MAENAD

Grant Agreement 260057

Model-based Analysis & Engineering of Novel Architectures for Dependable Electric Vehicles

Report type Deliverable D6.1.2

Report name Case study model and specification

Dissemination level	**Public**
Status	**Intermediate**
Version number	**2.0.1**
Date of preparation	**2012-08-31**

Authors

Editor	E-mail
Stefano Cerchio	stefano.cerchio@crf.it

Authors	E-mail
Stefano Cerchio	stefano.cerchio@crf.it
Sandra Torchiaro	sandra.torchiaro@crf.it
Dejiu Chen	chen@md.kth.se
Frank Hagl	frank.hagl@continental-corporation.com
Henrik Lönn	henrik.lonn@volvo.com
Birgit Rösel	birgit.roesel@continental-corporation.com

The Consortium

Volvo Technology Corporation (S)	4SG(I)	Centro Ricerche Fiat (I)
Continental Automotive (D)	Delphi/Mecel (S)	CEA LIST (F)
MCO (SF)	Systemite (S)	PAR (F)
Kungliga Tekniska Högskolan (S)	Technische Universität Berlin (D)	University of Hull (GB)

Revision chart and history log

Version	Date	Reason
0.1	2011-03-08	First internal realease
0.2	2011-05	Minor changes
0.3	2011-06-15	Introduction updated
1.0	2011-06-30	Intermediate Release
1.0.1	2011-08-30	BBW Diagrams updated
1.0.2	2012-06-18	Integrated new inputs for “Range evaluation” and BBW model
1.0.4	2012-06-18	Fixed references errors – Document review
1.0.5	2012-06-29	Propulsion model updated
2.0	2012-06-30	Second release corresponding to M21 status
2.0.1	2012-08-31	Minor adjustments for P2

List of abbreviations

Table of terms and abbreviations used in this document

Abbreviation	Description
EVC	Electric Vehicle Controller
FEV	Fully Electric Vehicles
HVJB	High Voltage Junction Box
PE	Power Electronic

Table of contents

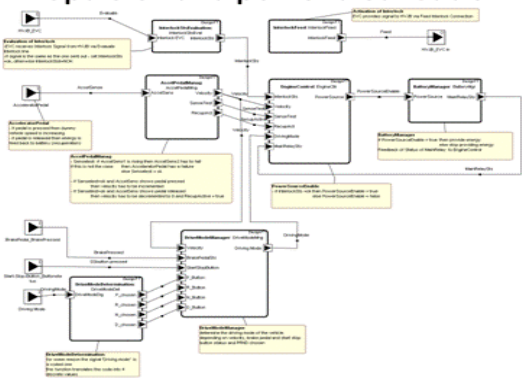
List of figures

1 Introduction

The main goal of this document is to describe all the modelling aspects that have been performed within WP6 on the selected case studies, and a preliminary overview about the steps carried out towards the analysis of the project outcomes.

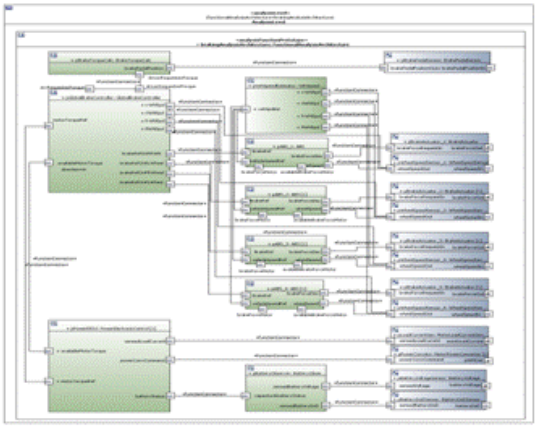
To meet the project objective, three different case studies related to Full Electric Vehicle application have been proposed to exercise the modelling aspect, modelling techniques and analysis framework

Propulsion and power distribution



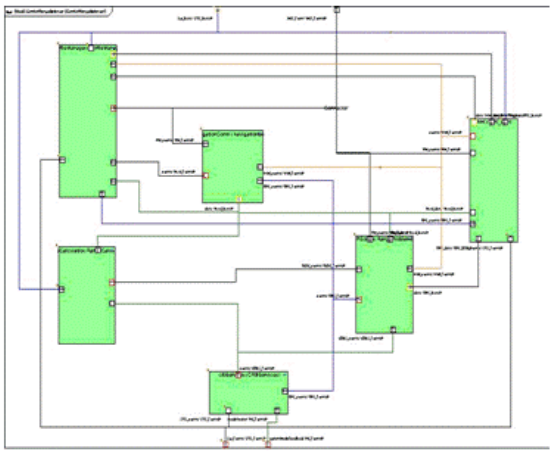
power and signal distribution subset of a FEV with the associated interlock functionality for safety features, and the driving mode selection management,

Regenerative Braking



regenerative braking systems based on an innovative brake by wire distributed architecture.

Mode and range management



Driving mode management for electric vehicle, with enhanced power and energy supervision algorithm to support the driver in critical range situation, as well the related HMI to interact with the driver

The availability of different case studies guarantee a major degree of confidence about the completeness of the analysis, with the main goal to demonstrate feasibility and effectiveness of Maenad main artefacts on evaluating key FEV functions and concepts in terms of:
• performance and dependability of design proposals
• compliance with FEV standards and ISO 26262 standards,
• ability to interface with the 14V architecture,
• electrical isolation in accordance with high voltage standards,

1.1 Document Overview

The three case studies have been modelled using the methods and tools developed in the Maenad project.

The Maenad development framework heavily relies on East-ADL modelling language, a domain specific language for the design of automotive electronic architecture that has been settled and enriched in various phases within different European research projects.

In the context of the MAENAD project, the original languages, design methodology and related tools for the development and evaluation of complex automotive architectures further grow to support and capture specifics aspect related to the design of Electric vehicles, while evolving to maintain compatibility with existing commercial tools and design standard.

With this background, the structure of the document reflects and embraces the approach that the modelling languages provide to organize and represent the engineering information related to a particular system

Models are organized in different levels of abstraction, each of which provides a particular view of the entire vehicle embedded system.

At the Vehicle Level, through the Vehicle Features Model, the EE architecture of the vehicle is described in terms of “features” that characterize the vehicle. Features describe the intended functional and non-functional characteristic of the vehicle without giving detail on how they are implemented. The Vehicle Feature Model provides also a mechanism to capture and describe the different “variant“ of a vehicle, supporting the definition of rules for the inclusion of the features on the final product.

The Analysis Level support the design of the EE architecture in term of functions that concur on the realization of the different features captured at the Vehicle Level

In the Design Level, the functional architecture of the vehicle is addressed in detail. This layer of design concern with the Hardware architecture of the vehicle embedded systems, the mapping of functionalities on electronic devices, the definition of constraints related to sensor and actuators, definition of signal data types exchanged between functionalities and time properties.

At the implementation level, the different macro functionalities that concur to the realization of the vehicle features are detailed and mapped to the AUTOSAR software components.

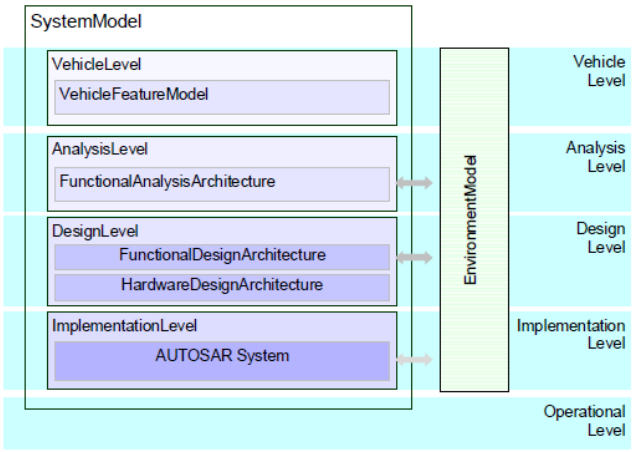


Figure 1-1: EAST-ADL Abstraction levels

2 Case studies modelling

2.1 Propulsion and power distribution

The "*Propulsion and power distribution*" subsystem is a part of the EV Demo Car which is currently under development at Continental. The EV Demo Car shall demonstrate the car's entire potential set of features and functions. Furthermore the capability of Continental, as a leading automotive supplier, to provide a wide range of not only traditional but also innovative components and functions for an EV is a focus point to be demonstrated by means of this Demo Car.

The different systems in the car – tires, brakes and e-propulsion – have to be tightly integrated to achieve best efficiency. The HMI needs to reflect the special requirements of electric vehicles by displaying relevant information and support the user inputs.

The architecture and interfaces of the EV Demo Car system are defined in such a way that the components support best energy efficiency of the vehicle as well as to provide the required information to the driver. Thus the EV Demo Car shall represent a particularly well adapted platform to propose enhanced ergonomic-driven cockpit solutions facing the issues of always increasing complexity (see e.g. Continental's concept "Simplify your Drive"). This depends on the kind of function and on the safety relevance of the function/component.

The new concept of electric vehicle requires adapted system architecture and new system components to match the desired functionality. Not only the combustion engine is changed to electric propulsion but as well new additional functions have to be considered. The EV will be successful in the market if it is easy, simple, fun to drive and affordable in comparison to conventional combustion engines driven vehicles.

The costs issues shall also consider a comparison as complete as possible (environmental issues, inspection and workshop services, costs of operation, insurance, tax – also with respect to regional specificities –, etc…).

For the Propulsion subsystem as part of Maenad some of the newly developed components for the EV Demo Car will be a physical part of the demonstrator – the High Voltage Junction box and the driving mode selector.

The aim of the EV Demo is to show the power distribution and interlock concept as well as the Driving mode selection.

As there is parallel project EV demo car running at Continental this model was not created from scratch but is an adapted part of this whole vehicle model. The aim of this propulsion model is to use the programs and tools developed by the MAENAD consortium for a real model.

That is why first the Hardware design architecture was created with MetaEdit+ 4.5, based on that the functional design architecture was created with the same tool.

2.1.1 Vehicle level

The main feature of a fully electric vehicle (FEV) is the using of high voltage electrical energy for driving, provided by a battery. The High Voltage Junction Box is distributing the energy to different consumers or providers. The main consumer is the drivetrain, consisting of power electronic and e-machine. But there are others as heater or compressor. These consumers are not part of this model. The energy is provided by a charger. There might be different chargers connected to the high voltage junction box. They are not modelled either.

It has to be assured that no one touches high voltage unintentionally. Furthermore it is important to supervise the proper function of all high voltage connections. For this reason the interlock line is established. That is every high voltage connector has two additional contacts which are connected

to each other as long as the connector is plugged in completely. As soon as one connector is released the interlock is opened. When this occurs, the high voltage supply is disconnected immediately.

This function is required to assure that persons do not have contact to the high voltage under all circumstances. Maybe a connector is damaged after an accident. Then the high voltage supply has to be stopped to avoid any further damage of persons. It is dangerous to stop the electrical machine in case the interlock line was opened by mistake. If this happens during a takeover manoeuvre the vehicle will lose driving energy immediately.

As the Electric Vehicle Controller is the main controller for many powertrain functions of an electric vehicle. It provides the information to feed the interlock line to the HVJB. This is done by a dedicated signal on the connection *Feed interlock* between EVC and HVJB. Thus the EVC is the beginning of the interlock line. Furthermore the EVC is then the endpoint of the interlock line – connection *Evaluate interlock* between HVJB and EVC. In the EVC the evaluation of the status of the interlock line is done. As a result of this evaluation the EVC may decide to shut down the high voltage which is done with the connection *PowerSourceEnable* and to stop the torque request from the Power electronic.

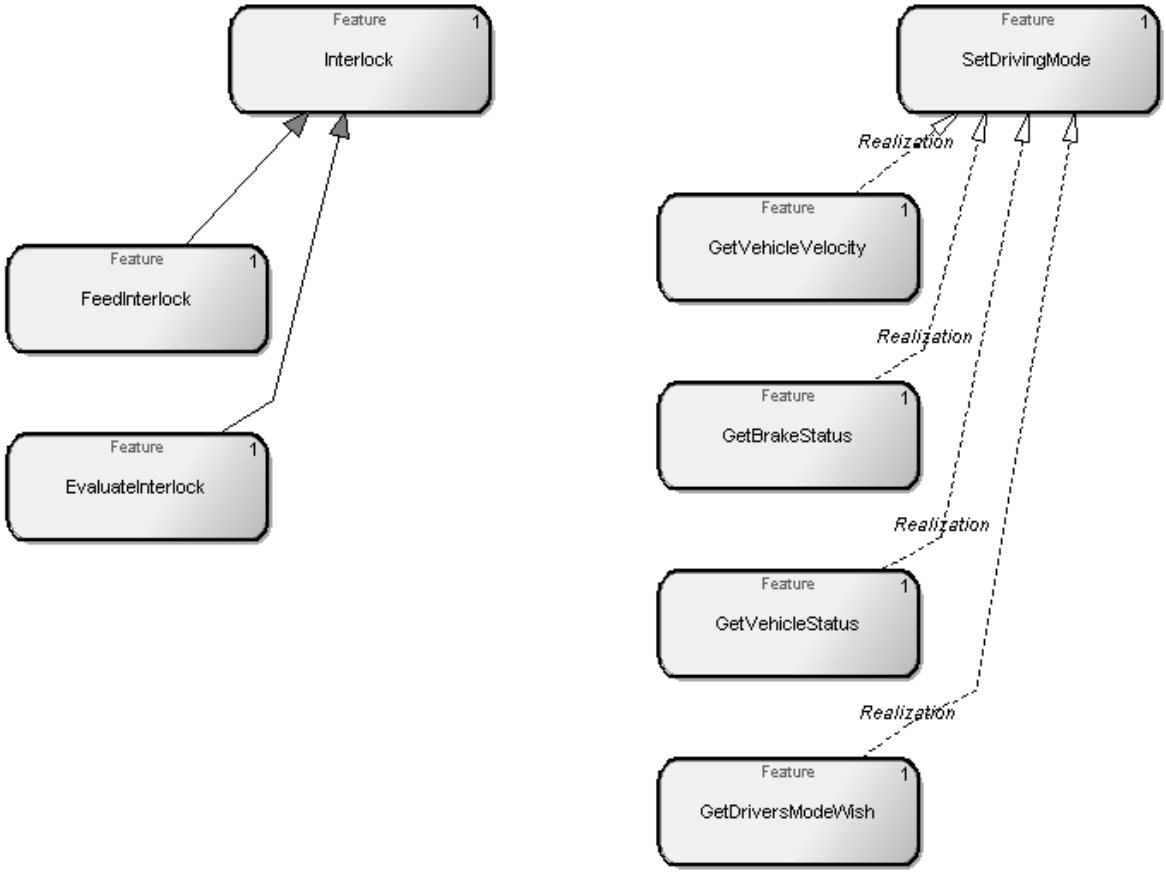


Figure 2-1: Vehicle Feature Model of the "*Propulsion*" subsystem

As an electric vehicle does not need gears for transmission there is no need for a transmission box. But the vehicle has to be able to change direction forward and backward electrically. Furthermore it has to be possible to bring the vehicle in a parking mode. That is why a Driving Mode Selector (PRND) is necessary. To accept a certain indication for a driving mode by the driver the brake signal, the speed and e-propulsion status of the vehicle have to be evaluated.

This two features only are modelled within this model (refer to Figure 2-1).

Interlock feeds the Interlock lines and evaluates the status. If status is ok then the Power source is enabled otherwise the main relays are opened immediately.

The gears PRND are set according to the drivers wish and status (velocity and status of epropulsion and brake)

2.1.2 Design Level

Functional Design Architecture

The following Functional Design Architecture describes one realization of the features explained in chapter 2.1.1

Both features are implemented on the EVC. Please refer to the comment fields within Figure 2-2 for further details of each function.

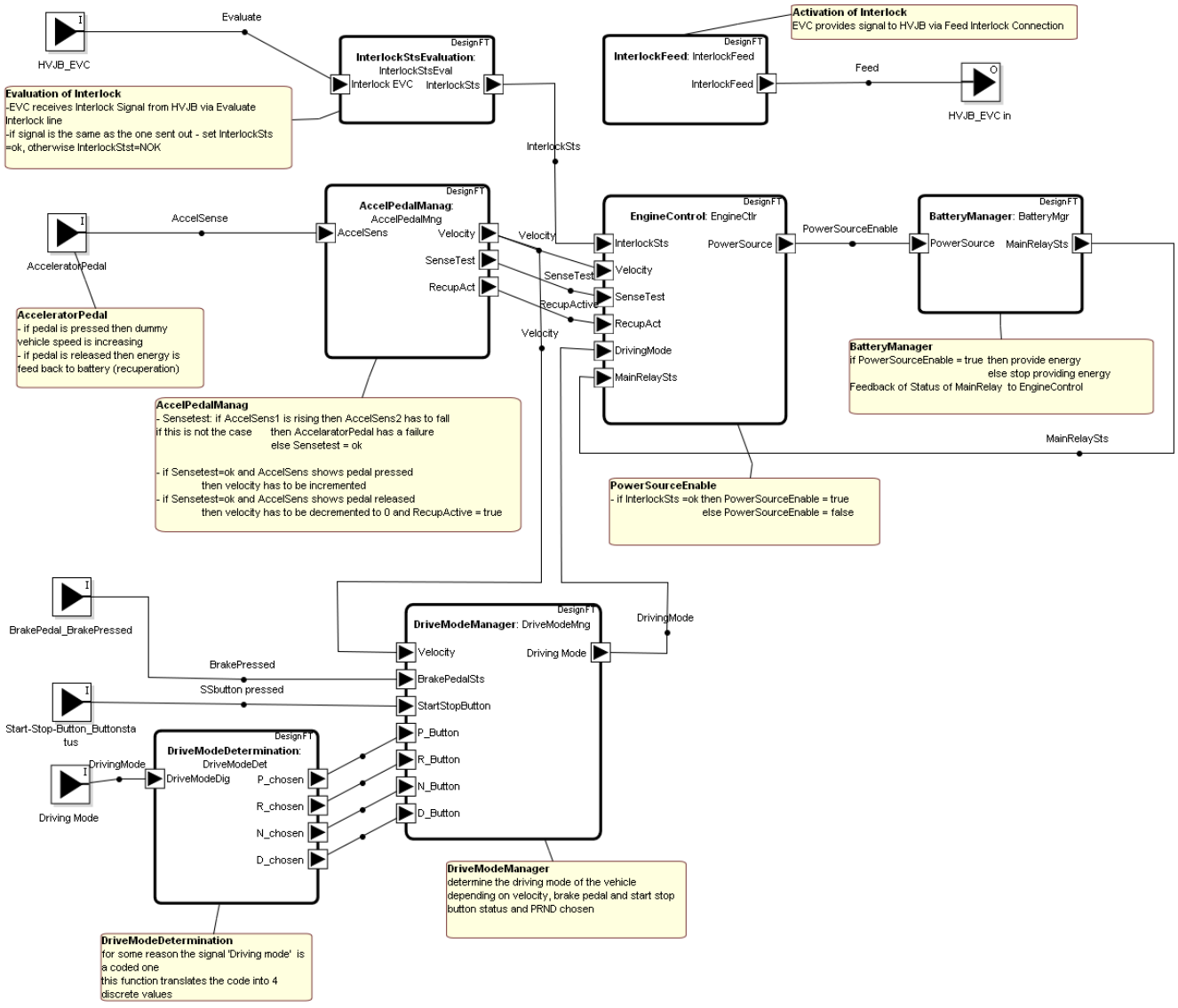


Figure 2-2: Functional Design Architecture of the "*Propulsion and power distribution*" subsystem

Hardware Design Architecture

The following Hardware Design Architecture shown in **Figure 2-3** describes the hardware realization for the features explained in chapter 2.1.1.

