Aspect Weaving for OSGi

Martin Lippert (akquinet it-agile GmbH)
Aspect-oriented programming

- Modularity improved a lot by OO concepts
  - Classes, interfaces
  - Information hiding, polymorphism, inheritance
- AOP adds additional concepts
  - To modularize so called “cross-cutting concerns”
AOP today

• Meanwhile AOP is an established concept
  ▪ Useful for many situations
  ▪ Mostly technology-centric usage scenarios
• Established languages and frameworks available
  ▪ AspectJ: powerful language extension to Java
  ▪ Spring-AOP: simple to use AOP for enterprise apps
• Used in production:
  ▪ Spring itself makes heavy use of AOP concepts
  ▪ App-servers are using AOP inside
  ▪ Direct AOP selectively used in enterprise apps
AspectJ = AOP for Java

• AspectJ is a powerful language extension for Java
  ◦ Hosted as an Eclipse project
  ◦ Still very active (latest release 1.6.1 in July 2008)

• AJDT:
  ◦ Great tooling for the Eclipse IDE (3.3, 3.4)
  ◦ Comes close to the JDT feeling

• Spring-IDE:
  ◦ Integrates AJDT with Spring-AOP
  ◦ AJDT feeling for Spring apps
The Standard Use Case

Diagram showing relationships between classes and aspects within a project source, single application classpath, and Java virtual machine.
Library Aspects

![Diagram showing library aspects and their interactions with class A, class B, class C, and aspect X.](image-url)
Aspects for Existing Code

![Diagram showing JARs, project sources, single application classpath, and Java virtual machine with class definitions and concerns.]

Aspect Weaving for OSGi | © 2008 by Martin Lippert; made available under Creative Commons Att. Nc Nd 2.5 license
Java + OSGi

- OSGi:
  - "A dynamic module system for Java"

- Modularity
- Dynamic
- Service-Oriented
What does it mean for us?

- We would like to **modularize**
  - … classes and interfaces into bundles
  - … *and* aspects into bundles

- The obvious next step:
  - **modularize cross-cutting concerns into bundles**

- Takes modularity to the next level
Intra-Bundle Aspects

Bundle A

ClassA

ClassB

Bundle-Classpath

Bundle A

ConcernX

Bundle B

ClassC

AspectX

ConcernX

Bundle-Classpath

Java Virtual Machine
Co-Op Bundle Aspects
Abstract Aspect Bundles

Java Virtual Machine

Bundle-Classpath

ClassA
ConcernX

ClassB
ConcernX

Concrete Aspect
ConcernX

Bundle A

Abstract Aspect
ConcernX

Bundle B
Dynamics for Aspect Bundles

• OSGi allows dynamic bundle
  • … installs
  • … uninstalls
  • … updates

• Same should be possible for aspect bundles
  • … dynamic installs, uninstalls and updates of aspect bundles
  • … dynamic installs, uninstalls and updates of bundles that are affected by aspects
How could all this possibly work?
Equinox Aspects

• Equinox Incubator Project
  • [http://www.eclipse.org/equinox/incubator/aspects](http://www.eclipse.org/equinox/incubator/aspects)

• Enables AspectJ/AOP for OSGi
  • Supports all presented use-cases
  • Ready-to-use

• Setting
  • Works with Eclipse 3.4 (and 3.3 deprecated)
  • Works with AJDT 1.5.2, 1.5.3, 1.6.0, 1.6.1
What can I do?

• Put aspects into standard OSGi bundles
  • Just like Java classes
• Define what and where to weave
  • aop.xml and manifest headers
• Go!

• Feels like a natural combination of AOP and OSGi…
Load-Time Weaving for OSGi

• Let the OSGi runtime take care of weaving the aspects
  - (and not the compiler)
  - Leads to load-time weaving within OSGi

• This means:
  - No recompilation of existing bundles necessary
  - Supports “aop.xml” load-time weaving config of AspectJ
Live Demo

• Monitoring Eclipse bundles…
Caching

• Wasn’t that a fast startup?

• The reason: caching for woven classes
  • Load-time weaving happens only once
  • Second time startup is same as without aspects
  • Available for standard JREs and IBM J9 shared classes
  • Supports configuration switching
Dynamics

• Dynamics for aspect bundles
  - Means re- or un-weaving existing bundles

• How is it realized?
  - Silent update of bundles to be woven again
  - Bundles must behave nicely within dynamic situations
Live Demo

• Installing, updating, uninstalling aspects at runtime…
AOP in Spring

• Spring uses AOP a lot for all kinds of purposes
• @Configurable is one example

• Realized by Spring via load-time aspect weaving
Spring Dynamic Modules & Equinox Aspects

• Equinox Aspects can do load-time aspect weaving for Spring Dynamic Modules…

• Live Demo
  ✷ @Configurable for Extensions (Views in Eclipse RCP apps)
Equinox Aspects in Production Systems

• Allianz Business System (ABS)
  - Mission critical system at the core of the insurance business.
  - Addresses all major concerns in the various classes of insurance.
  - Online, offline and integration scenarios.

• Uses Equinox Aspects for highly specific performance monitoring (QoS) in production
APIs and Implementation

- **org.eclipse.equinox.weaving.hook**
  - Hooks into the runtime
  - Provides API for injecting weaving and caching implementations

- **org.eclipse.equinox.weaving.aspectj**
  - Implements aspect weaving using AspectJ

- **org.eclipse.equinox.weaving.caching**
  - Implements caching for standard VMs

- **org.eclipse.equinox.weaving.caching.j9**
  - Implements caching for IBM J9 VMs (shared classes feature)
Conclusions

• Equinox Aspects brings full AOP to OSGi
  - Load-time weaving integrated into OSGi
  - Combines OSGi and AOP modularity features

• Can be used for production systems today

• Give it a try
  http://www.eclipse.org/equinox/incubator/aspects
Thank you for your attention!

Q&A

Martin Lippert: lippert@acm.org