Test Front Loading in Early Stages of Automotive Software Development Based on AUTOSAR

Alexander Michailidis, Uwe Spieth, Thomas Ringler, and Bernd Hedenetz
Group Research & Advanced Engineering
Daimler AG, Sindelfingen, Germany
{alexander.michailidis, uwe.spieth, thomas.ringler, bernd.hedenetz}@daimler.com

Stefan Kowalewski
Chair of Computer Science 11
RWTH Aachen University
Aachen, Germany
kowalewski@embedded.rwth-aachen.de

Abstract—Embedded software development has become one of the greatest challenges in the automotive domain, due to the rising complexity of vehicle systems. A method to handle the complexity of automotive software is Model Based Design (MBD). As MBD offers great advantages in early simulation and testing, it has become today’s mainstream method for automotive software engineering. However, some aspects can be initially tested after the integration of software on real hardware components (usually by the supplier) and when all parts of a system (e.g. bus systems, sensors, actuators) are present. The consequence is that the requirement specification of the according system possibly contains gaps that can lead to software defects. New technologies like the AUTOSAR standard enable additional potentials for the validation of model based developed software. Due to the AUTOSAR software architecture it is possible for an OEM to realize an early „virtual“ software integration with an acceptable effort and perform at next step a front loading of system tests. In this paper we present an approach that improves the quality of the requirement specification artifacts by using test front loading. In detail, we analyze the requirement engineering part of the software development process to identify aspects that can not be tested without having all system components. Afterwards, we classify these aspects and define an abstract test pattern that can be globally used for testing. Additionally, we illustrate our approach in a case study on an interior light system for the next Mercedes-Benz M-Class generation.

Keywords—model based design; validation and test; AUTOSAR; virtual integration; front loading

I. INTRODUCTION

The application of software in the automobile has drastically increased over the last few years and represents increasingly a set of criteria for the ability of a car manufacturer to compete. According to a McKinsey study [1] automotive software furthers approximately 80% of all future innovations and its validation is just as important as the hardware. In addition to the growing software amount, the complexity of the vehicle systems is also increasing. Instead of isolated functionality on separate electronic control units (ECUs), distributed systems located on several ECUs with a high degree of interaction are introduced.

Important prerequisites for an accurate and efficient development for such electronic systems are defined development processes as well as the use of practice orientated powerful tools [2]. In order to meet the short development time, a change in the development of the ECU-software of vehicle systems has for several years been taking place. This is characterized via a transition from the conventional software development (via hand coding) to model based development methods. In model based development, the behaviour of the diverse software contributions of a system is described by graphic models. The use of models makes the transition from the textual system specifications to the design of the ECU software considerably easier. The model in this case receives the roll of a ‘executable specification,’ which serves as a common understanding of different groups of people in the system development. In addition a simulation of the software behaviour is made possible via the feasibility of the model. Therefore, in a very early stage of the development, gaps and errors in the system specification can be recognized and unnecessary iterations in the development process can be avoided.

Up to now, the model based software contributions of a system have been predominantly developed and tested separately from one another. The system integration and thus the observation of the interaction of the contributions both among themselves and also with the hardware specific basic software on the ECUs is taking place late in the development process. Previous projects have demonstrated that in the integration of model based developed software, integration errors could not have been avoided to the same extent unlike the case for the functional errors [3]. For the consistent continuation of the model based approach a “virtual integration” of the designed software based on system models is needed. The setup of such models was until now not target leading due to the heterogeneous ECU basic software and the limited availability of hardware at this stage. With the introduction of AUTOSAR (Automotive Open System Architecture) [4], the design of system models with a justifiable effort is possible for the first time. The standard AUTOSAR software architecture and the decoupling of the application software from the hardware via a specific abstraction layer provide the basis. Due to the new standard, a market of AUTOSAR tools [5], [6] is developing which provides the design and simulation mechanisms for system models. The necessary prerequisites are therefore given in order to carry out further analysis from the perspective of an OEM and to identify optimisation potential in the system development process.

In the present paper, an approach is being developed which focuses on the early improvement of the maturity degree of

---

1 Electronic systems are composed of software and hardware parts which together realize a specific functionality in the vehicle.
system specifications and the resulting models of system software contributions. The necessary principles for the new approach are explained in the following section. There is an overview of current processes, technologies, and concepts. Subsequently, the relevant aspects for a front loading of tests on the system design stage are demonstrated. These include a detailed analysis of the system requirements engineering process part for the identification of scenarios which are being tested lately, and the creation of abstract test-patterns for the generalisation of the identified scenarios. With a case study, the main ideas will be illustrated. Finally, there will be a conclusion and an outlook on further activities.

