

## Going live mit einem eigenen Geronimo-basierten Application Server



Orientation in Objects GmbH

Weinheimer Str. 68  
68309 Mannheim

www.oio.de  
info@oio.de

Version: 1.0

### Wer steht vor Ihnen?

- 20+ Jahre Erfahrung in der Softwareentwicklung
  - Java EE Technologien
  - Java EE Application Server
  - Service Orientierte Architekturen
- Veröffentlichungen
  - Buch „Apache Geronimo im Einsatz“ als Autor
  - Buch „J2EE- Entwicklung mit Open-Source-Tools“ als Mitautor
  - Mehrere Fachartikel
    - **JavaMagazin, XML & WebServices Magazin**
  - Vorträge auf Konferenzen
    - **OOP, JAX**



2

## „Orientierung“ in Objekten

# Java und XML

### ) Akademie )

- Schulungen, Coaching, Weiterbildungsberatung, Train & Solve-Programme

### ) Beratung )

- Methoden, Standards und Tools für die Entwicklung von offenen, unternehmensweiten Systemen

### ) Projekte )

- Schlüssel fertige Realisierung von Software
- Unterstützung laufender Projekte
- Pilot- und Migrationsprojekte

3

## Agenda

- Einführung
  - Motivation (J2EE bis heute)
  - Problem
  - Geronimo
- Szenarien
  - Konfiguration- und Installationsoptionen
  - Mandantenfähigkeit
  - Clonen, Aktualisieren und Erweitern



4

## Java Enterprise Hype



- J2EE 1.x
  - standardisierte plattformunabhängige Betriebsinfrastruktur für die „neue Java Welt“
    - **multiusersicher**
    - **skalierbar**
    - **transaktional**
  - Entwicklungs- und Strukturierungsparadigma mit feingranularen Komponentenmodellen
- J2EE 5
  - serverunabhängiges Deployment
  - Verbesserung von Administration und Management der Anwendungen
  - „Ease of Development“

5

## Newer Hypes ?



- Komponentenorientierung verliert ihre Bedeutung
  - CBD vs. SOA
  - sind EJB-Komponenten zu feingranular ?
  - stürzt J2EE ab wie CORBA?
    - „The recent release of Java EE 5 (formerly known as J2EE 1.5) has sparked a debate about the future of this popular platform. Although some industry watchers predict a rapid decline due to its growing complexity, its viability is rock solid.“ Gartner Group 9/2006
- leichtgewichtige Frameworks für POJO-Komponenten stellen zwar standardisiertes aber starres Modell von J(2)EE in Frage
  - Picocontainer
  - Spring
    - **“Strong Performer in Open Source Projects“ Q2/2006 Forrester**
  - Google Guice

6

## Agenda



- Einführung
  - Motivation (J2EE bis heute)
  - **Problem**
  - Geronimo
- Szenarien
  - Konfiguration- und Installationsoptionen
  - Mandantenfähigkeit
  - Clonen, Aktualisieren und Erweitern



7

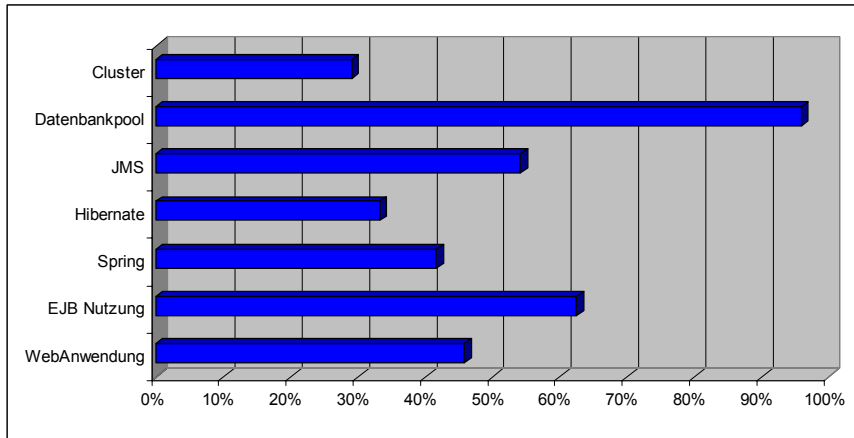
## Java auf dem Server - aus OIO Kundensicht



