

Gerd Wütherich ♦ Freelancing software architect

# Spring Dynamic Modules by example

# Who am I?

- » Gerd Wütherich – [gerd@gerd-wuetherich.de](mailto:gerd@gerd-wuetherich.de)
- » Freelancing software architect, Hamburg, Germany
- » With a focus on:
  - » Large Java-based enterprise applications
  - » JEE, Spring, OSGi
- » Frequent speaker at conferences
- » Author of articles and some books

# Die OSGi Service Platform



- » Detailed introduction into OSGi
- » April 2008, dpunkt.verlag
- » ISBN 978-3-89864-457-0
- » Already available
  
- » (but German only at the moment...)

# Pro Spring DM (techn. Reviewer)



- » Written by D. Rubio
- » Detailed introduction into Spring DM
- » February 12, 2009, Apress
- » ISBN 978-1-4302-1612-4

# Agenda

## **OSGi:**

- » Overview of OSGi™ technology
- » A short example

## **Spring Dynamic Modules:**

- » Overview of Spring DM
- » A short example

## **Web Support:**

- » Overview of web support in Spring DM
- » A short example

# OSG – What?

- » The OSGi Service Platform...
  - » ... is a dynamic module system for Java.
  - » ... allows the integration and the management of modules (*Bundles*) and services.
- » Bundles and services can be *installed, started, stopped* and *uninstalled* at runtime.
- » Consist of:
  - » OSGi Framework (Container for bundles and services)
  - » OSGi Standard Services

# Where does the OSGi Service Platform come from?

## *The OSGi Alliance:*

- » <http://www.osgi.org/>
- » Founded in 1999
- » Currently more than 80 members:
  - » Deutsche Telekom, Eclipse Foundation, IBM, Oracle, Prosys, SAP, Siemens, Sun, SpringSource etc.
- » Specifies and supports the OSGi Service Platform

