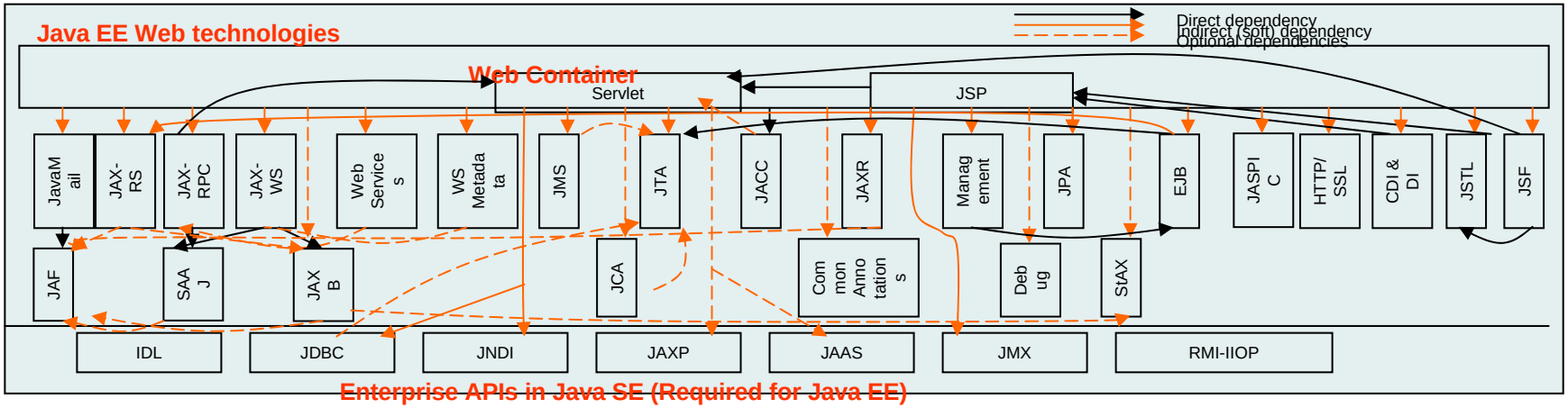
Enterprise OSGi in WebSphere and Apache Aries

Ian Robinson, IBM

Agenda

- OSGi and Java Enterprise – who needs who?
- The Culture Clash
- Apache Aries
- What's Next in Aries and Enterprise OSGi?
- Aries and WebSphere Application Sever

OSGi and Java Enterprise – who needs who?



HTTP Session Replication | Connection Management | Persistence Providers | User Registries | Event Mgmt

Logging and Tracing | MBean Server | Admin Console and scripting | DataSources | Messaging Engines | Security

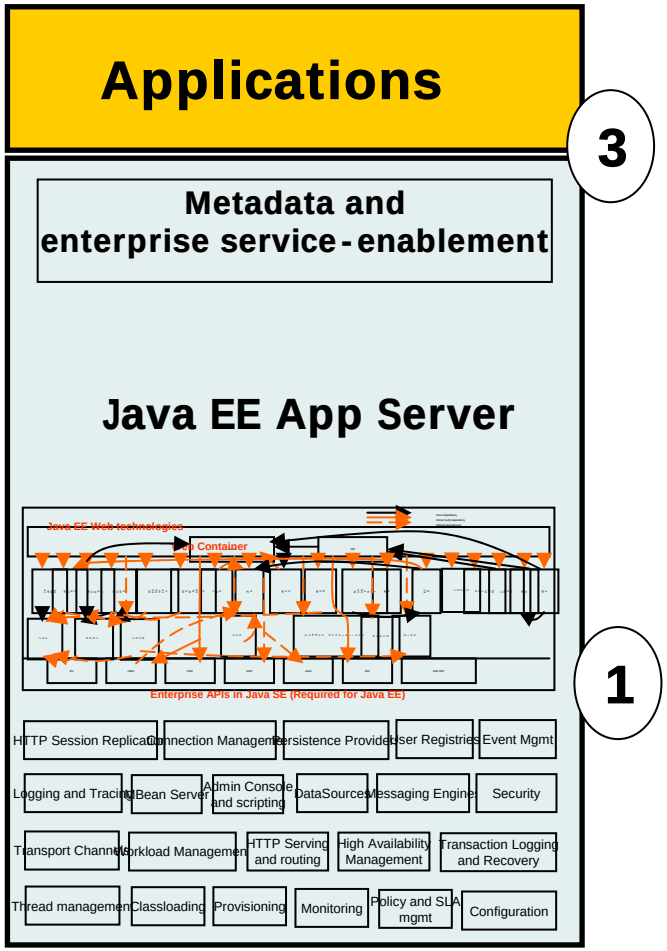
Transport Channels | Workload Management | HTTP Serving and routing | High Availability Management | Transaction Logging and Recovery

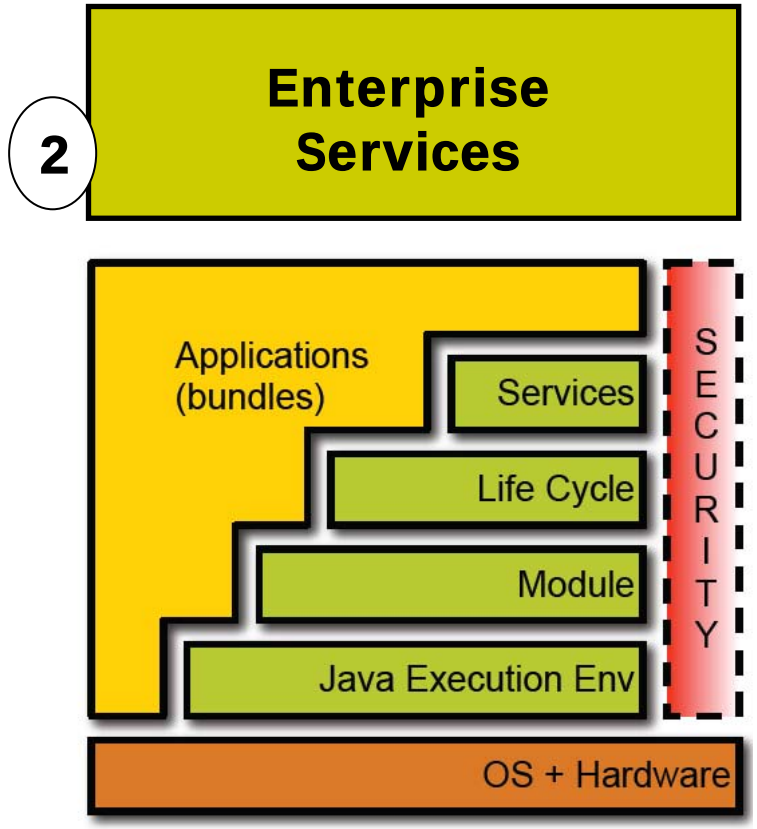
Thread management | Classloading | Provisioning | Monitoring | Policy and SLA mgmt | Configuration

...

OSGi and Java Enterprise – who needs who?

