Baltimore
Linux Users Group

Apache Geronimo

J2EE Server and more

Aaron Mulder
Chief Technical Officer
Chariot Solutions

CHARIOT SOLUTIONS

About the Speaker

- Working with Java since Java Alpha
- Open Source contributor to Geronimo, JBoss, OpenEJB, PostgreSQL JDBC drives, etc.
- Author, speaker, member of JSR-88 EG
- CTO of Chariot Solutions
  - Focused on Java and related technologies
  - Technology Architecture, Strategy & Mentoring
  - Application Diagnostics, Rescue Missions
  - Full Lifecycle Software Development

About Apache

- The Apache Software Foundation (ASF), founded in 1999
- 501(c)3 non-profit corporation
- 1100+ committers, 25+ major projects, numerous subprojects
- Apache License v2; IP procedures
- Focus on project communities

About Geronimo

- Original intent was to build a J2EE application server
- Turned out we needed a foundation to build a server on
- Now Geronimo consists of a general service container, with J2EE and other services

Agenda

- Geronimo: The Container
- J2EE compared to LAMP
- Geronimo: The J2EE Application Server
- Current Status, TODOs, & Contributing to Geronimo

Baltimore
Linux Users Group

Geronimo: The Container

CHARIOT SOLUTIONS

Why A Container

- Geronimo needs to be modular
- Modules must have a lifecycle
- Something must know what modules are available, which ones are running, how they relate...
- Modules must be able to work with each other

Why A New Container

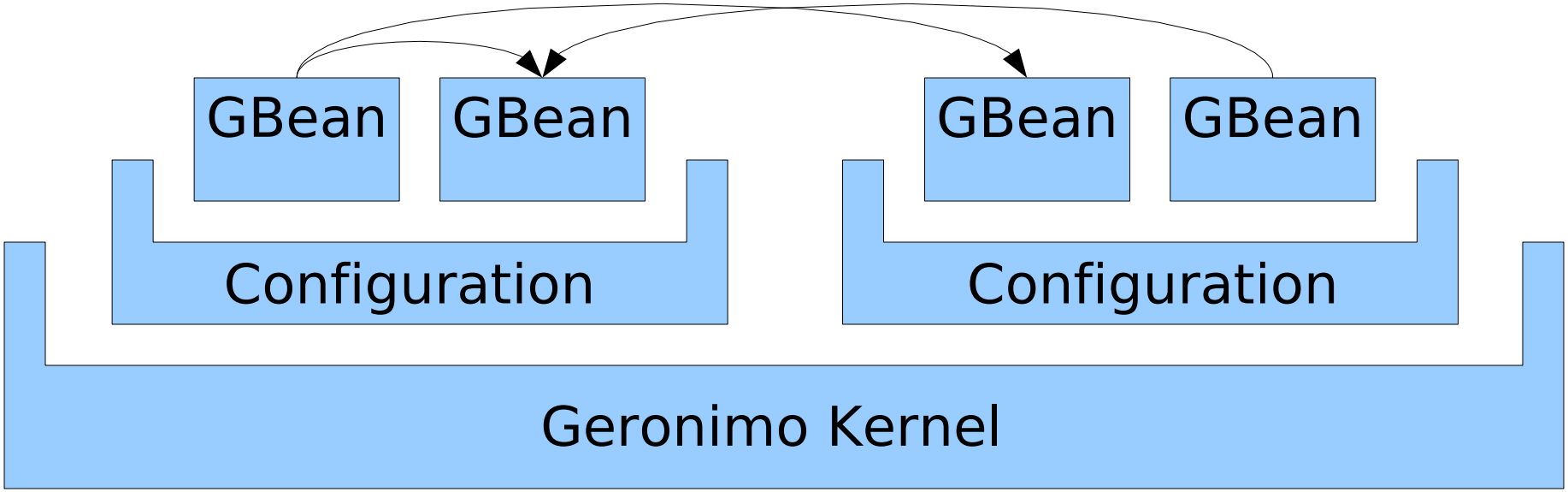
- Nano/Picocontainer, Avalon...
- Geronimo needs core JSR-77 (management) support
- Geronimo needs to be able to hot deploy/save/undeploy modules and applications
- Geronimo needs rich module configuration syntax