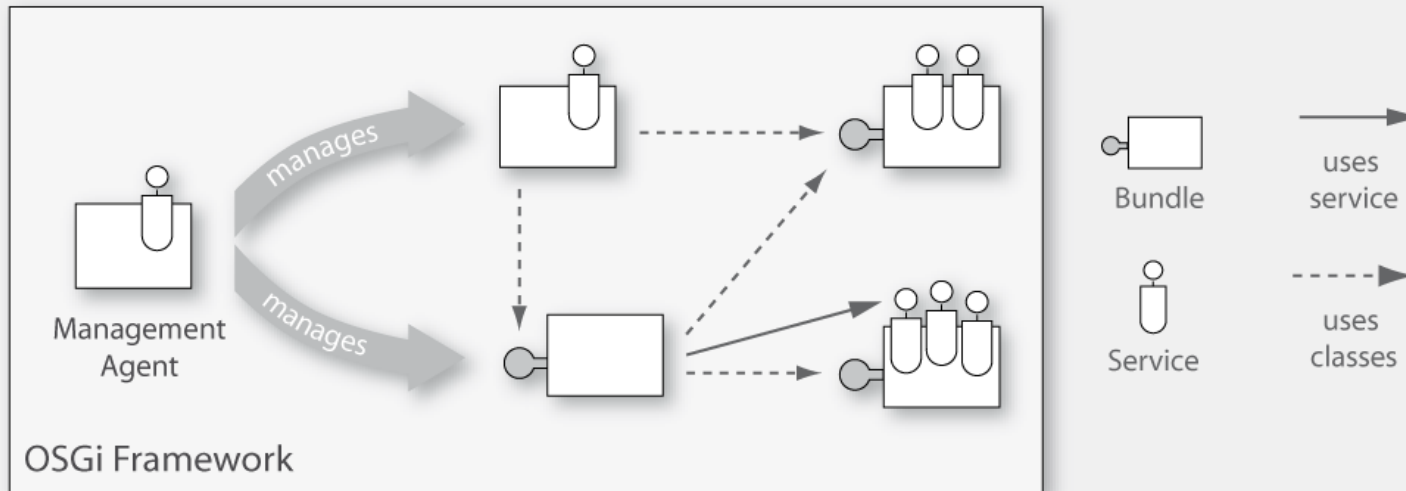


# Where is the OSGi Service Platform used?

## *Some examples:*

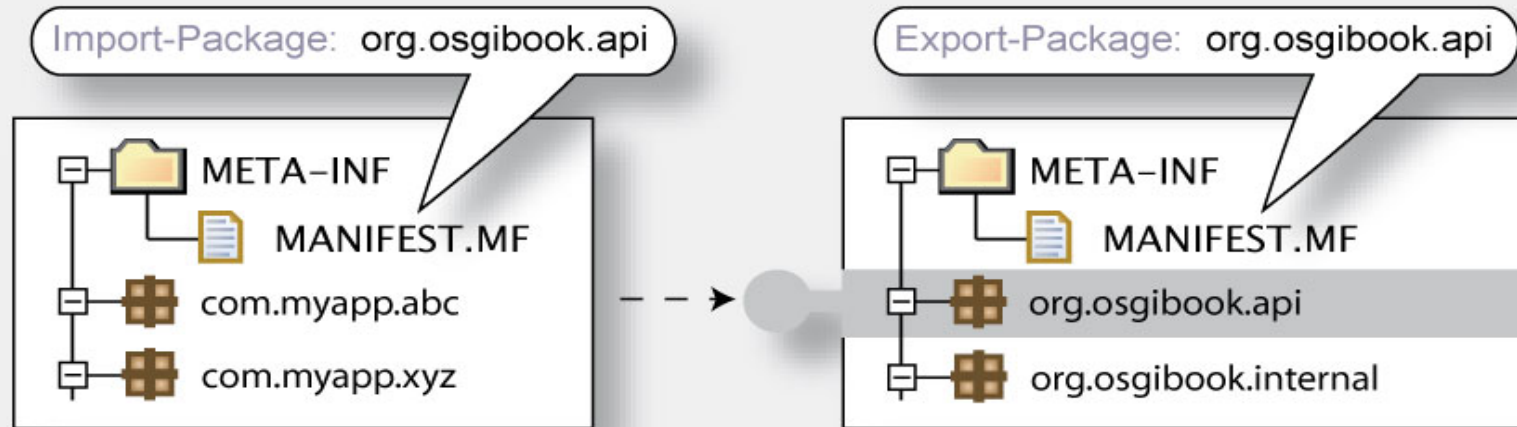
- » Eclipse Platform:
  - » Eclipse SDKs (IDEs), RCP, eRCP, ...
- » IBM
  - » Websphere App Server (based on OSGi)
  - » Lotus
  - » Jazz
- » BEA/Oracle
- » SpringSource Application Platform / dm Server
- » Adobe
- » ...

# Overview of the OSGi Framework



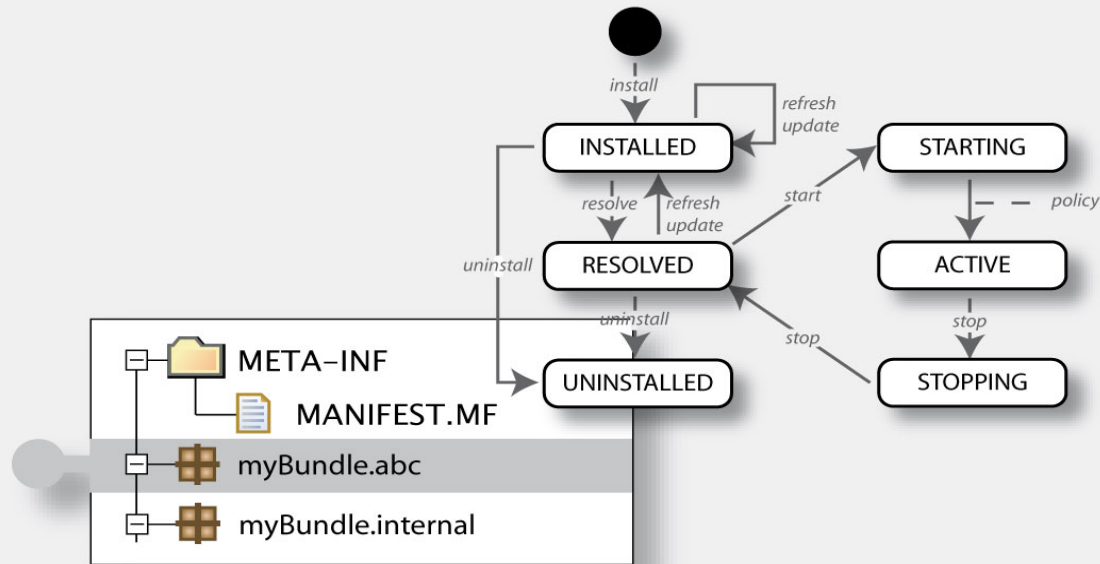
- » Core of the OSGi Service Platform
- » Allows the installation and the administration of bundles and services
- » Manages dependencies between bundles
- » Managed through management agents

# The OSGi Framework: Bundles



- » The framework allows the definition of
  - » modules (so called bundles),
  - » visibility of module elements (public-API vs. private-API)
  - » dependencies between module, and
  - » versioning of modules.

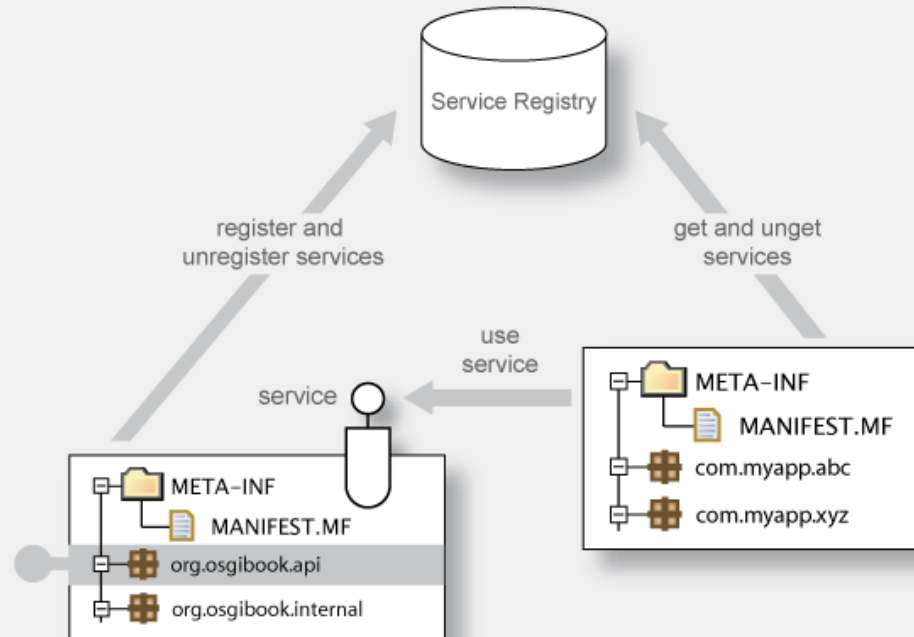
# The OSGi Framework: Bundle lifecycle



» The OSGi framework is dynamic:

- » Bundles can be installed, started, stopped and updated dynamically at runtime.
- » Bundles can execute code during start and stop via the Bundle-Activator.

