OSGi in GlassFish V3

Jerome Dochez - Architect and Lead, Glassfish v3

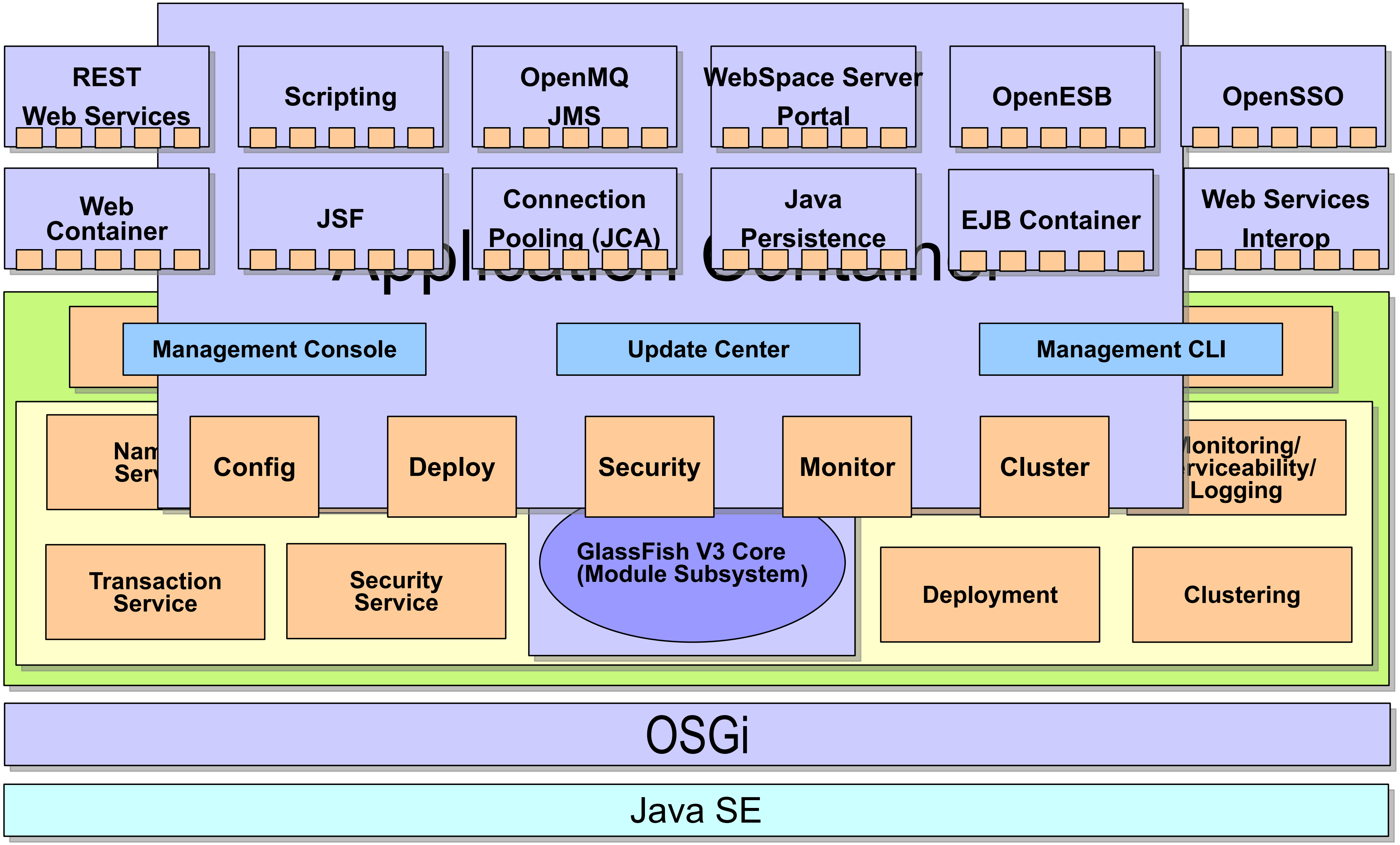
Looking Back

- GlassFish v3 Prelude
  - > Big move to OSGi technology
  - > Big move to a more modular development approach
- Benefits
  - > Demands and enforces stronger modularity
  - > Provides a foundation with well-defined, dynamic module lifecyle management
- However, OSGi is largely under the covers
  - > Visible to GlassFish developers, but not GlassFish users

