Timing Modeling with AUTOSAR

Current State and Future Directions

Marie-Agnès Peraldi-Frati
INRIA, CNRS
I3S, Université de Nice Sophia-Antipolis, CNRS, France
map@unice.fr

Hans Blom, Daniel Karlsson
Volvo Technology AB
405 08 Göteborg, Sweden
hans.blom,daniel.b.karlsson@volvo.com

Stefan Kuntz
Continental Automotive GmbH
93055 Regensburg, Germany
stefan.kuntz@continental-corporation.com

***Abstract*—In the automotive industry, the Automotive Open System Architecture AUTOSAR is established as a de-facto standard and is applied in a steadily increasing number of development projects. In addition, AUTOSAR attracted the attention of other non-automotive industries, like railway, agriculture and construction machines, power generation and marine technology. The first versions of the standard successfully achieved the objective of integrating in a common framework various components from different suppliers and ensuring their interfaces interoperability. In actual and future versions of the standard, the objective becomes even more ambitious as it considers behavioral and timing characteristics of these components. Therefore, this paper presents the current status of AUTOSAR Release 4.0 concerning the behavioral modeling and timing characterization of components and opens several research and development directions for future extensions of the standard.**

Keywords-Timing Modeling; Timing Analysis, AUTOSAR; EAST-ADL, Multiform Time, Probablisitic Timing

I. INTRODUCTION

There are no doubts about the fact that the Automotive Open System Architecture AUTOSAR [1] is one of the prominent success stories in the automotive industry. This architecture ensures proper exchange of information between various roles involved in the development of automotive software systems and in addition supports modularity, scalability, transferability, re-usability, and facilitates integration. From a structural point of view AUTOSAR succeeded in establishing a framework that provides significantly more certainty about integration of software components from a variety of suppliers. However, due to the fact that dynamic and temporal aspects are becoming more and more important in any development in the automotive industry, timing modeling and analysis are crucial for the further success of AUTOSAR. Due to this fact the AUTOSAR has introduced the AUTOSAR Specification of Timing Extension [2] in its Release 4.0.1 and continues to maintain this specification in subsequent releases. This paper describes the current state of these activities in section II and envisions some future directions in section III that should be considered by AUTOSAR.

II. CURRENT STATE

A. AUTOSAR

The Automotive Open System Architecture AUTOSAR provides a common software architecture and infrastructure for automotive systems of all vehicle domains. The main success factors of AUTOSAR are standardizes interfaces on different levels respectively layers of the software architecture. In essence, AUTOSAR is a component and module based architecture consisting of three layers: AUTOSAR Software Components commonly called AUTOSAR Software, Virtual Function Bus (VFB) including the Runtime Environment (RTE), and Basic Software. Along with these layers a methodology has been defined that supports development activities in all steps of the software development for an automotive software system: Vehicle system description including system constraints, system configuration, ECU description and configuration. In the system phase the OEM specifies all software components that are required to provide all the functionalities in a vehicle, as well as the system's topology consisting of sensors, actuators, electronic control units (ECU) and networks connecting these hardware components. In a subsequent step the vehicle manufacturer maps the software components to the various ECUs in the system and provides every supplier an extract of this system description that contains all relevant information about the system in order to develop the specific ECU. In the ECU phase the suppliers in turn use this information to continue the development and providing the required runtime environment and basic software to operate the software components mapped to this specific ECU. This approach eases not only the integration of single software components in a given system, but also the integration of all components delivered by a large number of suppliers.

Since the introduction of the AUTOSAR Release 4.0.1 the AUTOSAR Specification of Timing Extensions, often referred to as the AUTOSAR Timing Extensions—are part of the

978-3-9810801-8-6/DATE12/©2012 EDAA

AUTOSAR specifications and enable one to express timing constraints in a standardized format.

B. Events and Event Chains

AUTOSAR timing models consist of timing descriptions, expressed by events and event chains, and timing constraints that are imposed on these events and event chains. Events refer to locations in AUTOSAR models at which the *occurrences* of events are observed. The AUTOSAR Specification of Timing Extensions defines a set of predefined event types related to one of the AUTOSAR views: Virtual Function Bus VFB View (VFB Timing), Software Component View (SW-C Timing), System View (System Timing), Basic Software Module View (BSWM Timing), and ECU View (ECU Timing). In particular, one uses these events to specify the reading and writing of data from and to specific ports of software components, calling of services and receiving their responses (VFB Timing); sending and receiving data via networks and through communication stacks (System Timing); activating, starting an terminating executable entities (SW-C Timing); and last but not least calling basic software services and receiving their responses (ECU Timing and Basic SW Module Timing).

