Spring & OSGi kombiniert: Plattform der Zukunft

- Gerd Wütherich, freiberuflicher Software-Architekt
- Bernd Kolb, KolbWare
- Martin Lippert, akquinet agile GmbH

gerd-wuetherich.de
Software-Architektur • Software-Entwicklung • Konfigurationsmanagement

KolbW@re//

it-agile

Agenda

- OSGi™ Technologie im Überblick
- Die OSGi™ Service Platform in Action: Ein praktisches Beispiel
- Spring Dynamic Modules
- Spring DM in Action: Ein praktisches Beispiel
- Ein "Real-World-Example": jPetStore mit Spring-DM

OSG – was?

- Die OSGi™ Service Platform:
  - „A dynamic module system for Java"
- Besteht aus:
  - OSGi™ Framework (Container für Bundles und Services)
  - OSGi™ Standard Services (verschiedene, horizontale Dienste)

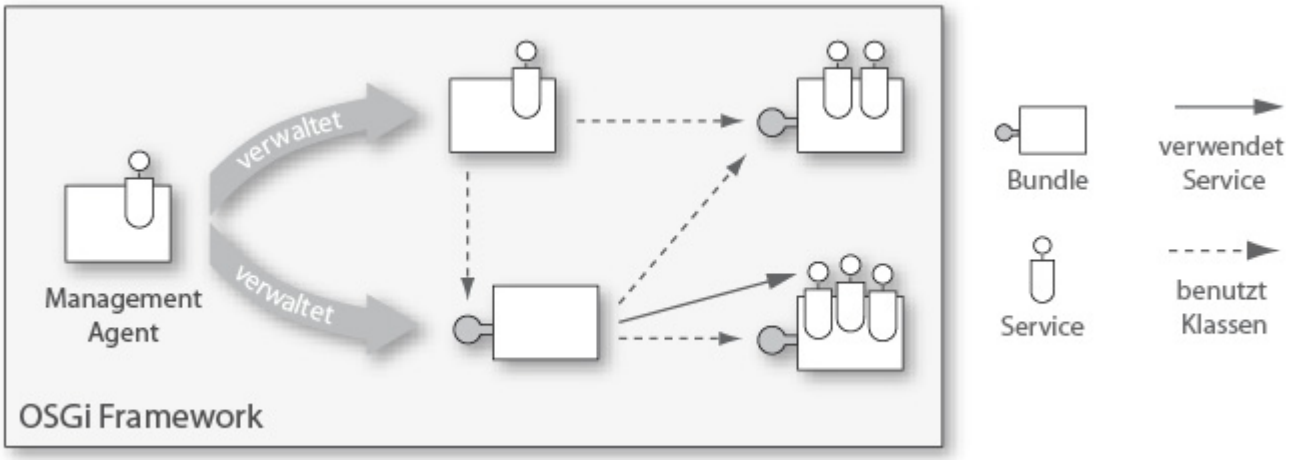
Woher kommt OSGi™ Service Platform?

- OSGi-Alliance:
  - http://www.osgi.org/
- Wird seit 1999 mit einem Fokus auf Leichtgewichtigkeit und Dynamik entwickelt
  - Ursprünglich für Embedded-Systeme
  - Inzwischen verbreitet für Server- und Client-Systeme

Wo wird OSGi™ die Service Platform heute verwendet?

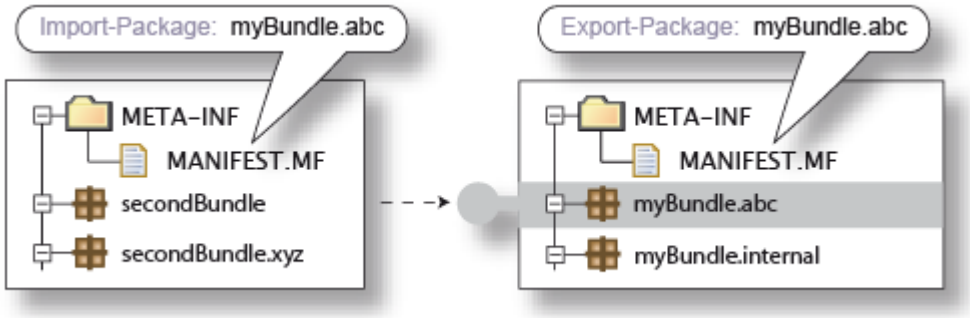
- Die Eclipse-Plattform:
  - Eclipse-SDK (IDEs), Server-Side-Eclipse, eRCP, …
- IBM
  - Websphere App Server 6.1 (basiert auf OSGi)
  - Lotus (basiert auf Eclipse-RCP, damit auch OSGi)
  - Jazz (basiert auf Server-Side-Eclipse)
- BEA hat diverse Systeme basierend auf der OSGi-Plattform
- Oracle interessiert, JBoss prototypt OSGi-Umstellung
- Adobe
- …

