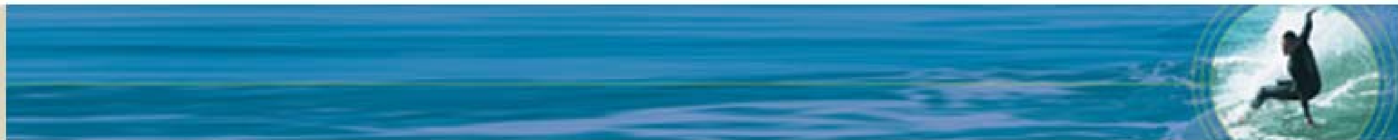
Spring und Eclipse Equinox kombiniert

Martin Lippert (it-agile GmbH)
Gerd Wütherich (comdirect bank AG)

Inhalt

- Eclipse Equinox
- Server-Side Eclipse
- Spring und Eclipse Equinox
- Beispiele
- Fazit



jax 2006
Konferenz für Java, XML, Web Services

Eclipse everywhere

- Eclipse als Java-IDE ist längst kalter Kaffee
- Eclipse als Rich-Client-Platform setzt sich durch
  - Beeindruckende Beispiele (siehe IBMs Lotus)
  - Gute architekturelle Grundlage, insbesondere für große und komplexe Systeme

jax 2006
Konferenz für Java, XML, Web Services

Eclipse Equinox

- Equinox seit 3.2 M4 eigenständig nutzbar
- OSGi R4 Implementation + Extension Mechanismus

Eclipse Platform
Debug
Search
UI IDE
...
Help
Search
RCP
SWT
Generic Workbench
...
JFace
Equinox
ExtensionRegistry
OSGi
...

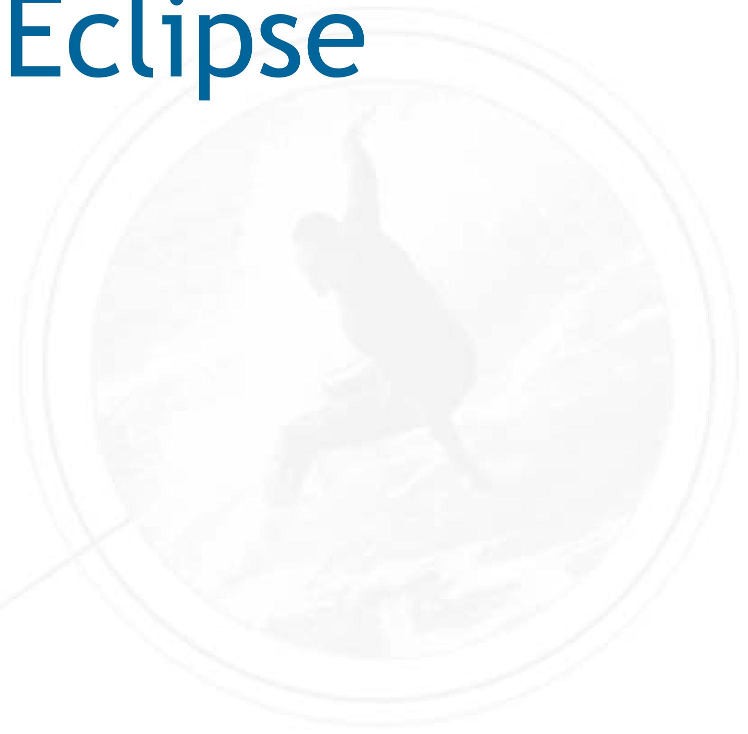
Was kommt als nächstes?

- Server-Side Eclipse:
  - Eclipse Equinox als Basis für server-seitige Anwendungen

Vorteile:

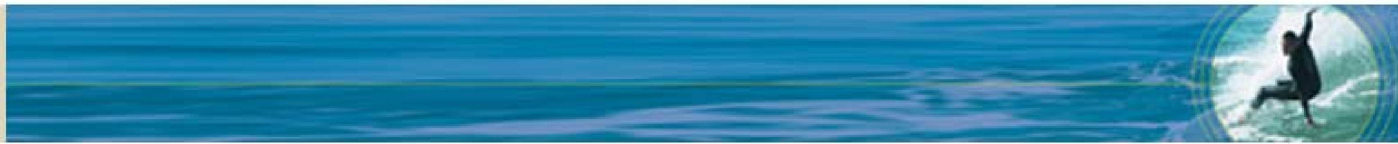
- Modularisierung mittels OSGi
  - Definierte Abhängigkeiten, Versionierung, public vs. private APIs, Updating, Services, …
- Flexibilisierung mittels Extension-Registry
  - Plattform-basierte Entwicklung, Komponentenmodell, Erweiterbarkeit

