

Swordfish 0.10 (Helios) Release Review



Planned Review Date: 06-11-2010

Communication Channel: [eclipse.swordfish](#)

Zsolt Beothy-Elo (Project lead)

Introduction

- Swordfish provides an extensible runtime framework aimed at creating service-oriented applications
- Swordfish is internally based on Apache ServiceMix 4 as the core messaging engine
- Swordfish hooks into ServiceMix and adds functionality that is required for enterprise environments, such as service registry integration, remote configuration and monitoring
- Swordfish includes basic tool support and additional components such as a Service Registry

Features

- General interceptor framework that hooks into the underlying messaging engine (Apache ServiceMix NMR)
- APIs and exemplary plug-ins based on the general framework for specific areas that are significant for enterprise usage:
 - Dynamic Service Resolution: Resolve logical service endpoints into physically addressable endpoints by querying a service registry at runtime
 - Monitoring: Generate monitoring events that allow for detailed tracking of how messages are processed and that can be stored for later analysis or reporting.
 - Remote Configuration: Configure framework via a local Configuration Agent that can retrieve configurations from a remote server and uses the OSGi Configuration Admin service to provide them to the framework
- Basic tools supporting the most important use cases
- Service Registry to dynamically resolve logical service names into service endpoint addresses

New in 0.10

- Remote Deployment of Swordfish participants. This includes
 - on the runtime site a swordfish provisioning manager
 - on the tooling site remote deployer based on WTP server tools.
- A swordfish server based on Equinox including p2, Swordfish runtime and provisioning manager.

Features – not accomplished yet

- Some features that were originally planned for this release have not been accomplished for various reasons
 - Enhanced Policy Processing
 - Service Activity Monitoring
 - JMS Transport

Non-Code Aspects

- 45 min “Getting Started” video available
- Swordfish is now included in the SOA package
- Unit test coverage has improved since last release (from 50% to 60%)
- JUnit-based integration tests further extended
- Tool components are well documented (Eclipse Help, Cheat sheets)
- Added documentation how to setup a Swordfish runtime
- Javadoc for the framework APIs

APIs

- All APIs are still provisional and are expected to further evolve based on community feedback, stabilization is planned for 1.0 release
- Extensibility and customizability is one of the key aspects of Swordfish
 - General interceptor API (creation of custom interceptors, custom processing planners etc.)
 - Service Resolver API (integration of custom service registries, custom service description document types etc.)
 - Configuration API (integration of custom configuration backends)
 - Monitoring API (integration of custom event types, event sources and event receivers)
 - based on open standards (such as JBI)

Architectural Issues

- Swordfish makes extensive use of OSGi services in order to reduce coupling of internal components and allow for extensibility and customizability
- Plug-ins are registered using the OSGi Whiteboard pattern
 - Plug-in developers are free in their technology choice: OSGi DS, Spring DM, SAT, manual service registration
- All internal OSGi services are registered with a low service rank and can be replaced by custom implementations
- Swordfish core is based on Spring DM as the dependency injection framework
 - robust tracking of service registrations/de-registrations → fully dynamic behaviour, no re-starts required
 - components easily replaced by Mocks for testing purposes

Architectural Issues (cont'd)

- Swordfish tooling is using internal API's from the following projects: PDE, WST, and P2. After Helios we will work together with the mentioned projects to remove the usage of the internal API's

Tool Usability

- Swordfish provides basic tooling to support the most relevant service development use cases:
 - Implement service consumers and providers based on a WSDL document (WSDL-first approach)
 - Implement services based on a Java interface (Code-first approach)
 - Publish WSDL documents into the service registry
- We are working closely with various sub-projects and components of STP in order to make service creation and deployment into Swordfish even more smooth and seamless

End-of-Life

- No features from the previous release have been end-of-life'd in Release 0.10

Bugzilla

		Status					
		NEW	ASSIGNED	REOPENED	RESOLVED	CLOSED	Total
Severity	blocker	.	.	.	<u>6</u>	.	<u>6</u>
	critical	.	.	.	<u>1</u>	.	<u>1</u>
	major	<u>1</u>	.	.	<u>10</u>	.	<u>11</u>
	normal	<u>3</u>	<u>1</u>	<u>2</u>	<u>67</u>	<u>5</u>	<u>78</u>
	minor	.	<u>1</u>	.	.	.	<u>1</u>
	trivial	<u>4</u>	<u>4</u>
	enhancement	<u>3</u>	.	.	<u>4</u>	<u>1</u>	<u>8</u>
	Total	<u>11</u>	<u>2</u>	<u>2</u>	<u>88</u>	<u>6</u>	<u>109</u>

Bar | Line | Table | CSV

Standards

- Parts of Swordfish's API make use of concepts from JBI (JSR 208) instead of re-inventing the wheel
- All standards relevant in the SOA space are supported through third-party components, e.g. WSDL, SOAP, WS-Security, WS-Addressing,...
- JAX-WS can be used to implement services and is specifically supported by the Swordfish tools

UI Usability

- The Swordfish tool components expose only a very limited UI which adheres to the Eclipse User Interface Guidelines. Most UI elements are actually reused from the Workbench.

Schedule

- 0.9.1 GA: planned for Galileo SR1, delivered in time
- 0.9.2 GA: planned for Galileo SR2, delivered in time
- 0.10 M3 - M7: planned for the corresponding Helios milestones, delivered in time
- 0.10 RC1: planned for 05/19/2010, delivered in time
- 0.10 RC2: planned for 05/26/2010
- 0.10 RC3: planned for 06/02/2010
- 0.10 RC4: planned for 06/09/2010
- 0.10 GA: planned for 06/16/2010

Communities

- Swordfish currently has 7 committers and 2 contributors
 - 6 committers from SOPERA, 1 from Progress Software (former IONA)
- Open and public development process
 - following the Scrum methodology
 - planning and progress tracking done in the open in a public group chat (currently on Skype)
 - Incorporating feedback from forum and mailing list
- Actively Evangelizing
 - Talks on Eclipse Summit Europe 2009, EclipseCon 2010 and various other conferences, e.g. Java User Group Stuttgart, Open Source Expo 09.
- Close interaction with the Apache Community
 - working with the ServiceMix 4 and especially the CXF project to resolve integration issues

IP Log

- All applicable IP policies and procedures as defined by the Eclipse Foundation have been followed.
- For all third-party libraries in use, the corresponding CQs have been approved.
- All source code was either 100% written by one of the committers or by a contributor who works for the same member organization as the committer.
- Swordfish's IP log can be found at http://www.eclipse.org/projects/ip_log.php?projectid=soa.swordfish
- A frozen copy of the reviewed-and-approved-by-Eclipse-legal IP log has been supplied as part of the Release Review documentation

Other

- Swordfish originally planned to have its graduation review before the Helios release and to deliver a 1.0 for Helios. Because only recently several projects started to use and extend Swordfish we postponed graduation to be able to solicit feedback from the projects regarding the API's.

Communication Channel for Feedback

Please provide feedback in the Swordfish forum

[http://www.eclipse.org/newsportal/thread.php?
group=eclipse.swordfish](http://www.eclipse.org/newsportal/thread.php?group=eclipse.swordfish)