It should be noted that the approach elaborated here focuses on the model based system development of the body domain at Daimler AG. Therefore only a section of the software development for embedded vehicle systems can be covered. This is however quite representative according to our estimation.

II. BASIC PRINCIPLES

A. Model Based Design at Daimler

At Daimler AG model based development in particular in the area of body and comfort is already being applied for some years. For the current C/E-class software contributions are implemented for approximately fifteen systems (e.g. interior lights, exterior lights, power network management). Since then, a model library is being build up and rolled out in other vehicle lines [3].

The procedure for model based system development is represented in Fig. 1 as a V-Model. For all system software contributions such as interior light control, exterior light control etc. independent models are designed (with Matlab/Simulink [7]) based on a textual system specification and are systematically tested on the basis of a detailed test plan (model in loop, MIL). After a successful test, the models will be released for implementation and ECU integration. The models, together with the system specification, the test plan and the implemented model tests, will be handed over to the supplier. The supplier is responsible for the model implementation, code generation, software module tests (software in loop, SIL) and integration of the software modules in the ECUs together with basic software. Afterwards, the ECU with the integrated software will be handed back to Daimler AG where the component tests (component hardware in loop, C-HIL) system tests (system hardware in loop, S-HIL) and finally the integration into the vehicle with the corresponding vehicle integration tests will be accomplished.

For the functional tests of the different development stages the single source principle applies, this means only one artefact (the textual system specification) is used as reference. Thus, the test cases of all development stages are early available and can therefore be universally used [3].

B. Methods for ModelTesting

In the context of model development, there are different types of test methods for early validation of the software being developed. The focus thereby is on the dynamic tests for the functional validation of the models. The basis for such tests is the execution of the software to be tested (software under test, SUT) with systematically defined input data (test cases). For each test case, the input as well as the expected output data is specified. The output data generated from the test run is compared with the respective expected data. In case of a deviation, an error has occurred.

For the functional validation on model level via dynamic test methods, intensive research work has been carried out in the past few years at Daimler AG. These include among others the development of the classification tree method (CTM/ES) [8] and time partition testing (TPT) [9], the design of model based tests [10], the analysis of structural coverage criteria on model level [11] and the automated test analysis within back-to-back tests [12]. In the area of body and comfort, the TPT method is applied to the series development due to its good properties for the testing of reactive systems with continuous behaviour. In section IV the TPT method is further elaborated.

C. AUTOSAR Standard

The series development demonstrated that for a successful realisation of the development process described above and for successful software integration, project specific coordination between the OEM and the respective suppliers regarding the software interfaces, the software architecture, and the development process were necessary. With the publication of the AUTOSAR standard [4], the essential elements for the smooth integration of vehicle systems over OEMs and suppliers have been defined and are now available as a basis for the development of new vehicles.

AUTOSAR defines a standardised software architecture for ECUs, an integration process and the necessary exchange formats. The software architecture is distinguished in application and basic software parts which communicate with each other via a middleware (the RTE - Real Time Environment) (see Fig. 2). The application software (the set of system software contributions on an ECU) is thereby divided into multiple software components (SWCs). SWCs encapsulate and characterise the software and allow a data exchange only via well defined interfaces. For this purpose, ports are created that are dedicated for an interface.

At Daimler AG the first steps towards the AUTOSAR-architecture for the next ECU generation in the area of body and comfort are being carried out. The existing models of the system software contributions have been restructured and enhanced by AUTOSAR interfaces. The functionality which
was previously realized in a hand coded frame software layer\textsuperscript{2} (e.g. signal conditioning) was included in the models. The standard software architecture is still based on the established Daimler standard core \textsuperscript{13} and is amended by selected AUTOSAR software services such as NVRAM\textsuperscript{1}, ECU-State- and COM-Management. For the development of the next Mercedes S-Class generation, the implementation of a complete AUTOSAR basic software stack is planned \textsuperscript{14}.

Apart from the adaptation of the behavioural models for the system software contributions, a formal description of the interfaces by means of SWCs and their interconnection is according to AUTOSAR required for the construction of a vehicle system. In case of distributed systems, communication artifacts (K-matrices) also have to be considered. Therefore, a central database is being maintained for some time at Daimler AG. In this database, interface and data type definitions are stored based on AUTOSAR and can be used for the definition of SWC-ports \textsuperscript{14}.