Das OSGi™ Framework: Überblick



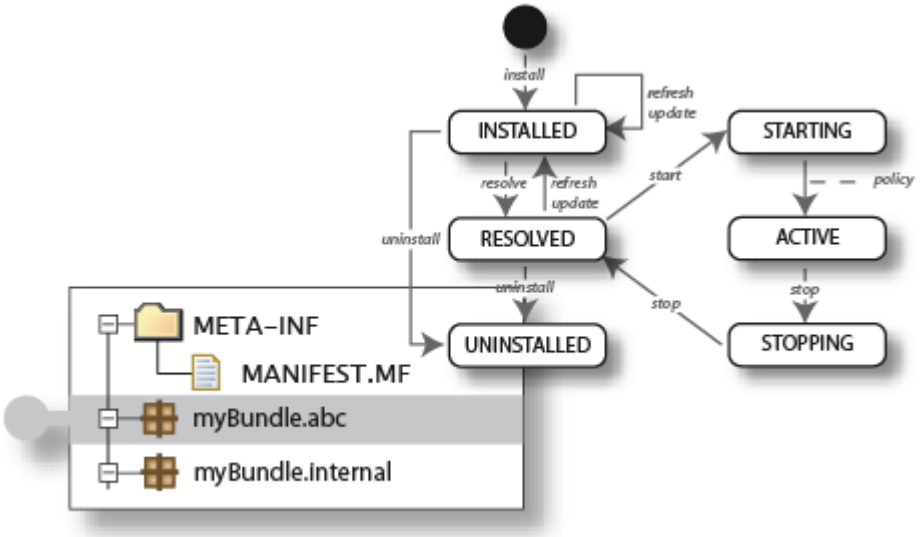
- Kern der OSGi™ Service Platform
- Ermöglicht die Installation und Administration von Bundles und Services
- Verwaltet (Klassen-)Abhängigkeiten zwischen Bundles
- Kann „von außen" administriert werden (Managment Agent)

Das OSGi™ Framework: Bundles



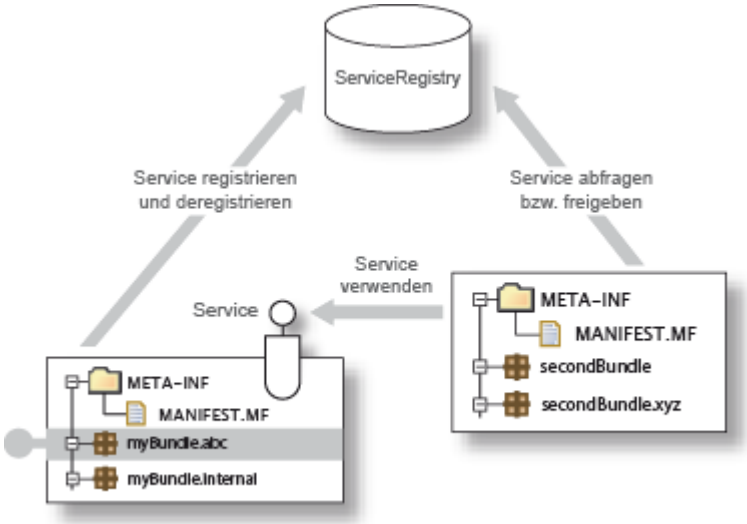
- Das Framework erlaubt die Definition von
  - Modulen (genannt „Bundles"),
  - Sichtbarkeiten von Modul-Bestandteilen (public-API vs. private-API)
  - Abhängigkeiten zwischen Modulen, sowie
  - Versionen von Modulen.

Das OSGi™ Framework: Lebenszyklus von Bundles



- Das OSGi™ Framework ist dynamisch:
  - Bundles können dynamisch zur Laufzeit installiert, gestartet, gestoppt, deinstalliert und aktualisiert werden.
  - Bei Start und Stopp können Bundles über den Bundle-Activator Code ausführen.

Das OSGi™ Framework: Services



- Das OSGi™ Framework ist service-orientiert:
  - Bundles können Services zur Laufzeit über die Service-Registry veröffentlichen und wieder entfernen
  - Bundles können über eine Service-Registry Services finden und verwenden

Das OSGi™ Framework: Management Agents I

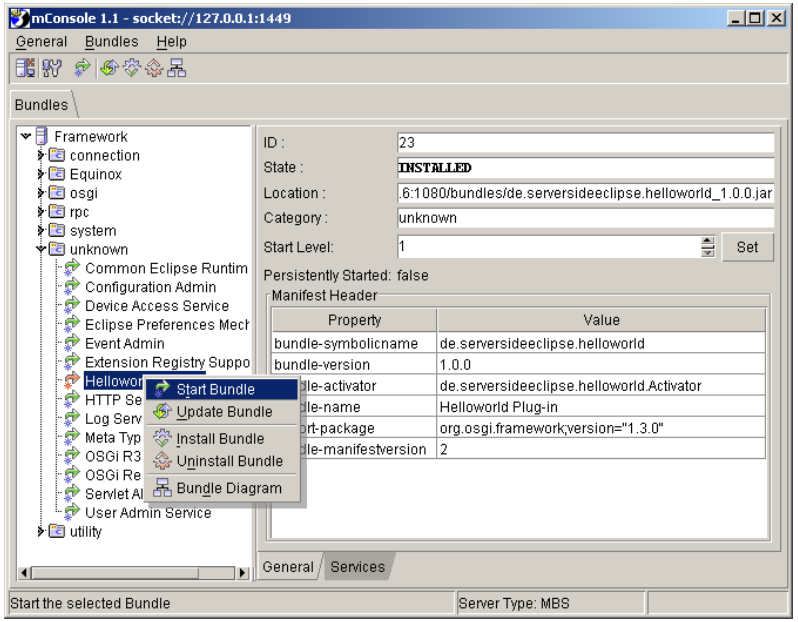
- Management Agents ermöglichen die Administration eines OSGi™ Frameworks.
- Breites Spektrum verfügbar:
  - Kommandobasierte Konsole (Equinox Konsole)
  - Grafisch-interaktive Anwendungen (Knopflerfish Desktop, Prosyst mConsole)
  - Web-basierte Oberfläche (Knopflerfish Web-Konsole)