Three ways of looking at Enterprise OSGi technologies





The Culture Clash

Java EE	OSGi
An application is a collection of wars/jar modules scoped by an EAR	There is no multi-bundle "application" scope other than an OSGi framework
Applications are isolated from one another in a server	Bundle exports are visible throughout the framework
Container-centric. Distinct contracts for containers and applications	All bundles are created equal.
Defines complete enterprise programming model (JTA, JPA, Web, JMX, Web, EJB, JAX-WS...)	OSGi Enterprise Specification V4.2 defines services-based access to Java EE technologies

OSGi Enterprise Specification

- Released 22 March 2010
  - The product of the OSGi Enterprise Expert Group (EEG)
- Brings Enterprise technologies and OSGi together
- Using existing Java SE/EE specifications:
  - JTA, JPA, JNDI, JMX, WebApps…
- Plus Spring-derived *Blueprint* component model and DI container

- **Java EE provides the core enterprise application programming model**
- **Deploying modules as OSGi bundles simplifies reuse between applications, provides versioning, encourages (and enforces) modular design and enables dynamic module updates.**

Enterprise OSGi in Apache Aries

- Apache Aries Project started in Sep 2009
  http://incubator.apache.org/aries/
  - Delivering a set of pluggable Java components enabling an enterprise OSGi application programming model.
  - Implementations and extensions of application-focused specifications defined by the OSGi Alliance Enterprise Expert Group (EEG) and an assembly format for multi-bundle applications, for deployment to a variety of OSGi based runtimes.
  - to build a broad development community to encourage implementation and adoption of EEG specs
- Just delivered 2nd (incubator) release.
  - OSGi Compliance Tests published for each release.
- Aries componentry supporting an enterprise OSGi programming model has been integrated into both Geronimo and WebSphere Application Server.
  - As well as Apache Felix Karaf, JBossOSGi and others
- Project was 1 year old last week

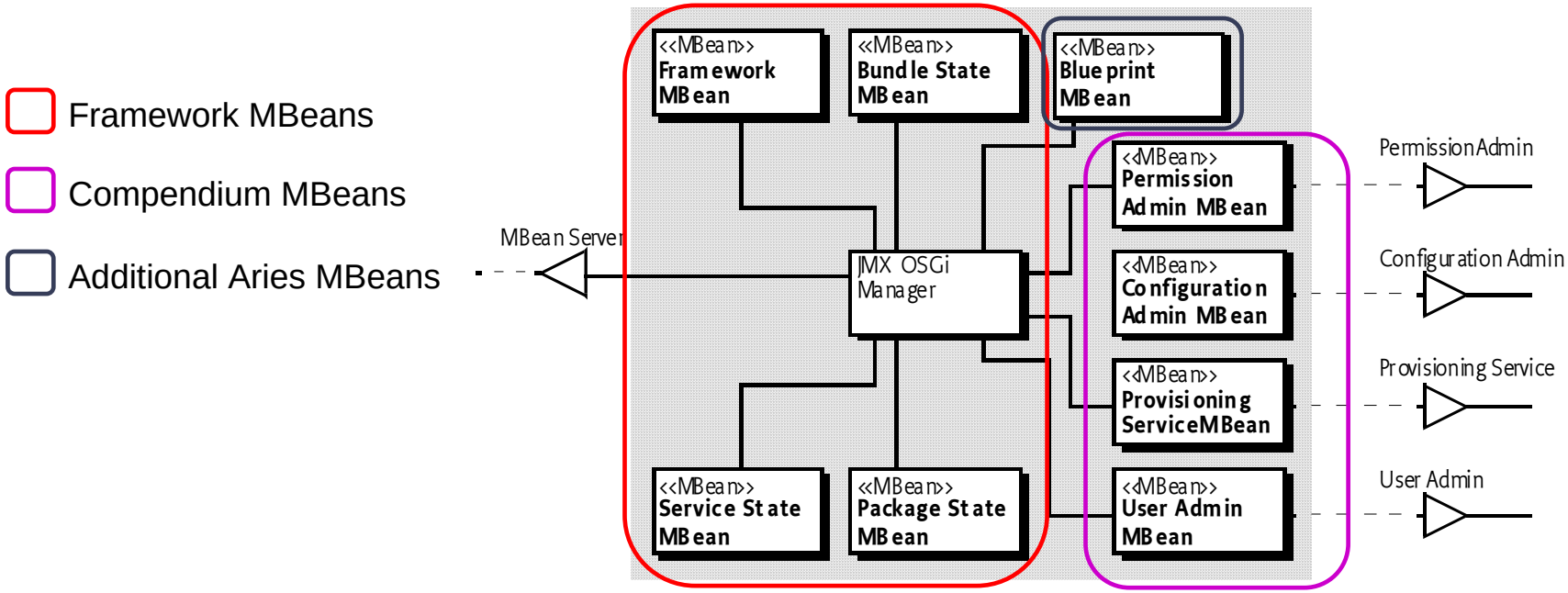
Aries projects include...

- JMX
- JTA integration
- Blueprint container
- JPA integration
- JNDI integration
- Application assembly and deployment
- Samples, documentation, integrator's guide

http://svn.apache.org/repos/asf/incubator/aries/trunk/

Aries JMX Integration

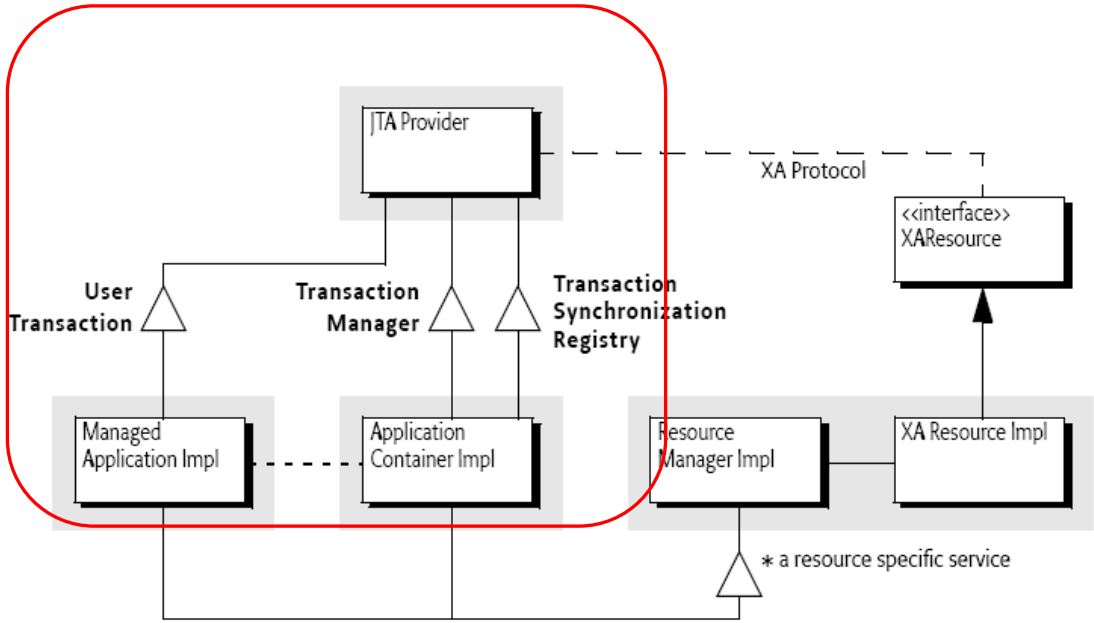
- Implementation of OSGi JMX specification.
- Aries JMX bundle automatically registers the JMX MBeans into any javax.management.MBeanServer service in the OSGi Service Registry.



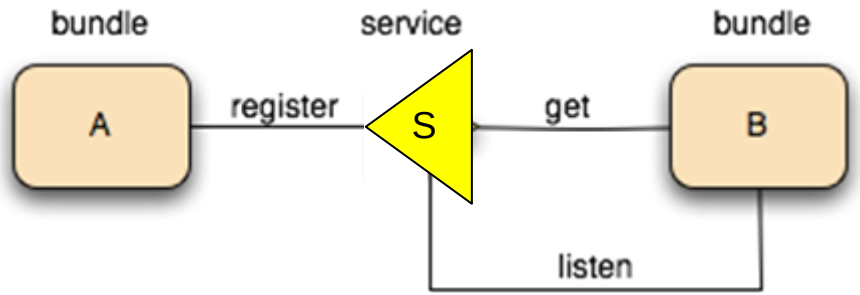
Aries JTA integration

- Apache Aries integrates the OSGi Transaction Service Ref Impl from Apache Geronimo.

