



Goddard Space Flight Center



Java for Flight Software

Ricardo J. Hassan II

Jet Propulsion Laboratory

California Institute of Technology

Software Engineering Technology Infusion Group

4th Quality Mission Software Workshop

Dana Point, CA, May 7-9 2002

Motivations

- Leverage advantages of Java programming language vs. C/C++
 - Easier to use
 - Fewer defects
 - Automatic Memory Management
 - Faster Development Time
 - Available Libraries
 - Strictly Object-Oriented
 - Encourages better designs
 - Write Once (Carefully), Run Everywhere (Conditionally)

Thrusts

- Two separate efforts
 - RTSJ Test Suite
 - For AFRL, In cooperation with The Boeing Co.
 - Two goals
 - Develop a set of requirements for Java platforms to be used for flight applications
 - Create a set of tests of verify that a platform meets those requirements
 - Java Flight Software Prototyping
 - Demonstrate the feasibility of the RTSJ by developing a working attitude control software module in Java,
 - Based on real mission software architecture (DS1)
 - Designed to meet real mission requirements
 - » Functionality
 - » Precision
 - » Performance
 - » Flight like platform

RTSJ Test Suite

- Intended to exercise all required functionality of RTSJ platform
 - Threads
 - Scheduling
 - **Memory Management**
 - Timing and Timers
 - Asynchronous
 - Dynamic Class Loading

RTSJ Memory Management

- Motivation
 - In past flight missions, all memory allocation is done at initialization time, and forbidden thereafter
 - With robust memory management, this can be relaxed
 - Easier and potentially more efficient
 - Memory Management, (i.e. pointers) is one the major sources of defects in flight software
 - Java memory model largely protects developer from such errors
- Caveats
 - Automated memory management induces latency into the system
 - When picking a platform, garbage collection scheme(s) and memory areas should be chosen and used wisely
 - For now, may be harder to meet performance requirements

RTSJ Memory Management

- Test Types
 - Object Allocation Performance
 - CPU Throughput vs. Memory Area
 - CollectionPerformance
 - Collection Induced Latency
 - Finalization Performance
- All tests run in all (applicable) memory areas
 - Stack
 - HeapMemory
 - LTMemory (Linear Time Memory)
 - VTMemory
 - ImmortalMemory

Environment

- Development
 - Eclipse IDE
 - Open Source, Freely Available <http://www.eclipse.org>
 - Standard Sun JDK 1.2 compiler
 - <http://www.java.sun.com>
- Runtime
 - Intel Pentium II 400MHz w/ 128 MB RAM
 - RTSJ Reference Implementation
 - Timesys Real Time JVM
 - <http://www.timesys.com>
 - Timesys Linux GPL 3.0
 - Note: This is not a deployable platform, it is for evaluation purposes only



Results

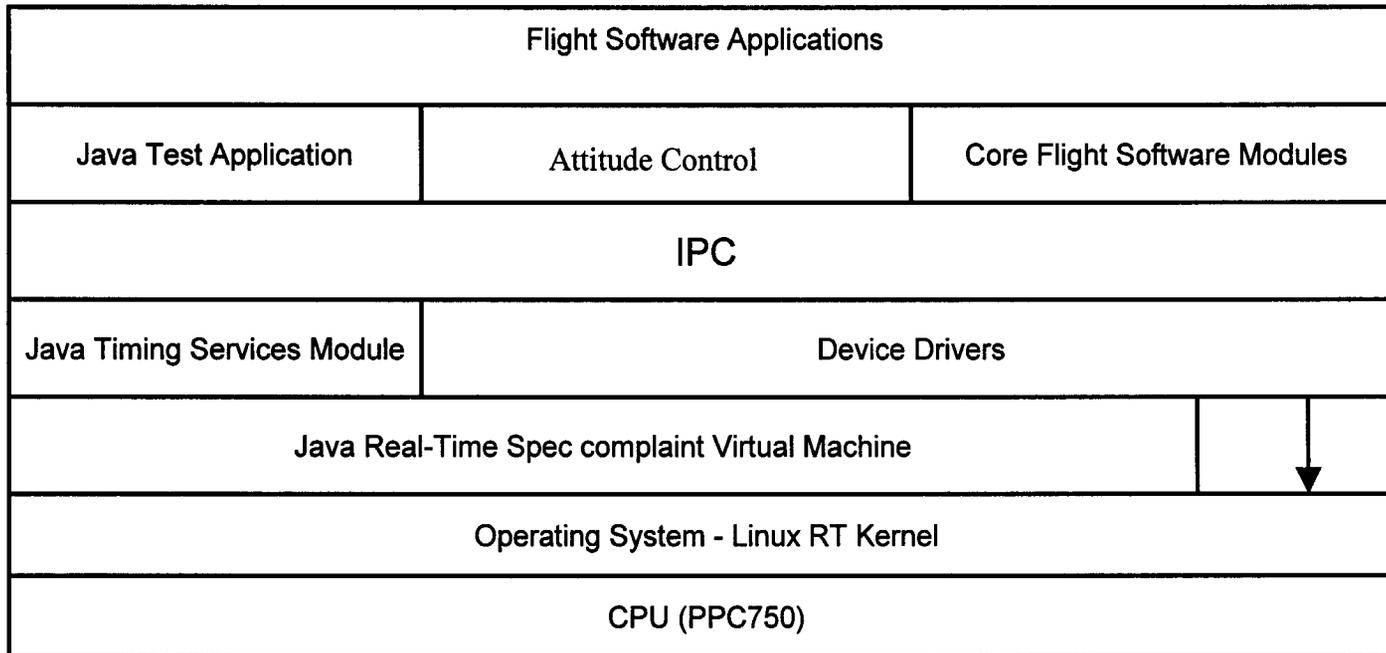
- TBD

Flight Software Prototyping

- Goal - Demonstrate a working flight software system in Java as a proof of concept
 - Real-Time control
 - Run closed loop with spacecraft simulation (DS1)
 - Mission-critical performance
 - 8 Hz control loop
 - Flight-like platform
 - CPU (PPC 750)
 - Real-Time Linux
 - VxWorks not a viable option for pure Java system
 - Object-Oriented Design
 - Keeping reuse in mind
 - Maintain external interfaces

Architecture

- Layered View



Schedule

- Begin ACS Design Work 4/1/02
- Design Review 5/15/02
- Real-time Java FSW prototype 9/1/02
 - Other evaluation/analysis tools 9/1/02
- Public (internal) Real-time Java prototype demonstration 9/20/02
- Final report 9/20/02
- EOY TIPR Package 9/20/02

Looking Ahead

- Make the lab aware that Java is an advantageous choice for flight software development
- Look for (flight) missions which would benefit from using Java for flight software implementation
- Develop library of reusable classes for mission-critical software