- Nicht standardisiert
- **Aber:** Zugriff auf das Framework über definierte Schnittstellen, deshalb Framework übergreifend einsetzbar.

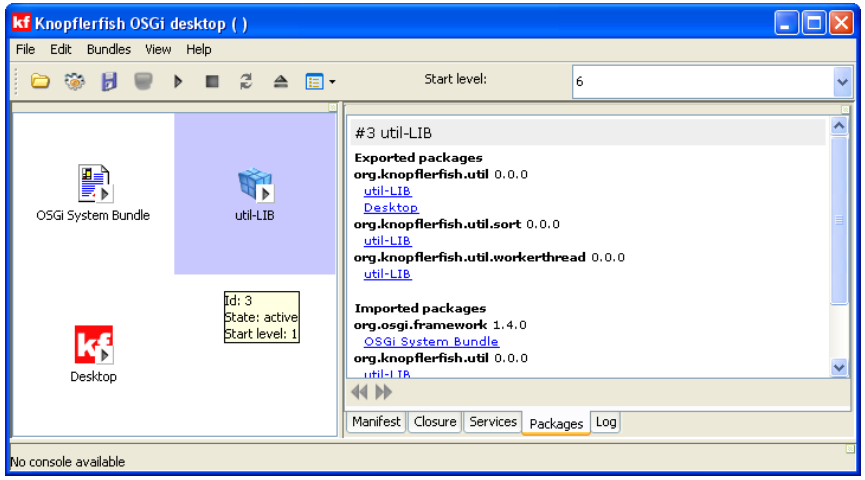
Das OSGi™ Framework: Management Agents II

```
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 Konsole



Prosyst mConsole



Knopferfish Desktop

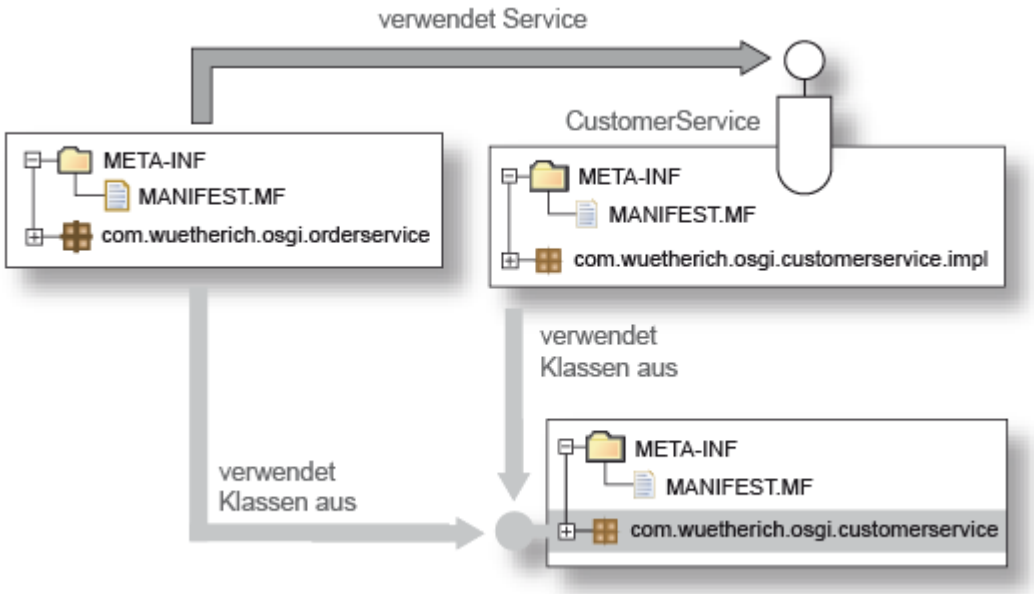
Implementationen der OSGi™ Service Platform

- Open-Source-Implementationen:
  - 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)

- Kommerzielle Implementationen:
  - ProSyst (http://www.prosyst.com/)
  - Knopflerfish Pro (http://www.gatespacetelematics.com/)

(kein Anspruch auf Vollständigkeit)

Die OSGi™ Service Platform in Action: Ein praktisches Beispiel



Der OrderService…

- …ermöglicht die Order von Aktien für einen Kunden
- …ist Remote über RMI verfügbar

Der CustomerService…

- … liefert dem OrderService die passenden Kundendaten

Zwischenbilanz

- OSGi™ Service Platform ermöglicht Modularisierung der Anwendung.
- Module können zur Laufzeit installiert, gestartet, gestoppt, deinstalliert werden.

Aber:

- Abhängigkeit von OSGi™ Framework-Bibliotheken
- Services müssen programmtisch verwaltet werden
- Keine Unterstützung für „Enterprise-Technologien"

Spring Dynamic Modules for OSGi™ Service Platforms

- Formerly known as „ Spring-OSGi“
- Ein neues Mitglied der Spring-Familie
  - http://www.springframework.org/osgi
  - Keine eigene OSGi-Implementation, sondern eine Brücke zwischen Spring und OSGi-Implementationen
  - Erlaubt es, Spring-Anwendungen mit OSGi zu implementieren
  - Ist kompatibel zu Equinox, Felix und Knopflerfish

Die Ausgangsbasis

- Das Spring-Framework
  - Einfaches Programmiermodell (POJOs)
  - Dependency Injection
  - AOP
  - Viele Technologie-Vereinfachungen und -Abstraktionen
- Inzwischen ein de-facto-Standard für JEE-Systeme

Das Ziel

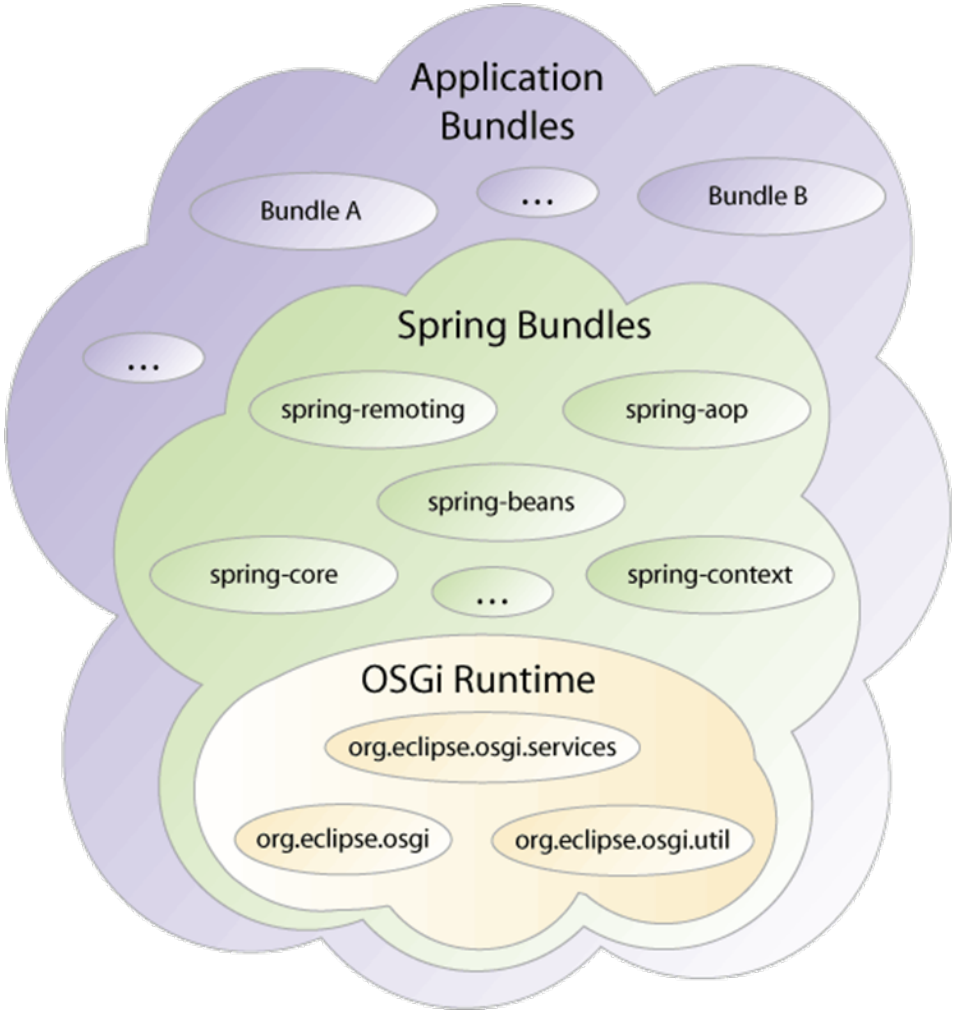
- Die Möglichkeiten von OSGi nutzen!
  - Anwendungen in Bundles aufteilen können, inkl.
    - Dependency Management
    - Sichtbarkeits-Definitionen
    - Versions-Management
    - Dynamic Management
  - Spring selbst in Form von Bundles zur Hand zu haben

Randbedingungen

- Komplexität
  - Darf nicht erhöht werden, weiterhin POJO-Programmiermodell
