Grant Agreement: 644047

INtegrated TOol chain for model-based design of CPSs

D1.1a- Case Studies 1 (Public)

Deliverable Number: D1.1a

Version: 1.5

Date: 11 2015

Public Document

http://into-cps.au.dk

Contributors:

Francois Hantry, CLE
Thierry Lecomte, CLE
Stelios Basagiannis, UTRC
Christian König, TWT
Natalia Balcu, TWT
José Antonio Esparza Isasa, AI

Editors:

Francois Hantry, CLE

Reviewers:

Stefan Hallerstede, AU
Simon Foster, UY
Christian Kleijn, CLP

Consortium:

Aarhus University	AU	Newcastle University	UNEW
University of York	UY	Linköping University	LIU
Verified Systems International GmbH	VSI	Controllab Products	CLP
ClearSy	CLE	TWT GmbH	TWT
Agro Intelligence	AI	United Technologies	UTRC
Softeam	ST		

Document History

Ver	Date	Author	Description
0.1	02-02-2015	Francois Hantry	Initial document version
0.2	19-05-2015	JAEI	KK requirements added for first internal review.
0.3	20-05-2015	Francois Hantry	CLE-Mid-Year-Contribution
0.4	01-06-2015	JAEI	KK requirements added for case study and tools after feedback.
0.5	02-06-2015	Francois Hantry	add CLE requirements priorities and related D7.3 requirements
0.6	09-09-2015	Francois Hantry	update Deliverable plan
0.7	15-09-2015	JAEI	Updated industrial needs for the AI case study
0.8	17-09-2015	Stelios Basagiannis	Updated industrial reqs. for the UTRC case study
0.9	21-10-2015	JAEI	Agro Intelligence contribution for internal review added.
1.0	21-10-2015	Stelios Basagiannis	Updated UTRC public use case
1.1	28-11-2015	JAEI	Agro Intelligence contribution updated based on the internal feedback
1.2	02-12-2015	Francois Hantry	document structure update, introduction, summarize
1.3	04-12-2015	Christian König	Updated revised TWT case study description after reviewer comments
1.4	11-12-2015	Stefan Hallerstede	Changes to AI case study description based on internal review
1.5	14-12-2015	Stelios Basagiannis	Changes to UTRC case study description after reviewer comments

This document presents the public version of the four industrial partners' case studies. The goal of this deliverable is to give insights on the benefits and drawbacks of the INTO-CPS baseline tools from an industrial perspective, at Year 1. For each case study, the deliverable presents the description, the architecture model, the design modelling and simulation using the INTO-CPS baseline tools. The baseline tools are assessed according to the industrial needs. Industrial needs are also related to targetted INTO-CPS requirements agreed by all tools providers in [LPH+15]. At the same time, the deliverable prepares the monitoring of INTO-CPS tools evolution according to the INTO-CPS planning. The monitoring will be possible thanks to industrial needs indicators.
Other details about key points of modelling and valuable engineering work (proprietary formula, architecture or other confidential requirements) can be found in each partner's confidential Deliverable in [HL15], [EGH15], [Bas15], [KB15].

Contents

1 General Introduction

This document presents the public version of the four public industrial partners' case studies. The goal of this deliverable is to give insights on the benefits and drawbacks of the INTO-CPS baseline tools from an industrial perspective, at Year 1. The next section presents the Railway case study from CLE. Section three presents the Agriculture case study from AI. Section four presents the Building case study from UTRC. Section five presents the Automative case study from TWT. Each case study contains the public description of the industrial case, the architecture model, the design modelling and simulation of the case study using INTO-CPS baseline tools. Then the assessments of the baseline tools according to the industrial needs are shown. The industrial needs were quantified using indicators. Links between the current deliverable D1.1a industrial needs and INTO-CPS tools requirements in D7.3 [LPH+15] are also shown. At the same time, the deliverable prepares the monitoring of the evolution of the INTO-CPS tools during the next year (Year 2 and Year 3). Finally, discussion and conclusion will close the end of each case study part.

