP (XML-binary optimized packaging)
  - MIME / Multipart serialization of SOAP messages
  - XML document as root part
  - Base64 data comes in separate MIME parts

Agenda

- Introduction
- Base specifications
- **Metadata specifications**
- Notification
- Other standards
- Conclusion

WS-MetadataExchange

- Microsoft, IBM, SAP, BEA, SUN and others
- Web Service consumers need metadata to interact with them
  - Messages, protocols, endpoint addresses
  - Policies (capabilities, requirements, …)
- Simple interaction to get service description
  → communication bootstrapping
- Request for specific metadata type
- SOAP 1.2, WSDL 1.1
- Intended for retrieval of WSDL, WS-Policy and XML Schema information for endpoints or target namespaces

Retrieving Metadata

- Metadata dialect / identifier
  - Response must include only Metadata Sections that are related to the indicated dialect / identifier
  - Specified as URI, interpreted by the service
- Metadata Section element content
  - requested information
  - Metadata Reference (Endpoint Reference)
    - GetMetadata operation
  - Metadata Location (URL)
    - HTTP GET operation

GetMetadata Request Message

```
<s:Envelope ...>
  <s:Header ...>
    <wsa:Action>
       http://schemas.xmlsoap.org/ws/2004/09/
              mex/GetMetadata/Request
    </wsa:Action>
    <wsa:MessageID>xs:anyURI</wsa:MessageID>
    <wsa:ReplyTo>endpoint-reference</wsa:ReplyTo>
    <wsa:To>xs:anyURI</wsa:To>
    ...
  </s:Header>
  <s:Body>
    <wsx:GetMetadata ...>
      [<wsx:Dialect>xs:anyURI</wsx:Dialect>
        [<wxs:Identifier>xs:anyURI</wsx:Identifier>]
  ]
    </wsx:GetMetadata>
  </s:Body>
</s:Envelope>
```

GetMetadata Example - Request

```
<s12:Envelope
   xmlns:s12='http://www.w3.org/2003/05/soap-envelope‘
   xmlns:wsa='http://schemas.xmlsoap.org/ws/2004/08/addressing'
   xmlns:wsx='http://schemas.xmlsoap.org/ws/2004/09/mex'>
<s12:Header>
   <wsa:Action>
       http://schemas.xmlsoap.org/ws/2004/09/
              mex/GetMetadata/Request
   </wsa:Action>
   <wsa:MessageID>4f6d5c6027f4</wsa:MessageID>
   <wsa:ReplyTo>
      <wsa:Address>http://example.com/MyEndpoint</wsa:Address>
   </wsa:ReplyTo>
   <wsa:To>http://server.example.org/YourEndpoint</wsa:To>
</s12:Header>
<s12:Body>
   <wsx:GetMetadata>
   <wsx:Dialect>http://schemas.xmlsoap.org/wsdl/</wsx:Dialect>
</s12:Body>
</s12:Envelope>
```

GetMetadata Example - Response

```
<s12:Envelope
   xmlns:s12='http://www.w3.org/2003/05/soap-envelope‘
   xmlns:wsa='http://schemas.xmlsoap.org/ws/2004/08/addressing'
   xmlns:wsx='http://schemas.xmlsoap.org/ws/2004/09/mex' >
<s12:Header>
   <wsa:Action>
       http://schemas.xmlsoap.org/ws/2004/09/
              mex/GetMetadata/Response
   </wsa:Action>
   <wsa:RelatesTo>4f6d5c6027f4</wsa:RelatesTo>
   <wsa:To>http://client.example.com/MyEndpoint</wsa:To>
</s12:Header>
<s12:Body>
   <wsx:Metadata>
     <wsx:MetadataSection
        Dialect='http://schemas.xmlsoap.org/wsdl/'>
          <wsdl:definitions … </wsdl:definitions>
     </wsx:MetadataSection>
   </wsx:Metadata>
</s12:Body>
</s12:Envelope>
```

