SAVE – Component Based Design of Safety Critical Vehicular Systems

Brief version of the program proposal submitted to SSF "Ramanslag för forskning inom informationsteknik 2001"
Full proposal available at http://www.docs.uu.se/artes/++/SAVE/

Coordinator: Hans Hansson, Mälardalen Real-Time Research Centre, Dept. of Computer Engineering, MdH

Summary

The goal of *SAVE is* to establish an engineering discipline for systematic development of component-based software for safety critical embedded systems. This will be vital to the Swedish industry, and paves the way for establishing an industry for safety-critical and other components.

The main innovation of SAVE is the interdisciplinary combination of architectural and component based design with analysis and verification, in the specific context of safety and real-time. While several aspects and issues have been studied in isolation a merging of the various competencies of the applicants will be required to provide an appropriate holistic approach. The focus on a single application area (vehicular systems) will reduce the overall project complexity to a manageable level.

The main challenges in component-based development of safety critical applications are to handle the multitude of conflicting requirements, including safety vs. cost and time-to-market. Reuse of earlier work and integration of external components and sub-systems are essential in reducing cost and time-to-market, and the use of proper design methods and architectures is instrumental to accomplish this. Structuring is equally important, together with verification, to ensure safety.

SAVE will address the above by developing a general framework for component-based development of safety-critical vehicular systems, including

- **Methodology and process** for development of systems with components
- **Component specification and composition,** providing a component model which includes the basic characteristics of safety-critical components and infrastructure supporting component collaboration.
- **Techniques for analysis and verification** of functional correctness, real-time behaviour, safety, and reliability.
- **Run-time and configuration support**, including support for assembling components into systems, run-time monitoring, and evaluation of alternative configurations.

Introduction

The goal of this project is to develop techniques for component based development of safety-critical real-time systems.

To exploit the competence of the involved researchers, and due to the importance of the specific application domain, we will focus on vehicular type of systems. For such type of systems, e.g. cars, train systems and aircraft, the component based approach has a relatively long tradition, as these systems are typically built from physical components that are either developed in-house or provided by external suppliers. Today, the physical components also include several computer nodes (or Electronic Control Units, ECUs) equipped with software that implements vehicle functions. The next major step in designing these systems is to go from the current situation with "one node – one supplier" to a situation with "one node – several suppliers", i.e. there will be several software components of different origins executing on a typical node. This may seem like a small step, but the implications for the design process and division of responsibilities is quite dramatic. Furthermore, satisfactory handling of safety-critical functions, such as emerging brake and steer by wire systems, will require the integration of methods for establishing functional and temporal correctness for each component, as well as system-wide attributes such as safety and reliability. This should be done while minimising resource demands and maximising reuse to keep costs and time-to-market at competitive levels.

Complexity in current and future vehicular systems has many facets, including the sheer amount of functionality (number and multiplicity of components), complex relations between requirements, functions and components, and the fact that the applications and functions have widely varying characteristics. Essential functional requirements include different degrees of timing criticality and safety criticality. Different characteristics include discrete time control (part of sampled data functions), event-triggered control, mode logic, safety logic, and discrete state control. New functions are typically distributed over the in-vehicle distributed computer systems. Models and modelling frameworks that describe and relate components, functions and requirements are imperative for complexity management, as are the development of suitable architectures, architectural styles, and system and software platforms.

For development of this type of complex systems we need to work in several scientific areas: (1) component-based software engineering and its adaptation towards systems engineering; (2) support for run-time and real-time environments using components; (3) analysis of system properties given specifications of components and the required models, tools and technologies; (4) integration with safety-critical systems development processes.

Component based design addresses the development of systems as an assembly of parts (components), the development of parts as reusable entities, and the maintenance and upgrading of systems by customising and replacing such parts.

Most vehicular systems vendors are players on highly competitive markets, where *time-to-market* is the essential competitive factor. At the same time, *product quality and dependability* must meet the demanding requirements imposed by *regulations* and the *standards* of the respective market. The key to efficient development of trustworthy systems is a combination of reuse and support for architectural analysis. Component-based Development (CBD) is based on assembling components already developed and prepared for integration. However, current experience (e.g. from Microsoft, Siemens and ABB [Spenc99, Mrva97, CL00]) indicate that developing a reusable component requires several times the effort of developing a unit for a single use. Also, the demand for reusability gives a tendency towards more general and less efficient components [Szy98, CL00]. In safety-critical systems the separate lifecycles and requirements of components and applications introduces a risk that a component may include concealed characteristics which, especially at component or application update, may cause system failure, such as in the well-known case of the Arianne rocket. To manage these risks, we need a systematic approach to component-based development at the process and technology levels. This includes a systematic treatment of system requirements (both functional and non-functional/extra-functional), the satisfaction of these requirements by adequate choices at the architectural level, and well-defined interfaces and analysis techniques which enable the construction of safety cases and arguments.

