UNIVERSITY of York

From Java to Real-Time Java: A Model-Driven Methodology with Automated Toolchain

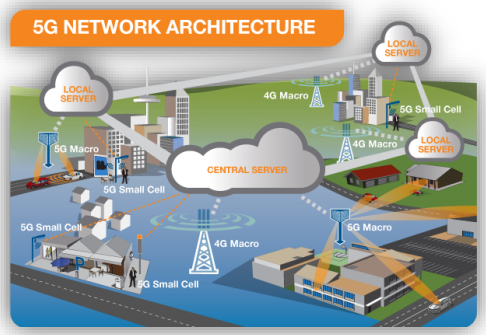
Wanli Chang, Shuai Zhao, Ran Wei, Andy Wellings, Alan Burns

Assistant Professor
Real-Time Systems Group
University of York, UK
wanli.chang@york.ac.uk

Outline

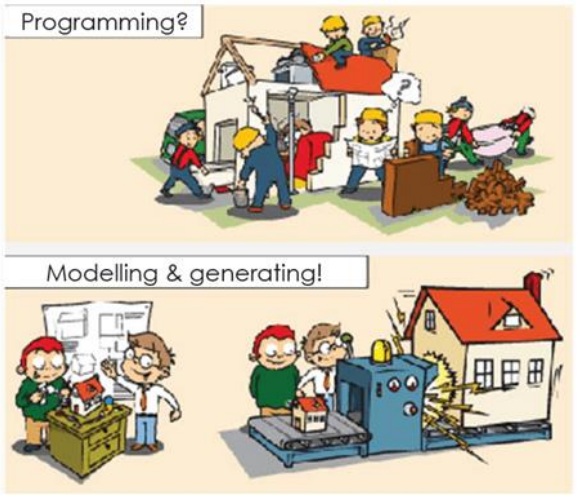
- Background and Motivation
- Model-based Engineering (MDE)
- Real-Time Specification for Java (RTSJ)
- Proposed Methodology and Toolchain
- Future Research Directions

Background and Motivation

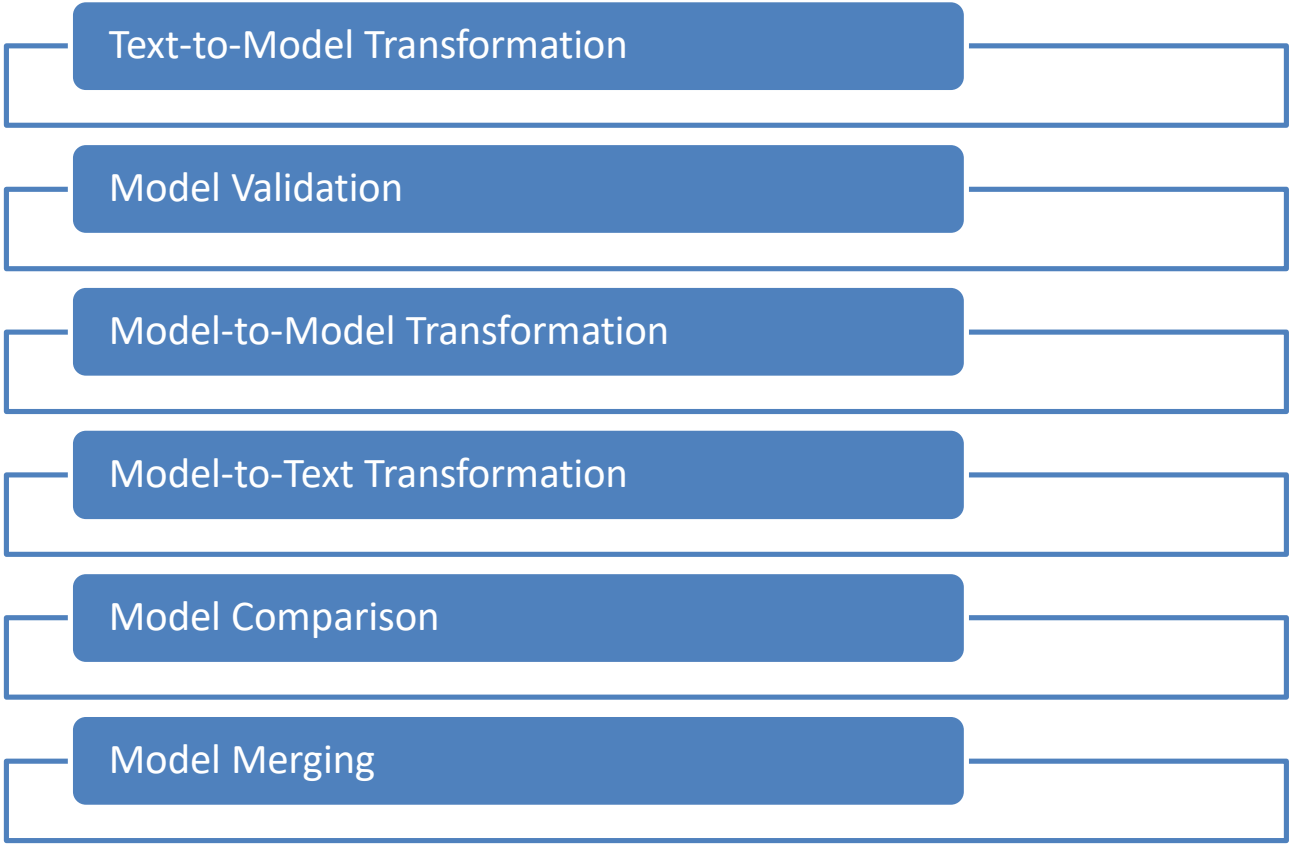


Model-based Engineering (MDE)

