Quality Attribute Support in a Component Technology for Vehicular Software

Mikael Åkerholm, Johan Fredriksson, Kristian Sandström, and Ivica Crnkovic
Mälardalen Real-Time Research Centre (MRTC)
Department of Computer Science and Engineering
Mälardalen University, Västerås, Sweden
mikael.akerholm@mdh.se

Abstract

The electronics in vehicles represents a class of systems where quality attributes, such as safety, reliability, and resource usage, leaven all through development. Vehicular manufacturers are interested in developing their software using a component based approach, supported by a component technology, but commercial component technologies are too resource demanding, complex and unpredictable. In this paper we provide a vehicular domain specific classification of the importance of different quality attributes for software, and a discussion of how they could be facilitated by a component technology. The results can be used as guidance and evaluation for research aiming at developing component technologies suitable for vehicular systems.

1. Introduction

Component-based development (CBD) is of great interest to the software engineering community and has achieved considerable success in many engineering domains. CBD has been extensively used for several years in desktop environments, office applications, e-business and in general Internet- and web-based distributed applications. In many other domains, for example dependable systems, CBD is utilized to a lesser degree for a number of different reasons. An important reason is the inability of component-based technologies to deal with quality attributes as required in these domains. To identify the feasibility of the CBD approach, the main concerns of the particular domain must be identified along with how the CBD approach addresses these concerns and what is its ability to provide support for solutions related to these concerns are.

There is currently a lot of research on predicting and maintaining different quality attributes within the Component Based Software Engineering (CBSE) community, (also called non-functional properties, extra-functional properties, and illities), [9][21][24][30][31]. Many of the quality attributes are conflicting and cannot be fully supported at the same time [6][15]. Thus, it is important for application and system developers to be able to prioritize among different quality attributes when resolving conflicts.

We provide a domain specific classification of the importance of quality attributes for software in vehicles, and discuss how the attributes could be facilitated by a component technology. The discussion contribute with a general description of the desired quality attribute support in a component technology suitable for the vehicle domain and it indicates which quality attributes require explicit support. In addition, it discusses were in the technology the support should be implemented: inside or outside the components, in the component framework, on the system architecture level, or if the quality attributes are usage dependent. Quality attributes might be conflicting; e.g., it is commonly understood that flexibility and predictability are conflicting. The ranking provided by industrial partners gives domain specific guidance for how conflicts between quality attributes should be resolved. The results also enable validation and guidance for future work regarding quality attribute support in component technologies for software in vehicular systems. This guideline can be used to verify that the right qualities are addressed in the development process and that conflicting interdependent quality attributes are resolved according to the domain specific priorities.

The starting point of this work is a list of quality attributes ranked according to their importance for vehicular systems. The list is provided through a set of interviews and discussions with experts from different companies in the vehicular domain. The results of the ranking from the vehicular companies are combined with the classification of how to support different quality attributes provided in [18]. The result is an abstract description of where, which, and how different quality attributes should be supported by a component technology tailored for the vehicular industry.

A component technology as defined in [4] is a technology that can be used for building component based software applications. It implements a component model defining the set of component types, their interfaces, and, additionally, a specification of the allowable patterns of interaction among component types. A component framework is also part of the component technology, its role can be compared to the role of an operating system, and it provides a variety of deployment and run-time services to support the component model. Specialized component technologies used in different domains of embedded systems have recently been developed, e.g., [22][23]. There are also a number of such component technologies under development in the research community, e.g., [16][20][35]. The existence of different component technologies can be motivated by their support for different quality attributes, although they follow the same CBSE basic principles. It has been shown that companies developing embedded systems in general consider different non functional quality attributes far more important than efficiency in software development, which explains the specialization of component technologies [16].

The outline of the remaining part of the paper is as follows. Section 2 describes the conducted research method, and section 3 the results. Section 4 is a discussion of the implications of the results, regarding the support for quality attributes in a domain specific component technology. Section 5 discusses future work, and finally the section 6 concludes the paper.