Server-Side Eclipse



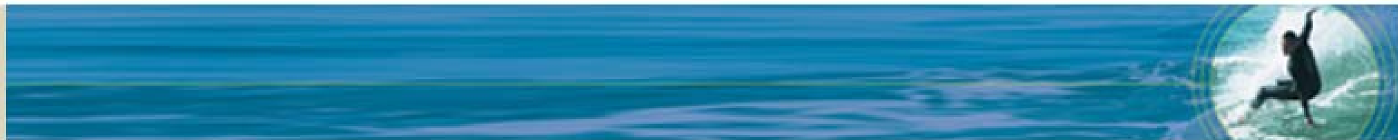
- Es besteht reges Interesse, z.B.
  - ECF Project
  - Open Healthcare
  - Rich AJAX Platform
  - Eclipse Component Framework
  - …
- Wie geht's im Detail?
  - *Server-Side Eclipse*
    Di. 11.45 – 13.00 Uhr, Saal 2a

jax 2006
Konferenz für Java, XML, Web Services



Eclipse ist nicht alles

- Gerade für server-seitige Anwendungen gibt es etablierte Technologien und Frameworks
- Allen voran: **Spring**



jax 2006
Konferenz für Java, XML, Web Services

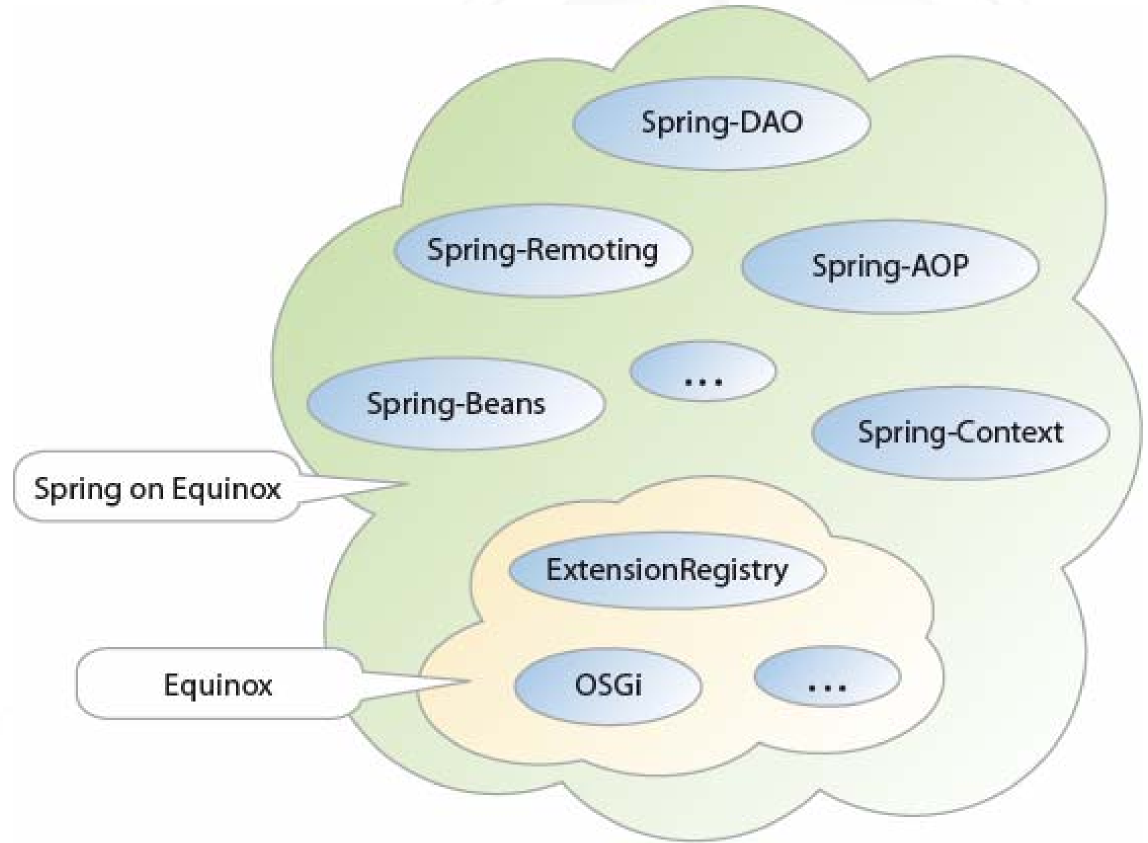
Warum Spring?

