OSGi, COSi and component verification

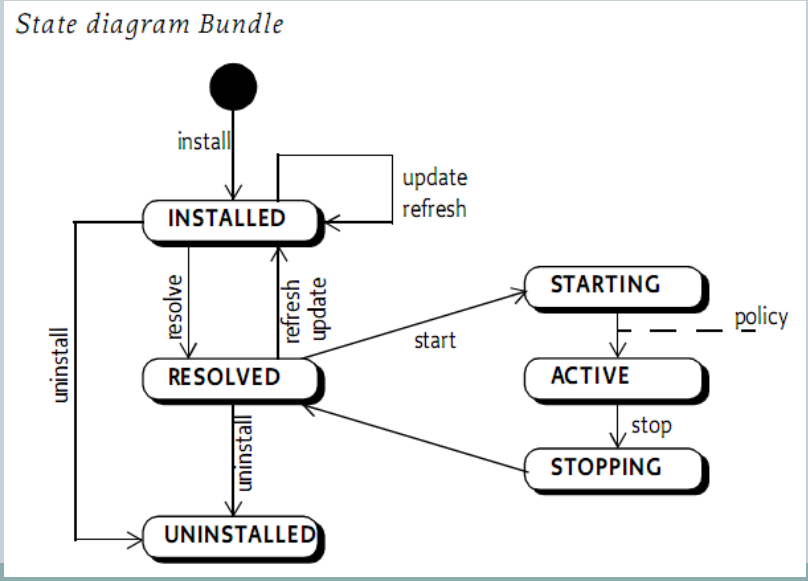
Recap project goals

OSGi

- Goal, consortium, etc.
- Framework
- Bundle
- Services

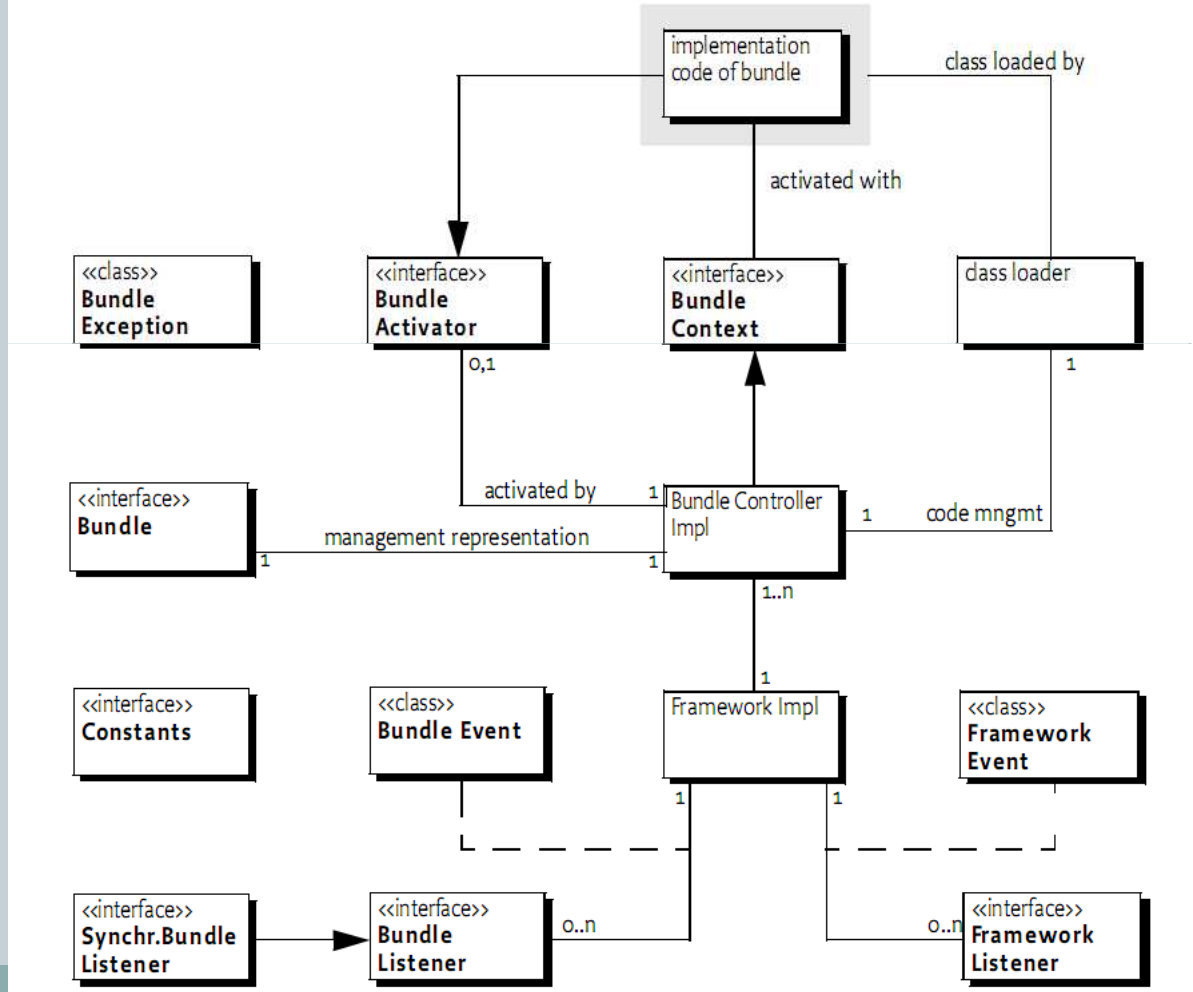
Bundle

- industry standard Java components
- metadata in bundle manifest
  - provided + required packages
  - services
- lifecycle, wiring



Bundle in Framework

Figure 4.25 *Class diagram* org.osgi.framework *Life Cycle Layer*



Discussion: the good, the bad, the ugly

- Dynamic – good but no decl
- Versioning – good but fragile
- Binding checks version numbers only
- Fine granularity for big apps

Extensions to OSGi

- Equinox
- Eclipse plugins, features

Work done so far

- OSGi component model => COSi
- Bundle checks => OBCC, OSGi Bundle Compatibility Checks
- Feature upgrades => EPIC

COSi

- Obsessively pure black-box component model
  - Versions
  - Control class
  - Features: Interfaces, Events, Attributes, Types
- Dynamic typing
- Future: playground for extra-functional attributes

OSGi Bundle Verification

OSGi Bundle Verification: Goals

- Implementation of strict, and potentially contextual, substitutability checks for OSGi
- ENT representation of bundle surface
  - packages -> types
  - “syntactical” analysis, introspection
  - bytecode analysis (tricky)
- Contravariance on surface features
- Lukáš Valenta RIP ;-)

Implementation on OSGi

- Goal: Enhance existing framework with strong update checks
- OBCC
- Jaroslav Bauml