- OSGi-Service-Modell integrieren
  - OSGi-Services als Beans und umgekehrt
- Testing
  - muss auch außerhalb eines OSGi-Containers möglich sein
  - Möglichst keine Abhängigkeiten zu OSGi-APIs
- Alle Enterprise-Features von Spring müssen weiterhin genauso nutzbar sein

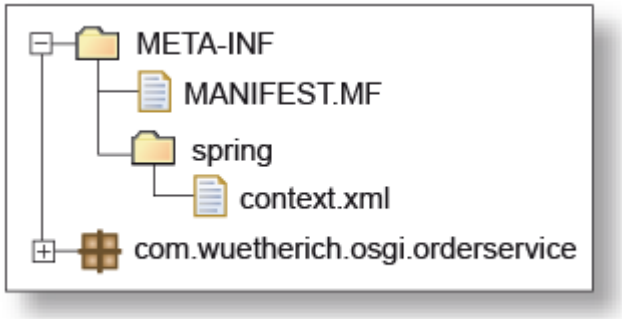
Spring als Bundles

(hier am Beispiel von Equinox)



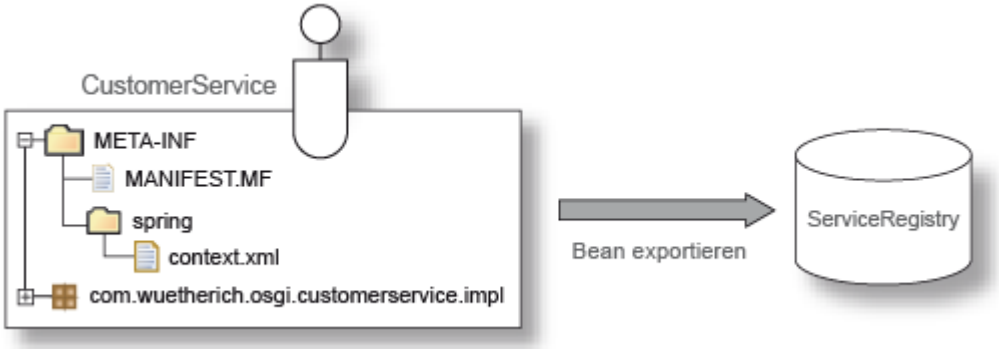
Bundles und Spring

- Ein Application-Context pro Bundle:
  - Spring-OSGi erzeugt und zerstört den Context automatisch, wenn das Bundle aktiviert bzw. deaktiviert wird
  - Definition über XML-Dateien unterhalb von ‚META-INF/spring'



Beans als OSGi-Services

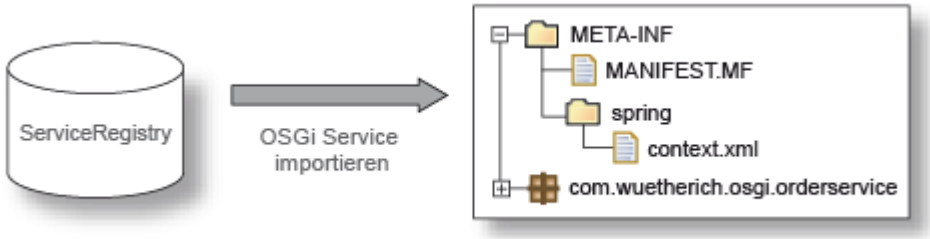
- Spring-Beans können als OSGi-Services exportiert werden
- OSGi-Services sind bundle-übergreifend sichtbar




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

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

OSGi-Services als Beans

- Existierende OSGi-Services können als Beans in den Spring-Context integriert werden

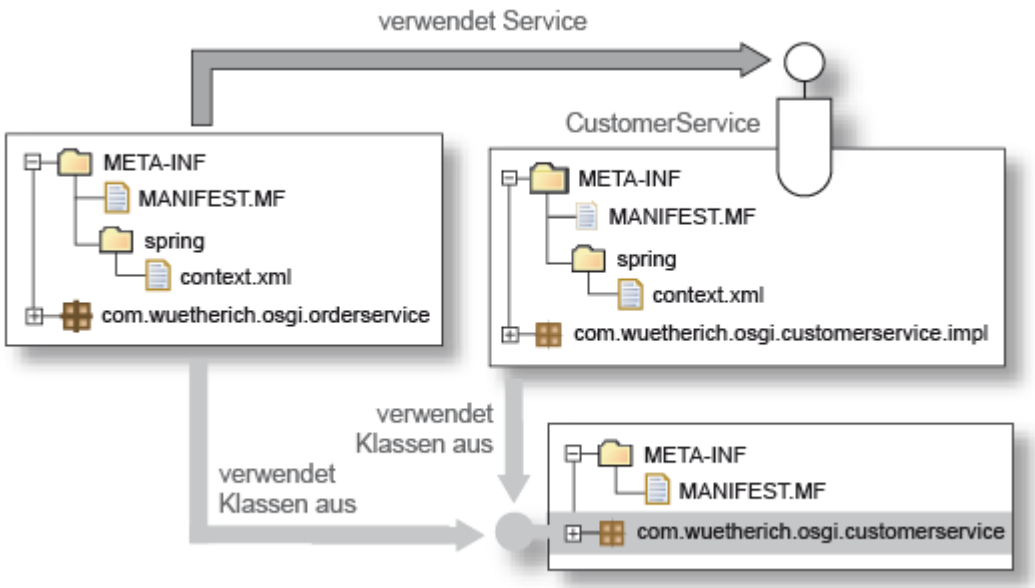



