

# Geronimo Performance Baseline

Performance measurements using *Geronimo* at M5

Matt Hogstrom

[matt@hogstrom.org](mailto:matt@hogstrom.org)

10/8/2005

# Acknowledgements

- Intel Corporation supplied the servers for the Application Server and Driver
- Intel has also provided their VTune performance tooling to analyze and improve Geronimo. (Very cool stuff)
- IBM for donating their Trade 6 workload (now named DayTrader)

# Overview

- Performance target is an aggregation of Commercial and Open Source Application Server results using this benchmark
- Goal is to communicate performance as of M5 for planning purposes for 1.0 targets
- Driver system used an IBM load generator
  - Working to move to JMeter or another OpenSource load generator for broader use and repeatability

# Overview

- Planning on to include Oracle as an additional commercial database
- The workload and related information can be found by downloading the application (using Subversion) by:

```
svn checkout http://svn.apache.org/repos/asf/geronimo/trunk/sandbox/daytrader daytrader
```

Note: as of 10/08/2005 there are many references to WebSphere. They are being updated to reflect the intent of the Geronimo subproject at Apache.

# The Web Tests

## Web Container ping suite

- **PingServlet**  
Tests fundamental dynamic HTML creation through server side servlet processing.
- **PingJSP**  
Tests a direct call to JavaServer Page providing server-side dynamic HTML through JSP scripting.
- **PingHTTPSession1**  
SessionID tests fundamental HTTP session function by creating a unique session ID for each individual user. The ID is stored in the users session and is accessed and displayed on each user request.
- **PingJDBCRead**  
Tests fundamental servlet to JDBC access to a database performing a single-row read using a prepared SQL statment.
- **PingJDBCWrite**  
Tests fundamental servlet to JDBC access to a database performing a single-row write using a prepared SQL statment.
- **PingServlet2JNDI**  
Tests the fundamental J2EE operation of a servlet allocating a JNDI context and performing a JNDI lookup of a JDBC DataSource.

# The EJB Tests

- **PingServlet2SessionEJB\***  
Tests key function of a servlet call to a stateless SessionEJB. The SessionEJB performs a simple calculation and returns the result
- **PingServlet2EntityEJBLocal\* and PingServlet2EntityEJBRemote\***  
Tests key function of a servlet call to an EJB 2.0 Container Managed Entity. In this test the EJB entity represents a single row in the database table. The Local version uses the EJB Local interface while the Remote version uses the Remote EJB interface. (Note: PingServlet2EntityEJBLocal will fail in a multi-tier setup where the Trade Web and EJB apps are separated.)
- **PingServlet2Session2Entity**  
This tests the full servlet to Session EJB to Entity EJB path to retrieve a single row from the database.
- **PingServlet2Session2EntityCollection**  
This test extends the previous EJB Entity test by calling a Session EJB which uses a finder method on the Entity that returns a collection of Entity objects. Each object is displayed by the servlet
- **PingServlet2Session2CMROne2One**  
This test drives an Entity EJB to get another Entity EJB's data through an EJB 2.0 CMR One to One relationship
- **PingServlet2Session2CMROne2Many**  
This test drives an Entity EJB to get another Entity EJB's data through an EJB 2.0 CMR One to Many relationship

# The EJB Tests

- **PingServlet2MDBQueue**

Drives messages to a Queue based Message Driven EJB (MDB). Each request to the servlet posts a message to the Queue. The MDB receives the message asynchronously and prints message delivery statistics on each 100th message.

- **PingServlet2MDBTopic**

Drives messages to a Topic based Publish/Subscribe Message Driven EJB (MDB). Each request to the servlet posts a message to the Topic. The TradeStreamMDB receives the message asynchronously and prints message delivery statistics on each 100th message. Other subscribers to the Topic will also receive the messages.