Much of research in CBD has so far been geared towards the contribution of components towards functional requirements in a system. In this project we expand this horizon by considering non-functional attributes such a s timeliness, safety and reliability. The timeliness attribute requires treating run-time environments as components, and to extend real-time systems analysis techniques to operating systems, middle-ware, and communication architectures specialised for safety-critical and vehicular applications. A coherent treatment of safety in CBD requires a change in the traditional process for safety analysis. The traditional techniques used for assurance of safety and reliability have been used for over 40 years. They often entail analysis of systems via separate activities by different teams of people: one at the design and development stage with functionality in mind, and another in parallel, dealing with hazard analysis, risk analysis, fault-tree analysis (FTA), Failure modes and effects analysis (FMEA), and extended testing. The models of the system used for these purposes are often derived separately and may lack coherence when changes are introduced in the development and life cycles. We believe that a process in which CBD is integrated with analysis for safety and reliability is the key to efficient development of trustworthy systems. In the research we pursue, the integration aspect will have a major influence in the choice of component models, technologies and analysis techniques.

This will provide a strong basis for creating a Component Based Safety Critical Systems Engineering (CBSCS) discipline.

Objectives

The main objective of SAVE is to develop SAVEComp – a component-based development (CBD) framework for safety-critical embedded real-time systems (RTS). The primary focus is on designing systems with components, based on component and system models.

The purpose of developing SAVEComp is to reduce the system development time and to increase the product quality. This will be achieved by a component-based approach, similar to the ones proposed for other business domains. That is, the ambition is to develop a method and infrastructure for CBD for safety-critical embedded RTS, corresponding to existing general component technologies, such as COM and JavaBeans. SAVEComp will include the following parts:

- **Methodology and process**, including a system development process using components, as well as identification (definition, classification) of relevant system architectures.
- **Component specification and composition,** providing a component model which includes the basic characteristics of safety-critical components and infrastructure supporting component collaboration, together with a semantic basis for component modelling that will make it possible to specify the essential aspects of

safety-critical RT components and systems, and to make abstractions, decompositions and compositions of components.

- **Techniques for analysis and verification** for establishing functional correctness, real-time behaviour, safety, and reliability. Specific techniques for analysis and verification of components and their compositions will be developed.
- **Run-time and configuration support**, including support for assembling components into systems, run-time monitoring, and evaluation of alternative configurations.

SAVEComp

Due to the strict requirements on the considered systems, SAVEComp has a dual emphasis on both the component technology needed to build applications from systems of components, and on the analysis/verification technologies needed to establish that the requirements are fulfilled. Modelling of components and aggregates of components provides the semantic basis for linking the component and analysis/verification technologies. Also, the scope and applicability of SAVEComp is defined by the associated development process and system architectures. The diagram in Figure 1 shows the different parts of SAVEComp.

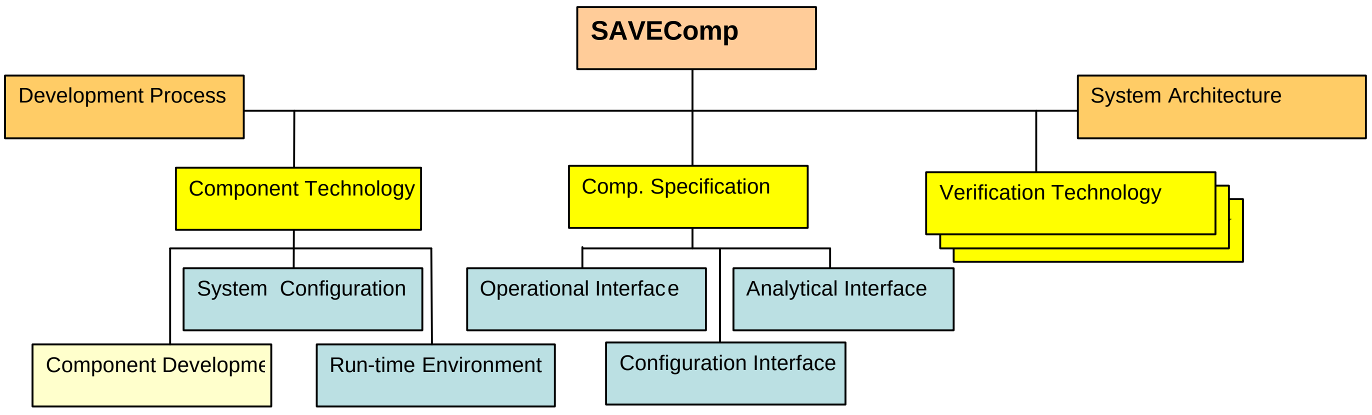

Figure 1 The SAVEComp ingredients (inspired by [HMSW01]).

