V22.0480-004
Web Services Architecture and Programming

Lecture 18
Overview of GXA Specifications
WS-Routing and Web Service Intermediaries

Announcements

• Lab 5 available now from web site, due Nov. 26th (Wednesday)
• Lab 6 will be available from web site on November 26th
  – Due back December 9th
• Final exam: December 18th
  – Will hand out sample questions in the next couple of classes
• No class on Thursday, November 20th

Global XML Web Services Architecture (GXA)

• An attempt by Microsoft, IBM, BEA, others to define a set of higher-level specifications on top of the core web services specifications

• Core specifications define client-service and client-broker-service interactions
  – SOAP, WSDL, UDDI

• GXA specifications build on above to define how groups of web services can interact with each other
  – Need these specifications to allow construction of more complex web service applications
  – E.g., an online book store that requires to interact with a credit card web service to verify the user’s credit card number
  • Database updates to the book service and credit card service must either both happen, or neither should happen

GXA Specifications (Still Evolving)

• WS-Inspection
  – A simpler UDDI-like discovery protocol
  – Caters to scenarios where source can directly announce availability

• WS-Routing, WS-Referral

• WS-Security
  – Specifies how security credentials are passed in SOAP messages, how SOAP actors should act on them …

• BPEL4WS
  – Encoding of “business process activities”
  – First invoke this service, then use its results to invoke other services, …

• WS-Transaction (old), WS-Coordination (September 2003)
  – Atomic actions involving multiple services
  • All encoded as special headers in the SOAP message
WS-Routing and WS-Referral

- WSDL specifies end-to-end connection (from client to service URL)
- Need for more general structures
  - Peer-to-peer and store-and-forward networking
  - Should be possible to send messages to distributed processing nodes (despite being named by the same URL)
- WS-Routing
  - Enables specification of a complete message path for the message (including its return path)
- WS-Referral
  - Permits routing between SOAP nodes on a message path to be dynamically configured
  - Allows delegation of part or all of processing responsibility to other nodes
- Together, permit use of intermediaries in web services applications
  - Caches, load-balancing agents, transcoders, ...

Motivating Example

- Consider the application you put together as part of Labs 4 and 5

Motivating Example with Intermediaries

- Caching and transcoding intermediaries

Routing Web Service Requests via Intermediaries

- Common structure behind example intermediaries
- We could realize these more sophisticated applications if there was a way of specifying the path along which a request/response message must be routed prior to reaching its eventual destination
- WS-Routing protocol provides this functionality
WS-Routing

- Defines a new SOAP header element named path
  - From the http://schemas.xmlsoap.org/rp namespace

```xml
<wsrp:path xmlns:wsrp="http://schemas.xmlsoap.org/rp">
  <wsrp:action /> <!-- indicates message intent -->
  <wsrp:to /> <!-- identifies ultimate receiver -->
  <wsrp:fwd /> <!-- identifies forward intermediaries -->
  <wsrp:via /> <!-- identifies an intermediary node -->
  <wsrp:rev /> <!-- identifies reverse intermediaries -->
  <wsrp:from /> <!-- identifies sender -->
  <wsrp:id /> <!-- uniquely identifies this message -->
  <wsrp:relatesTo /> <!-- correlates message with another -->
  <wsrp:fault /> <!-- provides extra fault details -->
</wsrp:path>
```

.NET Support for WS-Routing

- Part of the Web Services Extensions (WSE) package
  - `Microsoft.Web.Services` assembly

Enables construction of WS-Routing aware

- Web services
  - Must be able to handle the additional headers
  - Can act upon information encoded in the headers (e.g., security tokens)

- Web service clients
  - Can add header fields (<via> elements in WS-Routing)

- Web service intermediaries
  - Can extract header fields, interpret them, modify both header/body

Building WS-Routing Aware Applications in .NET

[Code walkthrough]

- Need to use the D:\VSDev\Public\ folder
  - WSE poorly integrated with HTTP-level Windows authentication

- Creating web services
  - Adding `soapExtensionTypes` info to web.config
  - `[WebMethod]` attribute works as earlier

- Creating web clients
  - Proxy inherits from a different class
  - Access to `{Request,Response}SoapContext` objects

- Creating web service intermediaries
  - Endpoint, RouterBasic, RouterForward, RouterReverse, RouterAdvanced