- A software engineering methodology that reduces the accidental complexity of software systems by promoting models that focus on the essential complexity of systems.
- Has been effectively applied to development of real-time systems for increased productivity and reduced human-related errors during system design and development.

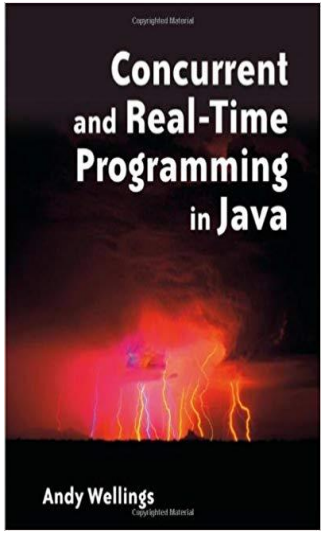


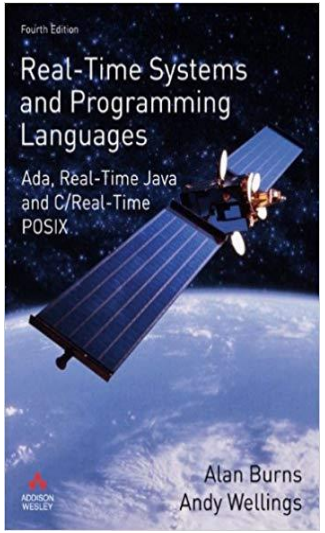
Model Management Operations

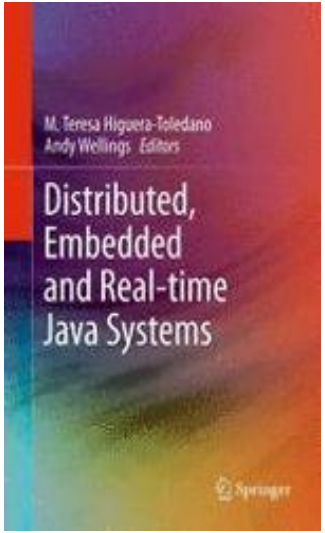


Real-Time Specification for Java (RTSJ)

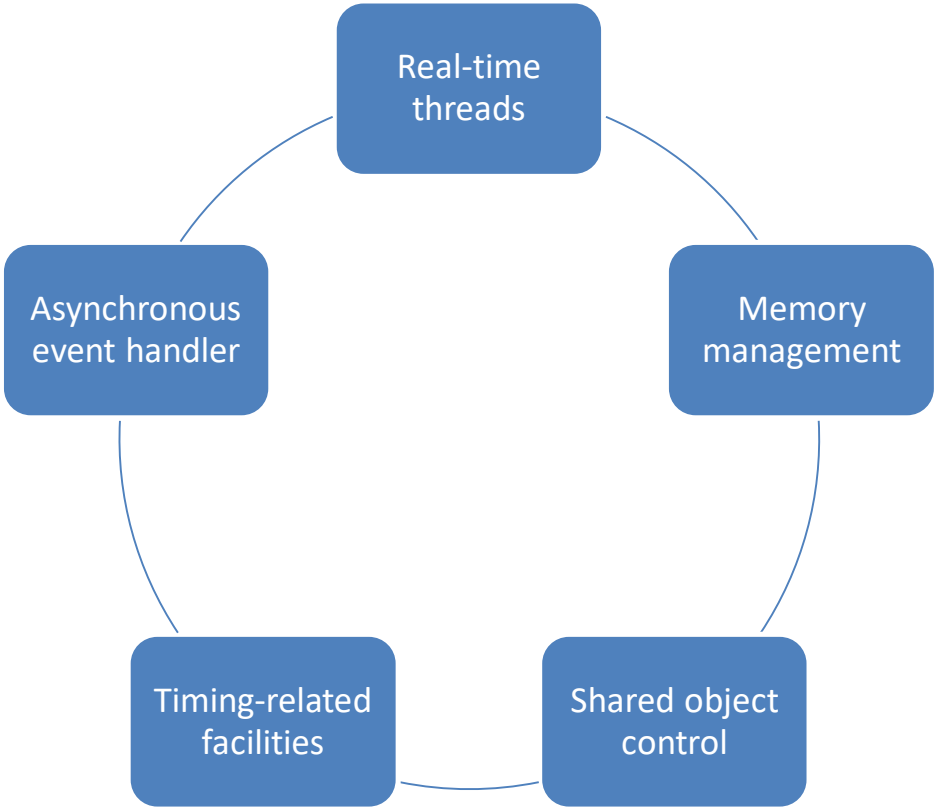
- RTSJ reserves the intrinsic advantages of Java and provides real-time facilities to guarantee the system temporal behavior.
- RTSJ consists of two major components:
  a) extensions from the Java programming language;
  b) modifications on the semantics of the standard Java Virtual Machines (JVM).







