# Component Based Design of Safety-Critical Automotive Systems

**Hans Hansson, Mälardalen University**

- Issues and approaches in a nat'l project
- Questions, ....
- ... answers will be provided later

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

## SAVE – Program overview

- The title says it all:
  **Component Based Design of Safety-Critical Vehicular Systems**
- Nat'l Swedish project (2 M€/3yrs) starting September 2002 (Now!)
- 5 partners from 4 universities:
  - **CSL/MdH** (Ivica Crnkovic)
  - **DAMEK/KTH** (Martin Törngren)
  - **RTSLAB/LiTH** (Simin Nadjm-Tehrani/Jörgen Hansson)
  - **SDL/MdH** (Hans Hansson/Christer Norström/Henrik Thane)
  - **UppAal/UU** (Wang Yi/Paul Pettersson)
- Industrial participation from ABB, Bombardier, CRT, Mecel, Saab, Scania and Volvo (and a few others)

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# SAVE – Time and Size

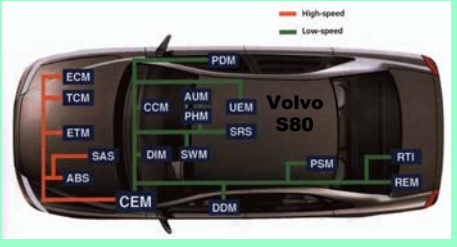
- 3 years
- ~ 10 graduate students funded
- At least 10 additional students with separate funding (from participating industries and other) will contribute
- 2002—2005 (2008; hoping for an extension;-)

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

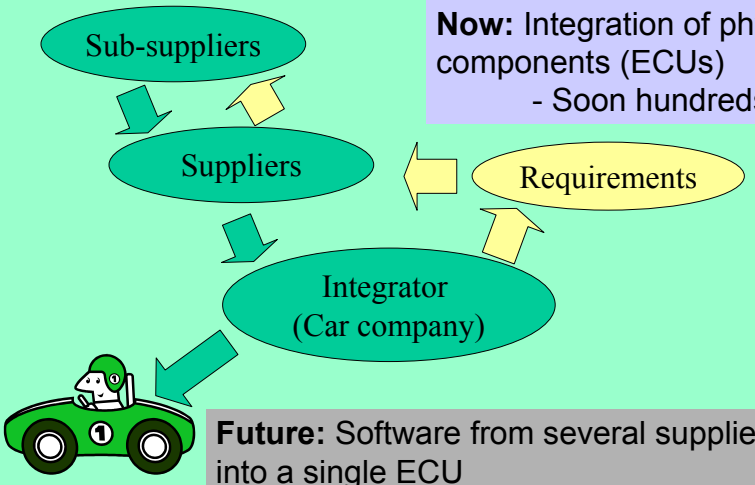
# A Distributed Real-time System

- Networks instead of cables
- Nothing must go wrong ⇒ Robust Design with Predictable Delays

- Changing Requirements
  - Design Methods
  - Production Process
  - Competence
- Software (not just nuts and bolts)




MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE




**Now:** Integration of physical components (ECUs)
- Soon hundreds of ECUs

**Future:** Software from several suppliers integrated into a single ECU

Level of criticality and number of safety-critical functions are increasing (e.g. x-by-wire)

**Reuse** of architecture and components to reduce time to market and reduce costs

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# We've done it before

**(and will do it even better this time;-)**

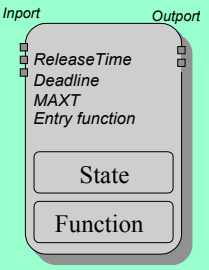
# Application characteristics

- Vehicle control system with high demands on safety, reliability, and timeliness
- Two nodes connected via redundant buses
- 80 tasks
- 150 I/O channels
- Execution times: 10 µs to 1 ms
- Period times: 10 ms to 1 s