- This spec is a good example of the OSGi/Java EE culture clash:
  - Different services for application and container



OSGi Services – No Java EE analog



- Services registered in Service Registry.
- Services are dynamic
- Clients (bundle B) need to cope with them going away
- There is no analog for dynamic services in Java EE.

Using Services can be hard

```
private BundleContext ctx;

        private AtomicReference<LogService> ls = new AtomicReference<LogService>();
        private AtomicReference<ServiceReference> lr = new AtomicReference<ServiceReference>();

        public void start(BundleContext ctx) throws InvalidSyntaxException
        {
                this.ctx = ctx;
                ctx.addServiceListener(this, "(objectClass=org.osgi.service.log.LogService)");
                ServiceReference ref = ctx.getServiceReference(LogService.class.getName());
                if (ref != null) {
                        ls.set((LogService) ctx.getService(ref));
                        lr.set(ref);
                }
        }

        @Override
        public void serviceChanged(ServiceEvent event)
        {
                ServiceReference ref = event.getServiceReference();

                if (ls.get() == null && event.getType() == ServiceEvent.REGISTERED) {
                        ls.set((LogService) ctx.getService(ref));
                } else if (ls.get() != null && event.getType() == ServiceEvent.UNREGISTERING &&
                                ref == lr.get()) {
                        ref = ctx.getServiceReference(LogService.class.getName());
                        if (ref != null) {
                                ls.set((LogService) ctx.getService(ref));
                                lr.set(ref);
                        }
                }
        }
```

Blueprint Simplification

```
<blueprint>
  <bean id="shop" class="org.example.ecomm.Bean">
    <property name="log">
      <reference interface="org.osgi.service.log.LogService"/>
    </property>
  </bean>
</blueprint>
```

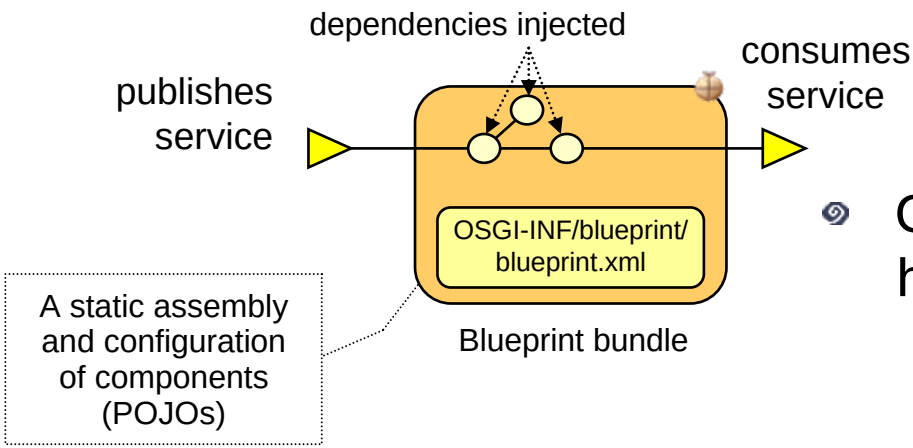
```
public class Bean {

private LogService log;
void setLog(LogService logService) {
      log = logService;
      }

void process(Order o) {
      log.log(LOG_INFO, "processing: " + o);
      }
}
```

Aries Blueprint Container

- XML Blueprint definition describes component configuration and scope
  - Optionally publish and consume components to/from OSGi service registry.
- Simplifies unit test outside either Java EE or OSGi r/t.



- The **Aries BP container implementation** is highly extensible:
  - Namespace handlers supported to extend the Blueprint definitions
  - Bean interceptors can be registered by handlers
- Other Aries components contribute handlers – “jpa” and “jta” handlers.

Aries Blueprint Extensions

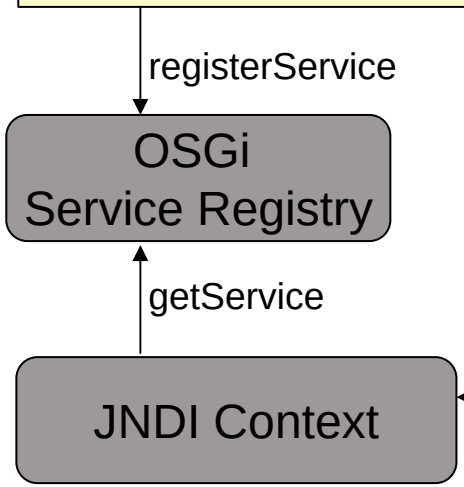
- Container managed JPA support integrated with Aries Blueprint container:
  - @PersistenceUnit or @PersistenceContext (managed)
  - or <jpa:unit>, <jpa:context> bean property injection
  - Familiar development experience for JPA developers
  - Load-time enhancement of Entity classes (WebSphere)
- Same container managed transaction attributes as EJBs:
  - Required, RequiresNew, Mandatory, NotSupported, Supports, Never

```
<blueprint>
  <bean id="shop" class="org.example.ecomm.ShopImpl">
      <jpa:context property="em" unitname="myUnit"/>
      <tx:transaction method="*" value="Required"/>
  </bean>
</blueprint>
```

Aries JNDI integration

- Provides JNDI-based access to OSGi Service Registry

```
<blueprint xmlns=...>
   <bean id="bloggingServiceComponent"
       class="org.apache.aries.BloggingServiceImpl">
   </bean>
   <service ref="bloggingServiceComponent"
      interface="org.apache.aries.samples.blog.api.BloggingService"/>
   ...
</blueprint>
```

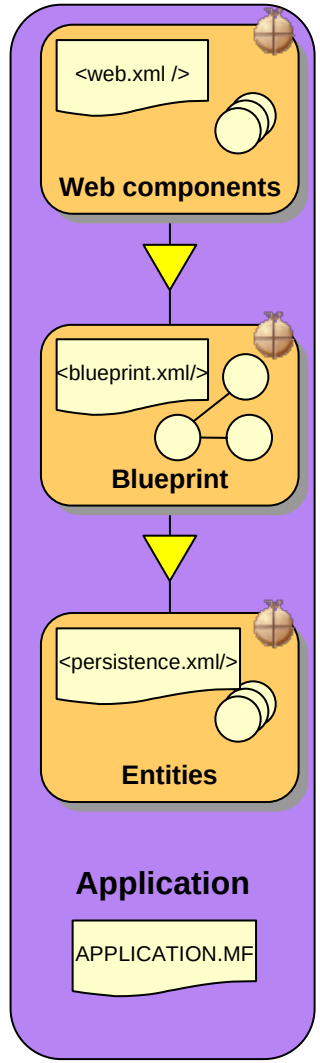


- A way for a Web component to access a Blueprint component

```
InitialContext ic = new InitialContext();
BloggingService blog= ic.lookup("osgi:services/"
                + BloggingService.class.getName());
```

Application-level Metadata and Archive

- Apache Aries defines the notion of an “enterprise bundle archive” (EBA) to represent a multi-bundle application.
- Constituent bundles may be contained (“by-value”) or referenced from a bundle repository.
- Config by exception - absence of APPLICATION.MF metadata means:
  - application content is the set of bundles contained by-value plus any repository-hosted dependencies identified during deployment.