Event chains specify a causal relationship between events and their temporal occurrences. The notion of event chain enables one to specify the relationship between two events, for example when an event A occurs then the event B occurs, or in other words, the event B occurs if and only if the event A occurred before. In the context of an event chain the event A plays the role of the *stimulus* and the event B the role of the *response*. Event chains can be composed of existing event chains and decomposed into event chain—in both cases the event chain plays the role of *event chain segments*.

C. Event Triggering Constraints

The notion of *Event* is used to describe that in a system specific events occur and also at which locations in this system the occurrences are observed. In addition, an *Event Triggering Constraint* imposes a constraint on the occurrences of an event, which means that the event triggering constraint specifies the way an event occurs in the temporal space. The AUTOSAR Specification of Timing Extensions provides means to specify periodic and sporadic event occurrences, as well as event occurrences that follow a specific pattern (burst, concrete, and arbitrary pattern).

D. Other Constraints

Like event triggering constraints impose timing constraints on events and their occurrences; the *latency* and *synchronization* timing constraints impose constraints on event chains. In the former case, a constraint is used to specify a reaction and age, for example if a stimulus event occurs then the corresponding response event shall occur not later than a given amount of time. And in the latter case, the constraint is used to specify that stimuli or response events must occur within a given time interval (tolerance) to be said to occur simultaneous and synchronous respectively.

E. Timing Modeling and Analysis

Different notion of time in a design process: Time is a major concern in the design of embedded automotive systems. In this domain the mechanical parts together with the physical laws introduce a first form of time referred to as *physical time.* In computer systems, this ideal time is approximated by a *discrete model* of time. Unfortunately, this notion of discrete time is not sufficient in distributed automotive systems. The distributed nature of the software on different ECUs connected by networks requires the modeling of *multiple time bases*. Each hardware timing behavior is modeled with its own time base (speed rate, jitter, offset …) and this raises the problem of time bases synchronization. For performance evaluation or hard real-time constraints, a time model restricted to discrete, time and events is not sufficient. Synchronization with physical time becomes necessary. Section III.D explains how a high level modeling of time can be related to the physical time.
For analysis purposes, the notion of worst-case time is essential. However, these worst-case scenarios may be very rare in practice. Thus, *uncertain time* is introduced for soft real-time systems [3] and even for some hard real-time systems where the application allows for a given failure rate (e.g. probability of missing a deadline could be as small as the probability of hardware failure).

Time modeling and analysis: Embedded systems with hard real-time constraints need reliable guarantees for the satisfaction of their timing constraints. At the different levels of a development process, these guarantees can be obtained by multiple timing-analysis methods.

The underlying concept of time used in EAST-ADL and AUTOSAR is a *discrete time*. In these models, time is implicit and refers to the universal chronometric one. The behavior of software components representing the physical system is generally handled by tools such as Modelica [3] and Simulink [5] which are suitable for modeling and simulating heterogeneous systems (linear and non-linear, continuous, discrete and hybrid, …). These solvers can simulate systems with state events, such as discontinuities, including instantaneous changes in system dynamics by considering an ideal fast-enough processor execution support. This is called the zero execution time assumption.

The integration of the hardware architecture characteristics in a design becomes a major improvement in embedded systems design. Integration of parameters such as predicted execution for high level functions, CPU allocation, CPU timing behavior, network rate, allows a high level modeling and analysis of systems. A first step for introducing the hardware modeling and allocation of software onto the hardware has been proposed in SysML [6], AADL [7] and MARTE [8]. The timing model of MARTE allows an explicit modeling of multiple time bases systems. This concept is of particular interest when considering multi clocks distributed systems. A MARTE multi time base system can be simulated with the TimeSquare tool [9].

At the AUTOSAR level, traditional scheduling algorithms and analysis methods provide deterministic timing guarantees

(i.e., all task instances meet their deadline) which take into account worst-case timing information. Depending of the timing characteristics of tasks, one can choose algebraic algorithms such as deadline monotonic (DM) for fixed priority policy or an Earliest Deadline First (EDF) for variable priority policy. These algorithms have been proposed in [10]. Results obtained from this static analysis are valid for all non-interrupted program runs with all inputs. Another analysis tools provide WCET analysis. A survey of these tools can be found in [11]. Some of them can be used at different levels (analysis, design) such as SymTA/S from Symtavision, INCHRON Tool-Suite, or RTAW at design and implementation levels, In these tools, the integration of system parameters (communications controllers and buses) in allows a simulation and a more accurate prediction of worst case response.