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# Design language

- Simple design language
  - a few powerful and clearly defined constructs
- Communication and synchronization separated from programming code
- Key elements
  - Application
  - Modes and mode transitions
  - Transactions
  - Interaction graphs and precedence relationships
  - Tasks

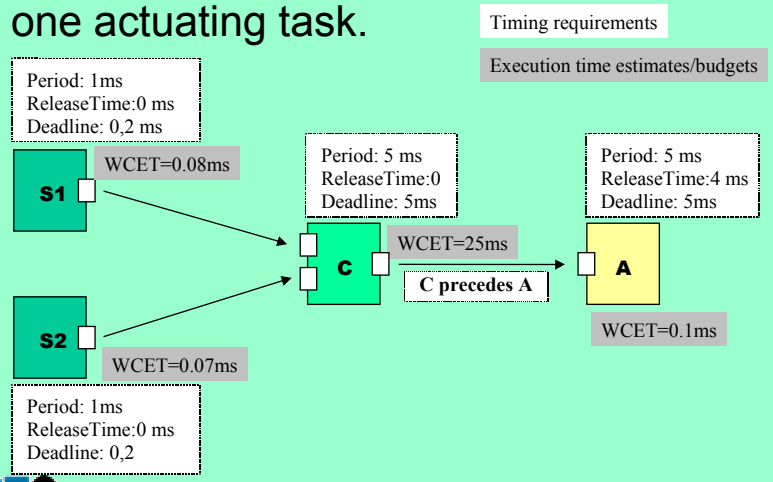



”Component”

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# Example: A controller

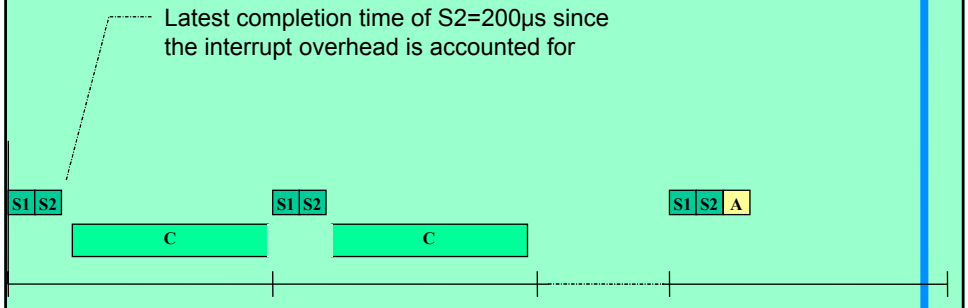
- Two sampling tasks, one controller task and one actuating task.




MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

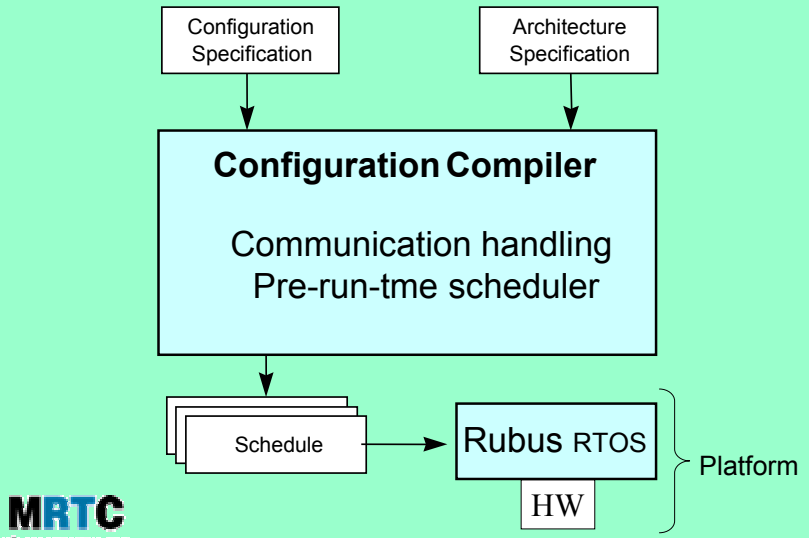
# Feasibility check

- Specification translated to a schedule.
  - Including preemptions
  - Considering clock-ticks and interrupts




MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# Mapping of the design




MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# The implementation

- Tasks can be independently implemented
- Timing requirements fulfilled if execution times less than estimates.
- If not, a redesign is needed.
  - Renegotiation of time budgets.