The system architecture provides guidelines for component identification, integration, configuration and modification, by defining the context of a component with respect to system structures and the underlying implementation. The architecture therefore forms the basis for defining component interfaces.

The SaveComp framework includes component specification and component technology. The component specification identifies all parts of a component needed for component development and component-based system development. The most important feature of a component is separation of interface from implementation. The functional features of a component are expressed by its operational interface, including specifications of how to interface the component with other components and the run-time environment. Since a component typically will provide some flexibility in how it is used, and since SAVEComp is focused on statically configured systems, we provide a separate configuration interface covering aspects related to the concrete instantiation of a component, i.e., its configuration parameters. There is additionally a need to determine non-functional properties of component assemblies. To be able to formally specify all relevant attributes of components and predict the behavior of component assemblies we introduced an analytical interface. The analytical interface specifies the component properties required for analyses. This may cover a wide range of both functional and non-functional aspects, including an operational component model, execution times, failure modes and component reliability.

Component technology identifies the methods for developing components and systems by using components. We distinguish between development of individual components and system development. As the focus of SAVE is embedded systems with fixed run-time configuration, we identify two different environments: A configuration environment in which we integrate the components, build the system and predict the system behavior, and a run-time environment that enables components collaboration and system execution.

Workpackages

Considering the outlined challenges and available resources, it is not realistic to arrive at satisfactory solutions to all issues within this 3 year proposal. Substantial progress will however be made by focusing on central issues and by providing concrete solutions to specific issues, as outlined by the below work-package descriptions.

WP1 – Methodology and process

WP1 includes activities that will determine the starting points for other WPs. It will define the overall requirements of the system and the common frames for further research in other WPs. The first results from WP1 will be taken and refined by other WPs. WP1 will continue with refinement and specification of the development process using results from other WPs. The main tasks for WP1 are:

- State of the art - Analyze the current trends, identify current and future requirements and open questions.
- Requirements specification - Overall specification of input requirements to other WPs
- Development process specification – Component-based development process for RT embedded systems

The state of the art analysis will collect results from two different and complementary domains: a) existing real-time safety critical systems and b) existing component models. Application studies and analyses of the achieved results in the respective domains will be performed. In a) vehicular systems and similar systems (avionics, space, industry automation) will be studied. The study of vehicular systems will give a present state and trends in this domain, while analysis of other real-time safety-critical domains will indicate results and experiences from these domains that can be applied to vehicular systems. The study of component models will consider “general-purpose” component models as well as the component models developed for specific domains. The main characteristics and underlying technologies will be analyzed and the work will result in a survey of component models characteristics and their relevance for embedded component-based systems, and with an analysis of problems with component frameworks for implementation of safety-critical real-time systems.

Plan and Deliverables:

1. State of the art (Year 1)
   a. Design of Safety Critical Vehicular Systems
   b. Component-based Software Engineering for real-time safety-critical systems
   c. Component-based development processes with product-line approach
2. Requirements Specification (Year 1)
   a. System architecture, software component-based architecture
   b. Component Framework, configuration and run-time environments
   c. Component specification, formal analyses and verification of the component model
3. Overall specification of Component Framework (Year 1-2)
4. Specification: SAVEComp development process, SAVEComp lifecycle (Year 2-3)

Deliverables will be produced in form of technical reports and research papers, workshops and a graduate course on “CBD for Safety-Critical Systems”..

WP2 – Component specification and composition

The objective of this WP is to develop a general framework for constructing abstract models of components, and for combining components based on their models to implement system specifications from architecture design (according to the development process defined in WP1) or to implement new components for new applications. A model of a component is an abstract and precise description, which should capture the essential aspects required for the composition of the component with other components. A model of a component should also provide necessary information for deployment for the given underlying platforms. Compositions of components based on models should support compositional analysis in the sense that the analysis of compositions can be carried out fully on the models of the components (described in WP3) (this may however require environment models). The compositions should also be implementable for given underlying platforms (given in WP4).

