SVERTS – Specification and Validation of Real-time and Embedded Systems

Susanne Graf[1] and Øystein Haugen[2] and Ileana Ober[1] and Bran Selic[3]

[1] VERIMAG, Grenoble, France, {Susanne.Graf, Ileana.Ober}@imag.fr
[2] University of Oslo, Oslo, Norway Oystein.Haugen@ifi.uio.no
[3]IBM, Canada bselic@ca.ibm.com

Abstract: This paper presents an overview on the workshop on Specification and Validation of Real-time and embedded Systems that has taken place for the second time in association with the UML 2004 conference. The main themes discussed at this years workshop concerned modeling of real-time features with the perspective of validation as well as some particular validation issues.

1. Introduction

Embedded applications have often strong constraints with respect to time related aspects. Moreover, overall systems may be huge, and even if the embedded hard real-time components are relatively small, there is some global interdependence and the existence of a global model in a uniform framework is an important issue. The Unified Modeling Language UML can play this role, even if the real-time aspects are not really integrated today in existing tools. UML aims at providing an integrated modeling framework encompassing architecture descriptions and behavior descriptions. A first step to the integration of extra functional characteristics into the modeling framework has been achieved by the "UML profile for schedulability, Time and Performance" [OMG03] and more recently a "UML Profile for Modelling Quality of Service and Fault Tolerance Characteristics and Mechanisms (QoS)" [OMG04]. One of the objectives of UML is to support the model driven approach (MDA) which consists in transforming models towards n executable implementations.

In order to be able to exchange models with the aim to apply formal validation, it is important to have a common understanding of the (dynamic) semantics of the given notations in the modeling and the validation tool. Other important issues in the domain of real-time are methodology and modeling paradigms allowing to break down the complexity and tools which are able to verify well designed systems.

The IST project Omega [Omega, GH04] aimed precisely at the definition of a UML profile for real-time and embedded systems with a semantic foundation and with tool support for validation. Some of the criteria for defining this profile were

- Taking into account the fact that validation is just one – although important - aspect of the problem, another main objective of modeling is deriving implementations. Therefore, the chosen profile should be appropriate for the domain of applications and not just for a particular validation tool

- Fixing a dynamic semantics and a notion of consistency between notations is important in order to guarantee consistency between the validated model and the implementation.

The profile that has been developed in Omega is a rich subset of UML with some extensions: it distinguishes a time independent subset for modeling systems consisting of reactive components, for which an operational semantics has been defined [DJP*02, ZH03] and a real-time profile compatible with SPT, but which contrary to SPT fixes a concrete syntax and provides a semantic foundation [GOO03]. Notice that this real-time framework defines a set of constructive time constraints, expressive enough to define a precise semantics for all the time constraints introduced in SPT as tag values or stereotypes by means of constraints between well defined occurrences of *events*. Events represent time points, and we have defined naming conventions for events associated with the execution of any syntactic construct[1].

Several verification approaches and tools have been adapted for handling this profile and connected to UML tools via the XMI standard exchange format. Some of the requirements for the tools and methods were:

- To be flexible with respect to the semantic choices so as to be open for easy integration of semantic variations, at least concerning the resolution of non determinism induced by the intrinsic concurrency
- Not to impose too strict constraints on the modeling approach and the development methodology, by nevertheless providing guidelines for the usage of tools
- The mapping to the input language of the tool should not provide an obstacle for the use of the tool compared with modeling directly in the tools notation, meaning that a careful reflection concerning the concepts to be preserved is needed.

An overview on the Omega validation approach can be found in [Omega, GH04], where the tool taking into account the most complete version of the profile, in particular the real-time aspects, has been presented last year at SVERTS [OGO03]. The work done in this project raised a lot of questions. Concerning the handling of semantic issues in the context of UML, one question was to which point the semantics should be fixed so that the diagrams are still able to represent an intuition that can be shared amongst different users. Another issue was to define a profile with a semantics being able to take into account the different modeling paradigms used in the context of real-time and embedded systems; there we had to realize that it is hard to model synchronous interaction directly[2] and this has been discussed at last years workshop. Concerning the interaction with CASE tools, the conclusion must be drawn that exchange of models between tools is not there today, and this is due to both weaknesses of the exchange format itself and of the existing tools.