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

Requirements engineering

Requirements analysis

High level system decomposition

Functional decomposition and structuring

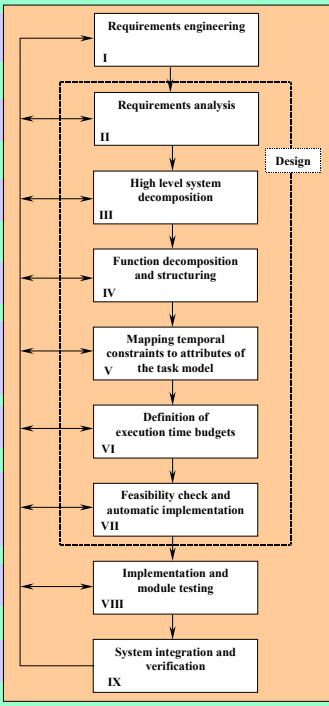
Mapping temporal constraints to attributes of the task model

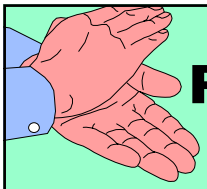
Definition of execution time budgets

Feasibility check and automatic implementation

Implementation and module testing

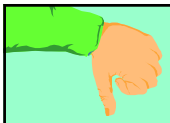
System integration and verification



# Positive results

- Precise design
  - Concurrent engineering:
    - Reduced Time for implementation and testing
- Early detection of design errors.
  - Execution time estimates very useful.
- Easy to add functionality
- Easy to introduce new personnel



# Limitations

- Restricted model (with RT focus)
- Safety and reliability not handled
- No modelling/analysis of functional correctness
- Proprietary (essentially)
- ...

- The ultimate solution must be standardized!

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

Next step.....

# SAVE

- Time to market rules!
- Faster; Better; Cheaper; (Safer?)
- Conflicting requirements! (safety vs. cost ...)
- Reuse is the key!
- Many aspects and several stake holders
- ”Lego-paradigm”
  - ”Horisontally” (functions and aspects)
  - ”Vertically” (requirements, models, impl., platforms)

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

”The goal of SAVE is to contribute towards an engineering discipline for systematic development of component-based software for safety-critical embedded systems”

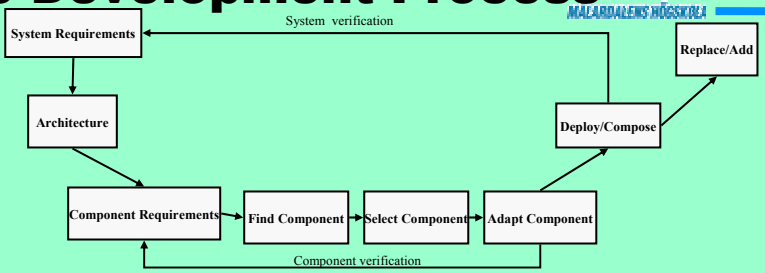
MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# What is a component?

- Definition (Szyperski)
  - A unit of composition with contractually specified interfaces and explicit context dependencies only.
  - Can be deployed independently and is subject to composition by third party.