Concerning industrial needs, in all recap tables from each industrial case study, a column of priority in a MoSCoW standard is provided:

- Must achieve (mandatory)
- Should achieve (strong suggestion)
- Could achieve (e.g. necessary for industrial scalability)
- Won't but would be interesting (e.g. for optimisation of industrial process)

Changes
As a living workpackage, some updates were done in WP1 during this first year. So now, no risk appear at the end of this first year. Three main changes happened during this first year.
Firstly, KK left the INTO-CPS project and was replaced by AI.
Secondly ClearSy (CLE) gave up its case study about screen door[1]. ClearSy restarted the INTO-CPS project with its new "interlocking" case study.
Finally, in november 2015 (Month 11), the industrial partners proposed and

[1]the cause was mainly that laser beams used for train sensor were too difficult to model using 20-SIM and Open Modelica, without any optical and physical knowledge.

added some new requirements in the [LPH+15]. These requirements were proposed because some mandatory industrial needs were not fully covered by the first version of INTO-CPS requirements listed in [LPH+15]. These new requirements are the 0090,0091,0092, 0093[2].

Confidential Deliverables
For further details, key points of modelling and valuable engineering work (proprietary formula, architecture or other confidential requirements) can be found in each partner's confidential Deliverable in [HL15], [EGH15], [Bas15], [KB15].

2 Railway case study

2.1 Introduction

In railway signalling, an interlocking is an arrangement of signal apparatus that prevents conflicting movements through an arrangement of tracks such as junctions or crossings. Usually interlocking is in charge of a complete line, computing the status of actuators (switches, signals) based on signaling safety rules that are encoded as so-called "binary equations". A typical interlocking is in charge of managing $\sim 180,000$ equations (see figure 2, for instance) that have to be computed several times per second. These equations compute the commands to be issued to track-side devices: they encode the safety behavior that enables trains to move from one position to another through routes that are allocated then released.
Currently, there are attempts to find the right trade-off between efficiency of an interlocking system (availability of routes, trains' delays and cost of interlocking system) and safety (collision avoidance, derailment prevention, availability and efficiency of emergency system).

Interlocking challenges
A central interlocking is able to deal with a complete line, all decisions are made globally. However the distance between devices distributed along the tracks and the interlocking system may lead to significant delay to update devices status. Moreover this architecture, well dimensioned for metro lines, is often overkill for simpler infrastructures like tramway lines.

[2] See the details in the industrial needs and assessment descriptions parts for each case study

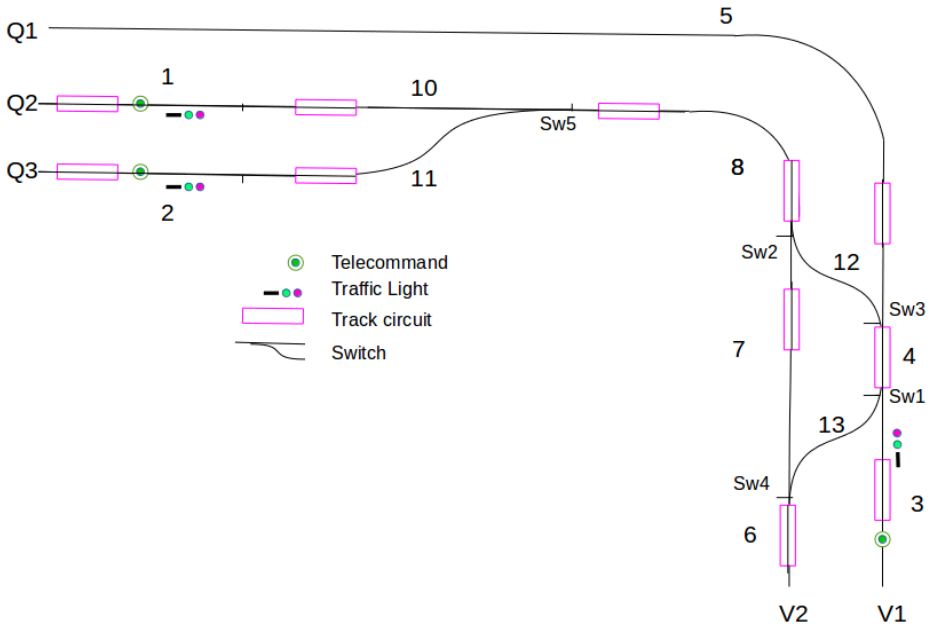

Figure 1: Partial scheme plan of a train line as seen from the sky, including track circuits, switches and Traffic lights