```
<beans>
  <osgi:reference id="customerServiceOsgi"
                  interface=„com.wuetherich.customerservice.CustomerService"/>

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

Weitere Möglichkeiten

- Cardinality
  - Beziehungen zwischen importierten OSGi-Services und der repräsentierenden Bean
    (1..1, 0..1, 1..n, 0..n)
- Service-Listener
  - Information einer Bean über Service-Änderungen
- <osgi:property-placeholder>
  - Liest Properties aus einer Properties-Datei
- <osgi:bundle>
  - Stellt ein Bundle-Objekt als Bean bereit
- <osgi:virtual-bundle>
  - erlaubt die Installation eines JARs als Bundle "on-the-fly"

Spring DM in Action: Ein praktisches Beispiel



Resultat

- Keine OSGi-API nötig
  - (ganz im Spring-Sinne)
- Bean-Sichtbarkeiten zwischen Bundles können definiert werden
  - nur was für andere Bundles sichtbar sein soll, wird als OSGi-Service exportiert
- **Dependency Injection über Bundle-Grenzen hinweg**

Ausblick: Web-Anwendungen

- Standalone-Server-Applikation:
  - Läuft auf Basis von OSGi
  - Enthält beispielsweise Jetty als Http-Service und Servlet-Container
- OSGi innerhalb eines Web- oder App-Servers:
  - OSGi innerhalb eines WAR deployen
  - Servlet-Bridge sorgt dafür, dass Requests an das entsprechende Bundle weitergeleitet werden

Ausblick: Web-Anwendungen

- In beiden Fällen:
  - Anwendung wird aus Bundles zusammengesetzt
  - Spring kann genutzt werden, inkl. Spring-Web-Support oder Remoting oder ähnlichem
- Deployment-Modell ist unabhängig davon
  - Im Entwicklungs-Modus der embedded-Jetty
  - Beim Deployment im Test oder Produktion dann die Bridge

Example V – jPetStore

- jPetStore is a Spring example
- 3 Layers
- iBATIS
- RMI, HTTP, Hessian and Burlap for remoting
- JSPs and Spring-WebMVC

```
INFO: Publishing service under classes [(de.kolbware.jpetstore.domain.logic.validator.OrderVa
14.06.2007 14:31:14 org.springframework.beans.factory.support.DefaultListableBeanFactory preI
INFO: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultListab
14.06.2007 14:31:14 org.springframework.osgi.context.support.AbstractRefreshableOsgiBundleApp
INFO: Publishing application context with properties (org.springframework.context.service.name
14.06.2007 14:31:14 org.mortbay.jetty.servlet.ServletHandler$Context log
INFO: Initializing Spring root WebApplicationContext
14.06.2007 14:31:14 org.springframework.web.context.ContextLoader initWebApplicationContext
INFO: Root WebApplicationContext: initialization started
14.06.2007 14:31:14 org.springframework.web.context.ContextLoader initWebApplicationContext
INFO: Root WebApplicationContext: initialization completed in 0 ms
14.06.2007 14:31:14 org.mortbay.jetty.servlet.ServletHandler$Context log
INFO: Initializing Spring FrameworkServlet 'petstore'
14.06.2007 14:31:14 org.springframework.web.servlet.FrameworkServlet initServletBean
INFO: FrameworkServlet 'petstore': initialization started
14.06.2007 14:31:14 org.springframework.context.support.AbstractApplicationContext prepareRef
INFO: Refreshing de.kolbware.springOsgiUtil.context.OSGiXmlWebApplicationContext@6197cc: disp
14.06.2007 14:31:14 org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDef
INFO: Loading XML bean definitions from OSGi resource[META-INF/dispatcher/petstore-servlet.xm
14.06.2007 14:31:14 org.springframework.context.support.AbstractApplicationContext obtainFres
INFO: Bean factory for application context [de.kolbware.springOsgiUtil.context.OSGiXmlWebAppl
14.06.2007 14:31:15 org.springframework.beans.factory.support.DefaultListableBeanFactory preI
INFO: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultListab
14.06.2007 14:31:15 org.springframework.web.servlet.FrameworkServlet initServletBean
INFO: FrameworkServlet 'petstore': initialization completed in 906 ms
14.06.2007 14:31:15 org.springframework.osgi.context.support.AbstractRefreshableOsgiBundleApp
INFO: Publishing application context with properties (org.springframework.context.service.name
14.06.2007 14:33:49 org.apache.jasper.compiler.TldLocationsCache processWebDotXml
```

