Basic Web Services Technologies Review

- XML Technologies
  - "Extensible Markup Language"
  - Base XML for documents
  - XML Schema for describing XML documents
- SOAP
  - A simple way to send documents (some people have called it "email for documents")
  - How to format XML documents for transmission
- WSDL - Web Services Description Language
  - Defines implementation details about a service
- UDDI - Universal Description, Discovery and Integration
  - Store, advertise and discover services

IONA

Basic Web Services Architecture

Advanced Web Services Topics

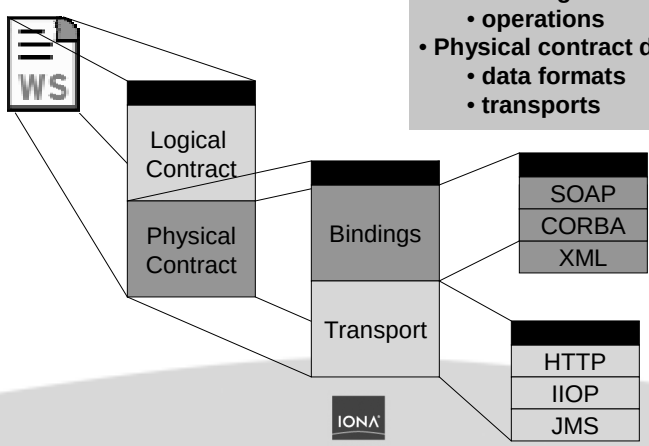
- Enterprise Web Services
- Industry trends and organizations
- Security and Reliability
- Transactions
- Metadata and Policy
- Choreography and Orchestration
  Composite web services
- Interoperability Requirements
  Encoding discussion
- Futures

Enterprise Web Services

- Extending WS throughout the enterprise and among enterprises ⇒ greater emphasis on Qualities Of Service (QoS)
  - Security, Reliability, Transactions, Scalability
  - Systems Management, Policy, Metadata Management/Governance
  - Interoperability
- It's also important to support multiple transports and data formats (other than SOAP over HTTP):
  - IIOP & RMI
  - WebSphereMQ
  - TIBCO Rendezvous
  - Tuxedo
  - FTP
  - JMS

Extending WSDL Contracts

- **Logical contract defines**
  - **data types**
  - **messages**
  - **operations**
- **Physical contract defines:**
  - **data formats**
  - **transports**

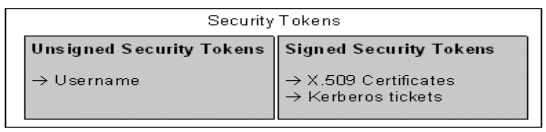


Security

- Confidentiality – no one can see my data
- Integrity – my data gets there in one piece
- Authentication – a user or program is confirmed to be who they say they are
- Authorization – ensuring access to services and data only by those who are supposed to access them
- No overall architecture yet defined

  May include firewalls, proxies, security servers, and identity management
- WS-Security is a well-agreed framework (OASIS)

SOAP Header example (WS-Security)



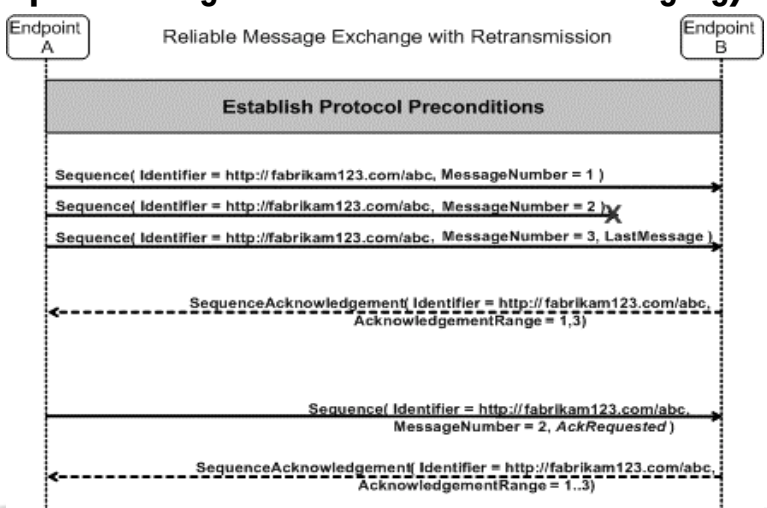

```
<S11:Envelope xmlns:S11="..." xmlns:wsse="..." xmlns:wsu="..." xmlns:ds="...">
  <S11:Header>
    <wsse:Security  xmlns:wsse="...">
      <fabtok:CustomToken wsu:Id="MyID"  xmlns:fabtok="http://fabr123/token" />

      <ds:Signature>  . . . </ds:Signature>
      <ds:SignedInfo> . . . </ds:SignedInfo>
    </wsse:Security>
  </S11:Header>
...
</S11:Envelope>
```