The aim of this workshop was to bring together researchers from academia and industry to discuss and progress on these issues, as well as other issues in the context

[1] This is a similarity to UML 2.0 where with every behavior execution is associated a start and a finish event, but we have introduced a concrete syntax for these events, and we have defined a set of concrete attributes these events may have.

[2] It is possible to define workarounds allowing the description of an equivalent behaviour as by using a synchronous approach, but not in a direct way at the same level of abstraction.

of time, scheduling and architecture in UML and UML related notations, such as notations for expressing time and architecture related requirements, semantic issues, analysis tool and modeling paradigms.

2. The Contributions

Seven contributions with very high quality were presented selected from 19 regular submissions. All presentations were backed by a full paper of between 8 and 20 pages. All of the papers together with a report on the workshop's result are also published separately as a technical report at Verimag [GHOS04]. The corresponding presentation slides have been made available from the workshop website at www-verimag.imag.fr/EVENTS/2004/SVERTS. In this section, we only give summaries of each paper. The papers presented looked at the workshop's themes from very different angles.

2.1 Comparing UML Profiles for Non-functional. Requirement Annotations: the SPT and QoS Profiles [BP04]

This contribution compares two of the before mentioned UML profiles adopted by OMG for annotating non-functional requirements of software systems, SPT, formally adopted in 2003 and the QoS profile. The SPT profile was the first attempt to extend UML with basic timing and concurrency concepts, and to express requirements and properties needed for conducting schedulability and performance analysis. While the SPT profile is focused on these two types of analysis, the more recent QoS Profile has a broader scope, aiming to allow the user to define a wider variety of QoS requirements and properties.

The SPT and QoS profiles are - together with the simple time model already included in UML 2.0 - the most important standardization efforts for modelling time and a comparison is therefore important. The authors applied the two profiles to the same rather elaborate example – an embedded automation system.

While the QoS profile is almost UML 2.0 compliant, the SPT profile is a standard profile for UML 1.x and the UML 2.0 version has yet to be made. The authors claimed that SPT is easier to apply but is less flexible.

According to the results of the study there are mechanisms that are lacked in both profiles and the authors have suggested improvements.

2.2 A Formal Framework for UML Modelling with Timed Constraints: Application to Railway Control Systems [MCM04]

In the context of railway signalling systems, time related features play a relevant role at the validation process and specialists are more and more confronted with the necessity of applying formal methods as means for preventing software faults. UML offers a standard notation for high quality systems modelling; however its lack of a standardized formal semantics explains the existence of few tools supporting analysis

and verification. The authors of this contribution propose a formal support of UML model-based verification by mapping a subset of UML to time-extended ***B*** specifications [Abr96]. The main goal is to enable consistency checking through UML diagrams using existing tools for *B*. The approach is illustrated by means of the application to a railroad level crossing system with convincing results.

UML's lack of formal semantics is a recurring theme and the common approach to remedy it is to give a transformation mapping from a subset of UML to some formal language with an existing tool support. This paper also does this. The subset of UML considered here consist of a subset of UML 1.4 state machines plus *OCL* [OMG03b] for the definition of pre- and post conditions. The formal language to which this subset is transformed is *B*. As verification using the *B* approach is an interactive process, the approach brings in some extra efforts for the designer.

2.3 On Real-Time Requirements in Specification-Level UML Models [PM04]

The design of software systems usually advances from abstract to more concrete. Unfortunately, proper specification of real-time related issues has often been postponed to the implementation phase, potentially leading to increased complexity in design. This has at least partly been due to the lack of suitable abstractions and notations for expressing real-time requirements at an abstract level, using e.g. use cases. In this paper, an approach is introduced, where use-case level behavioural specifications can be augmented with real-time properties. It is also shown that these properties can be treated as a separate issue from the underlying behaviour for e.g. eased reasoning. The verification and validation of such specifications from the viewpoint of automated tool support is briefly discussed.