# The OSGi Framework: Services



- » The OSGi Framework is service oriented:
  - » Services can be registered and unregistered at runtime with the service registry
  - » Bundles can get and unget service from the service registry

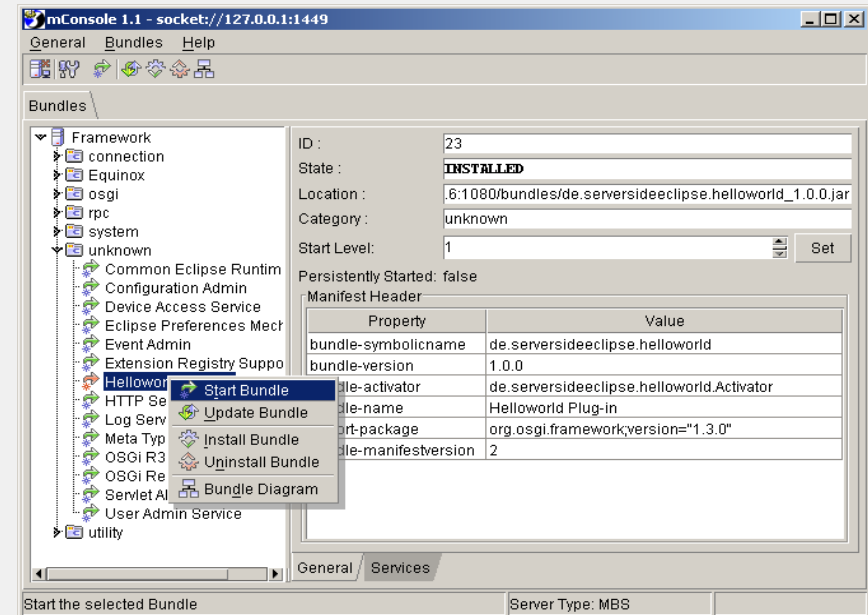
# OSGi Framework: Management Agents I

- » Management Agents allow the administration of OSGi Frameworks.
- » Many different implementations available:
  - » Command-based console (e.g. equinox console)
  - » Interactive GUI applications (e.g. Knopflerfish Desktop, Prosyst mConsole)
  - » Web based applications (e.g. Knopflerfish Web-Konsole)
- » Not standardised, but the framework is accessible through well defined interfaces.

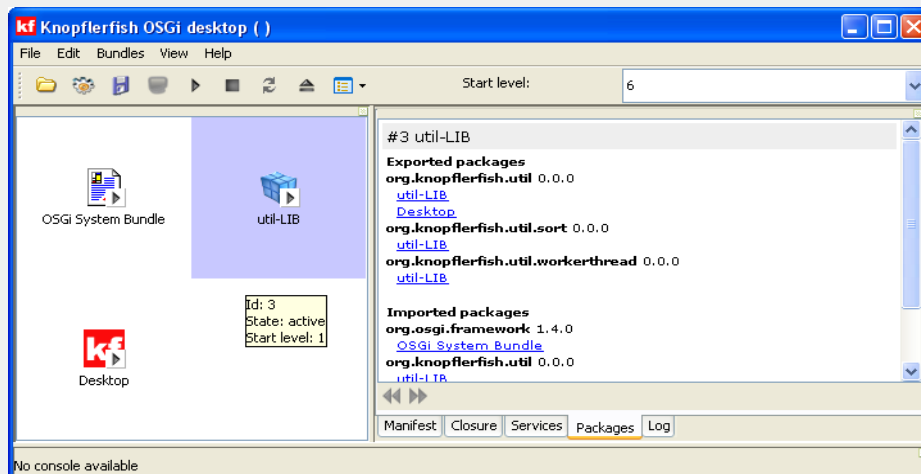
# OSGi Framework: Management Agents II

```
C:\WINDOWS\system32\cmd.exe - java -jar plugins/org.eclipse.osgi_3.3.0.v20070530.jar -console
C:\equinox>java -jar plugins/org.eclipse.osgi_3.3.0.v20070530.jar -console
osgi> ss
Framework is launched.
id      State      Bundle
0       ACTIVE    org.eclipse.osgi_3.3.0.v20070530
osgi> install file:/c:/bundles/de.serversideeclipse.helloworld_1.0.0.jar
Bundle id is 1
osgi> ss
Framework is launched.
id      State      Bundle
0       ACTIVE    org.eclipse.osgi_3.3.0.v20070530
1       INSTALLED de.serversideeclipse.helloworld_1.0.0
osgi> start 1
```