For all parts only the connectors relevant for this model are shown.

The battery system, delivers the energy which is guided though the High voltage junction box to the power electronic. The power electronic transforms the DC voltage to be provided to the EV motor.

The electric vehicle controller is the main controller for many powertrain functions of an electric vehicle. All functions of this model run at this controller.

The sensors in this model are needed for the selection of the driving mode.

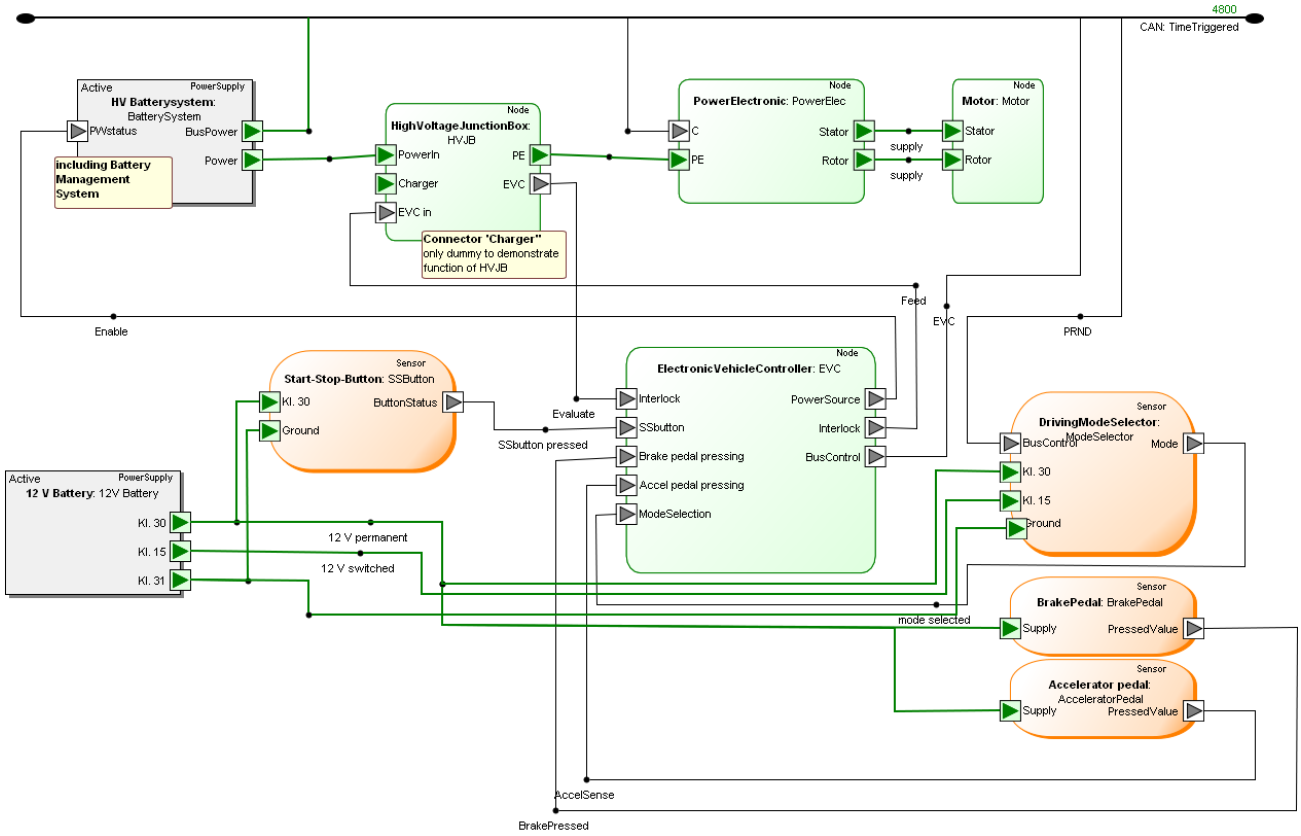


Figure 2-3: Hardware Design Architecture of the "*Propulsion and power distribution*" subsystem

2.2 Mode and range management

Within the WP5 of the ID4EV project – intelligent networking – the goal is to identify and control the driving modes and energy consumption of an electric vehicle. The component to be developed, a Comfort Range Balancer has to combine and coordinate the behaviour of several subsystems. Among them a driving profile and its management for the selection of an appropriate driving mode of an electrical vehicle, an energy management, which includes a range problem solver, the control and development of required navigation services, as well as an HMI component and interaction concept. All these components are developed within the ID4EV project. The components are also integrated in a physical demonstrator, an EV (Electrical Vehicle) developed by Continental. Various modelling concepts were applied during the project. EAST-ADL is applied on all abstraction levels of a system development. Modelica is used and applied in order to do simulations and verifications on design level. Dynamics and algorithms are also described with SysML/UML activity and state diagrams. A special focus during development and modelling activities lies on the development of algorithms for various tasks required for an electrical vehicle. This includes interaction concept which supports the driver in the various operation modes of an EV, as well as various algorithms for navigation control, range calculation, energy management, handling critical range situations, and various other tasks.

In the first phase of the project the dynamic and static models were developed on analysis level, as well as structural models on design level. In a second phase dynamic models are developed on design level using Modelica and ModelicaML. Also the verification a testing of models shall be done with the help of Modelica. Timing aspects of the model shall be modelled with TADL constraints and verified by simulations. The availability of EAST-ADL and ModelicaML as UML profiles makes it easy to combine the two approaches within the same model. ModelicaML as well as EAST-ADL is supported and customized for the usage within Papyrus. Besides Papyrus and the openModelica toolset, also a predecessor of Papyrus (TOPCASED-UML) and the CVM tooling for feature modelling and PrEEvision for HW modelling are used within the ID4EV project. The second phase of the project also includes the implementation of the components for the EV of Continental.

Below the different models and views for Mode and Range management will be shown. Although some of these are provided in non-EAST-ADL notations, they contribute to project objectives, by identifying needs and by identifying suitable approaches. Gradually these models can be migrated to EAST-ADL tools.

2.2.1 Vehicle level

In the early phases of the ID4EV project it was the goal to structure the domain of an EV with the help of a feature model. In this phase of the development also use cases and requirement documents were worked out by the partners. The main emphasis was given the definition of the requirements, but also use cases and features were discussed. On model level there is no direct linking from requirements, use cases, or features to the analysis and design model. This work could not be done within the ID4EV or MAENAD project.

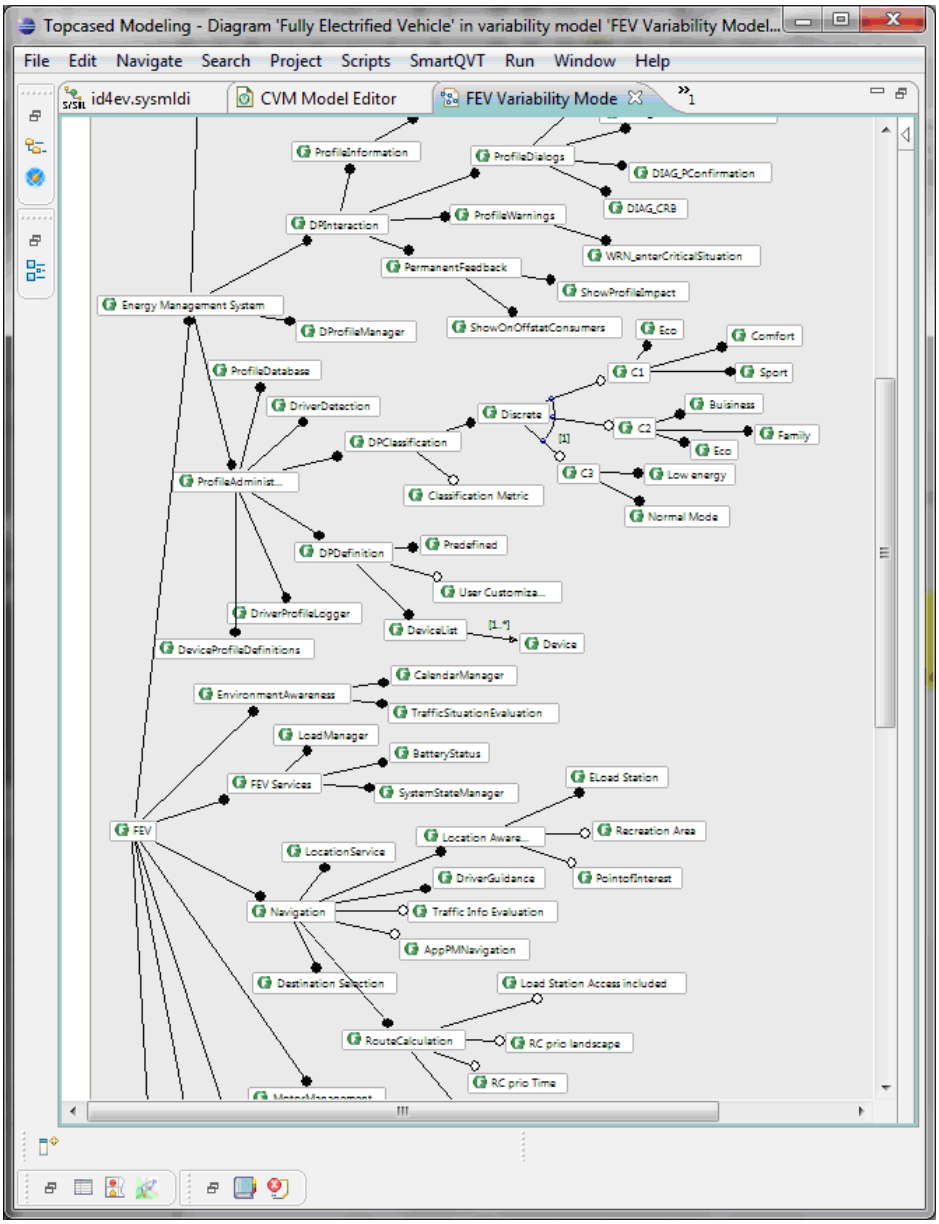


Figure 2-4. Vehicle Feature Model of the "mode and range management"

2.2.2 Analysis Level

In the first year of the ID4EV project, the analysis and design model of the system and subsystems were developed. The main system is the Comfort Range Balancer and its subsystems, among them the Range Problem Solver. Dynamic and static diagrams were developed on both abstraction levels. The models evolved from more abstract and vague analysis model to concrete and detailed design models. Especially the HMI interaction concept was worked out even in an early phase of the project, but also the dynamics or other components as the range problem solver was worked

out this way. For the most parts of the system, UML activity charts were used, one partner directly started to implement the dynamics in Simulink. As a sample for static and dynamic diagrams the behaviour and integration of the range problem solver, as worked out on analysis level, is shown below.

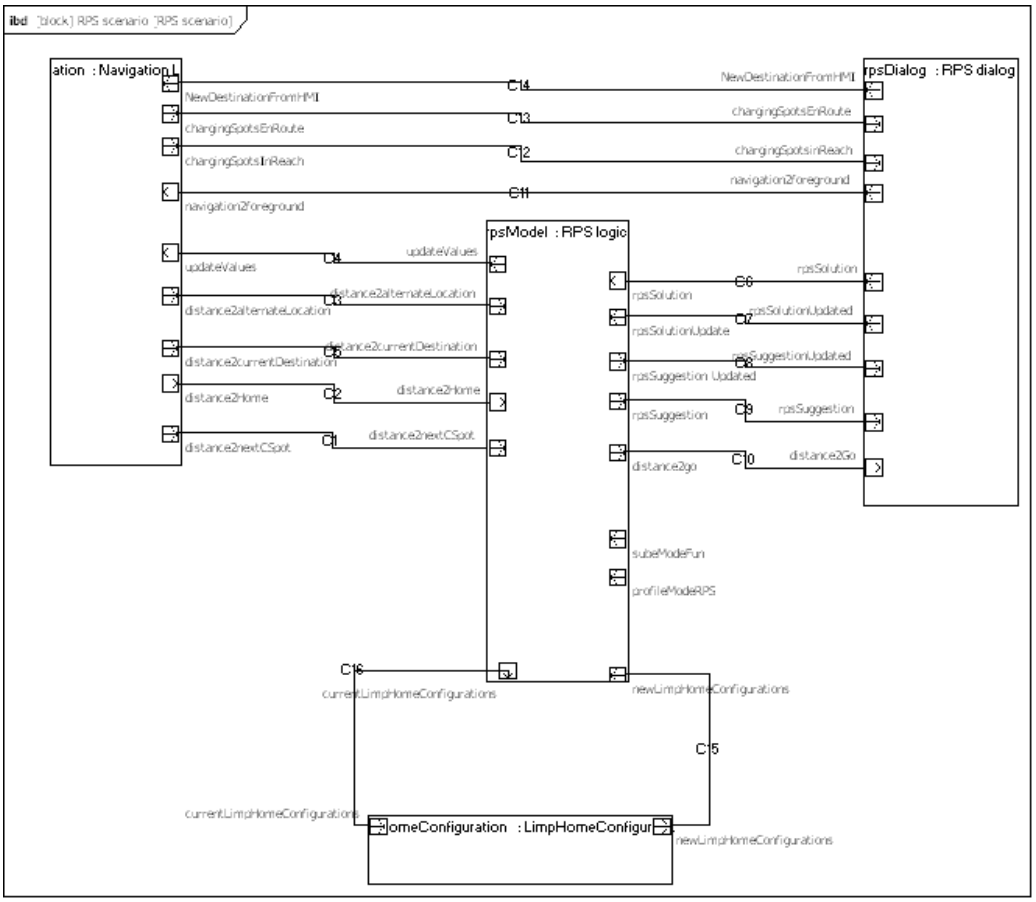


Figure 2-5: Embedded Range Problem Solver on Analysis level

The diagram above shows an earlier view of the range problem solver and its collaboration with other system components.

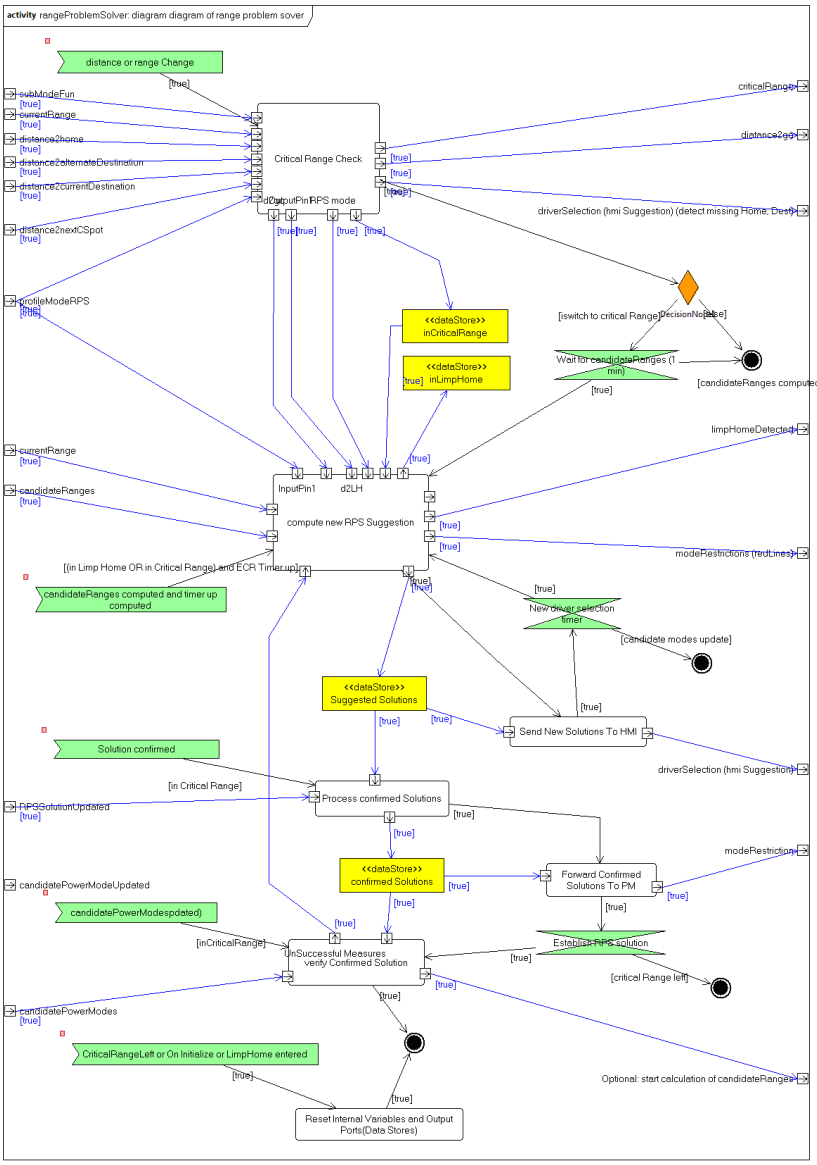


Figure 2-6: Dynamic View of Range Problem Solver (including data flows)

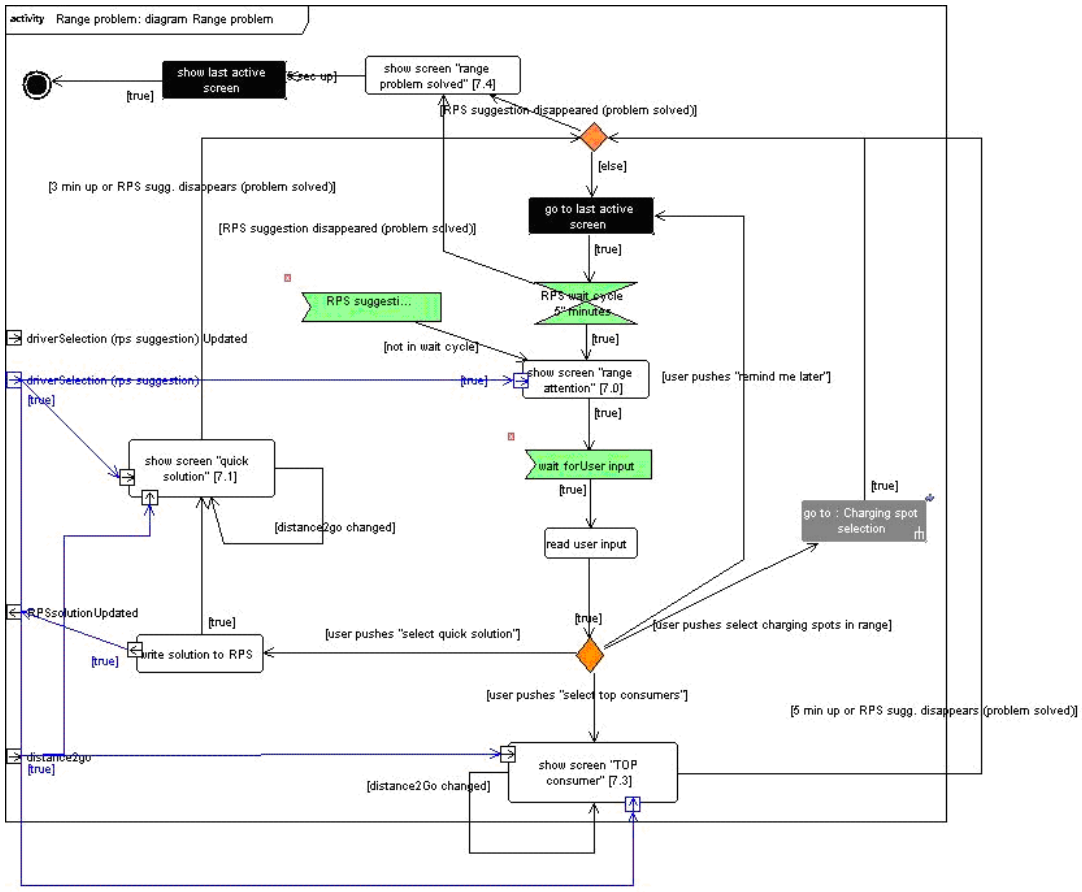


Figure 2-7: Dynamic View of Range Problem Solver Dialog (including data flows)

2.2.3 Design Level

On design level an overview of the overall System Design of the Comfort Range Balancer is given. For some selected subsystem the integration and internal components are also modelled in detail. As a sample below, the detailed view on the integration and internal view of the Range Problem Solver is given. The behaviour of some subsystems will be worked out on design level using ModelicaML and Modelica, in order to be able to perform simulations and verifications on design level. Besides the software systems the HW is modelled in the PrEEvision tooling.