Some others tools need implementation code as input which could be the implementation code for the application (AiT from AbsInt) or any code generated by intermediate tools (INCHRON Tool-Suite and TargetLink from dSPACE). They determine upper bounds for the worst-case execution times of tasks in real-time systems. For example, AiT takes as input an executable containing the task to be analyzed, a description of the hardware on which the task is running including a description of (external) memories and buses (i.e. a list of memory areas with minimal and maximal access times), and code annotations providing additional information like targets of indirect jumps, loop bounds, etc. RapiTime is used during system testing, where detailed timing measurements are taken of the software running on the real hardware, with operating system scheduling/interrupts etc. Static analysis of the source code structure is used in conjunction with test data to compute the WCET for functions execution on the system

Statistical and uncertain time refer to modeling uncertainties on task/resource parameters. Such information can be provided or processed by tools or can be inputs of analysis tools (RTAW tools). For example, with Trace Analyzer from Symtavision, RTaW-Sim or INCHRON Tool-Suite, statistical results on system behavior or bus response times can be obtained by analyzing execution. In addition, the INCHRON Tool-Suite takes this information as input for providing system simulation.

III. FUTURE DIRECTIONS

Besides the fact that utilizing AUTOSAR has a lot of advantages and provides significant benefits, there are still some areas were AUTOSAR can and should be improved in order to maintain and even increase momentum. The following sections mention some of these areas identified by the authors.

A. Timing Requirements and Traceability

An important issue in automotive design process is to close the gap between requirements and constraints modeling and verification of these constraints at the different levels of the development. Concerning AUTOSAR, the objective would be to improve the validation of higher level requirements and constraints while considering the effective properties at the implementation level.

At this level, timing analysis is based on worst case duration execution of tasks. Results obtained are generally over-pessimistic as they consider all possible execution of tasks and all possible task states, without considering the dynamic of resources. The impact of such analysis for dimensioning execution platforms is critical in embedded system where a limited amount of resource is available, like processor, memory, bus access, power, etc.

Usually the industrial designers take a margin equal up to 30% of the task WCET for dimensioning their execution platforms. One solution for reducing the set of possible state of tasks and for obtaining a more accurate view of the resource states is to obtain relevant traces of execution of the system. These traces can be generated during the software integration phase by an instrumentation of the code. The analysis of traces allows excerpting performance of the HW/SW-platform and some relevant information of the software. This topic is currently explored in the Artemis PRESTO [12] project which has the objective of improving test-based embedded systems, by integrating test trace exploitation and design space exploration techniques. The expected result of the project is to establish functional and performance analysis and platform optimization.

In the context of AUTOSAR, exploitation and analysis of these traces will confront effective platform results with the high level specification. Timing properties and errors must be traced back to the initial models allowing an improvement of the design and a more accurate modeling.

The French ANR RT-Simex project [13] addresses this traceability issue between execution code and models in the specific domain of embedded systems.

B. Probabilistic Timing

Several functionalities in automotive systems are so called *hard-real time systems* which mean that the violation of timing constraints leads to severe system failure. For these kinds of applications it is necessary to define deterministic timing constraints such as worst-case execution times (WCET) and strict end-to-end deadlines. However, in automotive system design, one also has to cope with functionalities which can be classified as *firm real-time systems*. For such functionalities infrequent deadline misses are tolerable, but degrade the system's quality of service. Timing constraints for such kind of functionalities cannot be very well described using deterministic timing constraints. For this reason, the TIMMO-2-USE project introduces the capability to specify probabilistic timing constraints and extends the methodology and tool landscape to deal with such timing information. In particular, it shall be possible to describe probabilistic timing properties and constraints for events and event chains. For example, the end-to-end delay of an event chain must be smaller than 10 ms in 99% of the cases. Obviously, existing methods and tools for analyzing timing constraints must be adapted: A schedulability test cannot only return true or false, rather the answer should be providing a probability of the schedulability.

C. Relationship with EAST-ADL

AUTOSAR is primarily concerned with topics related to software and hardware architecture and implementation. Decisions regarding timing are taken already on higher levels

of abstraction respectively in development phases conducted prior to software architecture, design and implementation. For example, decisions about appropriate sampling rates of signals are taken during control engineering activities and not during software design and implementation.

On the other hand, the architecture design language EAST-ADL provides the modeling and model framework with the description of the system at different levels of abstraction: Vehicle, Analysis, Design, Implementation and Operational Level. In this context, the AUTOSAR models are then kept in the Implementation Level and the solutions described by these models are based on the result from the EAST-ADL Design Level including requirements and timing constraints. Tracing between the abstraction levels is supported by concepts in EAST-ADL and the AUTOSAR model is hence stand-alone in this context.