Real-Time Specification for Java (RTSJ)



RTSJ-compliant virtual machines:

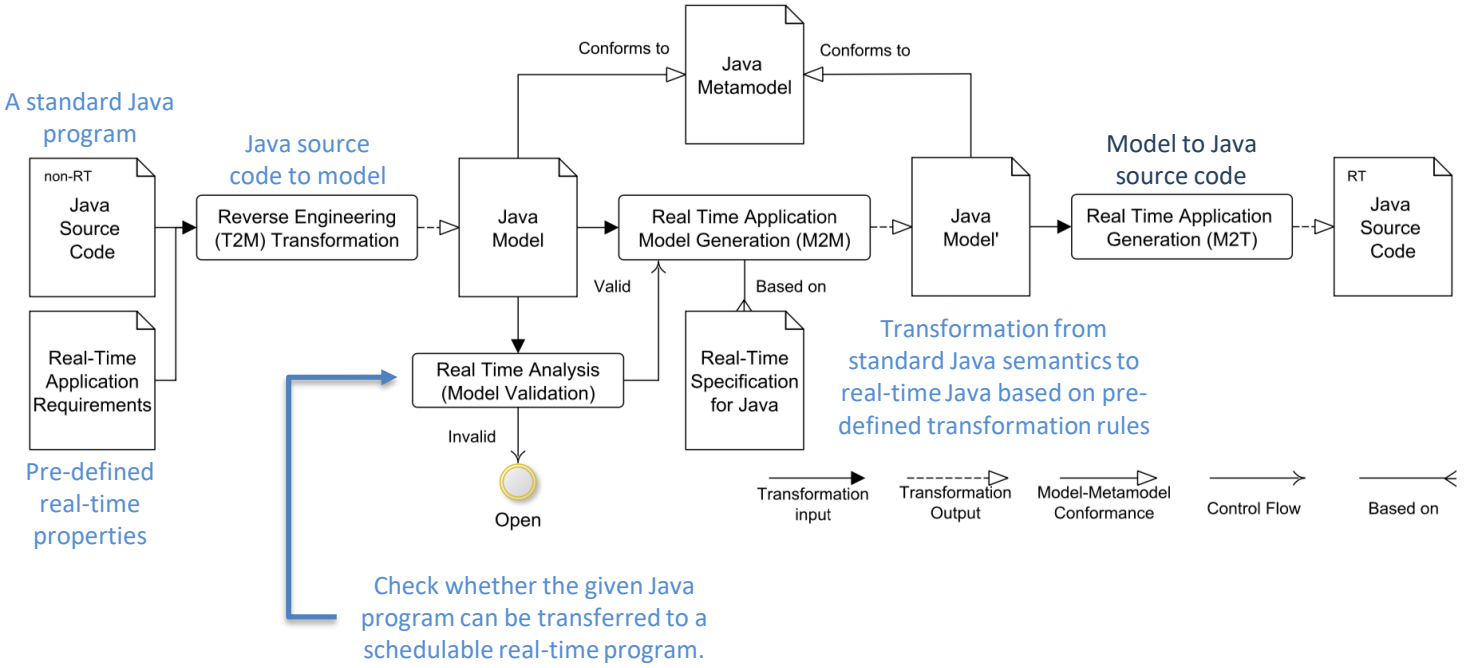
- **JamaicaVM v8.5 (RTSJ V1.0.2, Java 1.5)**
- TimeSys
- jRate
- OVM
- Aero JVM

Targeted systems:

- Uniprocessor system;
- Fixed-priority preemptive scheduling;
- Shared resources (priority ceiling);
- Periodic or aperiodic release;
- Real-time garbage collector (provided by JamaicaVM).

Proposed Methodology: Architecture

A Java to real-time Java automated toolchain



Possible Future Research Directions

- Allow self-defined memory management models when a real-time garbage collector is unavailable
  - ✓ Enforce an SCJ-like memory management model, which imposes restrictions towards the application structure but is still sufficient to provide the required functionalities.

- Support wider and more complex system models
  - ✓ Multiprocessor systems with various scheduling schemes (e.g., fully-partitioned, global).
  - ✓ In the presence of release jitters or shared resources (requiring multiprocessor resource sharing protocols).

- Address the open question where the system is found to be unschedulable
  - For systems where optimal scheduling solutions may not be available, reconfiguration of system scheduling parameters to achieve better schedulability would be desirable.
  - A search-based algorithm could be applied for searching threads' parameters and feasible resource sharing protocols that can achieve a schedulable system.

- Support other languages (e.g., C, Ada) and their real-time programming subsets
  - x For programming languages like C and Ada, reverse engineering facilities may not be available.
  - ✓ Model real-time systems from system specifications directly.
  - ✓ Generate implementation via code generation facilities such as AADL and UML.

Thank you !