WS-Policy

- General-purpose model to describe and communicate the policies of a web service (requirements, preferences, and capabilities)
- Policy: collection of policy alternatives
- Alternative: set of domain-specific policy assertions
- Policy assertion examples
  - ‘wire’ properties (authentication scheme, transport protocol)
  - Extended properties (privacy policy, QoS characteristics)
- Policy operators are describing acceptable combinations of assertions
  - **All** - all its child elements must be satisfied
  - **ExactlyOne** - exactly one of its child elements must be satisfied

WS-Policy Example

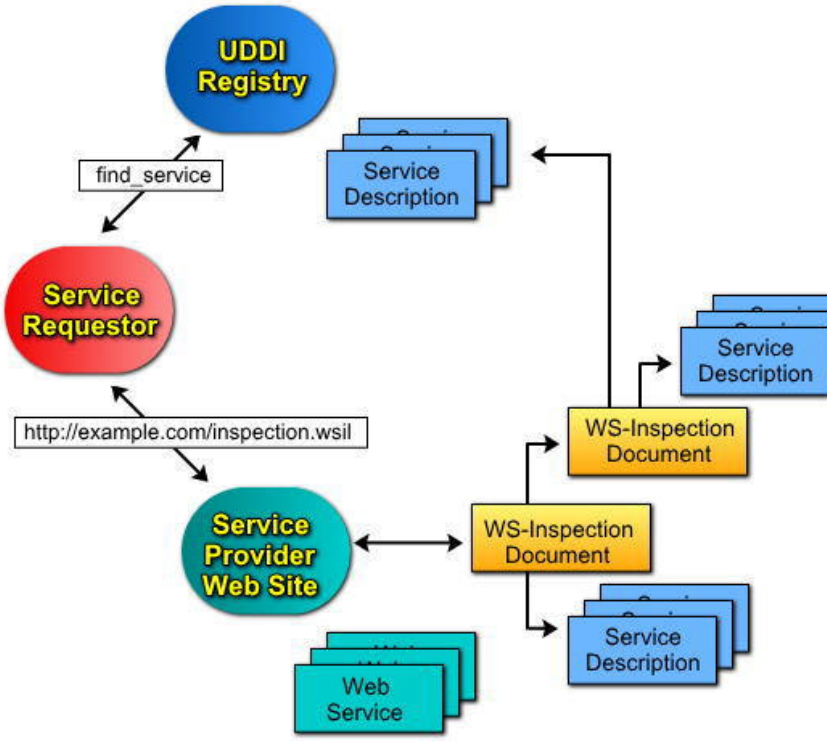
```
<wsp:Policy>
  <wsp:ExactlyOne>
    <wsp:All>
      <wsse:SecurityToken>
        <wsse:TokenType>wsse:Kerberosv5TGT</wsse:TokenType>
      </wsse:SecurityToken>
    </wsp:All>
    <wsp:All>
      <wsse:SecurityToken>
        <wsse:TokenType>wsse:X509v3</wsse:TokenType>
      </wsse:SecurityToken>
    </wsp:All>
  </wsp:ExactlyOne>
</wsp:Policy>
```

WS-Policy Extensions

- WS-PolicyAttachment
  - general-purpose mechanisms for associating policies with the subjects to which they apply
  - reference policies from WSDL definitions
  - associate policies with deployed WS endpoints
  - associate policies with UDDI entities
- WS-SecurityPolicy
  - WS-Policy assertions that apply to WS-Security

WS-Inspection Language

- IBM, Microsoft
- multiple implementations
- Service provider offers WSIL files
  - Inspection.wsil at common web site entry point
  - Link from another document
- Service descriptions in external formats (e.g. WSDL or UDDI entry)



WS-Inspection Example

```
<?xml version="1.0"?>
<inspection xmlns="http://schemas.xmlsoap.org/ws/2001/10/inspection/">
  <service>
      <description
           referencedNamespace=http://schemas.xmlsoap.org/wsdl/
         location="http://example.com/exampleservice.wsdl" />
  </service>
  <service>
      <description referencedNamespace="urn:uddi-org:api">
         <wsiluddi:serviceDescription
              location="http://example.com/uddi/inquiryapi">
            <wsiluddi:serviceKey>52946BB0</wsiluddi:serviceKey>
       </wsiluddi:serviceDescription>
     </description>
  </service>
</inspection>
```