Vorteile:

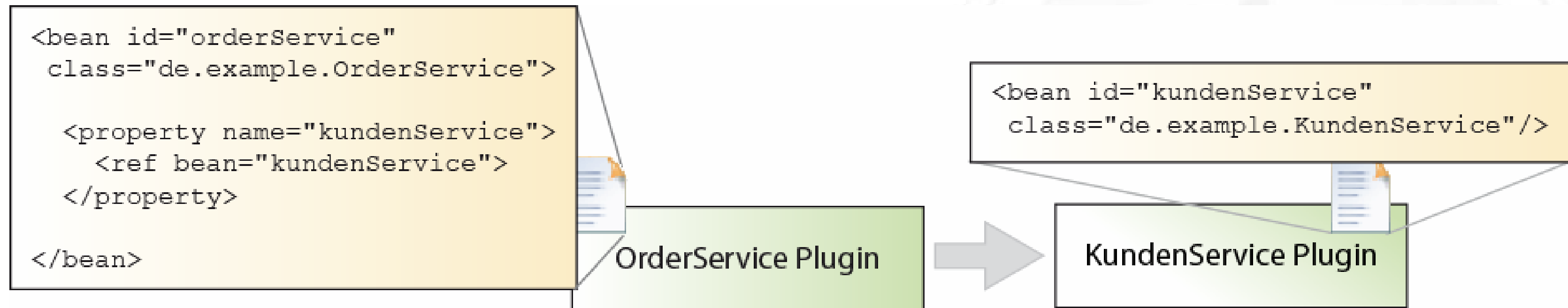
- Einfaches Programmiermodell (POJOs)
- Einfachere Testbarkeit
- Vermeidung von Abhängigkeiten durch Dependency Injection bzw. Inversion of Control
- Integrationsplattform für unterschiedliche Technologien (AOP, Persistence etc.)
- Vereinfachte Benutzung von Java EE und anderen Java APIs

Spring und Eclipse Equinox

- Equinox:
  - Modularisierung
  - Extension Mechanismus
- Spring:
  - POJOs
  - Einbindung Server-Technologien
- ‚Rich Server Platform'



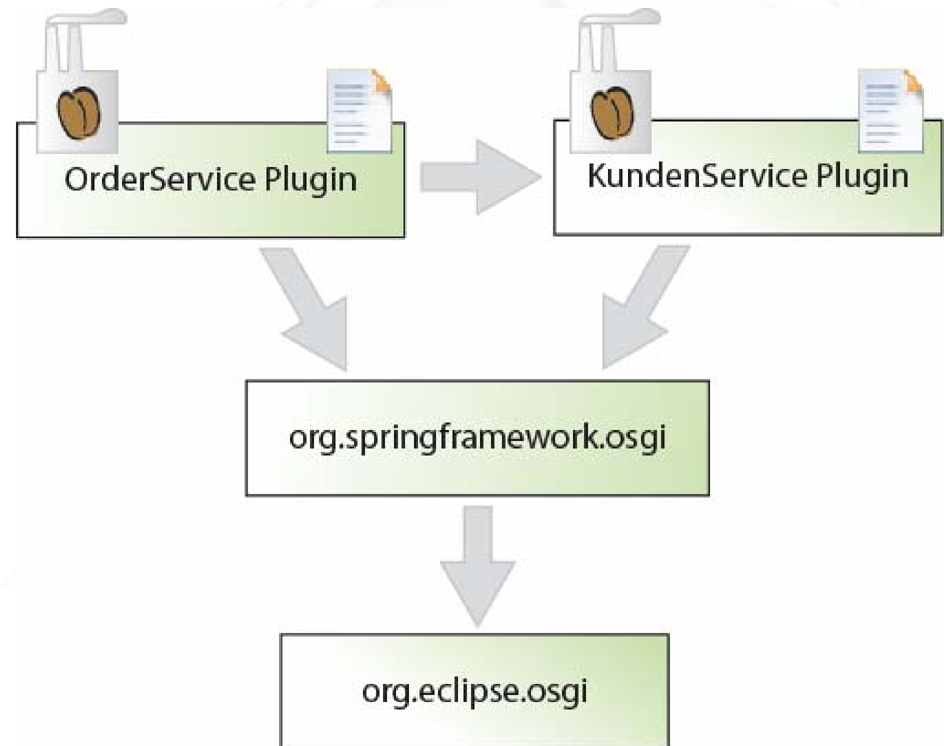
Ein einfaches Beispiel



- Plugin ‚KundenService' definiert eine Bean ‚kundenService'
- Plugin ‚OrderService' definiert eine Bean ‚orderService'
- OrderService hat Referenz auf die Bean ‚kundenService'