2. Method

The research method is divided into three ordered steps:

1. During the first step a list of relevant quality attributes were gathered;
2. In the next step technical representatives from a number of vehicular companies placed priorities on each of the attributes in the list reflecting their companies view respectively;
3. Finally a synthesis step was performed, resulting in a description of the desired quality attribute support in a component technology for vehicular systems.

The list of quality attributes have been collected from different literature trying to cover qualities of software that interest vehicular manufactures. In order to reduce a rather long list, attributes with clear similarities in their definitions have been grouped in more generic types of properties, e.g., portability and scalability are considered covered by maintainability. Although such grouping could fade the specific characteristics of a particular attribute, it put focus on the main concerns. In the ISO 9126 standard [17], 6 quality attributes (functionality, reliability, usability, efficiency, maintainability, and portability) are defined for evaluation of software quality. However, the standard has not been adopted fully in this work; it is considered too brief and does not cover attributes important for embedded systems (e.g., safety, and predictability). Furthermore, concepts that sometimes are mixed with quality attributes (for example fault tolerance) are not classified as quality attributes, rather as methods to achieve qualities (as for example safety). Finally, functionality is of course one of the most important quality attributes of a product, indicating how well it satisfies stated or implied needs. However, we focus on quality attributes beyond functionality often called extra-functional or non-functional properties.

The resulting list of quality attributes is presented below.

- *Extendibility* - the ease with which a system or component can be modified to increase its storage or functional capacity.
- *Maintainability* - the ease with which a software system or component can be modified to correct faults, improve performance, or other attributes, or adapt to a changed environment.
- *Usability* - the ease with which a user can learn to operate, prepare inputs for, and interpret outputs from a system or component.
- *Predictability* - to which extent different run-time attributes can be predicted during design time.
- *Security* - the ability of a system to manage, protect, and distribute sensitive information.
- *Safety* - a measure of the absence of unsafe software conditions. The absence of catastrophic consequences to the environment.
- *Reliability* - the ability of a system or component to perform its required functions under stated conditions for a specified period of time.
- *Testability* - the degree to which a system or component facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met. Note: testability is not only a measurement for software, but it can also apply to the testing scheme.
- *Flexibility* - the ease with which a system or component can be modified for use in applications or environments other than those for which it was specifically designed.
- *Efficiency* - the degree to which a system or component performs its designated functions with minimum consumption of resources (CPU, Memory, I/O, Peripherals, Networks).

Representatives from the technical staff of several companies have been requested to prioritize a list of quality attributes, reflecting each of the respective companies' view. The attributes have been grouped by the company representatives in four priority classes as shown in Table 1. The nature of the quality attributes imply that no quality attribute can be neglected. It is essential to notice that placing an attribute in the lowest priority class (4) does not mean that the company could avoid that quality in their software, rather that the company does not spend extra efforts in reaching it. The following companies have been involved in the classification process:

- Volvo Construction Equipment [33] develops and manufactures a wide variety of construction equipment vehicles, such as articulated haulers, excavators, graders, backhoe loaders, and wheel loaders.
- Volvo Cars [34] develops passenger cars in the premium segment. Cars are typically manufactured in volumes in the order of several hundred thousands per year.
- Bombardier Transportation [7] is a train manufacturer, with a wide range of related products. Some samples from their product line are passenger rail vehicles, total transit systems, locomotives, freight cars, propulsion and controls, and signaling equipment.
- Scania [27] is a manufacturer of heavy trucks and buses as well as industrial and marine engines.
- ABB Robotics [1] is included in the work as a reference company, not acting in the vehicular domain. They are building industrial robots, and it is the department developing the control systems that is represented.