Agenda

- Introduction
- Base specifications
- Metadata specifications
- **Notification specifications**
- Other standards
- Conclusion

WS-Eventing

- BEA, Microsoft, and others
- Publish / subscribe system with source and sink
- Subscription events
  - Subscribe (sent by the event sink)
  - Renew (sent by the event sink)
  - GetStatus (sent by the event sink)
  - Unsubscribe (sent by the event sink)
  - End (sent by the event source)
- Source sends notifications until
  - the subscription expires without renewal
  - the event sink sends an unsubscribe event
  - the event source cancels the subscription

WS-Eventing Subscription Example (1/2)

```
<s12:Envelope
    xmlns:s12='http://www.w3.org/2003/05/soap-envelope'
    xmlns:wsa='http://schemas.xmlsoap.org/ws/2004/08/addressing'
    xmlns:wse='http://schemas.xmlsoap.org/ws/2004/08/eventing'
    xmlns:ew='http://www.example.com/warnings' >
  <s12:Header>
    <wsa:Action>
        http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe
    </wsa:Action>
    <wsa:MessageID>
        uuid:e1886c5c-5e86-48d1-8c77-fc1c28d47180
    </wsa:MessageID>
    <wsa:ReplyTo>
      <wsa:Address>http://www.example.com/MyEventSink</wsa:Address>
      <wsa:ReferenceProperties>
         <ew:MySubscription>2597</ew:MySubscription>
      </wsa:ReferenceProperties>
    </wsa:ReplyTo>
    <wsa:To>http://www.example.org/oceanwatch/EventSource</wsa:To>
  </s12:Header>
```

WS-Eventing Subscription Example (2/2)

```
<s12:Body>
   <wse:Subscribe>
     <wse:EndTo>
       <wsa:Address> http://www.example.com/MyEventSink </wsa:Address>
       <wsa:ReferenceProperties>
           <ew:MySubscription>2597</ew:MySubscription>
       </wsa:ReferenceProperties>
     </wse:EndTo>
     <wse:Delivery>
       <wse:NotifyTo>
         <wsa:Address>http://www.example.com/OnStormWarning</wsa:Address>
          <wsa:ReferenceProperties>
            <ew:MySubscription>2597</ew:MySubscription>
          </wsa:ReferenceProperties>
        </wse:NotifyTo>
     </wse:Delivery>
     <wse:Expires>2004-04-26T21:07:00.000-08:00</wse:Expires>
   </wse:Subscribe>
</s12:Body>
</s12:Envelope>
```

WS-Notification

- IBM, Sonic, TIBCO, Akamai, SAP, HP
- Submitted to OASIS
- IBM works on interoperability to WS-Eventing since September 2004,
  Microsoft joined the effort
- Designed for Grid applications (WSRF)
- More features than WS-Eventing

WS-Notification documents

- Web Service Base Notification
  - Interfaces for consumers and producers
- Web Services Topics (WS-Topics)
  - Mechanisms to organize and categorize items of interest for subscription
  - Topic expression dialects for subscription
- Web Services Brokered Notification
  - Interface for Notification Broker
  - Intermediary that allows publication of messages from entities that are not service providers

WS-Notification Goals

- Support for small devices (restricted set of mandatory features)
- Support for direct and brokered notification
- Transformation and aggregation of Topics
- Runtime metadata
  (e.g. available subscription types)
- Broker federations
- Transport protocol independence
- Based on WS-ResourceProperties and WS-ResourceLifetime (from WSRF)

Base Notification

- Two ways of notification
  - Producer sends application-specific content
  - Producer send special *Notify* message
    - Allows additional information (e.g. Topic)
    - Consumer can receive a range of notification messages without explicit support
    - Batch transfer of notification messages
- Type is indicated during the subscription
- No response is expected

Notify Message

```
<wsnt:Notify>
  <wsnt:NotificationMessage>
  <wsnt:Topic dialect="xsd:anyURI">
    {any}
  </wsnt:Topic>
  <wsnt:ProducerReference>
    wsa:EndpointReference
  </wsnt:ProducerReference>
  <wsnt:Message>xsd:any</wsnt:Message>
  </wsnt:NotificationMessage>
</wsnt:Notify>
```