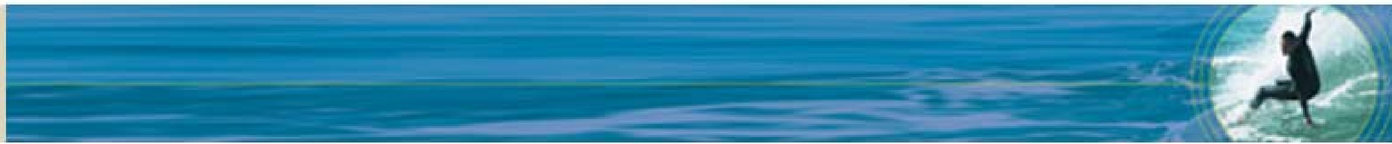
Realisierung

- Verschiedene Implementationen denkbar
- Beispiel basiert auf Spring „Sandbox“ Code
- Jedes Plugin besitzt eine eigene BeanFactory
- Plugin-übergreifende Injektion von Beans möglich
- Dynamisches Verhalten wird unterstützt



jax 2006
Konferenz für Java, XML, Web Services

Live Demo

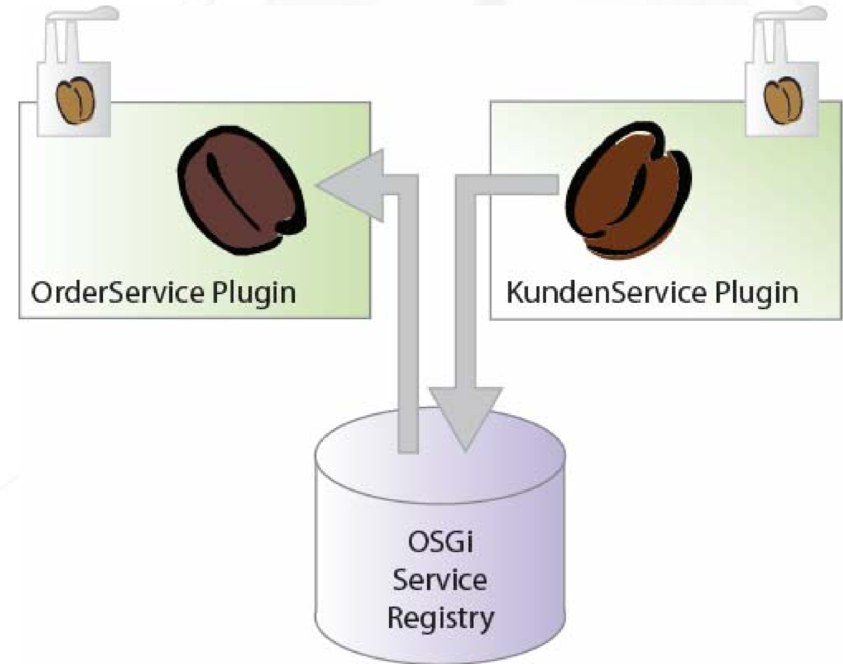


jax 2006
Konferenz für Java, XML, Web Services

Wie werden Beans plugin-übergreifend bereitgestellt?

Problem:

- Plugin-übergreifende Injektion (zunächst) nicht möglich



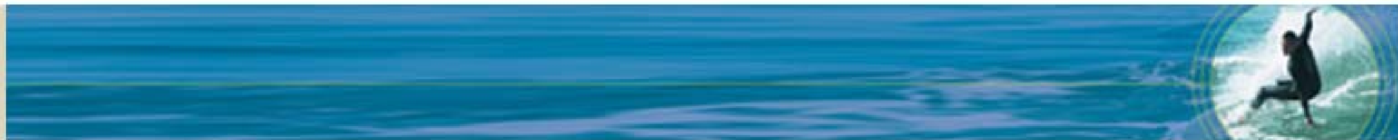
Lösung:

- Beans werden über die OSGi-Service-Registry als OSGi-Services bereitgestellt
- Zugriff über Proxy-Mechanismus möglich