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

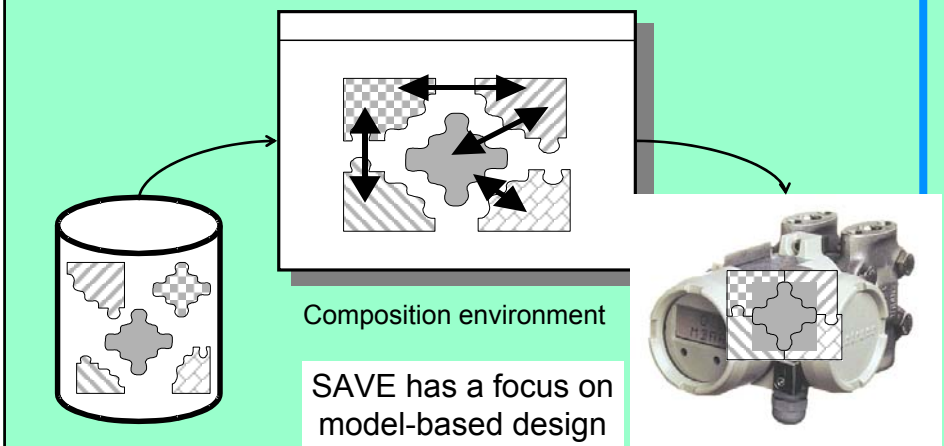
# The Development Process




- System req. Capture (including **real-time, safety and reliability**)
- Architecture Design
- Decomposition of system req. into component req.
- Component modeling and design (specification, selection, adaptation and composition)
- System configuration and verification
  - System configuration
  - Component composition/decomposition
  - System verification

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# Framework




SAVE has a focus on model-based design

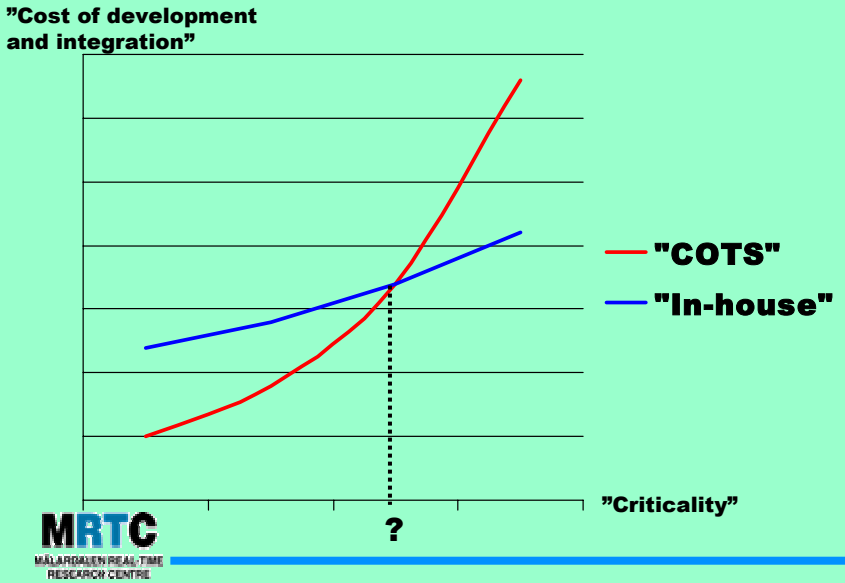
but includes implementation and platform