D. Virtual Integration Concept

The AUTOSAR conform model library which was developed at Daimler AG and the formalized AUTOSAR system artifacts are already available in early development stages and build the basic elements for a virtual software integration. The C-code generated out of the implementation models defines the system behaviour as it will be later on realized in the real vehicle.

In order to be able to virtually integrate the generated code, an appropriate tool is necessary in which the AUTOSAR-artifacts can be imported, configured, and eventually supplemented. The tool must handle the communication and scheduling mechanism which AUTOSAR provides and should generate the RTE layer automatically. In addition, a simulation on a PC has to be possible to allow a validation of the specified system models. With the introduction of AUTOSAR such tools are available which meet these requirements \textsuperscript{5}.

Through a virtual integration in the early development stages, there is a set of use cases that are becoming interesting. The main use cases are from the OEM perspective:

(1) Validation of distributed or neighbouring systems which have functionally related SWCs on several ECUs. In particular the functional interaction of the SWCs can be validated and communication properties within defined delay tolerance limits can be analyzed.

(2) Validation of functional aspects of SWCs as well as aspects regarding the access to NVRAM and the change of operating modes (start-up, shutdown, etc.). Therefore, the necessary basic software services have to be identified and implemented according to the AUTOSAR specification \textsuperscript{15}.

III. REQUIREMENTS ENGINEERING ANALYSIS

Due to the concept of virtual integration, it is possible to further increase the degree of maturity of system specifications and model based SWCs in early development stages. For system validation in reference to the use cases defined above, additional tests are necessary in the early design stages which up to now could not be considered in the validation of individual SWCs. In order to determine such tests, an analysis of the requirements of existing system specifications was carried out. Requirements which define implicit relationships between SWCs themselves and the basic software on the ECUs were thereby especially investigated. Implicit relationships are general preconditions for the system behaviour which in the early development stages are not yet complete and are therefore not always specified as requirements. These implicit relationships often give freedom for erroneous interpretations at the following development phases.

A. Global Facts

As an assumption for the analysis of system specifications, some basic facts are presented first:

(1) System specifications comprise functional and non functional requirements which describe the behaviour of a system. The functional requirements are mostly signal oriented and specify the technical implementation. Signal orientated requirements describe the reaction behaviour of a system which should result from a value change of the system input signals.

(2) SWCs of a system are described in detail by individual requirements. The requirements however do not always describe signals for individual SWCs but also signals that are composed of several SWCs.

(3) There are general requirements which must apply to an entire system. In addition, the requirements of neighbouring systems must also apply (e.g. conditions such as vehicle and clamp states which must apply for the activation of neighbouring systems).

(4) To check a SWC against the specification, at least one test per requirement will be carried out.

(5) Implicit relationships may result by requirements mentioned in (2) and (3). These relationships can not be tested in (4) since no requirements are defined.

B. Scenarios for System Validation

Via the analysis of the system requirements, a number of scenarios were developed which describe implicit relationships and are currently difficult to check in the early stages of system
specification. The scenarios were classified by means of the characteristics to be checked, as follows:

1. SWCs and communication interfaces are specified by different developers. Thus, there are different versions of the interface descriptions regarding development time which do not always have to be simultaneously present on all SWCs. To check the descriptions for equality, a static test can be carried out via AUTOSAR. However, the handling of values by the SWC behaviour is not checked (e.g. for subsequent extensions of SWC interfaces).

2. The ECUs are not only dependent among themselves via bus communication but are also dependent from each other via hardware relationships. Individual ECUs can create appropriate function conditions for other ECUs, e.g. controlling of standby-current switches. Therefore, it is necessary to examine whether all required SWCs for a system are in the same system state and whether not-defined states can arise.

3. ECUs are often powered in vehicle idle state and are in sleep state. Via hardware or software initiated wake-up methods they can switch into active state. By examining the wake-up functionality in the distributed system, it can be shown that the system can be correctly enabled or disabled and that it does not come to an undesired communication during sleep states.

4. Individual SWCs are partially dependent on data from neighbouring SWCs in order to represent the desired functionality. It must be validated whether the intended functionality is reliably being carried out or terminated even at non-defined communication states. Specifically, the execution of SWCs during the state change of a neighbouring SWC can be observed.

