Asynchronous and Synchronous Messaging with Web Services and XML

Ronald Schmelzer
Senior Analyst
ZapThink, LLC

The Business Objective

- **Automated Business Collaboration**
  - Facilitating *exchange of information* between systems, organizations, and markets
  - Allowing line-of-business managers to *represent business processes*, and IT organizations to enable them
  - *Promote business agility* by allowing processes to be defined, executed, and changed as needed
### A Web Services Roadmap

#### ZapThink Web Services Roadmap

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Static binding to static Services</td>
<td>Dynamic binding to static Services</td>
<td>Discover &amp; bind at runtime (JIT integration)</td>
<td><strong>Primary use</strong></td>
</tr>
<tr>
<td></td>
<td>Internal Focus</td>
<td>External (B2B) Focus</td>
<td>Full embedding of Web Services</td>
<td><strong>Invocation style</strong></td>
</tr>
<tr>
<td>Security</td>
<td></td>
<td></td>
<td></td>
<td>Security</td>
</tr>
<tr>
<td>Management</td>
<td></td>
<td></td>
<td></td>
<td>Management</td>
</tr>
<tr>
<td>Transactions</td>
<td></td>
<td></td>
<td></td>
<td>Transactions</td>
</tr>
<tr>
<td>Billing &amp; Metering</td>
<td></td>
<td></td>
<td></td>
<td>Billing &amp; Metering</td>
</tr>
<tr>
<td>Orchestration &amp; Workflow</td>
<td></td>
<td></td>
<td></td>
<td>Orchestration &amp; Workflow</td>
</tr>
<tr>
<td>Business Web</td>
<td></td>
<td></td>
<td></td>
<td>Business Web</td>
</tr>
<tr>
<td>Process Automation</td>
<td></td>
<td></td>
<td></td>
<td>Process Automation</td>
</tr>
</tbody>
</table>

Source: Copyright © 2003 ZapThink LLC

### Components of the Problem

- Reliable Messaging
- Reliable Transactions
- Process Definition
- Process Execution
Messaging

• Get a Message from Point A to Point B
  
  – "The greatest problem in communication is the illusion that it has been accomplished." -- Daniel W. Davenport
  
  – "Communication works for those who work at it." -- John Powell
  
  – "It was impossible to get a conversation going, everybody was talking too much." -- Yogi Berra

But what about processes…?  

• One-step messaging is relatively easy
• Multi-step messaging is difficult

... but for this presentation, we’ll focus on just the first part – reliable messaging for single-step processes...
Web Services *Idées Fortes*: Loose Coupling

- Consumer and Producer controlled by different people
- Changing one doesn’t break the other
- Build one without being aware of the other

Web Services *Idées Fortes*: Coarse Granularity

- Business-oriented requests and responses
- Blocks of information exchanged
- *Encapsulate* APIs into fine-grained, atomic Services and *compose* them into coarse-grained, business Services
Web Services **Idées Fortes**: Asynchrony

- The Web is *synchronous*: click a button and wait for a response
- Web Services can also be *asynchronous*: allow for long-running processes
Communications Models

- Synchronous
  - Request and Response
- Asynchronous
  - Or “Message-oriented”
  - Loosely-coupled systems (sounds like XML?)
- Fire-and-Forget
  - Message is sent, but we don’t care about getting any response

Benefits of Synchronous

- Real-time
- Efficient protocol
- The most “obvious” model
  - I ask a question
  - I wait for a response

- Problem: not all problems can be solved synchronously
Benefits of Asynchronous

- More closely follows natural business and process logic
- Allows processing of multiple concurrent processes
- Fire and Forget a Special Case
- Challenges: more overhead, reliability issues

Why Should Service-Oriented Architectures be Asynchronous?

- Fundamental tenet of loose coupling: not being aware of end point requirements
- Composited (virtualized) Web Services may require greater time for processing, requiring asynchrony
- B2B processes are often asynchronous
- Distributed systems can be more reliable when they are asynchronous
- Heterogeneous systems, especially those with limited bandwidth devices, function better asynchronously.
- Support human involvement in processes
Messaging Models

• Store-and-forward
  – Messages delivered to a one or more coordinating authorities
  – These forward to end destination
  – Post office model
  – SMTP

• Point-to-point
  – Real-time connection between systems

What is a Transport Protocol?