- ca. 60% der Kunden nutzen Enterprise Java Beans
  - EJB Anwendungen entstanden vor 2-6 Jahren
  - in manchen Fällen EJB nur für Remoting
- 0% der Neuimplementierungen starten momentan mit EJBs
  - Spring/ Hibernate ist absoluter Hype
  - EJB 3/ JPA läuft langsam an
- Web Anwendungen im weiten Sinne überwiegen
  - HTTP Übertragung
  - nur manchmal reine RMI Übertragung

8

## Bestandteile der Server Anwendungen - OIO Kundensicht (2001-2006)



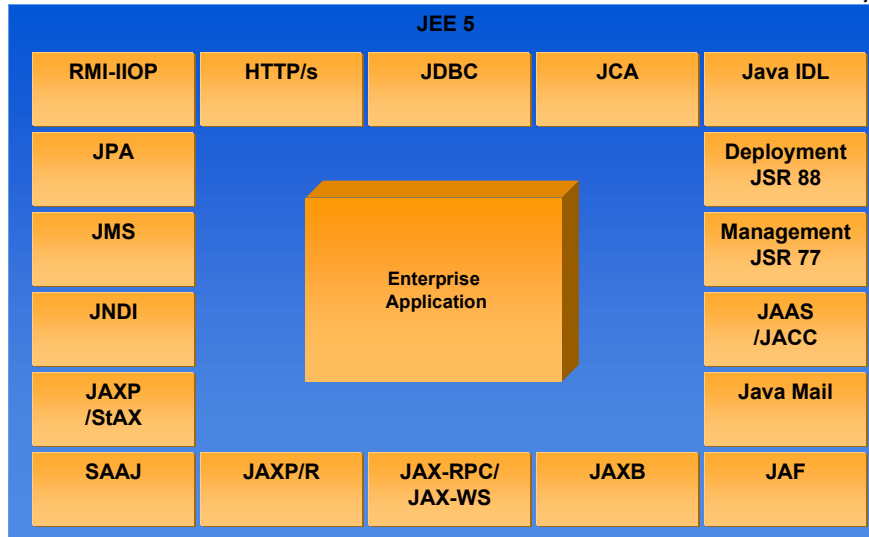
9

## Drei Lösungswege

- Full Feature J2EE-Server
  - Weblogic, WebSphere, Netweaver..
  - Geronimo, JBoss, Jonas, Glassfish..
- Kein Applicationsserver - Pure Spring
- Tomcat mit Erweiterungen

10

## JavaEE Services



11

## Full Feature JavaEE Server

- Pro
  - alle APIs unterstützt
    - JMS, JTA, JCA...
  - zertifizierte Kombination von Enterprise Services Komponenten in einer Serverdistribution
  - QoS-Angebote von Appserver-Anbietern
  - definierte Administrationsumgebung
  - Management und Monitoring „aus einer Hand“
- Contra
  - schwergewichtige Infrastruktur verbraucht Ressourcen
  - überflüssige Entwicklungsressourcen
  - überflüssiges Administrationskompetenz benötigt

12

## Pure Spring



- Pro
  - keine Ressourcenverschwendung
    - **Laufzeit und Speicher der Produktivumgebung**
    - **Administrationswissen**
    - **Entwicklerarbeitsplatz**
  - einfacher Integrations und Komponententest
    - **offeriert schnelle Entwicklungszyklen**
- Contra
  - kein shared Resources Environment
  - komplexe Konfiguration zur Bereitstellung spezieller API
    - **JTA, JMS, JCA**
  - kein zertifizierter Mix von Enterprise Service Komponenten
  - undefinierte QoS-Angebote
  - keine standardisierte Administrations- und Managementsicht der Anwendung

13

## Warum nicht Tomcat als Application Server?



- „Spring im skalierbaren Webcontainer“
  - again - Spring beantwortet nicht alle Fragen im Lebenszyklus
    - **JMS**
    - **JTA (XA)**
- kein Erweiterungskonzept für Deployments
  - Spring Deployment
  - alternative Modularisierungsansätze
- Administrations und Managementunterstützung begrenzt
  - Datenbankpool verwalten
  - mangelndes KnowHow bei Administratoren
- mangelnde Möglichkeiten der Konfiguration zur Laufzeit
- Softwareverteilungsprobleme
- optionale EJB Unterstützung ?
  - nur in Anwendung (z. B. OpenEJB)