The Geronimo Way

- Atomic unit of service is a GBean
- GBeans are grouped into modules (aka configurations)
- Geronimo kernel knows how to configure and manage GBeans
- Core service knows how to save/load configurations

Kernel & GBeans

- Configurations are started
- GBeans are wired to each other



More Kernel

- GBeans can be exposed via JMX (GBean Name -> ObjectName)
- GBeans include basic metadata (properties, constructors, etc.)
- GBeans have dependencies that the Kernel manages
- Configuration managed via Inversion of Control

IoC Configuration

- No lookups; GBean A is given references it requires to GBeans B,C,...

```
<gbean name="GerConfigStore"
       class="o.a.g.LocalConfigStore">
    <attribute name="root">
      config-store
    </attribute>
    <reference name="ServerInfo">
      <name>GerServerInfo</name>
    </reference>
</gbean>
```

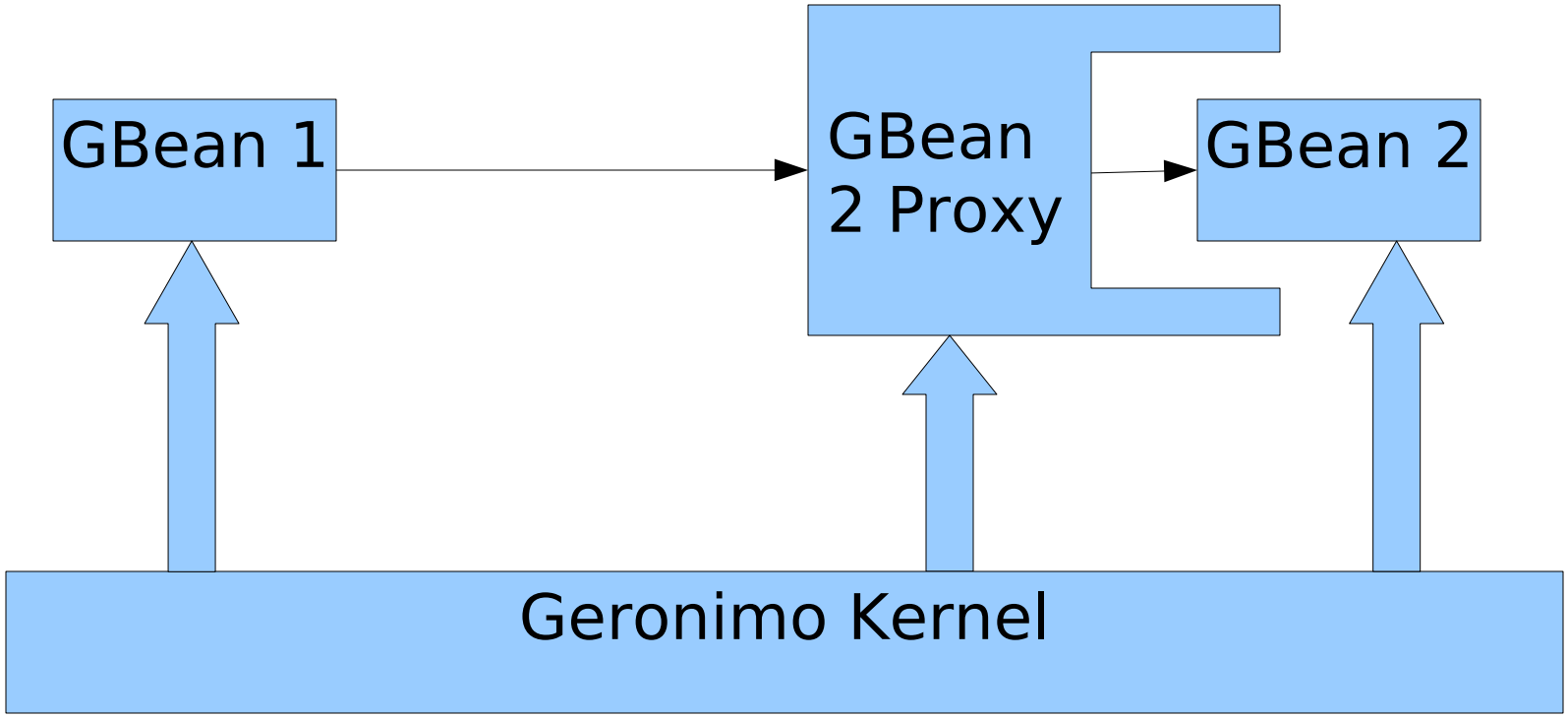
IoC In Action

- Configured references and attributes are converted to constructor arguments or property setter calls

```
attribute name="root"
