

Zwischenbericht DA

Thema: „Dynamische Rekonfigurierung eines
Application Servers“

Vortragsgliederung:

- Application Server JBoss
- Java Management Extensions (JMX)
- Dynamische Rekonfigurierung

Application Server **JBoss** v2.4.4

- Tomcat Servlet Container (v3.2.3)
- EJB 1.1 (zusätzlich Message Driven Beans)
- OpenSource Project
- JMX als Basis

Java Management eXtensions (JMX)

Wozu kann man JMX verwenden?

- Management zur Laufzeit:
 - Kontrolle (Monitor, Timer)
 - Konfiguration (setAttribute, getAttribute, invoke(methode))

Wer verwendet JMX bereits?

(<http://java.sun.com/products/JavaManagement/jmxadoption.html>)

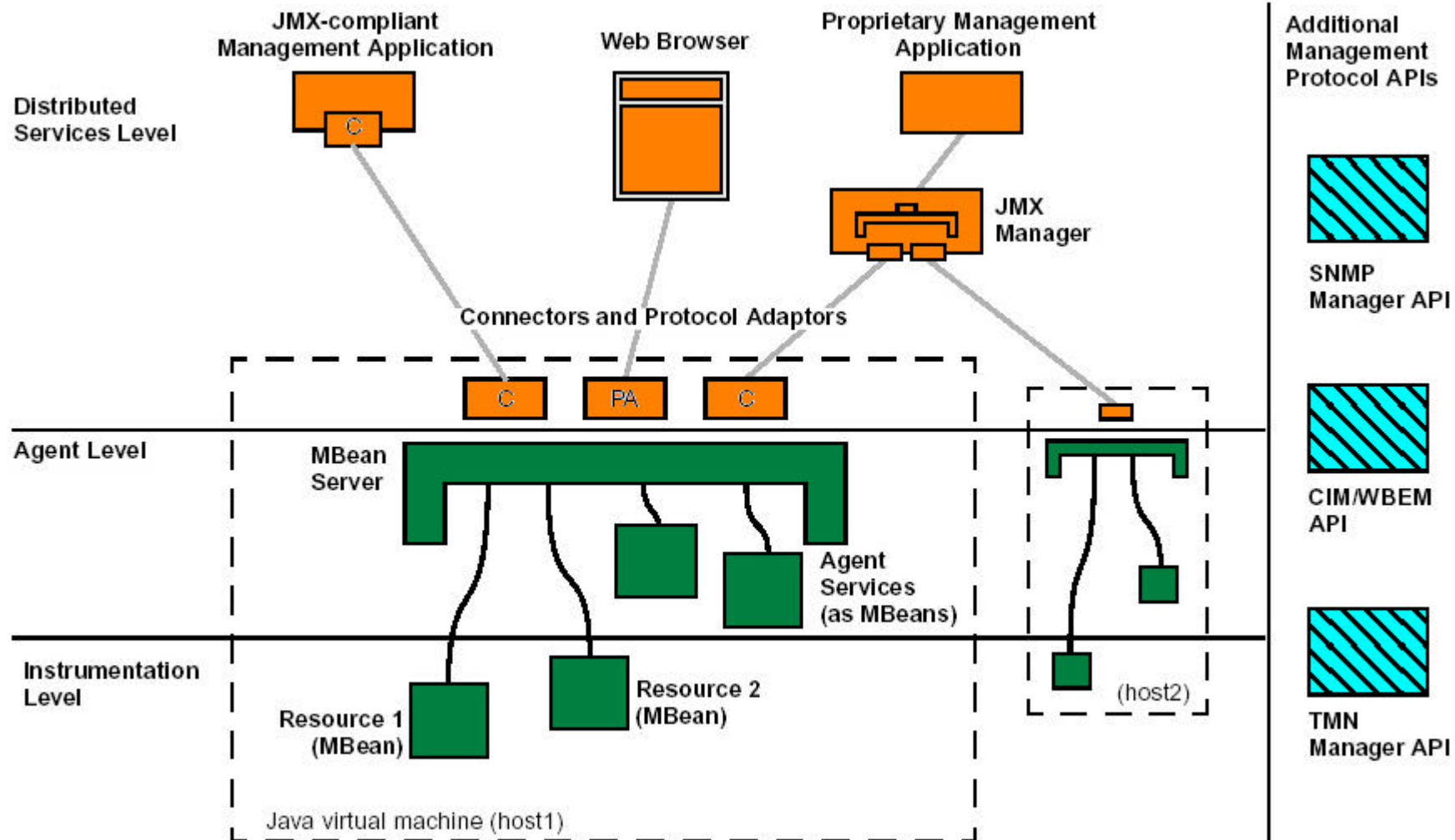
- JBoss, BEA Weblogic, iPlanet Application Server, ...

