JSR 282 Review

25th of September 2013

James Hunt
Agenda

• Goals
• Background
• Justification
• History
• Technical Scope and Features
• Deliverables: Specification, RI, TCK, IP, Other
• Schedule
• Publicity, Collaboration, Participation, and Transparency
• Implementation Notes
• Issues
• Questions, discussion, next steps
Goals

• The original goal was to addresses some of the simpler enhancements that have been requested in the Real-Time Specification for Java (RTSJ) of which 21 where listed explicitly.

• This has lead to a re-evaluation of the specification to clarify ill defined parts of the specification and complete partially defined features such a user defined clocks and happenings.

• Providing better integration with current conventional Java implementations has also become important.
Background

- Update to JSR-1
  Real-Time Specification for Java (RTSJ)
- RTSJ refines Java semantics and adds APIs for realtime
  - no changes to Javac necessary
  - fully compatible with conventional JVMs
- Targets all platforms
- Was included in J2ME
- This is a single JSR platform
- Necessary for extending Java ecosystem into realtime and embedded systems
• The RTSJ was a good starting point for using Java for realtime and embedded applications.
• JSR 282 updates the RTSJ to the current state of the art by clarifying its semantics and filling in major gaps.
• The RTSJ extends the Java ecosystem into deeply embedded systems, especially where realtime response is required.
• This is not a new standard, but a refinement of an existing one based on field experience.
• Required to make further inroads in replacing C and C++ in embedded systems, thus broadening the Java ecosystem.
History

• The RTSJ was completed in December 1998
• JSR 282 was approved in August 2005
• Early Draft Review was started in March 2009 and completed in May 2009
• Peter Dibble left TimeSys in May 2010
• aicas became specification lead in October 2012
• Just finished IP transfer from TimeSys in August 2014
• The EG consists of the following members:
  – Industrial: aicas, IBM, Atego, Ethan Blanton
  – Academic: Andy Wellings (realtime system expert)
  – Other Communities: Ben Bros gol (Ade Industrial)
• The EG meets weekly by teleconference
• The EG communicates internally with webex, e-mail, and an SVN repository
Technical scope and features

- **Raw Memory**
  - Typed device access
  - Factory Base
- **ActiveEvents**
  - Unify API for Timer, Happening, & POSIXSignal
  - Happening as Object
  - User defined Clocks
- **CPU Affinity**
- **Interrupt Service Routine Support**
Technical scope and features

- **Stateful Events & Handlers**
  - Object and long payloads
  - POSIX Realtime Signals

- **New Scope Types**
  - PinnableMemory (support for PC pattern)
  - StackedMemory (support for JSR 302)

- **Modularization**
  - base and three optional modules
  - make selectable at a reasonable granularity
Implementations

• There are not yet any publicly available implementations besides the TimeSys RI
• Two other vendors testing features on their own JVM
RI and TCK development

- The TCK is an extension to the RTSJ TCK and is being developed by the EG
- TimeSys had published an RI
- aicas is developing a new RI
• The licenses will be broadly similar to the RTSJ
  – just received text from TimeSys
  – in legal review
• We have not had any, but will set up a Contributor Agreement similar to that of OpenJDK
• The collaboration tools are free to use as EG member
  – Webex guest
  – open source tools
• Completed IP transfer from TimeSys
Other deliverables

• The Specification is more than just the JavaDocs.
• It includes
  – Semantics and
  – Rationale (including some examples)
• EG will consider providing
  – additional documentation,
  – user's guide,
  – sample code, and
  – FAQ
• How do other EGs integrate this with their work?
<table>
<thead>
<tr>
<th>Event</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Second Draft Review Start</td>
<td>Waiting on JCP</td>
</tr>
<tr>
<td>License Legal Review finished</td>
<td>07 Oct. 2014</td>
</tr>
<tr>
<td>Publish Licenses</td>
<td>10 Oct. 2014</td>
</tr>
<tr>
<td>EG Face-to-Face before JTRES (Full document review)</td>
<td>12 Oct. 2014</td>
</tr>
<tr>
<td>Next RI Release</td>
<td>1 Dec. 2014</td>
</tr>
<tr>
<td>TCK finished</td>
<td>End Jan 2015</td>
</tr>
<tr>
<td>Final Review Start</td>
<td>Feb 2015</td>
</tr>
</tbody>
</table>
Publicity

• Java Technology for Real-time and Embedded Systems
  – yearly conference dedicated to RTSJ and SCJ issues
  – meets every year since 2003
  – more than 100 papers published
  – jtres2014.compute.dtu.dk/

• Open Group Real-Time and Embedded Forum
  – regular updates presented
  – http://www.opengroup.org/sanfrancisco2014/rtes
Collaboration with other community groups

• We are collaborating with JSR-302 to ensure maximal compatibility between the specifications.
  – issues where collected from the JSR-302 EG
  – all changes have been feed forward to JSR-302
  – some small changes where made to support implementing JSR-302 on the RTSJ

• Three EG members are also JSR-302 members

• We also collaborate with the Open Group Realtime and Embedded Forum.
Participation and transparency

• Provide a pointer to the JSR page on JCP.org
  – [Reviewers: check that is this up to date. Does it point to the
    JSR’s project page and/or explain how to participate?]

• Provide a pointer to the “JSR project website" (eg, on
  Java.net.)
  – [Reviewers: how much content is here (how many
    pages)? Is the online project easy to navigate? Does it
    clearly explain how to participate?].
Issue tracker

- The work is nearly complete.
- Processed 38 specification issues (major issues to RTSJ that drove the JSR)
- 8 issues where dropped (considered detrimental or too complex)
- 3 issues where delays to the next RTSJ version (would overly delay specification release)
- 2 Superseded by later issues
- 3 issues are not completely resolved
- 22 are finished
- User issues will be tracked from upcoming Draft Review
Mailing lists or forums

• This is a new requirement for us.
• Mailing list: jsr282-feedback@aicas.com
• Twitter: @realtimejava #RTSJ
• Discussion: http://www.linkedin.com/groups/RTSJ-8147216?gid=8147216
• The Spec Lead has posted a few messages to twitter.
• We expect that the Draft Review that we are preparing will bring traffic to the discussion page
• These are listed on JCP.org
• Again, this is a new requirement for us.
• Spec revisions are available on the JSR-282 page: https://www.aicas.com/cms/en/rtsj
• Old versions will be maintained there as well.
Adopt-a-JSR

- Again, this is a new requirement that we were not tracking until now.
- What do we have to do?
Implementation notes

• Specifying a realtime language extension for realtime programming is quite complex:
  – differing scheduling requirements:
    fairness vs timeliness
  – Synchronization is more critical
  – must pay more attention to allocation
  – must specify timing behavior without loss of portability
Issues

• Should be RTSJ 2.0, not 1.1.
• How to include key API in OpenJDK?
• Where does J2ME fit in today?
Questions, discussion, next steps
Thank you!
http://jcp.org