5. During inactive vehicle phases, the environmental conditions can change. However the data stored in the NVRAM of the ECU is preserved. Here is to be validated whether the system allows no misinterpretation.

6. A system which is distributed over several ECUs may not be able to guarantee a strict synchronicity and therefore fluctuate between being activated and deactivated with a time delay. The bus communication will also possess delay times. The additional system states that arise from these delays can be checked for the desired behaviour.

IV. TEST PATTERN FOR FRONT LOADING

In order to describe the acquired scenarios in coherence with the test method that is deployed in the area of body and comfort at Daimler AG, the test setup topic will be discussed in this section. For that purpose the TPT test method and the test scheme (pattern) which is currently being used are briefly described and finally the necessary adaptations of the pattern for the front loading in the development process are presented.

A. Test Pattern for TPT

The description of the tests is carried out by the TPT method using a special state machine notation via which the state space of the SUT-behaviour over time can be striped into different partitions. The general pattern for the modelling and analysis of the TPT tests is represented in Fig. 3 (for a simple system with two states). The states of the state machine

![Figure 3. Test pattern for the validation of SWCs](image)

annotate actions through which the modification of the input values of the SUT is carried out. The transition from one state to another is triggered by a condition (guard). Mostly, time conditions will be used for this purpose (e.g. wait for 10s). The analysis of the output values of the SUT is limited to the corresponding time intercepts (checkpoints) in which the software behaviour which is triggered by a specific action can be checked. In order to be able to check the results of the test run automatically, the expected software properties in these time intercepts have to be specified. (e.g. due to python assessment scripts).

B. Adaptation of TPT Pattern for System Tests

Predominantly, only functional aspects were considered during the testing of individual SWCs of a system. For the front loading respectively to the identified scenarios additional aspects have to be comprised. This includes interface aspects for the composition of SWCs, control flow aspects for the interaction of the SWCs with the basic software and network aspects for the communication of distributed systems. For the implementation of these aspects the test pattern which was previously applied is being adapted.

Interface-aspects: The validation of the interfaces in a composition of AUTOSAR-SWCs is being carried out as discussed in section III.B via a static check. For that purpose, the interfaces and data elements of the connected ports are checked for consistency. However, the handling of data element values through the SWC behaviour cannot be validated by the static testing. A dynamic examination via a simulation of the SWC composition is thus necessary. For the simulation the functional test cases are sufficient. Therefore, no extension of the test pattern is necessary at this point.

Control flow aspects: For the interaction of SWCs with the ECU basic software, it is necessary to include parts of the basic software in the simulation model (see Fig. 4). The aim is to abstract as much as possible and to take into account only the necessary AUTOSAR basic software modules. Hence, the configuration effort of the modules can be kept small. For the selection of the necessary AUTOSAR modules regarding control flow aspects an analysis was performed in [15]. Via this analysis the AUTOSAR services: NVRAM-, COM-, and ECU-State-Manager were identified. In order to control the simulation with the identified services so that the desired test can be run through, a stimulation of these services is partially necessary. For the NVRAM-Management, the evaluation of the
NV-values on the corresponding SWC interfaces is sufficient. However, the COM- and ECU-State-Manager have to be stimulated in order to simulate the activation/deactivation of the bus communication as well as the start-up/shut-down of the ECUs. Therefore, to the main state machine of the primary test pattern two additional test state machines are added (ECU and communication state machine) which are concurrently triggered by the actions of the main state machine via state requests (see Fig. 5). The ECU state machine determines the current state of the ECU and passes it through to the SWCs via the simulated AUTOSAR-ECU-State-Manager. For this purpose each SWC has a special sender receiver port according to the AUTOSAR specification. The COM state machine determines whether or not the sending and receiving of messages in a network at any given time is possible. The signalling of the communication states to the SWCs is carried out via so-called AUTOSAR-RTE-Status values which give the receiver status of the SWC data elements whose transmission takes place via a bus. The generation of these status values occurs by the simulated AUTOSAR-COM-Manager.

**Network aspects:** Due to the inclusion of communication properties in the simulation, not only time spots but also specific time intervals $\Delta t$ (delay tolerance boundaries) are considered at the test assessment. The specification of these intervals can either be specified in the system specification or be determined via the simulation. In the second case the new finding can act as feedback for the system specification.

