# JBOSS

Professional Open Source

Ben Wang( 王文彬 ), 2004

## Introduction

1. Open-source JBoss development process
2. JBoss & Professional Open-Source model
3. JBoss technology and JBoss sponsored projects

## I. JBoss Open-Source Development Process

## What is JBoss ?

Open Source J2EE Application Server

- Full J2EE 1.4 and EJB 2.0 specifications
- Advanced features: clustering, distributed caches invalidations, etc.
- 2M downloads in 2002, 1.5M downloads during 2003
- Free!

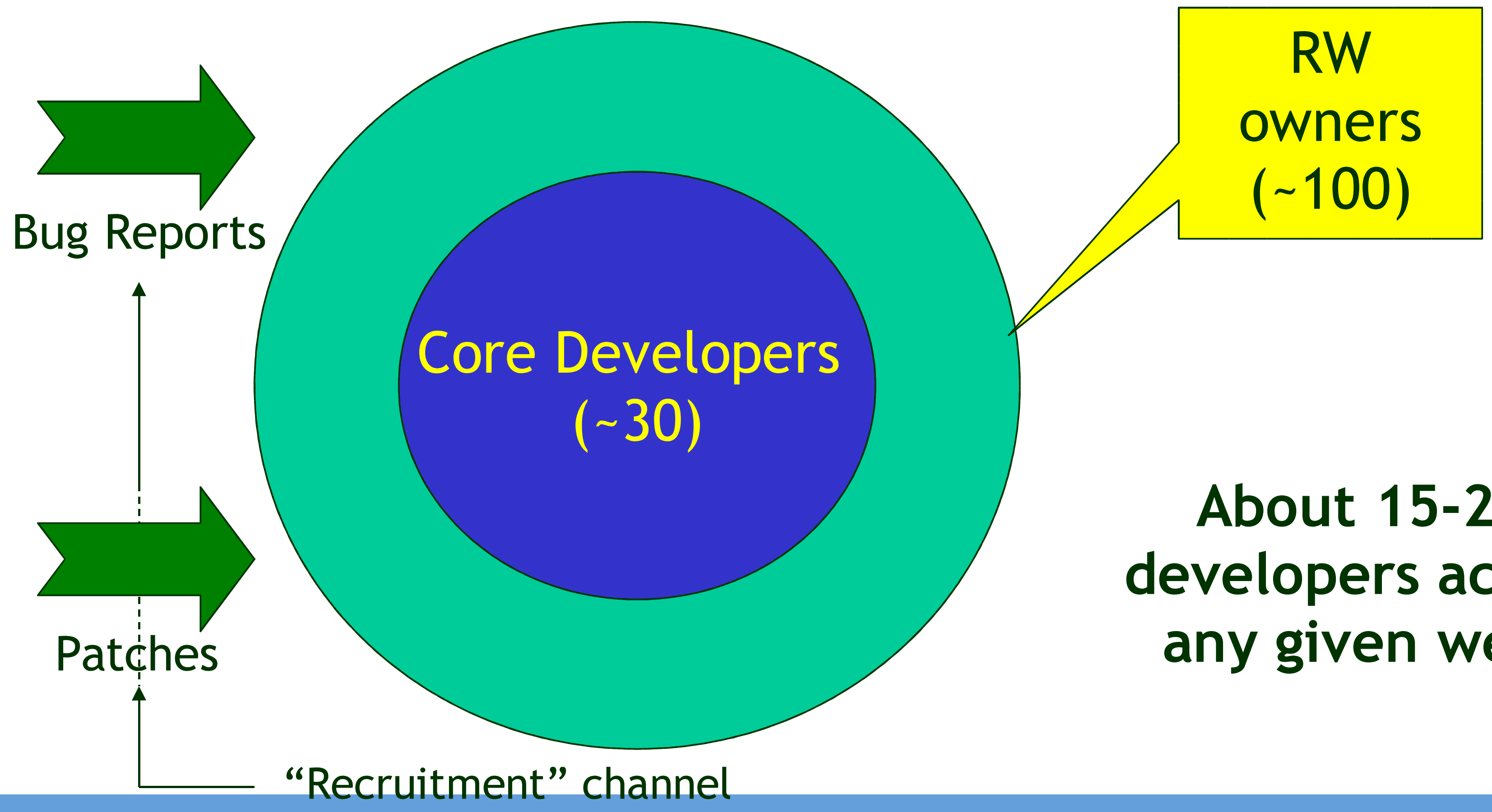
Strongly innovating

- 4.0 under development: AOP framework-based, distributed transactional caches, etc.
- Quality and Innovative middleware for the masses!

## Having JBoss, Inc. next to JBoss is very good for the project

- Core developers do support on JBoss: they want to make sure it works
- Most developers do consulting/training and are aware of real life problems and pressure
- Some people from JBoss, Inc. are “dedicated” to the project
- Allows JBoss to be tailored to the enterprise world

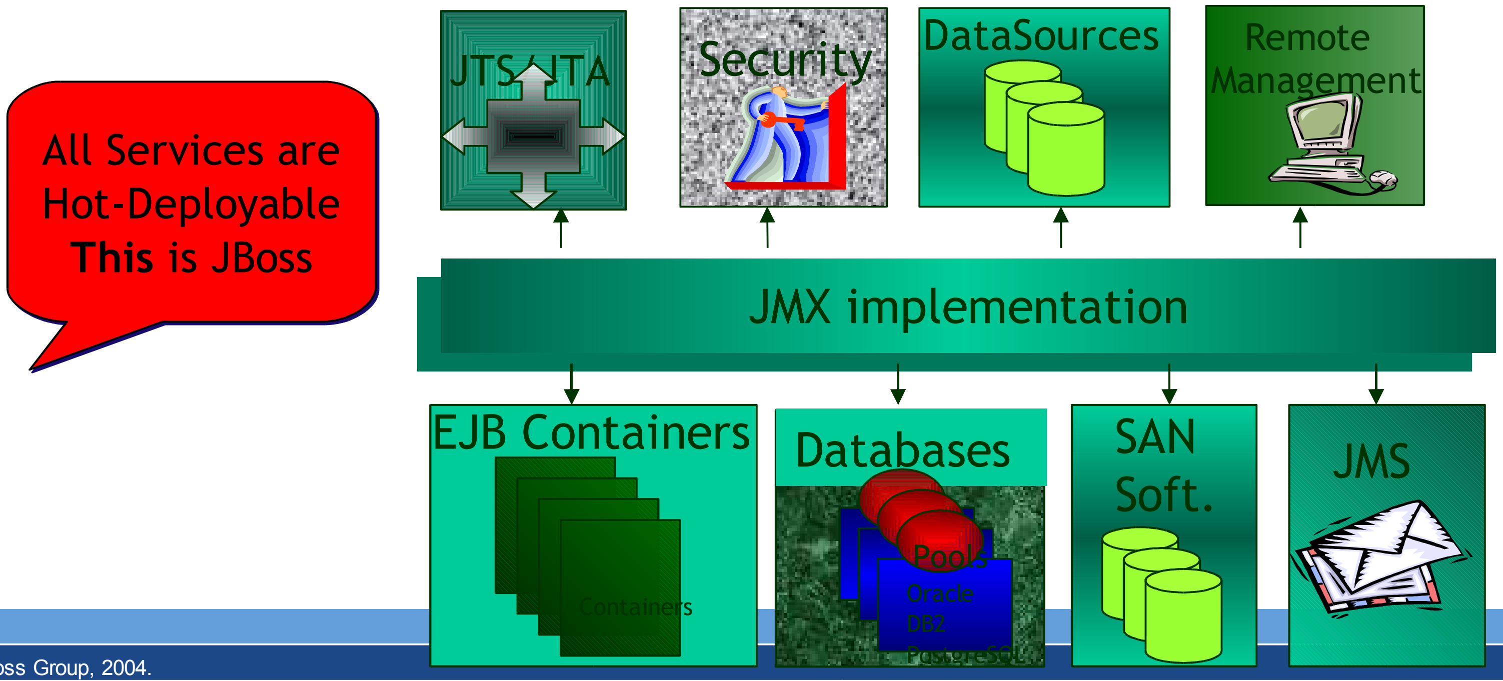
## Who Develops JBoss?



