



Sponsored by Platinum Solutions, I

Putting The Pieces Together

Presented by: Michael Nash (mnash@jglobalonline.com)

NOVAJUG Enterprise SIG

Leveraging Multiple Frameworks and the best of Open Source projects for web application development. Sponsored by Platinum Solutions, Inc.

Introduction

Putting the Pieces Together

Who am I?

- My name is **Michael Nash**
- President and Lead Developer of JGlobal Ltd. (<http://www.jglobalonline.com>)
- A Core developer of Keel meta-framework open source project

- Author of "Java Frameworks and Components: Accelerate Your Web Application Development"
 - Part of the JSR 127 (JavaServer Faces) Expert Group
 - Former lead developer of the Espresso Framework open source project
-

Overview

- Java APIs are powerful, but generally low-level and complex.
 - Many toolkits and frameworks exist, with various levels of overlapping functionality
 - Most of the frameworks designed for web application development are open source, with all of the attendant benefits and risks.
 - How can the best features from all of these capabilities be used together in a practical way?
-
-

Frameworks and Components

Definition and Description

What's a Framework?

A certain kind of tool for creating applications:

- A set of design patterns
- A set of interfaces and implementations
- Related tools

Why Frameworks?

- Developer Productivity
 - Consistency and resulting quality
 - Rapid adaptability
 - Design patterns and well/widely used components
-

Meta-Framework

A toolkit designed to:

- Span, bridge and connect together multiple application frameworks and toolkits.
 - Allow multiple component models to interact efficiently
 - Allow the features and benefits of one framework and toolkit to be used in another.
 - Allow J2EE/EJB to be used transparently when appropriate
 - Use the best component for the job, no matter what framework or architecture it is from
-

Non-Framework Toolkits

There are many toolkits available that are not specifically frameworks, but are often used in conjunction with frameworks to make your jobs easier.

JSRs: JavaServer Faces, JDO, many others

Tools such as Xerces, Xalan, FOP, Batik

And of course, the essential "Ant"

Components

What Is A Component?

"An entity that fulfills a specific role or provides a specific service".

Should be replaceable with another component that fulfills the same role

How to Choose the Best?

Criteria include:

- Organization
- Documentation
- Source quality
- Fit

- Size
 - Complexity
 - Flexibility
 - Interoperability
-

How to get them to work together?

(With each other, and with existing systems)

Problems:

- Different component models
 - Conflicting dependancies
 - Incompatible object hierarchies
-
-

Frameworks/Components and Open Source

The Business Advantages

Why Open Source?

- Not a philisophical decision - a financial and business one.
- The decision to use open source is a matter of balancing the risks and rewards

- Development projects usually need to solve the same kinds of problems, at least in part.
 - Open Source entails various kinds of risks and rewards
 - Some kinds of projects benefit more from the open source method than others do
-

Open Source: Risks

Risks vs. Rewards: Using open source in development is a matter of comparing risks and rewards.

- Quality
 - Continuity
 - Support
 - Learning Time/Cost
 - Rapid development - or very slow development!
 - Lock-in
 - Lack of Documentation
 - Licensing Issues
-

Open Source: Rewards

- Zero or low acquisition cost
 - Quality
 - Source availability
 - Developer Community: Free (but perhaps unreliable) Support
-

Open Source Business Model

- A different kind of cost to users
 - Time: finding, choosing, learning
 - Contributions, feedback
 - Risk

 - Developers and contributors receive a different kind of compensation
 - Peer Recognition
 - Influence Direction of Project
 - Opportunity
 - Related Services
-

"Pioneering"

Definition of a Pioneer: The guy up front, face down, with an arrow in his back.

Commercial development requires innovative, modern, but stable and reliable solutions.

The bleeding edge is too sharp to dance on.

Open source is a valuable proving ground for new technologies and techniques.

