ISO/IEC JTC 1/SC 22/OWGV N 0088

Liaison Report:
JSR-282 (Real-Time Specification for Java)
JSR-302 (Safety-Critical Java Technologies)

ISO/IEC JTC1/SC22/OWGV Meeting
Ottawa, Canada

18-20 July 2007

Ben Brosgol • brosgol@adacore.com

AdaCore
The GNAT Pro Company

European Headquarters:

8 rue de Milan
75009 Paris France

+33-1-4970-6716 (voice)
+33-1-4970-0552 (FAX)

North American Headquarters:

104 Fifth Avenue, 15th Floor
New York, NY 10011

+1-212-620-7300 (voice)
+1-212-807-0162 (FAX)

www.adacore.com

Introduction

Language vulnerability ⇨ application susceptible to safety hazard or security failure

Main language requirements for avoiding such hazards / failures

- Reliability
- Predictability
- Analyzability

A dilemma

- Features that are beneficial in general may complicate certification against safety or security standards
  - Object-Oriented Programming
  - Generic templates
  - Inline expansion
  - Exception handling
  - Concurrency features

General purposes languages (C, C++, Ada, Java, …) are too large / complex

- Subsetting is required
- Enforcement of subset should be automatable

Java for safety-critical / high-security systems

Some advantages

- Reliability
  - Avoids “buffer overflow” problems and “dangling reference” issues
- Predictability: precisely defined semantics, in general
  - Order of expression evaluation, “precise” exception behavior
- Analyzability
  - No uninitialized variables; no unreachable code”
  - Built-in security model