About 15-20 developers active any given week

# JBoss Code Management

JBoss architecture allows developers to work on sub-projects almost always without any interaction



## Meritocracy

- You are known for what you did
- Thrust can be acquired through
  - Development of critical parts
  - Quality and Simplicity of developments
  - Help you give to other developers
  - Bug fixing (do not develop for *fun* only)
  - Responsiveness
  - …

## JBoss Code Management

### Scott Stark

- CTO of JBoss, Inc.
- Clear-cut management:
  - “Negative” or “Affirmative”

# How do the team communicate?

Developers are spread all over the world:

- USA (Atlanta, Washington, Boston, …)
- France, Switzerland, Finland, Ukraine, …
- Brazil
- …

## How do we communicate?

Issues:

- Different time zone
- Different agenda
  - Not necessarily working on JBoss on a daily basis
- Different languages
  - English is used

“Comprehension is a specific case of Misunderstanding” (Bourdieu)

## How do the team communicate?

Mainly five channels

- Online Forums
- Mailing List (JBoss-dev)
- Private e-mails
- Instant Messenger
- Boot Camps

Wide topics,
Asynchronous,
Public audience

$\downarrow$

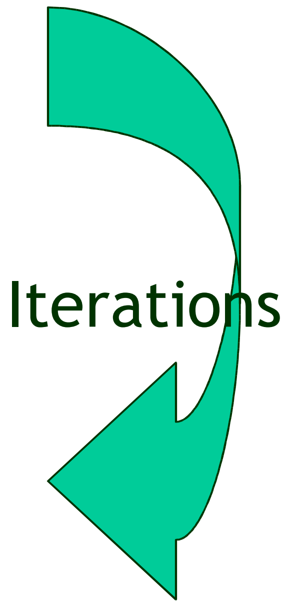
Highly focused,
Synchronous,
Face-to-face

## How do we develop JBoss?

### Lifecycle

- Proposal/requirements
- Analysis/design
- Implementation
- Tests
- Support

Remember something?



## How do we develop JBoss?

While it seems to mimic the development steps that exists in any commercial development, this is somehow misleading

- Requirements are not necessarily led by business cases (example: developer features)
- Analysis/design is reduced
- Implementation is part of the communication
- Regression Tests are very important in Open Source
- Iterations are MUCH shorter

## How do we develop JBoss?

People will not necessarily read your code… unless there is a bug!

Code must be easy to read, like a book!

Code at this point is somehow the design document!

Once you leave, other people will have to fix your bugs.

## How do we develop JBoss?

### Implementation

- Again: code must be easy to read
  - This is somehow part of the support requirements
- Very simple style guidelines
- Must be LGPL
- Developers keep their Copyright
  - JBoss or JBG do NOT own a copyright on the code

## How do we develop JBoss?

Tests

- Tests are CRITICALS
- Many developers are involved
  - In different sub-systems
  - At different time
  - For different reasons

Danger: creating new bugs while fixing others or implementing new features

## How do we develop JBoss?

### Tests

- To avoid this we have a big test suite
  - more than 1000 tests, each running many individual tests
- Runs automatically on all JBoss branches more than once every day
  - Different Hardware/OS
  - Different JVM
- Reports are automatically sent to JBoss-DEV mailing list

## When do we release JBoss?

JBoss has three branches:

- Branch_3_0: 3.0.x releases, stable
- Branch_3_2: 3.2.x releases, stable, minor evolution
- HEAD: development branch

## When do we release JBoss?

Release cycles

- Major releases: ~18 months (3.2 => 4)
- Minor releases: ~1 month (3.2.0 => 3.2.1)

We try to avoid too long development cycles ➔ risk management

- Hard to stabilize too many new features

## When do we release JBoss?

Release decisions are handled by Scott

- Decides when
- Decides about the freeze zone
- Decides what can go in after the freeze zone
- Does the actual release
- Publish it on sf.net

## Conclusion

- Mostly self-organizing structure
  - Self-rigor: nobody forces developers to code on JBoss
- But rules do exist!
  - Either explicitly or implicitly
- Avoid complexity: KISS
  - Short development cycles
  - Code readability i.e. code = knowledge

## II. JBoss and Professional Open Source

## What is JBoss?

JBoss is the premier Open Source java application server

JBoss Inc. employs the lead developers for the following projects
- JBoss Application Server: J2EE based
- Tomcat
- Hibernate
- JBossCache/JGroups:
- Nukes

JBoss Inc, the new safe choice
- Recent changed from JBoss Group to JBoss Inc.
  - Reflecting the $10 million VC investment deal
- 24/7 Support
- Indemnification
- Certification – J2EE, JASP

We call it “Professional Open Source”

## JBoss, Inc Customers

“JBoss Group’s people are super-smart and could help us at the technical level we needed without us having to work our way through levels of support staff. Compared with our old vendor, we get great support for relatively low cost.”

Jerry Shifrin, senior engineer, network management group, MCI (formerly WorldCom)

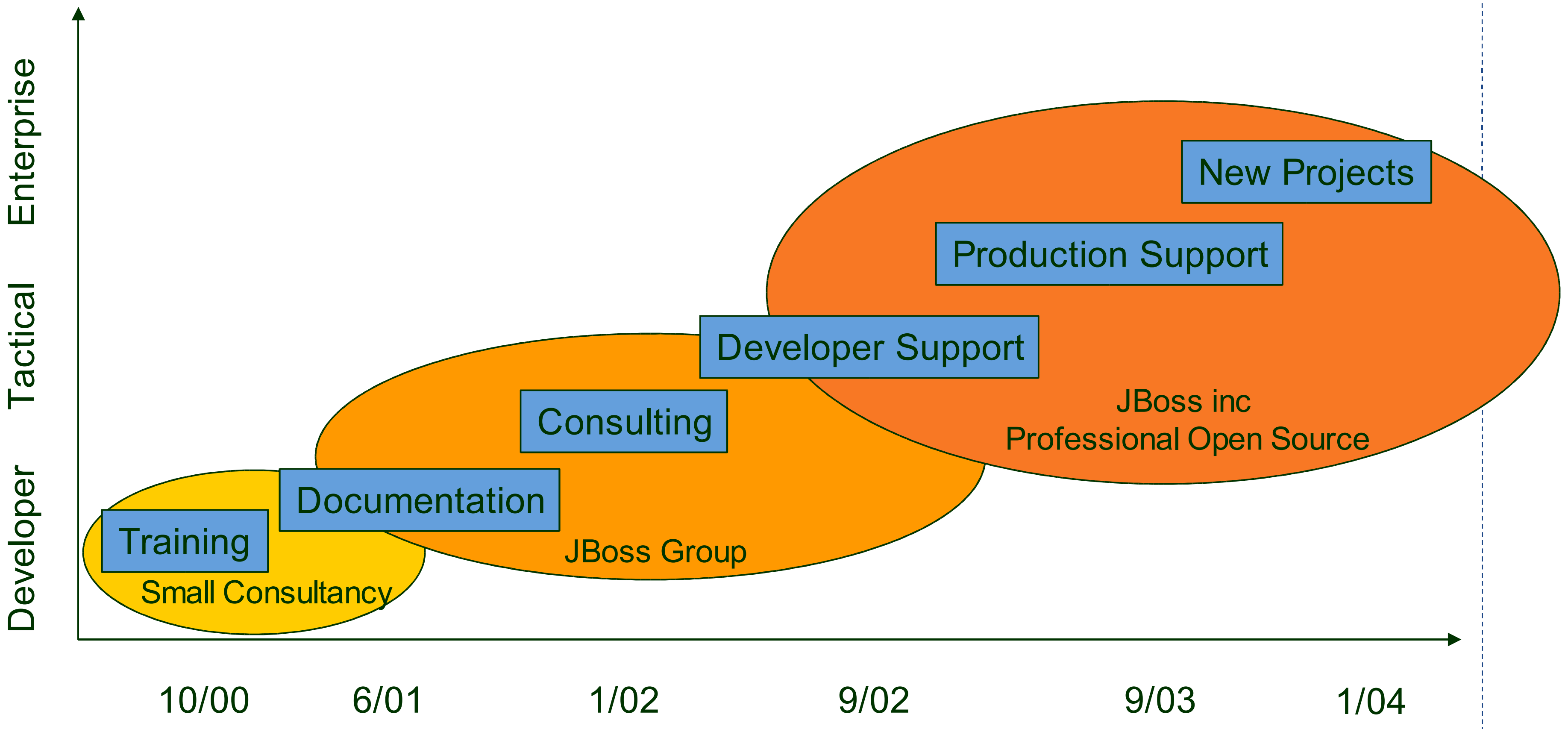
## CUSTOMERS (the tip of the iceberg)

accenture
AMD
BASF
CALIFORNIA ISO
Corporate Express
A Buhrmann Company
DOWJONES Indexes
Best Western
ca Computer Associates
Intuit
EDS
EA GAMES
GetThere
A Sabre Company
LastMinuteTravel.com
Time to Go!
LEAP FROG
LESSONLAB
HOME OF The SIMS
MCI
MCKESSON
Empowering Healthcare
COMPUWARE
Nielsen
Media Research
NORTEL NETWORKS
NUASIS
PLAYBOY.COM
QAD
SIEMENS
LION
WELLS FARGO
BOEING

## JBoss numbers

- A large community
  - 40,000 documentation sets sold
  - 500 contributors over time, 20 core (JBoss Inc)
- A standard in the market: #1 in development
  - More than 4M downloads in last two years alone
- A standard in the market: #1 in OEM
  - Analyst private communication
- A Standard for System Integrators #2 in growth
  - CRN survey puts JBoss certified consultant at #2 in fastest growing certification with large systems integrators
- A standard in the market: # 3 in production
  - BZResearch survey. 13% in 2002, 27% in 2003, largest growth of all servers
  - JDJ survey: 70% of users go to Deployment.

