WebServices - Axis

1. WebServices - Axis -

Notes from Axis Face-to-Face Meeting at Allaire, Newton, MA. Date: 28 Feb 2001 Attendees:

- Glen Daniels, Allaire. Host
- Doug Davis, IBM Raleigh
- Steve Graham, IBM Raleigh
- Jim Stearns, HP Redmond
- Jacek Kopecky, Idoox, Prague
- Waqar Sadiq, Vitria
- James Snell, IBM Fresno (by phone)
- Stuart Williams, HP Bristol

Glen expressed some frustration at the lack of good tools that Apache provides for project management. The mailing list archives kind of bite, etc. SourceForge seems to have good tools for scheduling / task management / collaboration...

ACTION ITEM : Glen will email Sam and ask about the possibility of "co-locating" the project with SourceForge? Investigating other tools and solutions?

Waquar Sadiq from Vitria, a new member of the team, volunteered to be Project Manager for Axis! (applause applause)

Steve presented a short Powerpoint presentation to frame the meeting.

Axis 1.0 will be a serious product. As such, it will have:

- product-level code
- product-level testing
- product-level doc + examples
- product-level performance

Goals for the 1.0 release:

- Full SOAP 1.1 implementation
  - Interoperability testing/discussions to answer ambiguities
  - Intermediary implementation must be in 1.0 release
- Relative priority (Glen)
  - Full implementation of 1.1
  - Customer requirements

    - Where xmlp is headed (Glen, Waqar, Jacek are all on XML Protocol group)
- Fast (need to do the streaming parsing)
- Pluggable

Support for requestor/provider/intermediary (one-way, initiated by either side); client notification. Support for request/response and the other 3 transmission primitives from WSDL

Glen : Don't think we should make the axis-user list until 0.9 or beta 1.0...

Issue : Migrating from Apache SOAP to Axis - should be easy, but how easy?

Waqar: We should be tracking ebXML as well as XMLP...

Testing/Process-related discussion...
- coding standards
- test-case centric development
- regression test cases
- scenario-centric design/architecture
- Unit tests are good, but not sufficient
- Is JUnit too oriented towards "white box" unit testing? Jim says they're using it as a framework at HP for both unit and functional.
- Three types of testing - unit, functional integration, interop
- conformance testing; maybe use the userland validator? leverage external efforts? Microsoft "bakeoff"?
- interop testing with big players (.NET) is crucial!
- Requiring some level of testing implementation from commmitters is key. The question is how to best codify this in a strong but not draconian way

The plan for the rest of today:
- Walk through requirements doc
- Generate scenarios
- walk through scenarios, updating requirements

Requirements discussion
- lose 1.1 and 1.1.1: architecture is settling not overdoing use of handlers
- server-initiated solicit/response (in WSDL spec)
- 4 options (synch/asynch, client-initiated vs server-initiated)
    - requestor/provider
    - Facets (Steve):
        - Who initiates (server or client)
        - 1-way/2-way
        - asynch/synch
        - (1-way synch makes no sense, so 7 options)
        - 2-way asynch?

    - Glen: what is underlying protocol doesn't support synchronous (such as smtp)?
    - Waqar: what about async request expecting ack, vs. one not expecting one?
    - Is this (a)synchronous at the API level, or the transport level?
    - Call this topic "message exchange patterns"
    - "Synchronous" == blocking
    - everything is initiated by a client of the axis engine, which can be on requestor, intermediary, or provider
    - Q: in a one-way WSDL message, can you specify a fault? James: yes, but spec doesn't say what to do with the fault. Stuart: thinks treatment of fault is above the message layer.
    - WSDL 1.1 one-way only has input, but doesn't have output or fault (Steve brought up)
    - Stuart: Service in wsdl is more like operation.
- What can we dispatch by?
    - Dispatch by transport URL
    - Dispatch by HTTP SOAPAction
    - Dispatch by QName of first body entry
    - Dispatch, using custom handler, by any information available)
- How many RPC's per message, and where
    - Axis will NOT support a RPC invocation specified in a header
    - Axis will support only one RPC invocation per body entry.
    - Suggestion to remove "first" from body child, 6-2 against.
- XML Prototocol - Status
    - Something will be published in April issues with SOAP 1.1 that need to be fixed
    - July-ish: first XML protocol spec soap 1.1 with fixes to some issues
    - November: proposed recommendation
    - December: 1.0 spec recommendation
- Intermediaries: Glen: TRLSoap has implemented them to some extent.
    - Steve: state of the art is Single-hop fake.
- SOAP Encoding Support
    - We use for RPC, and yet won't require xsi:type (type info can come from a reference to a schema outside the message)
- Performance
    - Action Item: need to research alternative benchmarks for SOAP performance (Jim volunteered) and for interoperability.
        - UI
    - Glen was willing to prioritize "not significantly slower" as want but not required. Jim advocated that we need at release, and Steve agreed.

Discussion of metrics - some arguments as to whether it's a Handler or an API. Well, logging is an API, maybe metrics are as well...

ACTION ITEM: Glen will investigate JRun's metrics API and share with the group.

We need to have some kind of "Required header" parameter either in the WSDD or as an API in the Handler itself if we want "pre-flight" knowledge of required headers (and perhaps a default fault response)...

Big discussion about header handling - jukebox handler, there were worries that it's a security risk. Glen notes that the handlers that are in the jukebox's registry are limited, just like the handlers on a chain, you just don't need to call them all in order...doesn't seem like a risk.

