



Katya Todorova | SAP Labs Bulgaria

Enterprise Platform Over OSGi: Migration Diary

Background

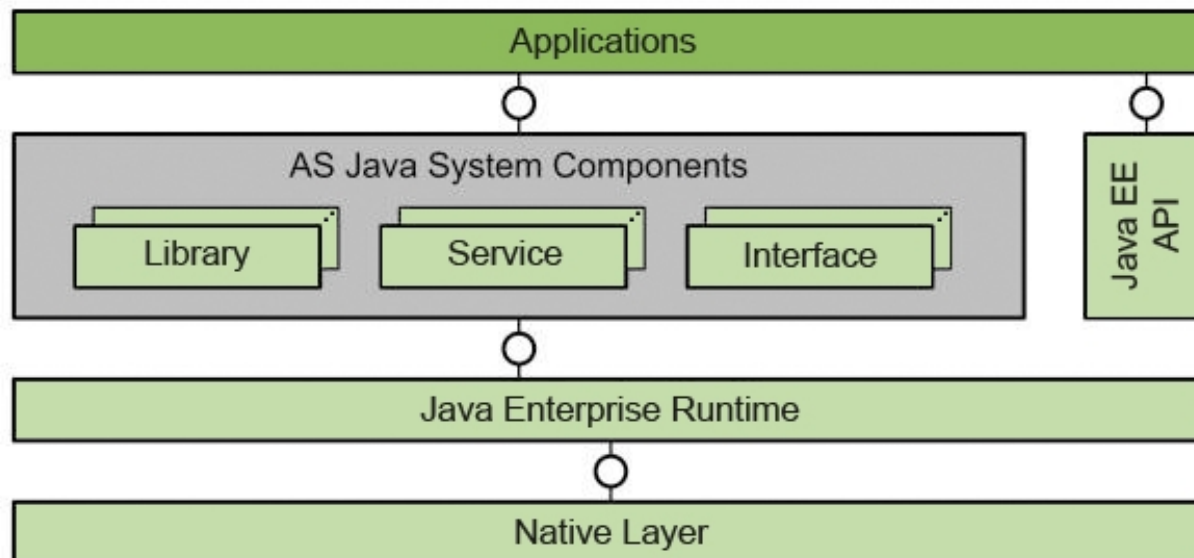


SAP NetWeaver – overview

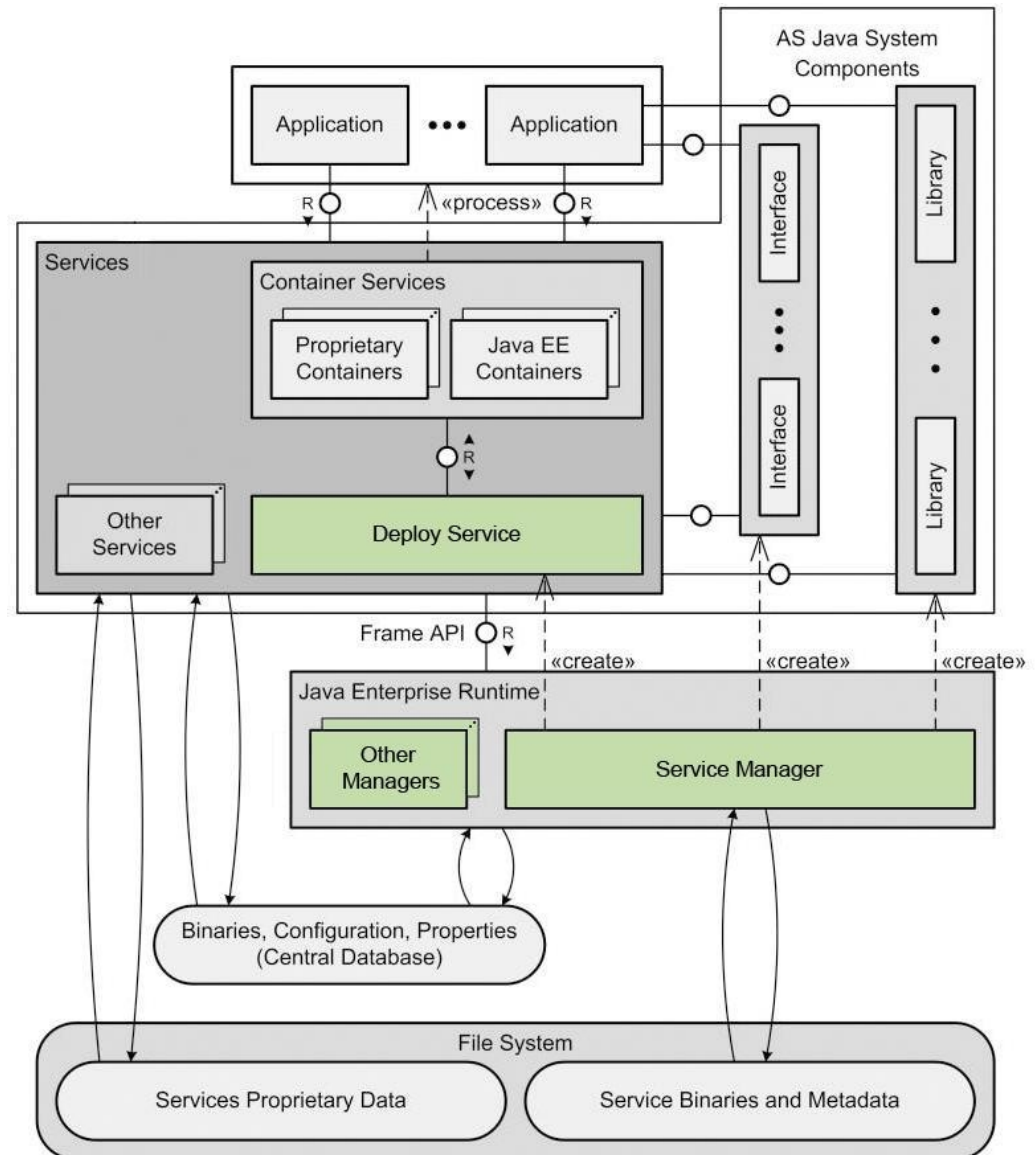
Intrinsic capabilities

- Clustering
- Life-cycle
- Isolation
- Monitoring

Extensibility of platform capabilities



Runtime view



~24 Million Lines of Java code
and 255k number of classes

~5000 applications deployed
on server

SAP NetWeaver over OSGi

Project roadmap

Goals

- Define a migration path to OSGi for the Java Server
- Provide a compatibility layer (passive as much as possible), which shields non-migrated components

Out of scope

- Re-componentization of the whole java stack
- Ready-to-use installation

Time Frame

- September, 2008 – January, 2009

Planned effort

- 7 Developers

Challenges

- Dependency and risk management

Migration path

Verify backwards compatibility

Compatibility layer design

Configuration of migrated components

Server bootstrapping

Server initialization

Build & test infrastructure

Component repackaging concept

Compatibility layer

Service Manager

- Perhaps the most obvious

Class Loading

- Perhaps the most dangerous

Event Infrastructure

- Perhaps the most underestimated

Runtime aspects of automatic repackaging

Properties / Configuration

Work directory / File structure

Monitoring / Administration

Replaced by DS

Special considerations:

- NW interfaces registered in OSGi after provider service appear
- Mapping between component names

Compatibility layer

Service Manager

- Perhaps the most obvious

Class Loading

- Perhaps the most dangerous

Event Infrastructure

- Perhaps the most underestimated

Runtime aspects of automatic repackaging

Properties / Configuration

Work directory / File structure

Monitoring / Administration

Delegation model

Loader names

Component information in loaders

Direct file access

Remote class loading

Verbose exception messages

Application to service references

Context class loader

Compatibility layer

Service Manager

- Perhaps the most obvious

Class Loading

- Perhaps the most dangerous

Event Infrastructure

- Perhaps the most underestimated

Runtime aspects of automatic repackaging

Properties / Configuration

Work directory / File structure

Monitoring / Administration

Modeled by OSGi events

Special considerations:

- Container started
- Service started / not started
- Interface available / not available
- Component loaded / unloaded