14

## Alternative Geronimo?

- Pro
  - „Geronimo And Tomcat Provide The Core Platform“
    - „ASF Projects Create Open Source Software For SOA And Web Services“ 3/2007 Forrester
  - Websphere Community Edition
  - “Strong Performer in Open Source Projects“ Q2/2006 Forrester
- Contra
  - J(2)EE Risiko und Potential als Transaktionsplattform niedrig (Gartner 2/2007)
  - Apache Geronimo Nischenprodukt „Magic Quadrant for Enterprise Application Servers,2Q06“Gartner

15

## Agenda

- Einführung
  - Motivation (J2EE bis heute)
  - Problem
  - **Geronimo**
- Szenarien
  - Konfiguration- und Installationsoptionen
  - Mandantenfähigkeit
  - Clonen, Aktualisieren und Erweitern



16



## Geronimo? Was ist das...



- J2EE 1.4 konformer Application Server
  - J2EE 1.4 TCK (Technology Compatibility Kit) im Okt '05 bestanden
- OpenSource Implementierung von Apache
  - Steht unter Apache Software License nicht LGPL
- „Zusammenstellung“ mehrerer Projekte
  - Vervollständigung des J2EE Stacks durch ASF/BSD lizenzierten Code
- Erstellung eines „eigenen Servers“ möglich



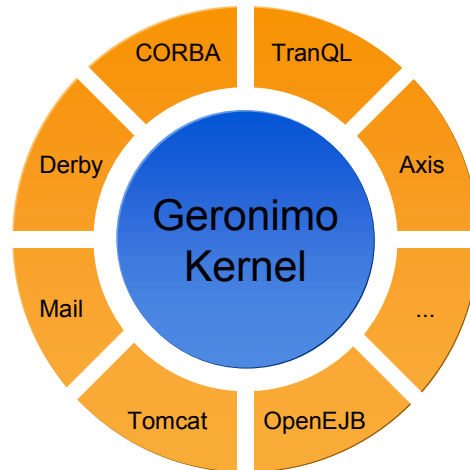
## Warum Application Server von Apache?



- Bisher kein J2EE Server mit BSD abgeleiteter Lizenz
  - JBoss und JOnAS haben GPL abgeleitete Lizenz
- Apache Software Foundation hat einige J2EE Projekte
  - einzelne Bestandteile aber keine Integration
- Best of breed app server reusing Open Source from the community



## Geronomos Integrationsgedanke



19

## Benutzte „FremdKomponenten“

- OpenEJB
  - EJB Container (<http://incubator.apache.org/openejb/>)
- Jetty
  - WebContainer (<http://jetty.mortbay.org/>)
- Tomcat
  - WebContainer (<http://jakarta.apache.org/tomcat>)
- ActiveMQ
  - JMS Provider (<http://incubator.apache.org/activemq/>)
- TranQL
  - EJB CMP Engine/ DB Connection Pools (<http://tranql.codehaus.org/>)
- HOWL
  - Logging (Transactions) (<http://howl.objectweb.org>)
- ...

20

## Agenda

- Einführung
  - Motivation (J2EE bis heute)
  - Problem
  - Geronimo
- **Szenarien**
  - Konfiguration- und Installationsoptionen
  - Mandantenfähigkeit
  - Clonen, Aktualisieren und Erweitern



21

## Anforderungen für den Live Betrieb - OIO Kundensicht I

- Konfiguration des Servers muss einfach sein!
  - Administratoren wollen nicht „kryptische“ XML Dateien editieren
- Aufsetzen eines weiteren Servers muss einfach sein
  - ebenso Update auf neue Serverversion auf mehreren Instanzen
  - „Cloning“ der Umgebung
  - Softwareverteilung, Updatemechanismus
    - „Cluster-Update“
- Stabilität, Performace
  - Ausfallsicherheit - Standby System
- Mandantenfähigkeit, Multi-Application Environment