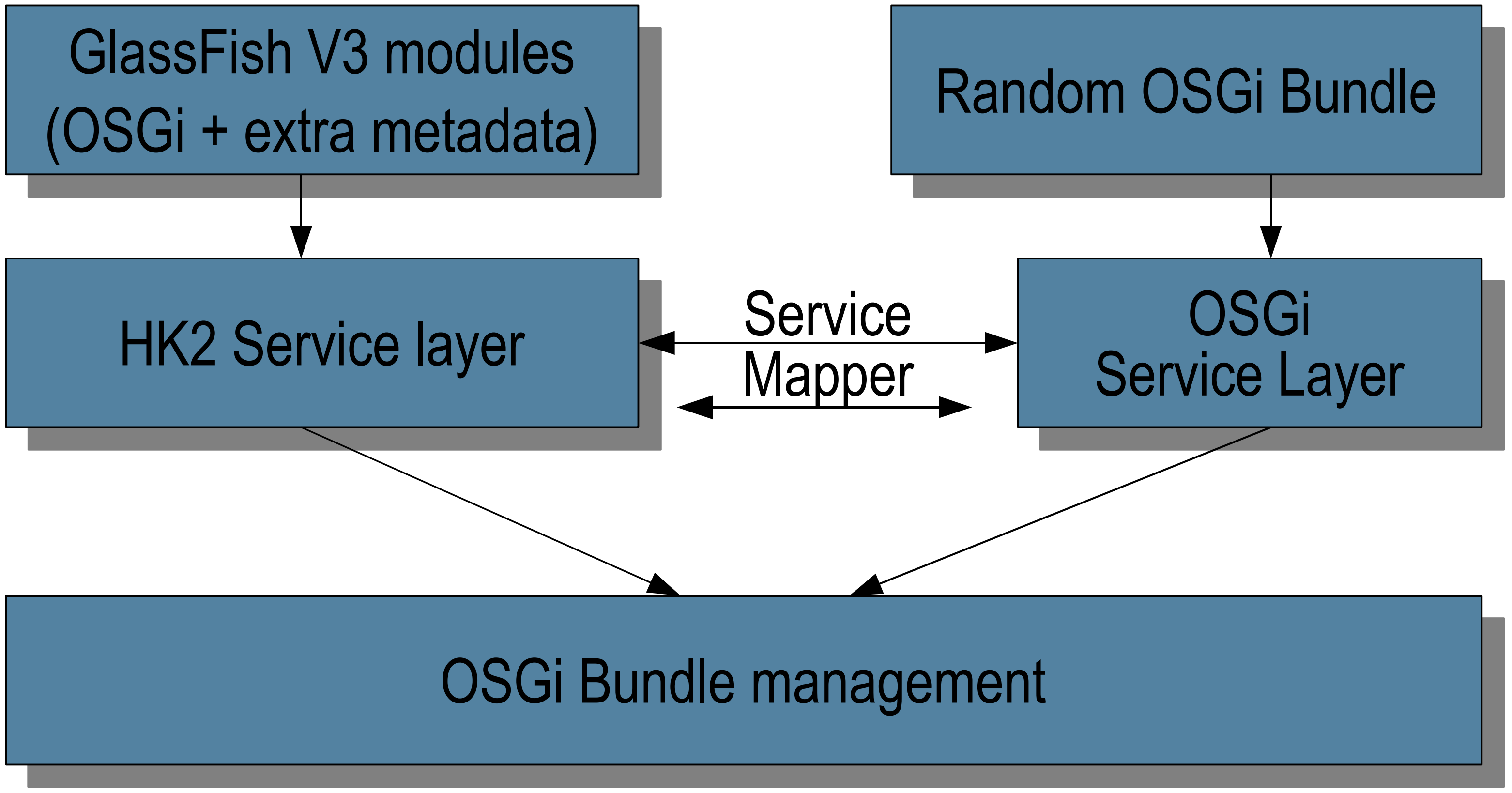
GlassFish v3 Modularization

- Based on OSGi
- Extensible
  - Extensive APIs to replace or extend features
  - OSGi also provides extensions capabilities
- Service based architecture
  - Services are defined by contracts and can be easily substituted
  - Lazy loading based on usage patterns
- Open for all JVM based technologies
  - JRuby/Grails
  - Native deployment (no war repackaging)
- Successfully maintained quick startup

GlassFish: The next generation platform



GlassFish v3 Runtime with OSGi



OSGi integration

- Module management
  - > Add, remove, update installed modules
- OSGi as a container !
  - > Treat OSGi just like any container, bundles are deployed to it.
  - > Can leverage OSGi to extend GlassFish
- Converged Applications
  - > Started investigating Java EE 6 + OSGi converged applications :
    - Dependencies in OSGi
    - Lifecycle still governed by Java EE.

