ITEA 2

INFORMATION TECHNOLOGY FOR EUROPEAN ADVANCEMENT

ITEA 2 - 09033

TIMMO-2-USE

Timing Model – Tools, algorithms, languages, methodology, USE cases

Report type	Deliverable D1.2
Report name	Use Cases
Report status	Consortium Confidential
Version number	Version 1.3
Date of preparation	2011-03-27

TIMMO-2-USE Partners

AbsInt Angewandte Informatik GmbH

Arcticus Systems AB

Chalmers University of Technology

Continental Automotive GmbH

Delphi France SAS

dSpace GmbH

INCHRON GmbH

Institute National de Recherche en Informatique et Automatique INRIA

Mälardalen University

Rapita Systems Ltd, UK

RealTime-at-Work

Robert Bosch GmbH

Symtavision GmbH

Technische Universität Braunschweig

University of Paderborn

Volvo Technology AB

Project Coordinator

Dr. Daniel Karlsson

Volvo Technology AB

Dept 6270, M2.7

405 08 Göteborg

Sweden

Tel.: +46 31 322 9949

Email: Daniel.B.Karlsson@volvo.com

© Copyright 2010-2011: The TIMMO-2-USE Consortium

Authors

Stefan Kuntz, Continental Automotive GmbH (Editor)

Document History

Version	Date	Description
1.0	2011-02-21	First version.
1.1	2011-02-28	Added a note to some of the Specific Use Cases in order to explain why they are not fully described. Corrected some of the formatting in order to improve readability.
1.2	2011-03-09	Added description for the specific use case “Develop Body Controller following Top-down Approach”.
1.3	2011-03-27	Added specific use case "Integrate Re-Usable Component" to the list of use cases.

Table of contents

1 Introduction

Purpose

The purpose of this document is to document all use cases that have been defined during the course of work package 1 “Use Cases and Requirements”.

Scope

This document contains the use cases important for work package 2 “Language” to work package 5 “Validation”.

Format of Use Cases

Every use case in this document is described using the format as described below:

Name: This field contains the use case identification (Main Use Cases), or an active-verb goal phrase that names the goal of the use case’s actor (Specific Use Cases).

Alias: An active-verb goal phrase that names the goal of the use case’s actor.

Description: **Objective/Goal**: This section states the objective/goal of the use case.

Description: This section describes the use case in more detail and provides further information about the background important to know.

Actors: A list of actors involved in the use case.

Stakeholders: A list of stakeholders interested in the results of the use case respectively the fact that the use case is performed.

Originator: The originator of the use case, when the field “Author” does not contain a name. This section is optional

Status: This field contains one of the following states: Proposed, Approved, Rejected, and Implemented.

Author: This field contains the author’s name who contributed the use case. Otherwise the name of the editor of this document is listed.

Explanation: In the course of the TIMMO-2-USE project this field is used later to provide further information on the requirement’s status.

Type: UseCase

Relations: This field lists the requirements associated with this use case. In case of a Main Use Case this field lists all the Specific Uses Cases assigned to this use case.

2 Use Cases

2.1 Main Use Cases

This section describes the Main Use Cases identified by the work package 1.

2.1.1 UC#0001 - Specify Time Budgets

Name: UC#0001 - Specify Time Budgets
Alias: OEM_Supplier_Timing_Analysis
Description: **Use Case Title/Name:** Specify Time Budgets

Objective/Goal: An E2E function normally spans over several ECUs and across the responsibility of multiple suppliers. OEMs need to divide the overall end-to-end latency to the ECUs and the communication channels, and assign these timing budgets to the suppliers.

Description: At the beginning of a project, the OEM must properly decide the time budgets for each ECU and communicate the specification to the suppliers. During the development process, the OEM and the suppliers want to keep the two-way feedback. When the suppliers have refined solutions at the proper abstraction level, the OEM can estimate if the time budgets are realistic, and may either ask the supplier to improve the solution or adjust the time budgets. On the other hand, given the timing estimates of the individual parts, the OEM may revise the timing requirements on vehicular functions to achieve optimal performance or cost of the entire vehicle.

We need to perform WCET analysis on all relevant levels of abstraction, although the cross-supplier issue arises mostly on the implementation level and possibly to some extent also on the design level.
Vehicle: N/A
Analysis: Simulink (or other behavioral) model using some hardware-independent time unit. This type of analysis can be used to determine properties on hardware needed to satisfy the timing budget.
Design: Simulink model (or other behavioral) on hardware with given characteristics. Can we say something about the OS task and communication bus schedules?
Implementation: C code on concrete hardware.

Implied TADL Support and Relevance to TIMMO-2-USE
This use case is related to the following work packages.
WP2 (Language)
Structural extensions, including (1) the modeling constructs for timing budgets, hardware timing characterization, timing properties of executable code and communication channels, etc. (2) traceability between the analysis results at different abstraction levels.
Algorithm specific extensions: Language constructs to support relevant methods for timing analysis, e.g., the estimation of WCET and communication latency.

Methodological extensions: Effective communication between the OEM and the suppliers; Progressive negotiation on timing budgets; Support for different proposals.
Semantical reasoning
WP3 (Algorithms & Tools)
More precise timing analysis methods to obtain good timing estimates at the analysis and design levels.
Improved timing information exchange between tools and stakeholders; optimized tool chain based on exchange format induced by the new TADL definitions.
WP4 (Methodology)
Better support for collaboration through (1) the investigation on what information has to be shared between OEM and the suppliers while maintaining IP integrity of the collaborators. (2) The negotiation on the timing budgets among all collaborators.
Virtual system integration: Estimation and validation of the overall timing requirements of the vehicular function based on the design models at the analysis and design levels, before the executable code is available.

Actors: Function owners, function developers, system testers

Stakeholders*:* Automotive OEM and suppliers

Status: Approved
Author: Daniel Karlsson
Explanation: None
Type: UseCase
Relations:

INRIA#0002 - Time bases
VTEC#0012
CAG#0051 - Reuse of timing constraints
CAG#0002 - Event chains between LoA
CAG#0025 - Safety (timing)
INRIA#0005 - Executable models
VTEC#0006
Transform Timing Information from Analysis Level to Design Level
Derive Timing Requirements from Open-Loop Control Algorithms
Perform Timing Analysis On Code-Level (ABS#UC0002)
Develop Cruise Control following Top-down Approach
Develop Engine Management System On Implementation (AUTOSAR) Level
Transform Continuous Time Model to Discrete Time Model
Transform Timing Information from Design Level to Implementation Level
Perform FlexRay Simulation
Derive Timing Requirements from Closed-Loop Algorithms
Generate Test bench for Non-functional Properties
VTEC#0003 - Methods for estimating WCET at analysis and design levels.
VTEC#0005
VTEC#0025
VTEC#0038
CAG#0038 - Timing Analyses
VTEC#0014 - Tool support for comparing alternative timing solutions
VTEC#0004 - Timing budget negotiation between OEM and supplier
VTEC#0032
VTEC#0002
VTEC#0034
CAG#0005 - Hardware

VTEC#0001
INRIA#0004 - Functional time
CAG#0001 - Events between LoA
VTEC#0013
CAG#0034 - Automation
VTEC#0035 - Methods for timing characterization of hardware
VTEC#0033 - Methods for timing characterization of behavior/algorithm
INRIA#0001 - Multiform concepts of Time
VTEC#0011

2.1.2 UC#0002 - Specify Mode Dependent Timing Information

Name: UC#0002 - Specify Mode Dependent Timing Information
Alias: Specify Mode Dependent Timing Information
Description: **Use case Title/Name:** Specify Mode Dependent Timing Information

Objective/Goal: A function behaves differently in time depending on the present vehicle mode. The vehicle mode, such as the vehicle is running or parked, determines the active states of software components and power states of ECUs and networks. It hence has a great impact on the timing performance of the vehicle functions.
Developers specify the timing characterization for each running mode of the application.