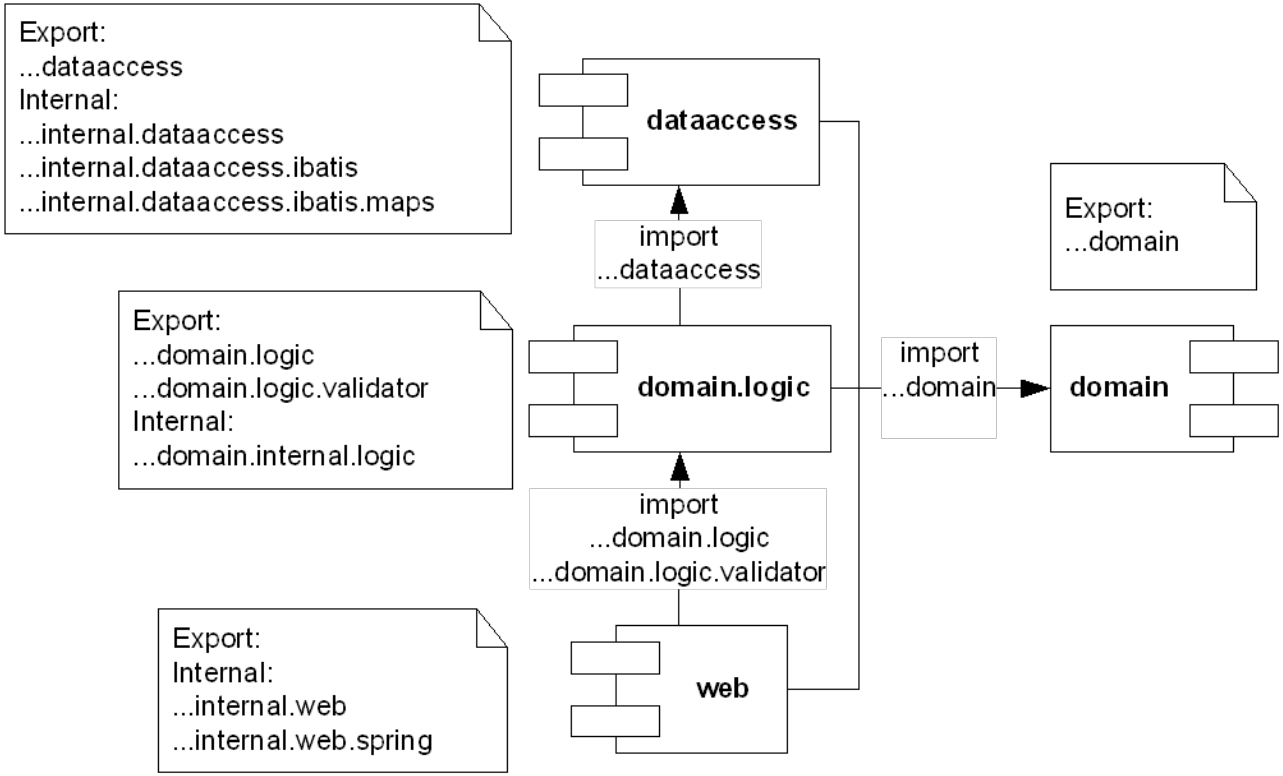
Powered By Spring

Powered By iBATIS © Clinton Begin

(Currently running on the Spring web tier)