Reliability requirements

- Reliability of message delivery is critical to many applications
  - Guarantee of message delivery (both request and response)
    - At least once (duplicates are allowed)
    - At most once ("best effort", ok to drop some messages)
    - Once and only once – the toughest to guarantee!
  - Eliminate duplicates (if any)
  - Preserve message ordering (when important)
- Some of these features are provided as standard when WS are carried on a connection-oriented protocol
  - The "sessions" and correlation IDs of the connection-oriented protocol ensures features such as ordering
  - But some QoS must still be added (e.g., store-and-forward to allow clients to send messages even if a server isn't running)

Example message flow: WS-ReliableMessaging)



Web Services: Implications for TP

- Persistent sessions are missing from HTTP
  - Need to define a common context sharing mechanism
  - Transport “agnostic” means LCD
    - It’s difficult to write a new WS spec that assumes a more sophisticated protocol than HTTP
- Need a solution that is also suited for document-oriented transactions
  - Compensation, transactional “queues” needed
  - Reliable messaging needed
- Business process orchestration
  - Cancel/adjust transactions in long-running flight
  - Mixture of existing and new technologies

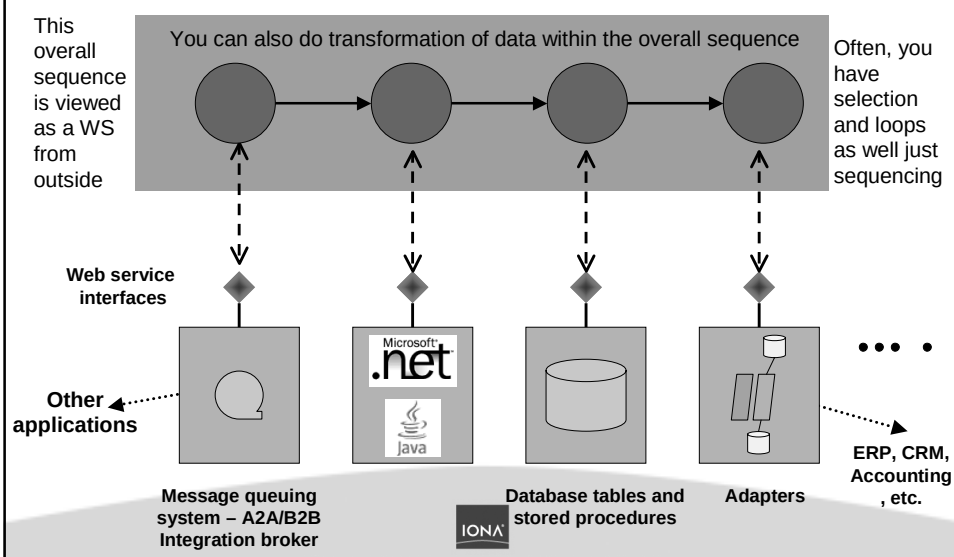
Current status of standardization

- OASIS WS-Composite Application Framework (WS-CAF)
  - Factors out context into WS-Context specification
  - Coordinates across different types of systems
    - Resolves the differences in the co-ordinator, not in the end-points
- OASIS WS-TX:
  - WS-Coordination
  - WS-Atomic Transactions
  - WS-Business Activity

Web Services Composition

- Once you have a “critical mass” of Web services, you may want to:
  - link them together to create (composite) complex Web services
  - implement whole business processes based on those services
- Of course you can “hard-code” this, but the result is
  - inflexible (hard to maintain)
  - proprietary, not explicit (business logic will be buried in code)
- Better: Standardized high-level mechanisms
  - Standardized declarative, non-programmatic way to compose sequences of web service calls to accomplish a business function
- Efforts ongoing to provide
  - WS-BPEL (seems to be winning at present) for run-time
  - WS-Choreography (more B2B oriented) between partners

