Integrated Development of Distributed Real-Time Applications with Asynchronous Communication

Marc Schanne

International Workshop on Java Technologies for Real-time and Embedded Systems (JTRES)
26-28 September 2007, Vienna, Austria
Agenda

● Context of this Work – Requirements on Methodology
  software development for embedded systems, distributed real-time applications, asynchronous middleware framework, integrated development

● Basic Thoughts and Results
  asynchronous communication with design pattern, descriptive development method, use of non-functional requirements, possibility of static analysis, integrated development

● Related Work
  UML 2.0, UML-RT, DSL/DSM

● Methodology Overview
  another way of developing distributed safety-critical systems

● Consequences for RTSJ?
Context and Requirements

● Software Development for Embedded Systems
  ○ small devices, limited resources, reduced communication mechanisms (field busses, broadcast) fit well to asynchronous messaging
  ○ multidisciplinary development 'cos of embedding requires standardized method for component description
  ○ non-functional requirements (e.g., real-time needs) have to be integrated in object-oriented component design

● Distributed Real-Time Applications
  ○ distributed safety-critical systems with higher scalability & reliability
  ○ prevention of single points of failure by decentralized application design
Context and Requirements

● Asynchronous Middleware Framework
  ○ object-oriented design and loose component coupling support
generation of standard code for implementation
and equivalent model for static analysis and verification
  ○ bounds for message transmission in physical network allows
dependable event communication in application layer

● Integrated Development
  ○ descriptive software development supports
application design, implementation, and verification
  ○ HIJA approach of development
    starting with code: generate additional code and model

● Development Methodology
  ○ structured and controlled process enables
  transparent software development for safety-critical
1) Asynchronous Communication with Event Channels
separation of timely reception and handling, concurrent events
requires plan, real-time requirements for event handling, sim-
plified application development with frames (hard or soft real-
time)

Gateway

Transmitter

EventChannel

Receiver

ActivityManager

Activity

- Period, Deadline
- Startup, Jitter
- Event type
- ...

Description:
- Component
  - Name
    - active Component (with RT thread)
    - Name
    - event-flow and event-handling
    - Attributes

2) Descriptive Development Method

design pattern for application design with standard and generated communication code, component design for loose coupling, publish/subscribe by topic, reuse of tested components
Basic Thoughts and Results

3) Verification of Scheduling Models
generation of analysis models equivalent to API and
generated code, use of WCETA results in verification
models, static analysis of systems with hard real-time
requirements in each node
Summary

Methodology for Integrated Development

- network independent abstraction for whole development process: event channel with communication characteristics and real-time requirements
- code generation of declared components, communication, and nodes
- verification of implementation (software and hardware) for schedulability and real-time requirements

- event channel network – methodology built on asynchronous communication framework!

http://www.eventchannelnetwork.org

asynchronous with real-time
Related Work

- Unified Modeling Language (UML)
  - standardized diagrams and languages for modeling and support of the development process
  - Model-Driven Architecture (MDA) for automated model transformation

- UML Profile for Schedulability, Performance and Time (UML-RT)
  - standard real-time extension for UML

- Domain Specific Language/Modelling (DSL/DSM)
  - development of applications in a distinct domain (e.g., distributed real-time systems)
Methodology Overview

1) **Description**
   in XML file for declaration of event channels with periods, and deadlines

2) **Generation**
   of standard communication code and equivalent model

3) **Extension**
   in generated program frames with application specific logic and model

4) **Analysis**
   for schedulability in each application node
Consequences for RTSJ?

- **Future Directions for RTSJ**
  - asynchronous communication with real-time requirements
  - standard annotations for tool-support and code generation

- **Software Development for Safety-Critical Systems**
  - descriptive development method
  - static schedulability analysis with hard real-time
Thank you! – Any questions?

- HIJA project website: http://www.hija.info
- HIDOORS project website: http://www.hidoors.org
- FZI Karlsruhe: http://www.fzi.de/ajc/