Description: The mode has an impact on the state of software component, the OS task schedule, and the network schedule. One may need to specify the timing property of the functions for each mode. For best performance, it is even preferable to find the optimal task and bus schedules for each mode.

When the mode needs to be changed, the change event or change request must be propagated to the related components via the network. To maintain global mode consistency and high performance, the mode manager must arbitrate the mode switch requests, decide the proper target mode, and respond to the affected components. This mode request-decision-reply process must be bound by a deadline. As a side-effect, this process may also significantly increase the transient bus traffic.

Implied TADL Support and Relevance to TIMMO-2-USE
This use case is related to the following work packages.
WP2 (Language)
Structural extensions: Mode-dependent timing descriptions; Mode-dependent descriptions on task and bus schedules at the implementation level; Timing constraints on mode management operations, etc.
Algorithm-specific extensions: Mode-dependent bus scheduling parameters; Requirements on mode management.

WP3 (Algorithms & Tools)
Methods and tools to obtain good task and bus schedules based on modes.
Methods and tools to manage the mode consistency and optimize system performance.

WP4 (Methodology)

Collaboration on the mode-dependent function distributed to multiple suppliers.

Actors: Function owners, function developers

Stakeholder*:* Function owner and developer

Status: Approved
Author: Daniel Karlsson
Explanation: None
Type: UseCase
Relations:

BOSCH#0005 - Mode dependent timing requirements for control applications
BOSCH#0006 - Mode dependencies
VTEC#0008
CAG#0025 - Safety (timing)
VTEC#0007
CAG#0002 - Event chains between LoA
Transform Timing Information from Analysis Level to Design Level
Develop Cruise Control following Top-down Approach
Transform Timing Information from Design Level to Implementation Level
Perform Timing Analysis On Code-Level (ABS#UC0002)
Develop Engine Management System On Implementation (AUTOSAR) Level
VTEC#0010 - Methodology support for mode-aware design
VTEC#0009 - Method and tool support for mode-dependent bus scheduling
CAG#0001 - Events between LoA

2.1.3 UC#0003 - Change Existing Timing Information

Name: UC#0003 - Change Existing Timing Information
Alias: Effective Change Management
Description: **Use Case Title/Name:** Change Existing Timing Information

Objective/Goal: Product development is mostly about modifying or improving an existing system with new functionality. Even for new product development, the process consists of several iterations with a lot of modifications between iterations. It is therefore of crucial importance to establish efficient change management on timing information.

Description: This use case discusses the change management process from the time perspective. Relevant issues to consider are
What other parts of the system (functions/features, ECUs, busses) are affected by a certain change in timing characteristics?
How can suppliers be notified and timing budgets/contracts most conveniently are negotiated again?
When should one notify a change to others and who should receive the notification? Note that notification/change request implies additional cost.
How can one capture several design alternatives in the same model?

Implied TADL Support and Relevance to TIMMO-2-USE
The following work packages are related to this use case.
WP2 (Language)
Methodological extensions: Cost estimates and other information needed for the

change process.
Semantical reasoning.
Specification of timing properties by expressions.

WP3 (Algorithms & Tools)
Improved timing information exchange between tools and stakeholders.
Tool support for storing and comparing several design alternatives.
Impact of change of timing information for different abstraction levels.

WP4 (Methodology)
An effective collaboration process for managing the change request and maintain the system consistency.

Actors: Function owners, function developers, system integrators, system testers

Stakeholders: OEM and function developers

Status: Approved
Author: Daniel Karlsson
Explanation: None
Type: UseCase
Relations:

VTEC#0012
CAG#0037 - EAST-ADL XML
INRIA#0003 - Timing properties
Develop Cruise Control following Top-down Approach
Generate Test bench for Non-functional Properties
Develop Engine Management System On Implementation (AUTOSAR) Level
VTEC#0015 - Methodology support for change management
VTEC#0013
VTEC#0014 - Tool support for comparing alternative timing solutions
VTEC#0011

2.1.4 UC#0004 - Negotiate Time Budgets

Name: UC#0004 - Negotiate Time Budgets
Alias: Iterative Design Process
Description: **Use Case Title/Name:** Negotiate Time Budgets

Objective/Goal: The development process of vehicle electronic systems is always iterative. Even when a completely new architecture is being developed, different functions are added at different times. Consequently, developers and function owners must keep on negotiating time budgets iteratively.

Description: This use case is closely related to the use case "Change Existing Timing Information". The emphasis is on the double way communication between the developers of different functions. The introduction or modification on one function requires negotiation and compromise with other related functions.

Handling timing requirements along such iterative design processes needs to be

addressed in a systematic way, for example when deciding a time budget for the different functions, one must anticipate the uncertainties imposed by future functions that may affect the overall time aspects of the system.

Actors: Function owners, function developers, system integrators

Stakeholders*:* Function owners and developers.

Status: Approved
Author: Alejandro Cortes
Explanation: None
Type: UseCase
Relations:

VTEC#0012
CAG#0020 - Revising timing constraints
CAG#0037 - EAST-ADL XML
Develop Engine Management System On Implementation (AUTOSAR) Level
Develop Cruise Control following Top-down Approach
VTEC#0005
VTEC#0017
VTEC#0018
CAG#0038 - Timing Analyses
CAG#0021 - Virtual integration (timing)
VTEC#0004 - Timing budget negotiation between OEM and supplier
VTEC#0013
VTEC#0016
VTEC#0011

2.1.5 UC#0005 - Develop Control Applications

Name: UC#0005 - Develop Control Applications
Alias: UC#0005
Description: **Use Case Title/Name:** Develop Control Applications

Objective/Goal: Developers of automotive control programs use TADL to specify both continuous time characterizations of the abstract controller and the discrete-time characterization of the implementation.

Description: In control engineering, the controller is usually designed using continuous or discrete time methods without considering the implementation and final deployment. In real implementation, various delays caused by computation time, resource contention, communication, and so on, may violate timing constraints and deteriorate the control performance.

Consequently TADL should be able to describe the timing requirements of the original controller and maintain the traceability between the controller and its implementation. For the original controller, TADL should support the description of its timing properties, e.g., settling time, rise time, allowable sampling period, etc. To account for the inevitable delays caused by implementation, the allowable delays within the control loop should also be captured in the TADL model. These high-level timing requirements on the control application will be converted to the timing requirements on the implementation components. Such information includes, for instance, WCET, computation deadline, maximal end-

to-end delay, etc.

A high-level control design can be decomposed to individual software components in many ways and the components can be allocated to the ECUs in different ways. The decomposition and allocation significantly influence the timing performance of the control application. While subject to practical constraints, the possible combinations may still be numerous. It is an interesting topic for TIMMO2 to study the algorithm for choosing the optimal combination.

