

---

# Safety Critical Java

27 September 2007

Doug Locke  
Locke Consulting LLC  
[www.douglocke.com](http://www.douglocke.com)  
[doug@douglocke.com](mailto:doug@douglocke.com)

for

THE *Open* GROUP

[doug.locke@opengroup.org](mailto:doug.locke@opengroup.org)



# Safety Critical Java

---

- Let's discuss Safety-Critical Java
  - Questions:
    - What is it?
    - What is being standardized?
    - What is in it now?
    - What are we still discussing?
    - What is the status so far?

# Background

---

- The Open Group is serving as the Specification Lead for a new Java Specification Request (JSR-302) under the Java Community Process.
  - I am leading the JSR-302 development on behalf of the The Open Group
- Thanks! This effort has been supported not only by the organizations sending the Expert Group members, but also a large part of the early work was supported in part by the EC's HIJA project
  - The Open Group's initial leadership of this effort was funded by the HIJA project
  - Many of the Expert Group members were partially funded by the HIJA project

# Introduction

---

- Goal – a specification for Safety Critical Java capable of being certified under DO-178B Level A and other safety critical certification standards
  - Certification implies a small, reduced complexity infrastructure (i.e., JVM)
  - Emphasis is on defining a minimal set of capabilities required of safety critical applications using Java implementations

# Creating the Specification

---

- ❑ A Java Specification Request (JSR) was submitted to the Java Community Process
  - The JSR was approved, and is designated JSR-302
  - This means that three things will be created:
    - A Safety Critical Java specification
    - A Reference Implementation
    - A Technology Compatibility Kit
- ❑ Maximum public exposure is planned throughout development
  - The Open Group will continue to keep non-Expert Group members informed about progress

# Expert Group

---

- Under the JCP, specifications are created by a Specification Lead (in this case, The Open Group) and an Expert Group
  - Robert Allen (Boeing)
  - Scott Anderson (Verocel)
  - \*Greg Bollella (Sun)
  - Ben Brosgol (AdaCore)
  - Arthur Cook (Alion)
  - Mike Fulton (IBM)
  - Allen Goldberg (Cadence)
  - David Hardin (Rockwell)
  - \*Thomas Henties (Siemens)
  - \*James Hunt (aicas)
  - \*Doug Locke (TOG & Locke Consulting)
  - Johan Nielsen (DDC-I)
  - \*Kelvin Nilsen (Aonix)
  - \*Martin Schoeberl (T.U. Vienna)
  - Takeoka Shozo (AXE, Inc.)
  - Sunil Srivastava (Apogee)
  - Joyce Tokar (Pyrrhusoft)
  - Jan Vitek (Purdue U.)
  - \*Andy Wellings (U. of York)

# Specification Content (1)

---

## □ Application Structure

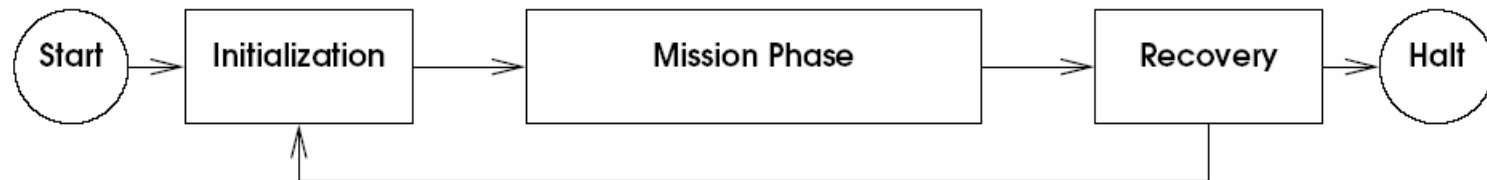


Figure 3.1: Safety Critical Execution Phases

- Data structures created at initialization in Mission Memory
- Scoped memory areas used for limited dynamic allocation
  - Restrictions on multiple memory area access
- Application startup will not require heap memory (uses a Safelet)

# Specification Content (2)

---

- Three Compliance Points (Levels 0, 1, 2)
  - Level 0 provides a cyclic executive (single thread), no wait/notify
    - Synchronization ignored
    - No threads – only periodic Async Event Handlers
    - Local memory in local scoped memory – emptied each period
    - All exceptions must be preallocated
  - Level 1 provides a single mission with multiple schedulable objects, no wait/notify
    - Multiple concurrent schedulable objects
    - Dynamic scoped memory allowed, but not shared
    - All exceptions must be preallocated
  - Level 2 provides nested missions with (limited) nested scopes
    - May have NoHeapRealtimeThreads, wait/notify, nested shared scoped memory (must be statically checkable)
    - All exceptions must be preallocated

# Specification Content (3)

---

- ❑ Garbage Collector
  - None required
- ❑ Scoped memory
  - Nesting restricted to Level 2
  - Reference safety must be statically analyzable
- ❑ All Schedulable Objects will be non-heap
- ❑ Initialization initializes each class in user-defined order
- ❑ Java memory model follows JSR-133
  - Circular reference initialization disallowed

# Specification Content (4)

---

- ❑ Support for SMP's to follow RTSJ lead
- ❑ No Finalizers
- ❑ Requires Priority Ceiling for priority inversion management in Synchronized methods
  - No synchronized blocks other than “this”
- ❑ Priority Inheritance not required
- ❑ Class loader not required
- ❑ Raw memory included, but Physical Memory not required
- ❑ Required annotation to support static analyzability defined
  - Separate presentation on this tomorrow

# Issues Being Discussed

---

- ❑ Reflection – would like to limit it, but may be needed.
- ❑ Use of final will be analyzed before completion of specification
- ❑ List of required library classes & methods has been started, but is still being completed

# Summary

---

- ❑ Specification draft writing assignments are being completed
- ❑ Still behind schedule, but we should have an initial specification this Fall or early Winter
- ❑ Reference Implementation expected to be open source RTSJ-compliant Java executable on any RTSJ-compliant JVM
- ❑ Technology Compatibility Kit still to be worked
- ❑ Strong Expert Group
- ❑ Stay tuned!