*Eclipse Equinox Console*



*Prosyst mConsole*

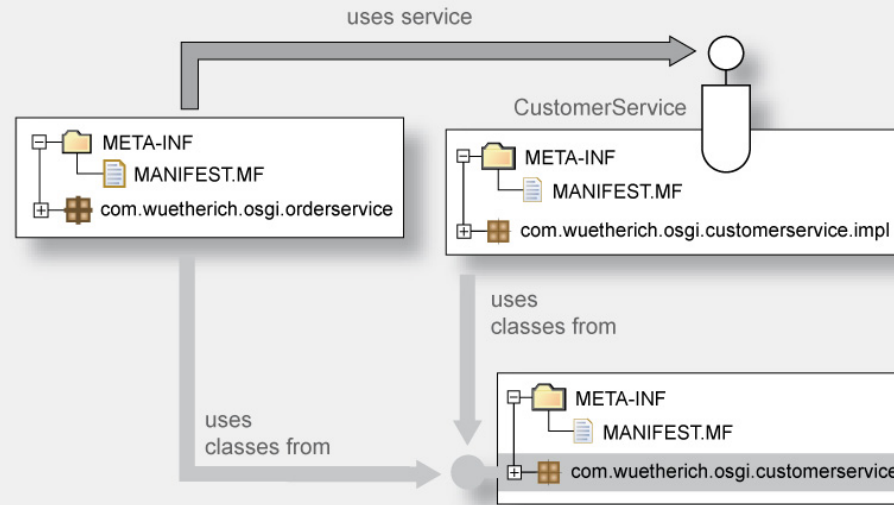


*Knopflerfish Desktop*

# Implementations of the OSGi Service Platform

- » Open Source implementation:
  - » Eclipse Equinox (<http://www.eclipse.org/equinox/>)
  - » Apache Felix (<http://cwiki.apache.org/FELIX/index.html>)
  - » Knopflerfish (<http://www.knopflerfish.org/>)
  - » ProSyst mBedded Server Equinox Edition ([http://www.prosyst.com/products/osgi se equi ed.html](http://www.prosyst.com/products/osgi_se_equi_ed.html))
  
- » Commercial implementations:
  - » ProSyst (<http://www.prosyst.com/>)
  - » Knopflerfish Pro (<http://www.gatespacetelematics.com/>)
  - » ...

# The OSGi Service Platform: An example



## The Order Service...

- » ... can be used to place a stock order for a customer.
- » ... is accessible via RMI.

## The Customer Service...

- » ... can be used to retrieve customer information.

# Intermediate result

- » The OSGi Service Platform allows you to unitze applications
- » Modules can be installed, started, stopped and uninstalled at runtime.

## **But:**

- » Dependent on OSGi framework libraries
- » Services must be managed programmatically
- » No support for „enterprise technologies“

# Spring Dynamic Modules for OSGi Service Platforms

- » Formerly known as „Spring-OSGi“
- » An open source project in the Spring portfolio
  - » led by SpringSource
  - » committers from BEA and Oracle
- » <http://www.springframework.org/osgi>
- » Allows you to implement Spring Applications on top of an OSGi framework
- » Works with Equinox, Felix, Knopflerfish

# Project Objectives

» Bring the benefits of OSGi:

» modularity

» versioning

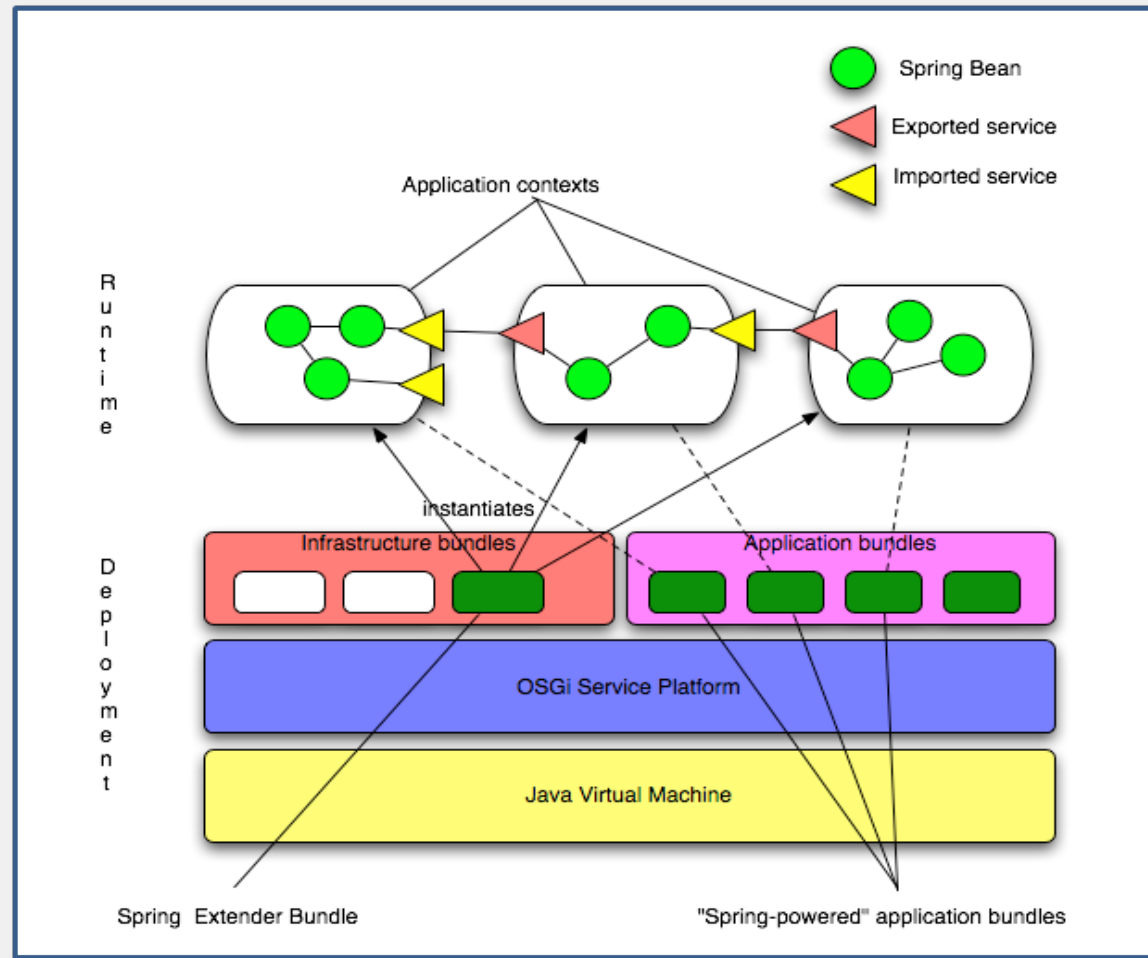
» lifecycle support

to Spring-based enterprise application development

# Challenges

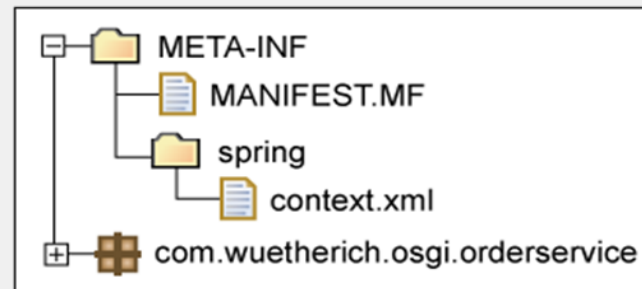
- » Complexity
  - » Bring the simplicity of POJOs to OSGi
- » Integrate with OSGi service model
  - » Spring beans  $\Leftrightarrow$  OSGi services
- » Testing
  - » Enable testing without OSGi container
  - » No dependencies on OSGi APIs
- » Allow people to use all the Spring technology abstractions

# The Spring OSGi Model

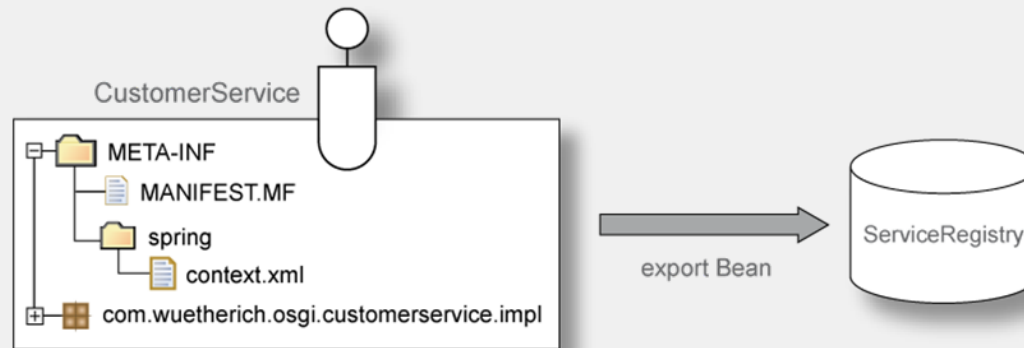


# Bundles and Spring

- » One application context per bundle:
  - » Spring DM creates and destroys the application context when the bundle is started or stopped
  - » META-INF/spring/\*.xml or Spring-Context header in MANIFEST.MF



# Beans as OSGi Services (1)



- » Spring Beans can be exported as OSGi Services
- » OSGi Service are visible across bundle boundaries

```
<beans>
  <bean name="customerService"
    class="com.wuetherich.osgi.customerservice.impl.CustomerServiceImpl"/>

  <osgi:service id="customerServiceOsgi"
    ref="customerService"
    interface="com.wuetherich.osgi.customerservice.CustomerService"/>
</beans>
```

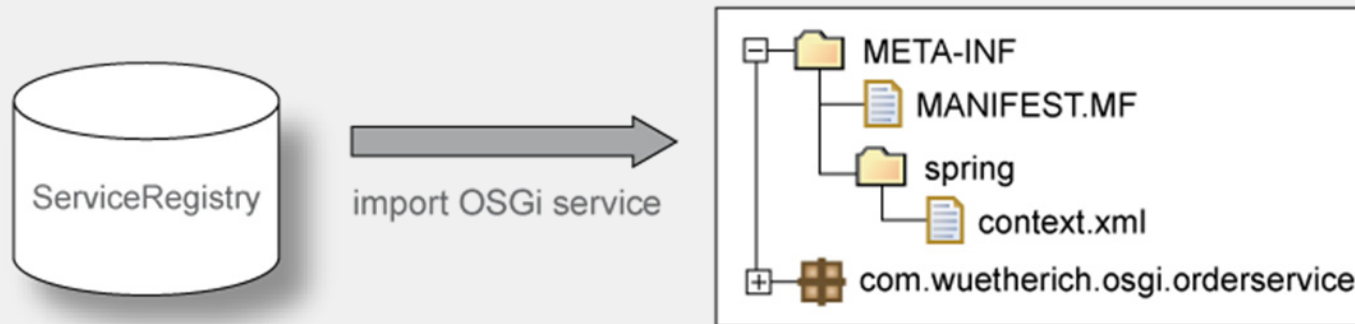
# Beans as OSGi Services (2)