22

## Anforderungen für den Live Betrieb - OIO Kundensicht II



- Erweiterung zur Laufzeit
  - z. B. auch neue Bibliotheken in Classpath
- Complete Deployment
  - ein Archiv pro Anwendung
- Management - Überwachung
  - Einfache Tools, JMX, etc
- Sicherheit
  - Standardmäßig nicht „alles offen“

23

## Optimierte Laufzeitumgebung...



- Keine ungenutzten Komponenten
  - EJB Container
  - JCA-Adapter
  - JMS-Provider
- Keine fragwürdigen Standardeinstellungen
  - Einstellungen meist anwendungsabhängig!
- Eventuell Erweiterbarkeit

24

## Agenda

- Einführung
  - Motivation (J2EE bis heute)
  - Problem
  - Geronimo
- Szenarien
  - **Konfiguration- und Installationsoptionen**
  - Mandantenfähigkeit
  - Clonen, Aktualisieren und Erweitern



25

## Geronimo Server und Module

- Server arbeitet modulbasiert
  - Modul Verwaltungseinheit des Servers
    - **elementare, zusammengehörende Bausteine (z. B. Connector)**
- Serverbestandteile sowie Applikationen sind Module
  - können auch „zusammen gepackt“ werden
    - **z. B. EJB Anwendung mit konfigurierter Datenbankanbindung**
- Module müssen nicht ausgeführt werden
  - können sich „offline“ im Server befinden
    - **Stop bedeutet kein Undeployment**

26

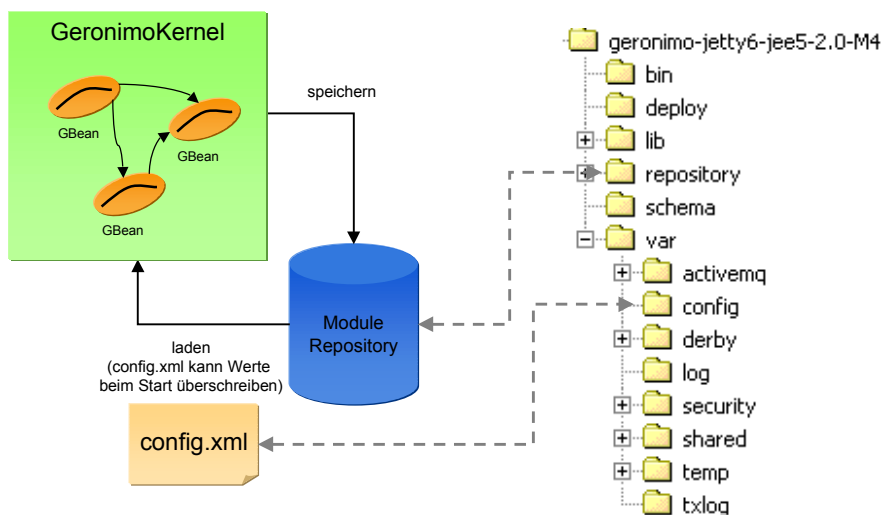
## Geronimo Module - (II)

- Modul besitzt eindeutigen Namen innerhalb des Servers (ModuleId)
  - Beispiel: `geronimo/welcome-tomcat/1.1-SNAPSHOT/car`  

{	{	{	{
groupId	artifactId	version	type
- Innerhalb des Servers Speicherung in Repository
  - Ablage erfolgt im proprietärem CAR Format
  - CAR Datei enthält Konfigurationsinformationen
- Module Zusammenstellung von konfigurierten GBeans
  - GBeans kleinste verwaltbare Einheit

27

## Geronimo Modulverwaltung- und Konfiguration



28

## Einfache Konfiguration mittels var/config/config.xml



- Zentrale Konfigurationsdatei für alle Module (config.xml)
  - Werte aus CAR im Repository können angepasst werden
    - **Portumstellungen, etc**
- Schnelle Anpassung an Umgebung möglich
  - Gleiches Modul mit Anpassungen in verschiedenen Servern
  - Platzhalterersetzungen in config.xml möglich
    - **config-substitutions.properties**
      - Ports umstellen mit Offset (portOffset)
- Änderungen zur Laufzeit werden persistiert
  - keine Änderung in Datei zur Laufzeit möglich!