Table 1. Priority classes used to classify the importance of the different quality attributes

Priority	Description
1	very important, must be considered
2	important, something that one should try to consider
3	less important, considered if it can be achieved with a small effort
4	Unimportant, do not spend extra effort on this

As the last step we provide a discussion where we have combined the collected data from the companies with the classification of how to support different quality attributes in [18]. The combination gives an abstract description of where, which, and how different quality attributes should be supported by a component technology tailored for usage in the vehicular industry.

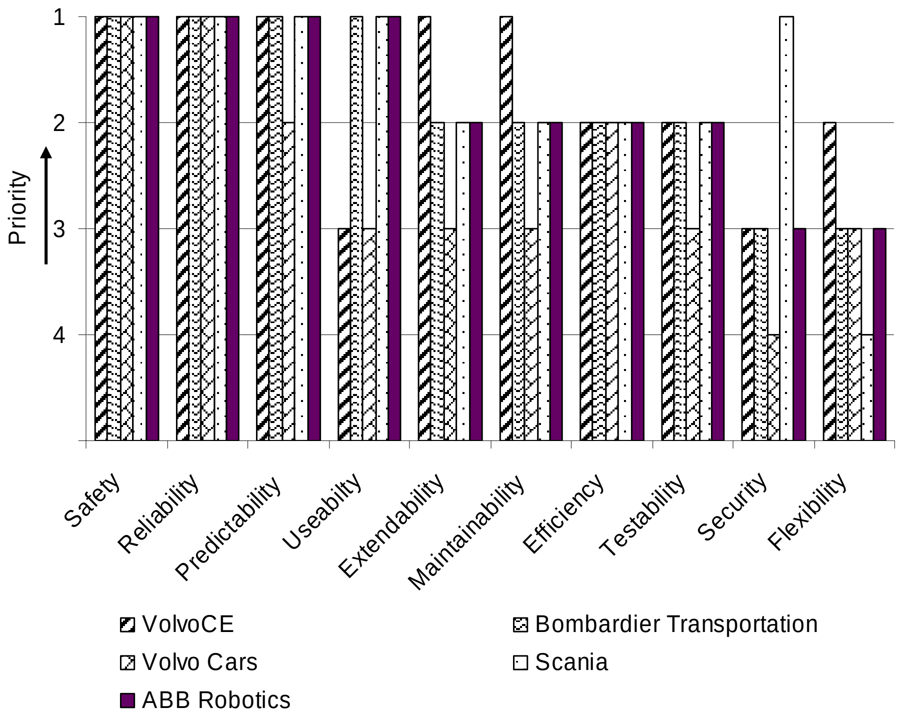

Figure 1, the results. Y-axis: priority of quality attributes in a scale 1 (highest), to 4 (lowest). X-axis: the attributes, with the highest prioritized attribute as the leftmost, and lowest as rightmost. Each of the companies has one bar for each attribute, textured as indicated below the X-axis.

3. Results

Figure 1 is a diagram that summarizes the results. The attributes are prioritized by the different companies, in a scale from priority 1 (highest), to 4 (lowest) indicated on the Y-axis. On the X-axis the attributes are presented with the highest prioritized attribute as the leftmost, and lowest as rightmost. Each of the companies has one bar for each attribute, textured as indicated below the X-axis. In some cases the representatives placed an interval for the priority of certain attributes, e.g., 1-3 dependent on application; in those cases the highest priority has been chosen in the diagram.

The result shows that the involved companies have approximately similar prioritization, except on the security quality attribute where we have both highest and lowest priority. Reasonably, the most important concerns are related to dependability characteristics (i.e. to the expectation of the performance of the systems): safety, reliability and predictability. Usability is a property important for the customers but also crucial in competition on the market. Slightly less important attributes are related to the life cycle (extendibility, maintainability). This indicates that the companies are ready to pay more attention to the product performance than to the development and production costs (in that sense a component-based approach which primary concerns are of business nature, might not necessary be the most desirable approach).