Actors: Function owners, control engineers, software developers, hardware developers

Stakeholders: OEM; Supplier of the control application

Status: Approved
Author: Lei Feng
Explanation: None
Type: UseCase
Relations:

INRIA#0002 - Time bases
VTEC#0020
BOSCH#0006 - Mode dependencies
BOSCH#0005 - Mode dependent timing requirements for control applications
VTEC#0026
INRIA#0005 - Executable models
Control Scheduling Co-Design with Fixed Rates
Derive Timing Requirements from Closed-Loop Algorithms
Develop Engine Management System On Implementation (AUTOSAR) Level
Transform Timing Information from Design Level to Implementation Level
Control Scheduling Co-Design with Flexible Timing Structure
Transform Continuous Time Model to Discrete Time Model
Integrate Several Control Applications on a Target Platform
Exchange Timing Information between Control Engineer and Software Engineer
Develop Cruise Control following Top-down Approach
Derive Timing Requirements from Open-Loop Control Algorithms
Explore Design Alternatives for Control Applications
VTEC#0003 - Methods for estimating WCET at analysis and design levels.
VTEC#0022
VTEC#0024
VTEC#0023
VTEC#0025
VTEC#0019
INRIA#0004 - Functional time
INRIA#0001 - Multiform concepts of Time
VTEC#0027
VTEC#0021
CAG#0039 - Sequence Constraint

2.1.6 UC#0006 - Specify Variability and Timing Information

Name: UC#0006 - Specify Variability and Timing Information
Alias: UC#0006
Description: **Use Case Title/Name:** Specify Variability and Timing Information

Objective/Goal: Variability is an important source for complexity in automotive systems because it leads to a very large number of possible combinations and therefore becomes difficult to handle.

Description: Variability on timing specifications can arise at different abstraction levels. At vehicle level, vehicle configurations are typically defined, each configuration being defined as the features or functions available for the end customer in that particular vehicle configuration. Knowledge on the possible vehicle configurations is often exploited to devise smart design solutions: not all the vehicle functions are present in a given vehicle configuration (that is, no vehicle will be manufactured with all the functions that are possible in that type of vehicle, the end customer cannot freely choose whatever combination, but there are a number of pre-defined vehicle configurations). This implies that it is possible to design the system in such a way that, when considering all the functions, the time budget exceeds 100%, yet the time constraints are fulfilled, simply because we know that there exist vehicle configurations that put constraints on which functions are present on the same vehicle.

It is therefore important to take into account variability, for instance how to capture timing information at the very high levels of abstraction (vehicle level) knowing that vehicle configurations do influence timing at lower levels. For instance, a system has commonly several variants of a specific functionality. This can be implemented as that one or more components are replaceable. The overall timing requirements must be met for each variant and support for specification and analysis at all abstraction levels is necessary.

Actors: Function owners, system architects, function developers

Stakeholders: Function developers; system engineer responsible for integration.

Status: Approved
Author: Henrik Lönn
Explanation: None
Type: UseCase
Relations:

VTEC#0029
VTEC#0028
Develop Engine Management System On Implementation (AUTOSAR) Level
Develop Cruise Control following Top-down Approach
CAG#0036 - Variability
VTEC#0030
VTEC#0031

2.1.7 UC#0007 - Develop Application and Infrastructure

Name: UC#0007 - Develop Application and Infrastructure
Alias: UC#0007
Description: **Use Case Title/Name:** Develop Application and Infrastructure

Objective/Goal: Developers separately design and implement the applications and the infrastructure. The separation is an effective way to manage the

complexity and to enable the easy re-allocation of applications to ECUs. TADL model reflects the separation and also describes the binding effect of the two.

Description: The challenge for this process is to capture timing aspects while keeping the separation of application and infrastructure. For example, the end-to-end latency of an event chain depends on both the application (e.g. the control algorithm) and the infrastructure (e.g. the target hardware). We need a smooth way to bind the application-specific timing information and the infrastructure-specific timing information.

Actors: System architects, software developers, hardware developers, system integrators

Stakeholders: System architects and system integrators; Vendors of platform and/or middleware.

Status: Approved
Author: Lönn Henrik
Explanation: None
Type: UseCase
Relations:

INRIA#0002 - Time bases
Develop Engine Management System On Implementation (AUTOSAR) Level
Develop Cruise Control following Top-down Approach
VTEC#0036
VTEC#0037
VTEC#0032
VTEC#0034
INRIA#0004 - Functional time
VTEC#0033 - Methods for timing characterization of behavior/algorithm
VTEC#0035 - Methods for timing characterization of hardware
INRIA#0001 - Multiform concepts of Time
CAG#0039 - Sequence Constraint

2.1.8 UC#0008 - Exchange Models

Name: UC#0008 - Exchange Models
Alias: UC#0008
Description: **Use Case Title/Name:** Exchange Models

Objective/Goal: Engineers are able to exchange models between different tools.

Description: TADL language works as the universal intermediate format. The timing information in other model formalisms can be extracted and automatically transformed into the TADL model and the TADL model can be transformed into other model formalisms for analysis and testing. The transformation must be done in such a way that the existing timing specifications are preserved.

Actors: System architects, function developers, system integrators, system testers

Stakeholders: System integrators.

Comment: This use case is closely related to UC#0001 and UC#0008.

Status: Approved

Author: Thomas Söderqvist

Explanation: None

Type: UseCase

Relations:

CAG#0004 - Synchronization constraint on ports
CAG#0037 - EAST-ADL XML
CAG#0015 - Assumptions on target systems
CAG#0032 - HW/SW Co-design (Language)
CAG#0003 - Age constraint on port
VTEC#0040
INRIA#0005 - Executable models
CAG#0029 - Exchange a component
Develop Cruise Control following Top-down Approach
Develop Engine Management System On Implementation (AUTOSAR) Level
VTEC#0038
CAG#0031 - HW/SW Co-design (Methodology)
VTEC#0039

2.1.9 UC#0009 - Perform Post-Build Parameterization

Name: UC#0009 - Perform Post-Build Parameterization

Alias: UC#0009

Description: **Use Case Title/Name:** Perform Post-Build Parameterization

Object/Goal: The developer must fill in a large number of system parameters for implementation, and these parameters such as the size of transmit and receive buffers and the configuration of network affect the timing performance. It is however not easy to choose the right values and is also a tedious job to manually fill in them. TIMMO-2-USE should provide tools for conveniently setting up system parameters.

Description: With the post-build feature, AUTOSAR allows parameterization on all levels of implementation, including functional features, routing tables, ECU and network characteristics. TIMMO-2-USE should study how to model these parameters using TADL. The optimal values are better determined by a dedicated tool and the values in the model can be automatically transferred to the AUTOSAR development tool with minimal human effort.

Because of the large number of parameters and their profound effect, TIMMO-2-USE can set limitations on what level of parameters TADL should handle. Allowing post-build for all possible parameters is unrealistic. A modest objective is to identify the parameters in the model and allow their values to be automatically transferred to the AUTOSAR implementation. The algorithm and tool to decide the optimal values of these parameters can be investigated by TIMMO-2-USE.

Actors: Function developers, system integrators

Stakeholders: System integrators.

Status: Approved
Author: Robert Karlsson
Explanation: None
Type: UseCase
Relations:
Perform FlexRay Simulation
VTEC#0043
VTEC#0042
VTEC#0013
VTEC#0041

2.1.10 UC#0010 - Specify Synchronization Timing Constraints

Name: UC#0010 - Specify Synchronization Timing Constraints
Alias: UC#0010
Description: **Use Case Title/Name:** Specify Synchronization Timing Constraints

Objective/Goal: An application may have synchronization requirements on the arrival time or age of multiple events from distinct sources and routes. Failure of the synchronization requirement may jeopardize the function of the application.

