SOFA HI COMPONENT SYSTEM
CURRENT STATUS
Outline

- SOFA HI
- Motivation
- Related Work
- Discussion Topics
- Implementation
- Real-life Use Case
- Future Plan
SOFA HI

- Extension of the SOFA 2 component model
- Profile of SOFA 2 targeted at high-integrity real-time embedded systems
- Effort to bring our knowledge of component systems into real-time environment

http://sofa.ow2.org/sofahi
Why Do We Need SOFA HI?
Motivation

- The SOFA HI idea originated at SciSys
  - Motivated by the European Space Agency initiative
- Effort to speed up the development and lower the costs of high-integrity systems
  - Especially spacecraft on-board software
  - Using the concepts of component systems


Related Work

- **MyCCM-HI**
  - Lightweight CCM component model
  - Based on AADL utilizing Ocarina toolkit

- **OMG Marte**
  - UML profile targeted at real-time systems

- **THINK**
  - Fractal C implementation
  - Missing real-time features

- **ROBOCOP**
Topics for Discussion
Timing Properties

- We need a way to express and verify the components’ timing properties
  - Timing Definition Language
    - Description of timing behavior of a hard real-time applications
    - Provides toolkit to validate the timing properties
    - Supports flat component model and active components
    - [http://www.preetec.org](http://www.preetec.org)
  - Extension of behavior protocols
Computational Model

How to define computational model?

- Formal specification of the component model
- Modern formal specification languages could be used for this purpose
  - Alloy
  - Event-B
- Other formal methods could be also used for this purpose e.g. Petri Nets
How to Implement It?
Implementation

- SOFA HI runtime support should be implemented in C programming language
- Existing SOFA 2 tools and infrastructure should be utilized to the maximum
  - With minimal changes and effort
- Runtime implementation should be independent of used environment
  - Using environment abstraction layer
- How to schedule active components?
FRESCOR

- Advanced scheduling framework for real-time embedded applications
  - Developed also at Czech Technical University
- Employs planning based on resource contracts
  - Allows planning for different resource types

http://www.frescor.org
Current FRESCOR implementation supports different operating systems

- **RTLinux**
  - Open-source version development already discontinued
- **PaRTiKle + XtratuM**
  - Replacement for RTLinux reimplemented from scratch
- **MaRTe OS**
  - Hard real-time operating system implemented in Ada
- **AQuoSA**
  - Soft-realtime Linux scheduler extension
  - Very simple implementation providing adaptive Quality of Service
Logion

- Robot control software created by MART team
  - Students’ team from our faculty
- Software solution for controlling robots used in past and future Eurobot competitions
- Entirely lacks real-time features
- Possible real-life use case application
  - Existing implementation could be modeled and reimplemented using SOFA HI
Logion Hardware

- **Computational Unit**
  - VIA EPIA (IA-32)

- **Communication Bus**
  - I²C
  - RS232
  - USB

- **Hardware Modules**
  - HBmotor boards
  - MCP23016 board
  - SRF02 distance sensor
  - CMPS03 compass
  - Localization beacons
Logion Example
What is the Plan?
Future Plan

1. Runtime environment
   - AQuoSA Linux kernel extensions
2. Stool: SOFA environment abstraction
3. „Coin“ SOFA 2 model extension
   - Property sets and environment description
4. Logion example implementation
   - Built as SOFA HI sample application
5. SOFA HI compiler
   - Implementation of deployment dock
Questions and Discussion