29

## Beispiel: Arbeiten mit Modulen



```
$/deploy.bat list-modules
```

```
$/deploy.bat stop geronimo/welcome-tomcat/1.1.1/car
```

```
<attributes xmlns="http://geronimo.apache.org/xml/ns/attributes-1.1">  
<module name="org.apache.geronimo.configs/rmi-naming/2.0-M4/car">  
  <gbean name="RMIRegistry">  
    <attribute name="port">1099</attribute>  
  </gbean>  
  <gbean name="NamingProperties">  
    <attribute name="namingProviderUrl">rmi://0.0.0.0:1099</attribute>  
  </gbean>  
  ...  
</module>  
...
```

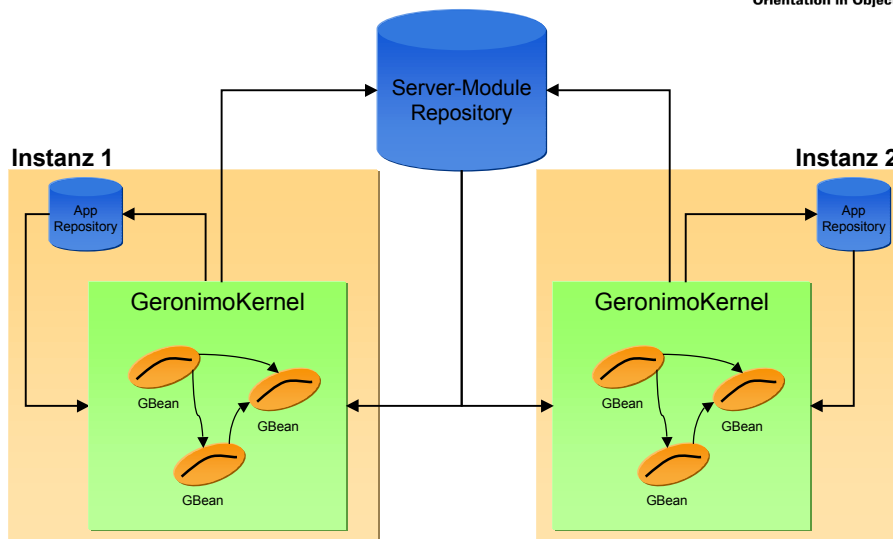


## Multiple Serverinstances - Geronimo 2.0

- Unterstützung für mehrere Instanzen aus einer Installation
  - nur Konfiguration unterscheidet sich
  - Repository wird geteilt
    - **Multiple Repositories möglich!**
- Servername muss beim Start angegeben werden
  - Name bezieht sich auf Verzeichnis, in dem var Dir gesucht wird
    - **org.apache.geronimo.server.name=servers/appServer1**
  - „Gleiche Anwendungen - Unterschiedliche Konfiguration“
- Portkonflikte können einfach behoben werden
  - Beim Start portOffset mit angeben
  - config-substitutions.properties verwenden

31

## Multiple Repositories - Geronimo 2.0 I



32



## Multiple Repositories - Geronimo 2.0 II



- Server kann mehrere Repositories besitzen
  - Trennung Server und Anwendungen
- Gleiche „Serverinstallation“ für mehrere Instanzen
  - nur Anwendungen unterscheiden sich
    - **schneller und komfortabler Update über „alle Server“**
- Ziel-Repository kann beim Deployment angegeben werden
  - Standardmäßig Deployment in alle Repos
  - leider nur Kommandozeilenunterstützung
    - **--target Option bei Deployer skript**

33

## Agenda



- Einführung
  - Motivation (J2EE bis heute)
  - Problem
  - Geronimo
- Szenarien
  - Konfiguration- und Installationsoptionen
  - **Mandantenfähigkeit**
  - Clonen, Aktualisieren und Erweitern



34

## Mandantenfähigkeit - 2 Anwendungen



- Möglichkeit 1
  - 2 Anwendungen auf 2 Serverinstanzen auf 2 Rechnern
    - **oft Realität - teuer und schwer wartbar**
- Möglichkeit 2
  - 2 Anwendungen in *jeweils* 1 Serverinstanz auf 1 Rechner
    - **siehe „Multiple Serverinstances“/ „Multiple Repositories“**
- Möglichkeit 3
  - 2 Anwendungen auf 1 Serverinstanz auf 1 Rechner
    - **So war und ist JavaEE Spec doch gemeint, oder? ;-)**
    - **oft der Wunsch aber nicht umgesetzt**