Description: A typical application with this synchronization requirement is the Electronic Stability Control (ESC). ESC continuously monitors the slipping conditions of the wheels. The signals from all wheels must represent the conditions of the wheels at the same time. Owing to the disturbance in the ECU and the network, one or more wheel slip signals might be delayed and the time synchronization is not preserved at ESC. The consequence of this is that the stability actions are not the optimal. In addition, along the other signal flow direction, the actuation signals from ESC to the wheels must also be synchronized. TADL must support the engineer to specify and analyze synchronization timing constraints to prevent such problems.

Actors: Control engineers, function developers, system testers

Stakeholders: Function developers.
Status: Approved
Author: Thomas Söderqvist
Explanation: None
Type: UseCase
Relations:
CAG#0004 - Synchronization constraint on ports
INRIA#0003 - Timing properties
Develop Cruise Control following Top-down Approach
Perform FlexRay Simulation
Develop Engine Management System On Implementation (AUTOSAR) Level
VTEC#0044
VTEC#0046
CAG#0027 - Synchronization constraint per runnable entity
VTEC#0045
INRIA#0001 - Multiform concepts of Time
CAG#0039 - Sequence Constraint

2.1.11 UC#0011 - Specify Probabilistic Timing Properties

Name: UC#0011 - Specify Probabilistic Timing Properties
Alias: UC#0011
Description: **Use Case Title/Name:** Specify Probabilistic Timing Properties

Objective/Goal: Timing properties and constraints may not be deterministic. They can be given as probabilistic values with distribution functions. Stakeholders need to describe and analyze systems with such timing properties.

Description: Probabilistic timing properties are often given for and even preferred by soft real-time applications, where certain amounts of constraint violations are acceptable.

Deterministic timing properties, e.g. WCET and deadline, are critical for hard real-time systems; however, the majority of the applications are soft, i.e., certain amount of constraint violations are acceptable. Timing properties of these soft real-time applications may be given as probabilistic values of certain distribution functions. The safety constraints need only to be guaranteed with a probability. This relaxed safety requirement admits tremendous flexibility to stakeholders.

TADL shall allow developers to describe such probabilistic timing properties of events and event chains. The safety constraints of the system should then be associated to probabilities. For example, the end-to-end delay of an event chain must be smaller than 10 ms in 99% of the cases.

Methods and tools for analyzing timing properties must be adapted. For example, the schedulability test cannot only return true or false. The answer should be the probability of the schedulability.

Development methodology must be adapted to allow the new type of specifications and analysis.

Actors: Function owners, function developers, system testers

Stakeholders: OEMs, suppliers, and all function developers.
Status: Approved
Author: Lei Feng
Explanation: None
Type: UseCase
Relations:

VTEC#0047
TUBS#0002 - Uncertain parameters
TUBS#0003
VTEC#0049
TUBS#0004 - Obtain uncertain timing information
VTEC#0048
TUBS#0001 - Uncertainty

2.2 Specific Use Cases

This section describes the Specific Use Cases identified by the work package 1.

2.2.1 Capture, Analyze, and Utilize Statistical Timing Information

Name: Capture, Analyze, and Utilize Statistical Timing Information
Alias:
Description: **Objective/Goal**: Utilize statistical timing information during the development process

Description: By and large, existing systems are altered in order to introduce new functionality respectively changing existing functionality, rather than developing system from the scratch with new functionality. Due to this, a lot of statistical data - from similar or previous iterations of the development - is available on the temporal characteristic of a system (execution times, occurrences of events, response times) even for different platforms (execution units).
When a system is going to be altered this statistical data can be utilized in order to predict - to a certain degree - the possible deviation of the temporal characteristics from a given one.

This use case addresses the following topics:

- How to capture the statistical timing information (methodology) and how to describe this information in a formal way (language)?
- How to analyze the captured statistical timing information (methodology) and what statements can be made on the results of this analysis?
- How and what conclusions can be drawn from the statistical timing information in order to utilize them in different phases of the development process (methodology)?

Actors: Timing Analyst(s), Timing Expert(s)

Stakeholders: VFM Architects, FAA Architects, [FDA | HDA | MWA] Architects, [VFB | System | ECU | Component] Architects

For more details about the mentioned roles refer to the TIMMO deliverable D7.

Originator: SymtaVISION
Status: Approved
Author: Kai Richter
Explanation: None
Type: UseCase
Relations:

2.2.2 Capture, Analyze, and Utilize Worst Case Timing Information

Name: Capture, Analyze, and Utilize Worst Case Timing Information

Alias:

Description: **Objective/Goal**: Utilize worst case timing information during the development process

Description: ...

This use case addresses the following topics:

- How to capture the worst case timing information (methodology) and how to describe this information in a formal way (language)?
- How to analyze the captured worst case timing information (methodology) and what statements can be made on the results of this analysis?
- How and what conclusions can be drawn from the worst case timing information in order to utilize them in different phases of the development process (methodology)?

Actors: Timing Analyst(s), Timing Expert(s)

Stakeholders: VFM Architects, FAA Architects, [FDA | HDA | MWA] Architects, [VFB | System | ECU | Component] Architects

For more details about the mentioned roles refer to the TIMMO deliverable D7.

Originator: SymtaVISION

Status: Approved

Author: Kai Richter

Explanation: None

Type: UseCase

Relations:

2.2.3 Derive Timing Requirements from Closed-Loop Algorithms

Name: Derive Timing Requirements from Closed-Loop Algorithms

Alias:

Description: **Objective/Goal**: Obtain timing requirements from the analysis of closed-loop control algorithms

Description: The use case describes the steps to be taken in order to derive timing information, in particular timing requirements, from the analysis of open-loop control algorithms. These timing requirements shall support the software developer to make the proper design decisions to satisfy the given timing requirements.

Actors: Function developer

Stakeholders: Software developer

Originator: Chalmers University

Status: Approved

Author: Stefan Kuntz

Explanation: None

Type: UseCase

Relations:

BOSCH#0007 - Explicit and implicit events
BOSCH#0002 - Solution dependent and solution independent timing requirements
BOSCH#0011 - Derivation of discrete timing requirements
CAG#0007 - Use of SystemC
CAG#0006 - Obtain timing information (closed-loop)
Transform Continuous Time Model to Discrete Time Model
BOSCH#0008 - Concepts of Time
BOSCH#0001 - Control Timing Requirements
BOSCH#0009 - Specification of events in the continuous environment

2.2.4 Derive Timing Requirements from Open-Loop Control Algorithms

Name: Derive Timing Requirements from Open-Loop Control Algorithms
Alias:
Description: **Objective/Goal**: Obtain timing requirements from the analysis of open-loop control algorithms

Description: The use case describes the steps to be taken in order to derive timing information, in particular timing requirements, from the analysis of open-loop control algorithms. These timing requirements shall support the software developer to make the proper design decisions to satisfy the given timing requirements.

Actors: Function developer

Stakeholders: Software Developer

Originator: Chalmers University
Status: Approved
Author: Stefan Kuntz
Explanation: None
Type: UseCase
Relations:

BOSCH#0002 - Solution dependent and solution independent timing requirements
BOSCH#0007 - Explicit and implicit events
BOSCH#0011 - Derivation of discrete timing requirements
CAG#0007 - Use of SystemC
BOSCH#0008 - Concepts of Time
BOSCH#0001 - Control Timing Requirements
Transform Continuous Time Model to Discrete Time Model
BOSCH#0009 - Specification of events in the continuous environment

2.2.5 Develop Body Controller following Top-down Approach

Name: Develop Body Controller following Top-down Approach

Alias:

Description: **Objective/Goal**: The goal of the use case is to derive the timing requirements of a body controller from the requirements specification until the control application level.