Contrary to the previous paper [MCM04], the authors provide also a notation for their UML-like concept. Some have compared "*joint actions*" with formalized use cases. This may be a valid comparison, but it is also possible to see these joint actions as a new concept based on pre- and post-conditions on the same general abstraction level as use cases. TLA theorem proving [Lam94] has been applied for formal verification of the example railroad crossing model, and a mapping to timed automata and corresponding model checking by the Kronos tool for model-checking of timed automata [Yov97].

2.4 Incremental Design and Formal Verification with UML/RT in the FUJABA Real-Time Tool Suite [BGHS04]

Model checking of complex time extended UML (UML/RT) models is limited today due to two main obstacles: (1) The state explosion problem restricts the size of the UML/RT models which can be addressed and (2) standard model checking approaches cannot be smoothly integrated into the usually incremental and iterative design process. The presented solution for incremental design and verification with UML/RT within the FUJABA[3] Real-Time Tool Suite [BG*04] overcomes these two

[3] "From UML to Java And Back Again"

obstacles by applying a compositional reasoning approach that is based on a restricted notion of UML patterns and components. A mapping of a – somewhat restricted - subset of the UML/RT component model and additional real time extensions for UML state diagrams to hierarchical timed automata of *Uppaal* [LPY97] is presented which enables compositional model-checking of partial models such as patterns and components. The developed tool support makes an incremental and iterative design and verification process possible where only the patterns and components which have been modified have to be rechecked rather than the whole UML/RT model.

This approach is based on the assume/guarantee paradigm for safety properties [Pnu85] which requires decomposing global specifications into properties of patterns and components and their environments. The case study used to illustrate the approach and where it can be applied successfully, is a shuttle railroad where several shuttles may join to build temporary convoys. This approach is interesting because of its obvious practical potentials.

2.5 An Analysis Tool for UML Models with SPT Annotations [HMPY04]

This paper describes a plug-in for the Rhapsody tool, which demonstrates how simple UML models with SPT annotations can be analysed using the Times tool - a tool for modelling, schedulability analysis, and code generation for timed systems. The plug-in takes as input an UML model corresponding to a model that can be handled by the Times tool, consisting of a set of components whose behaviours are specified by statecharts with operation calls, where operations are defined by SPT timing parameters for their execution time, deadline and priority. The output is a network of timed automata extended with tasks that can be analysed using the Times tool [AF*03]. In particular, the Times tool will show whether the operations invoked from the UML model are guaranteed to meet their deadlines or not under the given assumption.

A case study is presented where the method is applied to an SPT annotated UML model of an adaptive cruise controller. The tool Times is run as a plug-in to the commercial Rhapsody UML tool.

2.6 Worst-Case Execution Time Analysis from UML-based RT/E Applications [MGLT04]

Moving from code-centric to model-centric development seems to be a promising way to cope with the increasing complexity of real-time embedded systems. Validation is then one of the key-point of their development. Relating to this goal, schedulability analysis methods are generally used to validate a part of the system's real-time requirements. These methods rely on estimations of the Worst-Case Execution Time (WCET) of every task of the system. This paper presents some approaches to derive these WCET estimates from a detailed UML model of the application.

The approach aims to combine a static and a dynamic approach where the static analysis finds all possible execution paths and then the dynamic analysis means

selecting some of these executions and calculating WCET based on information on the execution time of the instruction set of the processor on which the system will be executed. The work is carried out in the Accord/UML modelling tool [LGT98] and the validation tool *Agatha* based on symbolic execution [Lug04]. The advantage of this approach over the usual one, consisting in measuring WCET of tasks, is that it provides over-approximations. It gives good scalability and will make it more attractive for practitioners. The disadvantage of the approach is that in its present form, without relatively precise information on the underlying platform, such as out-of-order executions and caches, the over approximations tend to be huge. Also, it remains to be shown if in the context of object orientation architecture dependent features can be exploited anyhow.