35

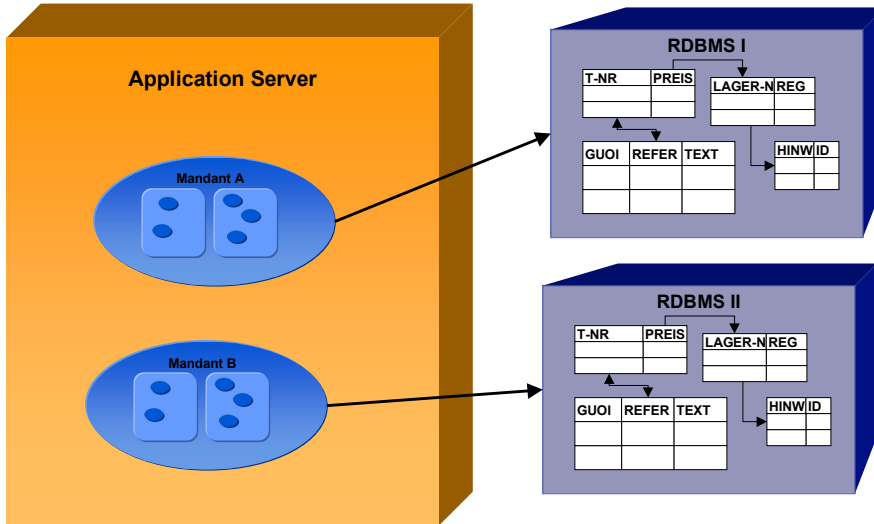
## Der 3. Weg zur Mandantenfähigkeit



- Probleme wenn Classloading nicht „richtig“ umgesetzt
  - Anwendungen müssen getrennt voneinander laufen
    - **z. B. Singletons, unterschiedliche Lib Versionen**
- Oft Mandantenfähigkeit über Buildmanagement - 2 Archive
  - Anwendungen unterscheiden sich nur in Konfiguration
    - **z. B. andere DB, anderer Web-Context**
- Potenziell Probleme beim „Austausch“ einer Anwendung zur Laufzeit
  - Anwendung besitzt Abhängigkeiten zu Fremdbibliotheken
    - **Datenbankpool -> Treiber nicht im Classpath**

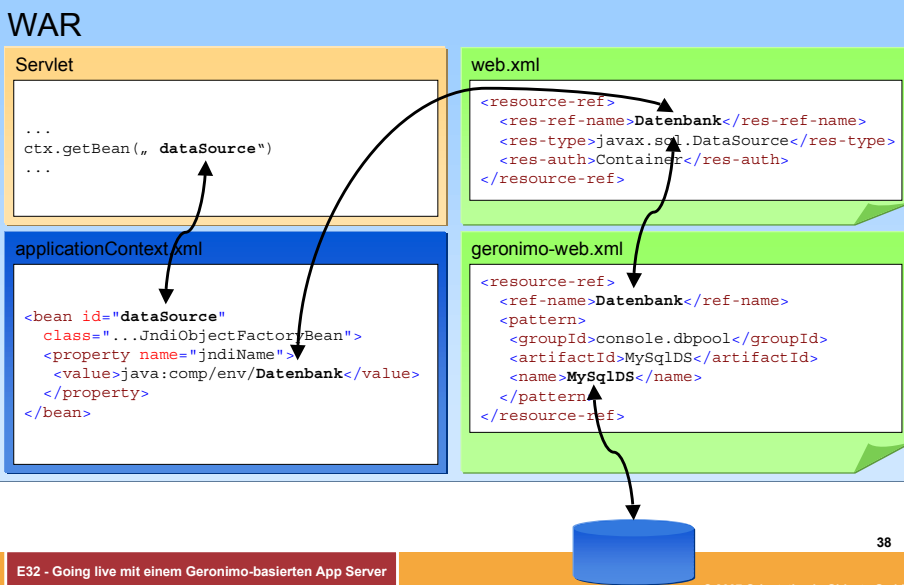
36

## Beispiel: Mandantenfähige Anwendung



37

## Beispiel: Anwendungsaufbau



## Beispiel: Deployment - Möglichkeiten



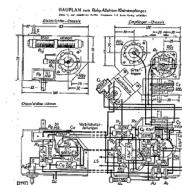
- Webanwendung greift auf Datenbank zu
  - Servlet holt über JNDI Lookup Referenz zur DB
    - Lookup „java:comp/env/Datenbank“
- Szenario 1 - „klassischer Fall“
  - Enterprise Archive (EAR) enthält Webanwendung
  - Datenbank separat innerhalb des Application Servers installiert
- Szenario 2
  - Enterprise Archive (EAR) enthält Webanwendung **und** ConnectionPool
  - Datenbank wird **mit** der Anwendung installiert und verwaltet