## Market Demand & JBoss Offering



## Professional Open Source

Revenue from Services

- Back Office model with EXPERTS (5% utilization)
- Focus on quality of service as sole source of income

Attract & Retain Top Developers

- Paid Open Source Development, boost to projects
- Support is “developer to developers”

Commercial Quality Code

- Control over the quality of source, dedicated resources

JBoss Group, the best support for JBoss

- Direct and unique chain of control in open source:

  Support → Bug Fix → Next Version

Expand Services offering

- Include support for Tomcat, hibernate and JavaGroup (JGroups)

## JBoss Production Support

Staffed with the lead developers of JBoss Inc. Projects

- Enables quick problem resolution from EXPERTS
- No navigating through levels of escalation

Prices range from $8,000 - $250,000

- Price determined by Service Level Agreement
- Number of named projects

NO PER CPU COSTS

- Eliminate procurement headaches
- Eliminate tracking of licenses
- Eliminate vendor audits
- Eliminate architecture decision based on cost of CPU licenses

## JBoss Production Support

| | PRODUCTION TRIAL | PRODUCTION 12 | PRODUCTION 2 |
|---|---|---|---|
| **Base price per year** | $8,000 | $25,000 | $40,000 |
| **Number of named applications included in base price** | 1 | 4 | 4 |
| **Price per additional named application** | $8,000 | $6,250 | $10,000 |
| **Target response time for production problems** | 24 hours | 12 hours | 2 hours |
| **Price for on-site production support** | – | – | $250/hour (min. 2 days) |
| **Development support included at no additional charge** | 5 hours | 20 hours | 20 hours |
| **Price for additional 10 hours of development support** | $2,000 | $2,000 | $2,000 |
| **Target response time for development problems** | 48 hours | 48 hours | 24 hours |
| **Available discounts** | – | 15% for 3-year term<br>25% for 3-year prepayment | |

## JBoss Group Value Proposition: 3 Year TCO Analysis

| COST COMPARISON | Zero TCO JBoss | | JBoss w/ Support | | Proprietary vendor | |
|---|---|---|---|---|---|---|
| | List Price | Extended Cost | List Price | Extended Cost | List Price | Extended Cost |
| **Production Licenses (100)** | $0 | $0 | $0 | $0 | $10’000/CPU | $1’000’000 |
| **Production and Development Support** | $0 Companies that leverage our software are not required to purchase support | | Production Support starting at $8’000 annually per project | $300’000 | 25% of list price for years 2 and 3 | $500’000 |
| **Three (3) years Total Cost of Ownership** | **$0** | | **$300’000** | | **$1,500’000** | |

## JBoss Authorized Service Partner (JASP)

Expand Partnerships & Channel

- ISV and OEM
- Systems Integrators
- Systems Vendors

Partner does 1st / 2nd line JBoss does expert 3rd line

- Leverage installed base of JBoss
- Leverage existing partner channels to increase service coverage
- Provide high level support with 1st line presence and 3rd line expertise.

## JBoss has licensed the TCK for J2EE 1.4

## Founders Program

- Partners who are helping JBoss with the Certification effort include
  - Borland
  - Iona
  - Intel
  - SchlumbergerSema
  - Unisys
  - WebMethods
  - Sonic

## JBoss the projects

**JBoss AS** Full J2EE support, EJB, JMX, JMS, JCA, JAAS

**Hibernate** O/R Mapping solution.

**Tomcat** JSP/Servlet/Web server.

**JBossIDE** Eclipse integration, tag driven development. Debugging.

**JBossCache** Distributed data.

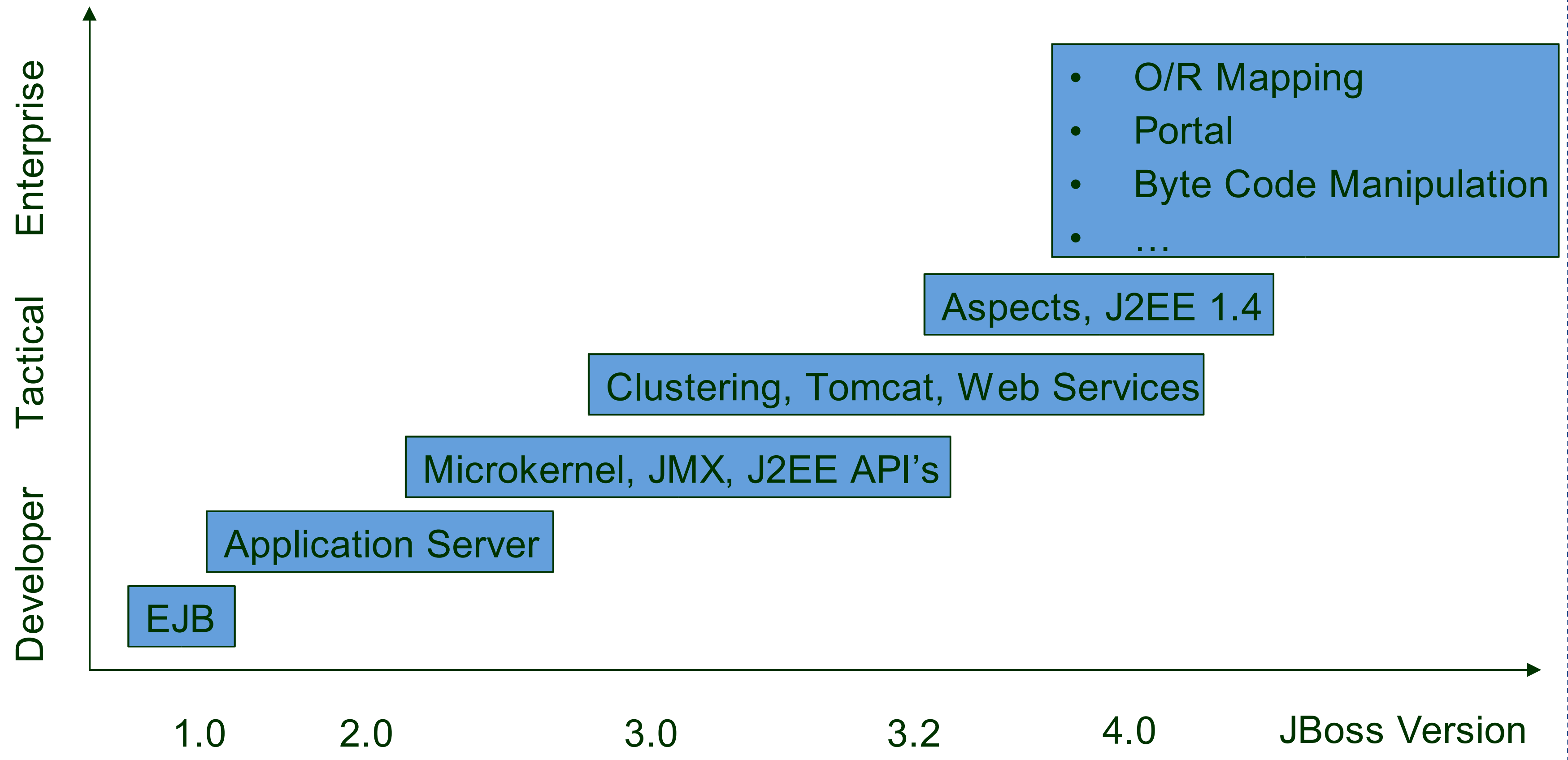
**JGroups** Reliable multicast and cluster communication

**Nukes** JBoss portal and CMS.

**JBossAOP** Aspect-Oriented Programming with JBoss 4.0.

**Javassist** Simple bytecode manipulation library

## Market Demand & JBoss Product Evolution



## What is Hibernate?

- Object/Relational Persistence Mapping for Java
- 1 ½ years old
- Now most popular Java O/R mapping library
- www.hibernate.org
- Why object/relational mapping?
- Solving the mismatch with tools
- Basic Hibernate features
- Hibernate query options

## *JBoss and Hibernate*

- Part of JBoss full-time
- Gavin King and Christian Bauer on board
- **Consulting and support available as part of JBoss inc**