Open Source Frameworks

- **Struts:** Presentation-oriented, includes powerful tag library for JSP
 - **Avalon:** Architecture-level framework, component management, pluggable implementations
 - **Cocoon:** XML publishing and handling, portal-building, complete applications
 - **Turbine:** Moving to Avalon-based architecture, multiple presentation and persistence options
 - **Expresso:** Complete application framework, monolithic, database-oriented, integrated with Struts
 - **Velocity:** Presentation, simplified template language, complete separation of logic and view
 - **OpenSymphony:** J2EE/EJB-oriented, a number of sub-projects
-
-

Apache Avalon

Flexible component management and foundation for Keel

Avalon



Formerly the Apache JServ Project

Keel builds on top of the Avalon project

Avalon consists of a **Framework** project and a number of optional sub-projects

Avalon Framework

- Component Management by Container
 - Inversion of Control
 - Separation of Concerns
 - Lifecycles and Lifecycle extensions
-

Avalon Sub-Projects

- Excalibur
 - LogKit
 - Phoenix
 - Cornerstone
-

Excalibur

Component Collection for Avalon Framework

- **Containers:** ECM, Fortress, Merlin, Tweety
- **Components:** Cache, DataSources, Collections, Pooling, i18n, many others

- **Resource Managers:** Event, Logger, Testcase
 - **ScratchPad:** Components under development
-

Cocoon



A highly sophisticated example of an application built with Avalon.

Cocoon calls itself an "XML Publishing Framework", but it is far more...

- Avalon-based extensibility and configurability
 - Access virtually any data source
 - Run as servlet or standalone
 - XML-based, but allows other presentation technologies, such as JSP, Velocity, etc.
 - Includes transformers to create HTML, XHTML, PDF, WML, SVG, and many more
 - Suitable for building static or dynamic websites, portals, web applications, web services, and more
-

Roles and Component Oriented Programming

A role is a **specific component definition**, as specified by an interface. For example:

```
public interface Persistent {
    public void save();
    public void load();
    public void find(Object key);
    public void setProperty(String propertyName, Object propertyValue);
    public Object getProperty(String propertyName);
}
```

Lifecycle Interfaces

An extensible set of lifecycle interfaces are defined by the containers in Avalon. A container's purpose is to take components through their defined lifecycles.

- LogEnabled
 - Contextualizable, Recontextualizable
 - Configurable, Reconfigurable
 - Parameterizable, Reparameterizable
 - Initializable
 - Startable
 - Suspendable
 - Stoppable
 - Disposable
 - SingleThreaded
 - Poolable
-

Implementation of the Persistent Role

A component is simply an object which implements a role, and zero or more lifecycle interfaces:

```
public class FilePersistent extends Foo
    implements Persistent, Servicable, LogEnabled, Configurable {

    public void save()...(and other Persistent methods)
    public void service(ServiceManager newManager)...
    public void enableLogging(Logger newLogger)...
    public void configure(Configuration newConfig)...
}
```

Avalon Disadvantages

- Highly complex
 - Difficult to learn - too many options
 - Relatively low-level
 - Definitely "some assembly required"
 - Large and diverse development community - many branches and "forks"
 - Politics in the developer community
 - Lack of Documentation
-
-

Keel

A Meta-Framework

Keel Design Objectives

- Don't re-invent the wheel - invent the axle
 - Use free bricks to build our houses wherever prudent
 - Open source that already works - ready to use for business applications
 - Maximize re-usability of components, no matter where they're from
 - Ability to select the best components and services available
-

Overview of Keel



- Based on Avalon
- Higher-level services
- Focus on cross-framework interoperability
- Series of interfaces, each with multiple implementations
- Business-application oriented
- Unit and functional tests
- Powerful Ant-based build system for generating complete web and/or enterprise applications (war/ear deployment)

J2EE/EJB

Aren't Keel's goals overlapping with those of J2EE/EJB?

- EJB is one choice of component architecture, not the only choice, not always the right one.
- Application of the correct amount of force: use the portions of the J2EE spec that are appropriate to the job at hand
- A framework should not tie components to a particular architecture

Primary Keel Capabilities

- Strict separation of roles and interfaces - multiple implementations for each role
- Model, View, Controller (in fact MVC "2" approach)
- Multiple presentation frameworks: complete independence of logic and interface
- Multiple persistence mechanisms
- Auto-configuration

Current Status of Keel

Keel at an early but usable stage now: Release 1.0 within weeks

Using Avalon: a mature, stable and flexible base.