So there is room for an alternate solution: distributed interlocking. A line is then divided into overlapping interlocked zones, each zone being controlled by an interlocking. Such interlocking would be smaller as fewer local devices have to be taken into account a local decision could be taken in shorter time and would result in potentially quicker train transfers. However overlapping zones have to be carefully designed (a train can't appear by magic in a zone without prior notice) and some variables states have to be exchanged between interlocking systems as the Boolean equations have to be distributed accordingly over the interlocking systems.

This distribution implies several engineering problems. An "optimal distribution" i.e. the decomposition of the line into overlapping areas such as to minimize delays, availability and costs, requires a smart exploration of the design space (decomposition is directly linked with railways signaling rules). It also implies that one has to define what information has to be exchanged

between interlocking computers and how many equations have to run on any of them (20,000 equations maximum for example).

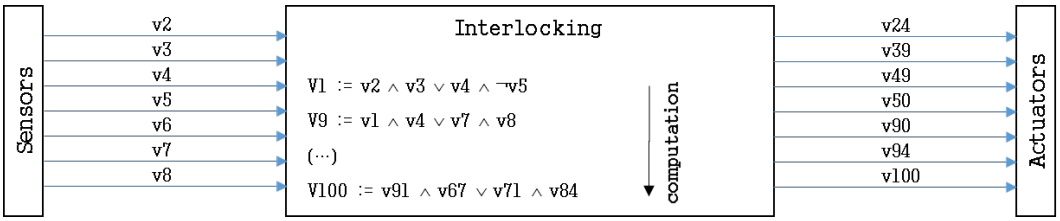

Figure 2: boolean equations that lead the signalling system

Accurate Train movement simulations and challenges
In order to have a realistic overview of the traffic of the trains and deal with safety concerns, the trains movements along the track map have to be simulated in a realistic way. The finer the movement is simulated, the more one can ensure an efficient but safe interlocking system. Usually, a train receives/consideres a Movement Authority Limit (MAL) : a stop point that the train must never overshoot. Such MAL is updated in real-time by interlocking mechanisms and communication facilities. For automated train, Automatic Train Operation (ATO) computes the best movement for reaching and stop at MAL. In parallel an Automatic Train Protection (ATP) guards against a failure of the "normal" service mode (e.g. service brakes failure, ATO software/sensor loss of train position). ATP checks that in the worst case, the MAL will never be overshot. Exploring the behavior of a "manual mode" train (with possible rollback movement) and ensuring a safe automatic protection is far from being clarified in the Railway community. Even ClearSy - which has used the ProB animator for years (a high level discrete modelling language, similar to VDM-RT) for railway use cases animations-cannot achieve, in a discrete way, a smart handling of continuous movement, of the maximal assessments of physical parameters or of the continuous time problems such as differential equations, Zenon paradox or controlling precision results.

Simulating together and in real time an efficient interlocking system with train movement would enable to enhance train performance with respect to delay or availability but keeping safe. Again, current solutions make their way but are still improvable with the help of co-simulation.