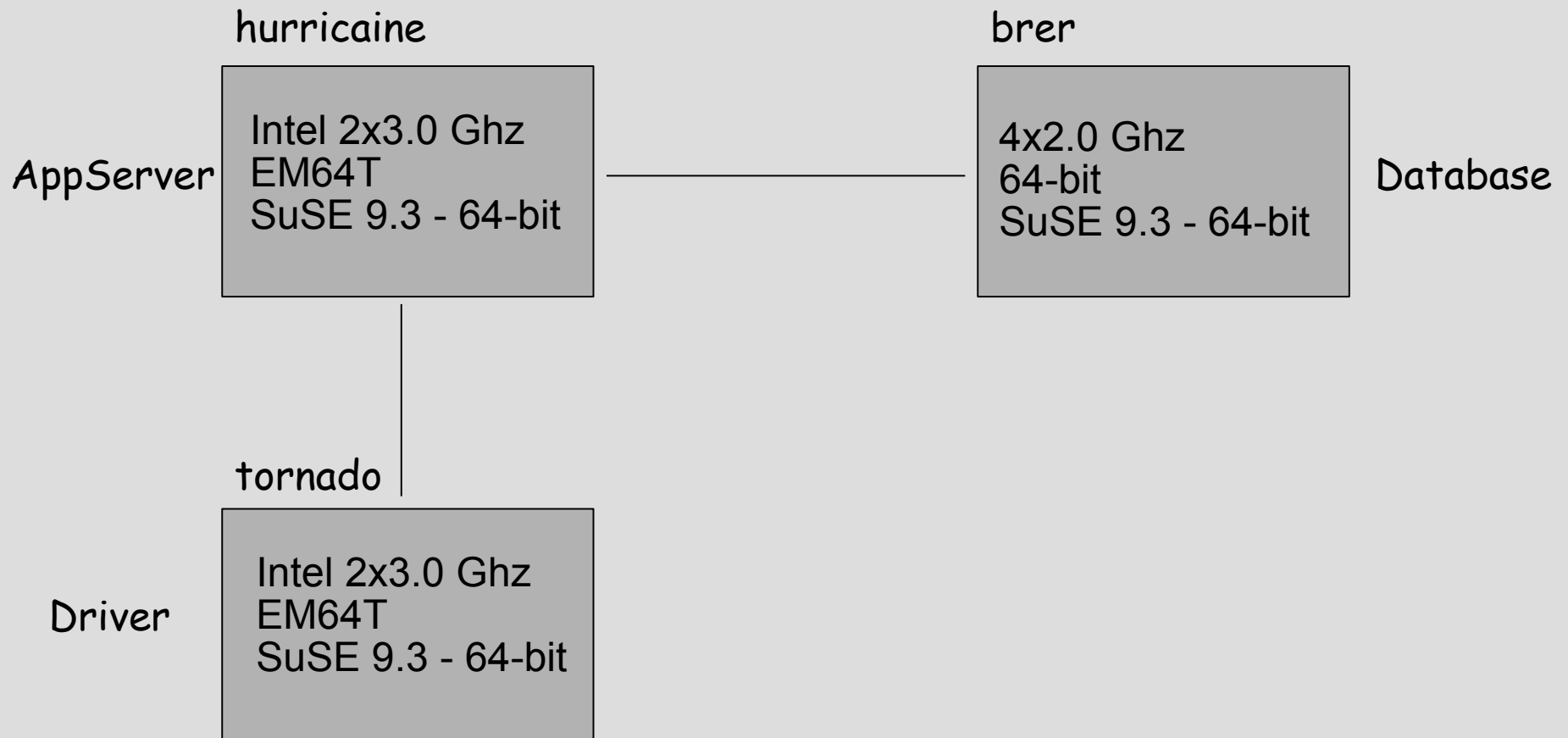
- **PingServlet2TwoPhase**

Drives a Session EJB which invokes an Entity EJB with `findByPrimaryKey` (DB Access) followed by posting a message to an MDB through a JMS Queue (Message access). These operations are wrapped in a global 2-phase transaction and commit.

# Testing the Full Workload

- Trade Direct
  - Uses servlets and JSPs to run the DayTrader Workload
- Trade EJB
  - Uses servlets, JSPs and CMP Entity beans to drive the workload

# Hardware / OS Configuration



WebSphere Studio Simulator (JIBE)

# Application Server Software

AppServer

Geronimo

M5 Test Build  
JDK Sun 1.4.2\_09  
DB2 JCC Driver

Database

DB2 Version 8.2

# Test Application

- DayTrader
  - Open Sourced Version of Trade 6
  - Modified to remove WebSphere specific features
    - Dynacache
    - Command Bean Pattern
    - Distributed Map
  - Slight restructuring to remove circular dependencies
  - Available at Geronimo SVN in sandbox

