June 10-11, 2008      Berlin, Germany

OW2 JOnAS 5.0 Java EE™ AS
An OSGi™ Based Integration Platform

François Fornaciari & Walter Rudametkin
Bull SAS & UJF Grenoble
Francois.Fornaciari@bull.net
Walter.Rudametkin@imag.fr
Agenda

- Introduction to JOnAS 5.0
- An integration platform based on OSGi™
  - Modular platform
  - Adaptive platform
  - Flexible platform
- Future work
- Demonstration
Agenda

• Introduction to JOnAS 5.0
• An integration platform based on OSGi™
  • Modular platform
  • Adaptive platform
  • Flexible platform
• Future work
• Demonstration
Java EE™ Application Server

- Open Source (LGPL)
  - Hosted by the OW2 consortium
- J2EE 1.4 Compliant (Sun Certification)
- EasyBeans: EJB3™ lightweight container
- Scalability and Availability
  - Dynamic clustering (HTTP, RMI, DB), Failover Optimization mechanisms (pooling, caching, …)
- Enterprise Integration
  - Multi-tier Infrastructure
  - Apache, LDAP, DBMSs, JCA connectors to ERPs, mainframes
  - Web Services, JMS
  - Application versioning
- Administration (JMX, WS, EJB)
  - Web console, script commands, JASMINe
JOnAS development community

- **Bull**
  - Leader
  - Support and services based on JOnAS

- **France Telecom**
  - Intensive operational use of JOnAS
  - Clustering (validation, migration)

- **UJF**
  - Collaboration on OSGi™

- **Peking University, CVICSE**
  - Services architecture, Clustering, Web Services, Management

- **INRIA**
  - Self Management

- **UPM**
  - Clustering

- **LIFL**
  - Administration/monitoring, deployment

- **UNIFOR**
  - Clustering
JOnAS 5.0: OSGi™ based Architecture

- Major refactoring
- OSGi™ benefits
  - Modularity / Maintenability
  - Classpath consistency
  - Dynamic [re]configuration
  - Facilitates on demand services
  - Self adaptability to application, user and environment constraints
- Access to the OSGi™ world
  - RFID, sensors, …
JOnas 5.0: The basics

- Configuration
  - List of services to start
  - List of bundles to deploy at startup
- OSGi framework implementation: Apache Felix
- At runtime
  - Service dependencies managed by Apache iPOJO
JOnAS 5.0: Management

- Specialized Java EE™ administration tools (JMX)
  - JOnAS commands (start, stop, halt, admin, …)
  - Web console
- Direct access to the OSGi™ framework
  - Recommended for knowledgeable users
  - Felix shell and Felix GUI
Agenda

• Introduction to JOnAS 5.0
• An integration platform based on OSGi™
  • Modular platform
  • Adaptive platform
  • Flexible platform
• Future work
• Demonstration
Modular platform (1/3)

- Bundle, bundle, bundle
  - All components and modules are OSGi™ Bundles
- Homogeneous and flexible platform
  - JOnAS technical services
  - Java EE™ applications (EasyBeans: EJB™ 3.0)
  - OSGi™ services

THEY’RE ALL BUNDLES!
Modular platform (2/3)

- Each module has its own lifecycle
  - Individual management
  - Better state control
- Easy to replace service implementations
  - Choose service implementation at any time
    - Initially in the configuration
    - At runtime
  - Minimal repercussions on other modules
Modular platform (3/3): JOnAS 5.0 architecture
Agenda

• Introduction to JOnAS 5.0
• An integration platform based on OSGi™
  • Modular platform
  • Adaptive platform
  • Flexible platform
• Future work
• Demonstration
Adaptive platform (1/5)

• Deployment of new modules at runtime
  • Depends on applications’ needs
  • Dynamic deployment using the OSGi™ Bundle Repository (OBR)
    • Local resources
    • Remote resources
• Possible to specify preferences:
  • Implementation (class)
  • Vendor
  • Version
Adaptive platform (2/5): Service on demand

- Application declares the required JOnAS services
  - Add an entry to the MANIFEST