The current techniques for component specification (modeling) are mainly based on syntactical specification. We shall specify (model) components in terms of the following classification of aspects or views of a component:

1. Syntactical views, such as interfaces for communication (SAVEComp operational interface),
2. Functional views, such as input and output functions or operational models based on state machines (SAVEComp operational interface),
3. Adaptability views, such as parametrizations of component interfaces and behaviours (SAVEComp configuration interface), and
4. Analytical views, (requirements and properties) such as deadlines, worst case execution times, safety and reliability parameters (SAVEComp analytical interface).

Plan and Deliverables:

1. Identify and study the essential aspects of a component to capture in a component model, including aspects related to function, real-time, safety and reliability. (Year 1)
2. Develop a modeling language with well defined semantics for component modeling (or adapt/extend an existing modeling language such as UML) and compositions. (Year 1-2)
3. Develop a method with tool support for component selection based on models. Given a system requirement from the development process (WP1), select the right components and the right composition such that the composition of the component satisfies (implements) the requirement. (Year 2-3)

WP3 – Techniques for analysis and verification

The objective of this WP is to advance the technology and knowledge for analysis and verification of safety-critical systems. The work is strongly related to WP2, where the analytical interface of components provides the extra information required for analysis and verification. The work is also related to WP4, where adequate choices of system architectures can facilitate the verification task, where enforcement of assumptions made in analysis is required, and where specific properties (parameters) required for analysis will be extracted from implementations.

In addition to functional correctness of a *component*, analysis techniques for *subsystems* in this work-package have the goal of deriving *system* properties. Subsystems can be software components, run-time system components, or (digital) hardware components. Subsystem properties that we consider include timeliness and resource optimisation characteristics. System properties will be the overall safety/reliability requirements, partly resembling those tested via traditional FTA and FMEA analysis. In contrast with the traditional separation of safety analysis, we propose methods to deal with these problems in the context of models within the normal development process.

Analysis techniques will span techniques for formal verification, simulation and testing (including run-time testing), each where they suit best. The subtasks of WP3 represent four different techniques that together will contribute to the development of a coherent methodology. This will for SAVEComp be realized by an environment in which concrete configurations can be built and analyzed using available analysis and verification techniques. Since pertinent aspects such as safety, reliability and schedulability require concrete configuration information the actual integration of this is an activity within WP4.

Plan and Deliverables:

1. **Integration of reliability and functional analyses**. In vehicular systems where "hardware" includes mechanic, hydraulic or electric components, traditional reliability analysis (FTA, FMEA) needs to be replaced by modern techniques whereby software and digital HW components' behaviour in presence of other failures is analysed. Our work will integrate component-based functional analyses and system reliability analyses building on earlier work [ÅNS99].
2. Techniques for **checking safety and real-time requirements by automatic testing** of component based systems. The problem is divided in two parts: The first, addressed in WP4, is how to generate software monitors for checking formally specified safety and real-time requirements. The second part, addressed here, is how to obtain high coverage of a set of test-runs sufficiently small to be applicable in practice. To reduce the size of the test sets, we plan to adopt techniques used in symbolic model checking of real-time systems [Dill99], and generate tests with time-points that obtain high coverage on the symbolic system representation.
3. Extending and adapting our framework for **timing/reliability trade-off analysis** [HNNP02] to fit SAVEComp and the considered platforms, as well as adding analysis of robustness (sensitivity analysis).
4. Techniques for **model based systems analysis** where components are simulated together with platform and environment models to analyze robustness and failures, subject to pertinent (and as relevant combined) failures including transient and permanent hardware failures as well as systematic failures. This extends our current work [RET01, ET01].

WP4 – Run-time and configuration support

The objective of this WP is to link the systems modeling in WP2 to concrete implementations and providing support for architectural patterns facilitating the provision of safety and reliability.

Modeling and implementation will be linked both via a method for combining component implementations (weaving) that should be consistent with the compositions performed at the modeling level, and by a method for extracting parameters from implementations, to be used in the analytical models from WP2. The latter will be combined with analysis and verification from WP3. Provision of safety and reliability will in this WP be achieved by introducing special wrappers – safety-kernels – for component/task monitoring. Since the wrappers may contain both alternative and redundant components as well as handling of these, a substantial increase in safety and reliability is possible. A special effort (originating from SSF's suggestion to include activities from the BEST program) will be to generate HW monitors which can police the system non-intrusively. This work will be performed in co-operation with a project at the Computer Architecture Lab at MdH, led by Prof. Lars Asplund and with additional partners from Saab Avionics and MIT (Dept. of Aeronautics and Astronautics).

