Tales from the OSGi Trenches
Adobe CQ5: a real-life OSGi case study

Bertrand Delacrétaz
Senior Developer, R&D, Adobe Digital Marketing Systems
Apache Software Foundation Member and (current) Director
http://grep.codeconsult.ch - @bdelacretaz - bdelacretaz@apache.org
OSGi Alliance Community Event, Ludwigsburg, October 2012
slides revision: 2012-10-22
What?

Share our **experience using OSGi** (Apache Felix) as the foundation of Adobe CQ5.

**More than five years** working with OSGi, very **high impact** on developers, customers, service people, *mostly* in a **positive** way.

OSGi is **no silver bullet** either.
What?

the GOOD

the BAD

the UGLY

Just my own, personal opinion as a software developer...

symbols by ppdigital, o0o0xmods0o0oon and clarita, on morguefile.com

Tales from the OSGi trenches

OSGi Alliance Community Event 2012
Bertrand Delacrétaz
Trenches?

Web Content Management and Digital Marketing system. Includes its own JCR content store. About 200 OSGi bundles. Install instructions fit in a tweet.

Built on Apache Sling, Apache Felix and Apache Jackrabbit

www.adobe.com/products/cq.html
http://jackrabbit.apache.org
http://felix.apache.org
http://sling.apache.org
What we use from OSGi

Bundles (using Maven plugins)
Lifecycle, Service Tracker
Configurations and Felix Web Console
Declarative Services (using Maven plugins)
Sling’s installer module (like fileinstall, using JCR)
Log, HTTP, Event services, JMX
Famous quotes
the short version
Famous Quotes, #2

“Each (OSGi) bundle can serve as a micro application, having it’s own lifecycle, having it's own citizens and each bundle can carefully decide which objects to expose to the outside world”

Peter Rietzler
http://peterrietzler.blogspot.com/2008/12/is-osgi-going-to-become-next-ejb-bubble.html
Famous Quotes, #4

“The lifecycle model of OSGi makes life complicated. Actually, tracking services and managing all the aspects of what to do when services come and go is nasty”

Peter Rietzler
http://peterrietzler.blogspot.com/2008/12/is-osgi-going-to-become-next.ejb-bubble.html
Famous Quotes, #6

“OSGi makes "impossible" things easy: hot deploy/upgrade, service discovery, ... and trivial things hard: hibernate, tag libraries, even deploying a simple war!”

But, for the first time in my career, I see software reusability that works: service reusability.

Filippo Diotalevi
The Good
Modularity

Classloading distinct from class visibility.

OSGi bundles as reusable components. Dynamic setup and updates that work. At last!

Matchless picture: Alvimann on morguefile.com
Declarative Services

```java
/** Servlet that implements GET and POST access
 * to our Storage component, mounted on /store
 * by default, path can be changed by configura
 */
@SuppressWarnings("serial")
@Component(metatype=true)
@Service(value=Servlet.class)
@org.apache.felix.scr.annotations.Properties(
    @Property(name=CoreConstants.SERVLET_METHOD_PROP,
              value=["POST", "GET"], propertyPrivate=true),
    @Property(name=CoreConstants.SERVLET_PATH_PROP, value="/store")
)
public class StorageServlet extends HttpServlet {

    @Reference
    Storage storage;
```
Clean OSGi APIs

installBundle

```java
public Bundle installBundle(java.lang.String location,
                           java.io.InputStream input)
    throws BundleException
```

getBundles

```java
public Bundle[] getBundles()
```

Returns a list of all installed bundles.

addServiceListener

```java
public void addServiceListener(ServiceListener listener,
                                java.lang.String filter)
    throws InvalidSyntaxException
```

update

```java
public void update(java.io.InputStream in)
    throws BundleException
```

Updates this bundle from an InputStream.

Just a few basic examples...
Dynamic load/unload

Just copy bundle jar to Sling’s JCR repository (WebDAV)

Bundle activated and started.
(using Sling’s installer module)
Plugins for everything

Servlets

- Content editors based on JCR node properties
- Mime-type based handlers

Debugging/monitoring tools

Content renderers and decorators

Legacy integration gateways

Mail and messaging services

etc, etc...
The Bad
Granularity?

How many bundles? Services?
*CQ5: about 200 bundles, 1000 services*

How to handle “implementation details” libraries.
*Extra bundles or private packages?*

Strict version management required.
*Are we there yet?*
Integration testing...

In-system testing?

but 😊 when done!

Tales from the OSGi trenches
Testing options

**Unit testing**

JUnit, TestNG, Mocks, etc..

Necessary but usually not sufficient. Mocks can be painful

**In-framework testing**

Pax Exam, Sling testing tools, etc.

Relatively new tools, can be slow

**Integration testing**

Start application and test it “from the outside”.
See Apache Sling’s launchpad/testing module

Much slower but ok for actual integration testing
I want my...

Spring  Guice  App Server

J2EE  do you really need it?
The Ugly
Asynchronicity...

“unpredictable” startup order

Is the system ready now?

Mostly from Declarative Services, not OSGi itself.
Unpredictable assemblies

Spot the monsters!
Discipline (and semantic versioning) help.
Integration (and in-system) testing required.
The Summary
My vision back in 2009...

- Developers got used to it (and read the book).
- Frameworks and tools improved.
- Distributed OSGi? Maybe.
- Customers understand OSGi and like it.
- Apache Sling paved the way.
Conclusions

- Modularity. modularity. modularity.
- OSGi fosters better structured code
- Dynamic services and plugins FTW!
- Tooling ... getting better all the time
- OSGi is often unfamiliar / unusual
- Asynchronous behavior can be problematic (declarative services)