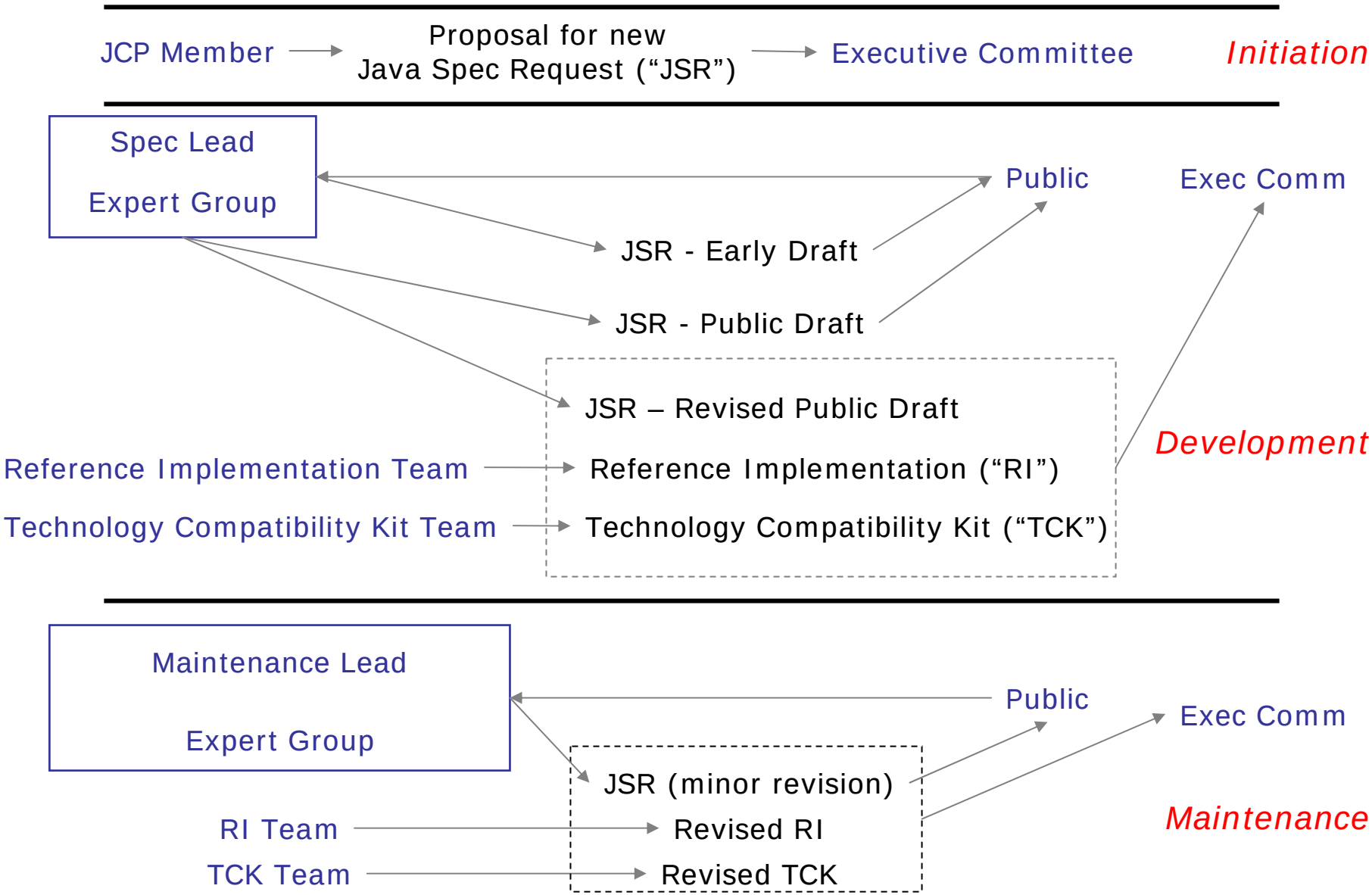
Some issues

- Reliability
  - C-based syntax (literals, “dangling else”), low-level thread model
- Predictability
  - Thread-related issues (priority semantics, unbounded priority inversions)
  - Garbage collection issues
- Analyzability
  - Unconventional execution model (JVM)
  - Language and API size/complexity
  - Built-in security model

Java Community Process ("JCP")

Sun-administered process for augmenting / modifying the Java platform

www.jcp.org/en/procedures/jcp2



Background – Real-Time Specification for Java

What is the Real-Time Specification for Java (JSR-001, JSR-282)?

- API + JVM constraints designed to give real-time predictability to Java platform

Addresses several major issues with Java for real-time systems

Imprecision of thread semantics for scheduling (role of priorities)	**"RealTime Threads" + priority-based scheduler, FIFO within priorities, for both wait queues and locks**
Possibility of unbounded priority inversion	**Monitor control policies:** **Priority Inheritance, Priority Ceiling Emulation**
Garbage collection interference / latency	**Non-GC'ed "memory areas"; special threads that are not allowed to reference the heap**
Inadequate functionality	**Asynchrony, high-resolution time, low-level features**

Status

- Original spec (JSR-001) completed in 2001, led by IBM (Greg Bollella, Peter Haggar)
  - Several maintenance releases since then, led by TimeSys (Peter Dibble)
  - Several commercial implementations available
- Minor update (JSR-282) now in progress, also led by TimeSys (Peter Dibble)

RTSJ not appropriate for safety-critical systems: analyzability issues

- Complex semantics (e.g., Asynchronous Transfer of Control)
- Scoped memory rules requiring run-time checks, complicate analysis

Background – Safety-Critical Java Technology

What is Safety-Critical Java Technology (JSR-302)?

- RTSJ profile, designed to allow certification to safety standards such as DO-178B Level A

Approach

- Remove unneeded classes, methods from RTSJ
  - Example: no asynchronous transfer of control
- Do not require Garbage Collection
- Require specific approach (Priority Ceiling Emulation) for priority inversion control
- Add statically-checkable annotations to facilitate analysis
  - Avoid run-time checks implied by RTSJ rules for memory reference assignment
- Define multiple levels of compliance, corresponding to required application generality
  - Most restrictive level reflects classical single-threaded “cyclic executive”
- No attempt to address general Java analyzability issues (e.g. OOP)

Some open issues

- Specifics of statically checkable annotations

Status

- In-progress, spec expected Q1 2008, led by The Open Group (Doug Locke)
- Inspired by work from HIJA (aicas, Univ. of York, …) and Aonix
- Several related commercial implementations available

Resources

Books

- P. Dibble; *Real-Time Java Platform Programming*; Prentice-Hall; 2002; ISBN 0130282618
- A. Wellings; *Concurrent and Real-Time Programming in Java*; John Wiley & Sons; 2004; ISBN 047084437X

Websites

- JSR-1: jcp.org/en/jsr/detail?id=1
- JSR-282: jcp.org/en/jsr/detail?id=1
- P. Dibble (spec. lead), R. Belliardi, B. Brosgol, D. Holmes, and A. Wellings. *Real-Time Specification for Java™, V1.0.1*, June 2005. www.rtsj.org
- JSR-302 (Safety-Critical Java Technology): jcp.org/en/jsr/detail?id=302