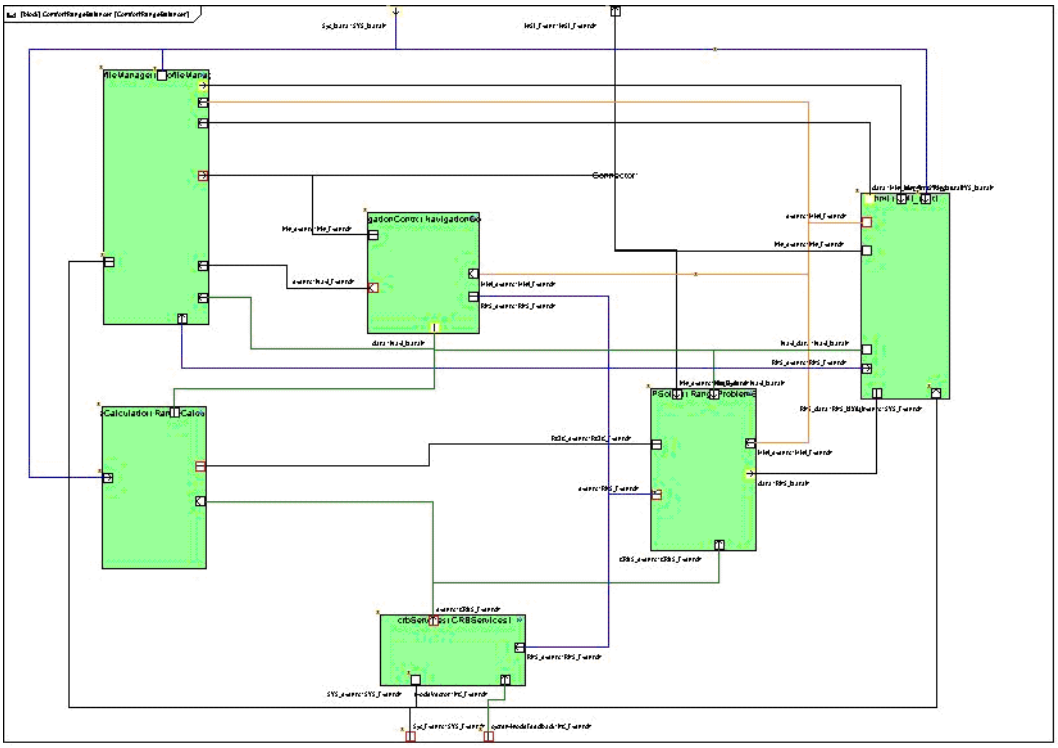


Figure 2-8 Overall Design of Comfort Range Balancer with subsystems

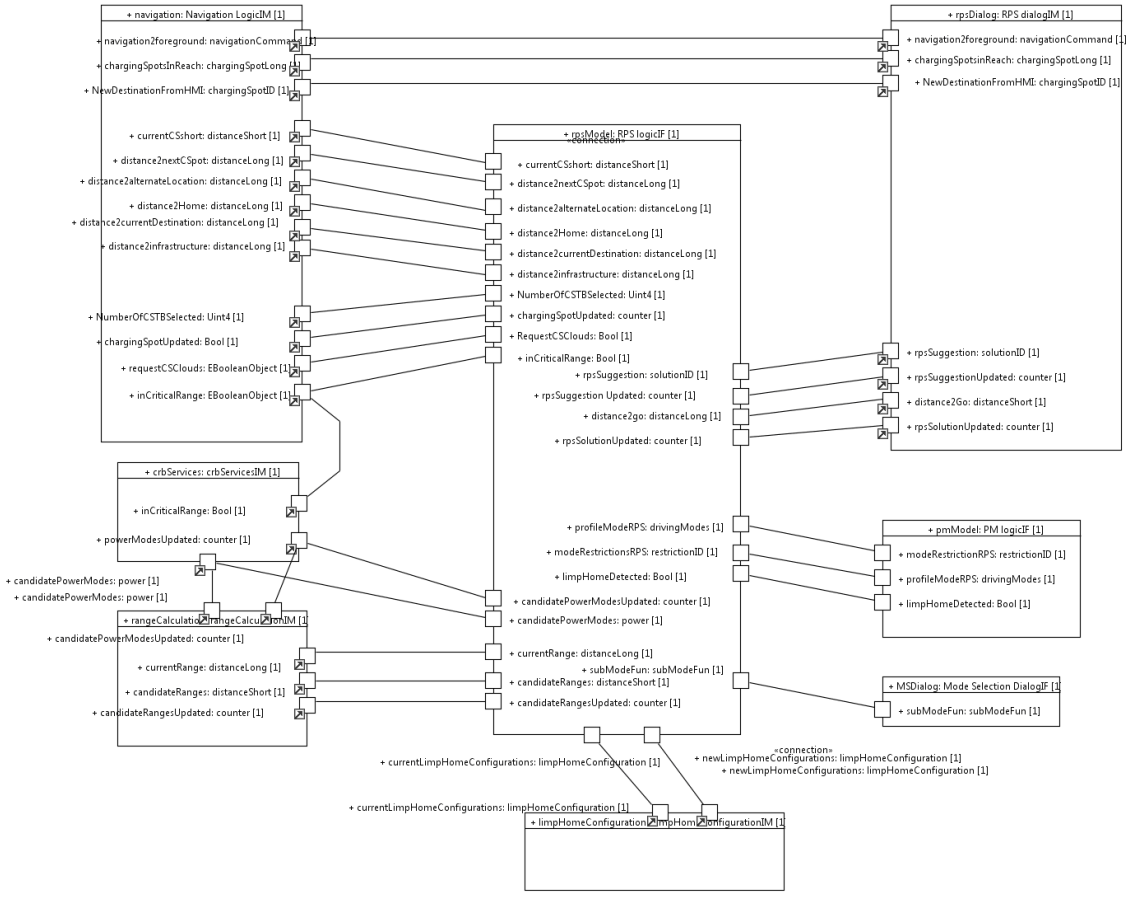


Figure 2-9: Embedded Range Problem Solver on Design Level

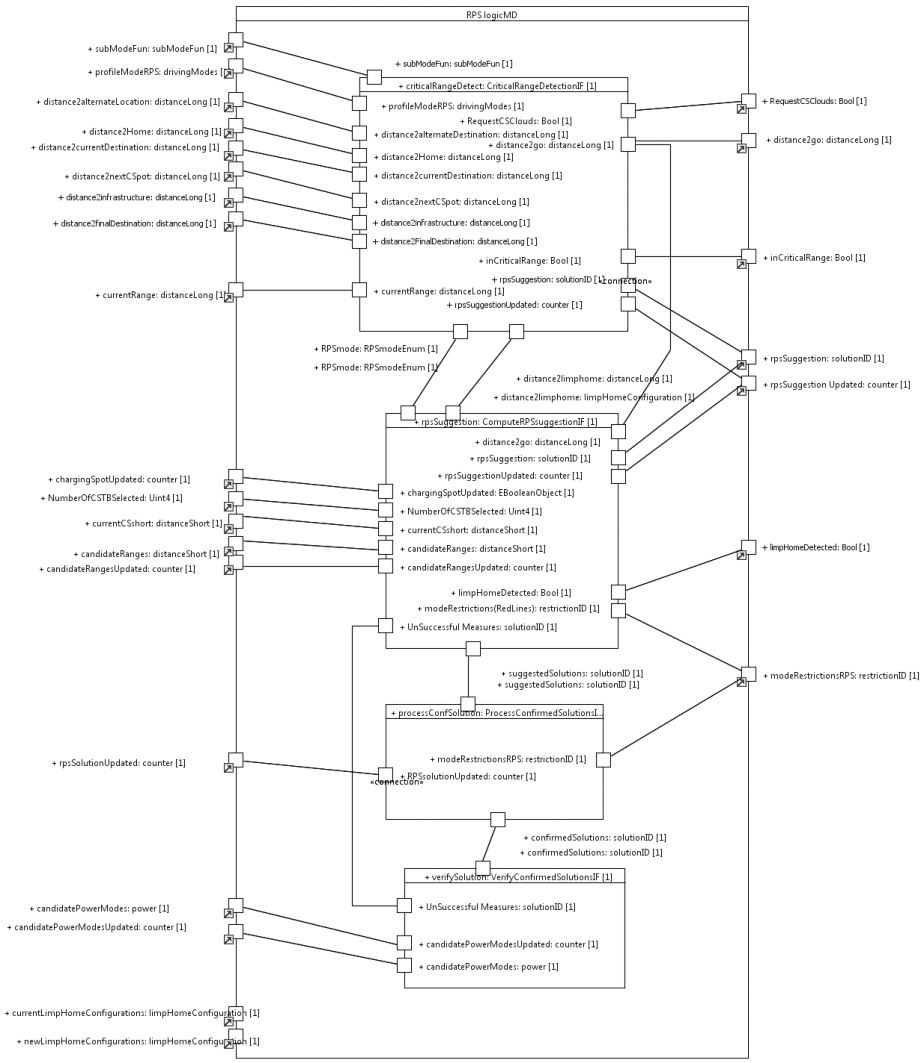


Figure 2-10: Internal View of Range Problem Solver

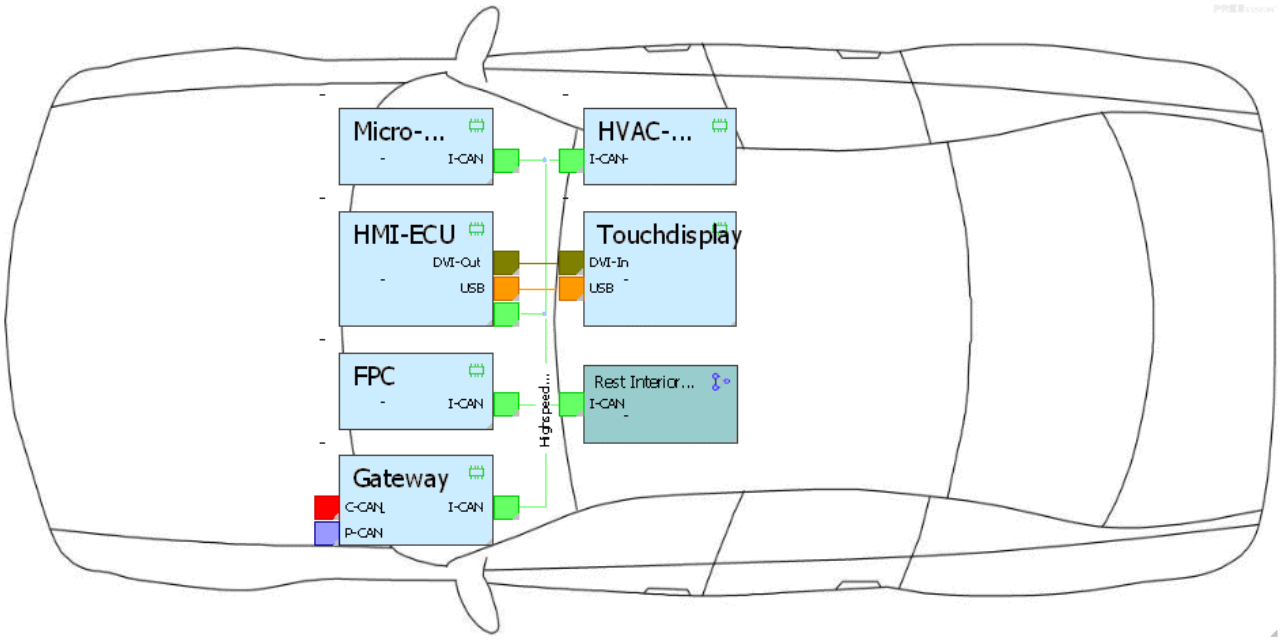


Figure 2-11: Hardware Design Architecture of Comfort Range Balancer

2.2.4 Implementation Level

Within the project proprietary runtimes like the MicroAutoBox, windows on a car PC and a proprietary Continental runtime on the gateway are used. As a consequence all required configuration were also done within the proprietary tooling as the Vector CAN tooling. All model elements of the design level are transformed into code manually. It is not inside the scope of the ID4EV project to develop and implement automated transformations from the design model into proprietary implementation tooling. However the transformation from design models into AUTOSAR, as developed within MAENAD, could be applied for verification reasons.

2.2.5 Extension Model

Behaviour model

I.) Behaviour on analysis level for specification purposes

In the analysis phase of the project, behavioural diagrams played an important role in the project. The goal here was to provide a common understanding of the various parts of the application. Mainly the HMI was described by activity charts, but also other parts of the range problem solver were described by state or activity diagrams. The level of detail of the diagrams was refined over the time, so that in the end the diagrams were a direct input for the implementation phase. An automatic transformation into code or an integration in the SW design could not be generated. Therefore manual coding of the behaviour was required.

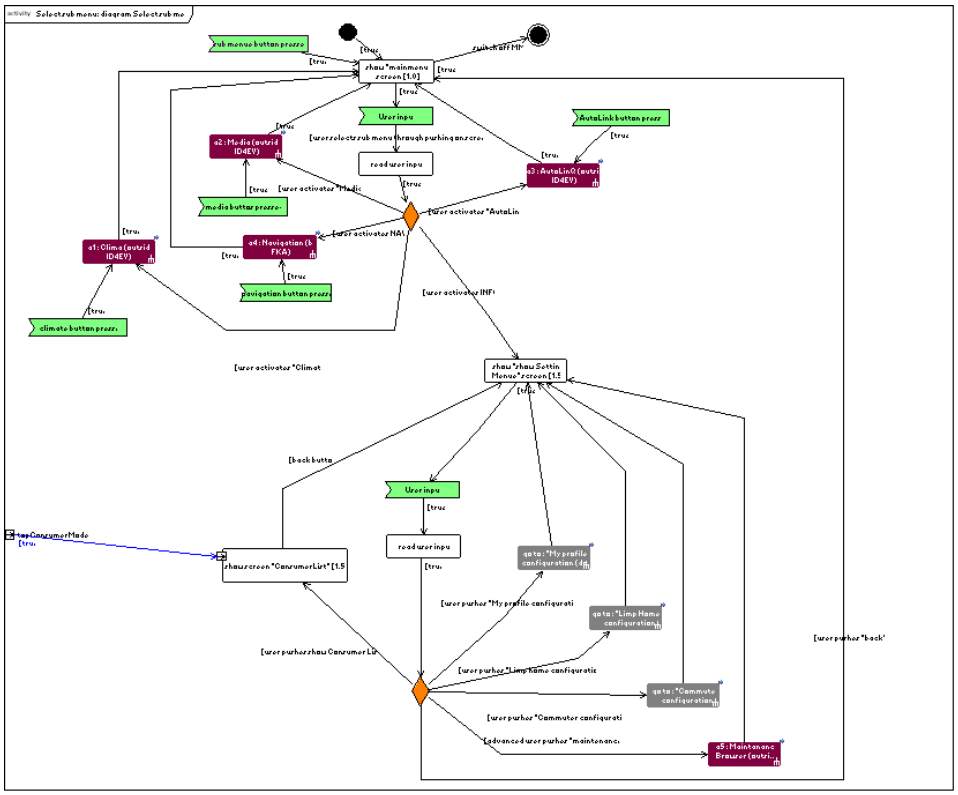


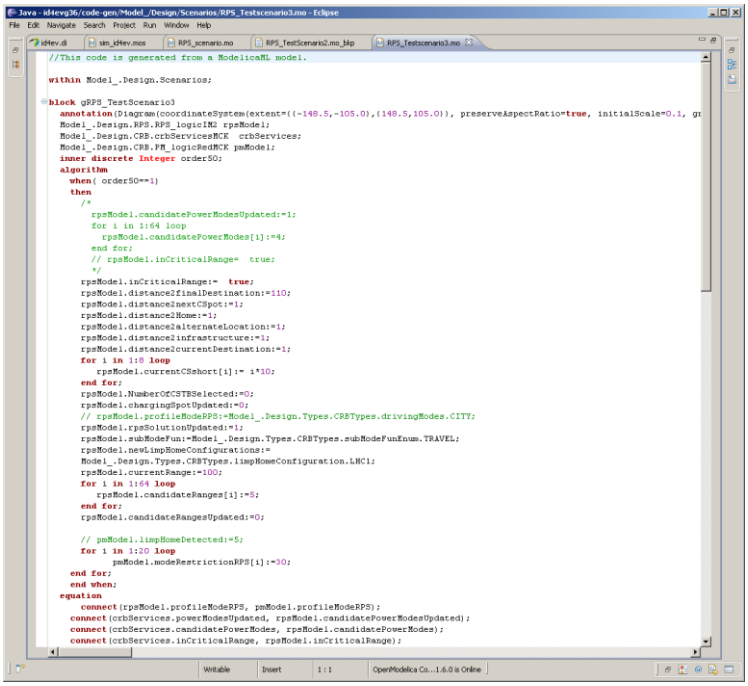
Figure 2-12: "*Mode and range management*" Activity diagram UML

II.) Behaviour on design level

The structural design of the System and SW architecture was given in an EAST-ADL model and later in an AUTOSAR model. In the first phase of the project the dynamic behaviour was partly captured in a Modelica model. When moving to the implementation phase this approach revealed several disadvantages:

- The behaviour was required in C. A further layer between behavioural diagrams and the implementation does not make sense. There should be the possibility to directly convert behavioural diagrams into C, so that no additional representation in necessary.
- The overall evaluation mechanism of Modelica brings in an additional unnecessary complexity for SW development. Modelica might be appropriate for simulation purposes, but the support of SW development is difficult, even though the model is close to SysML/EAST-ADL models.
- State Machines could be supported, but only on base of the Modelica evaluation mechanism. The integration into AUTOSAR requires a different and more flexible evaluation of State Machines. Modelica is not a dedicated State Machine tooling.
- Good User support of State Machines for application development is required, which could not be given by Modelica.