D. Multi-form Time on Implementation Level

The modelling of multiple heterogeneous time bases in the time model allows a high level modelling of temporal aspects of a system. The goal is to apply scheduling analysis onto these models. Such analysis requires two conditions. The first one is, at the behavioural modelling level, the necessity to cope with a common notion of time. The second condition is the integration of the execution platform model and the allocation constraints in order to integrate the physical time (the one of the CPU processor).

High level modeling of multiple time bases: The TIMMO-2-USE project proposes a language TADL2 which integrate an explicit notion of *time bases*. Time bases make it possible to integrate explicit and multi-form notions of time.

The ignition control system is a typical example where timing constraints are associated with different time bases, i.e. sensor acquisition periods are measured on the universal clock whereas the time for inlet valve opening is measured on a crankshaft angle position. Thus, two different dimensions are used (chronometric and angle) and timing constraints refer to these dimensions.

The following example shows examples of dimension declaration in TADL2:

(1) **dimension** *chronometric* { ns:1, micros:10^3, ms: 10^6}

(2) **dimension** *angle* { degree: 1, rotation: 360,}

Dimension (1) refers to the classical time dimension, the different units used in this dimension and the conversion factors between these units. Dimension (2) refers to a crank-shaft rotation.

Time bases are associated with a dimension and represent a possibly infinite set of instants (ticks) strictly ordered. Thus, the *timebase Universal* (3) is of type *chronometric* and its precision is the nanosecond (ns).

(3) **timebase** *Universal*: *chronometric* { 1 ns}

(4) **timebase** *Crankshaft*: *angle* {1 degree}

Relation between time bases

Time bases may be declared at different level of a design and relate to each others.

(5) 1 rotation **on** *Crankshaft* = *speed* ms **on** *Universal*

The timing expression (5) shows that a *Crankshaft* time base related to the *Universal* one with a conversion factor based on the engine rotation speed. Such timing expression allows conversion rules between values measured on different dimensions.

Multiple time bases at the implementation level: At the implementation level the main issue is the integration of the timing characteristics of the execution platform. One classical issue is the de-synchronization of CPU clocks and the potential distortion that can appear when allocating software on these processors. The time base declaration is a way to cope with this problem by modelling the inner characteristics of processor (speed, jitter, drift, etc.) or communication supports (rate…). Expression (6) reads that the ECU1 time base has a drift of 4 micros comparing to the Universal one whereas expression (7) means that clock of ECU1 goes two times faster than ECU1 clock.

(6) **timebase** *ECU1*: *chronometric* {1 *ms* on *ECU1* = 96 *micros* on *Universal*}

(7) **timebase** *ECU2*: *chronometric* {1 *ms* on *ECU1* = 2 *ms* on *ECU1*}

Multiple time bases allow a high level abstraction of time. This abstraction may concern both software and hardware parts of a system. A high level analysis of the system can be conducted that may lead to strategic choices earlier in the design or on the contrary, postpone to the lower level, choices concerning the execution platform. Arithmetic rules have been defined to convert timing expression of different time bases in order to deal with a common one at the implementation level.

E. Timing Analysis and Tools

Timing analysis of automotive systems includes worst and best case analysis and this includes statistics and information on worst and best seen cases. In the TIMMO-2-USE project we also consider probabilistic analysis where uncertainties and distributions are included in the description of the timing behavior.

Analysis tools use different algorithms to analyze the system and to present the results to facilitate design decisions. Tools may also include the simulation of the system behavior to find complex timing properties.

The TIMMO-2-USE methodology defines the following main steps:

1. Modeling of a solution from the requirements
2. Finding the timing properties of the solution
3. Analysis of the timing properties, this may include the determination of complex timing properties

4. Verification and validation where the fulfillment of the constraints is checked.
5. Specification of timing requirements that serves as the input for a more concrete abstraction level in the model.

The tool support for these methodology steps includes:

- Modeling at different abstraction levels using tools for EAST-ADL and AUTOSAR complemented with timing information. A bearing idea in TIMMO-2-USE is that well-defined semantics of the timing constructs are used to facilitate a common understanding in supporting analysis tools.

- Exchange of models between modeling tools and analysis tools. This is supported by a concrete syntax i.e. an exchange format.

- The feedback of analysis results to the system model to update it, possibly by integration of tools. This way the knowledge of the system and what can be guaranteed is collected.