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

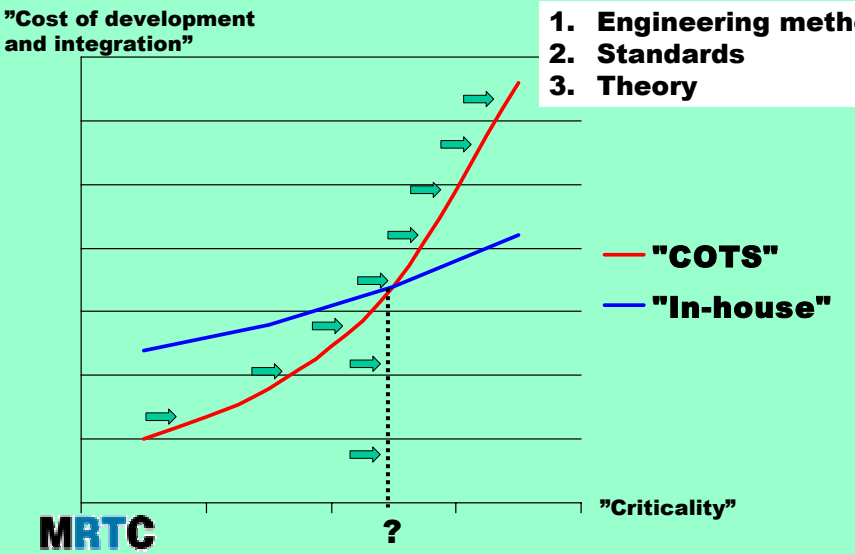
# SAVE application characteristics

- Hard real-time **static** configuration
- Additional, less critical, less static tasks
  - But SAVE focus is not on these
- Resource constrained (per unit cost-critical)
- Reuse of architectures and ”components”
- Integrator – supplier relation

# Research question?



# Research challenge!




MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# Component-based approach for embedded systems?

- Can we use the established component models and technologies for
  - Real-time systems?
  - Embedded systems?
- Can component-based models be directly applied on RT and embedded systems?
  If not which parts can be taken, which are missing?

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# Run-time vs. design time composition

- Run-time composition
  - Component model,
  - Run-time environment,
  - Dynamic communication,
- Design-time composition
  - Capable of generating monolithic firmware from component-based design,
  - Optimization

Component technology

More feasible for embedded systems

”Static configuration”

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# Component transparency

- Black-box reuse
  - From component’s user point of view,
- White-box reuse
  - From composition environment point of view
- Gray-box reuse (composition environment)
  - If clear conventions for knowledge about implementation are introduced

Component technology

More feasible for embedded systems

Models of implementations

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# Portability, Platform independence

- Binary independence
- Source level portability suffices,
  - Design-time composition,
  - Run-time environment restrictions

Component technology

More feasible for embedded systems

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# SAVE – The Challenge

- Conflicting requirements
  - E.g. Safety vs. Cost and time-to-market
- Combination of architectural and component-based design with analysis and verification
- ”Model-based design”
- Safety and Real-Time

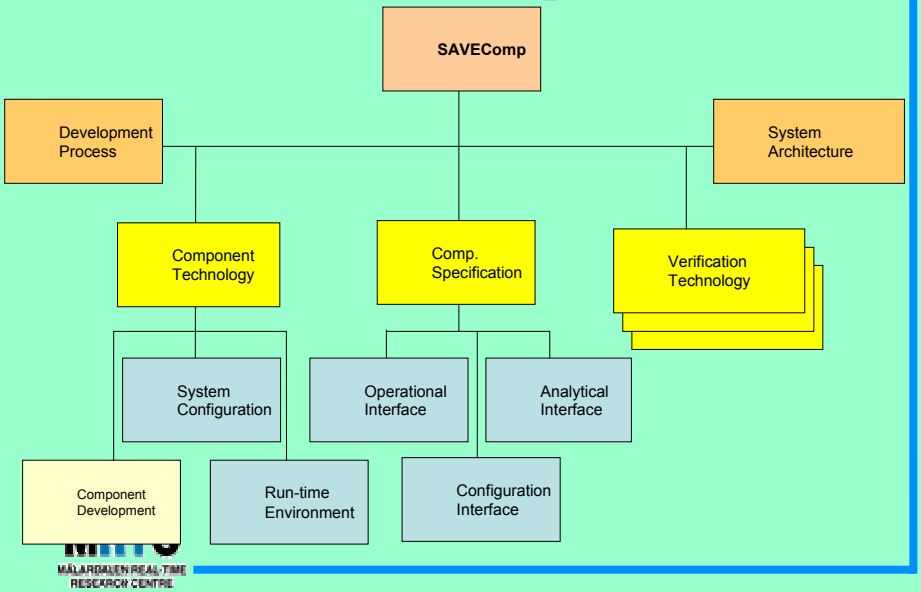
MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# SAVE – The solution