reference name="ServerInfo"
    - becomes -
new LocalConfigStore(root, ServerInfo)
    - or -
localConfigStore.setRoot(...)
localConfigStore.setServerInfo(...)
```

Kernel in Control

- Kernel actually inserts proxies between one GBean and another



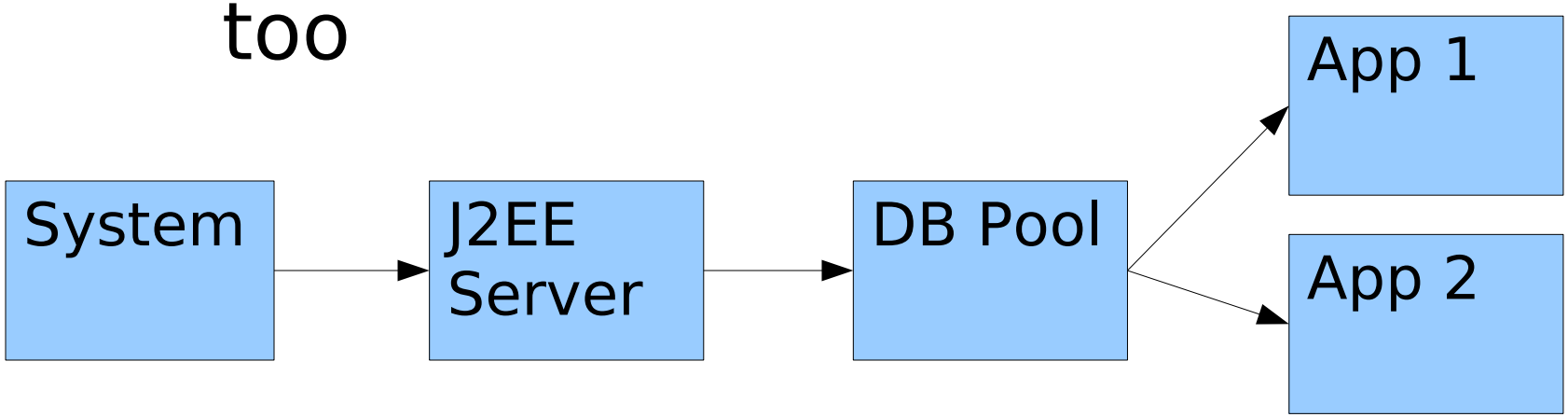
Sample Configuration

(XML pseudocode)

```
<configuration name="BasicServer">
  <GBean - ServerInfo
  <GBean - URLFactory
  <GBean - ConfigStore
  <GBean - ConfigManager
  <GBean - PersistentConfigList
  <GBean - Repository
  <GBean - Logging
  <GBean - RMI Registry
  <GBean - JNDI Naming Server
</configuration>
```

Configuration Tree

- Each configuration has a parent
- ClassLoaders follow configuration hierarchy
- Applications are configurations too



Container in Practice

- So what can you build on this?
  - A J2EE Server
  - Any other complex, modular, server product
  - Available components include what you'd expect given the J2EE focus: mail, web, database, logging, transactions, security, ...
  - Easy to build your own GBeans
  - JMX manageable "for free"

