Safety-Critical Real-Time Systems

ARTES PhD course

Lecture 7: Component-based development

Simin Nadjm-Tehrani ©

Real-time Systems Laboratory

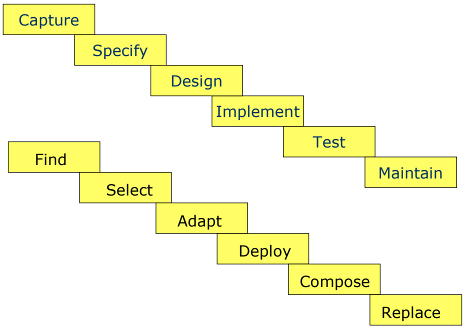
simin@ida.liu.se

A new trend?

By using COTS or Nondevelopmental Item (NDI) hardware, software, and formally verified microprocessors and real-time kernels ..., one can leverage the industry's large investment ..., essentially having others perform fault removal for free. Unfortunately, the tradition for building safety-critical systems ... is just the reverse...

[Lala and Harper 1994]

Building with Components



COTS and safety

- What are the needed changes in the development process to enable use of COTS in safety-critical systems?
- Will it be better then?
- Well, perhaps...

How to define Component?

- Study at Saab Aerospace shows that it could be any of the following:
  - Complete self contained system (e.g. Fuel system)
  - Device that needs completion with application software (e.g. Hardware & interface in sensor or actuator)
  - Customer-dependent unit to be embedded in a delivered product (e.g. a helmet)

How to define Component?

- cont'd:
  - Ground support system provided by customer (e.g. Pre-flight checking equipment)
  - Pure COTS software (e.g. Ada run-time kernel)
  - Outsourced software or FPGA
  - In-house developed software unit
  - Non-digital hardware (e.g. Pumps)

In SE literature

- COTS typically refer to bought-in components with no access to source code
- Mostly on mechanics on deploying and composing components
- Recent debate on how to adapt the developement process and metrics

In systems safety

- The first five types of component:
  - Change of supplier means a ”new” components. All assurance steps have to be repeated.
- For pure COTS software, one has to ensure that additional functions present in the product are not detrimental to safety!

Outsourced units

- Detailed precise and complete documentation of requirements at the procurement stage
- Review of vendor capabilities, procedures, track record

Where are the pitfalls?

- According to aerospace experts:
  - Errors in interface between components (due to inaccurate specifications)
  - Forgetting about dependencies when upgrading some component

COTS reliability?

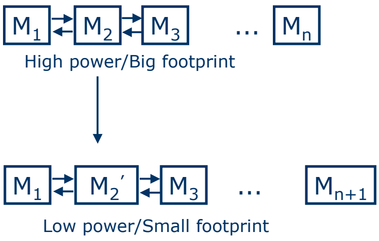
- Only recently studied [Hamlet et al]
- Has the same bearing on safety as classic reliability analysis
- Only is less meaningful without relating safety to components first...

COTS selection method

- Fan Ye and Tim Kelly [ICCBS04] provide a nice selection process:
  - Focusing on architecture proposal
  - Relating system hazard analysis and component FMEA to COTS aquisition contract
- Note iterations needed on COTS via architecture selection

Changing non-functional reqs

- Formal analysis of reconfigurable modules



M_1 M_2 M_3 … M_n

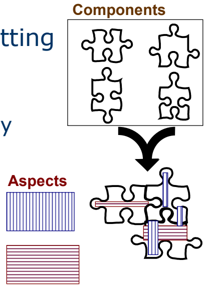
High power/Big footprint

M_1 M_2' M_3 … M_{n+1}

Low power/Small footprint

Real-time requirements

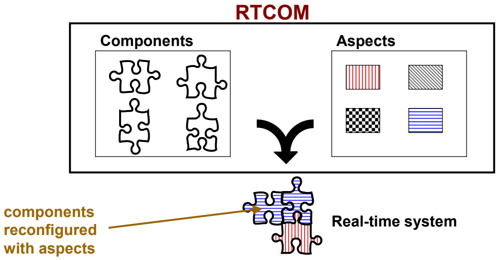
- Properties as a cross-cutting *Aspect*
  - Low development costs
  - High degree of tailorability
  - Short time to market
- Efficient handling of
  - memory optimization
  - power consumption
  - temporal attributes
  - synchronization
  - error handling



Components

Aspects

Reconfigurable components



RTCOM

Components

Aspects

components reconfigured with aspects

Real-time system

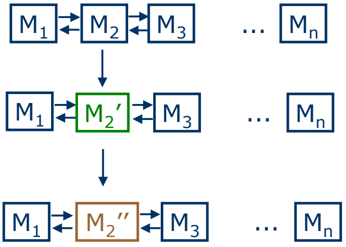
Recent work at LiU

[Tesanovic, Nadjm-Tehrani, Hansson 04]

- Formalises components with verification interfaces as augmented timed-automata (TA)
- Formalises aspects as augmented TA
- If a component C satisfies timing property P, formally shows that weaving an aspect A into it preserves P (using only C:s verification interface)

Evolutionary design

- Incremental formal analysis



M_1 M_2 M_3 … M_n

M_1 M_2' M_3 … M_n

M_1 M_2'' M_3 … M_n

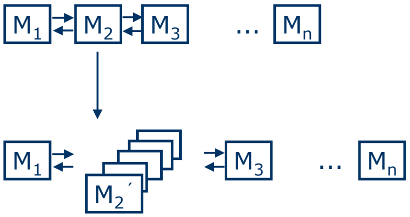
Current work at LiU

Formal tools to support that activity:

- FMEA-like: given a component, derive a safety-related (formal) interfaces that explains how the component behaves given faulty inputs
- FTA-like: Given a property P that is traced to this component from system hazard analysis
  - Show that component satisfies P
  - Show combinations of faults that lead to violation of the property

Distributed vs. Central?

- Analysis of fault tolerance in distributed flight control system



Back to mechanics…

- SE has made advances on easier deployment of component-based systems via support in middleware
- An important aspect that components middleware (CORBA, EJB, …) can improve upon is
  - Support for fault tolerance and fault containment
- Formal verification of MW support needed in safety-critical systems

Common mode failures

- Difficult to remedy by one method
  [Lala and Harper 1994]
- Need all available arsenal:
  1. Fault avoidance by formal methods
  2. Fault removal by testing
  3. Fault tolerance at run-time
- Formal analysis needed for 1 and 3.

Summary

- Component-based systems are desirable due to:
  - Higher (component) reliability
  - Shorter time to market, lower costs
- Unclear if higher component reliability converges to higher system reliability
- With no specific safety-related component techniques
  - Safety may be compromised
  - Costs may become higher!