Day 2 1 March 2001 Attendees:

- Glen
- Steve
- Doug
- Jacek
- Jim

Agenda (wish list):

- Finish revision of requirements
- Develop rough use case scenarios
- Revisit requirements after use cases
- Glen: supplier pattern
- Steve: WSDD
- All: streaming parser

Code on BOTH client and Server looks like this?

```
Handler handler = Axis.getHandler(name1);
MessageContext mc = new MessageContext();
// set up MC properties, including target!
handler.invoke(mc);
```

Requirements review, continued

- Service Description
  - Interaction between WSDL and WSDD. Each can be used to create the other?
  - Steve: hard to generate wsdl for document (rather than rpc).
  - Jacek: treat the WSDD as the master
  - Jim: no, WSDL is the standard, WSDD is just our team's proposal.
  - Glen: WSDD may not contain all the info needed to generate a WSDL file
  - WSDD can contain either (1) pointer to existing WSDL file or (2) enough data (possibly, but not certainly) to generate a WSDL from scratch, or (3) enough data which in combination with existing WSDL, to generate an updated WSDL.
  - WSDD should provide cross-checking (WSDL not consistent with WSDD)
  - (I missed some issues here: there's flexibility here that makes this difficult e.g. what serializer?)

    - DISCO on Axis root added.
    - Client-side
        - Default will be simple case: generate a proxy
        - But WSDL can require a digital signature header want to be able to have WSDD on client side to deploy a chain that includes a digital signature generator.
        - Doug: where to get digital signature handler?
            - Steve/Glen: wsdl can include pointer to handler.
            - Glen: there are extension elements in WSDL.
        - Added as wish proxy generation from wsdl (and wsdd). Stated: java, possibly c++
- Platform
    - Surprising amount of interest in doing C++ implementation (Glen, Steve, Jacek) for performance, but was listed as wish [P5]. Background: James Snell added this, including using local services (e.g. COM).
- Transports
    - SMTP sender and POP3 poller. Assume external mail server available. This is what Apache SOAP 2 does.
    - Glen: do we want to provide a generic (fast) transport between handlers?
        - Transport Sender
        - Glen: all transport-specific chains end in a router. Oops, maybe not; if no router at end, look at target service field and if not null, invoke that service; then look again to see if target service is not null. Loop until null.
        - Order
            - Transport chain no router
            - Global chain no router
    - (Lots of discussion on axis engine and dispatching)
    - Jim asked whether any of these transport cases could be punted these asymmetric cases of HTTP request asking for SMTP response. Glen, Steve, Jacek all said no, we need to support all of these.
    - (Discussion over how to send asymmetric response to request sent over one-way protocol: callback, or configure in a Transport Sender and if return message is non-null, send and then null out return message so it. Callback or null-out message? Steve and Jacek: putting up to committer vote would be a big morass? Jacek: ok, grumble, fallback.
    - Doug: are we going to support multi-cast? Jacek: why not? Jim: why?
    - Things that Transport Listener must call and set:
        - Static handler Axis.getHandler(String tlID);
        - Request chain bag ID
        - Response chain bag ID
        - Target bad ID
        - Transport Sender

    - Jacek prefers many small registries over fewer larger ones. Glen said that smaller hash tables perform lookups faster.
        - Services (deployed handlers)
        - Handlers (available handlers)
        - (After much discussion, settled on 2)
- Data Encoding
    - Steve: (de)serialization only makes sense for RPC, not document handling.
    - Jacek would love to see general serialization at 1.0 release.
    - Serialization may occur anywhere, not in one fixed place in Axis architecture.

Streaming Parser Discussion

For reading in input, we use the PDOM model -
We have either a SAX event source or a pull parser underneath something that looks like JDOM.
In the SAX case:

1. a thread starts processing the SAX event stream, and blocks until told to continue.
2. When people ask for a particular element (i.e. getNextChild()), the parser thread unblocks, and runs the SAX event stream through, building up the JDOM object model, until it gets to the desired element/attribute/etc. Then it blocks again, and returns control to the getWhatever() API.

Sot the Message API gets: (or maybe JDOM gets the latter three)

```
getAsElement()
parseAsSAX(SAXHandler)
getAsXMLStream()
getAsByteStream()
```

Does getAsByteStream() terminate the option of doing anything else afterwards?

How do we test this? Metrics:

1. Use PDOM, read first few elements, then stream rest to file
2. Use JDOM, process first few elements, then write to file
3. Stream bytes from file to file

do this for small (1K), medium (50K), and large (10 meg) files

On the output side, we should be able to setAsStream, and hand the element a stream which will eventually be pulled for content when someone does a getAsStream() on the whole doc. This means a proxy stream which clunks its way down the message, draining streams it encounters along the way.

Open question - is it easy/efficient to get a JDOM Element as a stream/string, as opposed to a whole doc? Looks like Element.getSerializedForm() isn't implemented yet?

Pull parser discussion - if possible, a real pull parser would avoid the multi-threading, and

probably perform better than the SAX-based PDOM model. We'll look into this!

Project Participants:

- PM: Waqar (Vitria)
- Coders: Doug, Steve, Glen, Jacek, Jim, James, Yuichi, Ryoh
  - Glen and Doug can work on this a lot
  - James doing stuff
  - Steve doesn't have much time for coding
- Others: George, Chris Nelson, Kevin Mitchell, Sanjiva, Matt,
- Sam for political and process stuff.
- Connection to XML Protocol: Glen, Waqar, Jacek
  - Stuart for Conversations
- Connection to Apache 2.x: Doug, Glen, Sanjiva
- (Glen: SOAP 2.1 documentation is bad, could impact us less credibility).