JMX - Spezifikation v1.0

Spezifikation v.1.0

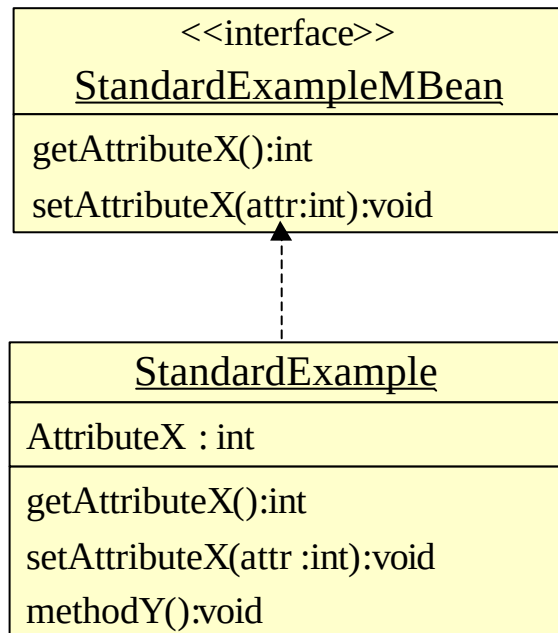
zukünftige Spezifik.

separate JSR

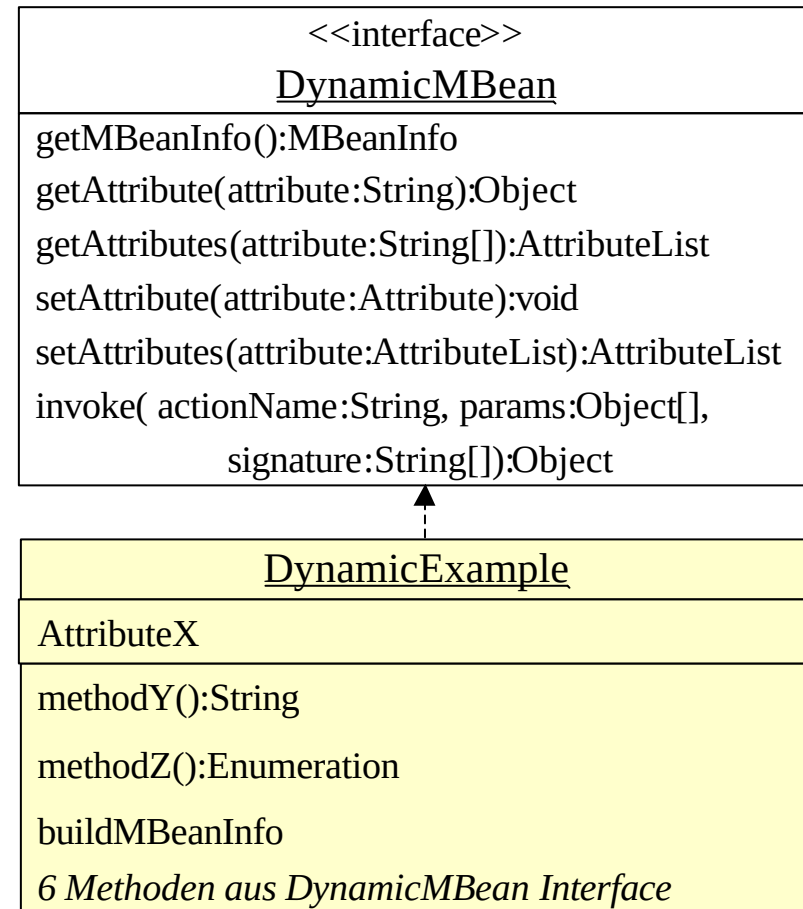


JMX - Standard MBean vs. Dynamic MBean

Standard MBean