.



```
//This code is generated from a ModelicaML model.

within Model_.Design.Scenarios;

block gRPS_TestScenario3
  annotation(Diagram(coordinateSystem(extent={{-148.5,-105.0},{148.5,105.0}}, preserveAspectRatio=true, initialScale=0.1, gr
  Model_.Design.RPS.RPS_logicIM2 rpsModel;
  Model_.Design.CRB.crbServicesMCK  crbServices;
  Model_.Design.CRB.PM_logicRedMCK pmModel;
  inner discrete Integer order50;
  algorithm
    when( order50==1)
    then
      /*
        rpsModel.candidatePowerModesUpdated:=1;
        for i in 1:64 loop
          rpsModel.candidatePowerModes[i]:=4;
        end for;
        // rpsModel.inCriticalRange=  true;
        */
      rpsModel.inCriticalRange:=  true;
      rpsModel.distance2finalDestination:=110;
      rpsModel.distance2nextCSpot:=1;
      rpsModel.distance2Home:=1;
      rpsModel.distance2alternateLocation:=1;
      rpsModel.distance2infrastructure:=1;
      rpsModel.distance2currentDestination:=1;
      for i in 1:8 loop
          rpsModel.currentCSshort[i]:= i*10;
      end for;
      rpsModel.NumberOfCSTBSelected:=0;
      rpsModel.chargingSpotUpdated:=0;
      // rpsModel.profileModeRPS:=Model_.Design.Types.CRBTypes.drivingModes.CITY;
      rpsModel.rpsSolutionUpdated:=1;
      rpsModel.subModeFun:=Model_.Design.Types.CRBTypes.subModeFunEnum.TRAVEL;
      rpsModel.newLimpHomeConfigurations:=
      Model_.Design.Types.CRBTypes.limpHomeConfiguration.LHC1;
      rpsModel.currentRange:=100;
      for i in 1:64 loop
         rpsModel.candidateRanges[i]:=5;
      end for;
      rpsModel.candidateRangesUpdated:=0;

      // pmModel.limpHomeDetected:=5;
      for i in 1:20 loop
            pmModel.modeRestrictionRPS[i]:=30;
    end for;
    end when;
  equation
      connect(rpsModel.profileModeRPS, pmModel.profileModeRPS);
    connect(crbServices.powerModesUpdated, rpsModel.candidatePowerModesUpdated);
    connect(crbServices.candidatePowerModes, rpsModel.candidatePowerModes);
    connect(crbServices.inCriticalRange, rpsModel.inCriticalRange);
```

Figure 2-13: Modelica behavioural description

In the next phase of the project, it is the goal to re-implement behaviour with other State Machines and integrate these State Machines into the AUTOSAR SW design. Candidates here are StateFlow and the upcoming Yakindu tooling. It has to be worked out, if these tooling are flexible enough and can be integrated in and AUTOSAR runtime environment. These results should be taken into account for the definition of an EAST-ADL behaviour. A further requirement is that the same State Machine description can be used on all abstraction levels.

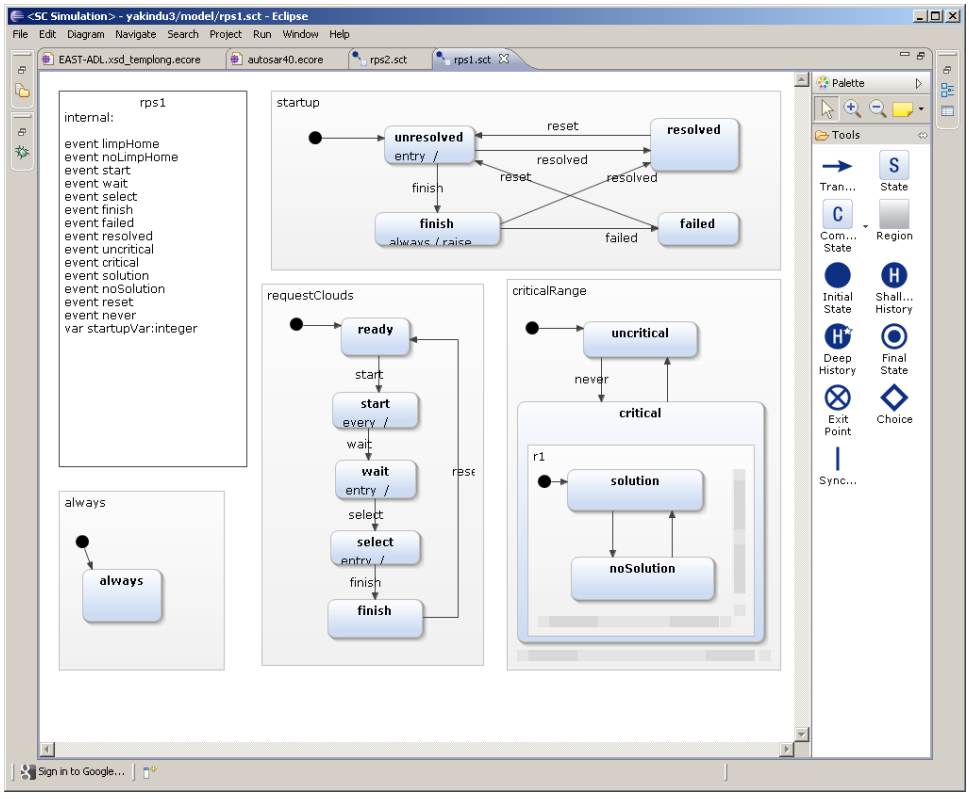


Figure 2-14: Yakindu State Machine

I.) Behaviour on Implementation level

Within the AUTOSAR model an internal behaviour can be defined, which can be seen as a link between a structural AUTOSAR model and a behavioural model. The internal behaviour defined the execution trigger and the execution context of all runnables. The AUTOSAR mode management is fully defined on this level. The detailed behavioural elements must be implemented in runnables. All runnables communicate only by defined AUTOSAR elements (Sender/Receiver, Internal Variables, …)

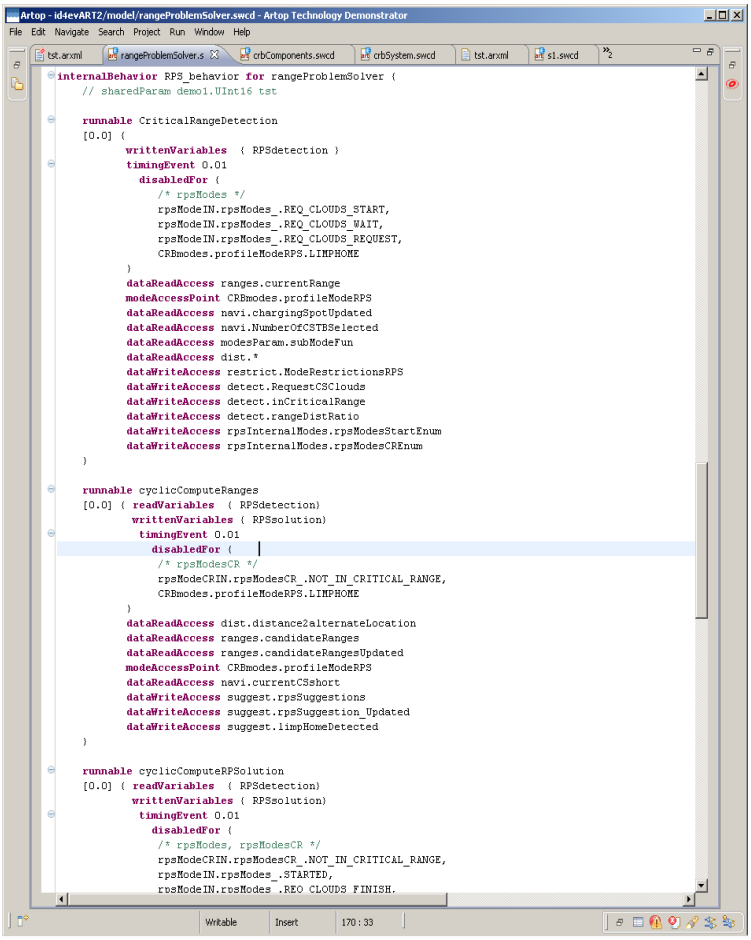
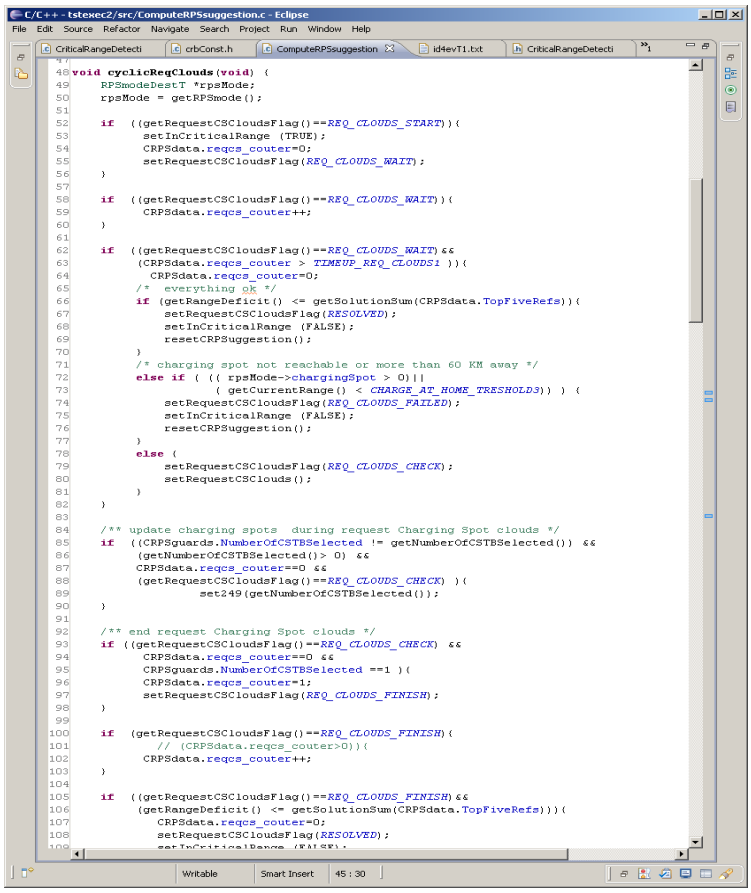


Figure 2-15: AUTOSAR internal behaviour of Range Problem Solver

The AUTOSAR model here is given in ARtext, which can be seen as an implementation language for AUTOSAR models. ARtext can be seen as a programming language for automotive/embedded systems, since the xtext/Ecore environment enables the code generation for any proprietary automotive/embedded platform. (e.g. for MMUs, AUTOSAR in normally not used). ARtext is part of the Artop AUTOSAR toolkit.

The programming language for automotive applications is normally C. For behavioural descriptions on higher abstraction levels a transformation to C is required, if the behaviour should be used on implementation level. Otherwise the behaviour description is a specification for the manual written code.




```
void cyclicReqClouds(void) {
    RPSmodeDestT *rpsMode;
    rpsMode = getRPSmode();

    if  ((getRequestCSCloudsFlag()==REQ_CLOUDS_START)){
         setInCriticalRange (TRUE);
         CRPSdata.reqcs_couter=0;
         setRequestCSCloudsFlag(REQ_CLOUDS_WAIT);
    }

    if  ((getRequestCSCloudsFlag()==REQ_CLOUDS_WAIT)){
         CRPSdata.reqcs_couter++;
    }

    if  ((getRequestCSCloudsFlag()==REQ_CLOUDS_WAIT)&&
         (CRPSdata.reqcs_couter > TIMEUP_REQ_CLOUDS1 )){
          CRPSdata.reqcs_couter=0;
        /*  everything ok */
        if (getRangeDeficit() <= getSolutionSum(CRPSdata.TopFiveRefs)){
            setRequestCSCloudsFlag(RESOLVED);
            setInCriticalRange (FALSE);
            resetCRPSuggestion();
        }
        /* charging spot not reachable or more than 60 KM away */
        else if ( (( rpsMode->chargingSpot > 0)||
                    ( getCurrentRange() < CHARGE_AT_HOME_TRESHOLD3)) ) {
            setRequestCSCloudsFlag(REQ_CLOUDS_FAILED);
            setInCriticalRange (FALSE);
            resetCRPSuggestion();
        }
        else {
            setRequestCSCloudsFlag(REQ_CLOUDS_CHECK);
            setRequestCSClouds();
        }
    }

    /** update charging spots  during request Charging Spot clouds */
    if  ((CRPSguards.NumberOfCSTBSelected != getNumberOfCSTBSelected()) &&
         (getNumberOfCSTBSelected()> 0) &&
         CRPSdata.reqcs_couter==0 &&
         (getRequestCSCloudsFlag()==REQ_CLOUDS_CHECK) ){
                 set249(getNumberOfCSTBSelected());
    }

    /** end request Charging Spot clouds */
    if ((getRequestCSCloudsFlag()==REQ_CLOUDS_CHECK)  &&
         CRPSdata.reqcs_couter==0 &&
         CRPSguards.NumberOfCSTBSelected ==1 ){
         CRPSdata.reqcs_couter=1;
         setRequestCSCloudsFlag(REQ_CLOUDS_FINISH);
    }

    if  (getRequestCSCloudsFlag()==REQ_CLOUDS_FINISH){
          // (CRPSdata.reqcs_couter>0)){
         CRPSdata.reqcs_couter++;
    }

    if  ((getRequestCSCloudsFlag()==REQ_CLOUDS_FINISH)&&
         (getRangeDeficit() <= getSolutionSum(CRPSdata.TopFiveRefs))){
            CRPSdata.reqcs_couter=0;
            setRequestCSCloudsFlag(RESOLVED);
```

Figure 2-16: C-Code: behaviour on implementation level

Variability model

A variability and feature model was worked out in the early phases of the project. It helped to come to decisions about the modularity of the system and the project. The model was not further maintained during the project. One reason is that on the tool level a direct linking of the model to models in tools like Artop or even for Papyrus on C level is not given. Another and more important reason is that variability does not play an important role within the scope of this demonstrator. The C-code is designed to be configurable and modular, but an overall system configuration is not required. The important relations between the modules are given in the AUTOSAR model.

Timing model

TADL constraints were considered on two development levels. First within the EAST-ADL model on design level, second in the AUTOSAR model, which represents the SW design on implementation level. Within the Range Problem Solver several TADL/AUTOSAR timing constraints could be considered:

the delay constraint (TADL,AUTOSAR)
the offset constraint (AUTOSAR)
the order constraint (TADL, AUTOSAR)
the execution time constraint (AUTOSAR)

Within the application there are of course various end2end constraints, but in Interior applications end2end constraints don't play the same important role as in Powertrain applications. Warnings about the critical stages of the energy consumptions have to be provided in (driver) real time, but the delay of one or two seconds can be accepted in general. Some harder time restrictions are on component level, like the sampling of device data or the responsiveness of the GUI.

State Machines play an important role in the development of the software. Many of the state transitions depend on timing conditions. It is therefore important to capture the mode and state transitions in the timing model. In the AUTOSAR timing model, an offset constraint is used in this case, in TADL a delay constraint has to be used. Unfortunately these timing constraints and events weren't available yet in the AUTOSAR tooling or the TADL editor. It is planned to extend the EAST-ADL model and the ARtext AUTOSAR language with these missing elements. In the screenshot below, an end2end constraint in the AUTOSAR ARtext language can be seen.

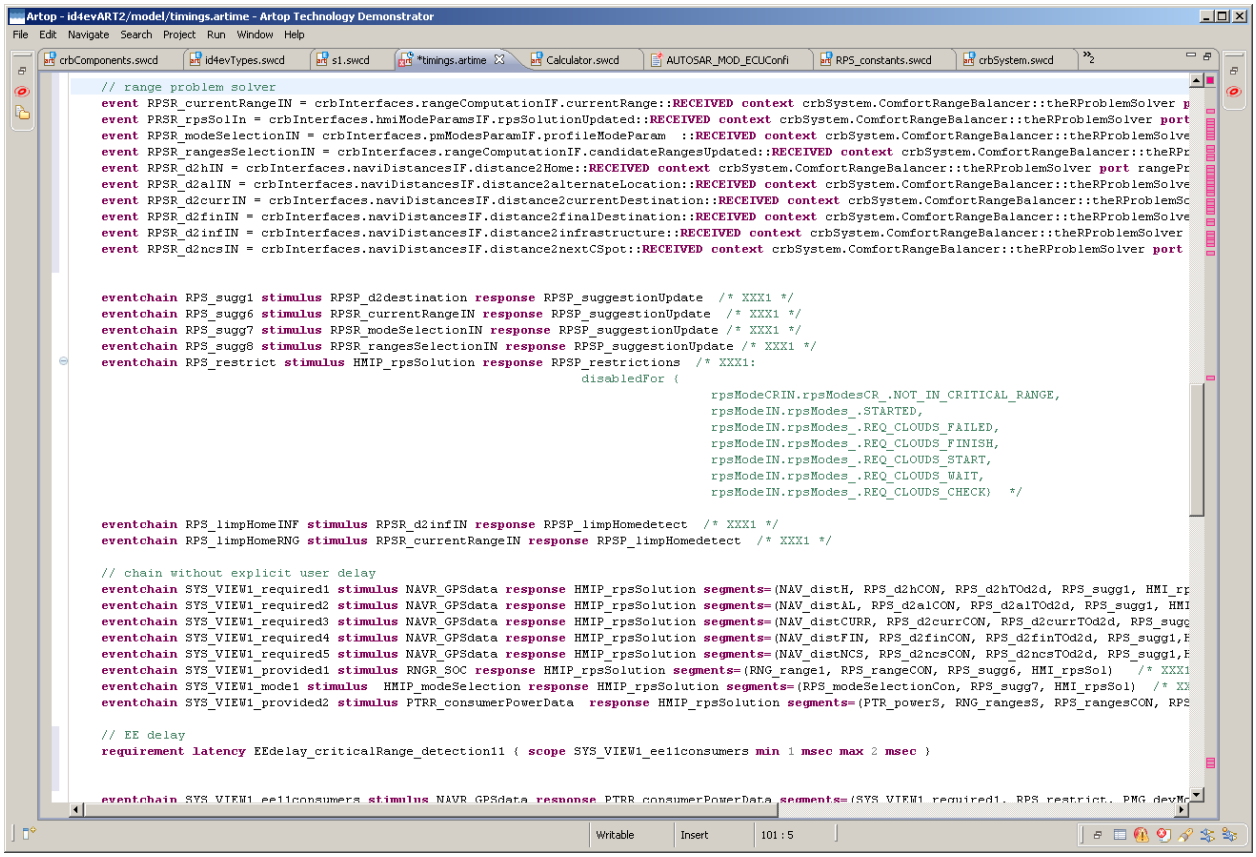


Figure 2-17: TADL constraints in ARtext

Dependability model

Not considered in demonstrator, due to the scope of the ID4EV project.