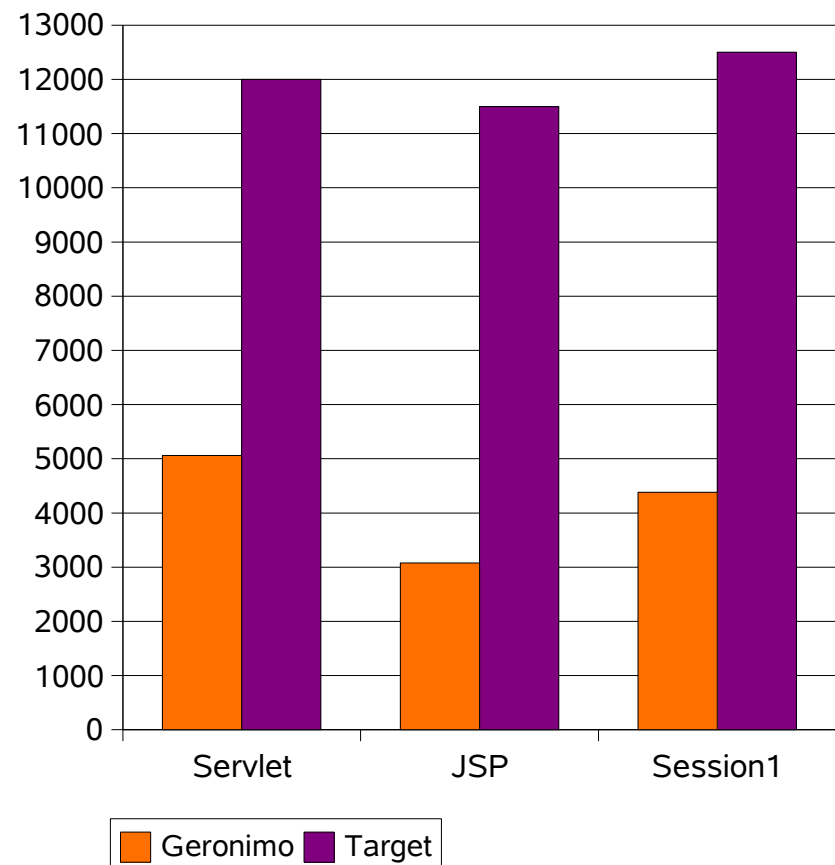
# Testing Methodology

- Goal
  - Using mostly the same basic parameters and settings compare the performance of the 3 Application Servers.
- Tuning was the same for each AppServer
  - Logging and other unnecessary function was disabled
  - Connection Pool of 60 database connections
  - Non Geronimo AppServers had a statement cache of 30 statements (no stmt caching for Geronimo yet 😊 )
  - WebContainer threads were set to 50
  - JVM heap set to 1024KB with -server

# Web Primitive Comparison

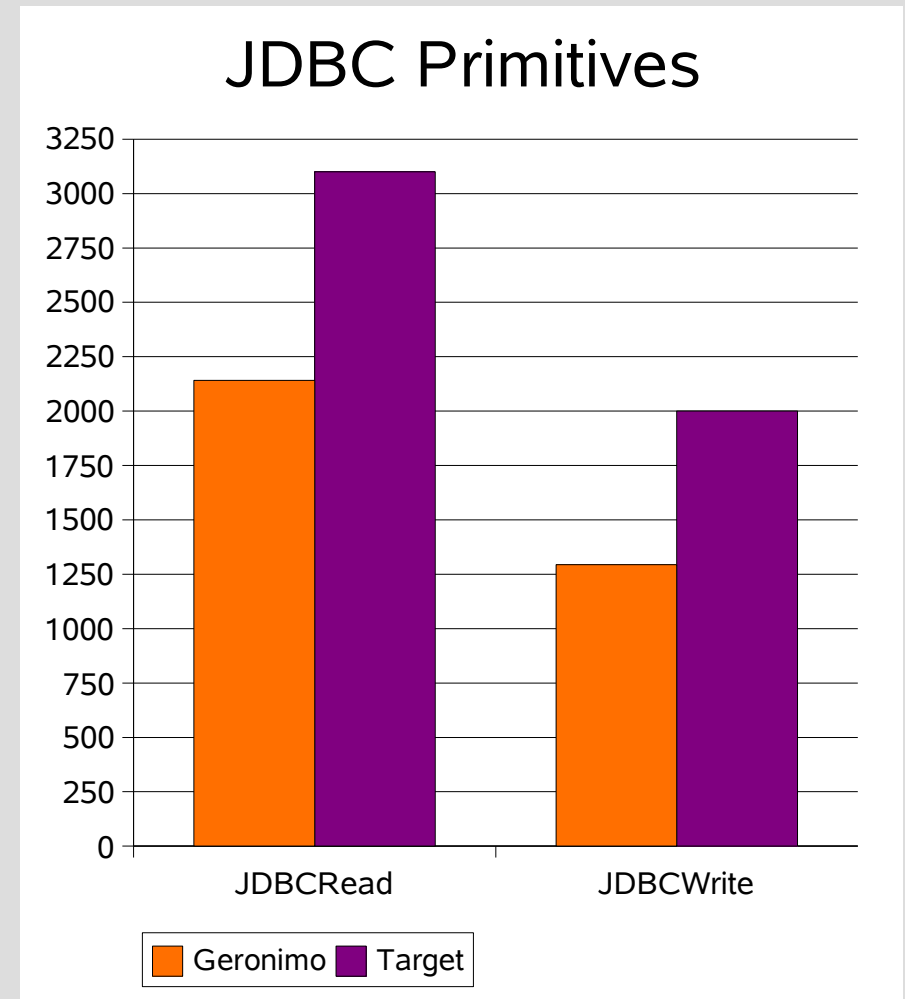
- Currently the Web primitives we're behind our target
- Jetty is currently logging every request. The target appservers had logging disabled
- Need to fix Geronimo to disable HTTP logging