Geronimo & Spring

- The one container I didn't mention before...
- Geronimo has an alternate Spring-based GBean configuration system
- There's also app-level integration in progress

Baltimore
Linux Users Group

Background: J2EE vs. LAMP

CHARIOT SOLUTIONS

LAMP

- Static content via Apache
- Dynamic content via PHP, Perl, or Python Apache modules
  - Scripting runtime generally managed by Apache modules
  - Varying compatibility with Apache 2 threading modules
- Typically MySQL (or PostgreSQL, etc.) database

J2EE

- Static content via Apache or integrated web server
- Dynamic content runs in J2EE "application server" product with own VM environment
  - Apache module knows how to call over to app server for dynamic content
- Virtually any database product

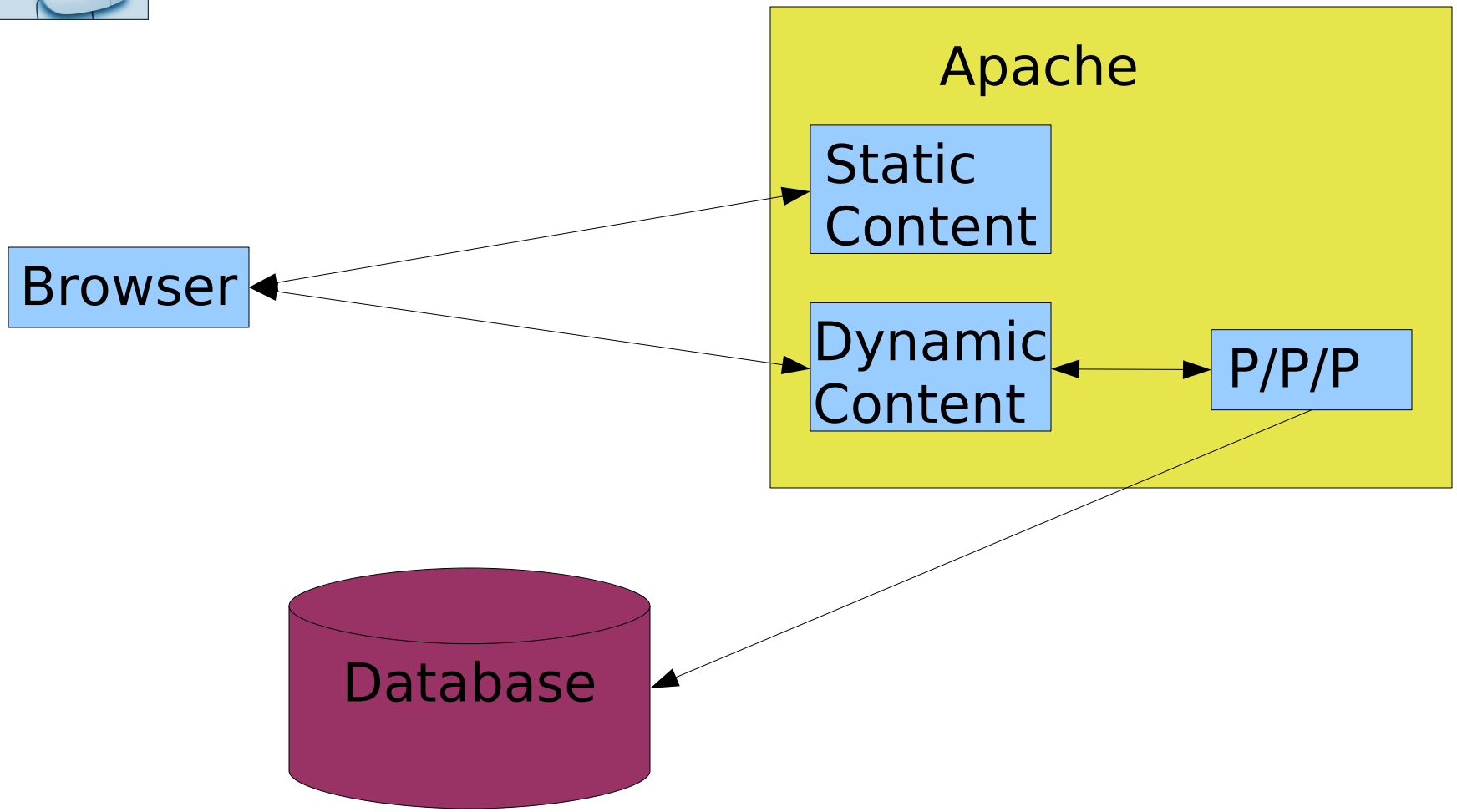
Java/JSP vs P/P/P

- All Java code must be compiled before executing (though JSPs can be compiled on the fly)
  - Faster to execute pre-compiled code
- J2EE applications must be "packaged" to a greater degree
- Can be quick & dirty either way, but JSP less mature for that