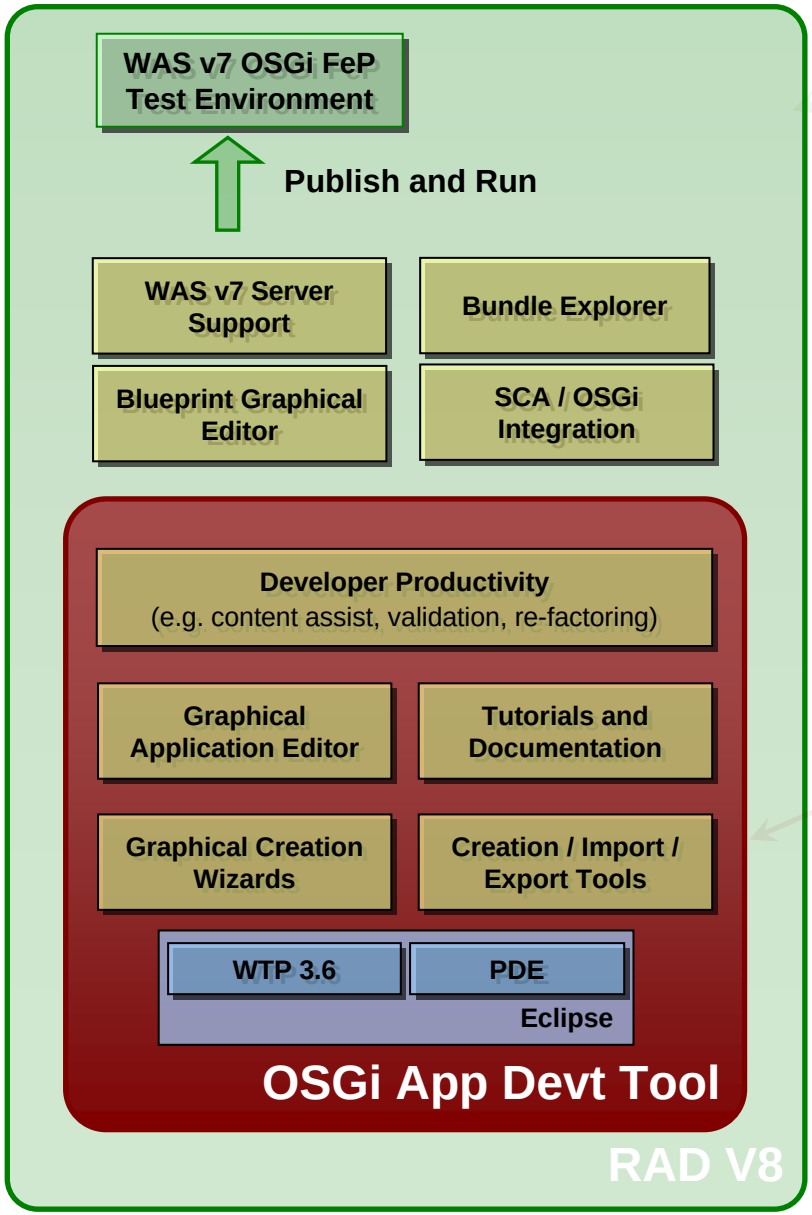
Aries Application Assembly and Deploy

- "application" project: resolve and deploy .eba archives
  - bundle converter extn point e.g for war to wab conversion
  - application resolver extn point for different type of repository e.g Felix OBR
  - application context extn point for runtime specific installation and management of bundles

```
Manifest-Version: 1.0
Application-ManifestVersion: 1.0
Application-Name: Blog Application
Application-SymbolicName: aries.sample.blog
Application-Version: 1.0
Application-Content:
  aries.sample.blog; version=1.0.0,
  aries.sample.blog-api; version=1.0.0,
  aries.sample.blog-persistence; version=1.0.0,
  aries.sample.blog-servlet; version=1.0.0
```

- Uses APPLICATION.MF metadata
- Metadata can be generated from POM configuration or authored
- Eclipse-based tools for authoring

OSGi Application Development Tools



OSGi Application Support in RAD V8

➢ Provide integrated development and test of OSGi Applications on the WebSphere platform

- Integrated with Web Tools, JEE productivity tools, and other capabilities in RAD
- Supports deployment to WAS v7 OSGi FeP and includes the FeP in the WAS Test Environment
- SCA support for OSGi Applications
- Additional OSGi tools:
  - Graphical and wiring editor for Blueprint
  - Bundle Explorer
  - Tools for WAS OSGi extensions / Value-add

http://www-01.ibm.com/software/awdtools/developer/application/index.html

Free Eclipse Plugin for OSGi Applications

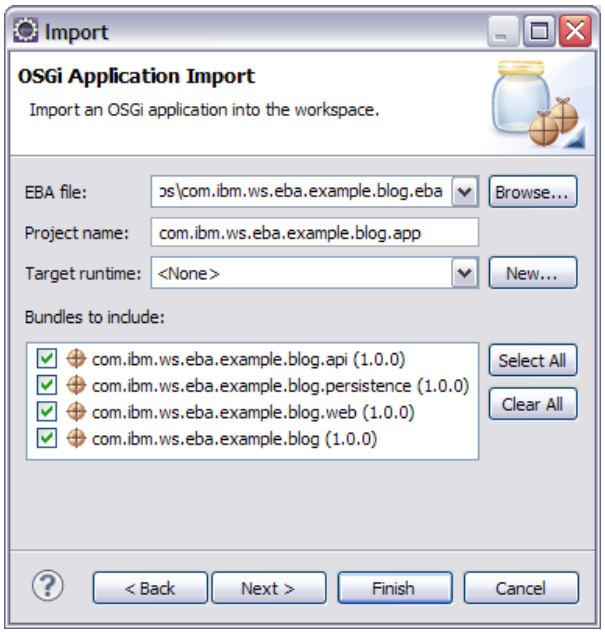
➢Graphical tools to develop OSGi applications and bundles

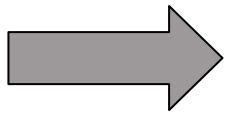
- Includes features that increase developer productivity
- Creates OSGi Applications for any Aries-based server runtime.
- Eclipse WTP 3.6 (Helios) required

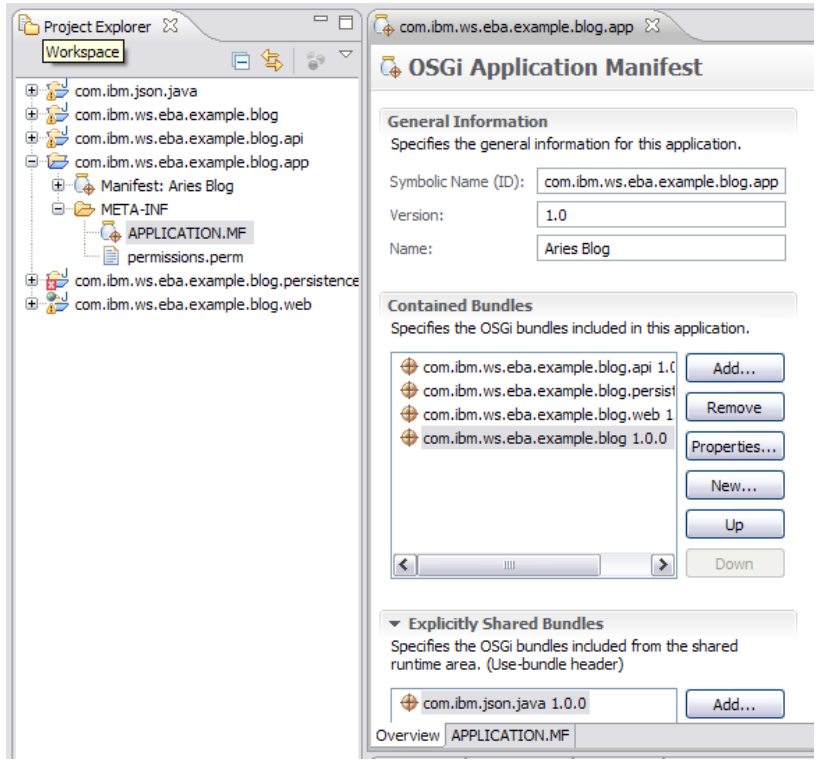
http://marketplace.eclipse.org/content/ibm-rational-development-tools-osgi-applications