Beans als OSGi-Services

- Beans können als OSGi-Services exportiert werden
- OSGi-Services sind plugin-übergreifend sichtbar

```
<beans>
 <bean id="kundenService"
      class="de.example.KundenService">
 </bean>

 <bean class="org.springframework.osgi.service.OsgiServiceExporter">
  <property name="exportBeans">
   <list>
    <value>kundenService</value>
   </list>
  </property>
 </bean>
</beans>
```

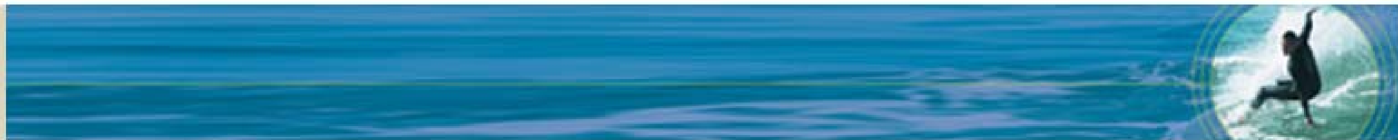


OSGi-Services als Beans

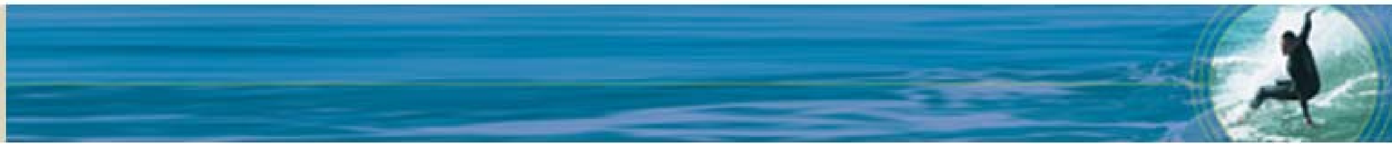
- OSGi-Services können als Beans bereitgestellt werden
- Proxy-Objekte kapseln den Zugriff auf die OSGi-Services
- Fehlerbehandlung bei nicht verfügbaren Services

```
<beans>
  <bean id="orderService" class="de.example.OrderService">
    <property name="kundenService">
      <ref local="kundenServiceProxy"/>
    </property>
  </bean>

  <bean id="kundenServiceProxy"
    class="org.springframework.osgi.service.OsgiServiceProxyFactoryBean">
    <property name="serviceType">
       <value>de.example.KundenService</value>
    </property>
    <property name="beanName"><value>kundenService</value></property>
</bean>
</beans>
```

jax 2006 Konferenz für Java, XML, Web Services

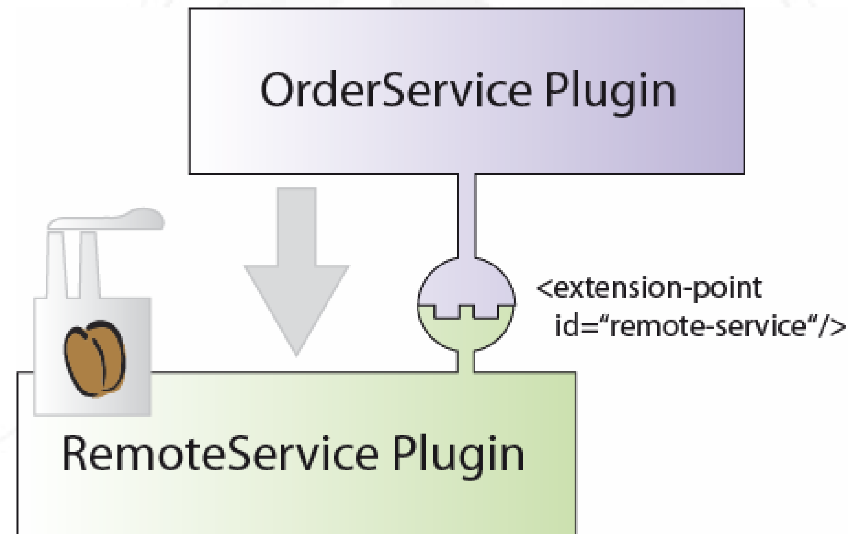


Live Demo



Der Extension Mechanismus

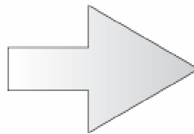
- Zusätzliche Möglichkeiten durch Verwendung der ExtensionRegistry
- Erzeugung von Beans in Abhängigkeit von Extensions
- Sinnvoll bei komplexen, gleichartigen Bean-Strukturen
- Erfordert Anpassungen an der Spring-OSGi-Implementation