Description: Specific scenarios of a real-world case study will be used to validate the results from WP2 to WP4 on the use of timing information across different abstraction levels respectively development phases, with particular emphasis on efficient collaboration between system, software and control engineers.

This use case will address the following topics, among others:

- How to derive (transform) the timing requirements following the EAST-ADL methodology and how to describe this information in a formal way (language)?
- How to analyze and validate the timing requirements in models?
- How to enable co-engineering between system, software and control engineers (exchange information between tools)?
- How to address the safety aspect?

Actors: System Architect, Function Architect, Software Designer, Control engineer, Timing Analyst

Stakeholders: System integrators, Supplier of the control application

Originator: Delphi

Status: Approved

Author: Kamel Maaziz

Explanation: None

Type: UseCase

Relations:

2.2.6 Develop Cruise Control following Top-down Approach

Name: Develop Cruise Control following Top-down Approach

Alias:

Description: At the moment no description is available, because the work package 1 could not draw a conclusion on the objective/goal of this use case. The decision has been postpone to the beginning of the work packages 2 through 5.

Objective/Goal: ...

Description: ...

Actors: ...

Stakeholders: ...

Originator: Continental Automotive

Status: Approved

Author: Stefan Kuntz

Explanation: None
Type: UseCase
Relations:

CAG#0025 - Safety (timing)
CAG#0011 - Time bases relation
CAG#0010 - Time bases
CAG#0009 - Scheduling Analysis
CAG#0035 - Task synthesis
CAG#0038 - Timing Analyses
CAG#0039 - Sequence Constraint

2.2.7 Develop Engine Management System on Implementation (AUTOSAR) Level

Name: Develop Engine Management System On Implementation (AUTOSAR) Level
Alias:
Description: **Objective/Goal**: Validate the results of work package 2 through 4 in the context of developing a combustion engine management systems and/or specific scenarios of this development

Description:
#1: Validation
The use case shall demonstrate how the results from work package 2 through 4 are applied respectively utilized in the software/hardware development on the EAST-ADL implementation level (AUTOSAR) and assesses their applicability.
In order to obtain reasonable results a case study is conducted using a real-world example Combustion Engine Management System.
During the course of the TIMMO-2-USE project possible results, specifically from work package 2, 3, and 4) are validated using this example and the applicability of these results are assessed and demonstrated.

Note: Indeed, the use case does not cover the entire development of a combustion engine management system, but identifies specific scenarios within this development where the application of the results are obvious and leads to an improvement of the development.

#2: Timing Requirements/Timing Constraints and Timing Properties
When developing according to the EAST-ADL methodology the timing requirements that shall be considered during the Implementation Phase (AUTOSAR) are given as a result from the Design Phase in the first place. Since AUTOSAR provides different views on the software system (Virtual Function Bus, System, ECU, and Component) the question to be answered is how the timing requirements are reflected in the various AUTOSAR timing views and how these are handled in subsequent steps during the implementation phase.

Actors: [Architect | Designer | Implementer | Tester | Integrator | Timing Analyst | Timing Expert] [VFB | System | ECU | Component]
For more details about the mentioned roles refer to the TIMMO deliverable D7.

Stakeholders: Customer

Originator: Continental Automotive
Status: Approved
Author: Stefan Kuntz
Explanation: None
Type: UseCase
Relations:

CAG#0008 - Multi-Core
CAG#0025 - Safety (timing)
CAG#0011 - Time bases relation
CAG#0014 - Composability of runnable entities
CAG#0022 - Transition from DL to IL
CAG#0024 - Multi-Core (Scheduling Analysis)
CAG#0010 - Time bases
CAG#0035 - Task synthesis
CAG#0009 - Scheduling Analysis
CAG#0038 - Timing Analyses
CAG#0016 - Use of AUTOSAR timing views
CAG#0027 - Synchronization constraint per runnable entity
CAG#0005 - Hardware
CAG#0026 - Age constraint per runnable entity
CAG#0039 - Sequence Constraint

2.2.8 Exchange Timing Information between Control Engineer and Software Engineer

Name: Exchange Timing Information between Control Engineer and Software Engineer
Alias:
Description: **Objective/Goal**: The goal of this use case is to enable control engineers and software engineers to exchange timing information in order to perform co-engineering. This includes in particular extending the TIMMO-2-USE TADL.

Description: In order to perform the co-engineering use cases described in Exploration of Design Alternatives for Control Applications, Control Scheduling Co-Design with Fixed Rates, Integration of Several Control Applications on the Target Platform and Control Scheduling Co-Design with Flexible Timing Structure the control engineer and the software engineer need a means for exchanging timing information. The concrete information that shall be exchanged has to be determined by these use cases.

Actors: Control Engineer, Function Developer, Software Engineer

Stakeholders: OEM and supplier
Status: Approved
Author: Stefan Kuntz, Arne Hamann
Explanation: None
Type: UseCase
Relations:

BOSCH#0003 - Tracing of control timing requirements
BOSCH#0007 - Explicit and implicit events
Integrate Several Control Applications on a Target Platform
Explore Design Alternatives for Control Applications

BOSCH#0004 - Collaborative Engineering of Control Applications
BOSCH#0008 - Concepts of Time
BOSCH#0001 - Control Timing Requirements
BOSCH#0009 - Specification of events in the continuous environment

2.2.9 Exchange Timing Information with worst case verification tool

Name: Exchange Timing Information with worst case verification tool
Alias:
Description: **Objective/Goal**: Bidirectional exchange of timing information between the tools that are used to design or to describe a system and the tools that are used to verify timing constraints based on worst-case analysis.

Description: Several specialized tools are used for different purposes at different steps of the development process. In the descending branch of the V-development cycle, timing constraints can be verified through worst-analysis at several levels (functional design, implementation). In order to be able to perform the analysis, the description of the system and the constraints to be verified need to be exported from some system description tool to the specialized verification tool. And the results of the analysis need to be exported back to the architecture description tool for tracing purposes.

Actors: System Architect, Function Architect, Software Designer

Stakeholders: Automotive OEMs, Suppliers

Originator: RealTime-at-Work

Status: Approved
Author: Jörn Migge
Explanation: None
Type: UseCase
Relations:

2.2.10 Exploit Uncertain Timing Information

Name: Exploit Uncertain Timing Information
Alias:
Description: **Objective/Goal:** The goal of this specific use case is to propose methods for analysis at component level and system level (if possible) in presence of uncertainty.

Description: Traditional scheduling algorithms and analysis methods (e.g. for processor utilization or response time), provide deterministic timing guarantees (i.e., all task instances meet their deadline) which take into account worst-case scenarios that may be very rare in practice. This is needed in hard real-time systems but too restrictive for soft real-time systems --- and even for some hard real-time systems where the application allows for a given failure rate (e.g. the

probability of missing a deadline could be as small as the probability of hardware failure). One solution to this issue is to analyze the system under uncertainty, while ensuring that the deadline miss ratios predicted by the approximated analysis are greater than (or equal to) the real ones.

Actors: To be defined during the course of work package 2 through 5.

Stakeholder: To be defined during the course of work package 2 through 5.

Originator: TU Braunschweig
Status: Proposed
Author: Sophie Quinton
Explanation: None
Type: UseCase
Relations:
TUBS#0002 - Uncertain parameters
TUBS#0003
TUBS#0004 - Obtain uncertain timing information
TUBS#0001 - Uncertainty

2.2.11 Explore Design Alternatives for Control Applications

Name: Explore Design Alternatives for Control Applications
Alias:
Description: **Objective/Goal**: The goal of this requirement is to enable the design team to explore alternative software realization for a given control task at hand.