Plan and Deliverables:

1. Identification and characterization (of relevant aspects and properties for determining safety, reliability and real-time) of 1-2 concrete target system platforms, including HW (CPUs, network, etc.), OS-kernel and middleware (from WP5). The concrete result will be **procedures for extracting parameters required by analytical models** from concrete system configurations. The work will be focused on extracting parameters required by the analyses considered in WP3. (Year 1-2)
2. **Techniques for monitoring and policing**, including a Safety-kernel methodology integrating generated monitoring specifications (from WP2) into a monitoring and policing framework, as well as development of monitors for extracting analytical parameters and validation of environment assumptions. (Year 2-3)
3. **A method for weaving**, compatible with the methods for specification and composition developed in WP2, and combined with a method for testing the conformance between modelled and implemented composition. (Year 2-3)
4. A **method and tool for explorative analysis of configurations**. Configurations are defined based on a concrete architecture and a set of available components. Using the techniques from 1, concrete analytical models will be derived. Analysis based on techniques from WP3 is then used to determine system properties (metrics) which form the basis for comparing configurations. The outcome of this analysis provides valuable information for selecting configuration, or even general architecture. (Year 3)

WP5 – Case-studies

Realistic examples are required to evaluate the developed techniques and to gain insights in their applicability and interoperability. Two types of case-studies will be considered: (1) simple but realistic "toy-examples" to demonstrate specific techniques and for early exploration of interoperability, and (2) more substantial realistic case-studies for evaluation of the applicability of specific techniques, as well as for evaluation of the SAVEComp development process and interoperability of specific techniques. The associated industries will be instrumental in defining both types of case-studies.

Plan and Deliverables:

1. Definition of **3-4 smaller examples** to be used by other WPs [Related to initial study in WP1] (Year 1)
2. Definition of **1-2 larger case-studies** [Candidates include: Automotive example based e.g. on input from the Eureka project EAST-EA, Scania or Volvo, Rail system from Bombardier, Avionic system from Saab, and Industrial Robot systems from ABB.] (Year 2)
3. **Demonstration** (at least partial) **of interworking** of architectures, component modelling, verification and configuration techniques in a common case-study (one of the cases from 2). (Year 3)

Industrial participation

Support letters are provided by the following companies: ABB ATP/Robotics, Bombardier, CR&T, Saab, Scania, Volvo Car and Volvo TD. We expect active involvement by these and other companies in terms of discussions, providing case-studies, as well as actually participating in some of the research activities in SAVE, via industrial graduate students or in other forms. The industrial co-operations will in particular include important complementary efforts regarding development of architectures.

Partner overview and profiles

- DAMEK/KTH (Martin Törngren) contributes with strong experience and research in architectural design for embedded control systems. The Mechatronics lab also contributes with knowledge in vehicular systems, in particular with respect to safety critical motion control systems such as braking, vehicle dynamics and x-by-wire systems, and design of the corresponding distributed control applications.
- RTSLAB/Linköping University contributes with expertise in modelling and formal verification of embedded systems, in particular safety-critical systems (Simin Nadjm-Tehrani), and analysis of middleware for optimal resource management, including real-time properties (Jörgen Hansson). RTSLAB has a long record of cooperation with the aerospace industry and the vehicular industry.
- SDL/Mälardalen Univ. (Hans Hansson/Christer Norström/Henrik Thane) contributes with knowledge in modelling and analysis of safety-critical real-time systems, specifically scheduling, timing and reliability aspects. The group additionally contributes with knowledge in engineering embedded systems, including

system software, software engineering and techniques for mixing hard and soft real-time. The track record in developing design tools, real-time kernels and communication systems for the automotive industry will be instrumental for tool-development and case-studies.

- CSL/Mälardalen Univ (Ivica Crnkovic) contributes with strong industrial experience in process automation and knowledge in component-based software engineering, component technologies, development processes, component configuration management and in general in software engineering. The experience related to component-based software engineering will be applied to real-time embedded systems. The CSL lab has a rapidly growing industrial software engineering group.
- UPPAAL group/UU (Wang Yi). The group's competence areas are in formal techniques in particular, formal semantics, modelling languages, and automated verification. It will participate in and be responsible for activities related to techniques and tools for formal verification of components, and compositionality issues of components.