2.7 Validating UML models of Embedded Systems by Coupling Tools [HMP04]

To support multi-disciplinary development of embedded systems, a coupling has been realized between a UML-based CASE tool (*Rose RealTime*) – used to model the embedded software - and a tool for modelling of the continuous dynamics of physical parts of the system (*Simulink*). The aim is simultaneous simulation of the software model and its environment model in both tools, thus allowing an early exploration of the possible design choices over multiple disciplines. A first prototype of the coupling has been implemented, where it turned out that realizing a common notion of time and a proper treatment of timers and data exchange was the most difficult part. To this end, a separate component is inserted as “glue” between the tools to take care of smoothing out the differences.

The work has been inspired by the need to model *Océ* copying machines where the software resides on a very intricate mechatronic system.

3. Workshop Results

Most presentations address modelling and validation or analysis of safety critical systems and more particularly real-time issues in the context of UML. The main subjects addressed in the papers and the discussions concerned the following themes.

3.1 Modelling and semantics for validation

Several papers address the modelling of real-time systems using an extended subset of UML for which validation support can be provided. The choice of the presented approaches was to identify a subset that could be directly mapped into the input language of some tool, providing the semantics and the validation support for this approach.

Most of the resulting frameworks propose useful modelling concepts, and at least one seems to be used today in some way integrated in a real development process [BGHS04]. Nevertheless they all represent partial frameworks for modelling certain aspects of systems and none of them provides a complete framework for a model

based approach, where a rich model is maintained and appropriate verification models are just as the code obtained in an algorithmic way. Nevertheless, the presented profiles represent interesting aspects and may be adapted in the context of such a framework.

A clear consensus is that in the context of safety critical real-time systems, the existence of a formal semantics of all the defined concepts is needed in order to allow reasoning on the modelled systems. Nevertheless, it seems to be unclear if in the context of UML a standard semantic framework could be achieved; there are many actors and many different possible semantic choices, even in the context of real-time and embedded systems. A reasonable requirement could be that tool providers have to provide a readable description of the semantics chosen in their tool. How to obtain such "readable" semantics is an interesting research topic.

3.2 Validation and analysis

The properties that are important in the context of real-time systems concern both functional and reactivity properties defining constraints on the duration between occurrences of events.

Functional properties may be completely time independent, but it might be useful to consider a timed model (which is often quite abstract) in order to guarantee progress properties or for systems where time is used for guaranteeing correct synchronization (e.g. through the use of timeouts).

In systems where computations are distributed or where communication times are more important than execution times, reactivity properties can often be verified on a model in which only assumptions on *durations* are made and resource constraints are abstracted.

Finally, schedulability of a system under a given constraint on the set of resources is verified generally on models where actions are abstracted to a duration constraint (e.g. a deadline) and an execution time constraint. Important parameters of this analysis are the execution time constraints used, and obtaining good approximations – mainly worst case execution times (WCET) – is an important topic. Results obtained in other contexts (see e.g. [TSH*03]) tend to indicate that good approximations of WCET can only be obtained in conjunction with a relatively detailed model of the platform, and on the other hand the dynamic aspects brought in by object orientation disallows to really profit from these aspects.

Finally, whenever the system under analysis comprises parts controlling a physical system with continuous behaviour, one has to analyse the correct interplay between a continuous and a digital behaviour where important properties are stability and controllability for example.

Some of the validation problems can be somehow associated with particular design phases or view points. Fixing the parameters of one analysis influences the options for the others, but there is not necessarily a predefined order in which things need to be done. Also, in the context of a model based development approach, any update of the model must allow to redo all the validations which might be changed by the change.