## Web Primitives



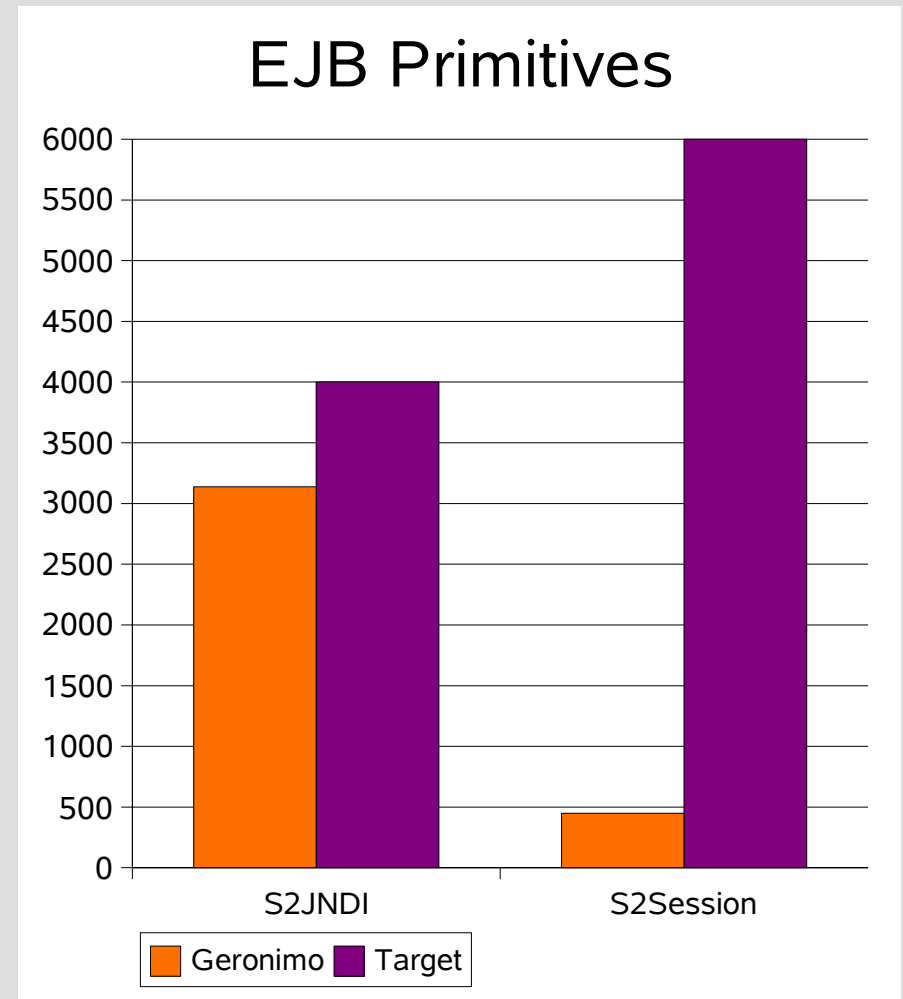
# Web Primitives (continued)

- Once additional pathlength is added the advantage in the web primitives
- Performance in this area can be improved by adding a statement cache to TranQL



# EJB Session Primitives

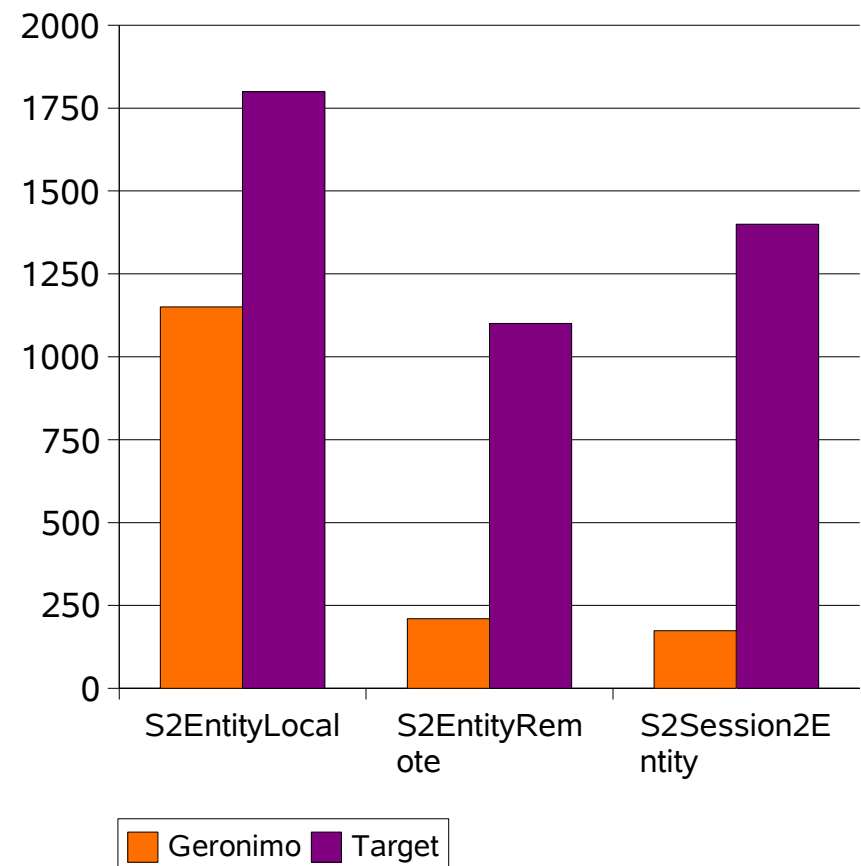
- The drop in performance of S2Session is being researched



# EJB Entity Bean Primitives

- Remote EJB calls is seriously degrading performance
- Focus on Parameter Copying is a current focus