• According to OSI Model: TCP
• According to Jabber, et. al: HTTP
• According to W3C: SOAP

• But, semantics:
  – TCP is really a transport protocol
  – HTTP is a transfer protocol
  – SOAP is a messaging protocol (even though ebXML identifies a TRP layer)
Wire-level vs. Business-level

- Wire-level Transport
  - Getting point A and point B to talk to each other
  - The carrier
  - HTTP, SMTP, FTP,…

- Business-level Transport
  - Getting point A and point B to understand each other
  - The envelope
  - SOAP, ebXML, XML-RPC, …

Transport Protocols

- TCP/IP
  - If we’re talking Internet, this is all we’re talking about
  - Ok, UDP too, but not really much any more

- LAN Transport protocols
  - Windows
  - UNIX
  - But, we don’t care much about these any more

- EDI VANs
  - XML over the VAN?

- Direct Dial-up
  - What, are we still in the 80’s (or early 90’s)?
Transfer Protocols

• Here’s where it gets more interesting
• HTTP
  – The basis of the Web
  – Single Connection Request / Response
• SMTP
  – The basis for email (along with POP)
  – Asynchronous communication mechanism
• FTP
  – Meant for file transfer over Internet
  – Synchronous communication mechanism

HTTP

• Synchronous Request / Response
• The Goal of HTTP
  – I want a web page
  – Here it is, now go away.
• Advantages of HTTP
  – Ubiquitous and Familiar
  – Lots of existing code
  – Firewall friendly
  – Compact
• Uses Octet-counting (the header says how big the body is)
  – The Server doesn’t need to look at the data
Problems with HTTP

- Conversation is Unilateral
  - A Client initiates the request
  - The Server terminates it
  - The session doesn’t persist past one “conversation”
- Requests have mandated length
  - Servers can truncate requests that are too long

SMTP

- Been around since 1981
- Individual Connection is Synchronous, but overall process is Asynchronous
- Send a message, wait for response
  - Seconds
  - Hours
  - Days
  - Never
- Goal of SMTP
  - Here’s some mail.
- Uses Octet-stuffing (line-oriented)
  - The server needs to look at the data
FTP

• Synchronous
• Here’s a File
  – Did you get it?
  – File interruptions
• Uni-directional
  – No real opportunity to respond
  – No real reason to respond
• Uses Connection-Blasting
  – New connections established for each exchange
  – Separate command and data pipes

What we REALLY need

• Reliability
  – Connection reliability
  – Non-repudiation
  – Transaction Control
• Security
  – Encryption
  – Authentication
  – Authorization
  – (Privacy)
• Synchronous and Asynchronous Modes
• Bi-directional Communication
The EDI VAN

- EDI in a Nutshell
- What the VAN offered (offers)
  - A Private Network
  - Security
  - Non-Repudiation
  - Transaction Control
  - A degree of commonality

A Better HTTP

- HTTPR
  - “Reliable delivery of HTTP packets between the server and client.” -- IBM
- HTTP 1.1 is the base protocol
  - SSL
  - Proxies and Firewalls
  - Keep-alive, etc.
- Features
  - Transaction control (once and only once)
  - Non-repudiation (guaranteed delivery)
- Mechanism
  - Keep trying to send until acknowledgement or timeout
- Challenges
  - Server needs persistent store
  - Server needs to record steps
- The Idea of the End-point manager
  - Queuing
P2P or P2P?

• Point-to-Point (ok, Client/Server)
  – One side designated a server, the other client
  – Browser and Server
  – HTTP, SMTP, FTP…
• Peer-to-Peer (the real P2P)
  – Any side can be a server or client for a particular conversation
  – Instant Messaging
  – Napster
  – Some say Client/Server is a subset of P2P
Jabber

- Originally: an open-source IM System
- Real-time messaging
  - Not store-and-forward
- Client/server model
  - Coordinated through a server
  - Subsequent application-specific activities are client-to-client
- Knowledge of availability
  - Presence
  - Servers know when a user is online
- A Network of Servers
  - Each user associated with a particular server
  - Jabber ID’s similar to email addresses
- Very Simple client: Just TCP and XML!

Application-level Transport

- The Envelope
- Components of an Envelope
  - The Addressee (Destination)
  - Return Address (Source)
  - Transaction Control (The Stamp)
  - Non-Repudiation (Return receipt and cancellation)
  - Security (non see-through envelope)
- And some extra stuff…
  - Routing (Behind the scenes at the USPS)
Components of the Problem… Revisited

- Reliable Messaging
- Reliable Transactions
- Process Definition
- Process Execution

Thank You!

Jason Bloomberg & Ron Schmelzer of ZapThink are coauthors of XML and Web Services Unleashed.

ZapThink is an industry analysis firm focused exclusively on XML and Web Services.

Find out more about ZapThink at www.zapthink.com

Ronald Schmelzer
rschmelzer@zapthink.com