» Possible element attributes for `<osgi:service>`:

Attributname	Mögliche Werte
interface	qualifizierter Klassenname
ref	Bean-Name
context-class-loader	unmanaged / service-provider
auto-export	disabled / interfaces / class-hierarchy / all-classes
ranking	Integer-Wert

» OSGi Services are beans (= object reference on the bean)

# OSGi services as beans (1)



- » Import existing OSGi services as beans into Spring's application context

```
<beans>
  <osgi:reference id="customerServiceOsgi"
    interface="com.wuetherich.osgi.customerservice.CustomerService"/>

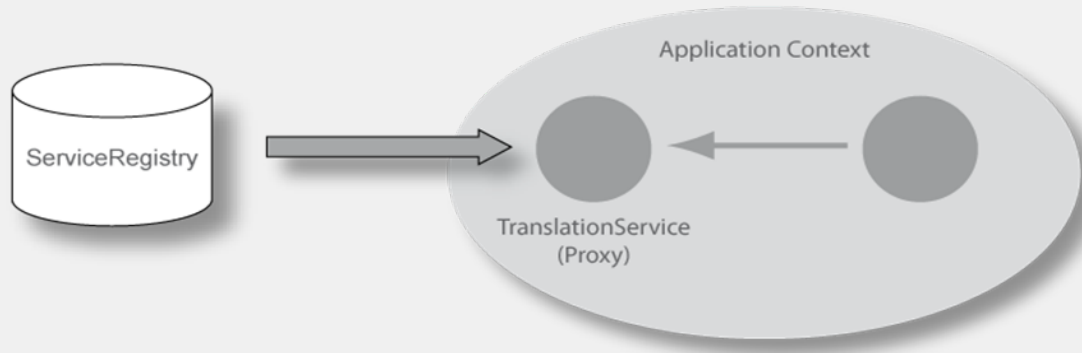
  <bean id="orderService"
    class="com.wuetherich.osgi.orderservice.internal.OrderServiceImpl">
    <property name="customerService">
      <ref local="customerServiceOsgi"/>
    </property>
  </bean>
</beans>
```

# OSGi Services als Beans (2)

» Possible element attributes for `<osgi:reference>`:

Attributname	Mögliche Werte
interface	qualifizierter Klassenname
filter	OSGi-Filter-Ausdruck
bean-name	String
context-class-loader	client / service-provider / unmanaged
cardinality	0..1 / 1..1
timeout	Long (positiv)