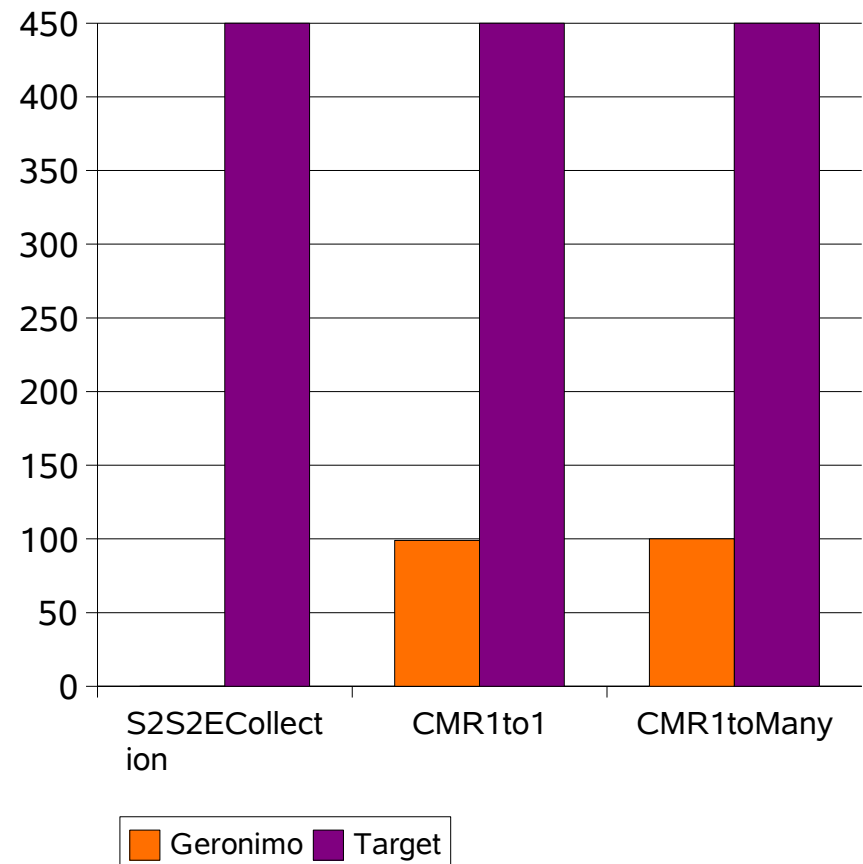
## EJB Entity Bean Primitives



# EJB 2.1 Entity Primitives

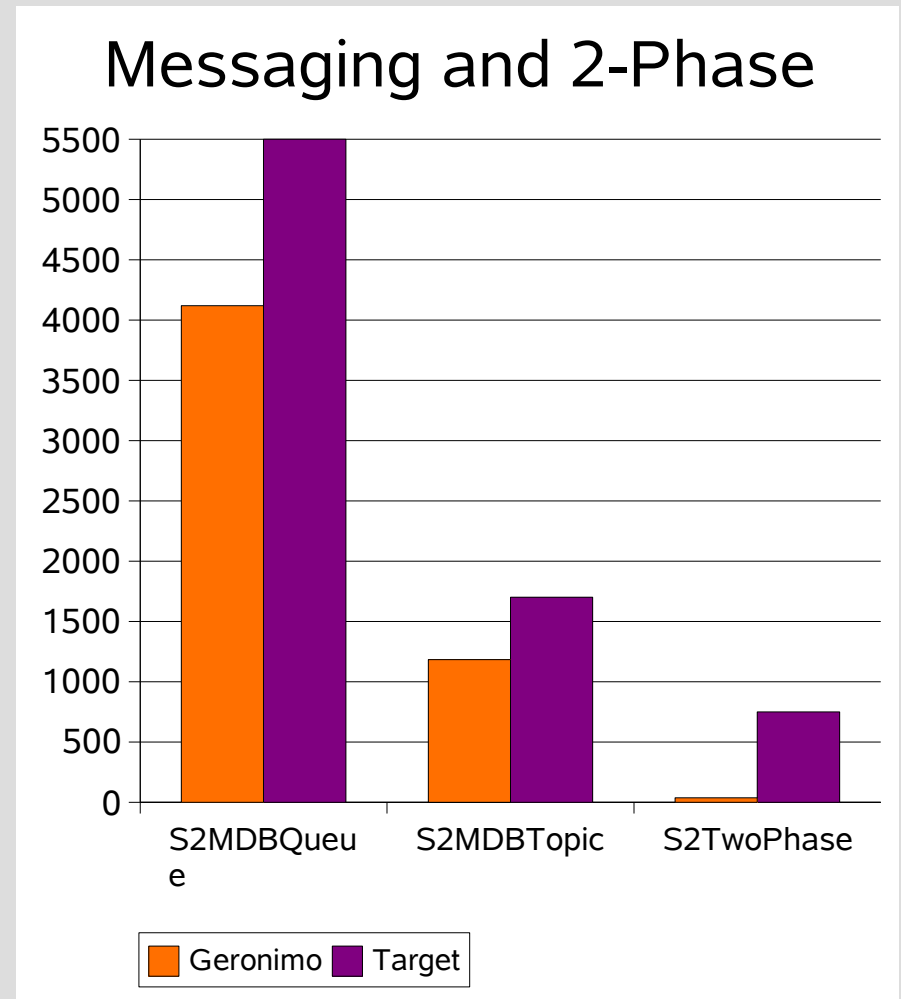
- Servlet2Session2EntityCollection was a problem for Geronimo due to improper SQL generation (fixed in TranQL 1.2)