V&V model

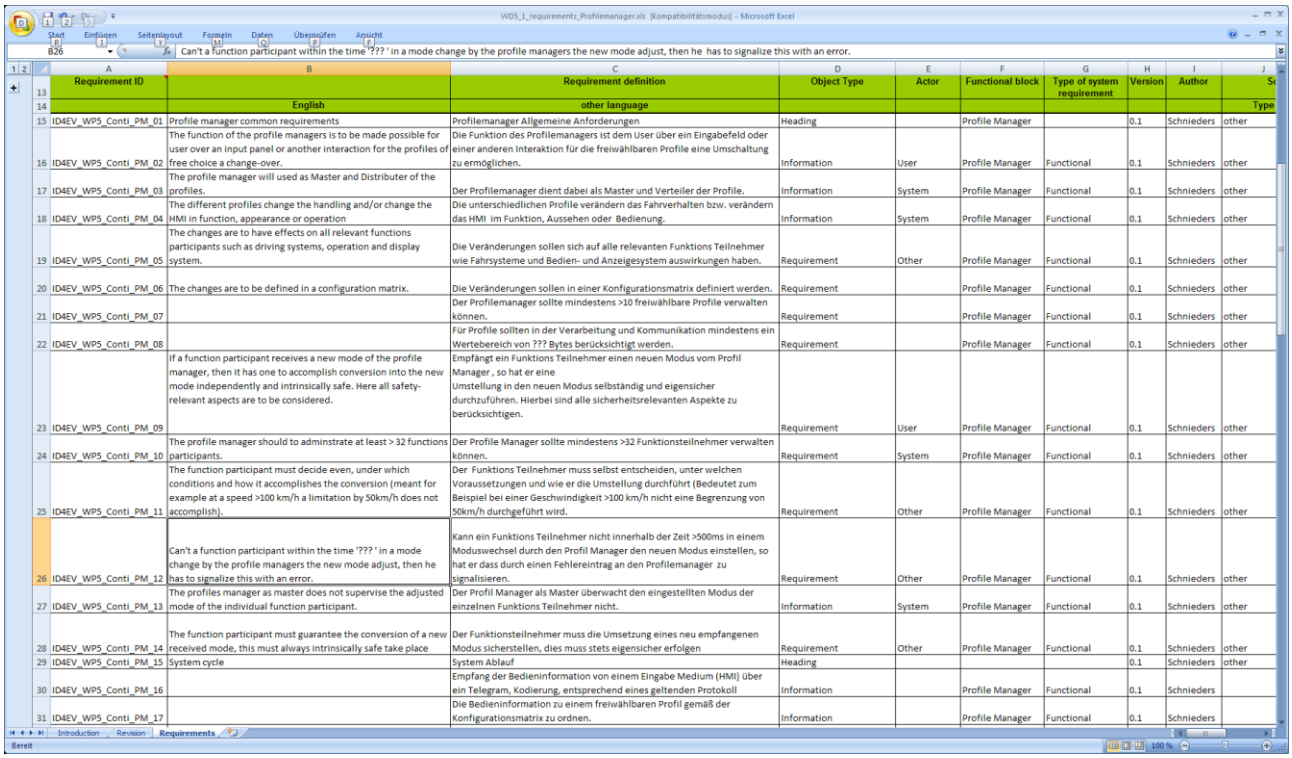
It is intended to combine/map behavioural models developed on design levels to the EAST-ADL behavioural constraints being developed. As already stated behavioural models will be developed with the help of state machine tooling Yakindu and/or StateFlow. A mapping in the EAST-ADL behaviour language will be given. In these cases, an evaluation of constraints within the state machine tooling would enable the possibility to evaluate constraints on model level. A further mapping of Constraints to EAST-ADL constraints is required in this case. This scenario has to be further worked out, along with the development of the EAST-ADL behaviour description language.

Also mapping of behaviour expressed in State Machines or EAST-ADL to AUTOSAR is required. State Machines mainly cover the internal behaviour of AUTOSAR runnables, which is not covered by AUTOSAR. Only for the remaining overlap between State Machines and AUTOSAR internal behaviour a mapping is required.

The mapping of the EAST-ADL behaviour and AUTOSAR behaviour should be in the scope of the XGA activities.

Requirements

The requirements in the ID4EV project were derived out of the description of work. The requirements were captured in excel tables/module. Due to the character of the project, requirements were not defined on model level. Required tooling for the transformation of the Excel table into the RIF format and from the RIF format into EAST-ADL was not available in the project. The definition of a fine granular link structure from model elements to requirements is not defined for these requirements.



	Requirement ID		Requirement definition	Object Type	Actor	Functional block	Type of system requirement	Version	Author	
14		English	other language							Type
15	ID4EV_WP5_Conti_PM_01	Profile manager common requirements	Profilemanager Allgemeine Anforderungen	Heading		Profile Manager		0.1	Schnieders	other
16	ID4EV_WP5_Conti_PM_02	The function of the profile managers is to be made possible for user over an input panel or another interaction for the profiles of free choice a change-over.	Die Funktion des Profilemanagers ist dem User über ein Eingabefeld oder einer anderen Interaktion für die freiwählbaren Profile eine Umschaltung zu ermöglichen.	Information	User	Profile Manager	Functional	0.1	Schnieders	other
17	ID4EV_WP5_Conti_PM_03	The profile manager will used as Master and Distributer of the profiles.	Der Profilemanager dient dabei als Master und Verteiler der Profile.	Information	System	Profile Manager	Functional	0.1	Schnieders	other
18	ID4EV_WP5_Conti_PM_04	The different profiles change the handling and/or change the HMI in function, appearance or operation	Die unterschiedlichen Profile verändern das Fahrverhalten bzw. verändern das HMI im Funktion, Aussehen oder Bedienung.	Information	System	Profile Manager	Functional	0.1	Schnieders	other
19	ID4EV_WP5_Conti_PM_05	The changes are to have effects on all relevant functions participants such as driving systems, operation and display system.	Die Veränderungen sollen sich auf alle relevanten Funktions Teilnehmer wie Fahrsysteme und Bedien- und Anzeigesystem auswirkungen haben.	Requirement	Other	Profile Manager	Functional	0.1	Schnieders	other
20	ID4EV_WP5_Conti_PM_06	The changes are to be defined in a configuration matrix.	Die Veränderungen sollen in einer Konfigurationsmatrix definiert werden.	Requirement		Profile Manager	Functional	0.1	Schnieders	other
21	ID4EV_WP5_Conti_PM_07		Der Profilemanager sollte mindestens >10 freiwählbare Profile verwalten können.	Requirement		Profile Manager	Functional	0.1	Schnieders	other
22	ID4EV_WP5_Conti_PM_08		Für Profile sollten in der Verarbeitung und Kommunikation mindestens ein Wertebereich von ??? Bytes berücksichtigt werden.	Requirement		Profile Manager	Functional	0.1	Schnieders	other
23	ID4EV_WP5_Conti_PM_09	If a function participant receives a new mode of the profile manager, then it has one to accomplish conversion into the new mode independently and intrinsically safe. Here all safety-relevant aspects are to be considered.	Empfängt ein Funktions Teilnehmer einen neuen Modus vom Profil Manager , so hat er eine Umstellung in den neuen Modus selbständig und eigensicher durchzuführen. Hierbei sind alle sicherheitsrelevanten Aspekte zu berücksichtigen.	Requirement	User	Profile Manager	Functional	0.1	Schnieders	other
24	ID4EV_WP5_Conti_PM_10	The profile manager should to adminstrate at least > 32 functions participants.	Der Profile Manager sollte mindestens >32 Funktionsteilnehmer verwalten können.	Requirement	System	Profile Manager	Functional	0.1	Schnieders	other
25	ID4EV_WP5_Conti_PM_11	The function participant must decide even, under which conditions and how it accomplishes the conversion (meant for example at a speed >100 km/h a limitation by 50km/h does not accomplish).	Der Funktions Teilnehmer muss selbst entscheiden, unter welchen Voraussetzungen und wie er die Umstellung durchführt (Bedeutet zum Beispiel bei einer Geschwindigkeit >100 km/h nicht eine Begrenzung von 50km/h durchgeführt wird.	Requirement	Other	Profile Manager	Functional	0.1	Schnieders	other
26	ID4EV_WP5_Conti_PM_12	Can't a function participant within the time '??? ' in a mode change by the profile managers the new mode adjust, then he has to signalize this with an error.	Kann ein Funktions Teilnehmer nicht innerhalb der Zeit >500ms in einem Moduswechsel durch den Profil Manager den neuen Modus einstellen, so hat er dass durch einen Fehlereintrag an den Profilemanager zu signalisieren.	Requirement	Other	Profile Manager	Functional	0.1	Schnieders	other
27	ID4EV_WP5_Conti_PM_13	The profiles manager as master does not supervise the adjusted mode of the individual function participant.	Der Profil Manager als Master überwacht den eingestellten Modus der einzelnen Funktions Teilnehmer nicht.	Information	System	Profile Manager	Functional	0.1	Schnieders	other
28	ID4EV_WP5_Conti_PM_14	The function participant must guarantee the conversion of a new received mode, this must always intrinsically safe take place	Der Funktionsteilnehmer muss die Umsetzung eines neu empfangenen Modus sicherstellen, dies muss stets eigensicher erfolgen	Requirement	Other	Profile Manager	Functional	0.1	Schnieders	other
29	ID4EV_WP5_Conti_PM_15	System cycle	System Ablauf	Heading				0.1	Schnieders	other
30	ID4EV_WP5_Conti_PM_16		Empfang der Bedieninformation von einem Eingabe Medium (HMI) über ein Telegram, Kodierung, entsprechend eines geltenden Protokoll	Information		Profile Manager	Functional	0.1	Schnieders	
31	ID4EV_WP5_Conti_PM_17		Die Bedieninformation zu einem freiwählbaren Profil gemäß der Konfigurationsmatrix zu ordnen.	Information		Profile Manager	Functional	0.1	Schnieders	

Figure 2-18: Requirements in excel table

2.3 Regenerative Braking System

This case study aims to demonstrate the support of EAST-ADL for the development of full electrical vehicles (FEV) as a whole, ranging from requirements specification, to architecture modelling with multi-level synthesis, and to analysis of various behaviours and qualities, verification and validation. As the first iterative step towards this goal, an initial EAST-ADL model for the target braking system specified in D6.1.1 (- Preliminary case study definition and evaluation metrics), which focuses on the architecture specification aspect, is currently being built up and introduced in this section. Two implementations of EAST-ADL will be supported: 1. Papyrus through UML profile; 2. MetaCase++ through DSL. In the following part of this section, we introduce only the Papyrus/UML based implementation (Papyrus 0.7.4 EASTADL 2.1.9).

2.3.1 Overall Model

Figure 2-19 provides a package structure overview of the expected EAST-ADL modelling elements for the braking system architecture, as well as its associated requirements, variability and other non-functional constraints (e.g., timing and dependability), and verification&validation (V&V) cases. The *SystemModel* (within the *0_TopPackage*) contains the entire braking electrical/electronic system architecture, for which specifications at various abstraction levels are applied. Figure 2-20 provides a graphical representation of this multi-level braking electrical/electronic system specification and its related environment model (*EnvironmentBBW*).

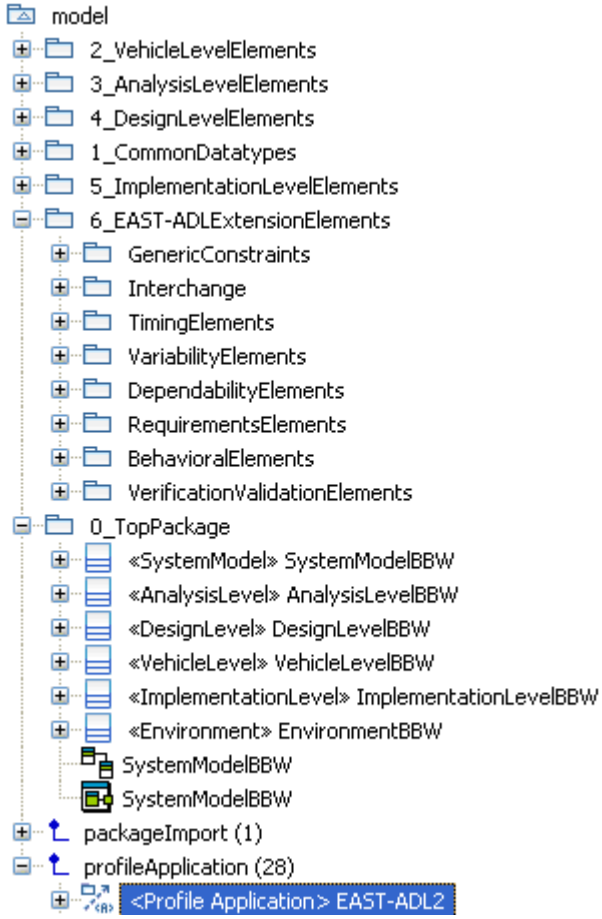


Figure 2-19: An overview of packages of an EAST-ADL model in Papyrus.

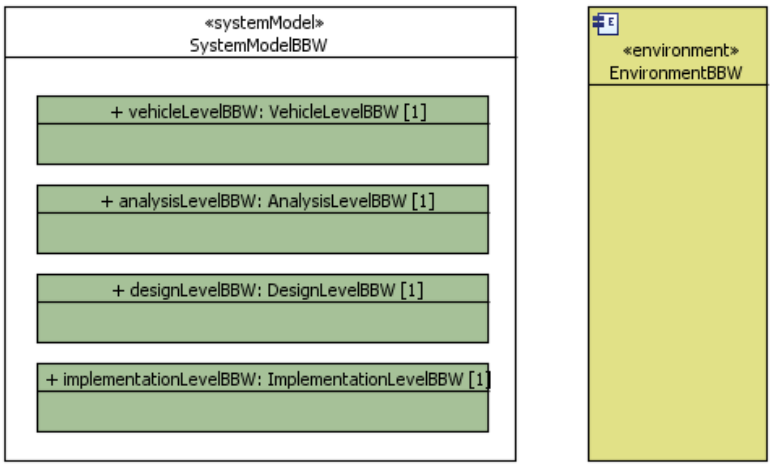


Figure 2-20: The braking electrical/electronic system and its environment in Papyrus.

EAST-ADL supports requirements, V&V cases, and the annotations of variability and other non-functional constraints through separate modelling packages shown in Figure 2-21. (Such extension packages are contained in the *EAST-ADL ExtensionElements* package in Figure 2-21). A requirement model specifies the conditions or capabilities that must be met or possessed by a system or its component. In a model-based approach, requirements are derived, refined, mapped, validated and verified along with the progress of system design. The specifications of variability and other non-functional constraints augment the multi-level system architecture specification with analytical information (e.g. timing, reliability, and safety integrity) for early quality predictions and contract declarations. Normally, an analytical model should have its level of abstraction according to its target artefacts.

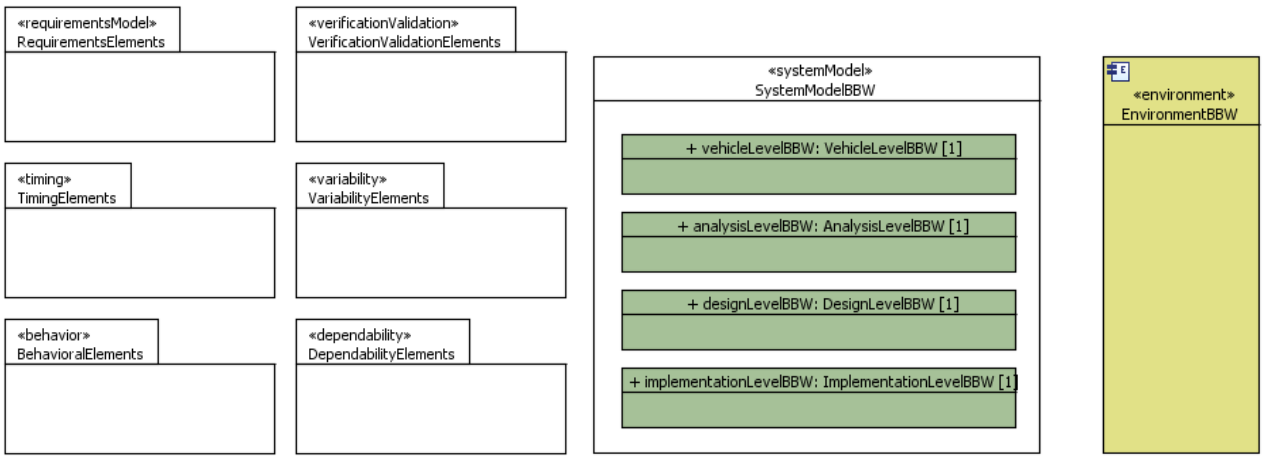


Figure 2-21: An overview of system model and related EAST-ADL packages for the specifications of requirements, V&V cases, and the annotations of variability and other non-functional constraints in Papyrus.

2.3.2 Vehicle level

A vehicle level architecture specification constitutes the topmost system description and manages the features of an entire product family. In Figure 2-22, the feature tree of the target braking system is shown. Each vehicle feature (*VehicleFeature*) denotes a functional characteristic, such as the functions, or non-functional properties, to be supported. While a braking control feature (*BrakingControl*) is needed for the vehicle longitudinal control, regenerative braking control

(*RegenerativeBrakingControl*) is a feature for power control in FEV, allowing the kinetic energy produced by braking to be converted to electrical energy and stored in capacitor or/and battery. As shown in Figure 2-22, the relations of features are supported by feature links (*FeatureLink*). In a feature link definition, the precise semantics of a feature relationship is given by the type attribute (*Kind*) and the direction attribute (*isBidirectional*).

