JBoss Middleware for Spacecraft Trajectory Operations

Kjell Stensrud  Ravi Srinivasan
United Space Alliance  Hewlett Packard
Software Engineer  Senior Consultant

Dustin Hamm
NASA
Aerospace/Software Engineer

February 13, 2008
Abstract

• Project Background
  - What is the environment where we are considering Open Source Middleware?

• System Architecture
  - What technologies and design did we apply?

• Testing overview
  - What are the quality scenarios and test points?

• Project Conclusion
  - What did we learn about Open Source Middleware?
Project Background

• What is our core business?

• What is our software environment?

• What is our software need?

• What are our project goals?
Core Business

- We perform all of the Trajectory Analysis, Pre-mission Design, Operations, and Post-Flight Assessments for the Manned Space Program
  - We support the Space Shuttle Program (SSP), International Space Station (ISS), Constellation Program (CxP) and other vehicles
  - We support Ascent, Orbit, Rendezvous, and Entry Phases
Core Business

- **Analysis:**
  - How changes in the vehicle, environment, or procedures impact the vehicle's trajectory, performance, and margins.
  - Both nominal and off-nominal conditions.
  - We answer “what if” type questions

- **Pre-Mission Design:**
  - Determines what the specific trajectory will be for an upcoming launch or mission phase
  - Determines the margins for the flight
  - Results in products that are consumed by the rest of the program
Core Business

- Operations:
  - Monitor the vehicle, Determine where the vehicle is, Tell the vehicle how to get where it needs to go
  - During critical phases (Ascent, Rendezvous, and Entry) system failures cannot be tolerated and performance is a key concern
    - Possible loss of crew and loss of vehicle
  - Done by flight controllers in a highly collaborative environment. A lot of concurrent use of data and user interaction with the system.
• Software environment

- Mission Critical Flight Control Operations
- Software systems with legacies dating 40 years.
- Highly Customized infrastructure
- New space program requiring considerable new functional capabilities

• Software Need

- Highly available computing system
- Sustainable infrastructure and applications
- Modifiable infrastructure, applications, and business models
Project Background

• Project Goals

  - Determine if Java technology is appropriate for high availability.

  - Determine if Java Open Source Middleware is a viable alternative to custom infrastructure.

  - Determine if JBoss Enterprise Application Platform can support our use cases and quality scenarios.
System Architecture

- What are the technologies employed?
- What is the test system architecture?
- What is the test application design?
System Architecture

• Technologies
  - Blade Server
  - RHEL
  - Sun JVM
  - MySQL (clustered for failover and redundancy)
  - JBoss AS (EJB3 and Service Beans)
  - JBoss Clustering (failover and redundancy)
  - JBoss Messaging (data distribution)
  - JBoss Cache (object state replication)
  - JGroups (failover of services)
  - Hibernate (object persistence)
System Architecture
System Architecture

Clustered Server Configuration

JBoss AS Node

- << component >>
  - Clustered JBoss AS
- << component >>
  - MySQL API daemon

MySQL Database Node

- << component >>
  - MySQL Data Server

MySQL Database Node
- MySQL Database Node
- MySQL Database Node
- JBoss AS Node
- JBoss AS Node
- JBoss AS Node
- JBoss AS Node
- JBoss AS Node
System Architecture

Supplier
External Data

Process Telemetry
Enterprise Component
AS Lifecycle Managed

Perform Work Units
Enterprise Component
AS Container Managed

Trajectory Broadcast
Java Messaging Service

Java Topic

TP - Test Point
System Architecture

**Application Server Node**

**Process Data**
- Enterprise Component
- AS Lifecycle Managed

**Perform Work Units**
- Enterprise Component
- AS Container Managed

**Replication**
- Enterprise Component
- Network Cache

**High Speed Broadcast**
- Java Messaging Service

**Replication**
- To N-AS Nodes

**Client Inputs**

**TP-#1**

**TP-#2**

**TP-#3**

**TP-#4**

**TP-#5**

**TP-#6**

**TP - Test Point**

**TP -#2.1**
System Architecture

MVC Use Case

<< component >>

Rich Client Workbench

- SwingObserver
- TransferObject
- MessageListener

Clustering JBoss Application Server

- SessionBean
- Topic
- Entity

JMS, RMI, JNDI

JMS, JPA, JTA, JNDI, RMI
System Architecture
System Architecture

Task Executor Use Case

<< component >>
Rich Client Workbench

Client
Command

Executor

<< component >>
Clustered JBoss Application Server

Request Queue
Temporary Queue

TaskMDB
TaskSessionBean
System Architecture
Testing Overview

• What are the primary quality attributes and measures?