Notification Metadata

- Producer interface must support a set of resource properties
  - Topics (0-n): collection of supported topics
  - FixedTopicSet (1): indicates whether the collection of Topics can change
  - TopicExpressionDialects (1-n): supported set of topic dialects

Subscription Parameters

- Consumer reference
  - WS-Addressing endpoint reference
- Topic expression
- “Use notify” flag
  - Decides whether or not the notification message is used
- Selector
  - Expression that must evaluate to “true”
  - Syntax and meaning is defined by the producer
- SubscriptionPolicy
- InitialTerminationTime
  - Mandatory termination time in the future
  - Suggestion by the requestor
  - Relative to the time source of the producer

Subscribe Message

```
<wsnt:Subscribe>
  <wsnt:ConsumerReference>
    wsa:endpointReference
  </wsnt:ConsumerReference>
  <wsnt:TopicExpression dialect = “xsd:anyURI”>
    {any}
  </wsnt:TopicExpression>
  <wsnt:UseNotify>xsd:boolean</wsnt:UseNotify>
  <wsnt:Precondition>wsrp:QueryExpression</Precondition>
  <wsnt:Selector>wsrp:QueryExpression</wsnt:Selector>
  <wsnt:SubscriptionPolicy>{any}</wsnt:SubscriptionPolicy>
  <wsnt:InitialTerminationTime>
    xsd:dateTime
  </wsnt:InitialTerminationTime>
</wsnt:Subscribe>
```

Subscription Results

- Endpoint reference to *subscription manager* is returned
  - Offers subscription resource properties
  - Allows *PauseSubscription* request
  - Allows *ResumeSubscription* request
- Notification producer allows *GetCurrentMessage* request
  - May result in a re-sending of the last notification message

Agenda

- Introduction
- Base specifications
- Metadata specifications
- Notification specifications
- **Other standards**
- Conclusion

Security

- WS-Security
  - protocol neutral mechanism for securing SOAP messages
  - builds upon XML-Signature and XML-Encryption
  - specifies how security tokens can be associated with messages
- WS-Trust
  - WS-Security extension for requesting and issuing security tokens
- WS-SecureConversation
  - defines mechanisms for establishing and sharing security contexts, e.g. for anonymous or mutual authentication
- WS-Federation
  - mechanisms for allowing and brokering trust of identities, attributes and authentication between Web Services