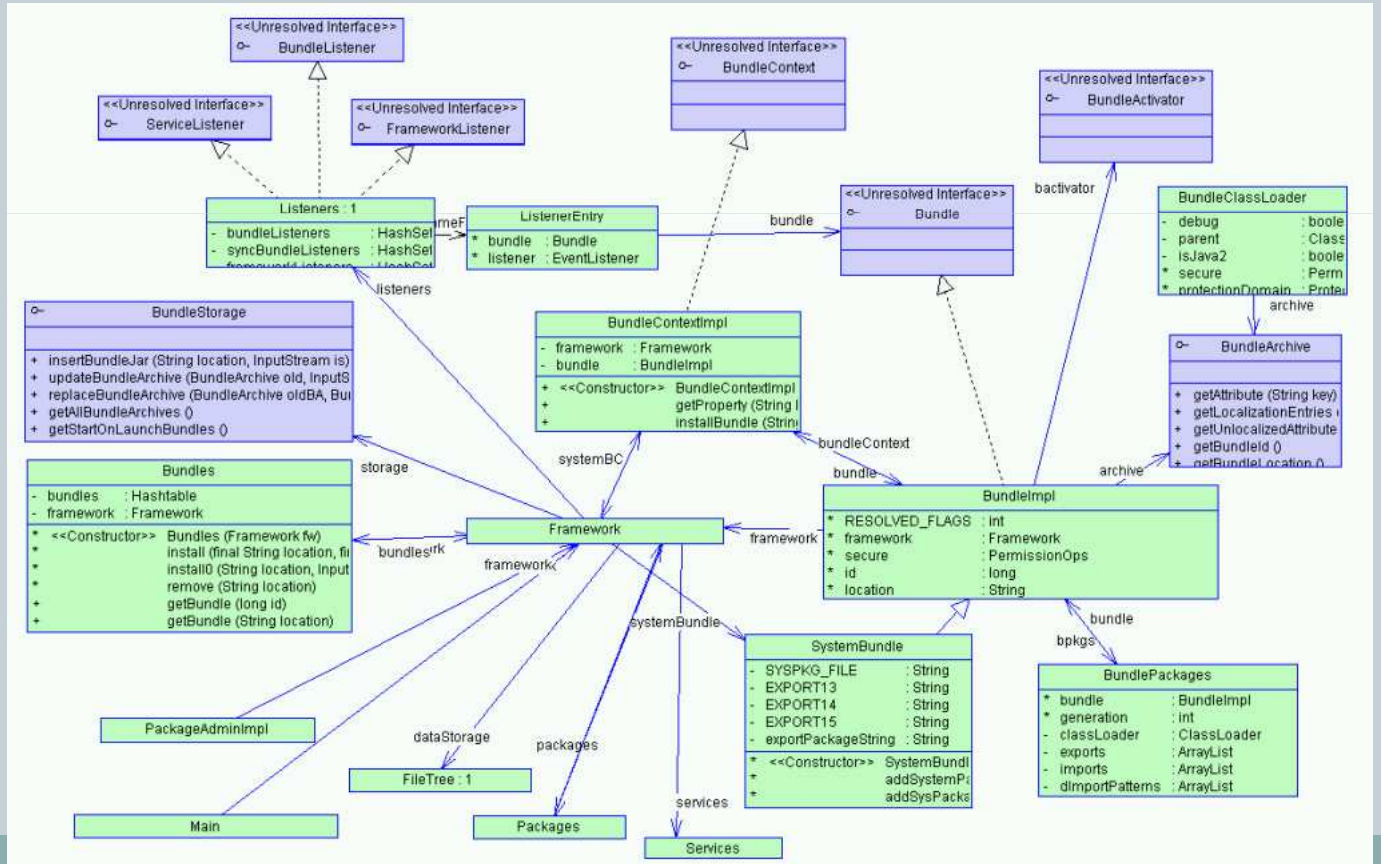
- Currently implemented for Knopflerfish

OSGi Bundle Verification

- Various options
  - Internal
  - User space (needs new event to be broadcasted)
  - Facade methods
- Last one currently pursued
  - Framework extended with update(bundleId) and updateWithCheck(bundleId) methods
  - Called from user-space „update manager“ bundle

KF Implementation

- org.knopflerfish.framework.
  Framework.updateBundle(long pid)



Obrázek 3.3: Výňatek API org.knopflerfish.framework

Lessons learned

- Issues uncovered
  - „update“ event missing + possibility to veto a change (like XxxVetoableListener in JavaBeans)
  - ??? Update process includes new bundle binary download
  - Dependency on internal APIs and implementation classes – mainly bundle (.jar) repository
- Open issues, further work
  - Pursue the Updating event path?
  - Apache Felix implementation (cleaner design)
  - Enhance the comparator (Java 5 features)

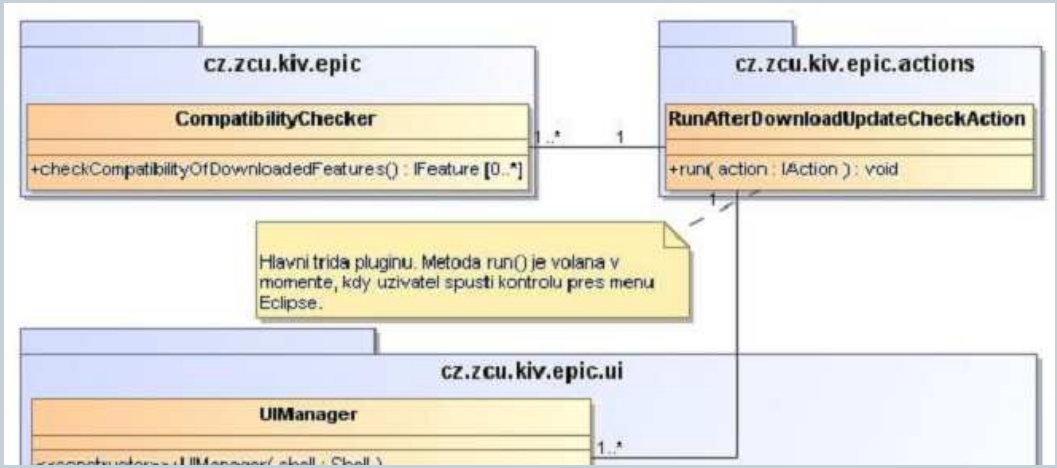
Implementation on Eclipse Plugins

- Use of strict substitutability on Eclipse plugins
- Smooth integration into the IDE