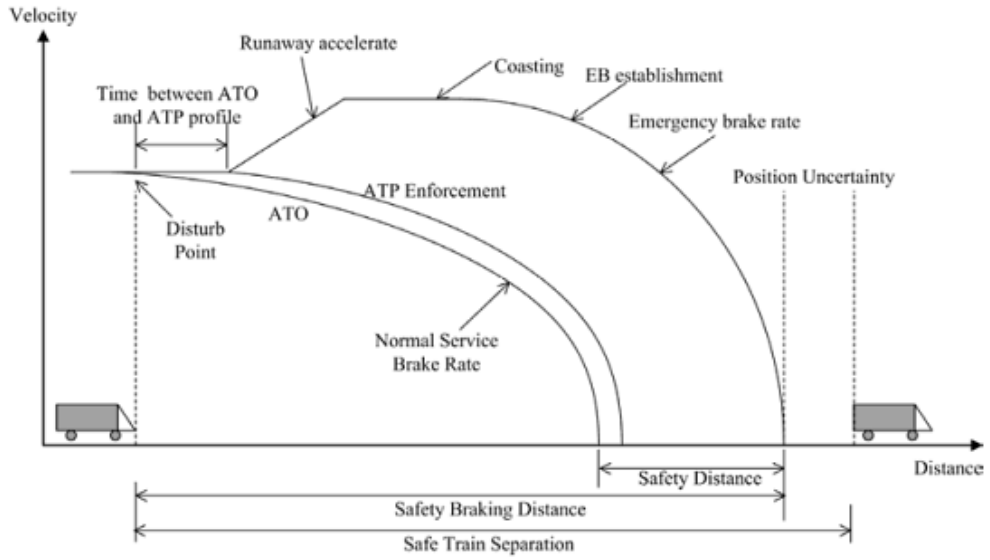

Figure 3: Usual Safe Braking Model

2.2 Case Study

For the first INTO-CPS year, ClearSy focused on the interlocking mechanism. The running example is the interlocking from a past ClearSy project in the french railway industry. This is an Interlocking which, originally, ClearSy programmed into a Schneider automata using the Ladder[3] Language. The track map is at the figure 1. There is five routes that a train can take: from $V1$ to $Q1$, from $V1$ to $Q2$, from $V1$ to $Q3$, from $Q2$ to $V2$ and from $Q3$ to $V2$. In the following we describe the several treatment and control parts of the interlocking. See figure 4 for an overview of the functionality.

Route Reservations Treatment
Any train that arrives near a telecomand point is able to ask/request a route. In figure 1, the green symbols stand for the telecommands. The interlocking system must handle all route requests, with multiple trains, even in the case of a simultaneous route requests (for instance a train A located at $Q2$, may request the route to $Q2 \rightarrow V2$, and another train B located at $Q3$ may also request the route $Q3 \rightarrow V2$, so a collision might occur without interlocking mechanism. After analysing, the interlocking system stores and transmits the elected route.

[3]the Ladder Language is a grahical language similar to cable logic

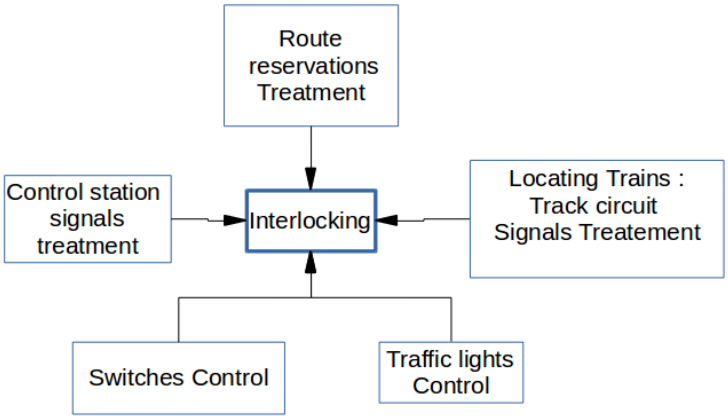

Figure 4: Control specification

Control Station signal treatments
The control station that supervizes the train station may set routes or the exploitation mode (emergency braking, Maintenance or normal service). According to the mode, the possible route may be different. The interlocking must handle such mode.

Locating Trains
In order to allow a train to follow its requested route, the interlocking must be aware whether there are trains on the route. This is concretized by track circuits (in pink color on the track map figure 1). Such track circuit is closed if there is a train on the corresponding segment of track.

Switches Control
In order to make the elected train go through the elected route, the switch must be triggered at the correct side. Otherwise erroneous trajectory or even derailment or collision is possible.

Traffic Light Control

In order to lock the elected route to trains assigned to other routes, the elected route must be not reachable from the points of the track map which are not at the entry point of the elected route. This is concretized by setting traffic light to red color, except the one of the entry point of the route, and just for the first entering train. For instance at figure 1, if a train arrives from $V1$ (traffic light number 3) and goes to $Q3$, then the traffic light number 3 will change to the green color, but the traffic lights 1 and 2 (at $Q2$ and $Q3$ are changed to the red color).