# OSGi Services als Beans (3)

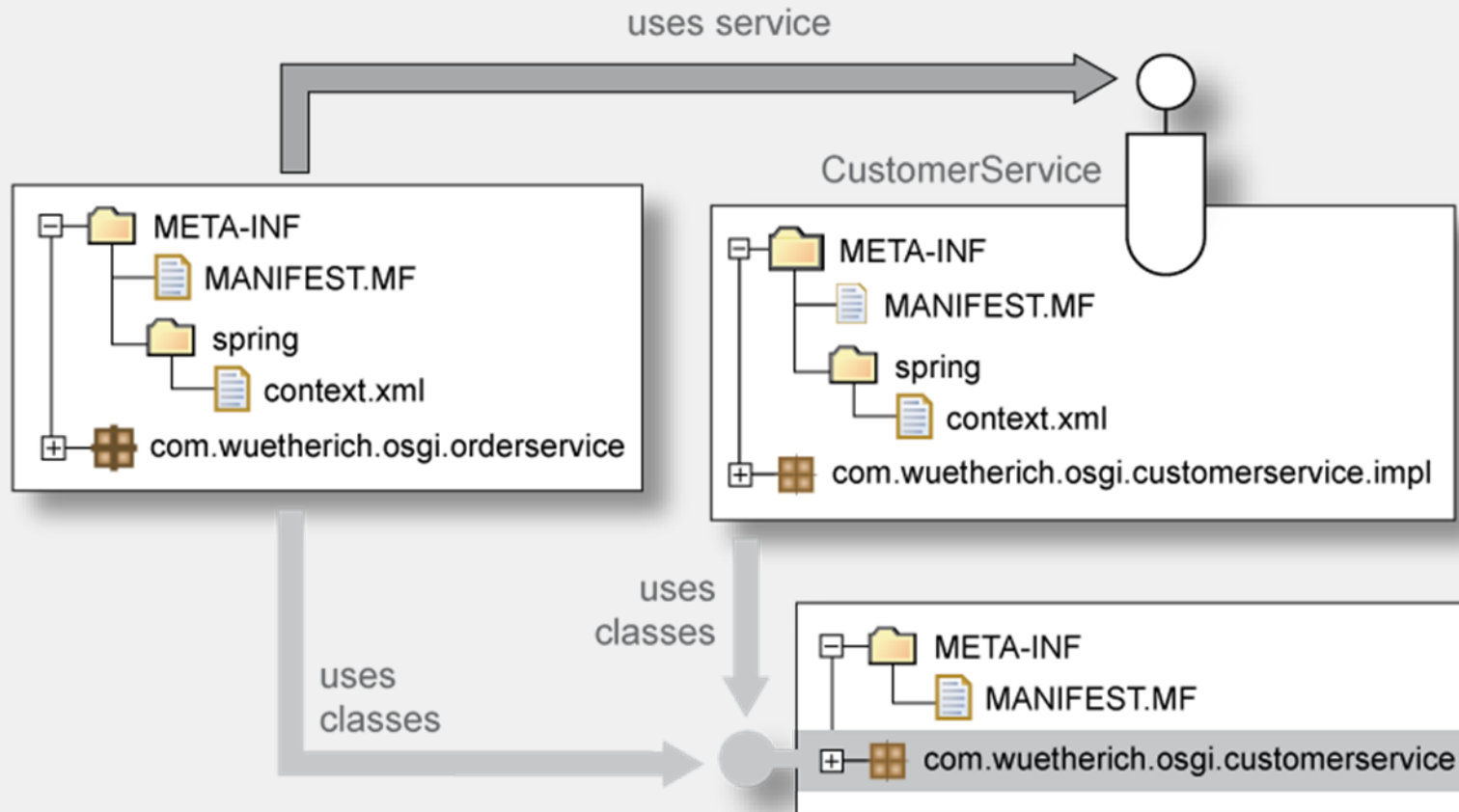


- » Imported services are proxies:
  - » The defined `<osgi:reference>` bean remains unchanged during the lifetime of the application context.
  - » The service it is based on can come and go.

# Further possibilities

- » `<osgi:list>`, `<osgi:set>`
  - » Allows you to work with sets of services
  - » Dynamic iterator
  - » greedy-proxying
- » `<osgi:bundle>`
  - » Presents a bundle object as a bean
- » `<osgix:property-placeholder>`
  - » Allows you to use configurations by the Configuration Admin Service
- » Support for annotations

# Spring DM: An example



# Result

- » No OSGi API necessary
  - » Spring philosophy
- » Bean visibility between bundles can be defined
  - » Only beans that should be visible to other bundles are exported as OSGi services
- » Dependency injection across bundle boundaries

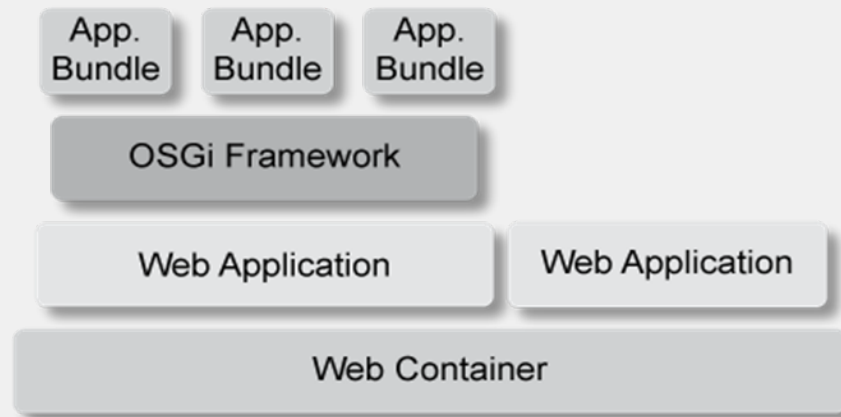
# Spring DM web support

- » Focus in Spring DM 1.1
- » Integration of Spring DM with the existing Spring web support

## Web applications in OSGi

- » In general, different deployment scenarios are possible

# OSGi based web applications 1

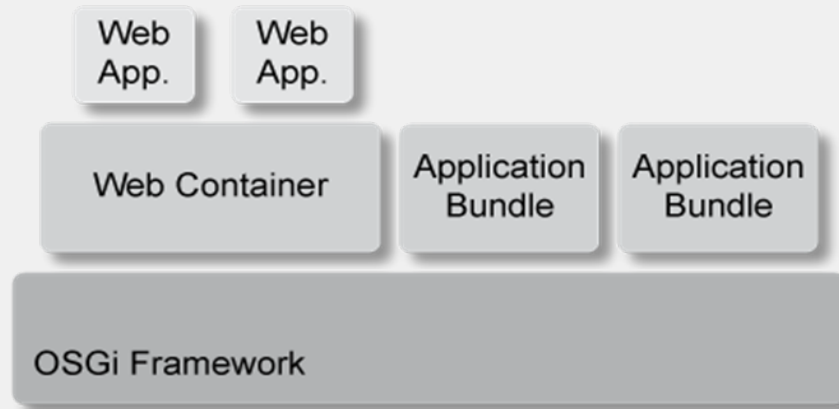


## Scenario 1:

### OSGi Framework within a web application

- » OSGi Framework is executed within the web application
- » Application bundles can be installed and started through the OSGi framework within the web application

# OSGi based web applications 2

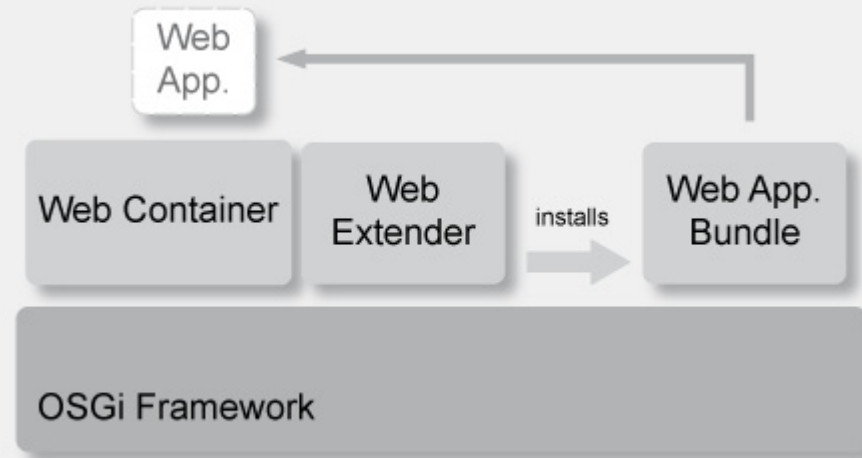


## Scenario 2:

### Web Container embedded in OSGi Framework

- » Web Container is installed and started as a bundle within the OSGi framework
- » Application bundles can deploy web applications programmatically

