Aspect Weaving for OSGi

Martin Lippert (akquinet it-agile GmbH)
Heiko Seeberger (Weigle Wilczek 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

Project Sources

Single Application Classpath

Java Virtual Machine

ClassA

ConcernX

ClassB

ConcernX

ClassC

ConcernX

AspectX

ConcernX
Library Aspects
Aspects for Existing Code

![Diagram showing JARs, Project Sources, Single Application Classpath, and Java Virtual Machine with ConcernX aspects.]
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
Co-Op Bundle Aspects
Abstract Aspect Bundles

Diagram showing the relationship between classes, concerns, and aspects in the context of OSGi bundles.
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 & Equinox Aspects

• Equinox Aspects can do load-time aspect weaving for Spring-powered bundles…

• Live Demo
  • @Configurable for Extensions (Views in Eclipse RCP apps)
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
Thank you for your attention!

Q&A

Heiko Seeberger: seeberger@weiglewilczek.com
Martin Lippert: lippert@acm.org