# GSoC 2011 - Fedora Web Services Update


- Objective
  - Personnel
- Key Decisions
- Work Plan
- WS Stack Comparison
  - Comparison
  - Criteria
  - Community Support
  - Proposed WS Stack
- References


## Objective

Fedora has used Axis 1.3 for SOAP support for quite some time. This is an obsolete library and better alternatives (with support for MTOM and for protocols in addition to HTTP) are available. This improvement will update Fedora to use a modern Java WS framework for SOAP and move away from Axis 1.3. In addition, it would update Fedora's SOAP API to use MTOM where appropriate.

When finished, it should be possible to integrate this work into a future feature release of Fedora, without requiring a major release. That means that it should not change Fedora's SOAP interface as perceived by outside applications.

### Personnel

Our GSoC student is Jiri "George" Kremser, a doctoral student at Masaryk University in the Czech Republic. He'll be mentored this summer by A. Soroka and Chris Wilper. He'll be participating in Fedora committers' conference calls and reporting on progress there, as well as on this wiki page and on the fedora-commons-developers mailing list.

## Key Decisions

So far, the following parameters/scoping decisions have been made for the work:

- Scope: SOAP API only, but consider how the library choice affects the REST API.
- Contract-first vs Code-first: Contract-first. This is compatible with our desire to keep the work relegated to the SOAP side, and wire-compatible with previous versions of Fedora. To produce MTOM functionality, Jiri will extend the current WSDL, but only in ways that do not break current clients. Naming conventions for new methods and new data types will be reviewed by the committers.
- The library in use will be Apache CXF, configured using the new Spring functionality available in post 3.4 series code.

## Work Plan

May 23 - 31: Analysis and framework recommendation

Jiri will report out his findings in this wiki page and in an email to the fedora-dev list before 31 May. He will compare the major contenders for the replacement library (which right now include Metro, Apache CXF Apache Axis 2, and JBoss WS) using several criteria including but not limited to: community support, tooling support (presence in Maven Central), ease-of-integration, MTOM support, inter al. After some period for comment, a decision will be taken and Jiri can proceed. *This work is complete. See above in Key Decisions for the results.*

***The following sections are now historical, because the decision to use Apache CXF (as per Jiri's recommendation) has been finalized.***

## WS Stack Comparison

I've picked following web service stacks as candidates:

- Axis2
- CXF
- JBossWS
- Metro (~ JAW-WS RI + WSIT/Tango + JAXB)

All the frameworks support JAX-WS specification so that it is easier to migrate from one to another. There exist two frameworks which are not mentioned in the following text, as they do not support JAX-WS standard and the community behind those product is not as big as behind the four frameworks mentioned above. The two frameworks:

- Spring Web Services
- Turmeric

### Comparison

In the table below, by "JBoss WS" I mean their native implementation, as they support CXF and Metro as well.

## Criteria

1. customization of generating the server-side code from WSDL
2. modularity of jar files
3. presence in Maven Central
4. Spring integration
5. wsdl2java (server-side skeletons JAX-WS annotations)
6. performance (the higher, the better)
7. MTOM
8. JAXB
9. JAX-WS
10. JAX-RS

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Axis2 | 0 | - | 1 | 0.5 | 1 | 1 | 1 | 1 | 1 | 1 |
| CXF | 1 | 1 | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 1 |
| Metro | 1 | 0 | 1 | 0.75 | 1 | 2 | 1 | 1 | 1 | 1 |
| JBoss WS | 0 | - | 0 | 0 | 0* | - | 1 | 1 | 1 | 0 |

*only java2wsdl

The performance comparison is taken from http://www.ibm.com/developerworks/webservices/library/j-jws14/index.html where large responses are taken into account.
The "Spring integration" comparison is based on the tutorials posted on here (Axis2), here (CXF) and here (Metro), and on the sample applications attached to the a WS stack. For instance, the sample application distributed with CXF for wsdl-first approach is well documented, there is a support for both Maven and Ant, the Spring integration is also done and the example of using MTOM is also present. Samples from Axis2 sometimes support Maven, sometimes Ant, there is an MTOM example, however, the Spring integration and the wsdl-first examples are missing. All sample applications from the Metro stack use Ant for building, there is an example of the wsdl-first approach and MTOM, but Spring integration showcase is missing.

## Community Support

It is hard to measure this aspect, so I put here links to the issue trackers or, where it is present, the mailing lists with commits. I forgot to mention that CXF and Axis2 are both supported by Apache Foundations and Metro by Oracle. If fcrepo is going to migrate to Java EE 6 (with Context Dependency Injection, JAX-RS, etc.) in the future, Metro or JBoss WS stacks are better prepared to this migration, in my opinion.

- CXF = http://mail-archives.apache.org/mod_mbox/cxf-commits/ (https://issues.apache.org/jira/browse/CXF/)
- Metro = http://java.net/projects/jax-ws/lists/cvs/archive (http://java.net/jira/browse/JAX_WS), part of Java EE 6 (Oracle)
- Axis2 = https://issues.apache.org/jira/browse/AXIS2
- JBoss WS = https://issues.jboss.org/browse/JBWS
- Spring WS = https://jira.springsource.org/browse/SWS

## Proposed WS Stack

I propose to use the CXF and as a binding technology the JAXB. When building the project from WSDLs (Fedora-API-A.wsdl and Fedora-API-M.wsdl) and from fedora-types.xsd, the server-side skeletons annotated with JAX-WS annotations and the datatype POJOs will be generated by wsimport tool, i.e. contract-first approach. Let's discuss this proposal. All the frameworks which support JAX-WS provide almost the same code, the main difference is in the infrastructure in behind, for instance deployment of WS, integration with Maven, etc. I am open to any other proposal.

# References


- http://axis.apache.org/axis2/java/core/
- http://cxf.apache.org/
- http://metro.java.net/
- http://www.jboss.org/jbossws
- http://www.ibm.com/developerworks/views/webservices/libraryview.jsp?search_by=java+web+services,%20sosnoski
- http://wiki.apache.org/ws/StackComparison
- http://www.ibm.com/developerworks/webservices/library/j-jws14/index.html
- http://www.googlefight.com/index.php?lang=en_GB&word1=cxf&word2=axis2