

Case Study: Advanced Java™ SOA Solution for European TelCo

The Company. Our Client is a global carrier committed to the IP and Ethernet wholesale market. It provides global IP Transit and Ethernet connectivity to Carriers, Service and Content Providers worldwide and is one of the largest vendors in Europe.

As a pan-European company with a strong international presence, the company was looking for a near-shore partner that could provide not only excellent technical skills but also proximity both in terms of time-zone but also culturally. Belatrix was chosen after a thorough process that included visiting prospect vendors in India and Eastern Europe before visiting and finally selecting Belatrix.

The Challenge. As a spin-off of one of the most successful internet providers in Europe, the Client inherited a number of legacy systems that were not fully integrated and that did not use consistent technologies, thus becoming extremely resource-intensive and costly to maintain and improve. Another issue was that because of all the different systems there was information fragmentation as well as different versions of the truth so it was almost impossible to obtain a coherent view of a Customer, their Orders and their Services.

Java™ technologies used

- Linux Servers,
- GIT for version and source code control,
- Maven,
- Java (JDK6),
- Jersey JAX-RS,
- Tomcat,
- MySQL,
- XML, XML Schema, Schematron, JSON, JSON Schema,
- MongoDB,
- Log4J,
- JavaMail,
- FreeMarker,
- Hudson CI server, Sonar QM tool,
- Fitnesse.
- For Development: NetBeans IDE, on Linux desktops, because its good Jersey support and previous developers' experience; JSONView Firefox addon, POSTER Firefox addon,

Because of these challenges our customer decided to engage Belatrix and leverage Belatrix's Agile Nearshore Outsourcing Services to re-design its internal applications using a state-of-the-art Service Oriented Architecture (SOA) based on Web Services. These Web Services would provide a centralized and consistent view of Customers, Services and Orders that other applications

would leverage thus also eliminating the closed coupling that existed before between the different systems that made upgrades to any component or interface a very big challenge

The Solution. Our team was asked to find the best way to implement these Customer, Order and Service central SOA repositories using Open Source technologies while keeping the solution and architecture as simple as possible both from a systems architecture and deployment perspective as well as from the end-user experience.

With these constraints in mind, our team assessed several technologies and decided to implement **SOA RESTful Web Services**. Adaptability, low overhead and existing applications were the main reasons for this choice.

The Technology Stack. Once RESTful Web Services were adopted, we needed to decide which implementation to use and we ended up selecting **Jersey**, Sun's reference implementation of JSR 311 (JAX-RS: The Java™ API for RESTful Web Services, more information can be found on <http://jcp.org/en/jsr/detail?id=311> and <https://jersey.dev.java.net>).

Data persistence was, at first, implemented as a **MySql** database even though Belatrix's recommendation was to use **PostgreSQL**, but the customer preferred **MySql** due to its own installed base. Later on, as we progressed through the iterative agile development process, the Client requested a more flexible data model to be able to handle frequent model changes so we agreed to implement a sort of **XML database on MySql** while exploring other alternatives. Even though this innovative approach had many advantages, it proved to have query limitations. Finally, after careful research, it was decided to adopt **MongoDB** for persistence. This decision has proven to be a wise one as it has achieved all the goals related to model flexibility while avoiding the limitations seen with XML-based storage on MySQL.

When using data models that do not validate data formats, data must be validated somewhere else in the system's stack. That was the case when persisting data as XML fields: we implemented validation using **XML Schemas** and **Schematron**, two interesting technologies but still not as convenient as what MongoDB offers. With MongoDB, a similar approach is being used to validate JSON data with **JSON Schema**. We chose an open source implementation of **JSON Validator** and Belatrix's Java™ Team is in close contact with the main developer to get support, provide feedback and contributions.

As part of system auditing features, we implemented a Notification Service to send emails to selected recipients

when some events are triggered. It is basically a lightweight, basic Workflow Engine. This service is configured easily through XML files. Mail templates are based on well known **FreeMarker** template library.

Logging is done through **Log4J** library, almost a de-facto standard for Java™ solutions.

The solution also implements **User Rights Management** by accessing the Client's **LDAP** server and combining that information with local rights.

Main Technologies and Frameworks used:

- Linux Servers,
- GIT for version and source code control,
- Maven,
- Java (JDK6),
- Jersey JAX-RS,
- Tomcat,
- MySQL,
- XML, XML Schema, Schematron, JSON, JSON Schema,
- MongoDB,
- Log4J,
- JavaMail,
- FreeMarker,
- Hudson CI server, Sonar QM tool,
- Fitnesse.
- For Development: NetBeans IDE, on Linux desktops, because its good Jersey support and previous developers' experience; JSONView Firefox addon, POSTER Firefox addon,

Project Management and Methodology. Belatrix is an expert in Agile Software Development using Scrum so for this project we leveraged Agile-specific support tools such as Pivotal Tracker to maintain user stories, calculate the

team's velocity, maintain the backlog and to provide seamless visibility to the Client's stakeholders and the Client's Product Owner.

The Customer's Architect, acting as the Product Owner was the person in charge of writing the user stories that the team would analyze, refine, estimate and implement. Iterations were very short, usually lasting two weeks, following the rule of "release early, release often".

Since the solution had almost no UI, testing the Web Services was another interesting challenge as we had to decide on the right processes and tools to test them. Developers' manual unit tests were done using Firefox add-ons: **POSTER** and **JSONView**. Both are excellent tools and they really help when testing Web Services. The project's Quality Assurance Engineers conducted their testing using **POSTER**, some with **Jersey Test Framework** and, recently, with the aid of the **Fitnessse** tool. Our goal was always to automate testing as much as possible and run all tests on each build. No user story was ever delivered to the client for approval without full QA approval and sign-off.

The source code was maintained in several **GIT** repositories: one for development, other for testing and a production one, too.

The building tool was **Maven**. We implemented a continuous build and integration framework linking **GIT**, **Hudson** and **Tomcat**. After each commit, GIT would launch a Hudson job that built the project and, if successful, deployed it to Tomcat. So, builds and integration were done after each commit, thus ensuring the source code stayed operational and as bug-free as possible at all times.

The Results. The Agile Software Development Methodology has proven to be an extremely fit approach for a Telecommunications Company as their business is in constant flux and priorities and requirements change often so it is important that the development process accommodates these changes effectively. The Client and Belatrix are both excited that even though the technology stack chosen for the project was leading edge that it has proven to be both reliable and maintainable.

We invite you to learn how a relationship with Belatrix will give your company a distinctive advantage through low cost, disciplined, and high quality software development and quality assurance services.

You may contact us at:

businessdevelopment@belatrixsf.com

Phone number: +1 (617) 608 – 1413 x2001