2.3 Holistic Architecture

PLC: Micro-Controllers The Progammable Logic Controller (PLC) is located on the way side. It is responsible of the control of the traffic equipement according to the interlocking mechanism. It contains a PIC 32 microcontroller, some physical input/output pins and some relays. It also enables ethernet communication. For the first year only one PLC has been used. Figure 5 shows an overview of the control architecture.

Signal transmissions
The signal transmissions between the PLC and the wayside equipement (track circuits, traffic light, switch, telecommand) are mainly based on TOR cables (a signal with only two modes). For safety purposes, all signals ares supported by a relay triggered on the corresponding cable. All relays take duration for switching their state. The trains request their routes to the PLC via beacons on board and on the wayside (telecommand). The control station communicates with the PLC using ethernet protocol.

Safety logic redunded by cables The Safety "part" of the interlocking logic is checked on the output signals from the PLC. The checking is done physically by cable logic. Such redundancy enables to keep already checked and installed cable logic.

Motorized Switchs The switches are motorized, so it may take time (0,5 sec) for the switches to change their states.

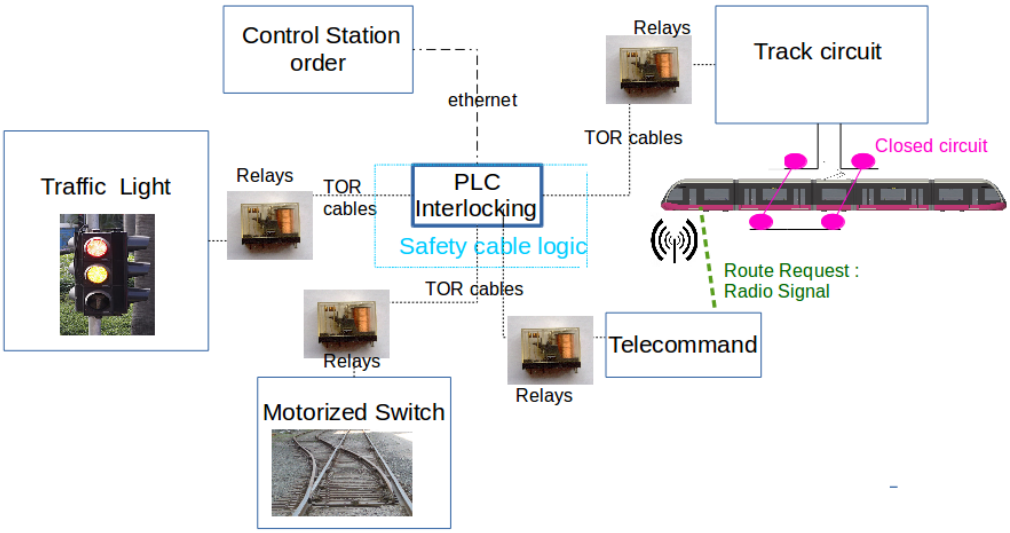

Figure 5: Control Architecture

2.4 Design Modelling

2.4.1 Discrete Event Model (DE)

In a past railway project, ClearSy had already developped such interlocking PLC, but in a graphical Ladder Language with a proprietary target hardware (Schneider automata), not on hardware PIC 32 (non proprietary) and not in a distributed version. For deploying on several pic 32, C coding is welcome. Furthermore, in order to prepare a smart interlocking distribution, high level discrete language as VDM-RT is mandatory. It is "set based" and real time oriented. This would ease the interlocking modelling and exploration of the best distribution parameters (Design Space Exploration). Additionally, VDM-RT should, at the end of the project, enable C code generation with VDM-RT real time artifacts (duration, postpone, time cycle) together with distribution facilities. The interlocking mechanism was modelled using VMD-RT. For ClearSy, one interest of the R&D INTO-CPS project, is to reuse such developped Ladder code but for a distributed ver-