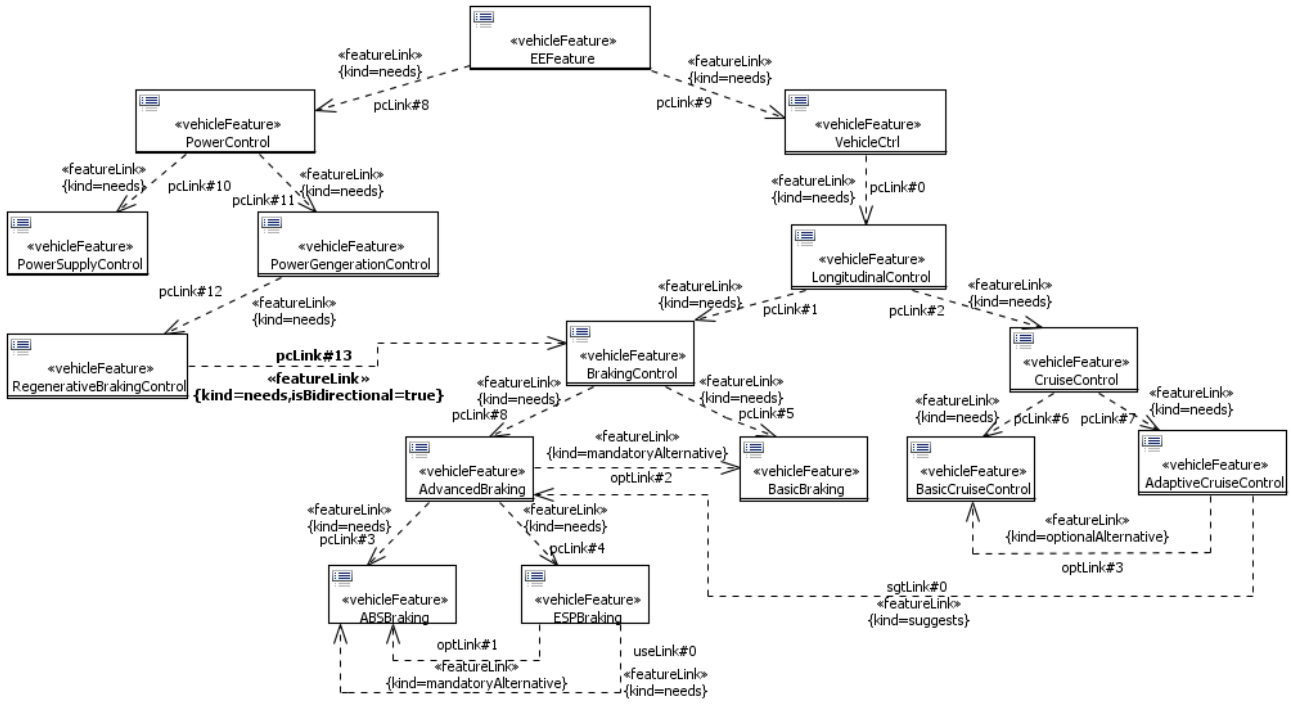


Figure 2-22: Vehicle Feature Model of the Regenerative Braking System in Papyrus.

Requirements at the vehicle level are directly based on system use cases and allocated to vehicle features denoting the expected system functions). See Table 1 for a list of requirements on braking control. By EAST-ADL, the relationships of a requirement in regard to other requirements, system artefacts, more detailed analytical models, and V&V cases are explicit supported.

Table 1: Top-level braking control requirements.

ID	Description
Req#1_BaseBraking	"The system shall provide a base brake functionality where the driver indicates that he/she wants to reduce speed and the braking system starts decelerating the vehicle"
Req#2_DriverBrakeRequest	"The driver shall be able to request braking"
Req#3_Anti-LockBraking	"The system shall be an anti-lock braking system (ABS) by preventing the wheels from locking while braking"
Req#4_BrakeReactionTime	"The time from the driver's brake request until the actual start of the deceleration shall be ≤ 300ms.(Value derived from expert judgment)"
Req#5_TimeToStandstill	"The time to stadstill shall follow the recommendations in EU braking systems Directive 71/320 EEC. The Swedish Road Administration claims that a factor of 3 (on braking distance) is acceptable for ice"
Req#6_OperationofBrakePedal	"The Operator shall be able to vary the desired braking force using the brake pedal. A fully pressed pedal means maximum brake force."
Req#7_BrakeRelease	"When the brake pedal is not pressed, the brake shall not be active."

While a feature tree model specifies composition of system functions and their logical dependencies, it often implies the refinement of vehicle level requirements. With EAST-ADL, the derived/derived by relationship of requirements is given by a dedicated requirement relationship: *DeriveRequirement*. When such a requirement relationship is declared, a modification of the supplier requirement would have effects on the derived client requirements. Figure 2-23 shows the

requirements model capturing four derived requirements and their relationships to a common supplier requirement and to each other.

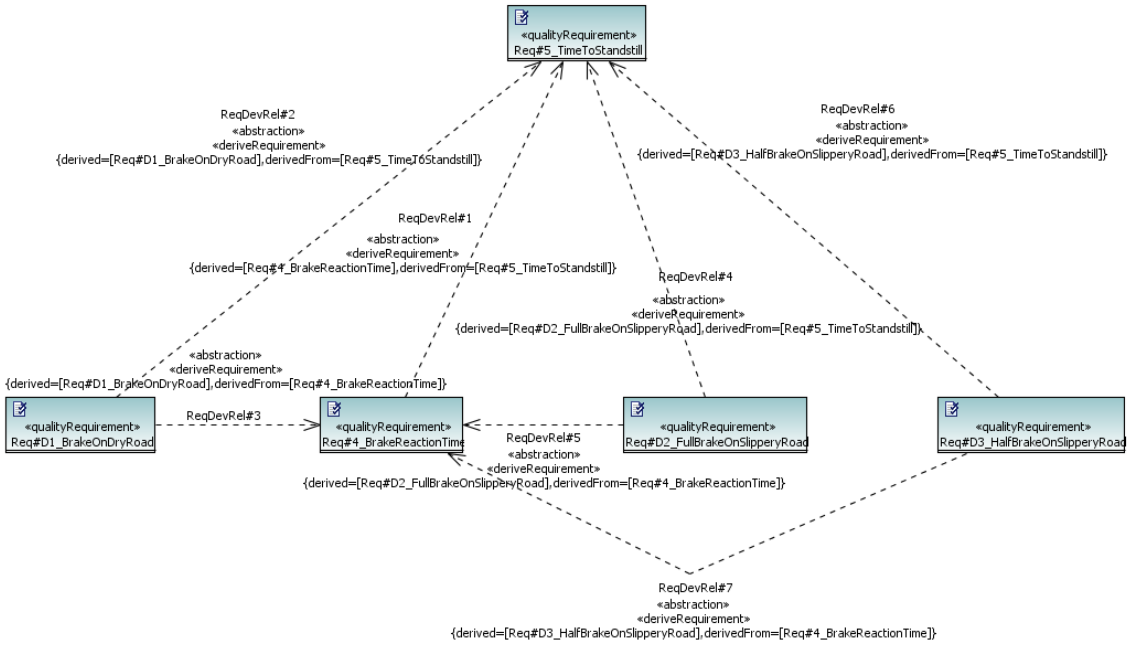


Figure 2-23: A model of braking performance requirements in Papyrus.

Figure 2-24 shows the allocations of functional and non-functional requirements to the braking control and its sub-features through the *Satisfy* links.

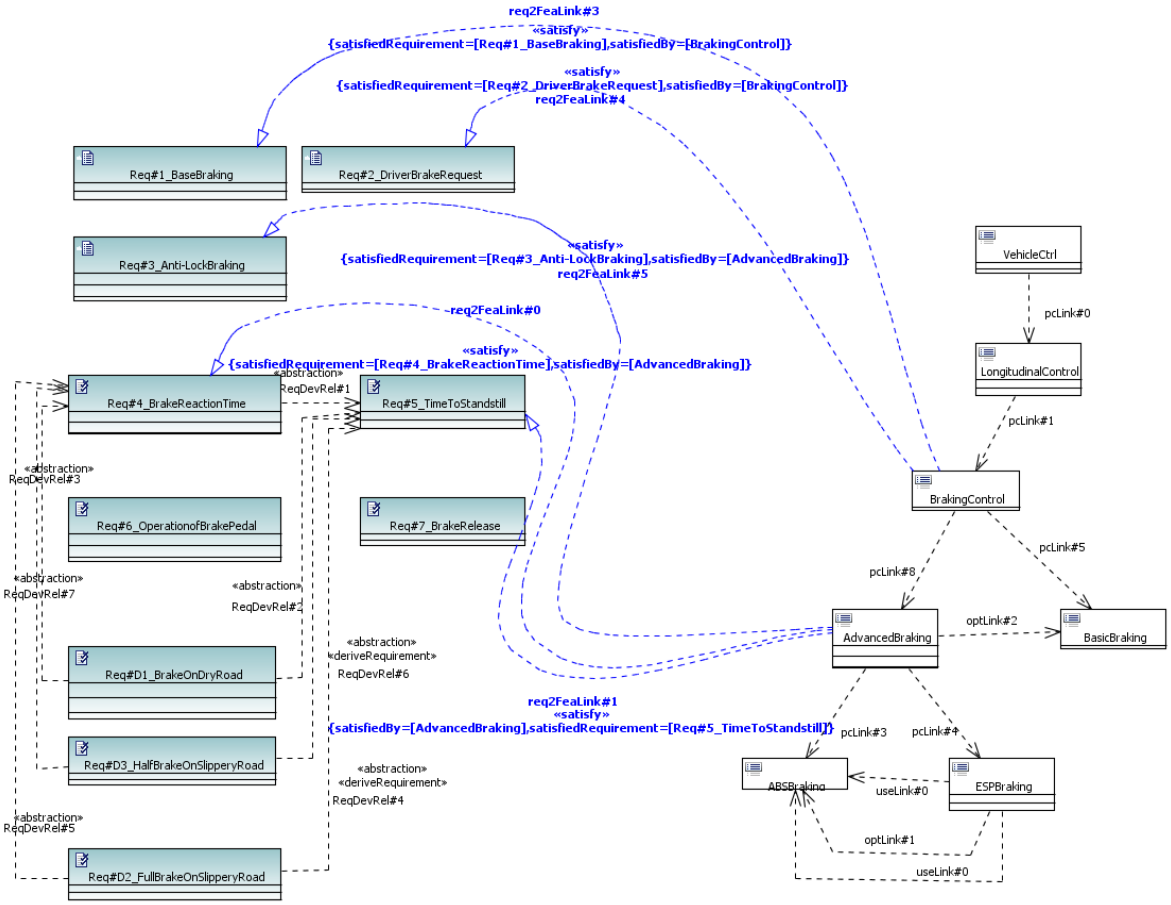


Figure 2-24: Allocations of braking requirements on vehicle features in Papyrus.

In EAST-ADL, a satisfy relationship signifies the relationship between a requirement and an architectural element intending to satisfy the requirement. Requirements can also be inherited along with the feature configuration hierarchy. For example, the requirements *Req#1_BaseBraking* and *Req#2_DriverBrakeRequest*, shown in Figure 2-24, should also be satisfied by the children of *BrakingControl*, such as the *AdvancedBraking* and the *BasicBraking.*

2.3.3 Analysis Level

As a step towards system realization, the vehicle level features are realised by some interconnected abstract functions at the analysis level, specifying the corresponding input functions, application functions, and output functions for each vehicle level function in an implementation independent way. For the target braking system, the vehicle features of concern are implemented by a set of analysis functions shown in Figure 2-25 and Figure 2-26.

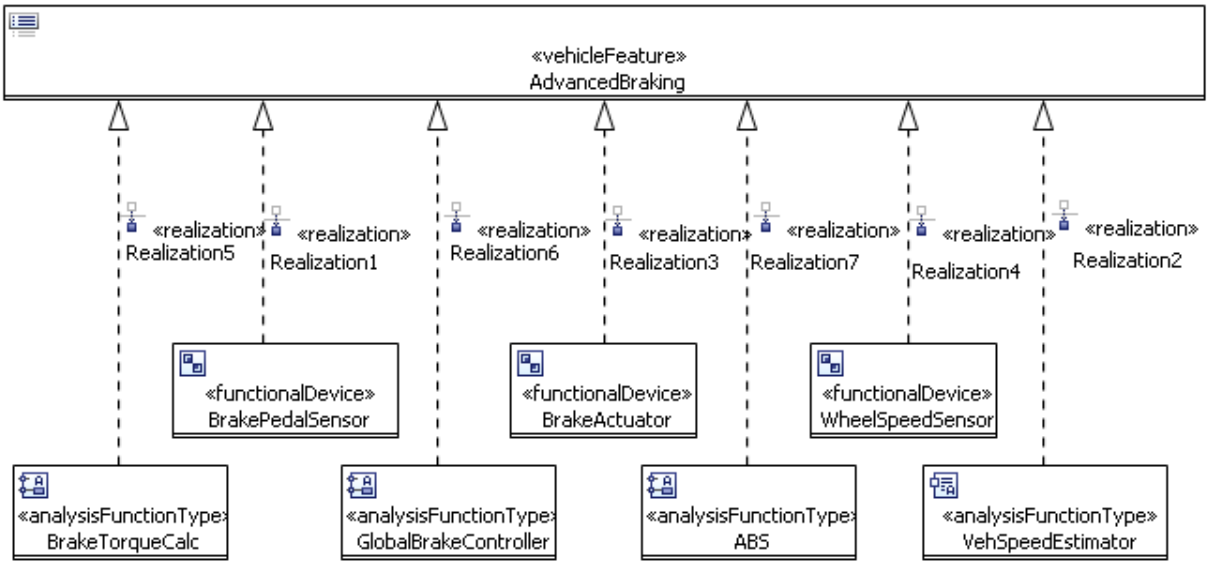


Figure 2-25: Advanced Braking feature and the specification of its functional realizations in Papyrus.

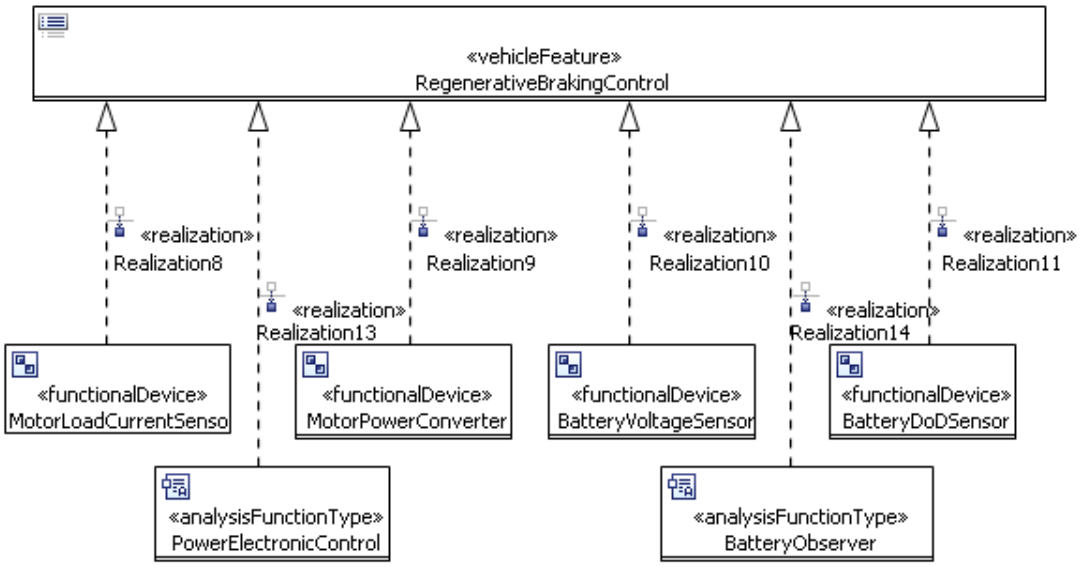


Figure 2-26: Regenerative Braking Control feature and the specification of its functional realizations in Papyrus.

Figure 2-27 shows the specification of functional architecture in EAST-ADL for the braking system (See also D6.1.1 for an overview the functional operation concept).

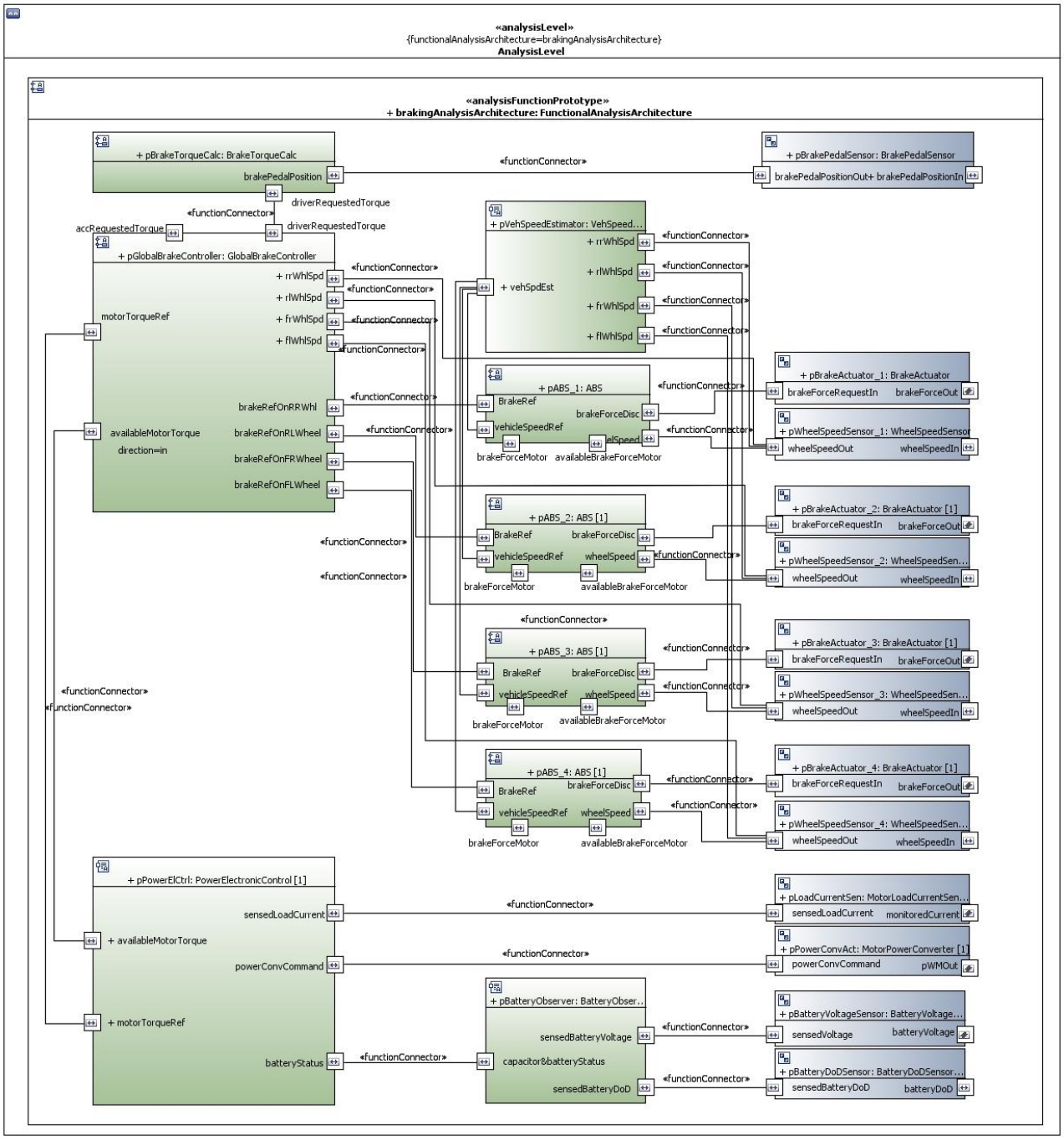


Figure 2-27: Functional Analysis Architecture specification of the Regenerative Braking System in Papyrus.