Development Support for EBAs

- OSGi App Dev Tool supports import, creation, and export of applications
- Form-based editor for application manifest
- Validation of manifest syntax and properties

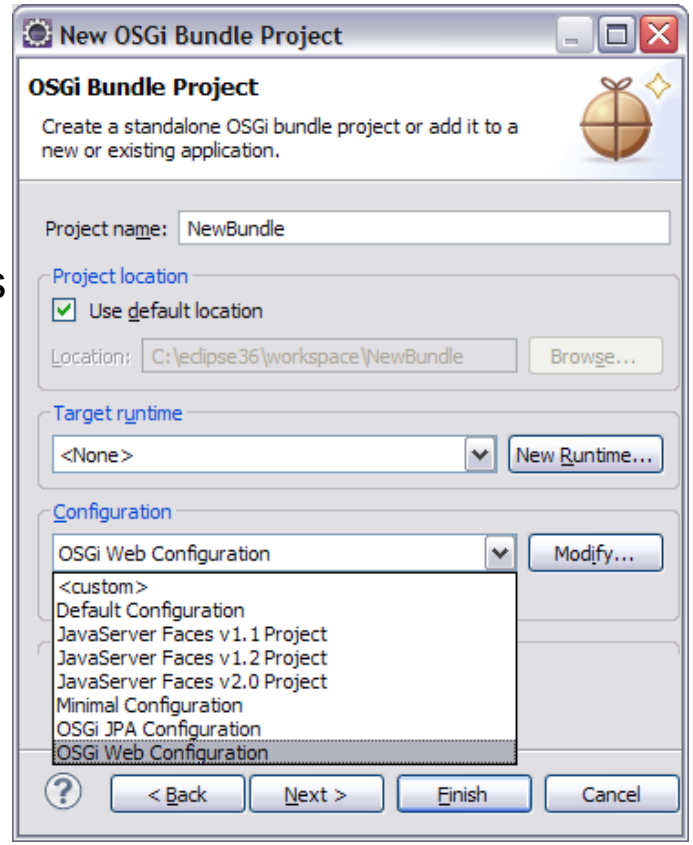






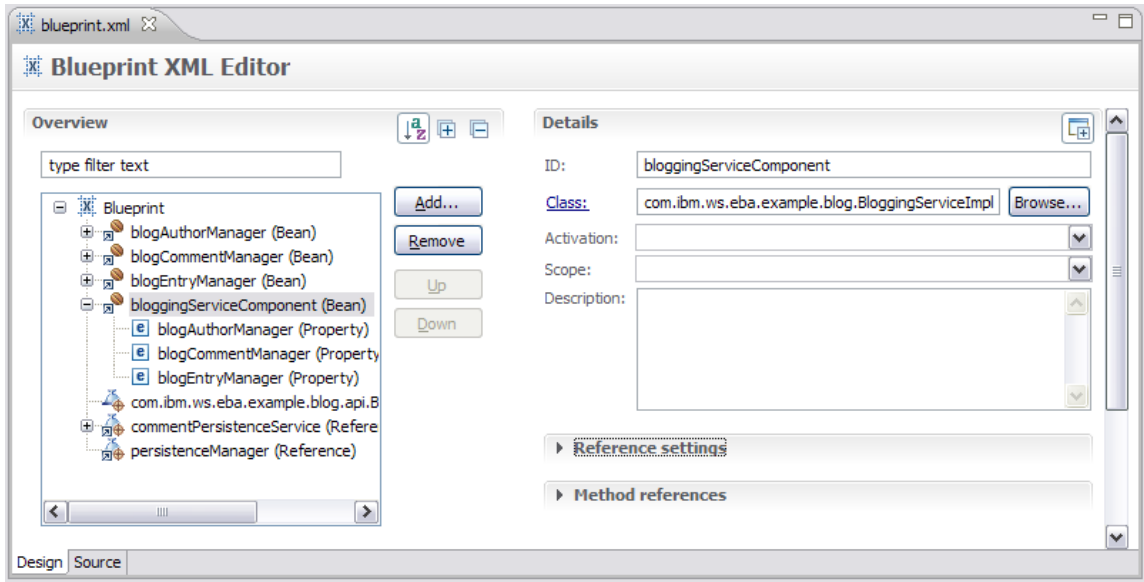
Development Support for Web and JPA Bundles

- Create OSGi bundle projects with Web or JPA configurations
- Can convert existing JEE Web or JPA projects into bundles:
  - Creates or updates a valid bundle manifest
  - Adds a Web-ContextPath or Meta-Persistence header
  - Adds package imports based on current module contents
- Graphical manifest editor for OSGi metadata. Project build paths respect OSGi visibility rules
  - Supports and enforces modular characteristics from design through to execution
- Plus in RAD: Comprehensive support for Web and JPA. These tools can continue to be used as-is in OSGi projects, including:
  - New wizards (Servlet, JSP, etc.)
  - Page designer and other web editors
  - Persistence.xml editor



Development Support for Blueprint Bundles

- Blueprint creation wizard
  - Supports extension schema including JPA and transactions
- A blueprint editor, including:
  - Source page with syntax highlighting, content assist
  - Form-based editing similar to Java EE deployment descriptors (RAD)
  - Syntax and reference validation
  - Hyperlinking to impl classes
  - Refactoring support



Aries Innovation

- Blueprint Bean Interceptors – RFP 137
- Blueprint Transactions – RFP 138
- Message Driven Services – RFP 131
- Subsystem Metadata and Lifecycle – RFP 121
- Container-managed JPA - RFP 115
  - RFP 136 (Classpath scanning)
  - RFP 139 (Bytecode weaving)
- META-INF/services RFP 128
- (Blueprint annotations, no RFP yet)