sion of interlocking, together with the INTO-CPS co-simulation facilities. See figure 7 for an insight of ClearSy approach. ClearSy achieved to translate automatically from the already existing Ladder code to the VDM-RT code. See figure 7. This was done by using a code translator. We fully achieved the translation in the VDM-RT. The generated code is toward 4000 code lines. The signals and relays input are encoded as boolean variables. Thus, our case study benefits from an "industry based" proof of concept. An example of piece of the Interlocking VDM-RT code is at figure 6. Figure 6 shows the handling of delays (TON,TOFF) according to micro-step time cycle (a subdivision of the time cycle).

```
operations

public Clock :() ==> bool
Clock() == duration(MicroStepPeriod)
(
dcl res : bool := false;

–IO‘println("Entering clock");
res := false or Init.clock(MicroStepPeriod) or res;
res := TOF_0.clock(MicroStepPeriod) or res;
res := TOF_1.clock(MicroStepPeriod) or res;
.
.
.
```

Figure 6: A piece of interlocking VDM-RT dealing with micro-step time cycle and delaying functions TON and TOF

2.4.2 Continuous Time Model (CT)

Track Map
In the railway industry, it is usual to store many track map data into XML or CSV files. Such configurations may be part of the Automatic Train Operation (ATO) and Automatic Train Protection (ATP). ClearSy renders compatible the track map data for 20-SIM.

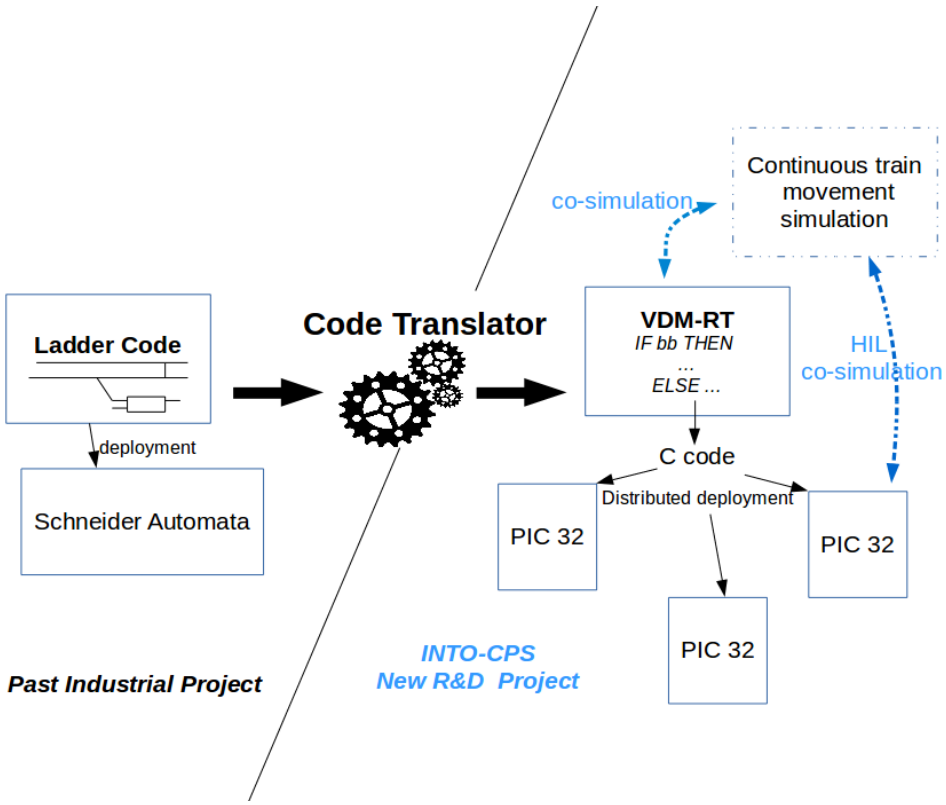

Figure 7: Enhancing past product with the INTO-CPS tools

Train Movement on a single track in 20 Sim
A continuous 20-SIM train movement were developed using 20-SIM. The trains are affected by the gravity force, the traction force and the braking force. The manual driving is simulated by a PID, that tries to make the train reach the target speed in function of the distance of the next stop point (the Movement Autority Limit (MAL) point) see Figure 8 for a high level view of the 20 Sim model.

