Industrial Requirements in Development of Embedded Real-Time Systems -Interviews with Senior Designers

Kaj Hänninen[1,2], Jukka Mäki-Turja[1], Mikael Nolin[1]
[1]Mälardalen Real-Time Research Centre, Västerås, Sweden
[2]Arcticus Systems, Järfälla, Sweden
E-mail: kaj.hanninen@mdh.se

Abstract

The area of embedded real-time systems is undergoing changes. More and more of traditional mechanical solutions are being replaced with computer-controlled systems.

In this paper, we aim at capturing the industrial viewpoint of today's and future requirements in developing embedded real-time systems. We do this by a number of interviews with ten senior designers at four Swedish companies developing embedded applications. The study shows that developers of embedded real-time systems strive for simplicity. The main goal in developing these systems is that they are, and perceived by customers as, reliable and safe. However, the complexity of the systems is ever increasing, at the same time as the safety requirements are believed to increase in importance. The interviewees believe that in order to develop these kind of safe and reliable systems, during an increasingly complexity, there has to be tool support for development and verifications of the systems. The main support in tools is sought in the areas of model-based development and verification.

1. Introduction

There is an increasing trend towards software solutions in embedded systems. Replacing existing mechanical solutions with software solutions gives opportunities for more advanced and flexible functionality. However, the flexibility of software raises additional concerns, in that the developers must be able to guarantee that software solutions are at least as safe as the mechanical systems they replace. These facts influence the requirements in developing embedded systems.

A lot of research has addressed the requirements in development of industrial embedded systems. For example, Möller *et al* [3] present the industrial requirements on component technologies. They address both technical as well as process related requirements for component technologies to be adaptable in the vehicle domain. Åkerholm *et al* [4] presents an investigation concerning classification of quality attributes for component technologies in the vehicle domain. The investigation shows that safety, reliability and predictability characteristics are considered as the most important ones. Koopman [2] presents a broader view, based on the authors own experience with industrial systems, of requirements and design issues in embedded systems design. Graaf *et al* [1] presents an industrial inventory of seven companies developing embedded software products. Their inventory of state of practice addresses requirements engineering and architectural issues such as design and analysis. The study covers companies from different domain, e.g., developers of, mobile phones and consumer electronics, distributed data management etc.

In this study, we investigate the industrial requirements in the vehicle domain, especially requirements related to real-time issues on both a high overall level, such as safety and reliability requirements, as well as on a lower technical level, such as choice of operating system and their supporting execution models. The study is performed as series of interviews with ten senior designers at four Swedish companies (referred to as A-D). Specifically, we investigate the following issues:

- What application properties are important?
- What are the considerations in choosing an OS?
- What resources are constrained in the systems, and to what degree?
- What kind of tool support is needed in the development of future systems?

The aim of this work is foremost to explore and describe the current and future industrial requirements, as perceived by the senior designers.

The paper is organised as follows. In section 2, we describe the investigated application characteristics. In section 3, we present the interviewees expressed importance of selected application properties. In section

4, we describe the resource mangers and execution models used in the investigated systems. In sections 5 and 6, we describe the current resource situations and expressed wishes concerning tool support for development of embedded real-time systems, as expressed by the interviewees. The paper ends with a conclusion of our study.

2. Application characteristics

The investigated applications are mainly used as control applications for various types of vehicles. In addition to control functionality, the applications typically contain functionality for information handling such as logging for diagnostic purposes and functionality for interaction with the system operators.

The architectures of the examined systems are of distributed character where several nodes, electronic control units (ECUs), perform computations and communicate with each other mainly via CAN buses. The number of ECUs in the systems has typically been increasing over the years. For example, company A has tripled the number of ECUs in their applications from ’91-‘02.

The examined applications are realized by hard and soft real-time tasks. Typical technical constraints in the examined applications include; precedence relations and jitter requirements. The timing constraints, e.g., deadlines on different functionality, can vary as much as three orders of magnitude in a single application. In all of the examined applications, the control functionality is considered as being most safety critical and developed mainly using the time-triggered paradigm.