39

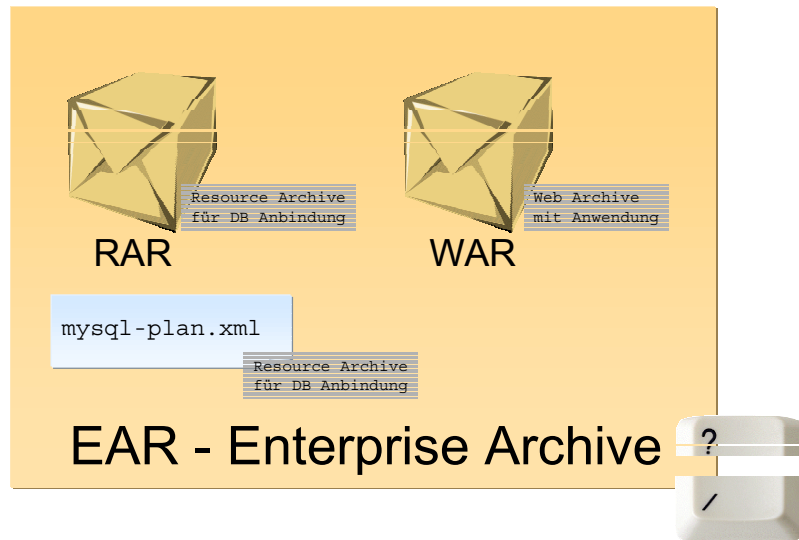
## Konfigurationen erstellen - Deployment Plan



- XML Datei zur Modulbeschreibung
  - für Dienste, Ressourcen und Anwendungen
    - z. B. PortNummer des WebServers, Datenbankangabe für EJBs
  - besitzt eindeutigen Namen
- Deploymentprozess benötigt einen Deployment Plan
  - im Archiv oder als Parameter für Deployer
    - META-INF/geronimo-application.xml
- Muss nicht mit eingepackt werden!
  - Andere Server erfordern Bundle
- Kann „mehrere Archive“ auf einmal beschreiben
  - Komplette EAR Beschreibung



## Beispiel: Anwendungsszenario 2



## Kompletter Geronimo Deployment Plan für EAR

```
<?xml version="1.0"?>
<application ...>
  <sys:environment>
    <sys:moduleId>...</sys:moduleId>
    <sys:dependencies>
      <sys:dependency>...</sys:dependency>
    </sys:dependencies>
  </sys:environment>

  <module>
    <web>db-web-application-1.0.war</web>
    <web-app xmlns="http://geronimo.apache.org/xml/ns/j2ee/web-1.2">
      ... Inhalt der geronimo-web.xml ...
    </web-app>
  </module>
  <module>
    <connector>tranql-connector-ra-1.3.rar</connector>
    <connector xmlns="http://geronimo.apache.org/xml/ns/j2ee/connector-1.2">
      ... Inhalt der geronimo-connector.xml ...
    </connector>
  </module>
</application>
```

## Agenda

- Einführung
  - Motivation (J2EE bis heute)
  - Problem
  - Geronimo
- Szenarien
  - Konfiguration- und Installationsoptionen
  - Mandantenfähigkeit
  - **Clonen, Aktualisieren und Erweitern**



43

## Geronimo Plugin (I)

- Beliebige Geronimo Module
  - Serverbestandteil oder Anwendung
  - „kompilierte Version“ eines Moduls
- Besitzen zusätzlich Metainformationen
  - Abhängigkeiten zur ausführenden Umgebung
    - **z. B. Geronimo Version, JVM Version, WebContainer**
  - Angabe zu externen Abhängigkeiten
    - **Welche Bibliotheken werden benötigt**
- Können einfach zwischen Servern übertragen werden
  - Weitergabe als Archiv (momentan CAR)
  - Abhängigkeiten können automatisch nachgeladen werden
    - **aus Remote (Maven) Repository**