V. **EXPERIMENTAL EVALUATION**

For the illustration of the adapted test pattern, an interior light system model for the next Mercedes M-Class generation was created based on the virtual integration concept introduced in Section II.D. The interior light system model is comprised of four SWCs which are distributed on three different body ECUs and communicate with one another via a CAN and a LIN bus system (see Fig. 6). The ILC (interior light control) acts here as the master for the two clients (client front, ICF / client rear, ICR) and the overhead control (OHC).

The modelled interior light system was tested using diverse simulation runs according to the scenarios worked out in Section III.B. Of great interest is the run in Fig. 7 which particularly addresses scenarios (3), (4), and (6). In this test run, cyclic activation requests of the master component ILC (top diagram), lead to the computation of nominal PWM values (in percent) for the foot well light (ICF) and the dimmable ambient light (ICR) in the vehicle (bottom diagram). In the middle diagram the change of the ECU (0=off, 1=start-up, 2=run, 3=shutdown) and communication states (0=nocom, 1=com) is shown. It can be observed that the expected system reaction takes place after the ECU is started up and the communication is activated (time 3s). Thereby, the ILC and the ICF are integrated onto the same ECU, the reaction for the foot well light takes place at the same time as the activation request. For the ambient light however there is a delay of 100ms due to the transmission of the ILC output data over a CAN bus. This delay represents the duration of one cycle run since the ILC output data will be read and processed by the ICF one cycle run later. In case of communication bursts the transmission time will increase so that in a worst case the delay can be 200ms or even more.

The assessment of the simulation run follows through a Python script. Such a script is represented in Fig. 8 for the ambient light functionality of the interior light system. It is checked whether the system behaves respective to the ECU and the COM states as specified. In addition, the time differences are analysed at the system response, i.e. whether the response time is within the specified delay tolerance boundaries ($\Delta t$).
beMinMfutureMnecessaryMtoMconsiderMsuchMscenariosMearlierMviaMaMdevelopmentMatMcomponentMandMsystemMHIL. DueMtoMtheMrelationshipsMhaveMonlyMbeenMinvestigatedMinMtheMlaterMstagesMofMalsoMwithMtheMbasicMsoftwareMonMtheMECUms. UpMtoMnowMsuchMsimulationMmodelMdcribedMhere,MthisMisMpossibleMalreadyMinMtheMuntilNowMwereMfirstlyMcheckedMinMtheMsystemMHIL.MThroughMtheMneededMSoFarMtheMdesignMofMsuchModelsMwasMnotMtargetMfrontMloadingMinMtheMdevelopmentMprocess.MForMtheMfrontMloadingModels,MweMintroducedManMapproachMthatMformallyMdescribesMtheMtimeMavailable.MInMorderMtoMbeMableMtoMtestMAUTOSARMsystemM“virtual”MvehicleMsystemsMwithMreasonableMeffortMareMforMtheFirstMtheMnecessaryMandMtoolsMforMdesignMandMsimulationMofMdevelopment.MWithMtheMpublicationMofMtheMAUTOSARMstandard,M

The simulation run represented here describes typical situations where system specification errors can occur which until now were firstly checked in the system HIL. Through the simulation model described here, this is possible already in the system design stage.

VI. CONCLUSIONS AND FUTURE WORK

In this study, scenarios for the model test were represented from the system point of view. These scenarios describe implicit relationships of SWCs both among themselves and also with the basic software on the ECUs. Up to now such relationships have only been investigated in the later stages of development at component and system HIL. Due to the increasing complexity of the vehicle electronic systems, it will be in future necessary to consider such scenarios earlier via a front loading in the development process. For the front loading of system tests, the set-up of corresponding system models is needed. So far, the design of such models was not target leading due to the great variety at the ECU software development. With the publication of the AUTOSAR standard, the necessary artefacts and tools for design and simulation of “virtual” vehicle systems with reasonable effort are for the first time available. In order to be able to test AUTOSAR system models, we introduced an approach that formally describes the represented scenarios for system model tests. For that purpose, a pattern based on the TPT method was defined and accordingly adapted. Finally, the adapted pattern was deployed in a case study for the testing of interior light system model.

Due to the presented approach, model tests can be reinforced and thus the maturity degree of system specifications and SWC models can be increased in the early stages of development. In a next step the application of the approach is planned in system series development. Thereby, our approach can be validated parallel to the established development process with respect to the interaction of the several process stages and groups of people.

Finally, it must be emphasised that there are still aspects that can only be tested with the availability of real hardware and other physical conditions. Therefore, only a partial front loading can take place. The later component and system HIL tests can therefore not be omitted.

REFERENCES