The results also shows that ABB Robotics, included as a reference company outside the vehicular domain has also approximately the same opinion. It is not possible to distinguish ABB Robotics from any of the vehicular companies from a quality attribute perspective. These companies might use the same component technology with respect to quality attribute support; thus the results in the investigation indicate that the priority among quality attributes scale to a broader scope of embedded computer control systems.

4. Discussion of the results

A component technology may have built in support for maintaining quality attributes. However, tradeoffs between quality attributes must be made since they are interdependent [6][15]. We will discuss how the different quality attributes can be supported by a component technology, and suggest how necessary tradeoffs can be made according to priority placed by industry. The discussion starts by treating the attribute that has received the highest priority (safety), and continues in priority order, in this way the conflicts (and tradeoffs) will be discussed in priority order. As basis for where support for a specific quality attribute should be implemented we use a classification from [18], listed below:

- Directly composable, possible to analyze given the same quality attributes from the components.
- Architecture related, possible to analyze given this attribute for the components and the assembly architecture.
- Derived attributes, possible to analyze from several attributes from the involved components.
- Usage dependent, need a usage profile to analyze this.
- System environment context, possible to analyze given environment attributes.

4.1. Safety

Safety is classified as dependent on the usage profile, and the system environment context. Similarly to the fact that we cannot reason about system safety without taking into consideration the surrounding context, we cannot reason about safety of a component: simply safety is not a property that can be identified on the component level. But a component technology can include numerous mechanisms that enhance safety, or simplify safety analysis. However, to perform safety analysis, usage and environment information is needed. A component technology can have support for safety kernels [25], surrounding components and supervise that unsafe conditions do not occur. Pre- and post conditions can be checked in conjunction with execution of components to detect hazardous states and check the range of input and output, used in specification of components in e.g., [8][11]. Tools supporting safety analysis as fault tree analysis (FTA) or failure modes and effect analysis (FMEA) can also be provided with the component technology.

4.2. Reliability

Reliability is architecture related and usage dependent. The dominant type of impact on reliability is the usage profile but reliability is also dependent on the software architecture and how components are assembled; a fault-tolerant redundant architecture improves the reliability of the assembly of components. One possible approach to calculation of the reliability of an assembly is to use the following elements:

- Reliability of the components – Information that has been obtained by testing and analysis of the component given a context and usage profile.
- Path information (usage paths) – Information that includes usage profile and the assembly structure.

Combined, it can give a probability of execution of each component, for example by using Markov chains.

Also common for many simple systems, the reliability for a function of two components is calculated using the reliability of the components, and their relationship when performing the function. An AND relationship is when the output is dependent on correct operation of both components, and an OR occurs when the output is created when one of the two components operates correctly.

A component technology could have support for reliability, through reliability attributes associated with components, and tools that automatically determines reliability of given usage profiles, path information, and structural relationships.

It is noteworthy that even if the reliability of the components are known it is very hard to know if side effects take place that will affect an assembly of the components. E.g. a failure caused by a component writing in a memory space used by another component. A model based on these assumptions needs the means for calculating or measuring component reliability and an architecture that permits analysis of the execution path. Component models that specify provided and required interface, or implement a port-based interface make it possible to develop a model for specifying the usage paths. This is an example in which the definition of the component model facilitates the procedure of dealing with the quality attribute. One known problem in the use of Markov chains in modeling usage is the rapid growth of the chain and complexity [29]. The problem can be solved because the reliability permits a hierarchical approach. The system reliability can be analyzed by (re)using the reliability information of the assemblies and components (which can be derived or measured).

Reliability and Safety are not conflicting attributes. Reliability enhances safety, high reliability increases confidence that the system does what it is intended to and nothing else that might lead to unsafe conditions.