Description: Computer-based control theory is based on equidistant sampling and negligible input-output latencies that can be ignored. However, in reality execution times vary due to preemption, blocking, data-dependencies, caches, pipelines, network communication, etc. This results in sampling interval jitter as well as non-negligible and varying latencies.
To solve this problem, software solutions for control tasks must be co-engineered between control and software engineers.
The solution space for the software realization of a given control task is vast. Thereby, the chosen solution influences on the one hand the control performance, and on the other hand the overall timing performance of the system.

1. The control engineer prefers small sampling and execution rates to achieve close-to-optimal control performance. This, however, leads to high system load, and consequently high system cost (in terms of hardware).
2. The software engineers prefer large sampling and execution rates to increase the composability and extensibility of the system. This, however, leads to decreased control performance.

This conflict of objectives spans the co-design area shown in the picture below. Extending the TIMMO-2-USE TADL and methodology to systematically explore the trade-off between control quality and composability is the main aim of this use case.

Actors: control engineer, function developer, system architect

Stakeholders: OEM and supplier

Status: Approved

Author: Arne Hamann

Explanation: None

Type: UseCase

Relations:

BOSCH#0002 - Solution dependent and solution independent timing requirements
Integrate Several Control Applications on a Target Platform
Control Scheduling Co-Design with Fixed Rates
Control Scheduling Co-Design with Flexible Timing Structure
BOSCH#0004 - Collaborative Engineering of Control Applications
CAG#0007 - Use of SystemC
BOSCH#0010 - Methodology for timing design of control applications

2.2.12 Generate Test bench for Non-functional Properties

Name: Generate Test bench for Non-functional Properties

Alias:

Description: **Objective/Goal**: ...

Description: Test bench with IEEE PSL-Timing specifications transformed from T2U TADL2
Focusing on: one ECU considering AUTOSAR Basic Software Timing and timing of applications
SystemC model architecture (modules) compatible to East-ADL2 architecture
Refinement of SystemC modules corresponding to refinement in East-ADL2 architecture
PSL properties valid on abstract and refined levels of abstraction
Defined on interfaces

Actors: To be defined during the course of work package 2 through 5.

Stakeholders: To be defined during the course of work package 2 through 5.

Originator: UPB

Status: Approved

Author: Kay Klobedanz

Explanation: None

Type: UseCase

Relations:

UPB#0006 - Transformation
UPB#0001 - Abstraction levels
UPB#0019 - Hardware relation
UPB#0022 - Software instruction level
UPB#0012 - Black box behavior

2.2.13 Handle Timing Information in Simulation Based Analysis Activities

Name: Handle Timing Information in Simulation Based Analysis Activities
Alias:
Description: **Objective/Goal**: Describe how timing information is handled in various simulation activities, and how timing information is exchanged with design and implementation tools in a roundtrip development process. It shall be identified which timing information is needed/provided by simulation tools and if the required data can be handled with TADL2 or the AUTOSAR timing extensions.

Description:
Developers perform various simulation activities
… to validate timing information, which has been assumed during the design and implementation phase, and
… to obtain additional information about timing and resource consumption by measurement on the target platform.

The activities include:

- Validation of time budgets (e.g. reaction time) in HIL and offline simulations.
- Comparison of timing behavior with reference simulations.
- Identifying event chains with critical/suspect timing.
- Measuring timing data and resource consumption on the target processor.
- Automated HIL test series.
- Optimization of the timing by step-wise modification of the implementation.
- Visualization of measured timing, for example, in sequence charts with timing annotation.

The tools applied in this activities are:

- Offline and HIL simulators,
- Experimentation and test automation tools,
- Profilers and Debuggers,
- Special tools for analysis and visualization

These tools consume timing data (reference data) and they provide new measured timing data which must be interchanged with other tools in a roundtrip development process.

Actors: Integrator, Tester, Implementer, Timing Analyst

Stakeholders: OEM, Supplier

Originator: dSPACE
Status: Approved
Author: Ulrich Kiffmeier
Explanation: None
Type: UseCase
Relations:

2.2.14 Integrate Re-useable Component

Name: Handle Timing Information in Simulation Based Analysis Activities
Alias:
Description: **Objective/Goal:** A re-usable Software Component is integrated into an existing system

Description: This use case describes how an available AUTOSAR Software Component is integrated into an existing system (VFB), which means that all required steps before the integration, during the integration, and after the integration are described.
In particular, the use case focuses on the timing information required to be present and exchanged between the customer and supplier in the mentioned three steps.

The major topic in this use case is that not only the system, a software component is integrated into, imposes requirements on the software component; but also the software component, to be integrated into a system, imposes requirements on this system. In other words, the use case raises the question how are assumptions taken from the software component's view are described using a language and what kind of assumptions shall be described in order to support: a) the selection of a component, and b) the integration of the software component.

Actors: System Integrator (AUTOSAR Role)

Stakeholders: OEM, First Tier Supplier

Originator: DENSO – Now taken care by dSPACE and INCHRON
Status: Approved
Author: Stefan Kuntz
Explanation: None
Type: UseCase
Relations:

2.2.15 Integrate Several Control Applications on a Target Platform

Name: Integrate Several Control Applications on a Target Platform
Alias:
Description: **Objective/Goal**: The goal of this use case is to extend the TIMMO-2-USE TADL and methodology to support the integration of several control applications on the same hardware platform.

Description: This use-case is closely related to the use case Exploration of Design Alternatives for Control Applications. However, here the focus lies on the integration of several control applications considering there interdependencies. For instance, it might be necessary to "widen" the sampling rate and execution rate of one control application to be able to accommodate another application on the same ECU. Thereby, the same trade-off between control quality and composability has to be considered.

Actors: control engineer, function developer, system architect

Stakeholders: OEM and supplier
Status: Approved
Author: Arne Hamann
Explanation: None
Type: UseCase
Relations:
CAG#0008 - Multi-Core
CAG#0015 - Assumptions on target systems
CAG#0051 - Reuse of timing constraints
CAG#0028 - Integrating a component
CAG#0037 - EAST-ADL XML
BOSCH#0004 - Collaborative Engineering of Control Applications
CAG#0021 - Virtual integration (timing)
BOSCH#0010 - Methodology for timing design of control applications

2.2.16 Perform FlexRay Simulation

Name: Perform FlexRay Simulation
Alias:
Description: **Objective/Goal**: ...

Description: Offline and restbus simulation of FlexRay networks considering timing properties
Offline simulation with SystemC and FlexRay library from UPB
Transformation of timing specification from T2U TADL2 to FIBEX vs. AUTOSAR
Restbus simulation for Steer-by-Wire validator from UPB
Real ECUs communicating over FlexRay bus
Additonal Bus components simulated with SystemC

Actors: To be defined during the course of work package 2 through 5.

Stakeholders: To be defined during the course of work package 2 through 5.

Originator: UPB
Status: Approved
Author: Kay Klobedanz
Explanation: None
Type: UseCase
Relations:
UPB#0017 - Synchronization
UPB#0016 - Network frame modeling
UPB#0021 - Communication simulation
UPB#0005 - Global time base
UPB#0008 - FIBEX Compliance
UPB#0019 - Hardware relation
UPB#0003 - Bus communication
UPB#0020 - Offline simulation

2.2.17 Perform Time-based Offline Simulation

Name: Perform Time-based Offline Simulation
Alias:
Description: At the moment no description is available, because the work package 1 could not draw a conclusion on the objective/goal of this use case. The decision has been postpone to the beginning of the work packages 2 through 5.