Application exploitation of OSGi in WebSphere

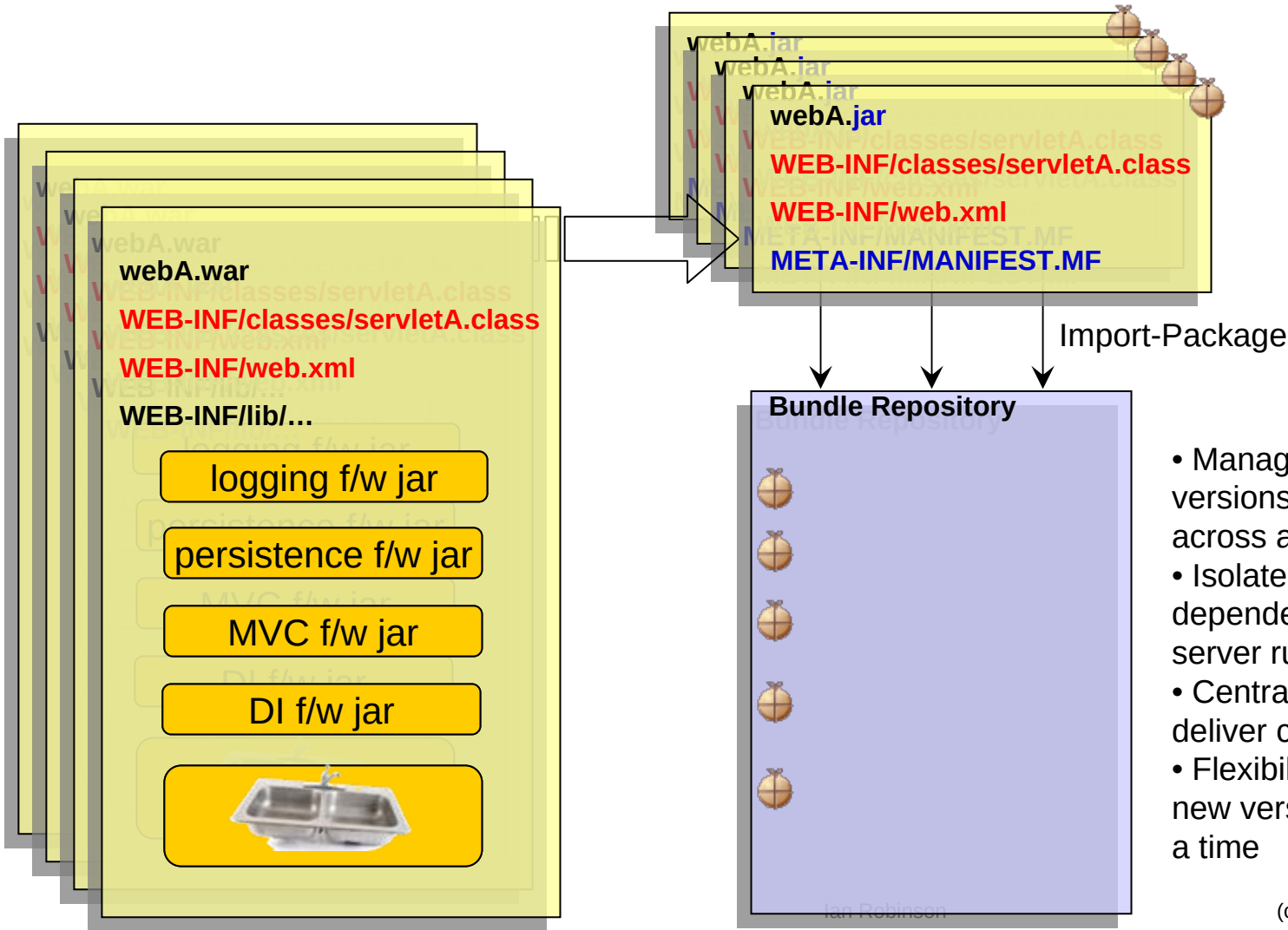
- OSGi has been used internally in WAS since V6.1 and in Eclipse since R3.

- Application-level exploitation is introduced in the **WebSphere Application Server V7 Feature Pack for OSGi Applications and JPA 2.0** http://www.ibm.com/websphere/was/osgi
  - Generally available since May 2010
  - Early Program available since Nov 2009

- Tools support in RAD V8: http://www.ibm.com/software/awdtools/developer/application/