## EJB 2.1 Entity Primitives



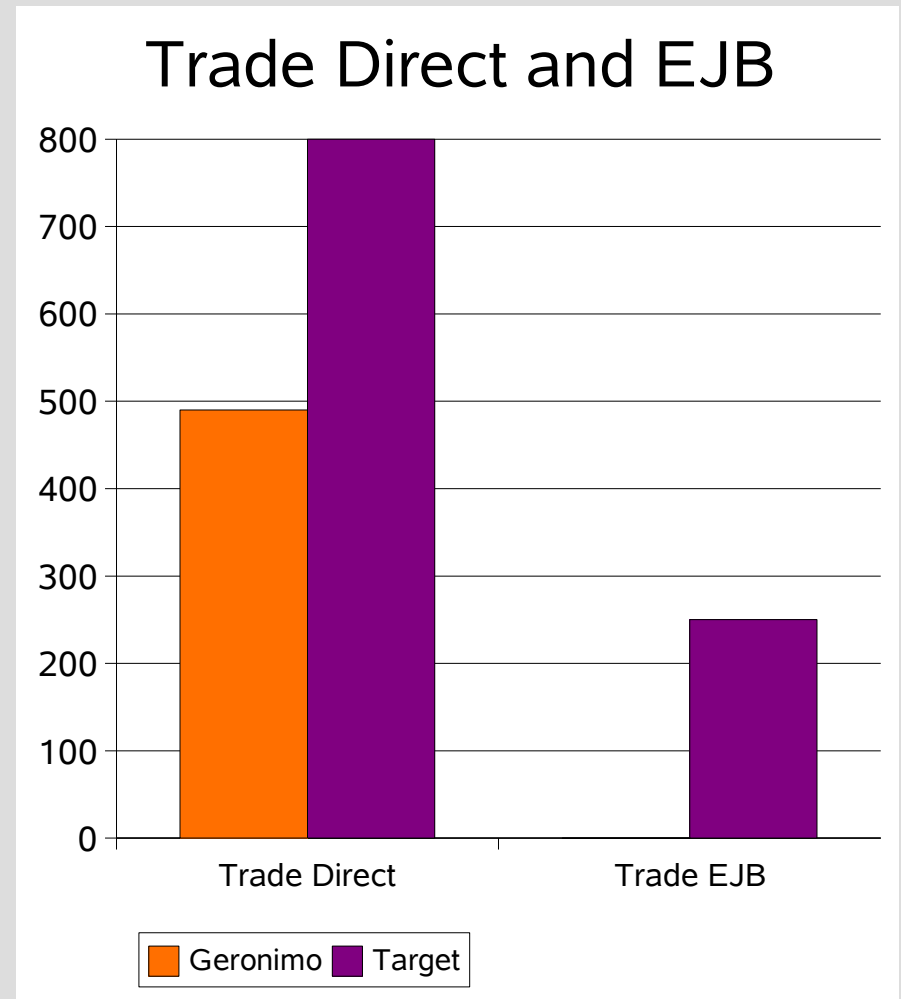
# Messaging and 2-Phase

- Geronimo ran at about 86% CPU Utilization
  - Appears to be related to Tx Logging
  - Possibly due to Derby as the Database
  - MDBQueue could improve dramatically when these issues are addressed



# Trade Direct and EJB

- SQL Issue has been corrected for Geronimo and has been committed to TranQL 1.2



# Conclusion

- M5 is the first certified release
- Performance for general workloads (ala Daytrader) is at a good point for the beginning of performance work
- Geronimo is well positioned to make its competitive targets in the near term

# Top 5 Performance Issues

- Parameter copying in the ORB
- Statement Caching (missing feature in TranQL)
- HTTP Access logging in Jetty
- Sub-optimal SQL generation in TranQL for DB2  
(need to improve for all commercial Dbs)
- Examine performance of transaction logging

# Interested in Participating?

- Contact the development team at [dev@geronimo.apache.org](mailto:dev@geronimo.apache.org)

# Geronimo Performance Baseline

Performance measurements using *Geronimo* at M5

Click to add text

Matt Hogstrom  
[matt@hogstrom.org](mailto:matt@hogstrom.org)  
10/8/2005

## Acknowledgements

- Intel Corporation supplied the servers for the Application Server and Driver
- Intel has also provided their VTune performance tooling to analyze and improve *Geronimo*. (Very cool stuff)
- IBM for donating their Trade 6 workload (now named DayTrader)

## Overview

- Performance target is an aggregation of Commercial and Open Source Application Server results using this benchmark
- Goal is to communicate performance as of M5 for planning purposes for 1.0 targets
- Driver system used an IBM load generator
  - Working to move to JMeter or another OpenSource load generator for broader use and repeatability

## Overview

- Planning on to include Oracle as an additional commercial database
- The workload and related information can be found by downloading the application (using Subversion) by:

```
svn checkout http://svn.apache.org/repos/asf/geronimo/trunk/sandbox/daytrader daytrader
```

Note: as of 10/08/2005 there are many references to WebSphere. They are being updated to reflect the intent of the Geronimo subproject at Apache.

# The Web Tests

## Web Container ping suite

- **PingServlet**  
Tests fundamental dynamic HTML creation through server side servlet processing.
- **PingJSP**  
Tests a direct call to JavaServer Page providing server-side dynamic HTML through JSP scripting.
- **PingHTTPSession1**  
SessionID tests fundamental HTTP session function by creating a unique session ID for each individual user. The ID is stored in the users session and is accessed and displayed on each user request.
- **PingJDBCRead**  
Tests fundamental servlet to JDBC access to a database performing a single-row read using a prepared SQL statement.
- **PingJDBCWrite**  
Tests fundamental servlet to JDBC access to a database performing a single-row write using a prepared SQL statement.
- **PingServlet2JNDI**  
Tests the fundamental J2EE operation of a servlet allocating a JNDI context and performing a JNDI lookup of a JDBC DataSource.