4.3. Predictability

We focus on predictability of the particular run-time attributes temporal behavior, memory consumption, and functional behavior. Predictability is directly composable and architecture dependent. Prediction of temporal behavior is well explored in research within the real-time community. Depending on the run-time systems scheduling strategy, the shared resource access and execution demands of the scheduled entities, suitable prediction theories can be chosen, e.g., for fixed priority systems that are most common within industry [3][28]. The choice of scheduling strategy is also a problem that has been addressed [36]. Static scheduled systems are more straightforward to predict than event driven systems that on the other hand are more flexible. Memory consumption can be predicted, given the memory consumption for the different components in the system [13]. However, two different types of memory consumption can be identified: static and dynamic. Static memory consumption is the most straightforward to predict, since it is a simple summation of the memory requirements of the included components. Dynamic memory consumption can be more complex, since it might be dependent on usage input, and thereby be usage dependent.

Predictability is not in conflict with the higher prioritized attributes reliability and safety. Predictable behavior enhances safety and reliability, e.g., unpredictable behavior cannot be safe because it is impossible to be sure that certain actions will not take place.

4.4. Usability

Usability is a rather complex quality attribute, which is derived from several other attributes; it is architecture related and usage dependent. Usability is not directly related to selection of component technology. Software in embedded systems (the most common and important type of software in vehicular systems) is usually not visible and does not directly interact with the user. However, more and more human-machine interaction is implemented in underlying software. In many cases we can see how the flexibility of software is abused - there are many devices (for example in infotainment) with numerous buttons and flashing screens that significantly decrease the level of usability. Use of a component technology may however indirectly contribute to usability – by building standard (user-interface) components, and by their use in different applications and products, the same style, type of interaction, functionality and similar are repeated. In this way they become recognisable and consequently easier to use.

Usability as discussed above is not in obvious conflict with any of the higher prioritized quality attributes.

4.5. Extendibility

Extendibility is directly composable and architecture related. It can be supported by the component technology through absence of restrictions in size related parameters, e.g., memory size, code size, and interface size. Extendibility is one of the main concerns of a component technology and it is explicitly supported – either by ability of adding or extending interfaces or by providing a framework that supports extendibility by easy updating of the system with new or modified components.

Extendibility is not in direct conflict with any of the higher prioritized attributes. However, conflicts may arise due to current methods used for analysis and design of safety critical systems real-time systems, the methods often results in systems that are hard to extend [5]. Predictability in turn enhances extendibility, since it makes predications of the impact of an extension possible.

4.6. Maintainability

Maintainability is directly composable and architecture related. A component technology supports maintainability through configuration management tools, clear architectures, and possibilities to predict impacts of applied changes.

Maintainability is not in obvious conflict with any of the higher prioritized attributes. But as for extendibility, current state of practice for achieving safety, dependability and predictability results in systems that often are hard to maintain [5]. Maintainability increases usability, while good predictability in turn increases maintainability since impacts of maintenance efforts can be predicted.

4.7. Efficiency

Efficiency is directly composable and architecture related. Efficiency is affected by the component technology, mainly through resource usage by the run-time system but also by interaction mechanisms. Good efficiency is equal to low memory, processor, and communication medium usage.

In the requirements for a software application it might often be the case that a certain amount of efficiency is a basic requirement, because of limited hardware resources, control performance, or user experienced responsiveness. In such cases the certain metrics must be achieved, but efficiency is potentially in conflict with many higher prioritized quality attributes. Safety related run-time mechanisms as safety kernels, and checking pre- and post conditions consume extra resources and are thus in conflict with efficiency. Reliability is often increased by redundancy, by definition conflicting with efficiency. Methods used for guaranteeing real-time behavior are pessimistic and result in low utilization bounds [19], although it is a widely addressed research problem and improvements exist, e.g., [2][10].

4.8. Testability

Testability is directly composable and architecture related. A general rule for testability is that simple systems are easier to test than complex systems; however, what engineers build is not directly related to the technology itself. Direct methods to increase testability provided by a component technology can be built in self tests in components, monitoring support in the run-time system, simulation environments, high and low level debugging information [32].