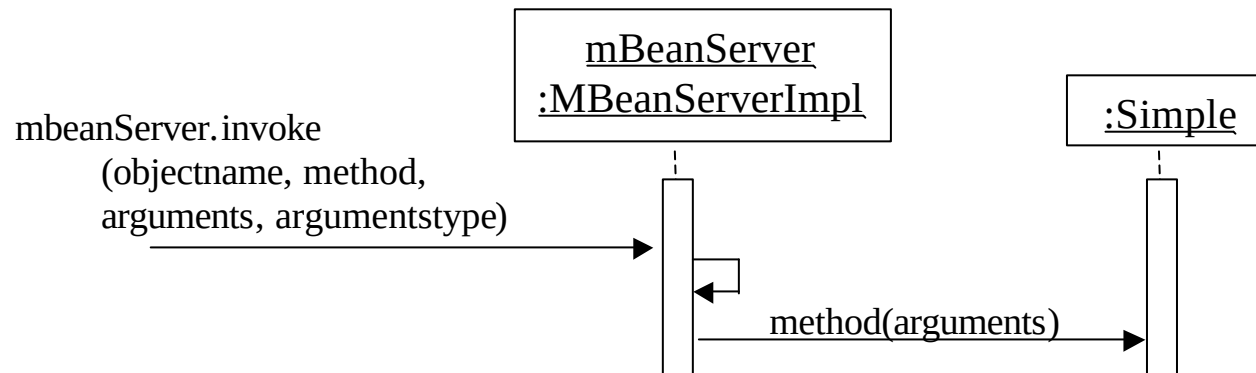


Dynamic MBean



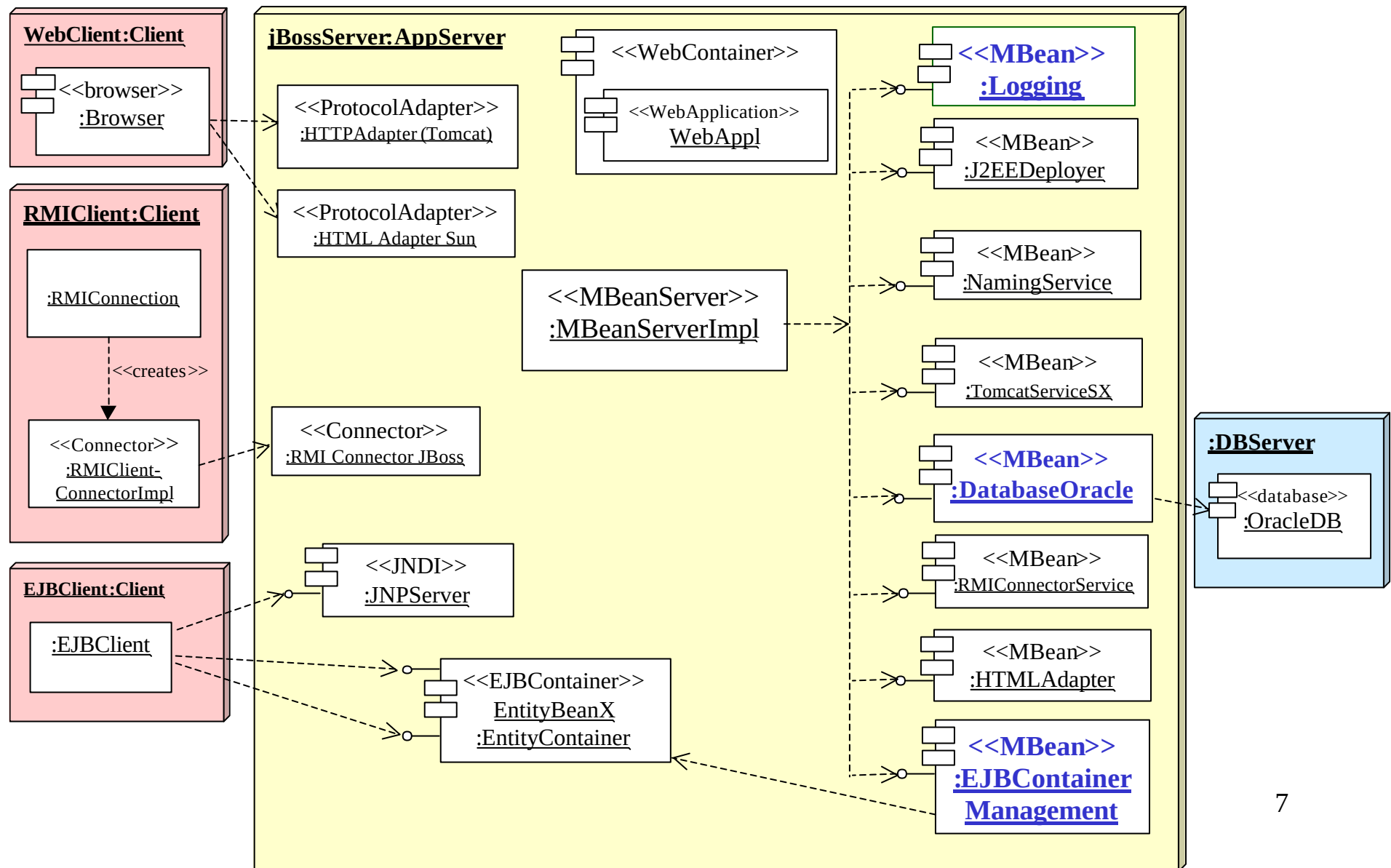
JMX - MBean Server

- **MBean Server:**
 - Registry für MBeans
 - ObjectName: [domainName]:property=value [, property=value]*



Deployment Diagramm (Auszug)