Keel applications already deployed in a commercial environment

Currently enabling integration between 8 frameworks

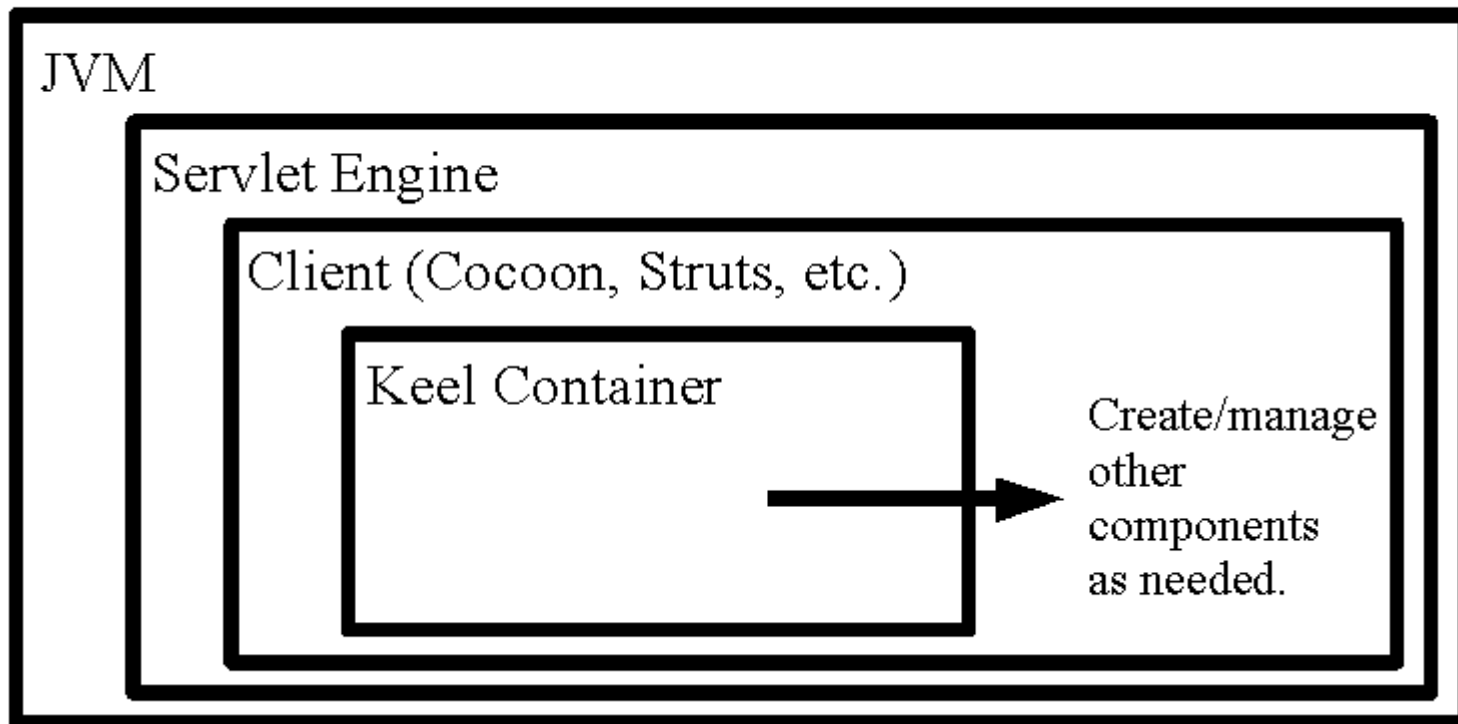
Current Service Interfaces and Implementations:

- **Persistence:** JDBC, Entity EJB, JDO (alpha), Apache Object-Relational Bridge
 - **Scheduling:** Quartz, Jcrontab
 - **Model (Application Logic):** Default (Local), Session EJB, Message-driven EJB (MDB)
 - **Encryption:** Base64, DES
 - **UI Interfaces:** Struts, Cocoon, Command-line, Axis (Web Services), Velocity (alpha)
 - Many others...
-