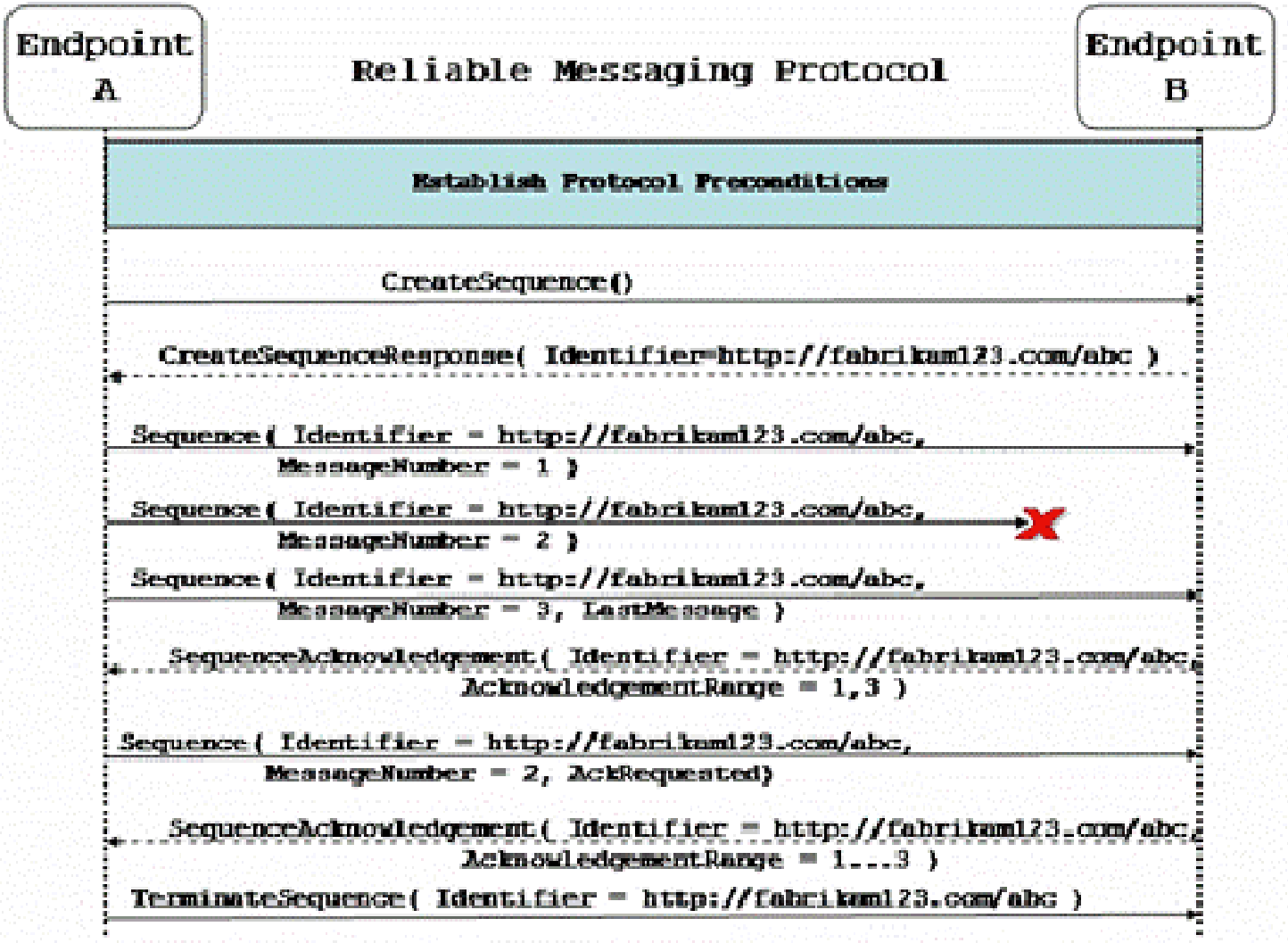
Fault Tolerance

- WS-Coordination
  - Framework for the definition of coordination types
  - Coordinator and coordination protocols
  - Creation and propagation of a new coordination context
- WS-Transaction
  - WS-AtomicTransaction and WS-BusinessActivity
  - reach consistent agreement on the outcome of distributed transactions → coordination protocol

Reliability

- WS-Reliability
  - SUN and others, OASIS submission
  - allows messages to be delivered reliably in a network-transport-independent manner
- WS-ReliableMessaging
  - IBM, Microsoft, BEA
  - Similar concept with some basic ideas
    - message sequence numbers
    - sequence creation, termination and acknowledgement messages

WS-ReliableMessaging Example



Grid Computing

- Web Service Resource Framework (WSRF)
- Based on WS-Addressing
- Concepts for stateful resources that are accessible through a Web Service
- Akamai, IBM, HP, Globus, Tibco, Fujitsu
- Upcoming implementations (Axis, Globus 4, IBM ETTK)

Web Service Resource Framework

- Access to stateful resources using Web Services – “Implied Resource Pattern”
- Family of specifications
  - WS-ResourceProperties
  - WS-ResourceLifetime
  - WS-RenewableReferences
  - WS-ServiceGroup
  - WS-BaseFaults

As Well As …

- WS-Discovery
  - multicast-based ad-hoc discovery protocol to locate available Web services
  - Intended for wireless and mobile devices
  - “Discovery Proxies” concept
- BPEL4WS
  - XML-based workflow definition language for web service invocation
- WS-MessageDelivery
  - Specify message exchange patterns and delivery properties

Conclusion

- Some important specs
- Many domain-specific specs
- Document quality differs, depends on the companies of the authors
- Fast evolving of implementations, leaded by IBM (ETTK) and Microsoft (WSE)