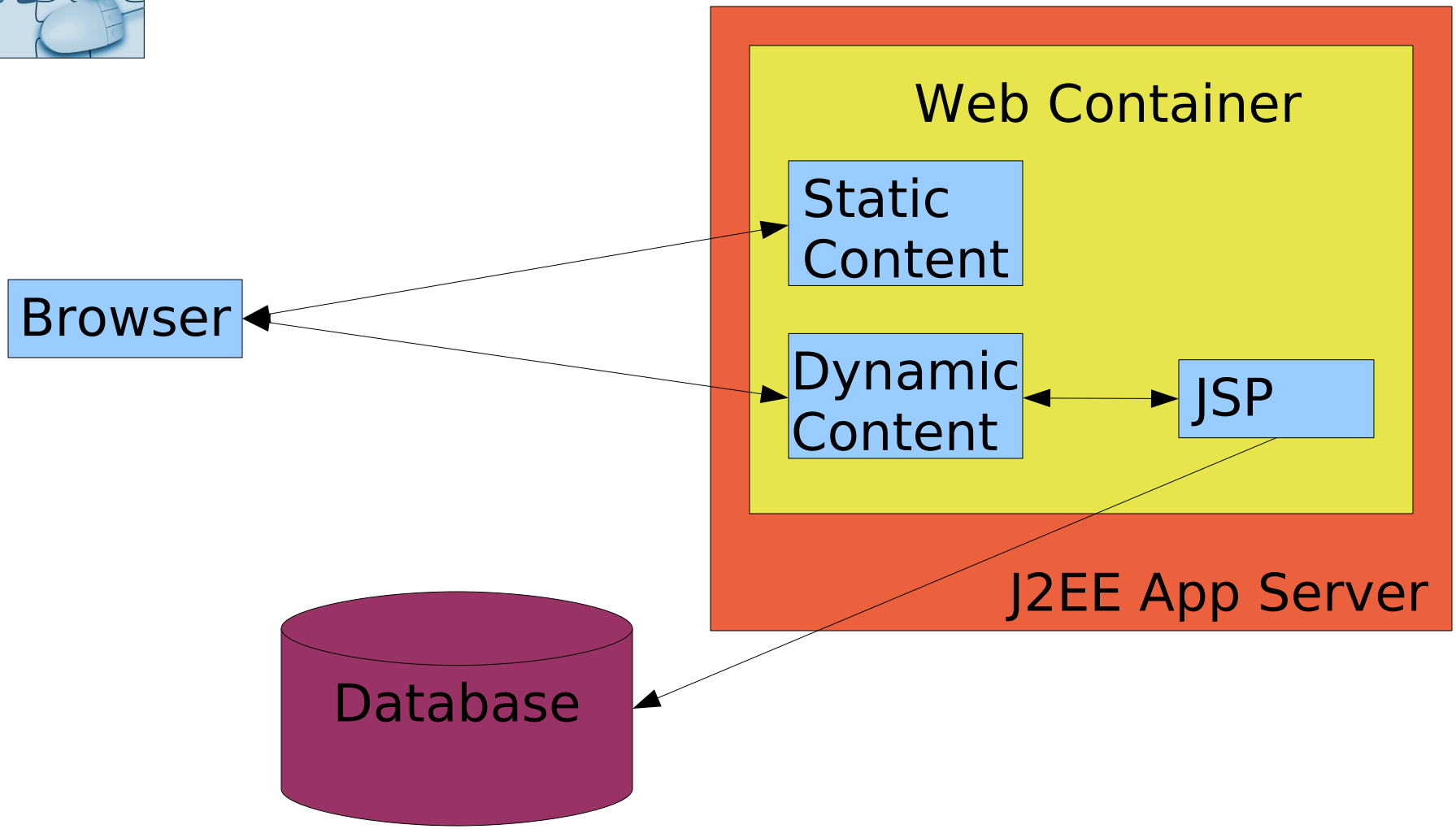
J2EE App Servers

- Include more "enterprise" features
  - Formal "business logic" layer
  - Database pooling
  - Several security schemes (not row-level)
  - Transactions and XA/2 phase commit
  - Asynchronous messaging
  - Integrated CORBA & Web Services
  - Object lifecycle, pooling, caching