- Development of a holistic framework:
  - Methodology and process
  - Component specification and composition
  - Techniques for analysis and verification
  - Run-time and configuration support

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# SAVEComp



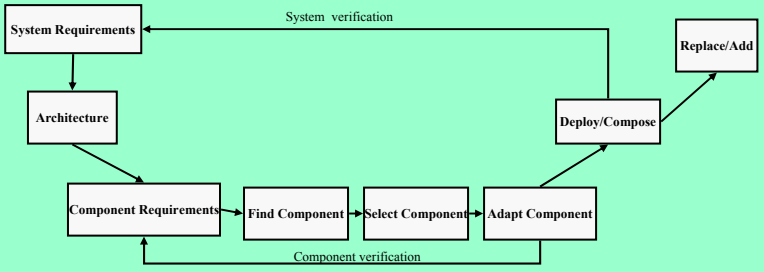
# What will we actually do?

- Investigate the feasability of CBD for safety-critical RTS by demonstrating solutions to critical problems
- Starting out by
  - Identifying issues and concepts
  - Close co-operation with industry

  →”minimal” set of aspects to consider

MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# Design process




Players and their roles:
- Integrators
- Suppliers
- Component providers
- Certifiers and verifiers
- ...

Activities and their relations:
- Requirements
- Architecture design
- Modeling and implementation
- Composition and configuration
- Verification and validation
- ...

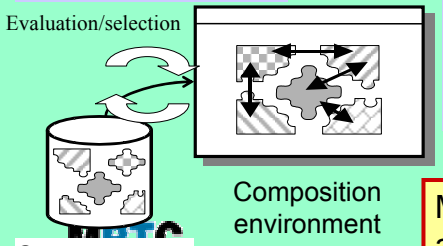
MRTC
MÄLARDALEN REAL-TIME RESEARCH CENTRE

# Component specification and composition

- Identify relevant reqs/aspects
  - Including timing and safety
- Select/define appropriate notation for reqs and properties of components and systems
- Composition