In EAST-ADL, system boundaries are explicitly defined by means of functional devices (*FunctionalDevice*). Through functional devices, an analysis function interacts with the physical environment. Figure 2-28 shows the connections between functional devices and the physical environment.

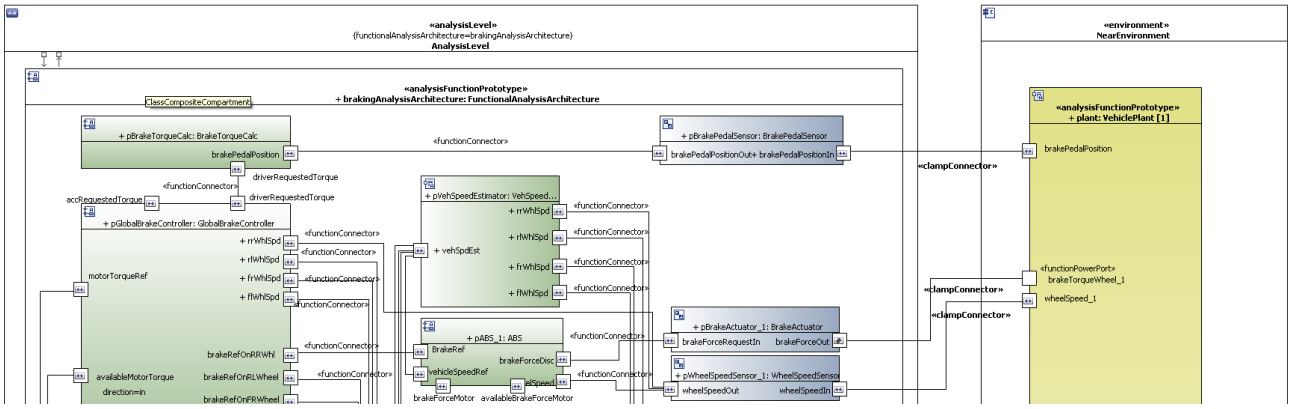


Figure 2-28: Connecting functional analysis functions with environment in Papyrus.

To define the timing requirements and timing design, constructs like TimingConstraint, EventChain and Event are available in EAST-ADL.

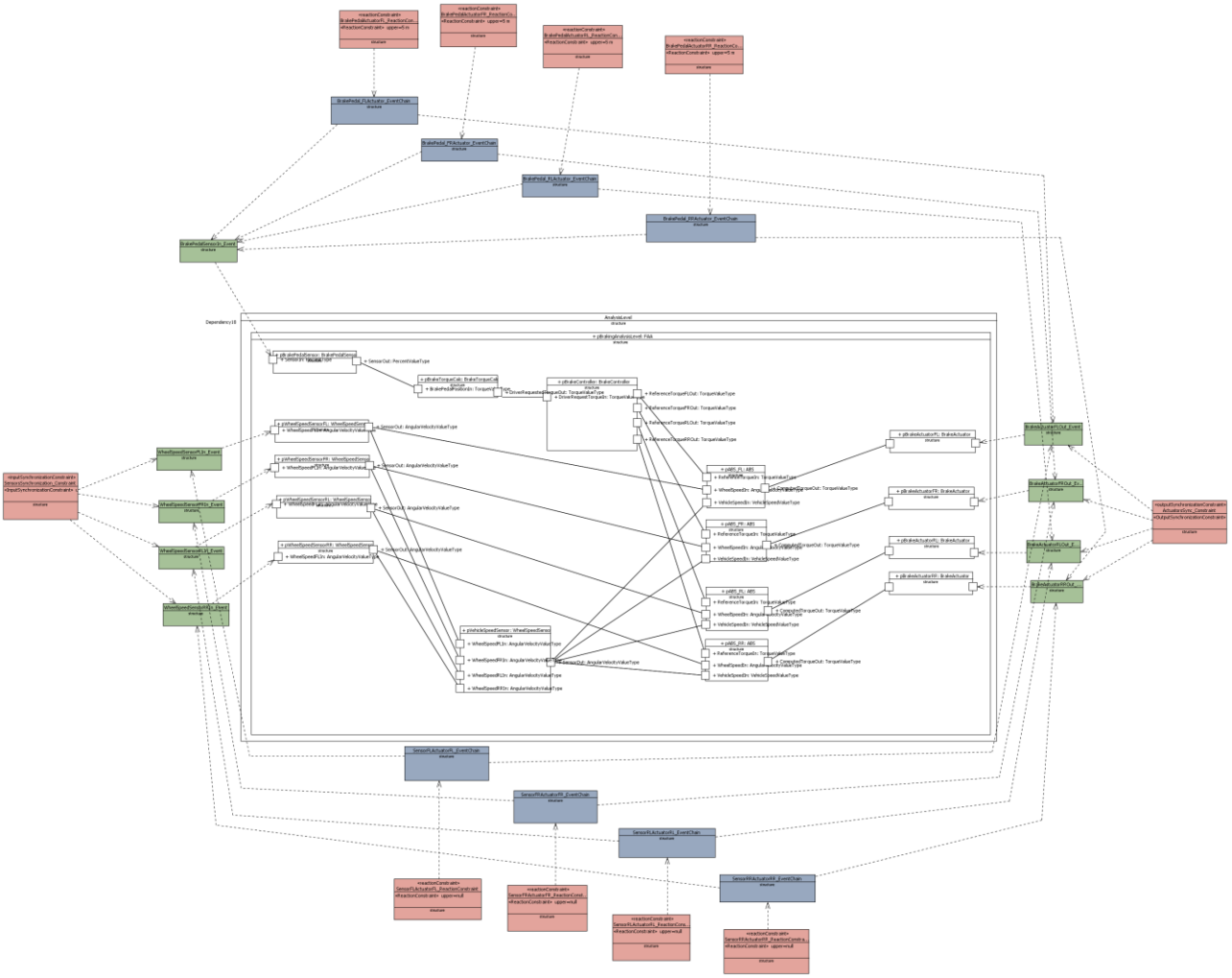


Figure 2-29. Synchronization and End-to-end timing from pedal to brake actuators

2.3.4 Design Level

The design level architecture further details the analysis level design by taking the software and hardware resources into consideration. (See also D6.1.1 for an overview the related design concept).

Functional Design Architecture

Figure 2-30 shows the FunctionalDesignArchitecture. This model is focusing on base braking and does not include energy regeneration functionality.

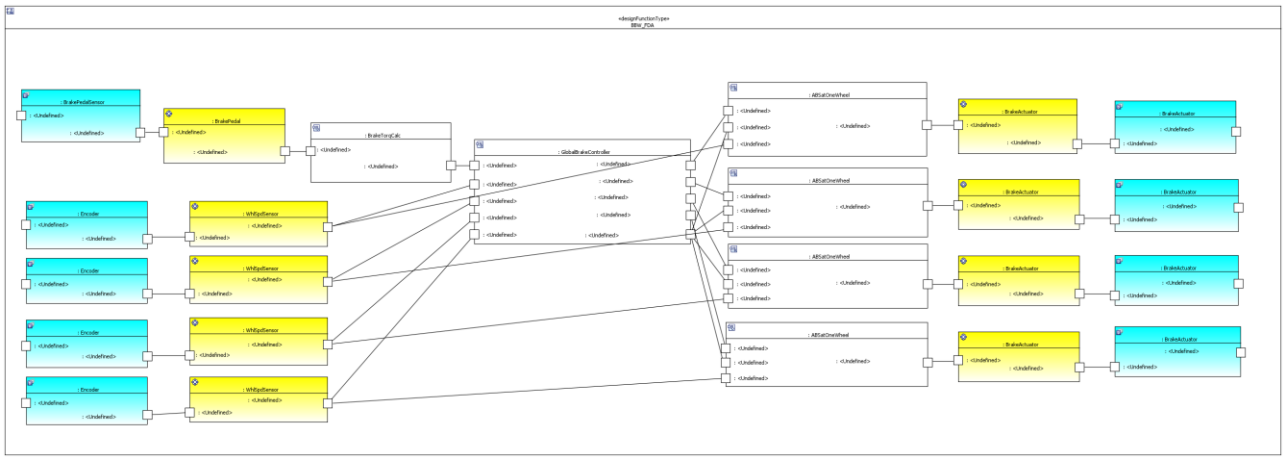


Figure 2-30. Functional Design Architecture of the Regenerative Braking System in Papyrus.

Figure 2-31 shows the period times of the included functions.

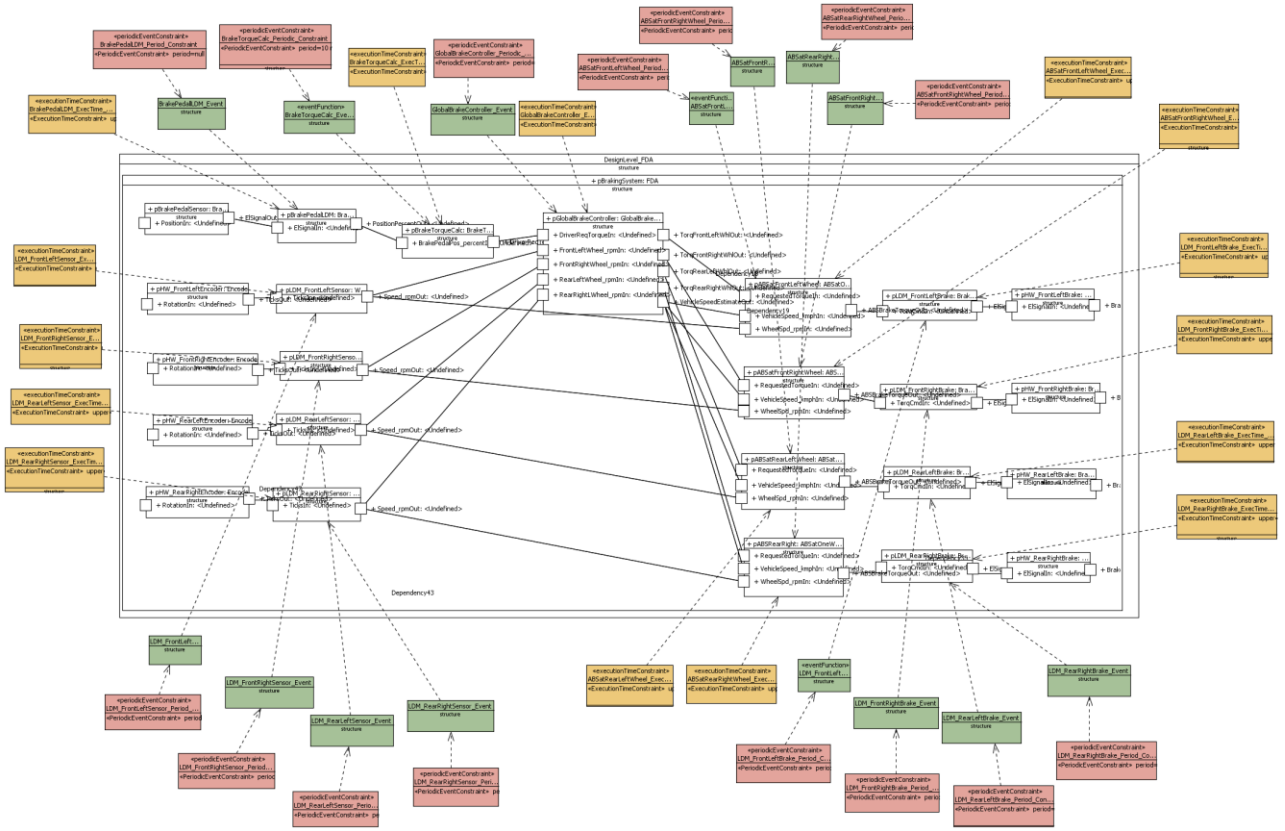


Figure 2-31. Period times of functions

Figure 2-32 (close-up) and Figure 2-33 (overall) shows timing constraints for end-to-end response requirements of the brake functionality. Figure 2-33 also show synchronization requirements and a brake-down of the end-to-end timing budget.

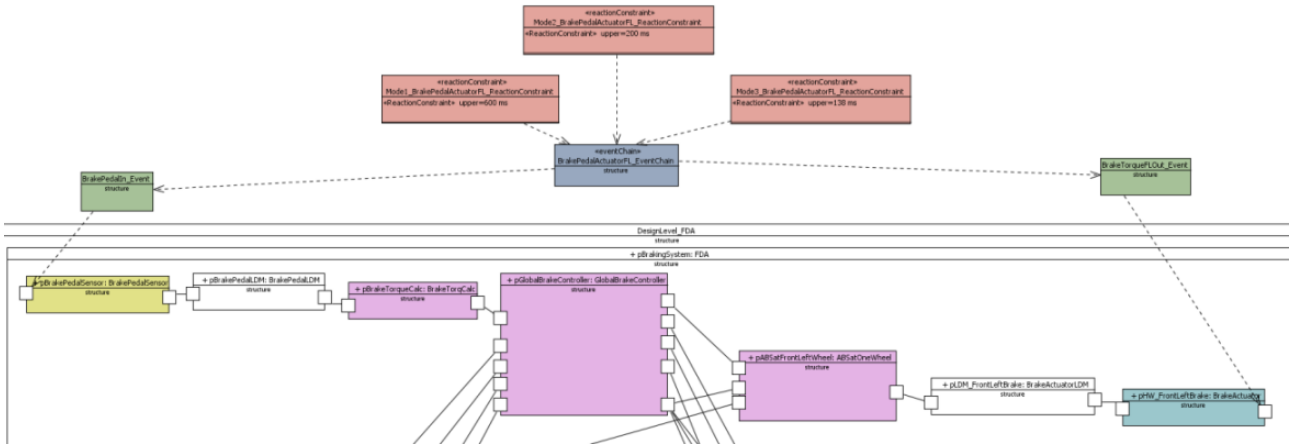


Figure 2-32. Functional Design Architecture with end-to-end timing

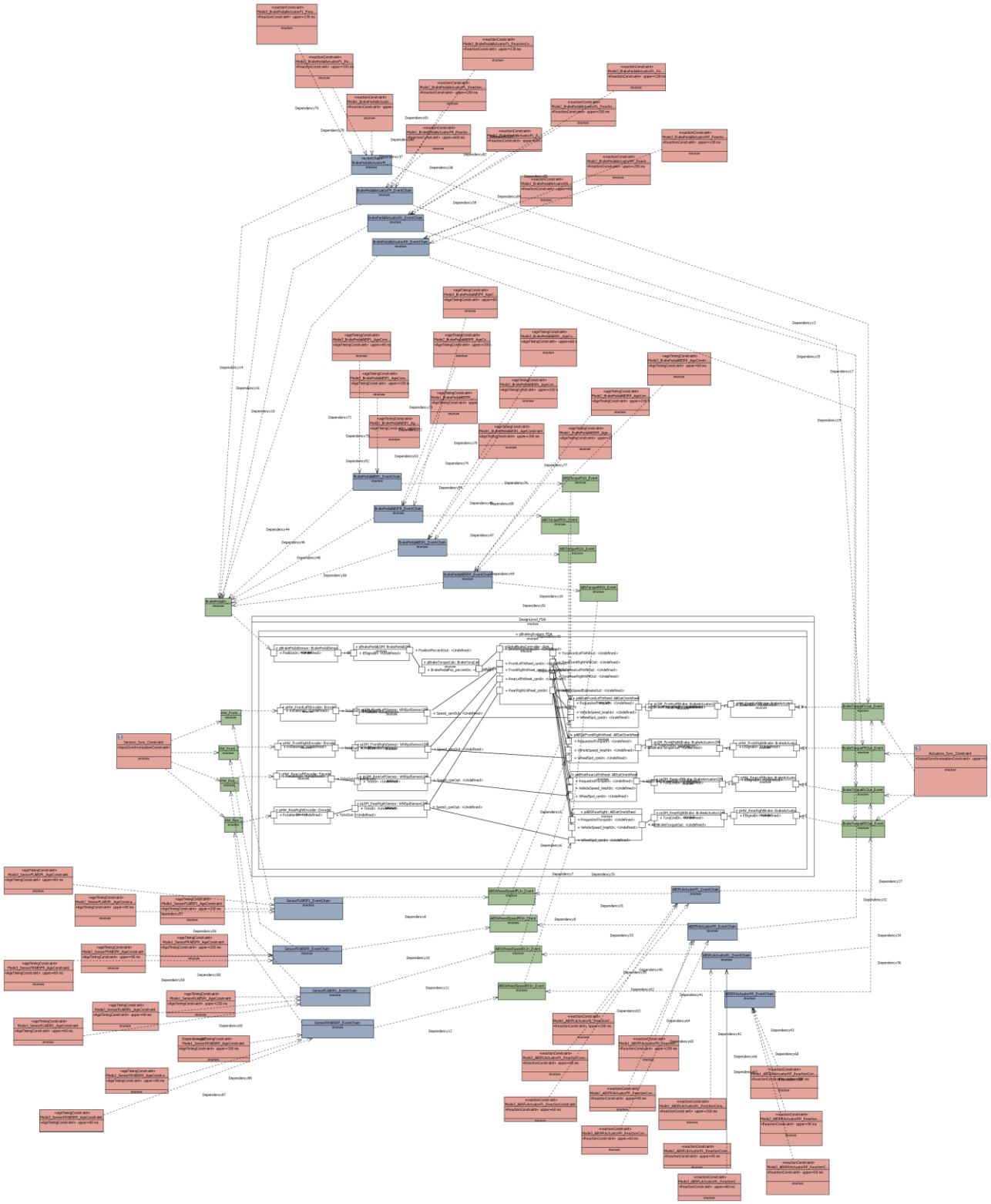


Figure 2-33. Functional Design Architecture with end-to-end timing

Hardware Design Architecture

Figure 2-34 shows an initial HardwareDesignArchitecture.