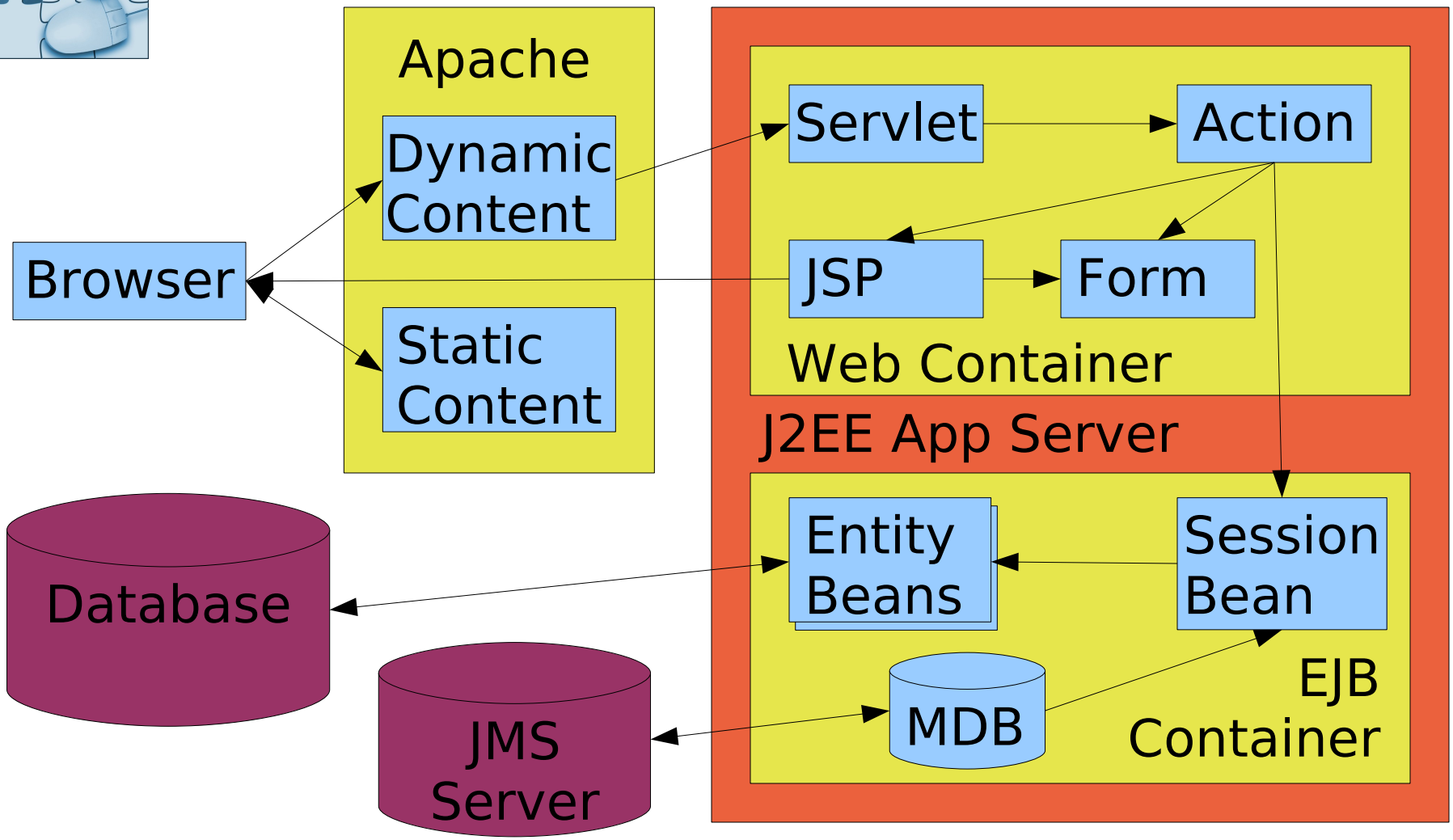
LAMP


Apache
Static Content
Browser
Dynamic Content
P/P/P
Database

J2EE: Quick & Dirty



J2EE: Enterprise App



J2EE Architectures

- Thriving design pattern & best practices community
- Focus on separation of concerns: 3-tier/MVC architectures, etc.
- Generally more code / harder to write but easier to maintain
- Can write ugly code either way!

Baltimore
Linux Users Group

Geronimo: The J2EE Application Server

CHARIOT SOLUTIONS

J2EE Server Agenda

- Current Status
- Install/Start/Stop routine
- Overview of J2EE modules & features

J2EE Server Status

- Stable kernel
- 20+ committers
- J2EE features complete
- Certification testing in progress (not allowed to say much more)
- Room for improvement in ease of use

Availability

- Dated Milestone 3 release
- Recent unstable releases
  - Actually more "stable" than M3
- Download from web site
- unzip and go
- Must use executable JAR installer to change ports, etc.

Starting Geronimo

- From the install directory, run

```
java -jar bin/server.jar
```

  - **`server.jar`** can actually be double-clicked or run from anywhere

- If it doesn't start up, check for exceptions
  - Usually a BindException because something else is running on a port it uses (8080, 1099, etc.)

Stopping Geronimo

- What can I say?
  - Ctrl-C
  - kill
  - kill -9 :)
- Shutdown command is overdue *(see "ease of use")*

Deploy Tool

- Used to deploy applications as well as GBean configurations
- Best when the server is running
- Typical syntax:

```
java -jar bin/deployer.jar help [command]
java -jar bin/deployer.jar deploy [archive] [plan]
java -jar bin/deployer.jar start [module name]
java -jar bin/deployer.jar stop [module name]
java -jar bin/deployer.jar undeploy [module name]
java -jar bin/deployer.jar list-modules
```

Deployment Plans

- All there is for service modules (GBean configurations)
- Server-specific deployment information for applications (similar to J2EE deployment descriptors)
- XML documents based on Geronimo XML schemas (with corresponding namespaces)