Abstract models of concrete components

- Method for component selection
  - Matching reqs with props



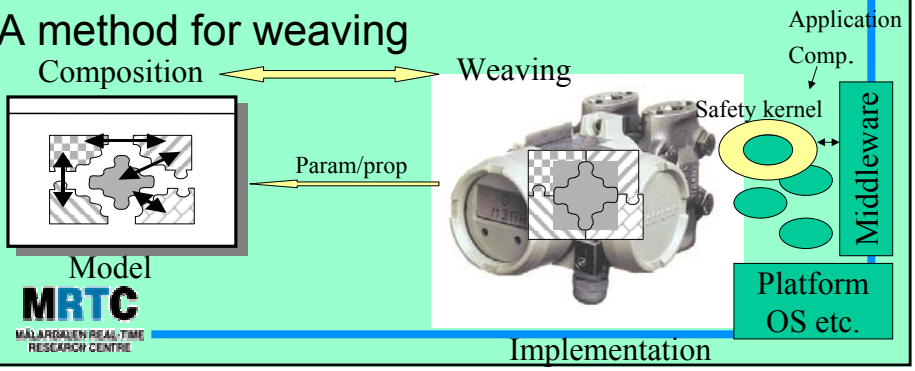

Model-based component selection and architecture evaluation

# Techniques for analysis and verification

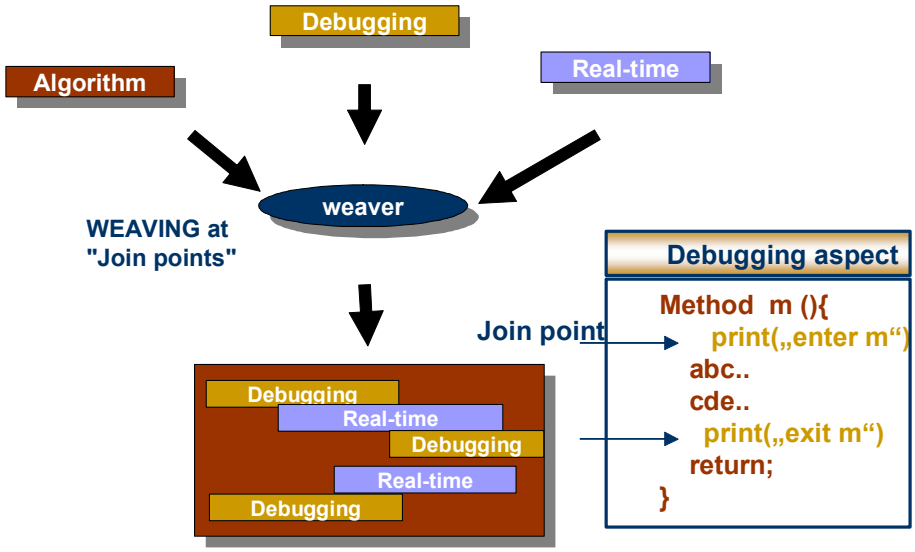
1. Integration of reliability and functional analyses (FTA, FMEA) (Analysis of compondents in presence of failures)
2. Automatic testing for checking safety and real-time (generation of test cases with high coverage)
3. Timing/reliability trade-off analysis (schedulability, reliability and sensitivity analyses)
4. Model based systems analysis (complete system simulation to analyze robustness and failures)

## Run-time and configuration support

- Extracting parameters/props from comp.
  - Those needed by our analysis
- Monitoring and policing
  - Safety kernels and Monitors
- A method for weaving

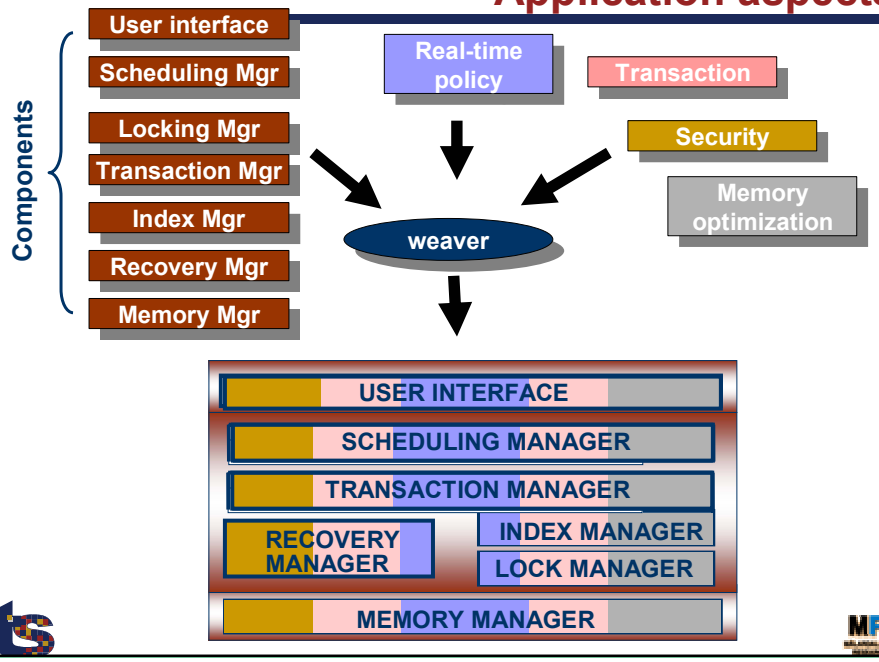



## AOSD in a nutshell



RTDB example

## Application aspects




## Real-time scheduling aspect

```
aspect RTscheduling{

  where
    pointcut applyPolicy(transaction * trans)=
      call("void putInQueue(trans)")&&args(trans);

    pointcut setPriority(DbTransaction * DbTrans)=
     call("void registerTrans(DbTrans)")&&args(DbTrans);

  what
    advice applyPolicy(trans):
     void around(transaction * trans){
       /*module that performs real-time scheduling
                              according to the policy chosen*/
     }

    advice setPriority(DbTransaction * DbTrans):
       void after(DbTransaction * DbTrans){
        /* set priority of the transaction */
      }
    }
```