Testability is not in conflict with any of the higher prioritized quality attributes. On the contrary, it supports several other attributes, e.g., safety is increased by testing that certain conditions cannot occur, predictions are confirmed by testing, maintainability is increased if it is possible to test the impact of a change. However, efficiency tradeoffs might have to be done to enable testing. A problem with many common testing methods is the probe effect introduced by software probes used for observing the system [14]. If the probes used during testing are removed in the final product, it is not the same system that is delivered as the one tested. To avoid this problem, designers can choose to leave the probes in the final product and sacrifice efficiency, or possibly use some form of non-intrusive hardware probing methods, e.g., [12]. Reliability implemented by fault tolerance decrease testability, since faults may become hidden and complicate detection by testing.

4.9. Security

Security is usage dependent and dependent on the environment context, meaning that it is not directly affected by the component technology. However, mechanisms increasing security can be built in a component technology, e.g., encryption of all messages, authorization of devices that communicate on the bus.

Methods to increase security that can be built in a component technology are often in conflict with higher prioritized quality attributes, e.g., encryption is in conflict with efficiency since it require more computing, and with testability since it is harder to observe the system. Furthermore security has a low priority, and the methods to achieve it are not dependent on support from the component technology. Hence, security can be implemented without support from the component technology.

4.10. Flexibility

Flexibility is directly composable and architecture related. A component technology can support flexibility through the components, their interactions, and architectural styles to compose systems. Methods increasing flexibility in a component technology can be, e.g., dynamic run-time scheduling of activities based on events, run-time binding of resources, and component reconfiguration during run-time.

Flexibility has received the lowest priority of all quality attributes, and is in conflict with many higher prioritized attributes, e.g., with safety since the number of different hazardous conditions increases, with testability since the number of test cases increases and it may not be possible at all to create a realistic run-time situation thus not to test the actual system either. On the other hand flexibility increases maintainability, since a flexible system is easier to change during maintenance. It is not possible to use completely static systems with no flexibility at all when user interaction is involved, but regarding to the numerous conflicts with higher prioritized quality attributes it should be kept to a minimum in component technologies for this domain.

4.11. Quality Attribute Support in a Component Technology for the Automotive Domain

Having presenting the basic characteristics of quality attributes related to component technologies, and identification of present conflicts, and suggestions on how to resolve the conflict we give a brief description of the resulting suggestion of support for quality attributes in a component technology tailored for vehicular systems in table 2.

Table 2, component technology support for particular attributes

Quality attribute	Support
Safety	Safety cannot be fully supported by a component technology. However, safety kernels surrounding components and support for defining pre- and post conditions are suggested as support.
Reliability	Reliability is supported to a large extent by a component technology. We suggest reliability attributes associated with components, path information including usage profile and assembly structure, and tools for analysis. There should also be support for redundant components when necessary.
Predictability	Predictability is supported to a large extent. Associated to the components, attributes such as execution time, and memory consumption can be specified. Tools for automated analysis can be provided with the technology.
Usability	Usability is not directly supported by a component technology.
Extendibility	Extendibility is well supported. The interfaces should be easy to extend and it should be easy to add new components to an existing system. There should be no size related restrictions with respect to memory, code, and interface.
Maintainability	Maintainability is well supported by a component technology. The support is provided through configuration management tools, and the fact that using well defined components gives a clear and maintainable architecture,
Efficiency	Efficiency is suggested to be supported to a fairly high level. We suggest support through small and efficient run-time systems, however not to the cost of suggested safety and reliability related run-time mechanisms.
Testability	Testability is supported to a large extent. The support is suggested to be monitoring possibilities in the run-time system, simulation and debug possibilities.
Security	Security is not directly supported.
Flexibility	Flexibility is not directly supported.