• What are our test criteria?

• What technologies were tested?

• What are our test results?
Primary Attributes & Measures

- Reliability
  - failover and fault recovery time in seconds of both processing load and connections

- Performance
  - processing load in compute units
  - data latency in seconds
  - data distribution in megabytes per second
  - data persisted in megabytes per second

- Scalability
  - number of client nodes
  - number of client workbenches
Test Criteria

- Reliability
  - program requirements for failover recovery
  - benchmark for current failover recovery

- Performance
  - benchmark of currently required processing units
  - program requirements for latency
  - benchmarks of current latency

- Scalability
  - benchmark of current number of clients
  - benchmark of current volume of data distributed
  - benchmark of current volume of data persisted
Technologies Tested

- JVM
  - computational performance

- JBoss Messaging
  - volume and latency of data distribution
  - failover of connections

- Hibernate/MySQL
  - volume and speed of data access
  - failover of data access

- JBoss AS
  - supported processing load
  - latency of processing steps
  - failover of processing
Test Architecture

Computational Load Modelling

- 1 work unit = 6 hour trajectory numerical integration
- total high speed processing in 62 work units

Processing Load Modelling

- computational load
- data access load
- messaging load
Testing Results

- Java Virtual Machine
  - algorithms complete is less than a fifth of the required time.
  - heap size stable during loading test
  - garbage collection consistent during testing
  - garbage collection does not impact latency of processing
  - Sun JVM proven to be most reliable
  - JRocket JVM generated segmentations faults
Testing Results

• JBoss MQ (example test case)

- 4 node cluster running consecutively for 1 day
- Continued for 5 more days with 3 node cluster
- 28.5 million work units submitted to cluster
  - 3 sets of 20 work units submitted per second
  - Round-robin load balancing policy across cluster
- 1.1 TB transferred, 2.4 MB/s average,
  - single HA Queue
- Master node failure and recovery successful
  - Cause: inadvertent machine reboot
Testing Results

- JBoss Messaging (example test case)
  - Blade server with dual CPU
  - 2MB RAM for JBoss AS
  - 6 node cluster, 4 JBoss AS nodes, 2 MySQL nodes
  - 480 work units submitted to cluster per second
    - 3 sets of 40 work units to each JBoss AS node
    - Round-robin load balancing policy across cluster
  - 480 clustered database record merges per second
  - 480 JMS messages routed per second
  - Constant load for fifteen minutes
  - Consistent one second processing times for each one second batch of request
Testing Results

• Hibernate/MySQL
  - Standard Java Application with clustered MySQL
    - 20 threads performing 25K transactions
    - 1800 inserts/sec using Hibernate
    - 2800 inserts/sec using JDBC
  - Integrated JEE Application with clustered MySQL
    - greater than 1000 merges per second per node
    - greater than 1MB per second per node

Note: MySQL connection time limit is configurable, but connection time required during initial start up creates need for 10 sec timeout setting
Testing Results

- JBoss AS
  - supports number of required sequential pipe and filter style processing steps
    - individual filter steps include computational and persistent data access
    - individual pipe steps include messaging
  - support concurrent asynchronous processing load
  - supports processing load greater than 5 times need
  - processing state failover in less than 30 sec
Project Conclusions

• Java application computational performance is acceptable given the proper hardware resources.

• JEE specification and supporting technologies can be used as a platform for mission critical operations.

• Open Source implementations of JEE, such as JBoss Enterprise Application Platform, are a viable alternative to custom enterprise solutions.

• NASA could elect to focus its expertise on aerospace solutions and reuse existing open source middleware solutions.
Next Steps

- Explore Additional Technologies
  - Web Services
    - to expand access to business components to non Java clients
  - Enterprise Service Bus
    - to expand interoperability with external systems
  - Business Process Management
    - for orchestration of business workflows
  - Rich Client Platform
    - to integrate and manage side client components