Objective/Goal: ...

Description: ...

Actors: ...

Stakeholders: To be defined during the course of work package 2 through 5.

Originator: Chalmers University
Status: Approved
Author: Stefan Kuntz
Explanation: None
Type: UseCase
Relations:
CAG#0010 - Time bases
CAG#0011 - Time bases relation

2.2.18 Perform Time-based Online Simulation

Name: Perform Time-based Online Simulation
Alias:
Description: At the moment no description is available, because the work package 1 could not draw a conclusion on the objective/goal of this use case. The decision has been postpone to the beginning of the work packages 2 through 5.

Objective/Goal: ...

Description: ...

Actors: ...

Stakeholders: ...

Originator: Chalmers University
Status: Approved
Author: Stefan Kuntz
Explanation: None
Type: UseCase
Relations:

2.2.19 Perform Timing Analysis on Code-Level

Name: Perform Timing Analysis On Code-Level

Alias:

Description: **Objective/Goal:** Determine WCET of non-interrupted tasks as input for system-level verification of timing properties.

Description: Deterministic timing properties, e.g. WCET and deadline, are critical for hard real-time systems.
WCET can be determined by timing tools **aiT** and **TimingExplorer** by AbsInt on the **Implementation Level**.

Timing analysis is performed on compiled executables for concrete hardware. For reliable and not too pessimistic results the WCET analysis needs a number of configuration parameters, which can be grouped as
hardware parameters (cache, ECU configuration, etc) - for Implementation Level only
software parameters (loop bounds, etc) - for Implementation Level only
system-level parameters (software modes) - for Design, Analysis and Implementation Levels

Some of parameters are optional and are used for better precision, others are mandatory.

Relation to Work Packages
This use case is related to the following work packages.

WP2 (Language)
Extensions of TADL for specification of system-level parameters like modes.

WP3 (Algorithms & Tools)
Identify hardware parameters that are missing in AUTOSAR and create a separate work package if needed.
Identify software parameters that should be provided by software developers.
Identify system-level parameters that should be specified by TADL.
Perform integration to other tools to get necessary parameters. Integration needs agreement on exchange formats. There are exchange formats already available. Agree on extensions needed to these exchange formats.

WP4 (Methodology)
Due to necessity of getting potentially confidential information on software implementation for WCET analysis, discuss the probable interaction between software developers and system integrators/testers.

Actors: system integrators, software developers, system testers

Stakeholders: system integrators, software developers, system testers

Status: Approved

Author: Olha Honcharova

Explanation: None

Type: UseCase

Relations:

ABS#UC0002 - Perform Timing Analysis On Code-Level
ABS#0010 - Improving precision of WCET analysis by additional parameters
ABS#0008 - Function Pointers for WCET analysis
ABS#0006 - Loop Bounds for WCET analysis
ABS#0003 - Executable for WCET analysis
ABS#0009 - Volatile Variables for WCET analysis
ABS#0004 - Mapping to Source Code for WCET analysis
ABS#0011 - Supported Processor for WCET analysis
ABS#0001 - Timing Analysis in Implementation Phase
ABS#0007 - Recursion Bounds for WCET analysis
ABS#0012 - Processor Configuration for WCET analysis
ABS#0005 - Analysis Start Point for WCET analysis
ABS#0013 - Processor-Specific Settings for WCET analysis

2.2.20 Process Timing Information for HIL-based Simulation

Name: Process Timing Information for HIL-based Simulation
Alias:
Description: At the moment no description is available, because the work package 1 could not draw a conclusion on the objective/goal of this use case. The decision has been postpone to the beginning of the work packages 2 through 5

Objective/Goal: ...

Description: ...

Actors: ...

Stakeholders: ...

Originator: University Paderborn
Status: Approved
Author: Stefan Kuntz
Explanation: None
Type: UseCase
Relations:

2.2.21 Process Timing Information for SIL-based Simulation

Name: Process Timing Information for SIL-based Simulation
Alias:
Description: At the moment no description is available, because the work package 1 could not draw a conclusion on the objective/goal of this use case. The decision has been postpone to the beginning of the work packages 2 through 5

Objective/Goal: ...

Description: ...

Actors: ...

Stakeholders: ...

Originator: University Paderborn
Status: Approved
Author: Stefan Kuntz
Explanation: None
Type: UseCase
Relations:

2.2.22 Specify End-to-End Latencies

Name: Specify End-to-End Latencies
Alias:
Description: **Objective/Goal**: Use multi-form time to express end-to-end latencies

Description: The TIMMO project only considered end-to-end latency values given in the unit of seconds [s]. For example, given data shall be processed within 500 ms. However, in many cases it is more convenient to express timing requirements in physical units, like "… when the temperature increased by 10° C." or "… at position 275° of the crankshaft …", " … the vehicle shall stop motion after 50 m.", etc.
The use case described the steps to be taken to state such timing requirements and what additional information is required to put it into the specific context. Furthermore, it describes how the mult-form time is translated respectively transformed into a time measured in seconds.

Actors: To be defined.

Stakeholders: Requirements Engineer

Originator: INRIA
Status: Approved
Author: Stefan Kuntz
Explanation: None
Type: UseCase
Relations:
INRIA#0002 - Time bases
INRIA#0005 - Executable models
INRIA#0003 - Timing properties
INRIA#0004 - Functional time
INRIA#0001 - Multiform concepts of Time

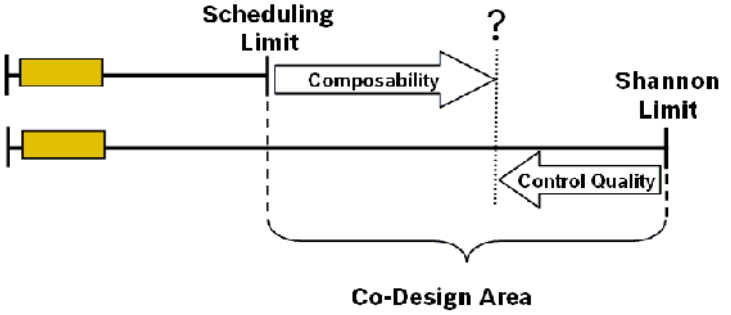
2.2.23 Transform Continuous Time Model to Discrete Time Model

Name: Transform Continuous Time Model to Discrete Time Model

Alias:

Description: **Objective/Goal**: The goal of this use case is to extend the TIMMO-2-USE TADL and methodology to support the transformation step between a continuous time model and a discrete time model.

Description: Typically, the control engineer first devices a continuous solution for a given control task. Then, in order to prepare a discrete software solution, the control engineer chooses a sampling rate and a discretization method (based on information including the analytical time constants of the plant, and the Shannon threshold frequency of the plant). Note that due to these choices several timing requirements can be derived for the discrete software solution. The software engineer and the system integrator then prepare a software solution and test it in the overall system.
This use case is the precondition to enable round-trip engineering between the control engineer and the software engineer.
For instance, in case of timing constraints violations or poor control quality, the control engineer can choose a different sampling rate or discretization method.