The papers presented at the workshop, presented methods for one of the before mentioned validation or analysis problems. Most of them provide semantics in terms

of timed automata [AD94] or some extension of them as they provide a convenient model for combining time constraints, control flow and concurrency. The work on computation of WCET uses a simpler model by considering execution times of basic instructions as costs of transitions which have to be added so as to be able to compute the maximal cost of a set of finite executions. The work on the interaction between a continuous and a discrete model does co-simulation between two tools providing both discretized timed executions. It does not necessitate a hybrid model encompassing both kinds of computations as it builds upon existing tools for such models.

The feasibility of validation and analysis for realistic models is an important issue. In the context of real-time systems however, the use of abstraction and compositional verification is made more difficult due to the fact that time constraints are hard to decompose. Approaches based on property decomposition can be applied only in absence of resource dependent time constraints.

3.3 UML and safety critical systems

Notice that the problems induced by inheritance or dynamic evolution of the system configuration are not addressed by any of the contributions, but are mostly excluded from the considered settings. The appropriateness of object orientation for this kind of systems has been questioned a lot. Should we consider these approaches as an additional argument for this doubt?

Reuse is sometimes mentioned as one of the main arguments for object orientation, but rarely brought into practise. But even without reuse, object orientation has a lot of advantages concerning the structuring of a system. It is also useful, when in every instance of the system all the parameters are fixed and the configuration has a more or less) static nature, as this is required when a system has to be certified.

Clearly, an interesting research topic is to study in which way more dynamics can be introduced in the specifications of safety critical real-time systems without compromising static (off-line) verification.

4. Conclusions

With respect to the expression of time constraints there are two opposed trends:

1. There are those frameworks based on a small set of relatively low level but expressive concepts as they are handled in validation tools,
2. And those providing the user mainly with a set of relatively rigid patterns for the expression of time constraints. The contribution [BP04] show that even closely related profiles define redundant concepts which are even incompatible at the syntactic level.

Some effort is clearly still to be done concerning this issue.

Concerning validation of timing constraints an important issue is to provide methodologies allowing the application of compositional methods also in a non distributed setting.

Concerning the computation of bounds of execution times of tasks, it remains to be understood in how good approximations can be obtained in an object oriented setting.

References

[Abr96] J.-R. Abrial. The B-Book: Assigning Programs to Meanings. Cambridge University Press, 1996.

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. Theoretical Computer Science, 126(2):183-235, 1994.

[AF*03] Tobias Amnell, Elena Fersman, Leonid Mokrushin, Paul Pettersson, and Wang Yi. Times: a tool for schedulability analysis and code generation of real-time systems. Proc. of 1st International Workshop on Formal Modelling and Analysis of Timed Systems, LNCS, 2003.

[BGHS04] Sven Burmester, Holger Giese, Martin Hirsch, and Daniela Schilling: Incremental Design and Formal Verification with UML/RT in the FUJABA Real-Time Tool Suite, SVERTS 2004, in [GHOS04], 2004

[BG*04] Burmester, S., Giese, H., Niere, J., Tichy, M., Wadsack, J., Wagner, R., Wendehals, L., Zündorf, A. Tool Integration at the Meta-Model Level within the FUJABA Tool Suite. Int. Journal on Software Tools for Technology Transfer STTT, 2004 (accepted).

[BP04] Simona Bernardi and Dorina Petriu. Comparing UML Profiles for Non-functional. Requirement Annotations: the SPT and QoS Profiles, SVERTS 2004, in [GHOS04], 2004

[DJP*02] W. Damm, B. Josko, A. Pnueli, A. Votintseva, Understanding UML: A Formal Semantics of Concurrency and Communication in Real-Time UML. Proc. of FMCO'02, November 5---8, 2002, Leiden, the Netherlands, LNCS Tutorials 2852.

[GHOS04] Susanne Graf, Oystein Haugen, Ileana Ober and Bran Selic, Proceedings of the Workshop on Specification and Validation of Real-time Embedded Systems, SVERTS 2004, Lisbon. Verimag research report 2004-10-x, 2004.