Flexible and intuitive mapping

Support for fine-grained object models

Powerful, high performance queries

Dual-Layer Caching Architecture (HDLCA)

Support for *detached* objects (no DTOs)

Transparent Persistence

Automatic dirty checking

Transitive Persistence

Smart fetching and caching

- Hibernate is the back engine for CMP
  - CMP is an API and XML mappings
  - Hibernate is the actual persistence engine
- Hibernate caches are being integrated with JBossCache
  - Full distributed data with OR backend on one node

# Hibernate & J2EE

- Hibernate & JDO 2.0
- Hibernate & EJB3.0
  - Entity bean/ CMP

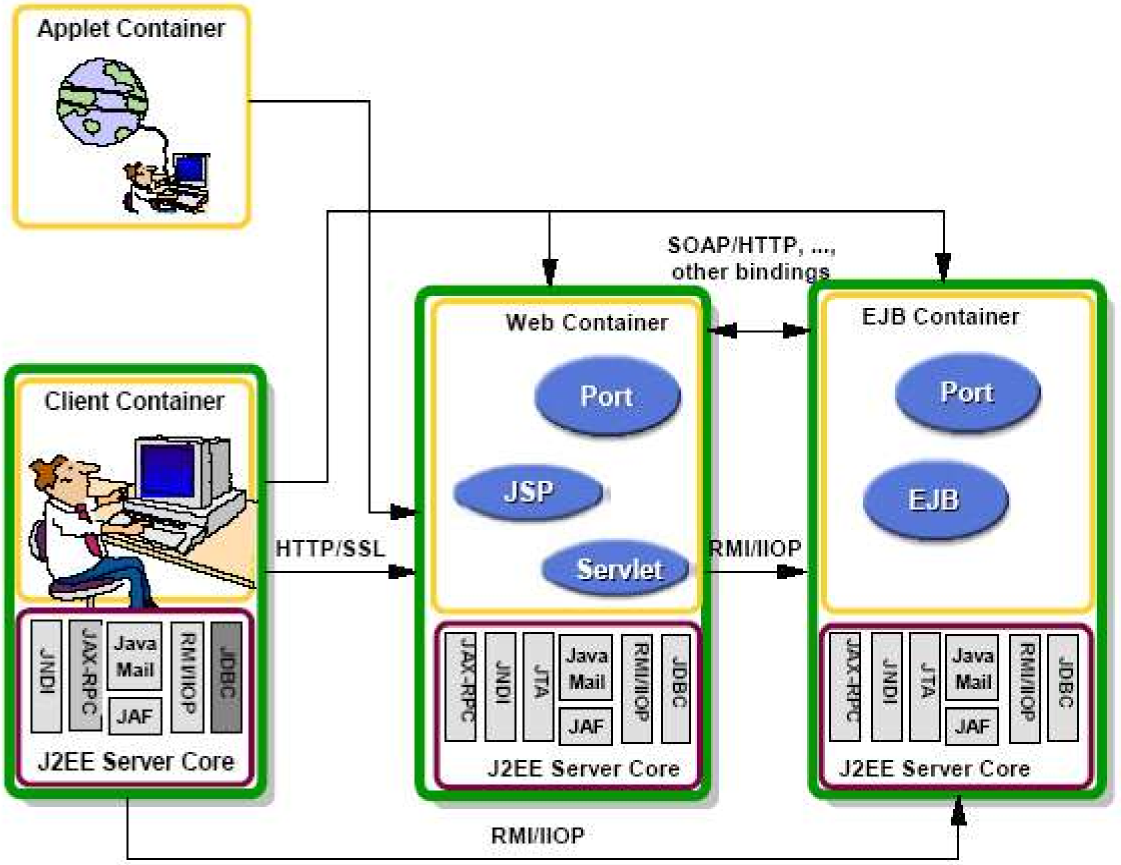
## Tomcat standalone or Tomcat inside JBoss ?

- Better JBoss deployer
  - Hot deployment
  - Deployment of nested archives (EARs, SARs)
  - Redeployment
  - Automatic undeployment
- Advanced clustering
- Integrated J2EE stack within one VM
  - Deployment descriptor
  - Optimized local calls
  - Integrated security
- AOP in JBoss 4.0 available in Tomcat components and webapps
- Easy to use classloader
- Nukes

## Web services in JBoss Overview: JBoss.net

Specifies how JBoss server components are exposed as Web service

- Stateless Session Beans
- Web components
- POJO as servlet



## What Is JBossCache?

- What is JBossCache?
  - A transactional replicated cache for JBoss with and without AOP (aspect-oriented programming)
- A cache for frequently accessed elements
  - Stateful Session Beans, HTTPSession
  - Caches are used in a number of places in JBoss
    - This one provides a central cache service (MBean interface)
- All access goes through the cache
  - Write-through (lazy or eager)
  - Reads only access the cache (very fast on cache hits)
  - Items not in the cache are loaded (e.g. from database)
  - Bounded size; old items are removed by eviction policy
- Local (=non-replicated) and replicated caches
  - Replicated caches are the interesting part

## So How Can We Use This Cache ?

### Use TreeCacheMBean directly

- Deploy treecache-service.xml
- Uncomment TreeCacheView MBean to have GUI
- Use the Facade pattern to provide access to TreeCache through a session bean. Clients use the session bean.

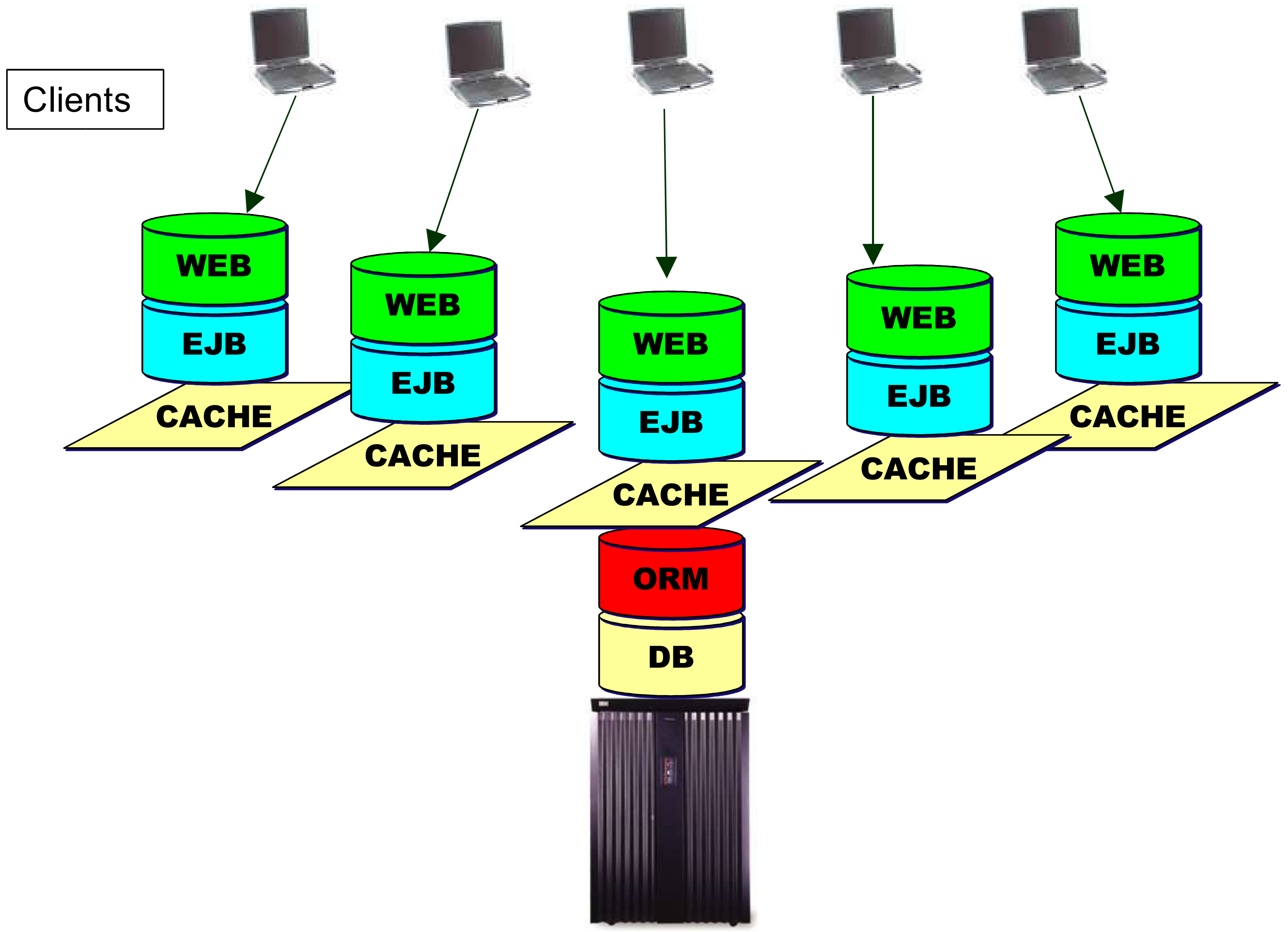
### Use TreeCache indirectly

- A service provided by the container and configured through deployment descriptors
- Example (to be implemented): entity bean cache, HTTP session replication