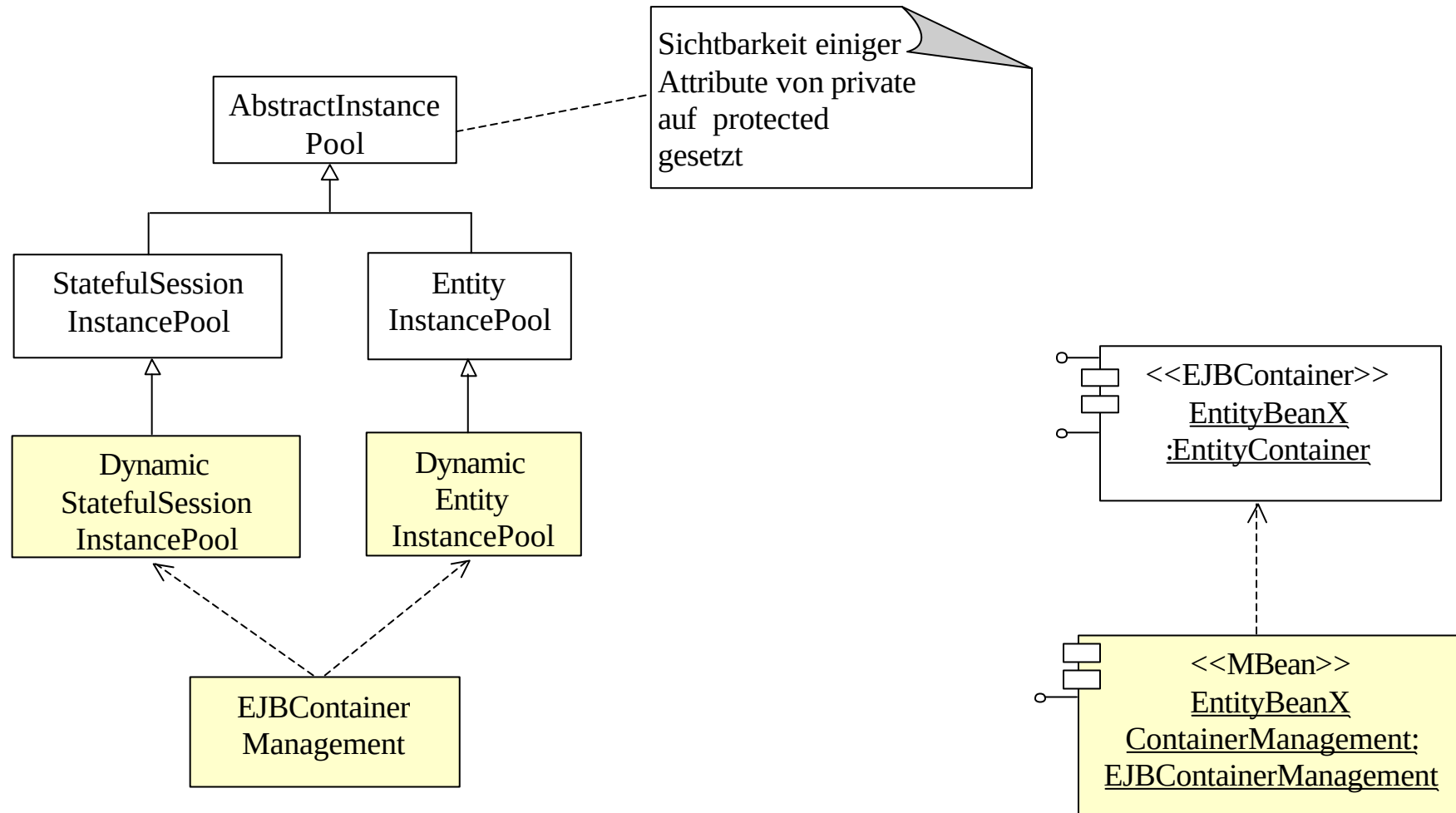
(JBoss, Clients, Datenbank)



Wahl einer geeigneten Architektur

- Clientzugriff über Web (WebApplication) (Client/Server)
- Nutzung von JMX, da JMX bereits Basis der JBoss Serverarchitektur (MBeans)
- Erzeugung XML-Dateien als Konfigurationsdateien
- APIs: JMX, JBoss, Log4j, Java2, J2EE
- Zugriff relationale Datenbank
- => Schritt zur Automatisierung (Monitoring, Notification)

Modifizierung zu dynamischer Rekonfigurierbarkeit



Aktueller Stand der Web-Applikation

- **Nutzung existierender „dynamischer“ MBean-Funktionalitäten**
 - ✓ MBean registrieren (Erzeugen XML-Datei)
 - ✓ dynamisches Klassenladen
 - ✓ Deploy / undeploy
 - ...
- **Erweiterung / Modifizierung existierender MBean-Funktionalitäten**
 - ✓ EJBContainer (InstanceCache, InstancePool)
Logging (Jakarta Log4j 1.1.3)
JBoss DatenbankConnectionpool
- **Erzeugung neuer MBeans**
 - ✓ rekonfigurierbare Anwendung Connectionpool
(Erzeugung Pool, ModelMBean, Persistence, PoolRegistry)

Screenshots Web-Application

• EJBContainerManagement

○ EJBContainerManagement

- [jndiName=CartHome, ContainerType=org.jboss.ejb.StatefulSessionContainer](#)
- [jndiName=NonOptimized, ContainerType=org.jboss.ejb.StatelessSessionContainer](#)
- [jndiName=NumberGenerator, ContainerType=org.jboss.ejb.EntityContainer](#)
- [jndiName=Optimized, ContainerType=org.jboss.ejb.StatelessSessionContainer](#)
- [jndiName=cd/CD, ContainerType=org.jboss.ejb.EntityContainer](#)
- [jndiName=cd/CDCollection, ContainerType=org.jboss.ejb.StatelessSessionContainer](#)
- [jndiName=demo/Demo, ContainerType=org.jboss.ejb.StatelessSessionContainer](#)
- [jndiName=hello/Hello, ContainerType=org.jboss.ejb.StatelessSessionContainer](#)