# Web support in Spring DM



- » Web applications are installed and started as bundles in the OSGi framework („bundle-ized“ WAR)
- » Spring DM Web Extender installs the web application in the web container when the war bundle is started
- » „Native“ support for different web containers
- » Full support for OSGi class loading

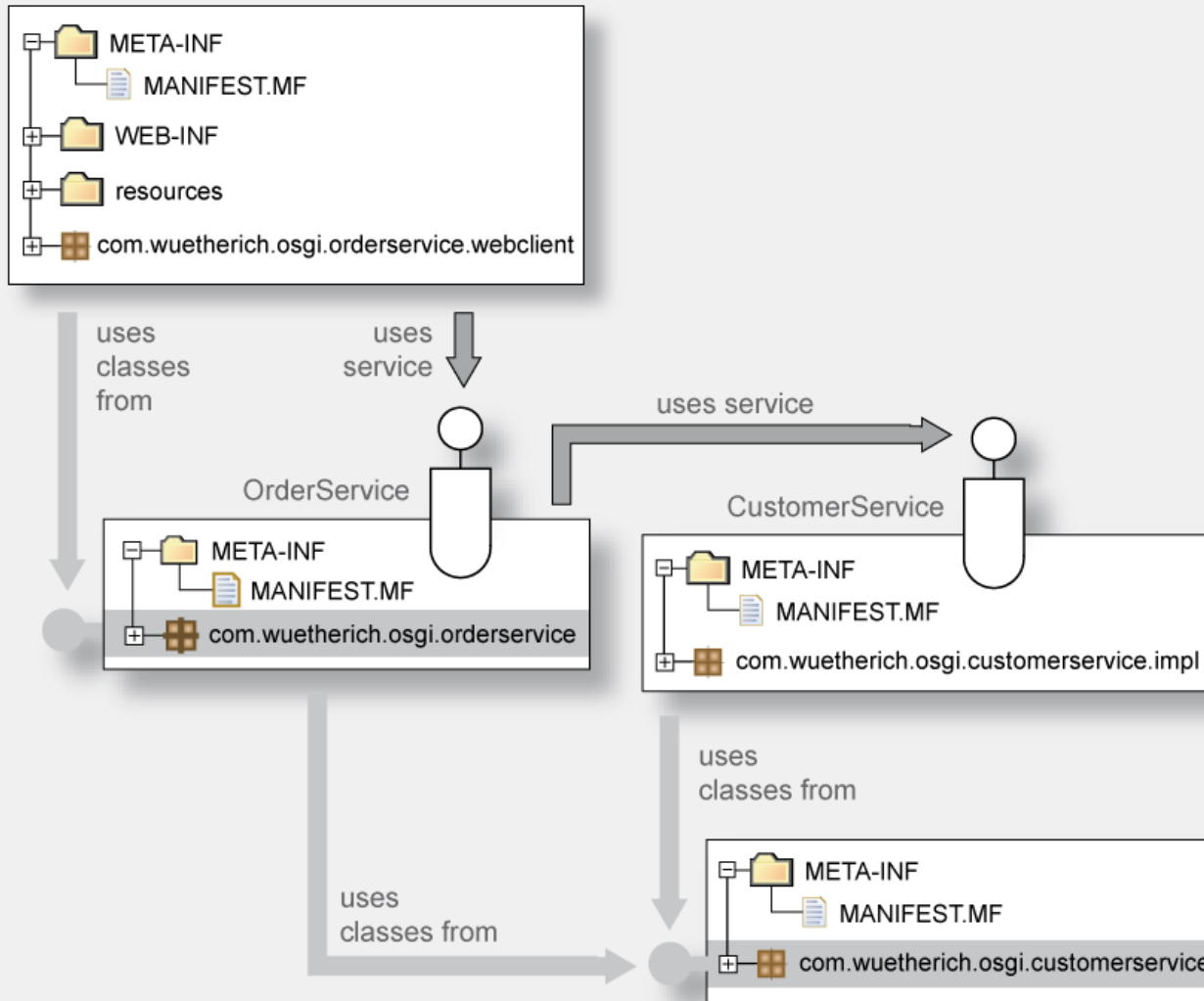
# Web applications as bundles

- » „Regular“ WAR files
- » Additionally: bundle manifest
- » Spring connection through `OsgiBundleXmlWebApplicationContext` and `ContextLoaderListener`
- » Specification in the file `web.xml`:

```
<context-param>
  <param-name>contextClass</param-name>
  <param-value>org.springframework.osgi.web.context.
    support.OsgiBundleXmlWebApplicationContext</param-value>
</context-param>

<listener>
  <listener-class>
    org.springframework.web.context.ContextLoaderListener
  </listener-class>
</listener>
```

# Spring DM Web Support by Example



# Perspective

- » Further information:
  - » <http://www.springframework.org/osgi>
  - » <http://www.springframework.org/osgi/specification>
- » Blueprint Service in OSGi Release 4.2

Thank you!