### Standalone TreeCache and TreeCacheAop

- A standalone version is also provided that includes the necessary libraries to use JBossCache
- Example: integration with Hibernate2.1

# Use Case: Replicated Cache for Entity Beans



## III. JBoss Architecture & AOP

## JBoss 4.0 Architectural overview

Microkernel design

- Independent cycling and loading

Hot Deployment of services and applications

- Unified ClassLoaders, total Class visibility/cyclability
- Service Archives (SARs) for easy configuration and net deployment

AOP Services

- Persistence, cache, transactions, acidity, remoteness, security
- Orthogonal aspects weaved in at run time under the objects
- In use in JBoss since 2.x series
- Generalized for public AOP consumption in the JBoss 4.x series
- NO COMPILER, FULL DYNAMIC DESIGN (byte code engineering)

With the introduction of a full-scale aspect oriented programming (AOP) framework, JBoss 4.0 brings high-level J2EE functionality, without J2EE complexity, to architects and J2SE developers.

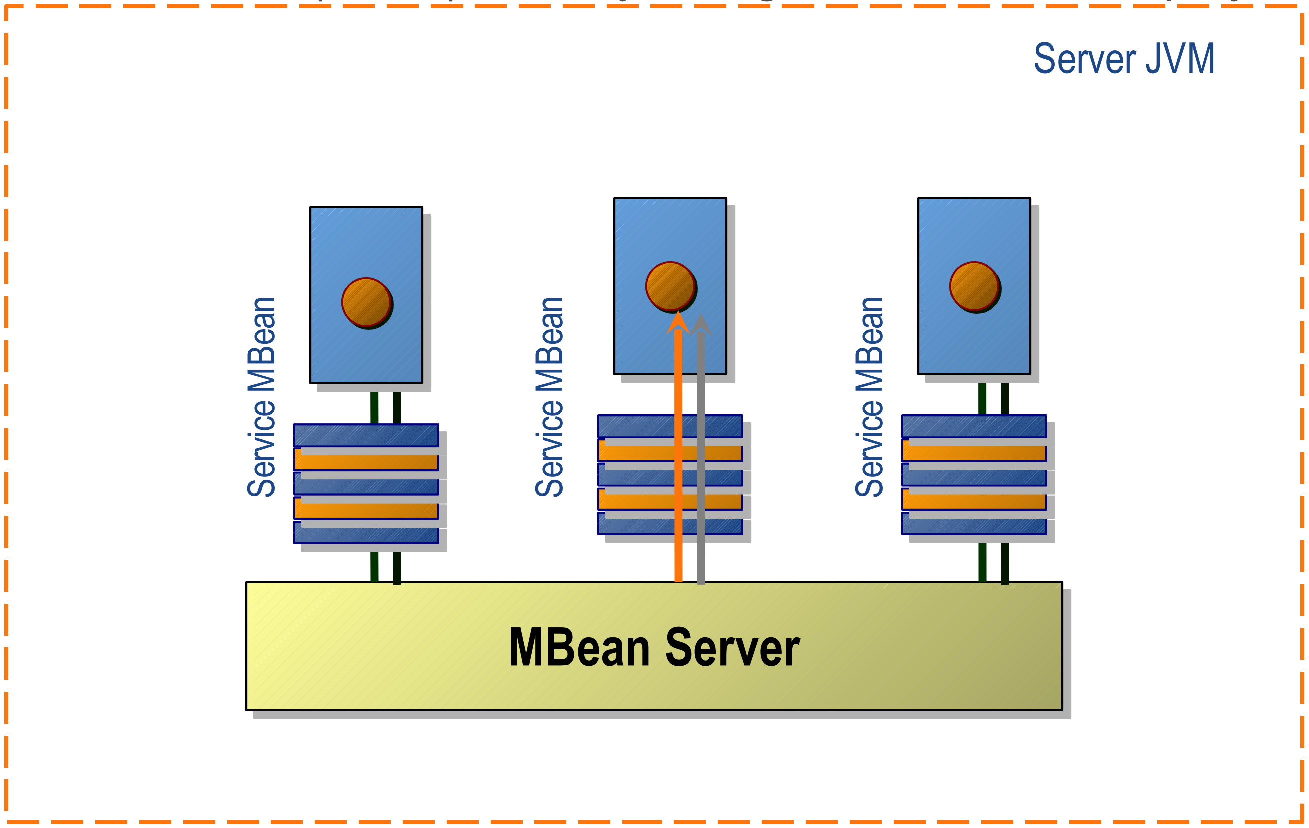
## JMX Architecture

Microkernel design

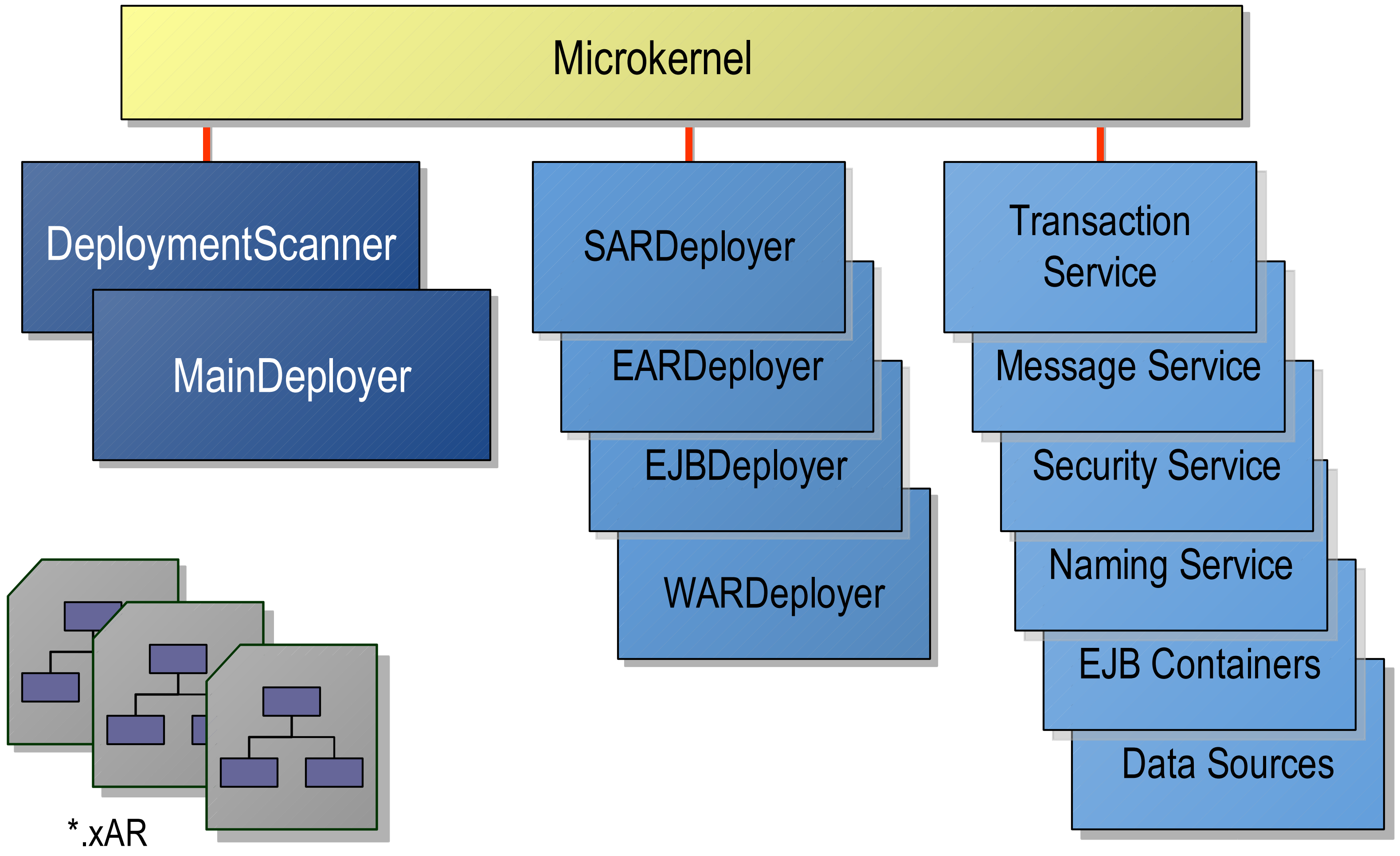
- Independent cycling and loading

Hot Deployment of services and applications

- Unified ClassLoaders, total Class visibility/cyclability
- Service Archives (SARs) for easy configuration and net deployment