Event dispatching

Compatibility layer

Service Manager

- Perhaps the most obvious

Class Loading

- Perhaps the most dangerous

Event Infrastructure

- Perhaps the most underestimated

Runtime aspects of automatic repackaging

Properties / Configuration

Work directory / File structure

Monitoring / Administration

Code analysis

Special considerations:

- Component dependencies

Compatibility layer

Service Manager

- Perhaps the most obvious

Class Loading

- Perhaps the most dangerous

Event Infrastructure

- Perhaps the most underestimated

Runtime aspects of automatic repackaging

Properties / Configuration

Work directory / File structure

Monitoring / Administration

Different mechanisms

Nested properties

Compatibility layer

Service Manager

- Perhaps the most obvious

Class Loading

- Perhaps the most dangerous

Event Infrastructure

- Perhaps the most underestimated

Runtime aspects of automatic repackaging

Properties / Configuration

Work directory / File structure

Monitoring / Administration

Compatibility layer

Service Manager

- Perhaps the most obvious

Class Loading

- Perhaps the most dangerous

Event Infrastructure

- Perhaps the most underestimated

Runtime aspects of automatic repackaging

Properties / Configuration

Work directory / File structure

Monitoring / Administration



OSGi container inside SAP NetWeaver

Enable bundle deployment on the server

- Ability to run bundles
- Add value to the bundle running environment by exposing non-functional features
 - Supportability
 - Administration