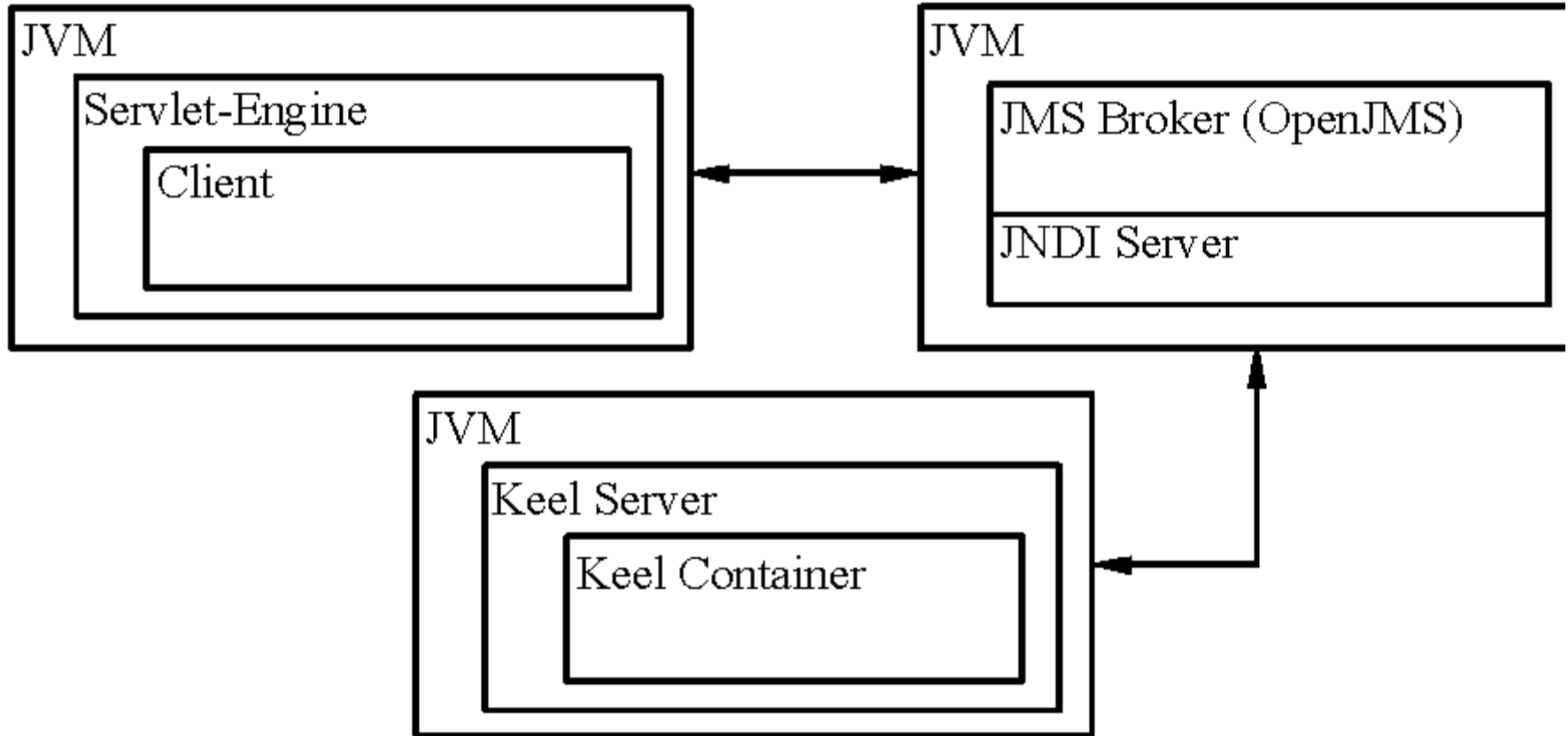
Keel Example Applications

- Poll/Vote Application
 - CRUD - Data Maintenance
 - Registration/Login (JAAS-Enabled)
 - Under development: Navigation
-