Actors: Control Engineer, Function Developer, Software Developer, System Integrator

Stakeholders: OEM and supplier

Status: Approved

Author: Stefan Kuntz, Arne Hamann

Explanation: None

Type: UseCase

Relations:

BOSCH#0003 - Tracing of control timing requirements
BOSCH#0002 - Solution dependent and solution independent timing requirements
BOSCH#0007 - Explicit and implicit events
Explore Design Alternatives for Control Applications
BOSCH#0011 - Derivation of discrete timing requirements
CAG#0030 - Distribute jitter
BOSCH#0010 - Methodology for timing design of control applications
BOSCH#0001 - Control Timing Requirements
BOSCH#0008 - Concepts of Time
BOSCH#0009 - Specification of events in the continuous environment

2.2.24 Transform Timing Information from Vehicle Level to Analysis Level

Name: Transform Timing Information from Vehicle Level to Analysis Level
Alias:
Description: **Objective/Goal**: Transform Vehicle Timing Requirements into Analysis Timing Requirements

Description: During the Analysis Phase the given Vehicle Timing Requirements, besides other functional and non-functional requirements, are the basis for taking decision to create the Functional Analysis Architecture FAA. The primary goal is to satisfy the Vehicle Timing Requirements and in addition to determine the important/relevant timing properties of this "design" - FAA. These timing properties are then transformed into the Analysis Timing Requirements which are a work product passed to the Design Phase.
The purpose of the Analysis Phase is to realize the features specified in the Vehicle Phase and to determine which Analysis Functions are required and how they shall inter-operate in order to realize these features. The [external] visible behavior of every Analysis Function is described, as well as their temporal characteristics.
It is important what kind/type of timing information describes the temporal characteristics of the Functional Analysis Architecture and how the timing information is transformed into timing requirements to be considered in the next phase.

Actors: FAA Architect, FAA Designer, FAA Implementer, Timing Analyst, Timing Expert
For more details about the mentioned roles refer to the TIMMO deliverable D7.

Stakeholders: VFM Architect, VFM Timing Expert

Originator: Continental Automotive
Status: Approved
Author: Stefan Kuntz
Explanation: None
Type: UseCase
Relations:

2.2.25 Transform Timing Information from Analysis Level to Design Level

Name: Transform Timing Information from Analysis Level to Design Level
Alias:
Description: **Objective/Goal**: Transform Analysis Timing Requirements into Design Timing Requirements

Description: During the Design Phase the given Analysis Timing Requirements, besides other functional and non-functional requirements, are the basis for taking decision to create the Functional Design Architecture FDA, Hardware Design Architecture HDA, and Middleware Architecture MWA. The primary goal is to satisfy the Analysis Timing Requirements and in addition to determine the important/relevant timing properties of these "designs" - FDA, HDA, MWA.

These timing properties are then transformed into the Design Timing Requirements which are a work product passed to the Implementation Phase. The Design Timing Requirements describe the temporal characteristic of the Functional Design Architecture FDA, Hardware Design Architecture HDA, and Middleware Architecture MWA.
The purpose of the Design Phase is to realize the Analysis Functions specified in the Analysis Phase and to determine which Design Functions, Hardware Elements, and Middleware Services are required and how they shall inter-operate in order to realize the Analysis Functions. The internal [and external] behavior of every Design Function is described, as well as their temporal characteristics.
It is important what kind/type of timing information describes the temporal characteristics of the Functional Design Architecture, Hardware Design Architecture, and Middleware Architecture; and how the timing information is transformed into timing requirements to be considered in the next phase.

In the Design Phase possible distributions of Design Functions in a given system topology are explored.

Actors: [FDA | HDA | MWA] Architect, [FDA | HDA | MWA] Designer, [FDA | HDA | MWA] Implementer, Timing Analyst, Timing Expert
For more details about the mentioned roles refer to the TIMMO deliverable D7.

Stakeholders: FAA Architect, FAA Timing Expert

Originator: Continental Automotive

Status: Approved
Author: Stefan Kuntz
Explanation: None
Type: UseCase
Relations:

CAG#0032 - HW/SW Co-design (Language)
CAG#0025 - Safety (timing)
CAG#0002 - Event chains between LoA
CAG#0012 - Semantics of event chains (component)
CAG#0013 - Semantics of event chains (connector)
CAG#0023 - Transition from AL to DL
CAG#0009 - Scheduling Analysis
CAG#0030 - Distribute jitter
CAG#0031 - HW/SW Co-design (Methodology)
CAG#0001 - Events between LoA
CAG#0005 - Hardware
CAG#0034 - Automation

2.2.26 Transform Timing Information from Design Level to Implementation Level

Name: Transform Timing Information from Design Level to Implementation Level
Alias:
Description: **Objective/Goal**: Transform Design Timing Requirements into Implementation Timing Requirements

Description: The very first steps of the Implementation Phase is to transform the Functional Design Architecture, Hardware Design Architecture, and Middleware (functional view) into the corresponding software and hardware architecture - represented by the following views in AUTOSAR:
Virtual Function Bus View
Software Component View and Basic Software Module view
System Topology and ECU Resource Descriptions

During this transformation the given Design Timing Requirements, besides other functional and non-functional requirements, are the basis for taking decision how to transform/map the elements of the functional domain into the software/hardware domain.
The primary goal is to satisfy the Design Timing Requirements and in addition to determine the important/relevant timing properties of the various AUTOSAR views.

Since there are a lot of different ways in developing the various AUTOSAR views, this use case focuses on the transformation from the Design Level to the AUTOSAR level and generating the VFB view and the System Topology, only (See also EDONA project http://www.edona.fr).

Actors: [VFB | System | ECU | Component] [Architect | Designer | Implementer], [VFB | System | ECU | Component] Timing Analyst, [VFB | System | ECU | Component] Timing Expert
For more details about the mentioned roles refer to the TIMMO deliverable D7.

Stakeholders: [FDA | HDA | MWA] Architect, [FDA | HDA | MWA] Timing Expert

Originator: Continental Automotive
Status: Approved
Author: Stefan Kuntz
Explanation: None
Type: UseCase
Relations:

CAG#0032 - HW/SW Co-design (Language)
CAG#0025 - Safety (timing)
CAG#0002 - Event chains between LoA
CAG#0012 - Semantics of event chains (component)
CAG#0013 - Semantics of event chains (connector)
CAG#0022 - Transition from DL to IL
CAG#0009 - Scheduling Analysis
CAG#0030 - Distribute jitter
CAG#0031 - HW/SW Co-design (Methodology)
CAG#0005 - Hardware
CAG#0001 - Events between LoA
CAG#0034 - Automation

2.2.27 Verify Timing Constraints

Name: Verify Timing Constraints
Alias:

Description: **Objective/Goal**: Verify timing constraints based on system models

Description: In the descending branch of the V development cycle, the system to be designed (and implemented) is described by models in an increasingly detailed manner. Based on the system models, timing constraints can be verified through probabilistic and/or worst-case timing analysis. The confidence in the verification and the precision of the analysis increases with the increasing knowledge of system details. On the one hand, only rough estimates of execution times are available at function design level and thus only rough estimates of response time bounds can be obtained by the analysis at that level. On the other hand, very precise WCET can be obtained when the implementation code is available and thus very precise response time bounds can be computed.

Actors: System Architect, Function Architect, Software Designer

Stakeholders: Automotive OEMs, Suppliers

Originator: RealTime-at-Work

Status: Approved

Author: Jörn Migge

Explanation: None

Type: UseCase

Relations:

CAG#0038 - Timing Analyses

3 References

[1] TIMMO Deliverable D7, Methodology Version 2, Version 1.1, 2009-10-01, The TIMMO Consortium.