Additionally enable “extension points” using bundles





Alternatives

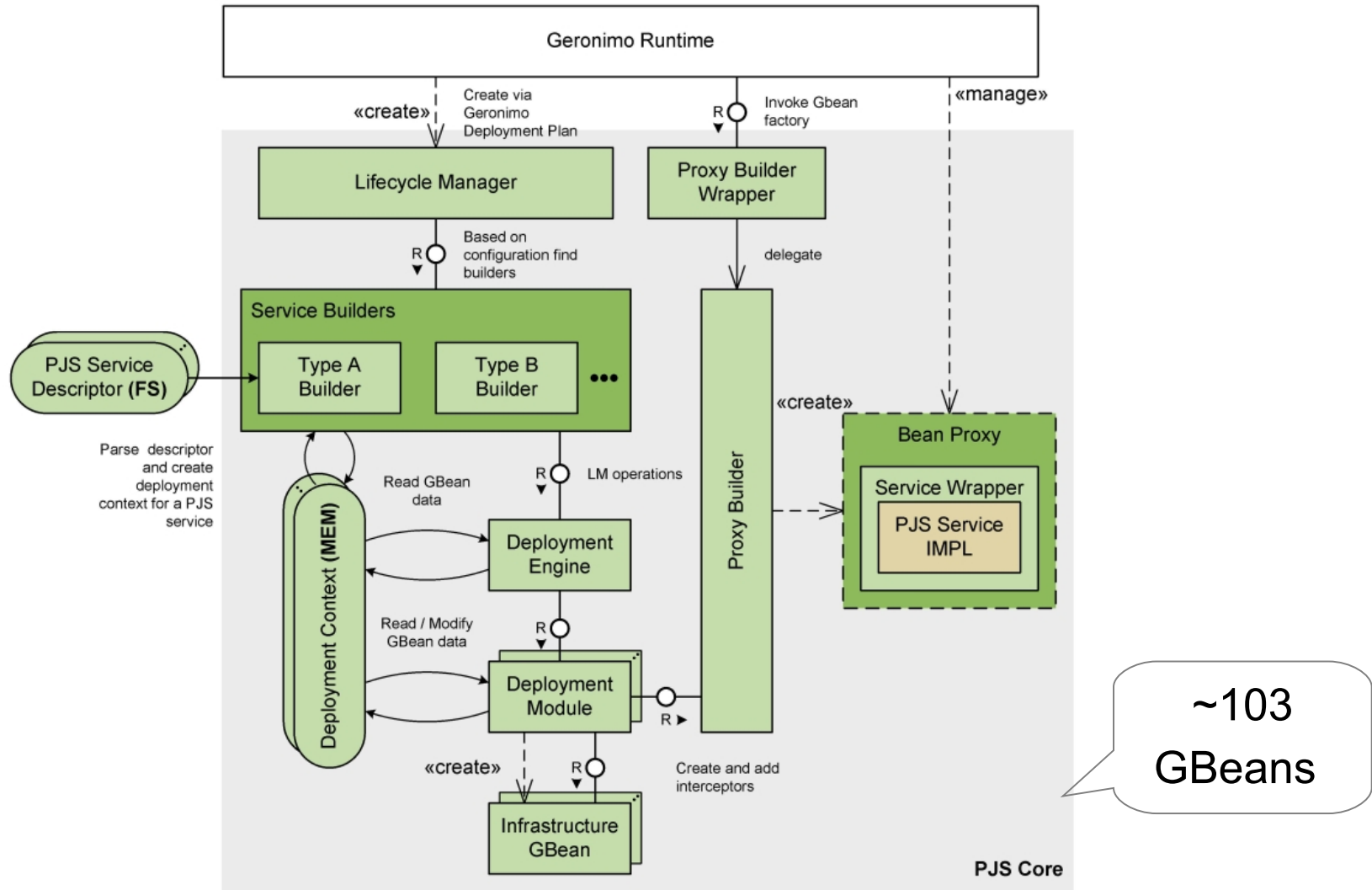
Geronimo

- is built around an IoC kernel and deals with GBeans
- manages GBean dependencies, state, and lifecycle
- kernel injects dependencies into the GBean at runtime according to rules defined in a deployment plan.

OSGi

- provides a publish/find/bind service model
- simplify service registration and handling service dependencies (DS)
- Use simple xml descriptor to define service prerequisites (DS)

Runtime view



Platform Java Services over OSGi

Project roadmap

Goals

- Replace low-level runtime/framework with OSGi
- Remove Geronimo from the stack

Out of scope

- Introducing Java EE capabilities
- Installation, update, upgrade tools
- Changing existing startup mechanism

Time Frame

- April, 2009 – September, 2009

Planned effort

- 7 Developers

Challenges

- Dependency and risk management

Migration path



Migration of the rest of PJS components

Supportability

Component configuration

Migration of one type of PJS components

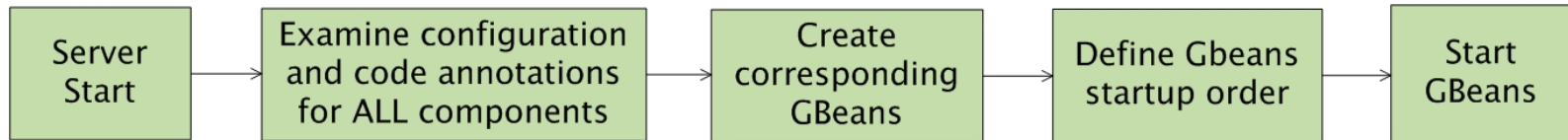
Server bootstrapping

Class loading: design

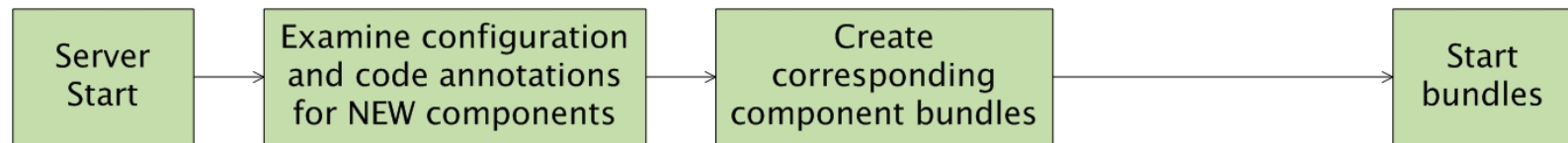
Build & test infrastructure

Startup

Geronimo:

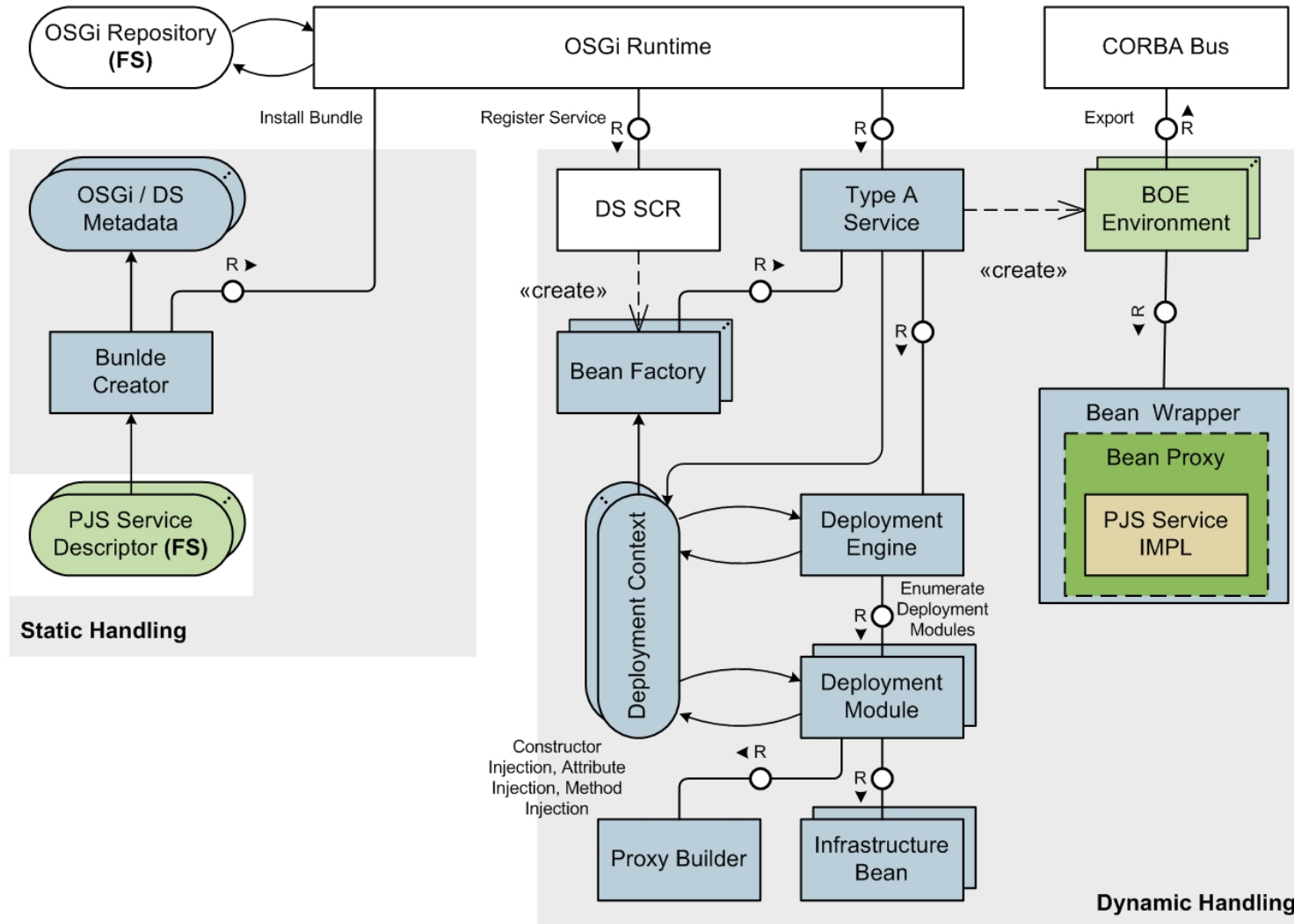


OSGi:



KPI	Result Geronimo	Result OSGi
Used Heap	16.59M (16993K)	12.14M (12144K)
Used Perm	25.21M (25811K)	20.79M (21291K)
Startup Time	16 sec (16583 ms)	6 sec (5874 ms)
Stop Time	1 sec (1148 ms)	1 sec (1146 ms)

Switch static to dynamic handling





Conclusion

Prerequisites for migration to OSGi	
Favourable	Unfavourable
IoC component model	Component model based on API contract
Less environment dependencies	Significant environment dependencies
No adoption required	Adoption required



Thank you

Katya Todorova
SAP Labs Bulgaria

katya.todorova@sap.com