F. Tool Support is Key to Timing Modeling and Analysis

The AUTOSAR Development Cooperation already realized that one of the key success factors for applying AUTOSAR are tools supporting various roles in the definition, development, integration, and testing of automotive software systems. For this purpose the development community AUTOSAR Tool Platform Artop [14] has been established that bases its work on Eclipse technologies, like EMF, and the Eclipse Platform. The primary objective of this tool platform is to provide basic functionalities for AUTOSAR development tools in order to ease the development of such tools and to relieve the tool vendors from dealing with the peculiarities of specific AUTOSAR releases. On the other hand the intention is to attract software tool vendors to use this platform and ensure interoperability with regard to accessing information contained in AUTOSAR models.

From a timing modeling and analysis perspective, further and strong tool support is required and those tools shall provide capabilities to deal with timing information during the software definition, design, implementation, analysis, testing, and verification process. The authors believe that a promising first step to gain more momentum in the development of such tools is to enlarge the community by making Artop available through the Eclipse Automotive Industrial Working Group [16]. Since AUTOSAR gained the attention of other industries, like railway and marine technology, the community shall be even enlarged.

IV. CONCLUSIONS

Indeed, AUTOSAR gained momentum and continues to find its ways into more and more development projects in the automotive industry. Compared to other topics already covered by AUTOSAR the capability to specify timing constraints using the AUTOSAR Timing Extensions has been introduced recently. Despite the fact that the introduction of this capability is a huge step forward regarding the topic of timing, there are still some gaps to close and some improvements to be initiated.

The main areas of improvement are introducing multiform timing and time base, probabilistic timing, integrate AUTOSAR into a seamless system development process and methodology spanning across various levels of abstraction. Last but not least, the tool support for maintaining, processing and analyzing timing information is of crucial importance.

ACKNOWLEDGEMENT

This document is based on the TIMMO-2-USE project in the framework of the ITEA2, EUREKA cluster N°3674. The work has been funded by The French Ministry for Industry and Finances, the German Ministry for Education and Research (BMBF) under the funding ID 01IS10034, and the Swedish governmental agency for innovation systems (VINNOVA). The responsibility for the content rests with the authors.

REFERENCES

[1] AUTOSAR Development Cooperation. http://www.autosar.org

[2] AUTOSAR Specification of Timing Extensions, 1.1.0, AUTOSAR Release 4.0.2, 2010-11-03, AUTOSAR Development Cooperation.

[3] Sorin Manolache, Petru Eles, Zebo Peng, Schedulability analysis of real-time systems with stochastic task execution times, Journal ACM Transactions on Embedded Computing Systems (TECS), Volume 3 Issue 4, November 2004, 2002

[4] Modelica, Modelica Homepage, https://www.modelica.org/, March 2011

[5] Simulink, http://www.mathworks.de/products/simulink/index.html

[6] OMG. *Systems Modeling Language (SysML) Specification 1.1*. Object Management Group, May 2008. OMG document number: ptc/08-05-17

[7] AADL, http://www.aadl.info/aadl/currentsite/start/index.html

[8] OMG UML *Profile for Modeling and Analysis of Real-time and Embedded Systems*, MARTE V1.0. Object November 2009. OMG document number: formal/2009-11-02.

[9] TimeSquare Model Development Kit, INRIA, http://www-sop.inria.fr/aoste/dev/timesquare/

[10] C.L. Liu and J.W. Layland, *Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment*, Journal of the ACM, Vol. 20, No. 1, pp. 46-61, January 1973

[11] Reinhard Wilhelm, Jakob Engblom, Andreas Ermedahl, Niklas Holsti, Stephan Thesing, David Whalley, Guillem Bernat, Christian Ferdinand, Reinhold Heckmann, Tulika Mitra, Frank Mueller, Isabelle Puaut, Peter Puschner, Jan Staschulat, and Per Stenström. The worst-case execution time problem - overview of methods and survey of tools. Transactions on Embedded Computing Systems, 7(3):1-53, 2008

[12] Artemis PRESTO project, http://www.presto-embedded.eu/

[13] J DeAntoni, F. Mallet, F.Thomas. G Reydet, J-P Babau, Ci Mraidha, L Gauthier, L Rioux, N Sordon. RT-Simex: Retro-Analysis of Execution Traces, 18th symposium on the Foundation of Software Engineering (FSE), Santa Fe, USA, November 2010

[14] Artop, AUTOSAR Tool Platform, http://www.artop.org

[15] TOPCASED The Open-Source Toolkit for Critcal Systems, http://www.topcased.org

[16] Eclipse Automotive Industrial Working Group, http://wiki.eclipse.org/Auto_IWG