# The EJB Tests

- **PingServlet2SessionEJB\***  
Tests key function of a servlet call to a stateless SessionEJB. The SessionEJB performs a simple calculation and returns the result
- **PingServlet2EntityEJBLocal\* and PingServlet2EntityEJBRemote\***  
Tests key function of a servlet call to an EJB 2.0 Container Managed Entity. In this test the EJB entity represents a single row in the database table. The Local version uses the EJB Local interface while the Remote version uses the Remote EJB interface. (Note: PingServlet2EntityEJBLocal will fail in a multi-tier setup where the Trade Web and EJB apps are separated.)
- **PingServlet2Session2Entity**  
This tests the full servlet to Session EJB to Entity EJB path to retrieve a single row from the database.
- **PingServlet2Session2EntityCollection**  
This test extends the previous EJB Entity test by calling a Session EJB which uses a finder method on the Entity that returns a collection of Entity objects. Each object is displayed by the servlet
- **PingServlet2Session2CMROne2One**  
This test drives an Entity EJB to get another Entity EJB's data through an EJB 2.0 CMR One to One relationship
- **PingServlet2Session2CMROne2Many**  
This test drives an Entity EJB to get another Entity EJB's data through an EJB 2.0 CMR One to Many relationship

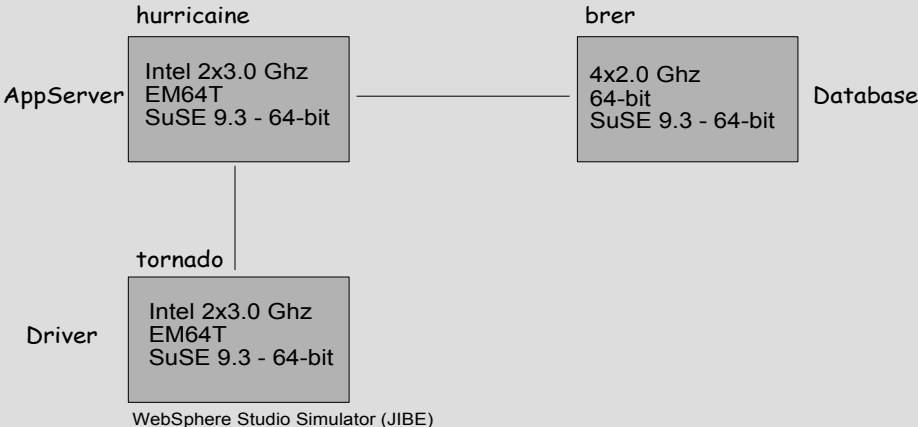
# The EJB Tests

- **PingServlet2MDBQueue**  
Drives messages to a Queue based Message Driven EJB (MDB). Each request to the servlet posts a message to the Queue. The MDB receives the message asynchronously and prints message delivery statistics on each 100th message.
- **PingServlet2MDBTopic**  
Drives messages to a Topic based Publish/Subscribe Message Driven EJB (MDB). Each request to the servlet posts a message to the Topic. The TradeStreamMDB receives the message asynchronously and prints message delivery statistics on each 100th message. Other subscribers to the Topic will also receive the messages.
- **PingServlet2TwoPhase**  
Drives a Session EJB which invokes an Entity EJB with `findByPrimaryKey` (DB Access) followed by posting a message to an MDB through a JMS Queue (Message access). These operations are wrapped in a global 2-phase transaction and commit.

## Testing the Full Workload

- Trade Direct
  - Uses servlets and JSPs to run the DayTrader Workload
- Trade EJB
  - Uses servlets, JSPs and CMP Entity beans to drive the workload

# Hardware / OS Configuration



# Application Server Software

AppServer

*Geronimo*

M5 Test Build  
JDK Sun 1.4.2\_09  
DB2 JCC Driver

Database

DB2 Version 8.2

## Test Application

- DayTrader
  - Open Sourced Version of Trade 6
  - Modified to remove WebSphere specific features
    - Dynacache
    - Command Bean Pattern
    - Distributed Map
  - Slight restructuring to remove circular dependencies
  - Available at *Geronimo SVN* in sandbox

When the jCache JSR becomes final the standard caching API will be re-introduced into DayTrader

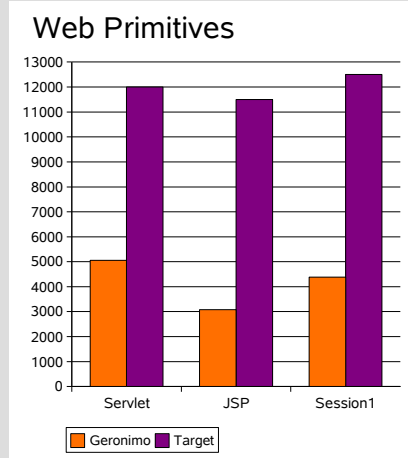
## Testing Methodology

- **Goal**
  - Using mostly the same basic parameters and settings compare the performance of the 3 Application Servers.
- **Tuning was the same for each AppServer**
  - Logging and other unnecessary function was disabled
  - Connection Pool of 60 database connections
  - Non Geronimo AppServers had a statement cache of 30 statements (no stmt caching for Geronimo yet ☺ )
  - WebContainer threads were set to 50
  - JVM heap set to 1024KB with -server

Some additional performance is more than likely possible with additional effort. The goal of this set of experiments was to determine a customer based experience out of the box from a competitive perspective of the Open Source projects against WebSphere.