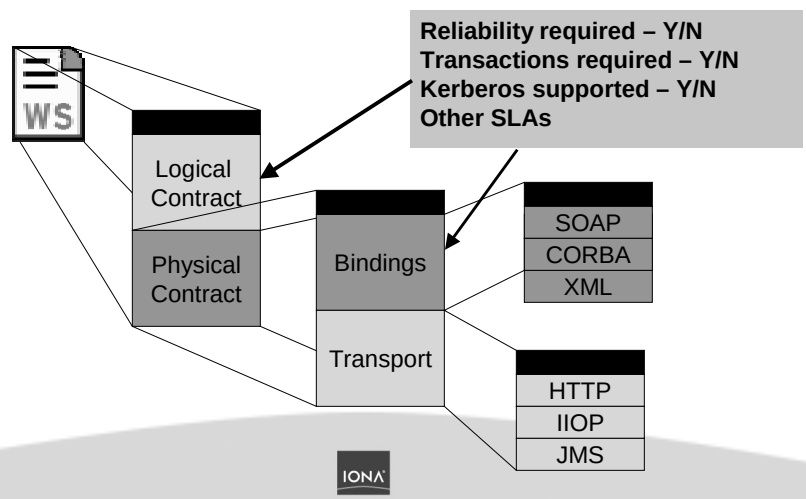
Orchestration Architecture



Metadata and policy requirements

- Need to add information to WSDL to specify:
  - Reliability
  - Security
  - Transactions
  - High Availability, Service Level Agreements, etc.
- WSDLs define only part of the SOA contract
  - Need to attach policy assertions to define quality of service and service level agreement requirements
  - Can use WS-MetadataExchange to retrieve
  - Store and manage in UDDI

Extending WSDL Contracts - Policy



Interoperability Requirements

- Within the enterprise and among enterprises
  Across hardware platforms, operating systems, programming languages, middleware and component technologies
- SoapBuilders and WS-I have been working to correlate the many WS standards into more interoperable configurations. E.g.,
  Defining a consistent selection of standards
  Subsetting ("profiling") a standard
- Ensuring compatible message *encoding* is another key part of interoperability
  As discussed in the new few slides

You have a choice of the encoding for complex data in msgs

- The WS-I recommendation is *literal*
  This uses XMLSchema in the normal way
  - The message references a schema that defines the elements and types for formatting arrays, sequences, records/structures, and other complex types
  - The message uses these elements and types
- The alternative is SOAP *encoded*
  This defines a set of elements that can be used for complex data
  - Mostly, these elements were good at formatting data that typically arises in RPC calls

  It was first introduced before XMLSchema was available
  - Harder to achieve broad interoperability
  - Especially when using more complex data types

You also have a choice between two message styles

- The choice is *rpc* or *document*
  document style is recommended by WS-I
  many existing Web services use *rpc* style)
- Operation have input and (optional) output messages
  In *rpc* style, each message can have zero or more parts

```
<message name="getQuoteRequest">
  <part name="symbol" type=s:string">
  <part name="date" type="s:string>
</message>
```

  In *document* style, each message have one part
  An XML element, not a type

WS-I has yet to profile SOAP 1.2

Industry organizations

- Developing Web services standards
  W3C
  - XML, SOAP (XMLP working group), WSDL, WS-Addressing, WS-Choreography, Semantic Web Services

  OASIS
  - UDDI, WS-Security, WS-Reliable Exchange, WS-Transactions, WS-CAF, WS-Notification, WS-Secure Exchange, etc.

  JCP (Java Community Process)
  - JAX-RPC, JWSDL, JAXM, JAXR, …

  WS-I
  - Basic profile, security profile, etc.

  OMG
  - IDL to WSDL, WSDL to IDL (WSDL 1.1), WSDL to C++

Futures -

- Advances in Standards
  - Ratification of newer versions of standards (SOAP, WSDL, etc.)
  - Emergence of newer standards to support the evolution of WS into enterprise environments
  - Service Component Architecture (SCA)
  - SOA Tools (Eclipse)
- Advances in deployment support
  - Additional availability of transports/protocols in distributed infrastructures
- Interoperability advances
  - More shake-out of interoperability issues