    Manifest-Version: 1.0
    ....
    Required-JOnAS-Services: mail

- Before deploying the service, all requirements are automatically deployed
- Service preferences are defined in the JOnAS configuration (implementation, version, …)
Adaptive platform (3/5): Automated undeployment

- Automatic undeployment of the modules automatically deployed and no longer needed by applications
- Checks performed when an application is undeployed
- Possible to define undeployment policy
  - Always undeploy
  - Never undeploy
  - Fine grained policies
Adaptive platform (4/5): Service update

• Administrative action
  • jonas update -services [s1, s2, …]

• Underlying process
  1. Stop the module
     • Stop the applications that require it
  2. Lookup the latest version and deploy it from a local (or remote) repository
  3. Start only the new module
     • Restart the stopped (dependent) applications
Adaptive platform (5/5): Summary

- The integration platform can dynamically react to changes in the execution environment
- On demand incremental service delivery
- Useful to reduce memory footprint and resource consumption
Agenda

- Introduction to JOnAS 5.0
- **An integration platform based on OSGi™**
  - Modular platform
  - Adaptive platform
  - Flexible platform
- Future work
- Demonstration
Flexible platform (1/4)

- Communications handled by the OSGi™ service layer
  - Loose-coupling between modules
  - Not dependent on a specific implementation
- Allows dynamic OSGi™ service replacement
  - Ex: replace a simple log service by a persistent log service at runtime
Flexible platform (2/4):
Java EE™ ➔ OSGi™ interactions

• Java EE™ modules communicate with the OSGi™ world
• EJB™ 3.0 can use the BundleContext

```java
/*
 * Bundle context that will be injected by the EZB container.
 */
@OSGiResource
BundleContext bundleContext = null;
```

• Listen to OSGi™ events (framework, bundle, service)
• Access to OSGi™ services
Flexible platform (3/4):
OSGi™ → Java EE™ interactions

• Why expose Java EE objects, like Stateless Beans, as OSGi™ services?
  • Beans offer entry points to the Java EE™ world

• Advantage:
  • OSGi™ applications can access Java EE™ components transparently
    • Benefit of persistence, transaction, security, …

Under development…
Flexible platform (4/4): Summary

- JOnAS offers dynamic interactions between OSGi™ services and Java EE™ applications
  - Benefit from the best of both worlds
  - OSGi applications use Java EE™ features as any other OSGi service, and *vice versa*!
  - Infinite combinations between applications
  - Open large perspectives in many domains
    - *Embedded, multi-paradigm, edge, self-managed application servers*
Agenda

• Introduction to JOnAS 5.0
• An integration platform based on OSGi™
  • Modular platform
  • Adaptive platform
  • Flexible platform
• Future work
• Demonstration
Future work

- Service on demand will be available in a future release
- Advanced diagnosis tool
- Improve @OSGiResource to support OSGi service injections
- Smart calls to the OBR to optimize service deployment
  - The dependencies resolver takes time (~4 s per resolution)
- How to implement an “obr undeploy” command?
  - Should it also undeploy service requirements if no longer used
- Virtual service
  - Service is loaded when a client uses it
- Update center
- How to add package dependencies at runtime?
  - Ex: bytecode injection adds new requirements
  - Plan to specify this issue in the next OSGi™ specifications?
JOnAS 5 ROADMAP

- Java EE 5 certified
- M1 Jul 07
- Services refactoring Tomcat 6 EJB3
- OSGi newCmi SE release
- May 08 5.0.3 RC
- Apr 08 5.0.2 RC
- Jan 08 5.0.1 RC
- Dec 07 5.0.0 RC
- 5.1 Nov 08
Agenda

• Introduction to JOnAS 5.0
• An integration platform based on OSGi™
  • Modular platform
  • Adaptive platform
  • Flexible platform
• Future work
• Demonstration
Demo: Service on demand
For more information

- **JOnAS**
  Documentation, download, mailing lists, …
  [http://jonas.ow2.org](http://jonas.ow2.org)

- **Apache Felix**
  [http://felix.apache.org](http://felix.apache.org)

- **Java EE 5**
THANK YOU

Q & A