ContainerInterceptor	org.jboss.ejb.plugins.LogInterceptor@46eb1539
ContainerInvoker	org.jboss.ejb.plugins.jrmp.server.JRMPContainerInvoker
ContainerType	org.jboss.ejb.EntityContainer
InstanceCacheCurrentCapacity	50
InstanceCacheCurrentNumberOfInstances	0
InstanceCachePolicyType	org.jboss.ejb.plugins.LRUEnterpriseContextCachePolicy@a791539
InstanceCacheType	org.jboss.ejb.plugins.EntityInstanceCache
InstancePoolCurrentNumberOfInstances	0
InstancePoolMaxSize	0
InstancePoolSizeStrictMaximum	☐ True ☒ False
InstancePoolType	dynamicReconfiguration.ejbcontainer.DynamicEntityInstancePool

• Logging

Appender	
AppenderName: Default	
Threshold	DEBUG
File	./log/server6.log
Class	org.apache.log4j.RollingFileAppender
MaxBackupIndex	1
Layout	org.apache.log4j.PatternLayout
MaximumFileSize (Bytes)	512000
AppenderName: Console	
Threshold	DEBUG
Class	org.jboss.logging.log4j.ConsoleAppender
Layout	org.apache.log4j.PatternLayout

Screenshots Web-Application

• Änderung des Management-Interfaces

Erstellen eines Connectionpools zu einer Datenbank

[Mit dieser Anwendung wird eine MBean erzeugt, die die Funktionalität eines Connectionpools zu einer Datenbank bietet.]

Um diese MBean zu erzeugen, müssen folgende Angaben zur Datenbank gegeben werden:

Datenbanktreiber
(z.B. org.jboss.jdbc.HypersonicDatabase)

URL zur Datenbank
(z.B. jdbc:hypersonic:localhost:1476)

Nutzername:

Nutzerpassword:

Und jetzt noch ein paar Angaben zum Connectionpool. Der Connectionpool hält immer eine bestimmte Anzahl an Connections bereit (initialSize). Wenn danach weitere Anfragen kommen, wird der Pool automatisch um einen Inkrementierungswert (increment) vergrößert.

initialSize des Connectionpools:

increment des Connectionpools:

Objektname der MBean: ConnectionPoolType=ModelMBean, name=org.jboss.jdbc.HypersonicDatabase, port=1476, logname=
Managed Resource Classname: MBeanConnectionPool
Managed Resource Description: Database Connectionpool

visibility:

propertyName:

descriptorType:

displayName:

logFile:

log:

propertyName:

value:

propertyName:

value:

Attribute der MBean

numberOfPooledDatabaseConnectionsInPool

Attribut Typ: java.lang.Integer
isReadable: true
isWritable: false
ModelMBeanAttributeInfo Descriptor:
descriptorType:

name:

displayName:

iterable:

getMethod:

numberOfNowUsedPooledDatabaseConnections

Attribut Typ: java.lang.Integer
isReadable: true
isWritable: false
ModelMBeanAttributeInfo Descriptor:
descriptorType:

name:

displayName:

iterable:

getMethod:

Notifications, die die MBean versendet

AttributeChange

Description: ModelMBean Attribute Change Event
ModelMBeanNotificationInfo:
descriptorType:

severity:

name:

displayName:

Operationen der MBean

shutdownConnectionPool

Description: shutdownConnectionPool
Signature: 0
ModelMBeanOperationInfo Descriptor:
descriptorType:

name:

displayName:

role:

callForPooledDatabaseConnection

Description: get pooled Connection
Signature: 0
ModelMBeanOperationInfo Descriptor:
descriptorType:

name:

displayName:

role:

recycleUsedPooledDatabaseConnection

Description: recycle used pooled DatabaseConnection
Signature: 1
ModelMBeanOperationInfo Descriptor:
descriptorType:

name:

displayName:

role:

Performance-Tests

- ????