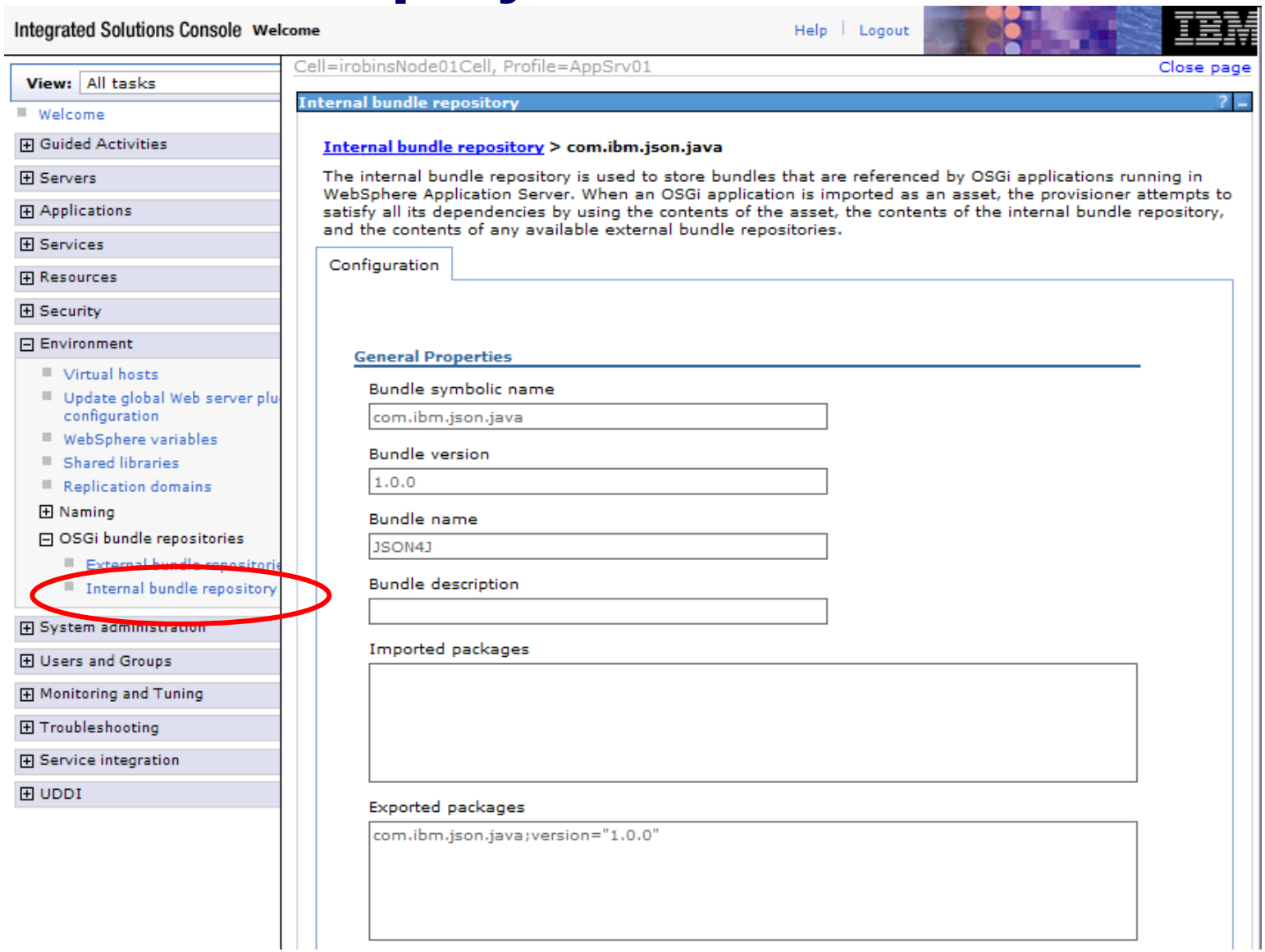
Modular Deployment

Common bundles factored out of the applications and used at specific versions



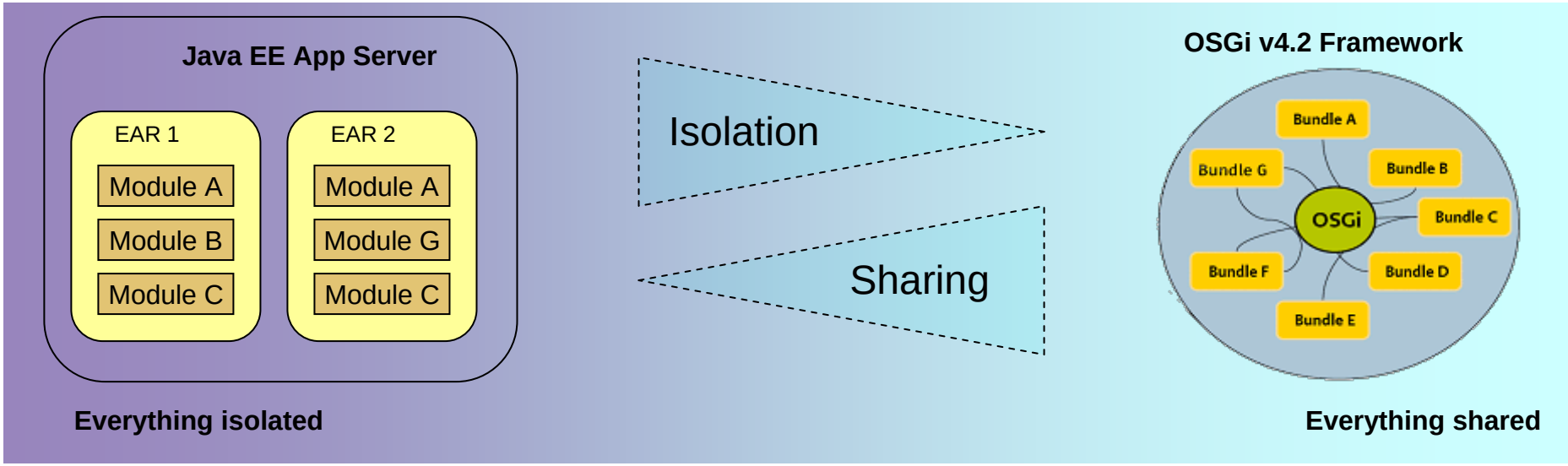
- Manage multiple versions of libraries across an enterprise
- Isolate application dependencies from the server runtime
- Centralized location to deliver critical fixes
- Flexibility to update to new versions one app at a time

Modular Deployment in WAS



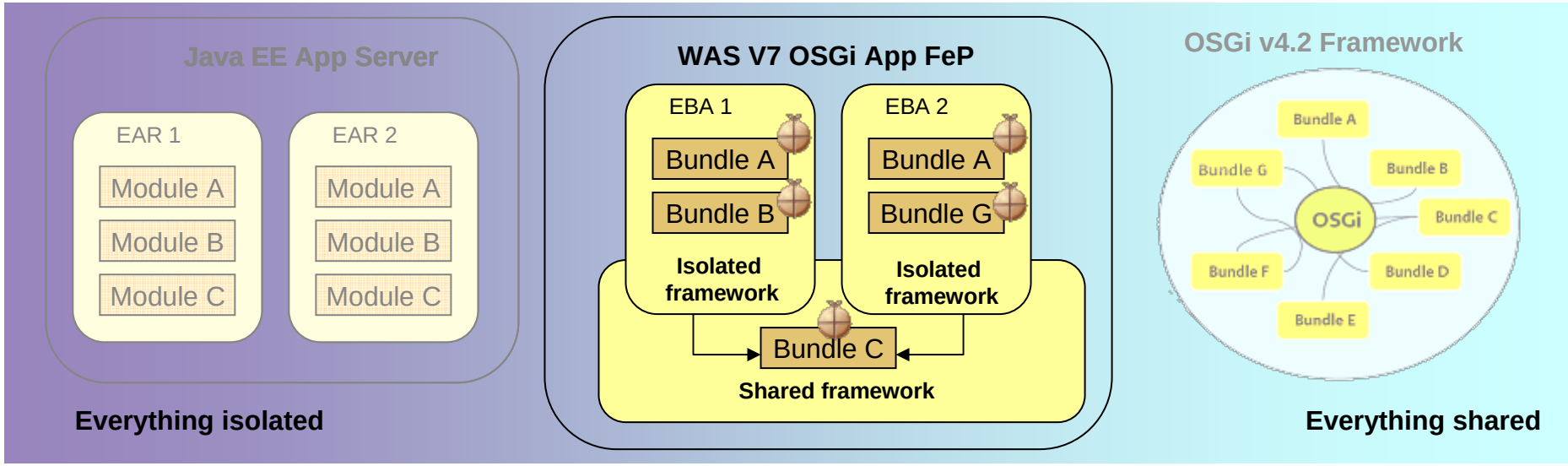
Details: Isolated and Shared Bundles

- In Java EE, modules are isolated within an application and applications are isolated from one another.
  - Makes sharing modules difficult
- OSGi 4.2 all bundles have shared visibility to the externals of all others bundles within an OSGi framework (JVM)



Details: Isolated and Shared Bundles

- Equinox 3.5 “nested framework” support enables “composite bundles” to run in isolated child frameworks
  - WebSphere installs each OSGi Application into an isolated child framework.
  - Shared bundles are installed into the (single) parent framework.



Example “Blog” Application

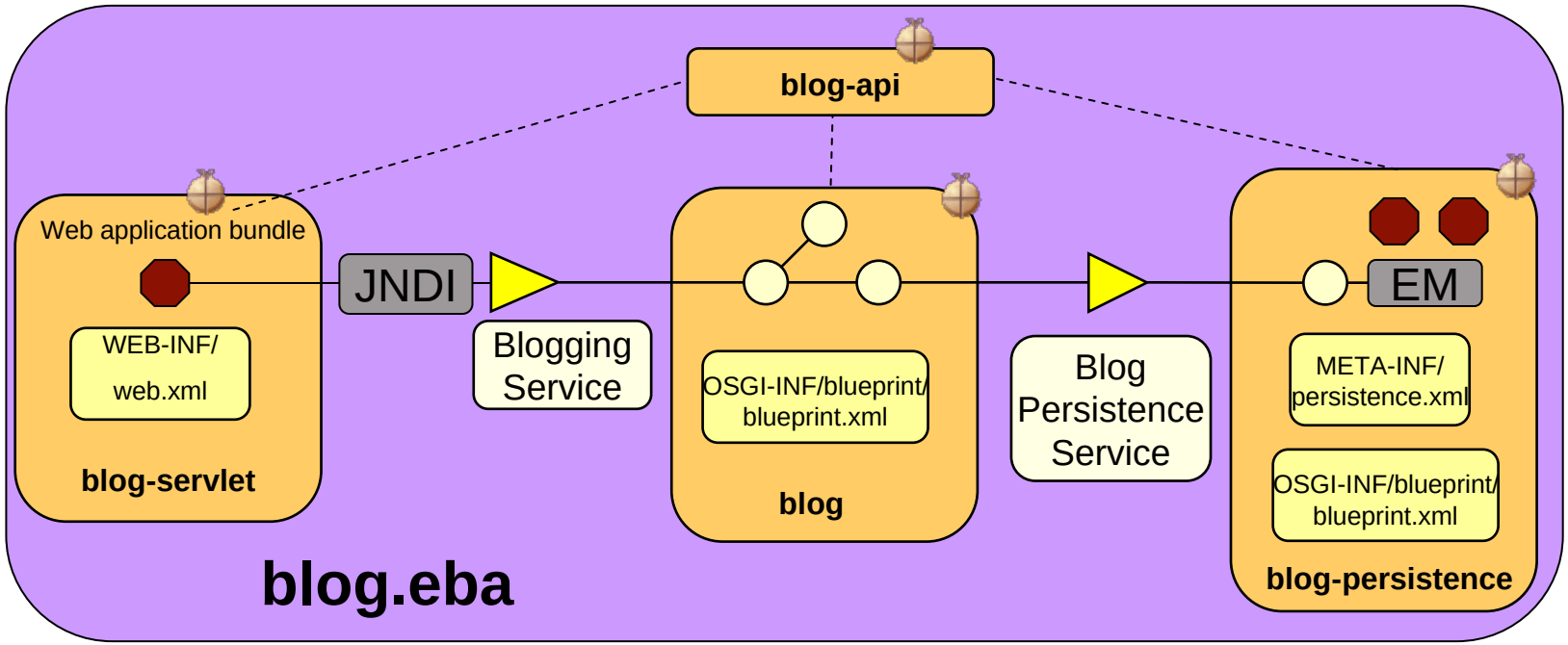
com.ibm.ws.eba.example.blog.app

```
Manifest-Version: 1.0
Application-ManifestVersion: 1.0
Application-Name: Aries Blog
Application-SymbolicName: com.ibm.ws.eba.example.blog.app
Application-Version: 1.0
Application-Content: com.ibm.ws.eba.example.blog.api;version=1.0.0,
 com.ibm.ws.eba.example.blog.persistence;version=1.0.0,
 com.ibm.ws.eba.example.blog.web;version=1.0.0,
 com.ibm.ws.eba.example.blog;version=1.0.0
Use-Bundle: com.ibm.json.java;version=1.0.0
```

