Apache Geronimo
JavaOne 2006

Aaron Mulder
Jeff Genender
The Apache Geronimo Community
Agenda

- Geronimo Overview & History
- Geronimo 1.0 (Jan 2006)
- Geronimo 1.1 (June 2006)
  - Geronimo Plugins
- Geronimo 1.2 (3Q/4Q 2006)
  - Java EE 5 features, Community Requests
- 2.0 and Beyond
- Discussion
Geronimo Overview & History
What is Apache Geronimo?

- J2EE certified application server
- Completely open source, under the Apache License v2
- Built from best-of-breed open source components (Tomcat, Jetty, ActiveMQ, OpenEJB, Howl, etc., etc., etc.)
- Developed by an open community at Apache
- Production distributions and support available, both large and small
Why Geronimo?

• JBoss and JOnAS encumbered by
  – *GPL licensing
  – Not entirely open community
• Tomcat has only web features
  – Even if you don't buy into EJB, the world needs more (messaging, transactions, etc.)
• The app server market is losing innovation (vs. e.g. Ruby on Rails)
History

• Started quite a while ago
  – Initial J2EE certification took an altogether unreasonable amount of time
• First passed the TCK at JavaOne 2005
• 1.0 release in January
• 1.1 in feature freeze now
• 1.2 up next
Community

- 28+ committers from 8+ companies
- Extremely active mailing lists (user/dev)
- Many companies providing stacks including Geronimo and/or 24/7 support
- Multiple companies building applications on Geronimo (did you visit the pavilion?)
- Sites as big as Ebay deployed on Geronimo
Geronimo 1.0
Geronimo 1.0

- J2EE feature release
  - Web, EJB, JMS, J2CA, JTA, ...
- J2EE certified bundles with either Tomcat or Jetty as the web container
- Initial web-based administration console
- Initial tool support (XDoclet, Eclipse, Maven, debuggers, etc.)
- Initial performance profiling
And the Benefits

- Server runs, redeployes, and serves apps without bumping up JVM memory args
- Administration console for configuration
  - web ports, database pools, security realms...
- Remote management and deployment
- Clean startup with useful progress and port/URL information
- Free documentation, published schemas
Under the Covers

• Modular architecture
  – Easy to create custom distributions with custom feature sets

• Easy to develop Geronimo services
  – GBeans, inspired by Spring, have few requirements (just some metadata)
Limitations of 1.0

• Dependency tracking too restrictive
• Limited clustering support
• J2EE distributions + samples are quite heavyweight for many purposes
• Only J2EE certified on Java 1.4
  – Plus misleading startup stack trace on Java 5
• May be easy to build add-ons, but not as convenient to distribute them
Geronimo 1.1
Geronimo 1.1 Status

- In feature freeze now
- Expecting release in next few weeks
- Big step forward compared to 1.0
- Unfortunately, including some changes affecting compatibility
  - Deployment plan format has changes to the header block
  - We include a tool to convert from 1.0 to 1.1
New in 1.1

• Plugin architecture supporting click-through installation of new features and application or server upgrades

• New features in admin console including:
  – live memory graph, JMS resource & keystore configuration, Apache HTTP configuration generator

• Little G – a lightweight distribution option

• Java 5 support (except CORBA)
Geronimo Plugins

• A plugin is a packaged Geronimo module, either an application or a service module

• Can be installed by pointing the console at a plugin repository and selecting from a list, or downloading the plugin and installing from the command line

• No configuration or XML required – plugins “just work”
Plugin Repository

- Plugins live in a Maven 2 repository
- Dependencies (other plugins or JARs) are downloaded from the same or other Maven 2 repositories
- Default plugin hosting site supports plugins with any license (OSS/proprietary)
- ibiblio.org is the default site hosting dependency JARs
Installing Plugins

- Geronimo checks prerequisites to ensure that the plugin can be installed
- Any obsolete modules are stopped
- Each dependency is downloaded if it's not already available to the server
  - And their dependencies, and so on...
- The plugin is installed and started, and is immediately available
Creating Plugins

• Any application or module running in Geronimo can be exported as a plugin
• Console prompts for the necessary metadata (friendly name, license, etc.), then gives you a “save as” dialog
• Just a few things to keep in mind:
  – Declare dependencies explicitly, and don't pack them into the application if you want to share them with other apps/plugins/etc.
Plugin Inspirations

• Geronimo uses plugin infrastructure to clone features from server to server
  – First developer gets everything set up, next developer just pulls it all down
  – Can migrate apps from dev to test, etc.

• Plugins help integrate other products
  – LDAP server, scheduler, portal, etc.

• Plugins can be used to simplify application installation
Limitations of 1.1

- Still don't have extensive clustering features
- Still not certified on Java 5
- No EE 5 features
- Plugins cannot install features into the admin console
- Limited monitoring/statistics support
Geronimo 1.2
Geronimo 1.2 Status

• Currently in the planning stages
• Everything we say here is preliminary
• We'd like your input and feedback!

• all that said...
High-Level Features

- Class loading improvements
- Global JNDI space
- Plugins can include console screens
- Better Spring integration
- Full Java 5 support
- JPA, web services, and web app features from EE 5
- Clustering
More 1.2 Thoughts

• Targeting a 3 month release cycle
• Will be releasing some updates incrementally via plugins
• Will work toward increased integration with other products via plugins outside the 1.2 release cycle
2.0 and Beyond

• Planning on full Java EE 5 support for Geronimo 2.0
  – Everyone heckle BEA until the OpenJPA code actually shows up!
• Expect to see increasing use of annotations
• Tighter Spring integration
• Will continue to improve the Geronimo ecosystem via plugins