Purdue’s team cooperates with key software vendors and the JSR-302 expert group on a new standard for Safety-Critical Java systems. With the goal to develop applications and infrastructures amenable for certification under safety-critical standards, the team provides the first open-source Java implementation suitable for the domain of safety-critical software systems.

Software reliability is a major issue for real-time systems, many of which are safety critical. Java brings a plethora of modern language features that make software engineering more cost-effective and safe, and it is thus an appealing platform for safety-critical applications.

Safety-Critical Java

The driving design principles in SCJ Specification are reduction of system’s complexity and cost of certification.

An SCJ compliant application will consist of one or more missions, where a mission consists of a bounded set of periodic event handlers. The thread model is largely restricted to periodic and asynchronous event handlers to simplify the schedulability analysis. The concept of missions and sub-missions leveraged to higher levels reintroduces dynamic features of the real-time Java in a safer form. For each mission, a dedicated block of memory is identified as the mission memory. A set of scoped memories with restricted hierarchy can be used for each schedulable object. Heap memory is not allowed. The simple motivation for this restricted memory model is to allow static analysis of the memory usage.

As the use of real-time Java has become more widespread, the demand for Java in real-time applications with safety requirements has led to an effort to define a new standard - Safety-Critical Java Specification (JSR-302 under the Java Community Process).

As a safety-critical system is a system whose failure or malfunction may result in death or serious injury to people, or loss or severe damage to equipment. For those reasons, safety-critical applications require an exceedingly rigorous development, validation, and certification process.

A growing complexity of safety critical software calls for high-level development technologies. To face this challenge, various approaches to real-time execution of Java have proven their worth in numerous commercial and defense applications. Finally, the Real-time Specification for Java has extended the Java platform with a range of features needed for real-time computing.

As the use of real-time Java has become more widespread, the demand for Java in real-time applications with safety requirements has led to an effort to define a new standard - Safety-Critical Java Specification (JSR-302 under the Java Community Process).
The compliance levels enable construction of variously complex and multi-model safety-critical systems, reducing thus the cost of their implementation and certification.

**Safety-Critical Virtual Machine**

Development of an SCJ compliant infrastructure is a non-trivial task that comprises several challenges. Safety-critical software runs on top of a compact SCJ library supporting Level 0-2 compliant applications. The library itself is communicating closely with a dedicated Virtual Machine constructed to effectively support application execution. Finally, the VM itself runs on top of a real-time operating system - Fig. 2.

The most challenging part of the oSCJ project represents development of a Virtual Machine compatible with the specification. The team currently develops a new VM based on Purdue’s successful OVM. Furthermore, SCJ extensions for industrial FijiVM are being developed in collaboration with Fiji Systems LLC. Both VMs allow safety-critical experts to configure the infrastructure to the operational requirements of a particular mission, while emphasizing the performance of the resulting system. To optimize the performance, SCJ code is compiled to C.

**Current Status**

As soon as the certifying authorities accept JVMs that implement SCJ, the use of Java will lead to higher productivity in the development of safety-critical applications. Purdue’s oSCJ project is on the way to be thus the first open-source implementation in the field.

Currently, oSCJ project already provides the technology compatibility kit and the static checker. Recent efforts are focused on the development of a VM supporting Level 0 compliant applications. The Level 0 compliance brings a simplified computation model that enables high optimization of the whole infrastructure, achieving a minimal footprint and a performance that is comparable even to C programs. The first release of oSCJ VM is planned for 2010.

**MORE INFORMATION**

**PUBLICATIONS**


**CONTACT**

Ján Vitek, jv@cs.purdue.edu
Gary Leavens, leavens@eecs.ucf.edu
Ales Plšek, aplsek@purdue.edu

**PROJECT WEBPAGE**

http://www.cs.purdue.edu/homes/plsek/soft/scj/

**TEAM MEMBERS**

Jan Vitek, Purdue University
Ales Plšek, Purdue University
Lei Zhao, Purdue University
Daniel Tang, Purdue University
Tomas Kalibera, Charles University
Veysel H. Sahin, Purdue University
Gary Leavens, University of Central Florida
Ghaith Haddad, University of Central Florida

The target hardware platform for oSCJ project is Xilinx FPGA board running RTEMS/LEON3 - Fig. 3. The board is used both by NASA and ESA to execute satellite’s on-board software (Venus Express Mission 2005, Dawn Mission 2007) and provides a unique platform for extensive benchmarking and evaluation of the oSCJ project.