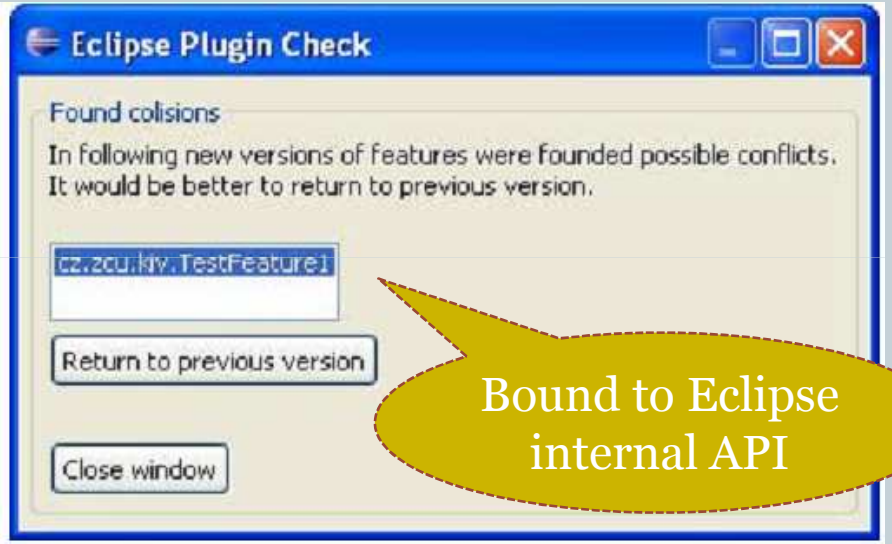
- EPIC
- Pavel Stahl

Eclipse Plugin Checks: How it Works

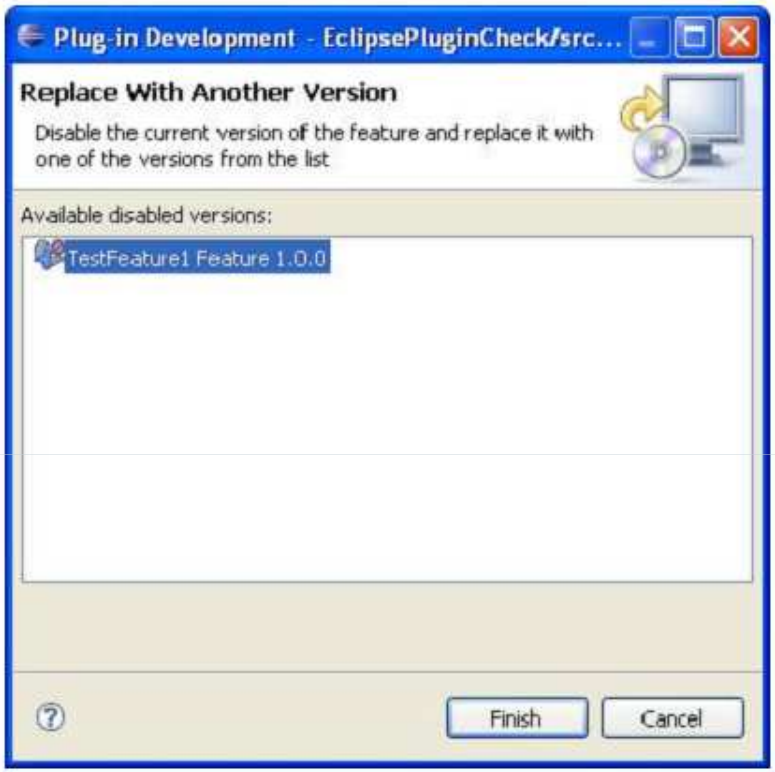
- Finding plugin JARs
  - platform.xml
- Hooking into update process
  - Help > Run Update Checks



Eclipse Plugin/Feature Checks



Obrázek 2 – dialogové okno, kdy byl nalezen konflikt



Obrázek 3 – dialogové okno pro změnu verze feature

Eclipse Plugin Checks: Issues

- Dependencies on Eclipse internals
  - Plugin/feature repository (partially)
  - Installed plugins/features
  - Update process – currently solved by avoidance
- Missing public listeners-API for full plugin lifecycle management
  - Hooks into the process