Train movement from track to track
A train is located by a track and a position on the track. So, when a train changes from track to track the corresponding position has to be correctly reinitialized. 20-SIM enables to handle such event using the "resint(f ,init ,event , reinit)" operator that reinitializes an integral function of "f", as soon as an "event" is at true, with the new value "reinit" . At figure 9, we can see the result of a co-simulation (see paragraph below). The second curve stands for the position of the train which is a piecewise continuous curve.

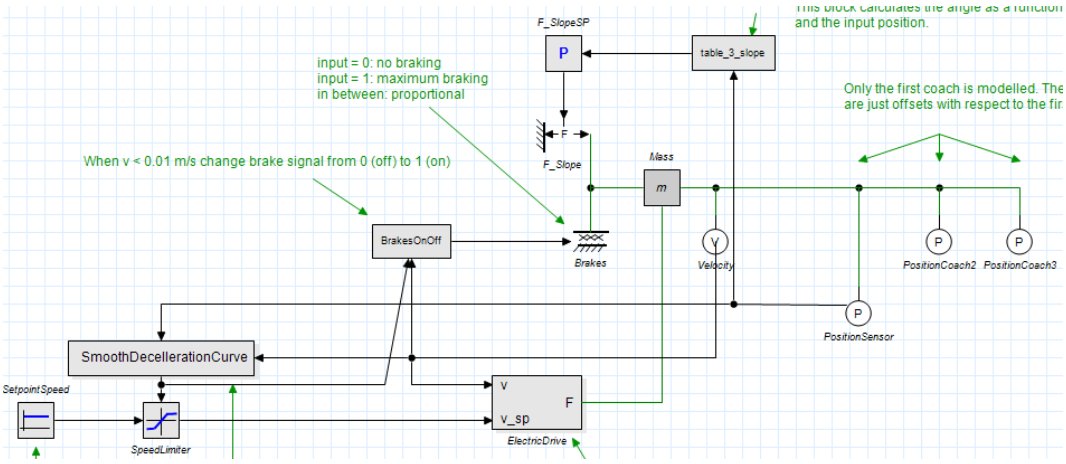

Figure 8: Single train movement model

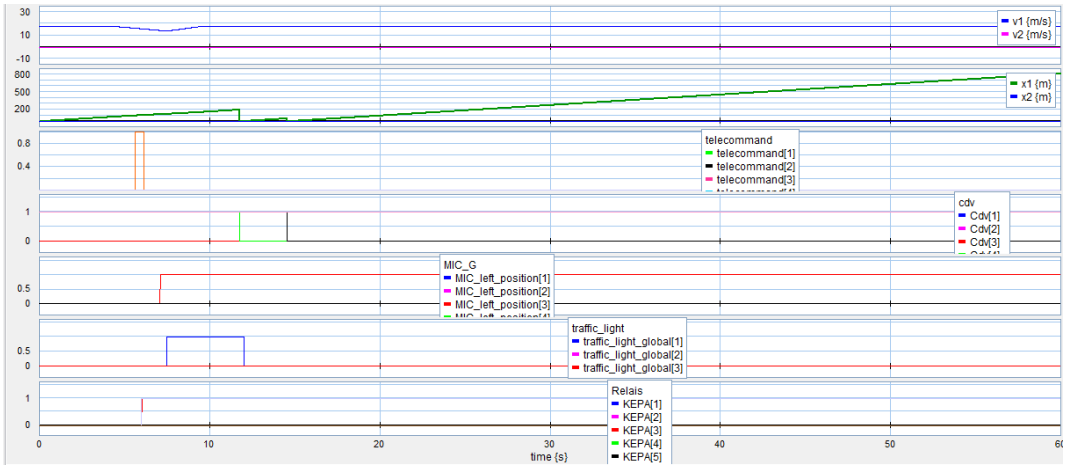

Figure 9: Train movement simulation from track to track (V1Q1)

Relay and Switch delays simulation
Part of the interlocking control equipment as relays or motorized switches spend a delay for changing their state. These delays were modelized in 20-SIM using the "tdelay" operator that enables to delay a continuous signal.