Future characteristics: The interviewees believe that the information intensity and the number of control functionality will increase in the future. They state that, in the future both legislation and assurance reasons will force development of more sophisticated control algorithms and require an increasing amount of information to be saved for diagnostic reasons. Classification of functionality in safety integrity levels (SIL) is also believed to be an important activity in the future.

3. Importance of application properties

We investigated the experienced importance of the following application properties: Analysability, Reliability, Safety, Maintainability, Testability, Portability, and Reusability. All of the properties are desirable and more or less important according to the interviewees. However, when asked to classify the importance among the properties, reliability, safety and analysability aspects are emphasised. The classification reflects the fact that developers strive for a high degree of control, both of temporal and functional properties of the application.

Analysability: Analysability of real-time properties, especially response-times, jitters and precedence relations are considered as important. However, some interviewees stress the desire of better analysis support in development tools, and state that analysing a whole system with respect to temporal and spatial attributes are very difficult, sometimes even intractable. Due to the difficulties in analysing a complete system, and for upgradeability reasons, some of the examined systems are intentionally over dimensioned with respect to processing power and memory consumption. One company considered analysability as the most important property among the examined ones (see Table 1).

Reliability: Reliability is considered, by two companies as the most important application property (see Table 1). It is stated that a company’s reputation is greatly dependent on the reliability of the developed systems; hence, it is considered as being of outmost importance to deliver reliable systems. Some interviewees state that failure in producing reliable systems often has its origin in erroneous requirement specifications rather than in the developed software itself.

Safety: One company considers safety as the most important property (see Table 1). Safety is considered as a derived property originating foremost from analysis and testability. In some examined systems, redundancy and certain safety properties are solved outside the actual software implementation, by physical cabling etc. The software in these systems can be overridden by mechanics to avoid catastrophic consequences.

Maintainability: The experienced importance of maintainability varies among the interviewees. Some interviewees consider this as an important property whereas others state that maintainability in the context of fixing bugs is unacceptable. However, there seems to be an agreement on that maintainability will have to be considered as a more important property in the future, especially in the context of upgradeability. The lifespan of the examined systems can be several decades and customers put demand on new features and require hardware replacement parts to be available during the entire lifespan of a system. This requires applications to be well structured and easy to understand for future developers.

Testability: Testability is stated as a very important application property. It is stated as a necessary property to achieve reliability and safety. System design is done with testability in mind to ease the testing effort.

Portability: The importance of portability varies. Some interviewees state that portability is of low importance since they do not change hardware or OSes that often, whereas other respondents claim that portability is definitely increasing in importance. In system where portability is an important issue, it is facilitated by separation of hardware dependent software from non-dependent.

Reusability: Reusability of both soft- and hardware is an ongoing activity in all of the examined systems. However, the amount of reusable software varies in the examined systems. Some interviewees state that reusability of architectures is not achieved until they have undergone several modifications, hence it may take years before certain parts of architectures are actually reusable. To facilitate reusability among different systems, some companies have developed common software platforms. The platforms contain all common hardware functionality and have standardised interfaces. General software components are also mentioned as reusable entities. The components are general in the sense that they are, to a large degree, application independent.

Additional properties: When asked for additional properties that are considered as important for their applications, the interviewees mentioned *Robustness*, *Scalability* and *Usability*. Robustness is defined by the respondents as ‘the absence of unexpected behaviour’ or as ‘an additional degree of reliability’. Scalability is considered in the context of development as the ability to scale systems using the available development tools. Usability of architectures is mentioned as a process related issue. In that context, the usability of architectures is said to be dependent on whether it facilitates understanding and communication between developers. All of the respondents stress the importance of architectural descriptions as means of communication between people, i.e., not only as logical or structural system descriptions.

Table 1. Relation of importance among selected application properties. Darker areas represents higher importance.