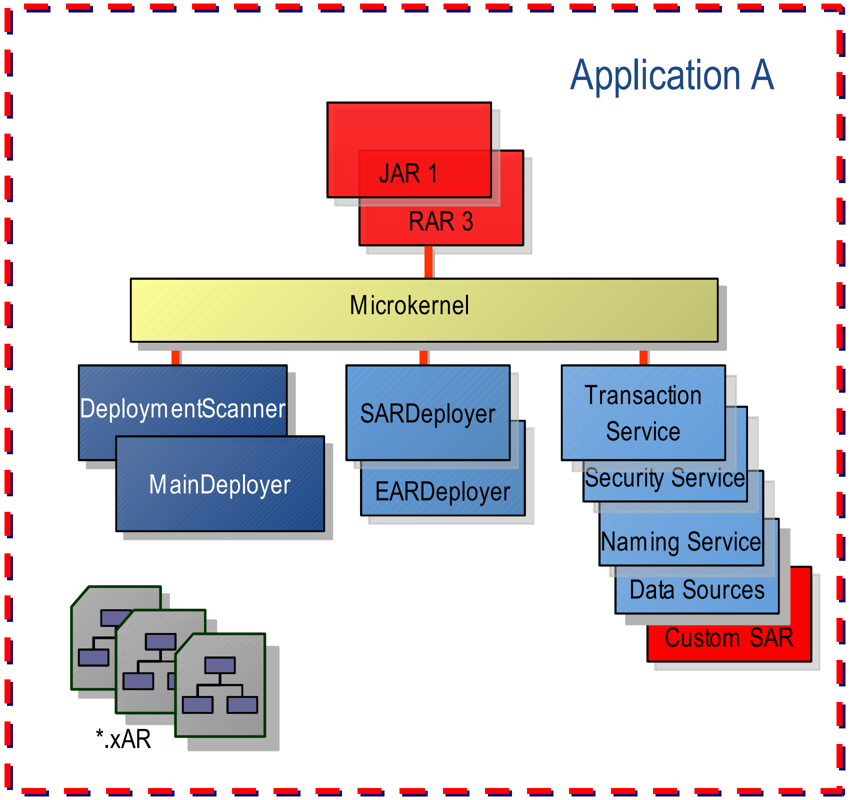
# Deployers: Bringing in the Services

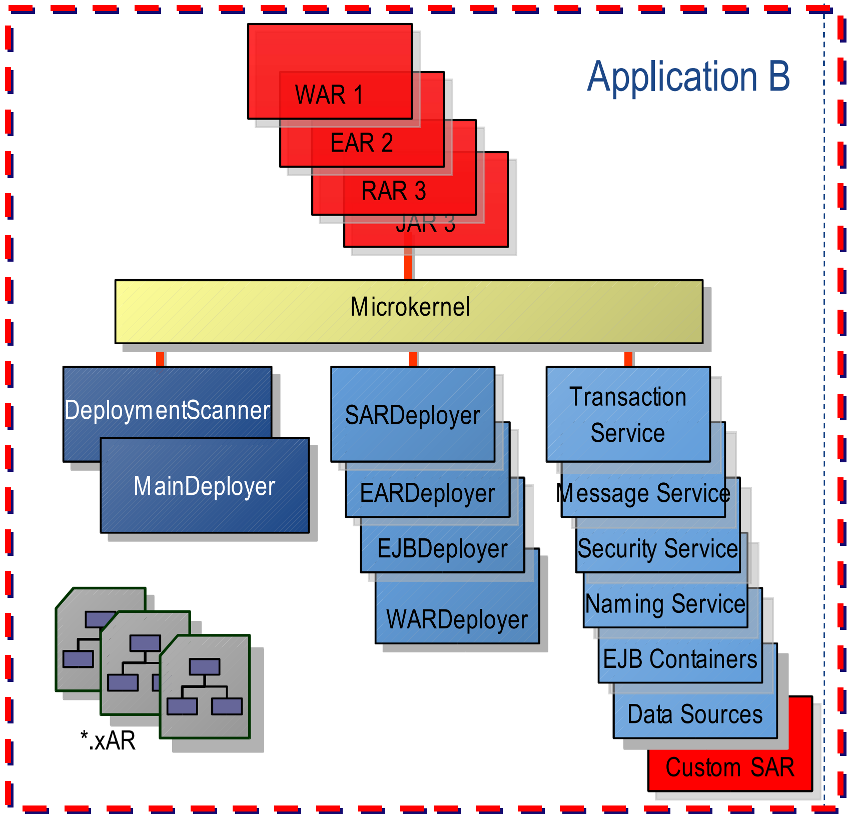


# Microkernel benefits for SOA architecture

## Microkernel approach ideal for ISV and OEM

- Easily remove the services you don't need
- Tight footprint and modular codebase and hot deploy/remove/redeploy
- JBoss is a TRUE Service Oriented Architecture (SOA)