jax 2006
Konferenz für Java, XML, Web Services

Der Extension Mechanismus II

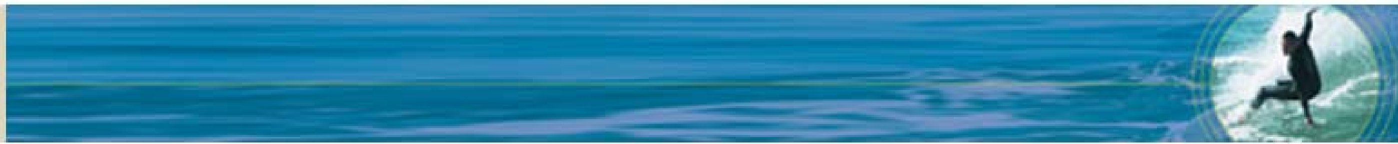
```
<extension
 point="de.example.remoteservice.remote-service">

  <remote-service
    id="orderService"
    class="de.example.OrderService"
    interface="de.example.IOrderService">

     <property name="kundenService">
        <ref bean="kundenService"/>
     </property>

  </remote-service>

</extension>
```



```
<beans>

  <bean id="orderService" class="de.example.OrderService">

     <property name="kundenService">
        <ref local="kundenServiceProxy"/>
     </property>

  </bean>

  <bean id="kundenServiceProxy"
   class="org.springframework.osgi.service.OsgiServiceProxyFactoryBean">

     <property name="serviceType">
        <value>de.example.KundenService</value>
     </property>
     <property name="beanName">
       <value>kundenService</value>
     </property>

  </bean>

  <bean class="org.springframework.remoting.rmi.RmiServiceExporter">

   <property name="service" ref="orderService"/>
   <property name="serviceInterface" value="de.example.IOrderService"/>
   <property name="registryPort" value="1199"/>

  </bean>

</beans>
```

jax 2006 Konferenz für Java, XML, Web Services

Live Demo



jax 2006
Konferenz für Java, XML, Web Services

Der Extension Mechanismus III

Vorteile:

- Ermöglicht die Definition eigener Konzeptionsmuster (z.B. RemoteService)
- Konzeptionsmuster können generisch implementiert werden

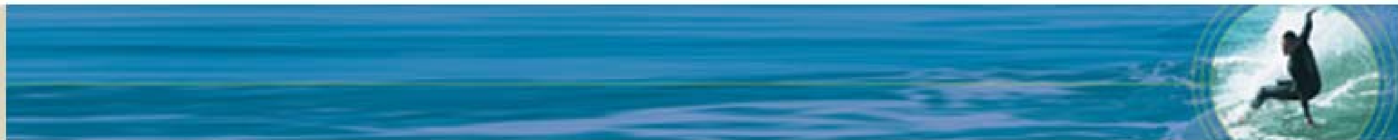
jax 2006
Konferenz für Java, XML, Web Services

Fazit

- Spring und OSGi ergänzen sich nahezu perfekt
- Sehr gute Basis für „Rich Server Platform“
- Zusätzliche Möglichkeiten durch Einbindung der ExtensionRegistry

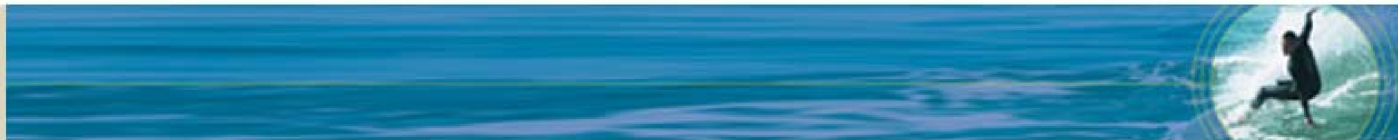
- Unsere Meinung: **Die Plattform für server-seitige Anwendungen oder Anwendungsteile**

jax 2006
Konferenz für Java, XML, Web Services



Vielen Dank…

- … für die Aufmerksamkeit!
- Fragen jederzeit gerne
  - martin.lippert@it-agile.de
  - gerd.wuetherich@comdirect.de
- Code-Beispiele verfügbar unter
  - http://www.it-agile.de/newsfeed oder
  - http://www.wuetherich.com



jax 2006 Konferenz für Java, XML, Web Services