5. Future Work

We will continue with research towards enabling CBSE for automotive systems. One part is to continue investigating the requirements on quality attributes from the domain, with our present and other industrial partners. Another part is an analysis of particular component models to investigate their abilities of supporting these quality attributes. A third part is to enable support for quality attributes in the component technologies we are developing as prototypes suitable for the domain AutoComp [26] and SaveComp [16], but we will also asses to which extent other existing component technologies can be used in order to meet the industrial requirements.

6. Conclusions

We have presented a classification of the importance of quality attributes for software made by some companies in the vehicular domain; the results showed that the companies agreed upon the priority for most of the attributes. The most important concerns showed to be related to dependability characteristics (safety, reliability and predictability). Usability received a fairly high priority. Slightly less important attributes where those related to the life cycle (extendibility, maintainability), while security and flexibility received the lowest priority. We also included a company outside the domain in the investigation, it turned out that they also agreed upon the classification; it might be that the classification scale to a broader scope of embedded systems.

Furthermore, we have discussed how the attributes could be facilitated by a component technology, and were in the technology the support should be implemented: inside or outside the components, in the framework, or if the quality attributes are usage dependent. The discussion is concluded by a brief suggestion of quality attribute support for a component technology.

Acknowledgements

We would like to thank our industrial partners for their time, and the interest they have shown in discussing quality attributes. Thanks to Joakim Fröberg from Volvo CE, Jakob Axelsson from Volvo Cars, Mattias Ekman from Bombardier Transportation, Ola Larses and Bo Neidenström from Scania, and Bertil Emertz from ABB Robotics.

References

[1] ABB Robotics Homepage: http://www.abb.com/robotics

[2] T. F. Abdelzaher, V. Sharma, C. Lu, A utilization bound for aperiodic tasks and priority driven scheduling, IEEE Transactions on Computers, 53(3) 334 - 350, Mar 2004

[3] N. C. Audsley, A. Burns, R. I. Davis, K. W. Tindell, Fixed Priority Pre-emptive Scheduling: A Historical Perspective. Real-Time Systems journal, Vol.8(2/3), March/May, 1995.

[4] F. Bachmann, L. Bass, C. Buhman, S. Comella-Dorda, F. Long, J. Robert, R. Seacord, and K. Wallnau. Technical Concepts of Component-Based Software Engineering, Volume II. Technical Report CMU/SEI-2000-TR-008, Software Engineering Institute, Carnegie-Mellon University, May 2000

[5] A. Burns, and J. A. McDermid, Real-time safety-critical systems: analysis and synthesis, Software Engineering Journal 9(6) 267 - 281, Nov. 1994

[6] M. Barbacci, M. H. Klein, T. A. Longstaff, C. B. Weinstock, Quality Attributes, Technical Report, Software Engineering Institute, Carnegie Mellon University, 1995.

[7] Bombardier Transportation Homepage: http://www.transportation.bombardier.com/

[8] J Chessman, and J. Daniels, UML Componets – A simple process for specifying Component-Based Software, Reading, MA: Addison-Wesley, 2000.

[9] I. Crnkovic, and M. Larsson, Classification of quality attributes for predictability in component-based systems, In DSN 2004 Workshop on Architecting Dependable Systems Florence, Italy , June 2004.

[10] C. Deji; A. K. Mok,, K. Tei-Wei, Utilization bound revisited, IEEE Transactions on Computers, 52(3) 351 – 361, March 2003

[11] D. D´Souza, and A. C. Wills, Objects, Components and Frameworks: The Catalysis Approach, Reading, MA: Addison-Wesley, 1998.

[12] M. El Shobaki, and L. Lindh, A Hardware and Software Monitor for High-Level System-on-Chip Verification, Proc. of the IEEE International Symposium on Quality Electronic Design, March 2001.

[13] A. V. Fioukov, E. M. Eskenazi, D. K. Hammer, M. R. V. Chaudron, Evaluation of static properties for component-based architectures, Proc. 28th Euromicro Conference, 2002.