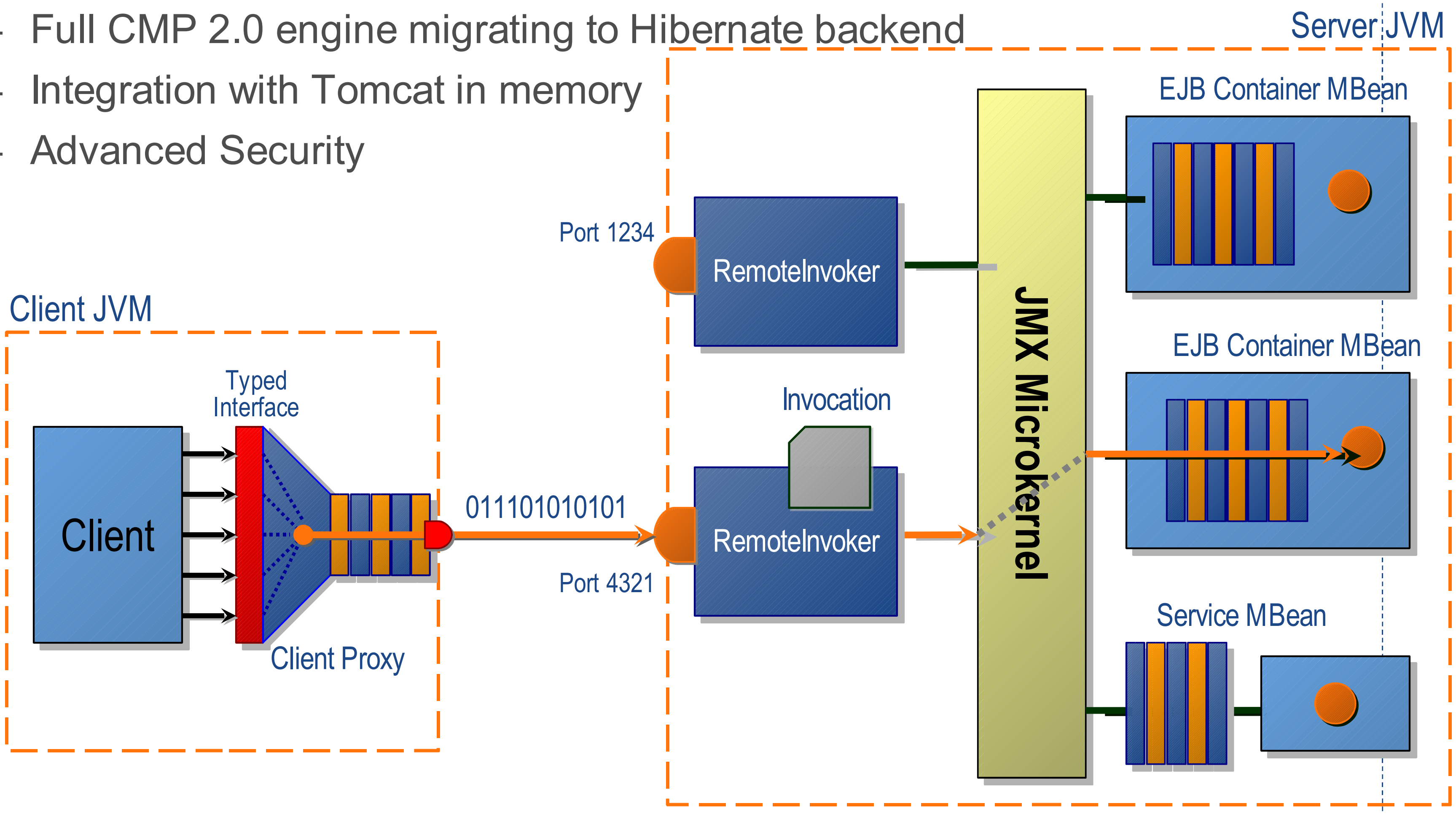




# JBoss 3.x Series EJB

## EJB 2.0

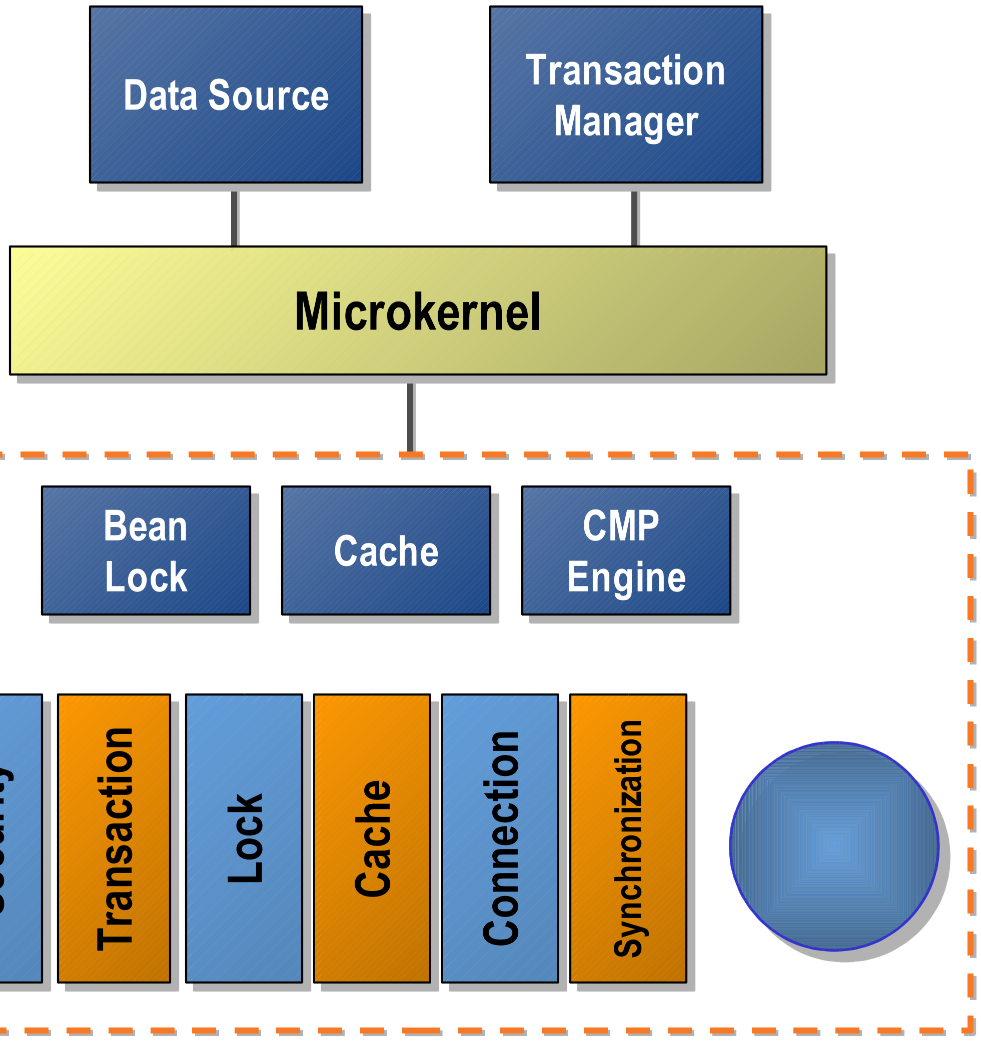
– No compiler approach (speed of development)
– Externalized stack of interceptors (AOP)
– Full CMP 2.0 engine migrating to Hibernate backend
– Integration with Tomcat in memory
– Advanced Security



## An EJB Container

An EJB container is the sum of

- Interceptors
- Plugins per container
- MBeans in the server



## Why roll out our own framework?

- Needed something that integrated seamlessly with our current EJB/JMX architecture and middleware
- Needed a dynamic runtime API
- Wanted hot deployment (we'll see this later in the demo)
- Wanted to work without a compilation step
- Integration with metadata
- Minor point: 100% Pure Java

## Basic Features:

- Execution pointcut definition for fields, constructors and methods
- Caller pointcut definition for constructors and methods
- Compositional pointcuts through an expression language pointcuts
- Interface introductions and mixins – Multiple inheritance with mixins
- Load or compile time bytecode enhancements
- XDoclet integration
- 100% pure Java
- Available integrated with application server
- Can run standalone in any Java program as well

## Distinguishing Features:

- Dynamic API
  - Advice bindings on a per instance basis
  - Metadata bindings on a per instance basis
- Hot Deployment of advice bindings at runtime
- Metadata facility (JSR-175 like annotations)
- Metadata pointcuts
- Configuration Domains
- Runtime GUI management console

## Terms

- Pointcut
  - Expression specifying a point within Java code
  - i.e. a method call, field access, construction
- Advice
  - Behavior you want to weave into your Java classes
- Aspect
  - Java class that encapsulates multiple advices
- Interceptor
  - An aspect with only one advice
  - Command pattern
- Joinpoint
  - Runtime encapsulation of a pointcut
  - Java event broken into its parts and encapsulated within an object
  - i.e. java.lang.reflect.Method, arguments, and target object
- Invocation
  - Class name of JBoss AOP’s joinpoints

# Metadata and Metatags

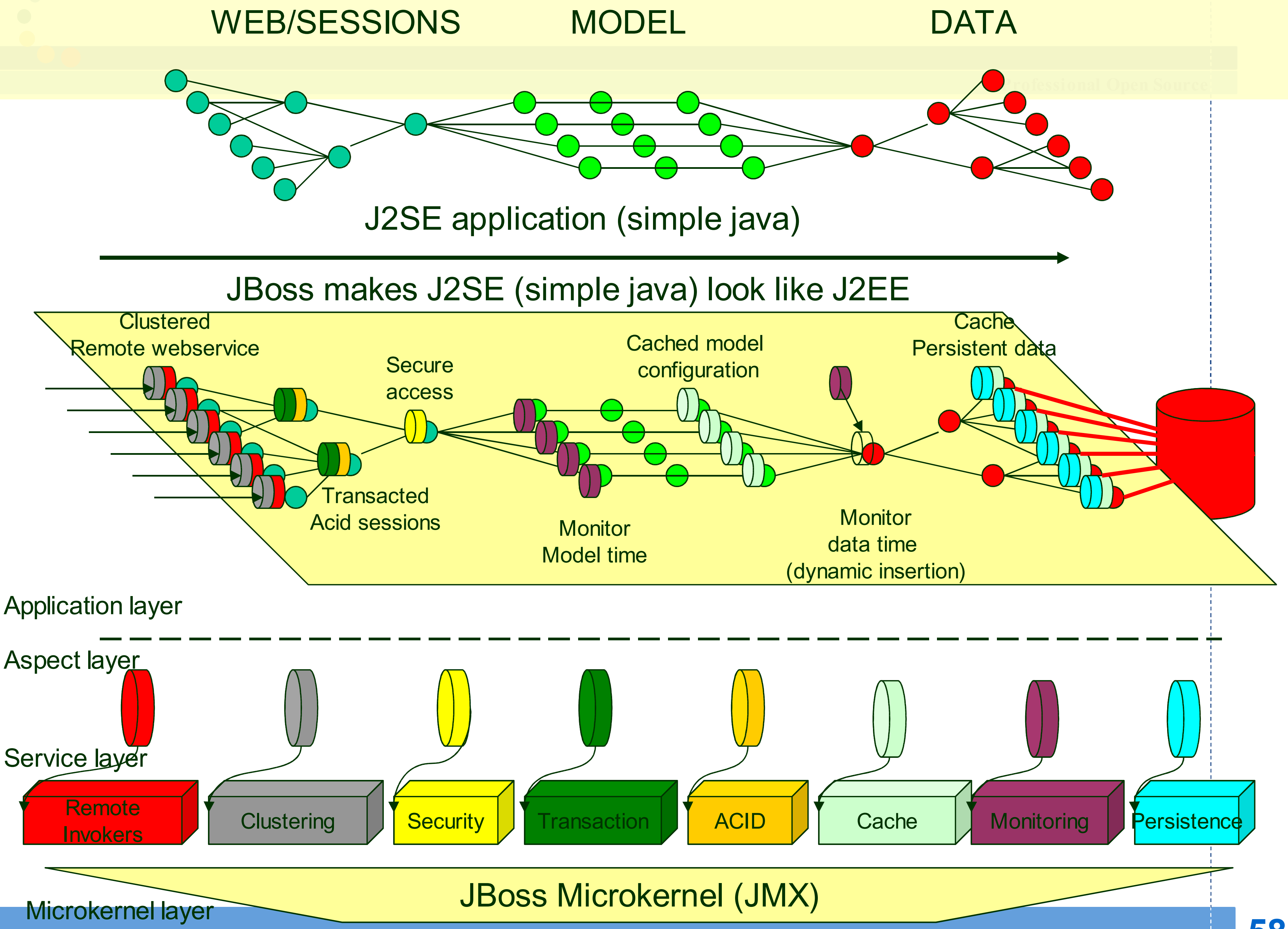
Aspect Configuration

Explicit through Xdoclet tags (later JSR-175)