[GOO03] Susanne Graf, Ileana Ober, Iulian Ober. Timed Annotations with UML. In: Workshop on Specification and Validation of UML models for Real Time and Embedded Systems (SVERTS2003), San Francisco, October 2003. accepted at STTT

[GH04] Susanne Graf, Jozef Hooman. The Omega project: Correct Development of Embedded Systems. In Proc. of European Workshop on Software Architectures, EWSA, associated with ICSE 2004, LNCS, 2004

[HMPY04] John Håkansson, Leonid Mokrushin, Paul Pettersson, and Wang Yi. An Analysis Tool for UML Models with SPT Annotations, SVERTS 2004, in [GHOS04], 2004

[HMP04] Jozef Hooman, Nataliya Mulyar, Ladislau Posta. Validating UML models of Embedded Systems by Coupling Tools, SVERTS 2004, in [GHOS04], 2004

[Lam94] L. Lamport. The temporal logic of actions. ACM Transactions on Programming Languages and Systems 16 (1994) pp. 872–923

[LGT98] A. Lanusse, S. Gérard, and F. Terrier. Real-Time Modelling with UML: The ACCORD Approach. In UML98, Beyond the Notation. Mulhouse, France. 1998.

[LPY97] K. Larsen, P. Pettersson, WangYi. UPPAAL in a Nutshell. Springer Int. Journal of Software Tools for Technology 1, 1997

[Lug97] D. Lugato, et al., Validation and automatic test generation on UML models: the AGATHA approach. Special issue of the Int. Journal on Software Tools for Technology Transfer, STTT 2004 (accepted).

[MCM04] Rafael Marcano, Samuel Colin and Georges Mariano. A Formal Framework for UML Modelling with Timed Constraints: Application to Railway Control Systems, SVERTS 2004, in [GHOS04], 2004

[MGLT04] Chokri Mraidha, Sébastien Gérard, François Terrier, David Lugato. Worst-Case Execution Time Analysis from UML-based RT/E Applications, SVERTS 2004, in [GHOS04], 2004

[OGO04] Iulian Ober, Susanne Graf, Ileana Ober. Validation of UML models via a mapping to communicating extended timed automata. 11th Int. SPIN Workshop. Barcelona, Spain, LNCS 2989, 04/2004, accepted for publication in STTT.

[OMG03] OMG. UML Profile for Schedulability, Performance, and Time, Version 1.0, formal/03-09-01, 09/2003.

[OMG03b] OMG. Object Constraint Language, version 2.0. final adopted specification, document ptc/2003-10-14, 10/2003.

[OMG04] OMG. UML Profile for Modelling Quality of Service and Fault Tolerance Characteristics and Mechanisms. Specification, ptc/2004-06-01, 06/2004.

[Omega] The homepage of the Omega project can be found at http://www-omega.imag.fr/

[PM04] Risto Pitkänen and Tommi Mikkonen. On Real-Time Requirements in Specification-Level UML Models, SVERTS 2004, in [GHOS04], 2004

[Pnu85] A. Pnueli. In Transition from Global to Modular Temporal Reasoning about Programs, in Logics and Models for Concurrent Systems, NATO, ASI Series F, Vol. 13, Springer Verlag, 1985

[TSH*03] St. Thesing, J. Souyris, R. Heckmann, F. Randimbivololona, M. Langenbach, R. Wilhelm, Ch. Ferdinand. An Abstract Interpretation-Based Timing Validation of Hard Real Time Avionics. Proc. of the Int. Performance and Dependability Symposium (IPDS), 2003.

[Yov97] Sergio Yovine. Kronos: A verification tool for real-time systems. In the Int. Journal on Software Tools for Technology Transfer, STTT 1 (1997) 123–133

[ZH03] M. van der Zwaag, J. Hooman. A Semantics of Communicating Reactive Objects with Timing. In Proc. of Workshop on Specification and Validation of UML models for Real-Time Embedded Systems, SVERTS 2003, technical report Verimag 2003/10/22, accepted at STTT