44

## Geronimo Plugin (II)



- Plugins können komplett in Repositories abgelegt werden
  - Geronimo selbst über CAR Archive verfügbar!
  - Maven Repository Struktur
    - **geronimo-plugins.xml Metainformationen müssen vorhanden sein**
- Bisher ein zentrales Repository für Module vorhanden
  - <http://www.geronimoplugins.com>
- Geronimo Server kann selbst als Repository auftreten
  - <http://<host>/console-standard/maven-repo/geronimo-plugins.xml>
  - „Clone“ einer Geronimo Instanz

45

## Geronimo Plugin Import/Export



- Softwareverteilung für Serverbestandteile oder Anwendungen
  - Einfaches Servercloning über diesen Mechanismus
- Export von Plugins über WebConsole möglich
  - Angabe der Metainformationen per Wizard
- Installation zur Laufzeit
  - WebConsole kann sich mit remote Repository verbinden
  - Kommandozeile bietet Option an

46

## Beispiel: Module Importieren/ Exportieren

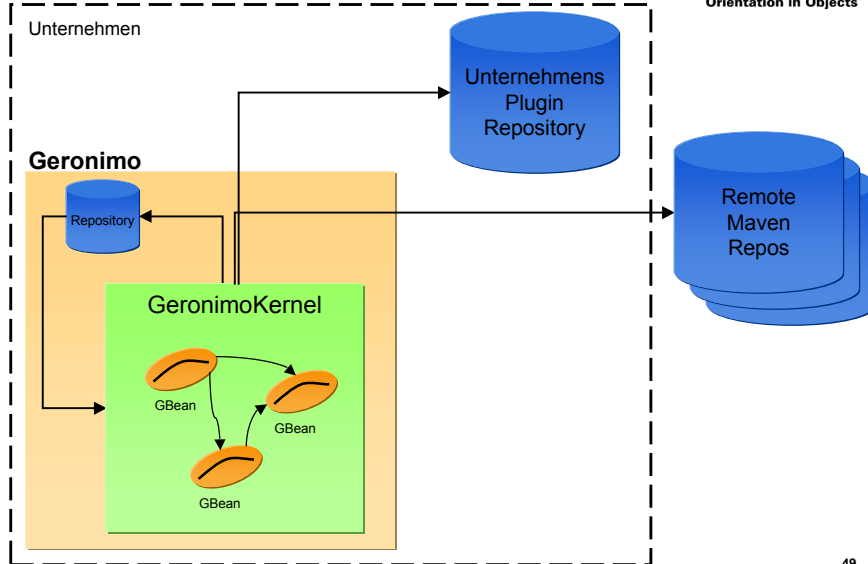


## Clusterunterstützung zur Laufzeit nachinstallieren?

- BeispielSzenario:
  - Minimaler Geronimo installiert (Little-G)
  - Anwendung 1 läuft
  - Anwendung 2 erfordert WebClustering und wird installiert
  - Server besitzt keine Clustering Unterstützung
  - Server soll nicht neu gestartet werden
  - Clustering Unterstützung soll zur Laufzeit nachinstalliert werden
- Lösung
  - Anwendung 2 wird als Plugin zur Verfügung gestellt (Maven-Plugin)
  - Metadaten des Plugins enthalten Informationen zum Clustering
    - **Abhängigkeiten, etc**
  - Server installiert Anwendung 2 und lädt fehlende Software nach



## „Pimp my installation“ - Big Picture



49

E32 - Going live mit einem Geronimo-basierten App Server

© 2007 Orientation in Objects GmbH



# Vielen Dank für Ihre Aufmerksamkeit !

Orientation in Objects GmbH

Weinheimer Str. 68  
68309 Mannheim

[www.oio.de](http://www.oio.de)  
[info@oio.de](mailto:info@oio.de)

Version: 1.0



Orientation in Objects GmbH

Weinheimer Str. 68  
68309 Mannheim

[www.oio.de](http://www.oio.de)  
[info@oio.de](mailto:info@oio.de)

Version: 1.0