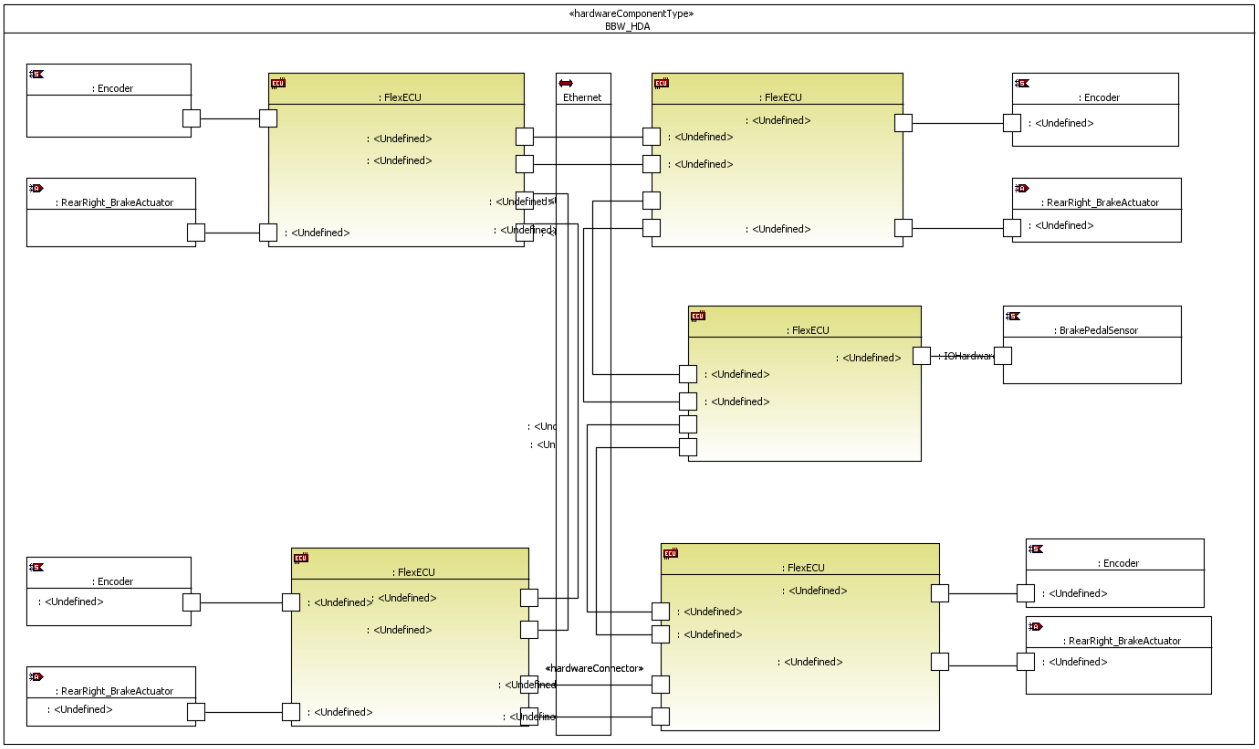


Figure 2-34: Hardware Design Architecture of the Braking System in Papyrus.

Allocation

Allocation on design level is represented in Figure 2-35, where function prototypes of the FunctionalDesignArchitecture are allocated to nodes in the HardwareDesignArchitecture.

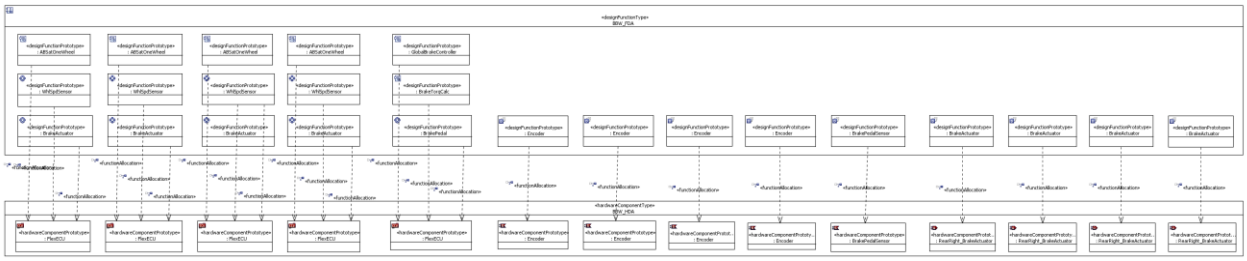


Figure 2-35: Function-to-node Allocation in the Braking System in Papyrus.

2.3.5 Implementation Level

Two variants of the Implementation level model are shown below. One made in Papyrus, and another in a dedicated AUTOSAR tool, Vector DaVinci.

AUTOSAR Software Component Template

Figure 2-36 below shows a DaVinci model of brake-by-wire. The view exposes ABS controller and actuator management for one wheel, and also electrical motor control. Figure 2-37 shows an AUTOSAR model with the core functionality of one pedal and 4 wheels with ABS control sensors and actuators.

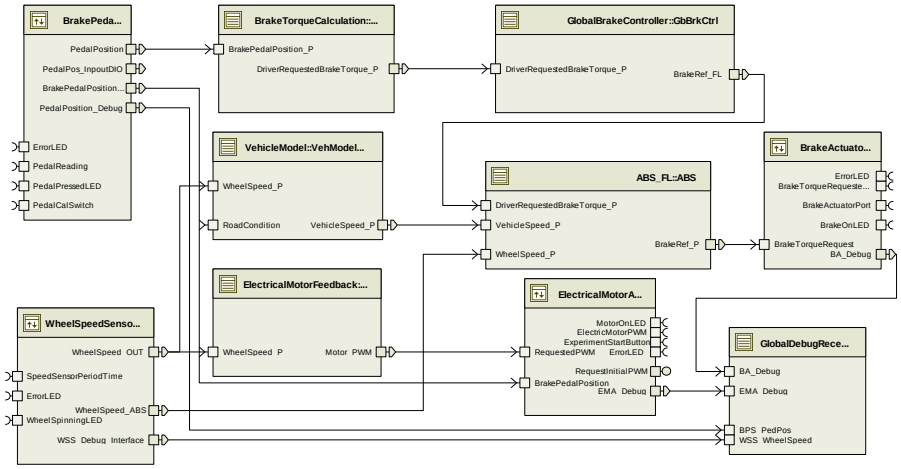


Figure 2-36. AUTOSAR Software Component Template of the Braking System

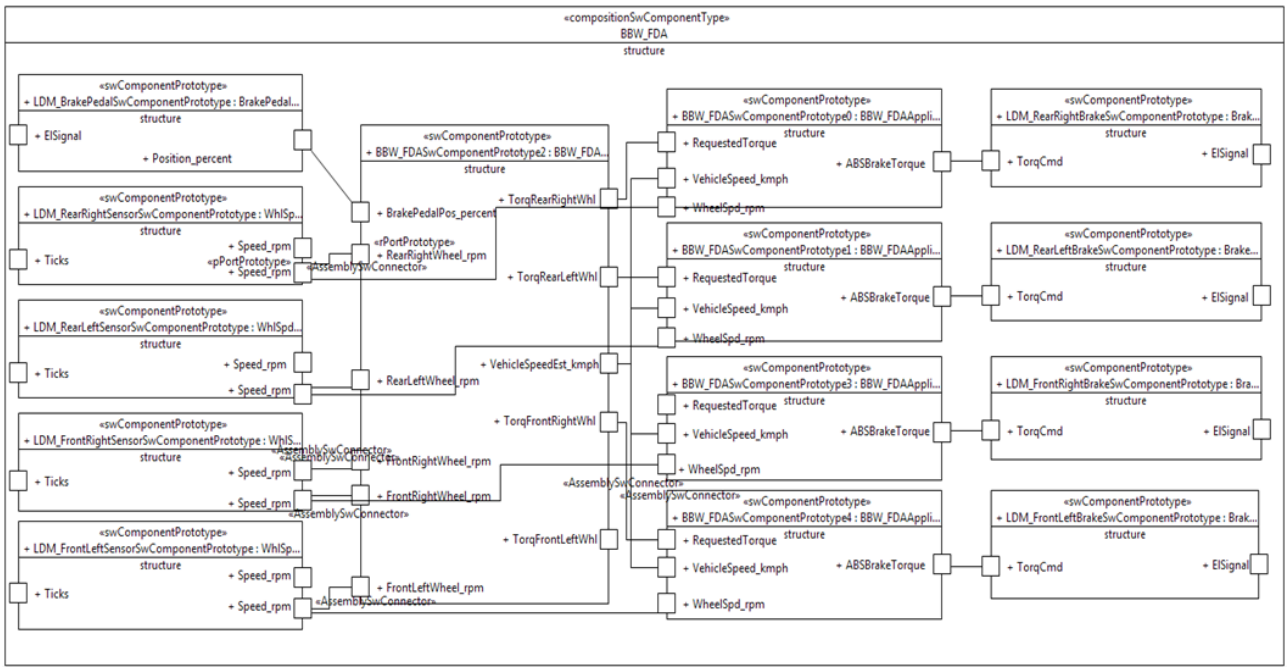


Figure 2-37. AUTOSAR Software Component Template of the Braking System

3 Functional Safety Analysis application

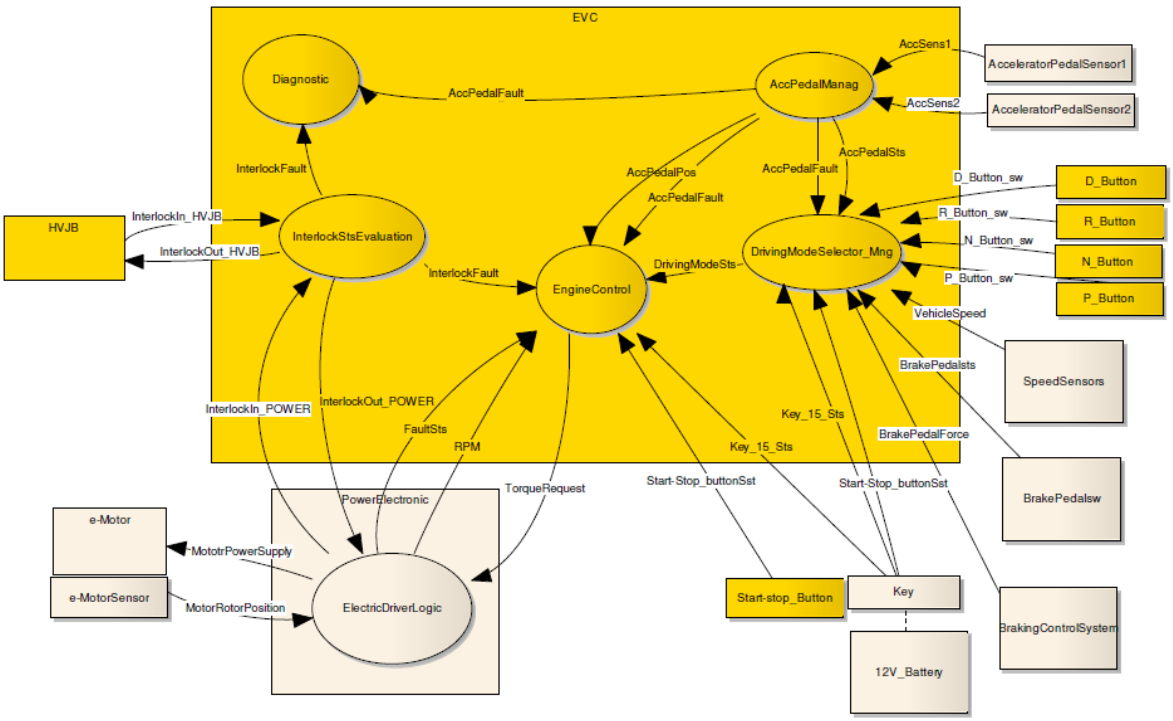
One of the goals of the WP6 is to evaluate the ability of MAENAD to support ISO 26262 safety process and safety concepts, as well as their integration with other aspects of system development. A preliminary step of the evaluation activity consists on the application of the main functional safety activities. In particular the case study selected is the "*Propulsion and power distribution*".
The Item definition, the hazard analysis and risk assessment have been performed in a preliminary way. The next step of the evaluation activity will be to integrate the main safety analysis results in the case study model and to exercise the analysis framework.

3.1 Item Definition

The first fundamental step of the safety life-cycle is the identification and description of the Item under analysis, and to develop an adequate understanding of it. This is an essential step, since the subsequent phases of safety design flow are based on the item definition and the safety concept is derived from it.

To have a satisfactory understanding of the Item, is essential to properly analyse the item itself in terms of input(s)/output(s), functionality, interfaces and, how the item interacts with the vehicle and/or with the environment.

In the following diagram the Item's element, together with the item interface elements have been collected. Moreover the boundary of the Item has been highlighted (yellow elements):



3.2 Hazard Analysis and Risk Assessment

In the following table the main results coming from the Risk assessment, in terms of maximum level of risk associated to each hazardous event, have been summarized. Actually, the table includes only the main hazards related the propulsion part, useful for testing on test bench.

Hazard		Scenario	Hazardous Event		ASIL
Id	Description		Id	Description	
H1	Unexpected forward movement	- VehicleSpeed=0; - Key Status = ON - ePRND Status = N; - Driver on board; - Vehicle in a queue at the traffic light, with interposed pedestrian.	HE1.1	Unexpected forward movement of the vehicle, when vehicle is stopped in a queue (with interposed pedestrian), due to an unwanted application of positive torque.	C
H1	Unexpected forward movement	- VehicleSpeed=0; - Key Status = ON - ePRND Status = N; - Driver out of board; - hand brake disengaged.	HE1.2	Unexpected forward movement of the vehicle, when vehicle is left by the driver with N selected, due to an unwanted application of positive torque.	C
H1	Unexpected forward movement	- VehicleSpeed=0; - Key Status = ON; - ePRND Status = N; - Driver out of board; - vehicle pluggedIn; - hand brake disengaged.	HE1.3	Unexpected forward movement of the vehicle, when vehicle is in charging mode (pluggedIn) with N selected, due to an unwanted application of positive torque.	D
H2	Sudden acceleration of vehicle	- Medium VehicleSpeed; - Key Status = ON; - ePRND Status = N; - urban scenario (Driving on urban roads in suitable traffic condition (e.g. approaching traffic light with N selected).	HE2 .1	Sudden acceleration of vehicle when vehicle is at medium speed, with N selected.	A
H2	Sudden acceleration of vehicle	- creeping threshold < VehicleSpeed < 50 kph; - brake bedal = OFF; - accelerator pedal = OFF; - ePRND Status = D; - urban scenario: Vehicle in natural deceleration , near to pedestrian crossing (with pedestrian is crossing the road).	HE2 .2	Sudden acceleration of vehicle when vehicle is in natural deceleration, with D selected .	B

H3	sudden deceleration of vehicle	- Low VehicleSpeed; - brake bedal = OFF; - accelerator pedal = ON; - ePRND Status = R; - reverse manouvre.	H3.1	Sudden deceleration of vehicle during a reverse manouvre, due to an unwented positive torque application	**B**
H3	sudden deceleration of vehicle	- Low/Medium VehicleSpeed; - brake bedal = OFF; - accelerator pedal = ON; - ePRND Status = D; - normal driving in urban road.	H3.2	Sudden deceleration when vehicle is in dynamic conditions, at low/medium speed, with D selected.	**B**
H3	sudden deceleration of vehicle	- Low VehicleSpeed; - brake bedal = OFF; - accelerator pedal = ON; - ePRND Status = D; - overtaking manouvre.	H3.3	Sudden deceleration when vehicle is in dynamic conditions, at medium speed, during an overtaking manouvre.	**C**
H4	Unexpected backward movement	- VehicleSpeed=0; - Key Status = ON - ePRND Status = N; - Driver on board; - Vehicle in a queue at the traffic light,with interposed pedestrian.	HE4.1	Unexpected backward movement of the vehicle, when vehicle is stopped in a queue (with interposed pedestrian), due to an unwanted application of negative torque.	**C**
H4	Unexpected backward movement	- VehicleSpeed=0; - Key Status = ON - ePRND Status = N; - Driver out of board; - hand brake disengaged.	HE4.2	Unexpected backward movement of the vehicle, when vehicle is left by the driver with N selected, due to an unwanted application of negative torque.	**C**
H4	Unexpected backward movement	- VehicleSpeed=0; - Key Status = ON; - ePRND Status = N; - Driver out of board; - vehicle pluggedIn; - hand brake disengaged.	HE4.3	Unexpected backward movement of the vehicle, when vehicle is in charging mode (pluggedIn) with N selected, due to an unwanted application ofnegative torque.	**D**
H5	Sudden loss of traction	- Low/Medium VehicleSpeed; - brake bedal = OFF; - accelerator pedal = ON; - ePRND Status = D; - overtaking manouvre.	HE5	Sudden of traction during an overtaking manouvre	**B**

The following picture reports the equivalent model of the above Hazard Analysis

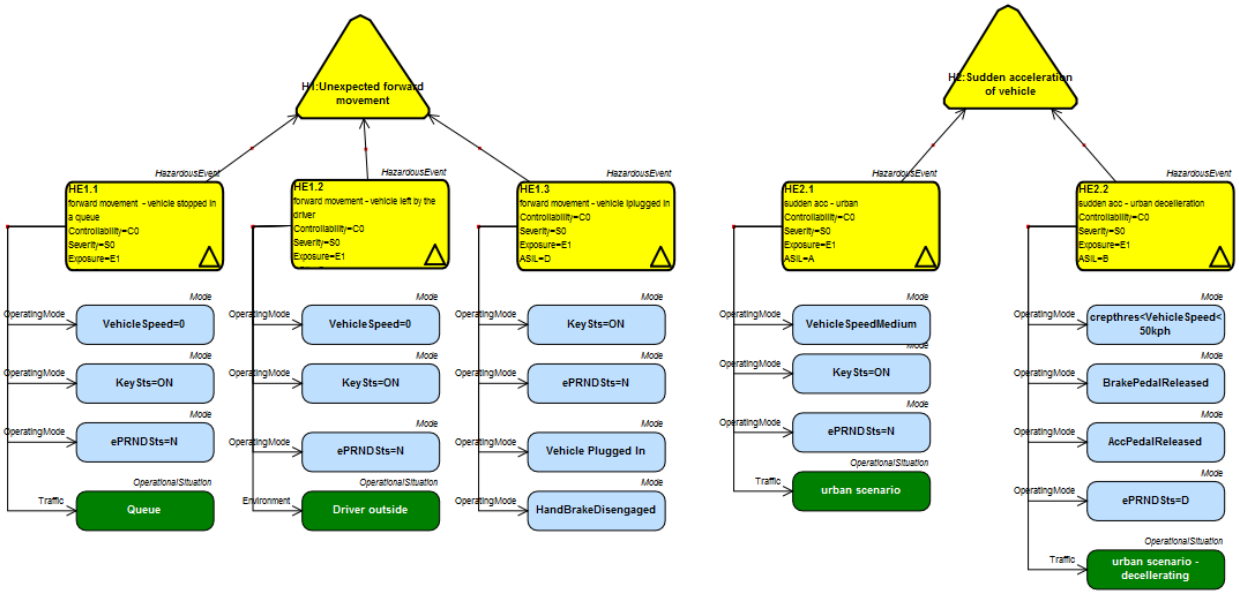


Figure 3-1: Dependability model – Hazard analysis of the Propulsion subsystem

4 Conclusion

Validator models in MAENAD are evolving and are currently representing many of the EAST-ADL constructs. Work continues to refine structural models and to develop software and hardware. In addition, analyses of the examples are being prepared, both regarding analysis of models and regarding physical fault injection. The intention is to provide a wide spectrum of MAENAD modelling and analysis concepts.