Implicit through XML (deployable at runtime)

```
/**
 *
 * @jboss-aop.metadata group="transaction" trans-
attribute="RequiresNew"
 */
 public void somePOJOmethod() { … }
```

```
<class-metadata group="transaction" class="com.acme.POJO">
      <method name="get.*">
          <trans-attribute>RequiresNew</transattribute>
      </method>
 </class-metadata>
```


WEB/SESSIONS
MODEL
DATA
J2SE application (simple java)
JBoss makes J2SE (simple java) look like J2EE
Clustered Remote webservice
Secure access
Cached model configuration
Cache Persistent data
Transacted Acid sessions
Monitor Model time
Monitor data time (dynamic insertion)
Application layer
Aspect layer
Service layer
Remote Invokers
Clustering
Security
Transaction
ACID
Cache
Monitoring
Persistence
JBoss Microkernel (JMX)
Microkernel layer

## JBoss 4: Pre-packaged Aspects

Middleware, by nature, is cross-cutting

Middleware implemented as Aspects allow for:

- Smooth, fluid, iterative development
- Clean separation between System Architect and Application Developer
- Less upfront design decisions

JBoss 4 is Aspect-Oriented Middleware

Specifications like EJB requires upfront design decisions

AOP provides clean separation from system architecture and application code

Architectural decisions can be made later on in the development process

AOP makes iterative development more fluid

Is it simplified EJB?

## JBoss 4: Pre-packaged Aspects

### Dynamic AOP

- Transactional, ACID, Objects.  Our Transactional Cache
- Replicated Objects.  Our Distributed Cache
- Optimized HTTP Session Replication
- Remoting – choose at runtime, SOAP, RMI, Sockets, IIOP
- Clustered Remoting – invocation failover

### Metatag based aspects:

- J2EE a la carte
  - Transaction demarcation
  - Role-based Security
- Transactional Locking.  Expanded Java “synchronized”

## AOP Cache

- AOP Cache
- POJO inserted into cache
  - Can become Transactional
  - Can become Replicated
  - Depends on Cache Configuration
- Goal to have transparent ACID properties
- Transparent Replication
- No application coding for inserted Objects
- Uses AOP Dynamic API
- Requires “prepare” step via <advisable>

## Work with POJOs

```
public class Person {
   String name=null;
   int age=0;
   Map hobbies=null;
   Address address=null;
   Set skills;
   List languages;

   public String getName() {
      return name;
   }

   public void setName(String name) {
      this.name=name;
   }
   ...
}
```

```
public class Address {
   String street=null;
   String city=null;
   int zip=0;

   public String getStreet() {
      return street;
   }

   public void setStreet(String street) {
      this.street=street;
   }
   ...
}
```

## Interaction With Cache

```
tree = new TreeCacheAop();

config = new PropertyConfigurator();
// configure tree cache.
config.configure(tree, "META-INF/replSync-service.xml");

joe = new Person();
joe.setName("Joe Black");
joe.setAge(31);

addr = new Address();
addr.setCity("Sunnyvale");
addr.setStreet("123 Albert Ave");
addr.setZip(94086);

joe.setAddress(addr);
```

**Use Pojos as Pojos**

## Interaction With Cache

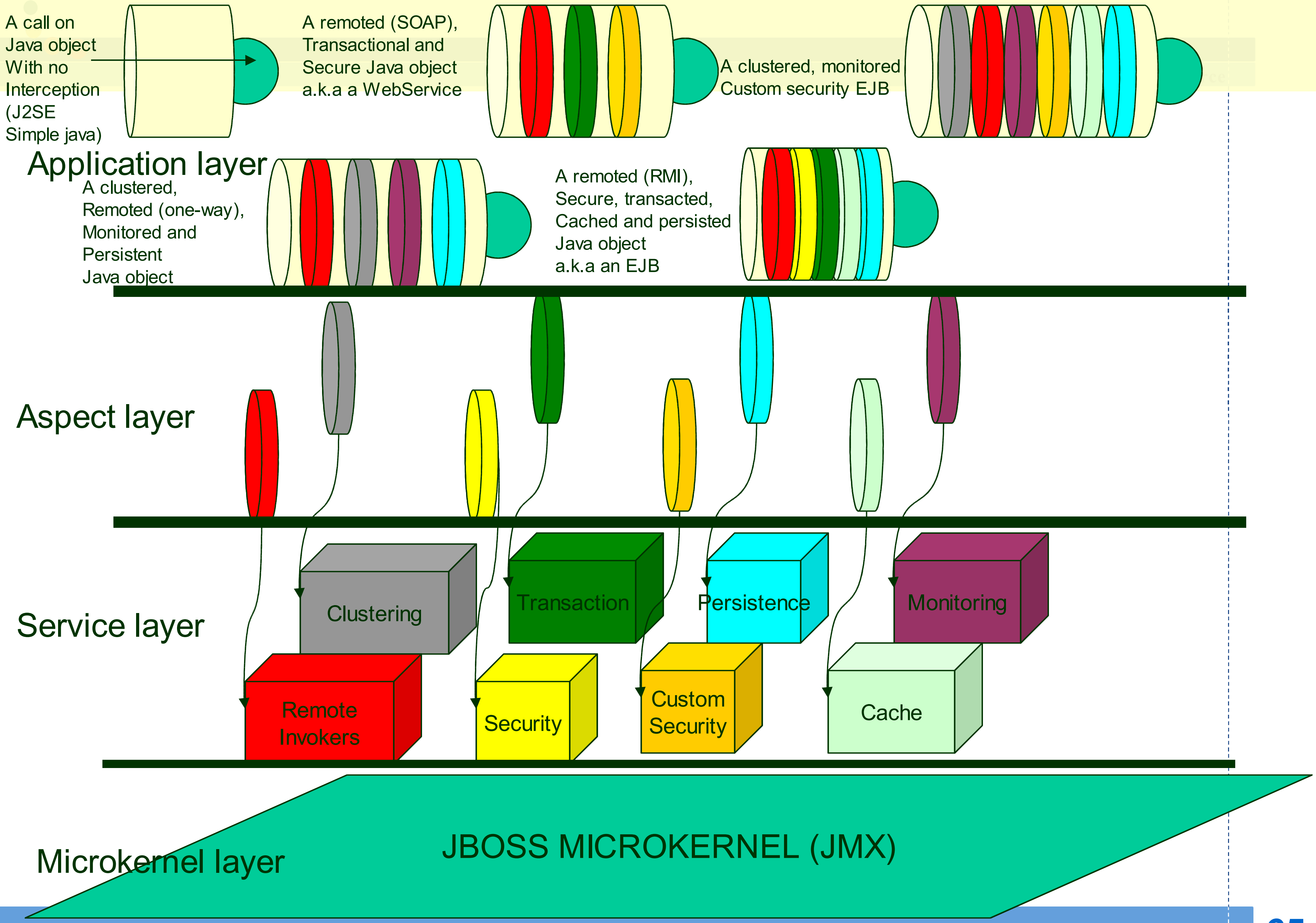
Joe's state is automatically transactional and replicated

State replicated, synchronized at transaction commit/rollback

```
tree.start();   // kick start tree cache
tree.putObject("/aop/joe", joe); // add aop sanctioned object



tx.begin();
joe.setAge(41);
joe.getAddress().setZip(95124);
tx.commit();
```


A call on
Java object
With no
Interception
(J2SE
Simple java)
A remoted (SOAP),
Transactional and
Secure Java object
a.k.a a WebService
A clustered, monitored
Custom security EJB
Application layer
A clustered,
Remoted (one-way),
Monitored and
Persistent
Java object
A remoted (RMI),
Secure, transacted,
Cached and persisted
Java object
a.k.a an EJB
Aspect layer
Service layer
Clustering
Transaction
Persistence
Monitoring
Remote
Invokers
Security
Custom
Security
Cache
Microkernel layer
JBOSS MICROKERNEL (JMX)

Website: www.jboss.org and www.jboss.com

Email: ben_wang@jboss.com

THANK YOU!

And remember we love you