[14] J. Gait. A probe effect in concurrent programs. Software Practise and Experience, 16(3):225–233, March 1986.

[15] D. Haggander, L. Lundberg,and J. Matton, Quality attribute conflicts - experiences from a large telecommunication application, In proceedings of the Seventh IEEE International Conference on Engineering of Complex Computer Systems, 2001.

[16] H. Hansson, M. Åkerholm, I. Crnkovic, M. Törngren, SaveCCM – a component model for safety-critical real-time systems, Component Models for Dependable Systems (CMDS), France, September, 2004

[17] ISO/IEC standard specification, Software engineering -- Product quality -- Part 1: Quality model, ISO/IEC 9126-1, 2001

[18] M. Larsson, Predicting Quality Attributes in Component-based Software Systems, Phd Thesis, Mälardalen University Press, March 2004.

[19] C. I. Liu and J. W. Layland. Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment. Journal of the ACM, 20(1), 1973.

[20] M. de Jonge, J. Muskens and M. Chaudron, Scenario-Based Prediction of Run-time Resource Consumption in Component-Based Software Systems, Proceedings of the 6th ICSE Workshop on Component-Based Software Engineering: Automated Reasoning and Prediction, May, 2003.

[21] G. A. Moreno, S. A. Hissam, and K. C. Wallnau, Statistical Models for Empirical Component Properties and Assembly-Level Property Predictions: Towards Standard Labeling, In Proceedings of 5th Workshop on component based software engineering, 2002.

[22] O. Nierstrasz, G. Arévalo, S. Ducasse, R. Wuyts, A. Black, P. Müller, C. Zeidler, T. Genssler, R. van den Born, A Component Model for Field Devices Proceedings of the First International IFIP/ACM Working Conference on Component Deployment, Germany, June 2002.

[23] R. van Ommering, F. van der Linden, and J. Kramer. The Koala component model for consumer electronics software. IEEE Computer, 33(3):78–85, March 2000

[24] R.H. Reussner, H.W. Schmidt, and I. Poernomo. Reliability prediction for component-based software architectures. Journal of Systems and Software, 66(3):241–252, 2003.

[25] J. Rushby, Kernel for safety, in Safe and Secure Computing Systems, Blackwell Scientific Publications, Londres, 1989.

[26] K. Sandström, J. Fredriksson, and M. Åkerholm, Introducing a Component Technology for Safety Critical

Embedded Real-Time Systems, In International Symposium on Component-based Software Engineering (CBSE7) Edinburgh, Scotland , May 2004.

[27] Scania Homepage: http://www.scania.com/

[28] O. Redell, M. Törngren, Calculating exact worst case response times for static priority scheduled tasks with offsets and jitter. In: Proc. Eighth IEEE Real-Time and Embedded Technology and Applications Symposium, IEEE (2002)

[29] H. Schmidt and R. H. Reussner, Parametrized Comtracts and Adapter Synthesis, In Proceedings of 5th ICSE workshop on CBSE, 2001.

[30] H.W. Schmidt. Trustworthy components: Compositionality and prediction. Journal of Systems and Software, 65(3):215–225, 2003

[31] J. Stafford, and J. McGregor, Issues in the Reliability of Composed Components, Proceedings of 5th workshop on component based software engineering, 2002.

[32] H. Thane, Monitoring, Testing and Debugging of Distributed Real-Time Systems, Ph D Thesis , Royal Institute of Technology, May 2000.

[33] Volvo Construction Equipment Homepage: http://www.volvo.com/constructionequipment

[34] Volvo Cars Homepage: http://www.volvocars.com/

[35] K. C. Wallnau. Volume III: A Technology for Predictable Assembly from Certifiable Components, Technical report, Software Engineering Institute, Carnegie Mellon University, April 2003, Pittsburgh, USA

[36] J. Xu and D. L. Parnas. Priority Scheduling versus Pre-run-time Scheduling. Journal of Real-Time Systems, 2000.