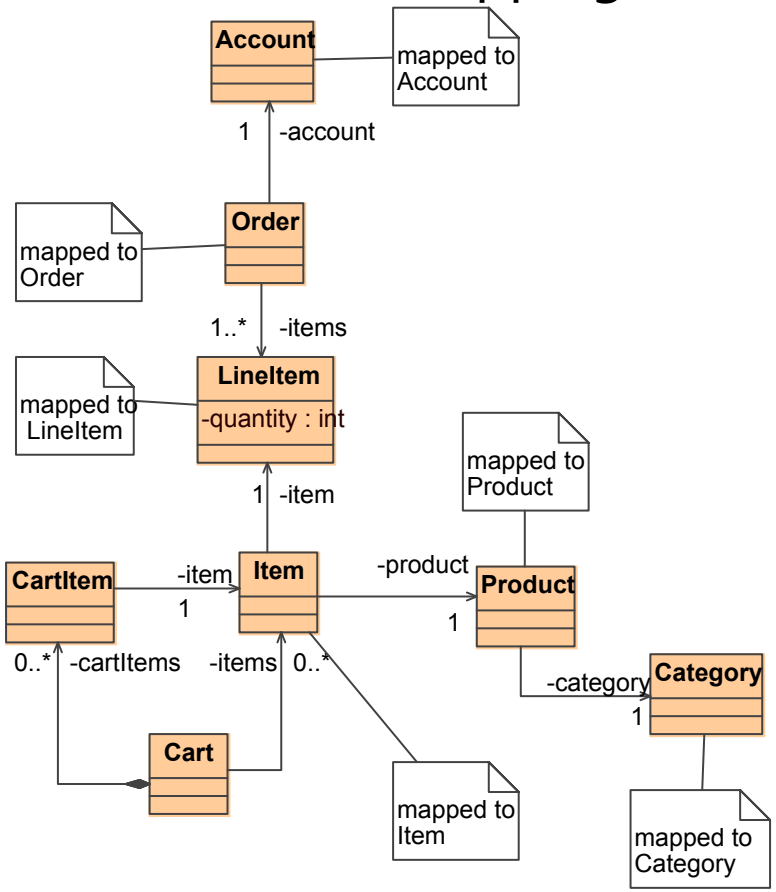
Example VI– jPetStore Architektur I

Bundles und ihre Abhängigkeiten



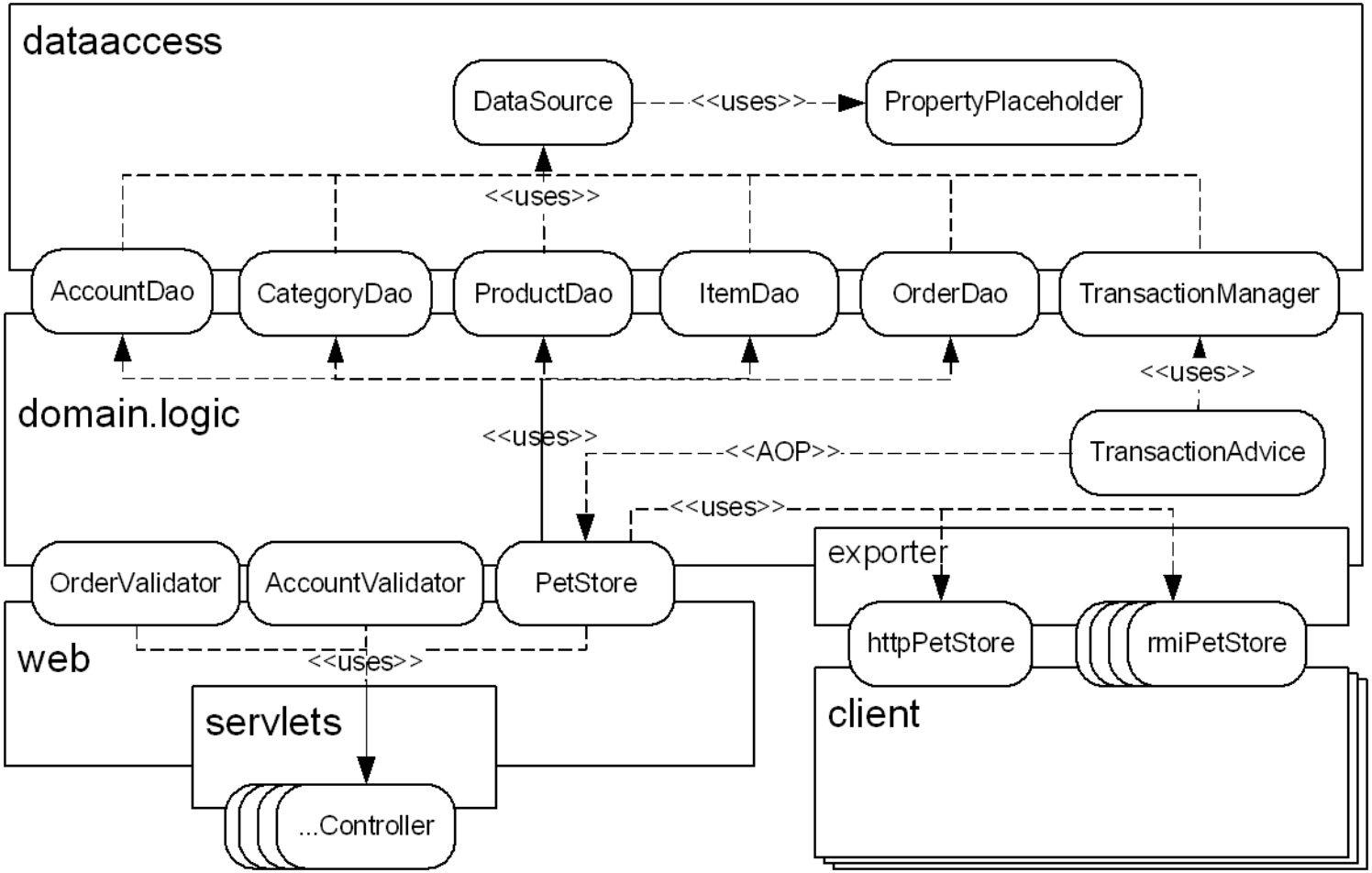
Example VII – jPetStore Architektur II

Domain-Model und Datenbank-Mapping



Example VIII – jPetStore Architektur III

Services



Stand der Dinge

- Spring-DM 1.0 M3
  - Enthält Basis-OSGi-Support (<osgi:...>-Namespace)
  - Context-Classloader-Handling
  - Unterstützung für Integrations-Tests
- Was noch fehlt:
  - Unterstützung für Web-Anwendungen (Spring DM- 1.1)
  - ???

Ausblick

- Release geplant zum Spring-Release 2.5
- Weitere Informationen:
  - http://www.springframework.org/osgi
  - http://groups.google.com/group/spring-osgi
  - http://www.springframework.org/osgi/specification
- Spring-OSGi-Tutorial (für Eclipse-Developer)
  - http://www.eclipsecon.org/2007/index.php?page=sub/&id=3632

Ausblick: Noch mehr Spring… ☺

- Auch für Clients kann Spring verwendet werden:
  - Dependency-Injection für Eclipse-RCP-Anwendungen
  - Server-Kommunikation mit Spring

- Weiter interessant:
  - Dependency Injection für Extension-Points (z.B. Views in Eclipse-RCP über Spring erzeugen und mit Dependencies versorgen)

Das Buch zum Thema OSGi

- Erscheint im Frühjahr 2008 im dpunkt.verlag

Vielen Dank!!!

- … für die Aufmerksamkeit!

- Fragen jederzeit gerne
  - b.kolb@kolbware.de
  - gerd@gerd-wuetherich.de
  - martin.lippert@akquinet.de