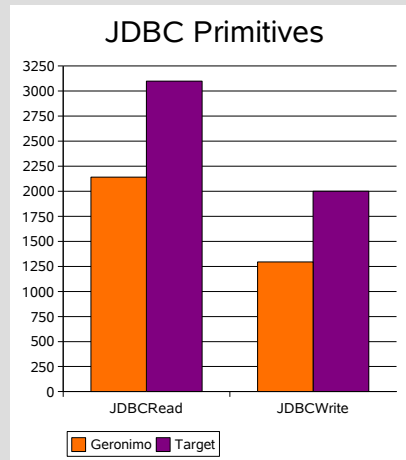
## Web Primitive Comparison

- Currently the Web primitives we're behind our target
- Jetty is currently logging every request. The target appservers had logging disabled
- Need to fix Geronimo to disable HTTP logging



## Web Primitives (continued)

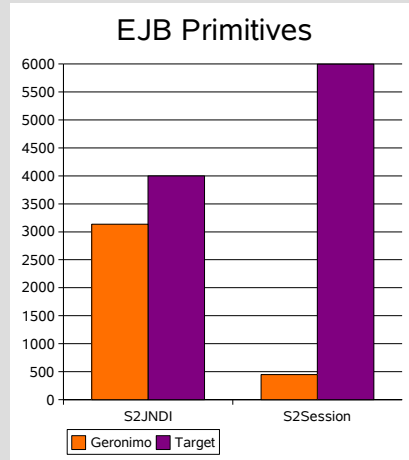
- Once additional pathlength is added the advantage in the web primitives
- Performance in this area can be improved by adding a statement cache to TranQL



Database CPU was less than 10% During this test.

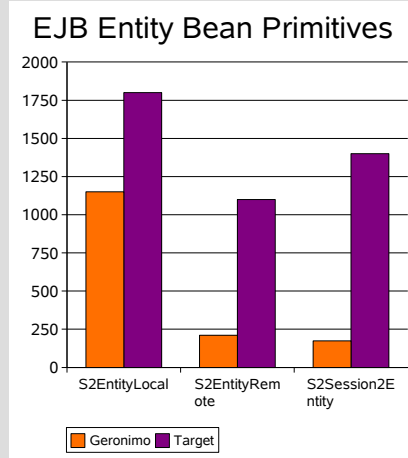
# EJB Session Primitives

- The drop in performance of S2Session is being researched



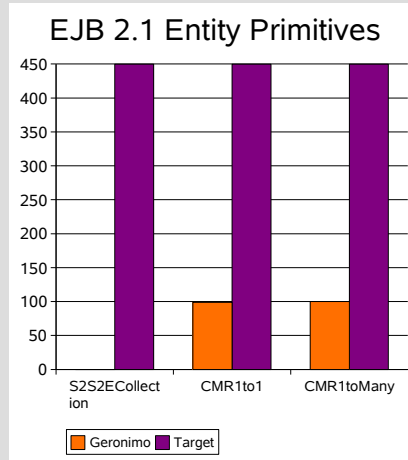
# EJB Entity Bean Primitives

- Remote EJB calls is seriously degrading performance
- Focus on Parameter Copying is a current focus



## EJB 2.1 Entity Primitives

- Servlet2Session2EntityCollection was a problem for *Geronimo* due to improper SQL generation (fixed in TranQL 1.2)

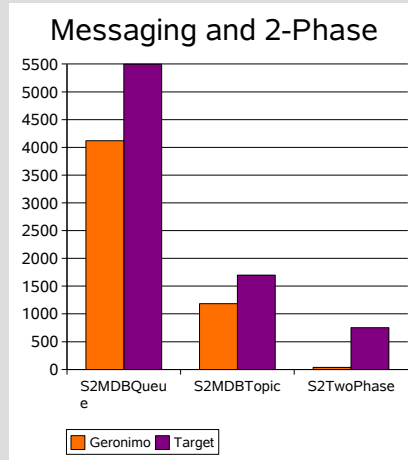


# Messaging and 2-Phase

- *Geronimo* ran at about 86% CPU

## Utilization

- Appears to be related to Tx Logging
- Possibly due to Derby as the Database
- MDBQueue could improve dramatically when these issues are addressed



## Trade Direct and EJB

- SQL Issue has been corrected for *Geronimo* and has been committed to TranQL 1.2



## Conclusion

- M5 is the first certified release
- Performance for general workloads (ala Daytrader) is at a good point for the beginning of performance work
- Geronimo is well positioned to make its competitive targets in the near term

## Top 5 Performance Issues

- Parameter copying in the ORB
- Statement Caching (missing feature in TranQL)
- HTTP Access logging in Jetty
- Sub-optimal SQL generation in TranQL for DB2 (need to improve for all commercial Dbs)
- Examine performance of transaction logging

## Interested in Participating?

- Contact the development team at [dev@geronimo.apache.org](mailto:dev@geronimo.apache.org)