OSGi Integration (2)

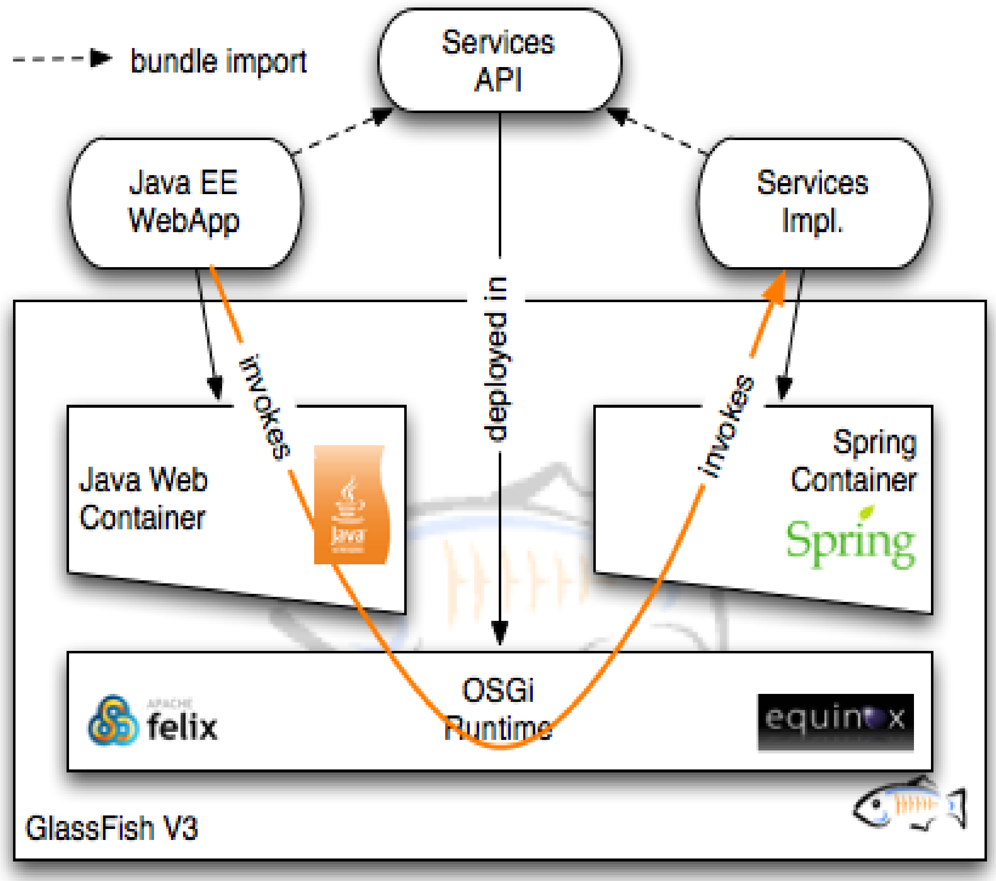
- OSGi services
  - Available to any Java EE application

```
@Resource(mappedName=”osgiName”)
SomeOSGiService injectedService;
```

  - JNDI lookup
  - Portable, no OSGi dependencies in your Java EE application code
- No bundle management access
- Bundles exported APIs visible to Java EE apps

Extending GlassFish v3

SpringDM – another example, demo and picture



- Extend GlassFish with an unmodified Spring dm container
- Simple Spring bean implementing the service
- Invoke the service from a servlet using standard `@Resource` injection
- Still no use of a GlassFish API
- Single runtime for both Spring and full Java EE

Step by step: http://blogs.sun.com/dochez/entry/glassfish_v3_extensions_part_4

OSGi-Enabled Java EE Applications

- No automatic wrapping
  - > Users must convert their existing Java EE archive format to an OSGi bundle
  - > In the future, this could be automated potentially
- Support for bundle WARs
  - > WAR is a single bundle
  - > RFC 66 support
  - > OSGi runtime classloader used by the web container to load classes and resources

OSGi bundles deployment

Pure OSGi bundles can be deployed to the application server :

- as a library
- managed like an application

OSGi aware runtime can use well know extended pattern to listen to bundle installation and trigger interesting behaviours.

Makes extended GlassFish possible without using a singe GF API.

Exposing Java EE Services to OSGi

- Application developers can choose to export
  - > EJBs
  - > Resources
    - JDBC DataSource, JavaMail resource, JMS resource
  - > JPA EntityManagerFactories

OSGi Service Implementations

- HTTP Service
  - > Simple dynamic servlet web server
- RFC 98
  - > Transactions in OSGi
- RFC 66
  - > OSGi-based web container

Looking Forward

- GlassFish v3
  - Support OSGi-Enabled Java EE applications
  - Implement Java EE-related OSGi services
  - Expose Java EE services as OSGi services
  - Improve underlying OSGi framework administration
- OSGi is no longer under the covers
  - Raises visibility from GlassFish developers to GlassFish users