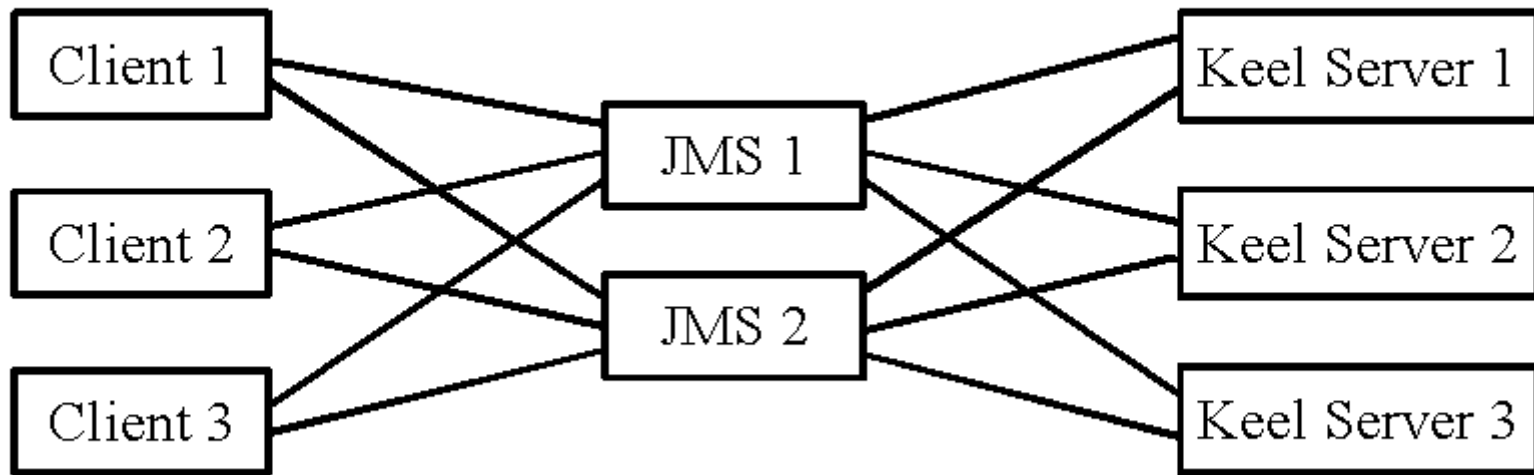
Single VM Deployment



Multiple VM Deployment



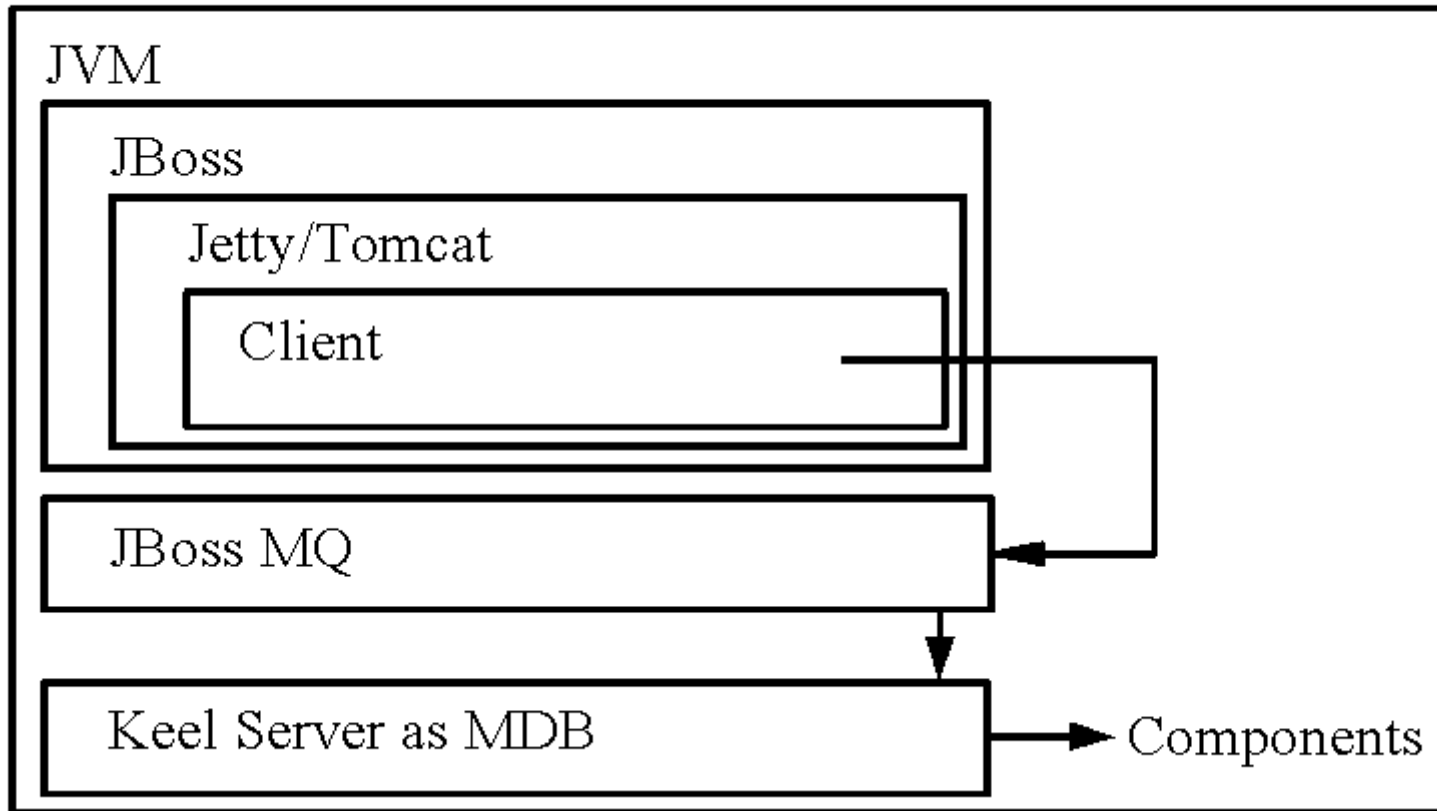
Clustered Example

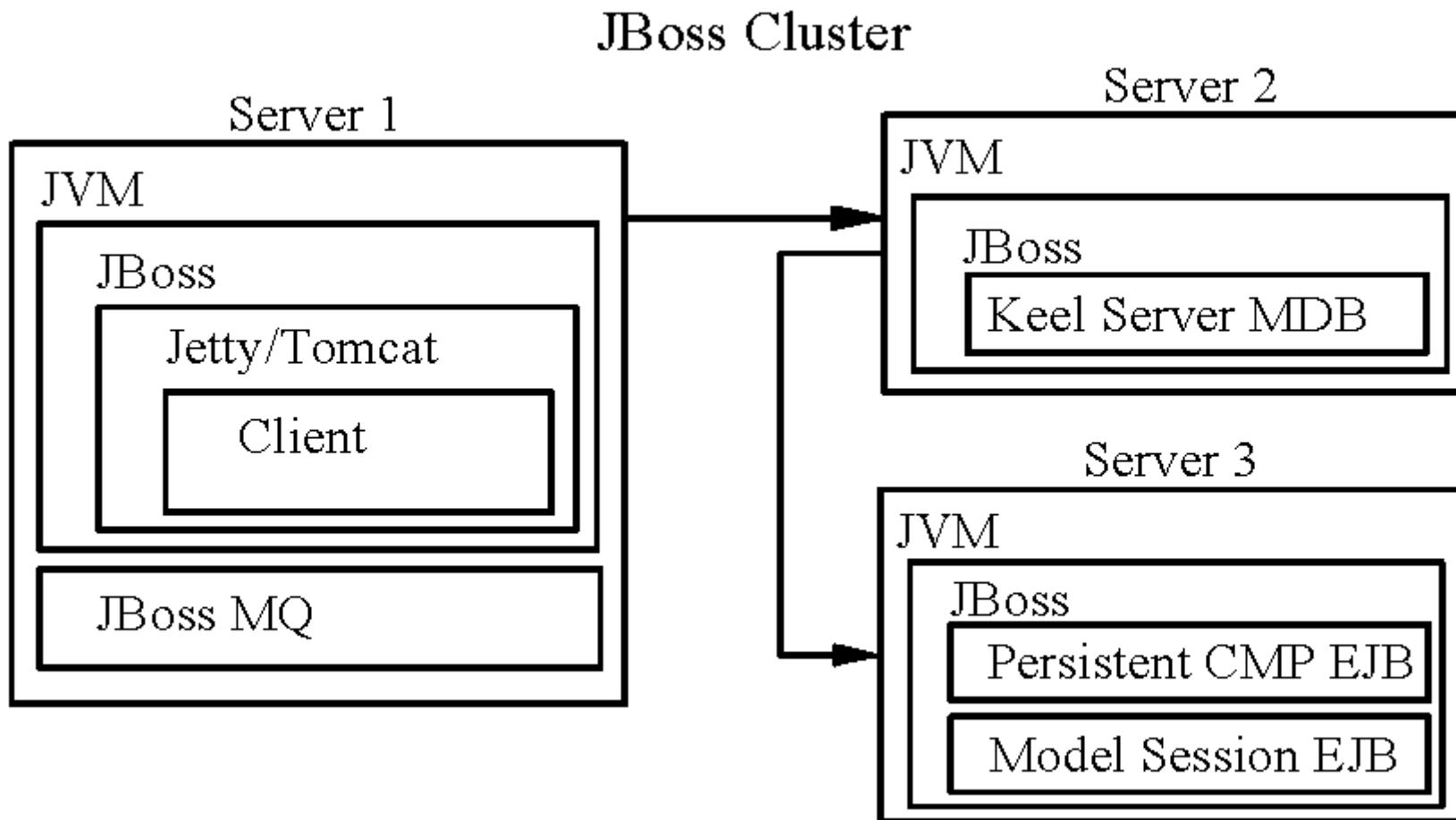


Keel and JBoss

- Ability to deploy Keel as a Message-Driven EJB in JBoss.
- Custom Ant tasks for creating .ear archives.
- Application logic can be Session EJBs
- Persistence can be CMP Entity EJBs

Single VM JBoss Deployment





Invitation to Join Keel Development

There are many levels of "contribution" to open source projects:

- Reviewers providing input, suggestions, comments: Feedback drives new interfaces
 - Users providing feedback, bug reports, etc.
 - Developers providing new implementations of defined interfaces.
 - Developers providing new interfaces to other services and frameworks.
-
-

Conclusion

What works and what doesn't

Summary