		Company			
		A	**B**	**C**	**D**
Property	*Analysability*	▓	■	░	░
	Reliability	▓	▓	■	■
	Safety	■	░	▓	░
	Maintainability				░
	Testability	░	▓	▓	▓
	Portability				
	Reusability				

4. Requirements in choice of OS

When investigating the type of technical considerations that has bearing on the choice of OS for the embedded applications, we discovered several non-technical considerations that are strong motivators in choosing OS. These requirements do not always reflect the technical need in development, e.g., unification to use the same OS within departments of a company. The technical requirements are commonly considered later on. However, the requirements on simplicity, i.e., ease of use, is a motivator both when choosing OS and among available execution models. The interviewees state that main motivations to the choice of commercial operating system are related to fact as:

- Cost (royalties, licenses)
- Availability of supported development tools related to the OS
- The supported execution models in the OS, i.e., its suitability for the application domain
- Coordination within a concern or subsidiaries to use a common OS
- The availability of fast and skilful technical support
- Recommendations originating from other companies evaluating the OS
- The popularity of the OS, i.e., the fact that the OS is used by several other companies
- The OS internal timing and memory overhead
- Safety classification issues

4.1. Technical requirements, execution models

Both time- and event-triggered execution models are used in all of the examined systems. The time-triggered model is commonly used for control functionality whereas the even-triggered model is utilised mainly for information handling for diagnostic reasons. The interviewees state that the choice of execution model in development is mainly dependent on:

- Verification possibilities, both functional and temporal
- Flexibility of adding new functionality
- Required response-time on functionality
- Simplicity of use in development

5. Resource limitations

We investigated whether, and to what degree, the following resources were considered constrained in the systems:

- Processing time
- RAM, ROM
- Communication bandwidth

On the contrary to what we expected, the resources are not considered as constrained in the examined systems. As described in section 3, many of the examined systems are intentionally over dimensioned. However, in case the systems would run out of resources, the interviewees state that they would most probably consider installing additional hardware resources rather than redesigning the way the applications utilises the resources. This is however, dependent on the urgency of system delivery. In extreme cases, functionality has been removed from the systems when the available resources have been near to full utilisation.

6. Desired development tool support

The expressed wishes, concerning support in development tools, amplify the requirements on verification, safety and reliability aspects. The concise picture seems to be requirements on simulation and verification possibilities of applications on a PC. Moreover, an integrated possibility for model-based development with MATLAB and Simulink together with automated code generation is another common desire expressed by the interviewees.

The following is a list of desired tool support, as expressed by the interviewees. The support is both technical and process related. The interviewees would like to see:

- Simulation of the embedded applications on PCs
- Support for model-based development with possibilities to exchange information between tools from different vendors
- Abstractions of graphical models, i.e., visualisation of architectures at different levels and views
- Automatic code generation, e.g., from models to source code
- Support for formal verification of source code
- Support for execution time analysis
- Fast schedulability analyses

The current support in development tools varies at the companies. For example, one company has support for simulation of embedded applications on a PC, whereas others do not have simulation possibilities at all. However, none of the examined companies has all of the listed support in their development tools.

When discussing the area of component-based software engineering, all of the interviewees state that the abstraction possibilities that components provide, is a more important motivator of component based development than the reuse motive, simply because it facilitates understanding and communication between developers.

7. Conclusions

In this paper, we presented some requirements in development of industrial embedded systems in the vehicle domain. The requirements were collected in a number of interviews with ten senior designers at four companies in Sweden.

We conclude that:

- Reliability and safety aspects are guiding the development of the applications
- Several non-technical considerations are strong motivators when choosing OS for the applications
- Verifiability as well as simplicity is sought for when choosing execution models during development
- The computational resources, in the examined systems, are not considered as specifically constrained
- A wide spectrum of different kind of tool support is desired in developing of the applications. Model-based development with simulation possibilities and automatic code generation is a prevalent desire

References

[1] B. Graaf, M. Lormans, H. Toetenel, *Embedded Software Engineering: The State of the Practice,* IEEE Software, Volume 20, Issue 6, 2003

[2] P. Koopman, *Embedded System Design Issues (the Rest of the Story),* Proceedings of the International Conference on Computer Design (ICCD), Austin, October, 1996

[3] A. Möller, J. Fröberg, M. Nolin, *Industrial Requirements on Component Technologies for Embedded Systems,* International Symposium on Component-based Software Engineering (CBSE7), Edinburgh, Scotland, May, 2004

[4] M. Åkerholm, J. Fredriksson, K. Sandström, I. Crnkovic, *Quality Attribute Support in a Component Technology for Vehicular Software,* Fourth Conference on Software Engineering Research and Practice in Sweden, Linköping, Sweden, October, 2004