Sample Plan

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://geronimo..."
         configId="MyWebApp"
         parentId="J2EEServer">
    <context-root>
        /MyWebApp
    </context-root>
    <context-priority-classloader>
        false
    </context-priority-classloader>
</web-app>
```

J2EE Services

- Geronimo includes
  - Web container
  - EJB container
  - J2EE Connector container
    - JDBC Pools & JMS Resources
  - Application Client container
  - JMS server
  - Security & Transactions
  - Management & Deployment APIs
  - CORBA & Web Services
  - and much more...

Thanks to...

- MX4J
- OpenEJB
- Jetty
- Tomcat
- ActiveMQ
- TranQL
- JOTM
- CGlib
- Axis
- Scout
- XMLBeans
- Commons

Web Applications

- via Jetty or Tomcat
- Security, classpath integration
  - Including HTTPS, certificates, etc.
- Config includes ports, logs, ...
- All J2EE 1.4 features supported
- Very stable

EJBs

- via OpenEJB
- EJB 2.1 support now complete
- Full CMP support
  - Still missing some optimizations, such as CMP/CMR load groups
  - Automatic PK generation in flux
  - Needs more extensive database testing
- Instant stubs via CGlib

J2EE Connectors

- via a custom Geronimo module
- J2CA 1.5 fully supported
  - Inbound and outbound connectors supported
  - Work manager thread pools supported
- JDBC pools, JMS connection factories, and JMS destinations implemented using connectors

Application EARs

- Deployment support for EARs
- May include common libraries or reference them within the server JAR repository
- May include resources to deploy when the app is deployed
- Web App ClassLoader under EJB/Connector ClassLoader

Application Clients

- Full application client container, including JAAS Login, EJB References, Resource References, etc.
- "Plain J2SE" clients supported as well, with JAAS authentication and JNDI resource lookup (though not all resources work remotely)

DB Connection Pools

- Configured using a J2EE Connector (from TranQL)
- Can be deployed at the server level (for use by multiple apps), or with an application (started/stopped with that application)
- Standard pooling features

JMS

- via ActiveMQ
- Topics, Queues, Connection Factories, etc. (all configured using a J2EE Connector)
- May be server-level or app-level
- Full MDB support, including message selectors, threading, various JMS options

JavaMail

- via custom Geronimo module
- Includes Java Activation Framework
- Includes configuration support for POP, IMAP, SMTP, but AFAIK no actual transport providers
- Can be used via application resource reference

Transactions

- via custom Geronimo module and HOWL
- Full JTA support, including crash recovery
- HOWL used for transaction log
- XA/2pc support (but could use broader testing)

Security

- Can configure realms with authentication, auditing, etc.
- Includes properties file, DB, Kerberos, client certificate support now (need LDAP!)
- EAR or J2EE module can map users/groups to J2EE roles, select a default identity, etc.

CORBA

- via ORB in JDK with other tools
- EJBs can be exposed or called via CORBA, including security support
- Can be used to interoperate with other app servers, as well as with non-Java code

Web Services

- via Axis, Scout, etc.
- Support for JAXR, JAX-RPC, etc.
- Session beans and servlets exposed as web services
- Any component can reference and call a remote web service
- Still a little bleeding-edge

J2EE Deployment

- Operational aspects work well with deploy tool (deploy, undeploy, etc.)
- Currently only supports tool running from same machine as server
- Configuration beans are incomplete

J2EE Management

- Includes full JMX management support
  - MEJB is provided
- All GBeans are manageable via this same interface (with no extra code needed)
- Already supported by tools like MC4J

Documentation

- Wiki has both the most and the least up to date information :)
- Numerous articles available
- Several books in progress, including partial/full online text
- Unfortunately, some file formats & features are still in flux

Baltimore
Linux Users Group

Geronimo: Status, TODOs, & Contributing

CHARIOT SOLUTIONS

J2EE Server Status

- Core features stabilizing, particularly via J2EE certification testing
  - Certification only covers application runtime features
- Needs wider testing
- Needs more developer-friendly and admin-friendly features

Testing

- Test compatibility with various databases (direct SQL and CMP)
- Test interoperability with various app servers
- Test performance
- Test OS & JVM platforms
- Test real-world applications

Ease of Use

- Hot deploy directory
- Management console
- Easier configuration editing (changing ports, etc.)
- Portal & Spring integration
- IDE integration

Getting Involved

- http://geronimo.apache.org/
- Try an unstable build
- Join the mailing list
- View/update the Wiki
- Review the road map / TODOs
- Submit bugs or patches to JIRA

Baltimore
Linux Users Group

Q&A

http://chariotsolutions.com/geronimo/

CHARIOT SOLUTIONS