Used properly and prudently, the benefits of open source can outweigh the risks.

Frameworks and related toolkits are particular good candidates for this.

Not every framework is right for every project, and none have all the answers.

The best solution I've found for more high-end web applications is to use a meta-framework to combine multiple frameworks & use the best of breed tools for each area of my application.

Followups: Where to go from here?

Book, Courses, and Projects

Book

Java Frameworks and Components: Accelerate your Web Application Development

- Available for pre-ordering from Amazon, etc (ISBN: 0521520592)
 - Comparison and Analysis of over 35 open source and commercial frameworks
 - How to find, choose and use frameworks and component technologies
-

Courses/Seminars

Information sheets available

- Developing Keel Web Applications

A technical introduction to the use of multiple open source technologies applied to the issue of web application development, using the Keel meta-framework as an enabler for integration.

- Building XML Web Applications with Cocoon

An introductory course in Cocoon with an emphasis on building flexible web sites, both static and dynamic, targeted at multiple end-user devices and displays.

- Struts

Introductory to Expert courses in Struts development.

- Got Web? What's next?

A Managers introduction to interactive internet applications.

Keel and Related Projects

- Keel: www.keelframework.org: info on mailing list on the site
 - Avalon: jakarta.apache.org/avalon/index.html
 - Struts: jakarta.apache.org/struts/index.html
 - Cocoon: xml.apache.org/cocoon/index.html
-

Questions and Answers

Feel free to email further questions/comments to mnash@jglobalonline.com

This presentation prepared with the Cocoon XML Publishing framework