# Demonstrator(s)

- In co-operation with our industrial partners

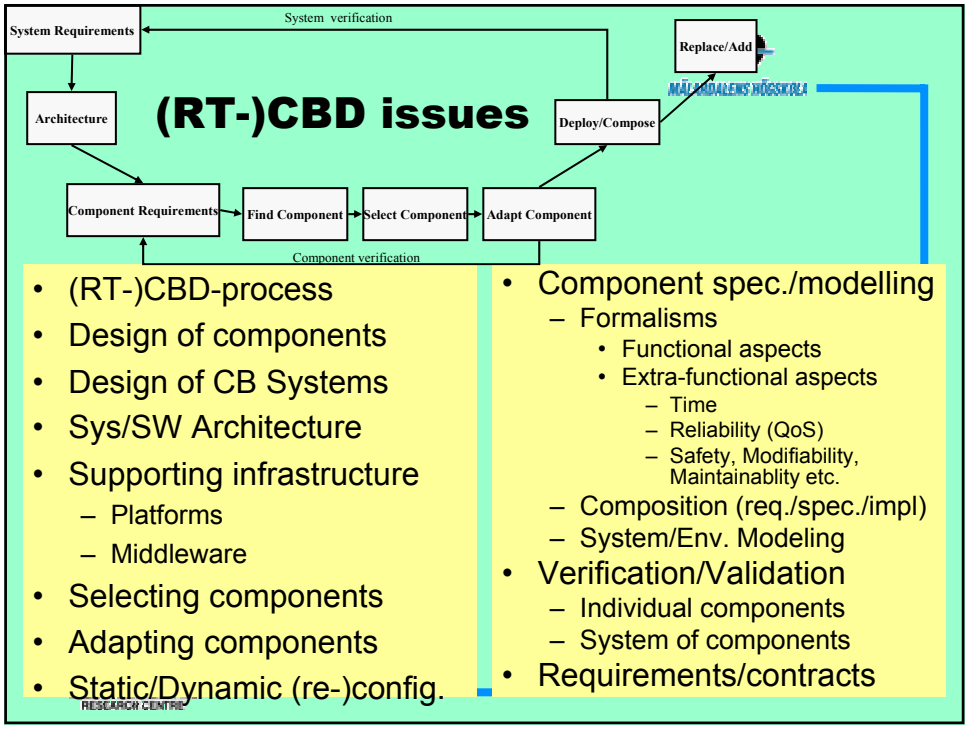
Reuse from previous projects and presentations;-)

# More on SAVE...

- See

  **www.artes.uu.se/++/SAVE/**

- ARTIST/Comp is an important forum for SAVE


System Requirements
System verification
Replace/Add
Architecture
(RT-)CBD issues
Deploy/Compose
Component Requirements
Find Component
Select Component
Adapt Component
Component verification
• (RT-)CBD-process
• Design of components
• Design of CB Systems
• Sys/SW Architecture
• Supporting infrastructure
– Platforms
– Middleware
• Selecting components
• Adapting components
• Static/Dynamic (re-)config.
• Component spec./modelling
– Formalisms
• Functional aspects
• Extra-functional aspects
– Time
– Reliability (QoS)
– Safety, Modifiability, Maintainablity etc.
– Composition (req./spec./impl)
– System/Env. Modeling
• Verification/Validation
– Individual components
– System of components
• Requirements/contracts