Overview | APPLICATION.MF

isolated content

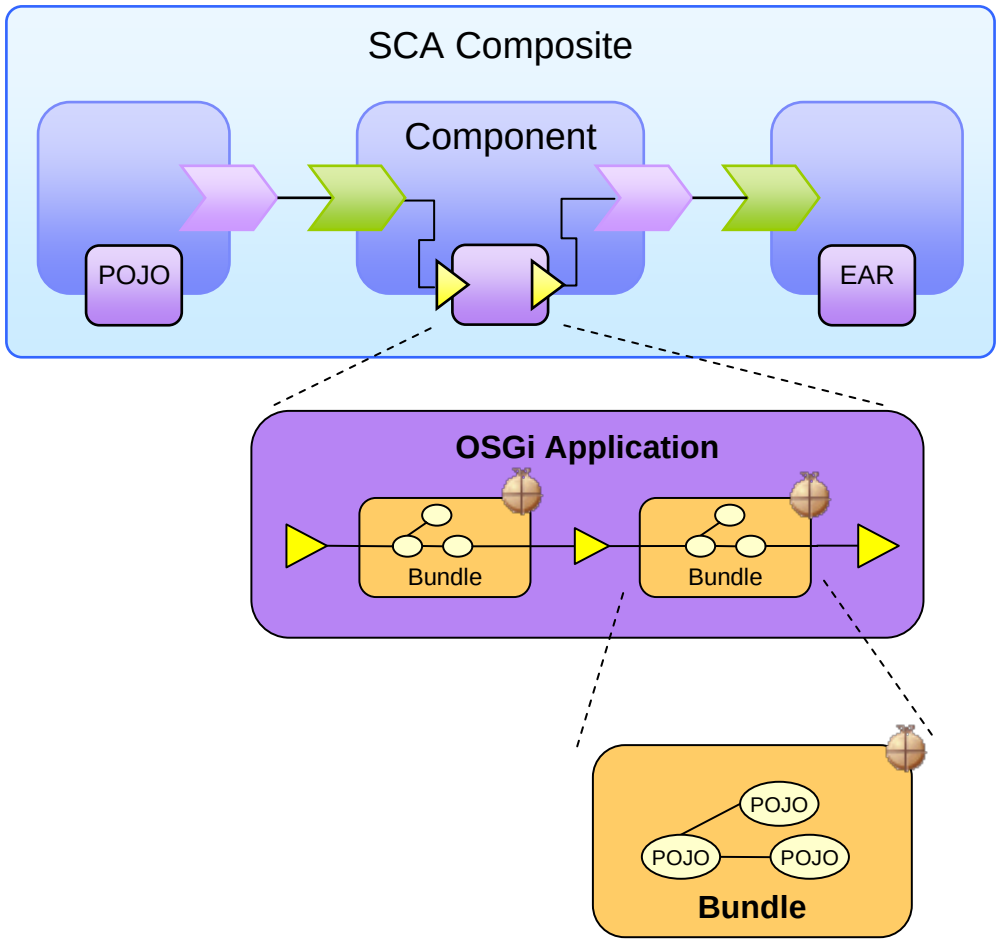
shared content



JMX and Isolated Frameworks in Action

```
wsadmin>list()
 ID   Framework                          Version  Node            Server
 0    SharedBundles                      7.0.0    irobinsNode01   server1
 1    com.ibm.ws.eba.example.blog.app    1.0.0    irobinsNode01   server1
wsadmin>connect(0)
CWSAJ0035I: Connecting to framework SharedBundles_7.0.0 on node irobinsNode01 an
d server server1.
CWSAJ0036I: Successfully connected to framework SharedBundles_7.0.0.
wsadmin>ss()
ID      State       Bundle
0       ACTIVE      org.eclipse.osgi_3.5.2.R35x_v20100126
1       ACTIVE      shared.bundle.framework_0.0.0
2       ACTIVE      com.ibm.json.java_1.0.0
3       ACTIVE      com.ibm.ws.eba.example.blog.app_1.0.0
wsadmin>connect(1)
CWSAJ0035I: Connecting to framework com.ibm.ws.eba.example.blog.app_1.0.0 on nod
e irobinsNode01 and server server1.
CWSAJ0036I: Successfully connected to framework com.ibm.ws.eba.example.blog.app_
1.0.0.
wsadmin>ss()
ID      State       Bundle
0       ACTIVE      org.eclipse.osgi_3.5.2.R35x_v20100126
1       ACTIVE      com.ibm.ws.eba.example.blog.app_1.0.0
2       ACTIVE      com.ibm.ws.eba.example.blog_1.0.0
3       ACTIVE      com.ibm.ws.eba.example.blog.web_1.0.0
4       ACTIVE      com.ibm.ws.eba.example.blog.persistence_1.0.0
5       ACTIVE      com.ibm.ws.eba.example.blog.api_1.0.0
wsadmin>
```

OSGi Applications and SCA: the assembly food chain



SCA Composite assembled from heterogeneous components including an **OSGi Application** component, and integrated through SCA services with configurable bindings (JMS, web services…).

OSGi Bundles assembled in an **OSGi Application** and integrated through services in the OSGi service registry

POJOs assembled using a Blueprint context and scoped by an **OSGi Bundle**.

Summary

- Java EE and OSGi are good for each other.
  - Java EE defines standard enterprise technologies with contracts for applications and runtimes
  - OSGi encourages and enforces modular design and enables modular deployment
- The Apache Aries project delivers enterprise OSGi technologies that have been integrated into a number of runtimes.
  - As well as experimental implementation of new EEG RFPs to inform the specs.
- WAS V7 and RAD integration of Aries provides complete develop-test-deploy-manage environment for modular enterprise applications.

References

- Apache Aries
  http://incubator.apache.org/aries/
- Apache Geronimo v3.0-M1:
  http://geronimo.apache.org/
- WAS V7 Feature Pack for OSGi Applications and JPA 2.0
  http://www.ibm.com/websphere/was/osgi
- OSGi Application Development Plugin:
  http://marketplace.eclipse.org/content/ibm-rational-development-tools-osgi-applications
- Rational Application Developer V8:
  http://www.ibm.com/software/awdtools/developer/application/