2.4.3 Co-Modeling and co-simulation

The VDM-RT Interlocking PLC (DE) and the 20-SIM Track Map + Train Model (CT) were co-simulated using the cosimulator engine Crescendo.

co-Model
The time cycle of the main VDM-RT interlocking program is 50 miliseconds. Such time cycle were modelized using the "Real Time" artifacts of VDM-RT. The signal variables shared by VDM-RT and 20-SIM were bound by a contract in the vdm.link file. It contains:

- from the continuous model → to the discrete model (interlocking PLC):
  - track circuits signals
  - switch sensors
  - traffic light sensors
  - telecommand
  - control station command
  - Route decision checking
- from the discrete model (Interlocking PLC) → to the continuous model
  - switchs order
  - traffic light order
  - route decision order

Running co-simulations
ClearSy designed several scenarios.

- *single train*
  The first one, At Figure 9, there is the simulation of a single train starting at $V1$ and going to $Q1$ (see figure 1 for locating $V1$ and $Q1$ on the track map).
  On the first top graph, the speed decreases while the train arrives near the traffic light, then when the traffic light changes to the green light (see sixth graph below), the speed increases. The traffic light changed to the green light because, when the train reached the telecommand beacon, the requested route is transmitted to the PLC. This signal is treated by the PLC and then the PLC triggers the corresponding route "release" using the (K)EPA3 signal (see last graph below). When the route signal EPA3 is triggered, the PLC triggers the change of the switch 3 called MIC3 (called Sw3 at the figure 1) (see fifth graph).
- *two Racing Trains*
  The second example is the case of two racing trains at figure 10. The

one starts at Q2 and the other starts at Q3, then they both want to reach the $V2$ direction.

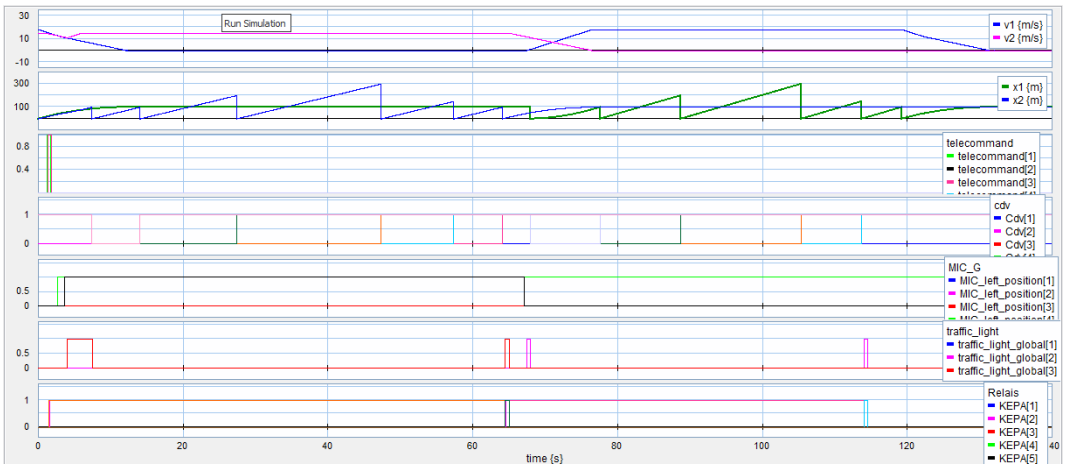

Figure 10: two racing Trains simulation (Q2V2vsQ3V2)

The $Q3$ train is first elected (signal KEPA1 at TRUE), and the switches number 4 and 5 are changed to the left. The $Q2$ train stops during toward 70 seconds. Then, It starts while the $Q3$ train has crossed the last switch 4. Then the route signal KEPA1 can be destructed and the route signal KEPA2 can be elected for the $Q3$ train.

- *two Unrelated Trains*
  At the figure 11, the first train starts at $Q2$ and goes to $V2$ the other starts at $V1$ and goes to $Q1$. Since, the both train routes does not share any track and does not cross, then, they are both allowed to continue their own route.

2.5 Industrial needs and assessments

In the Table 1, there is an overview of ClearSy's needs about the INTO-CPS baseline tools. In the following part, we detail each ClearSy (CLE) needs.

2.5.1 Cle_1: ClearSy_time_Modelling

- **Description**
  The baseline tool must enable to express delays of system's action-s/steps. The baseline tool must enable the description of delay and the

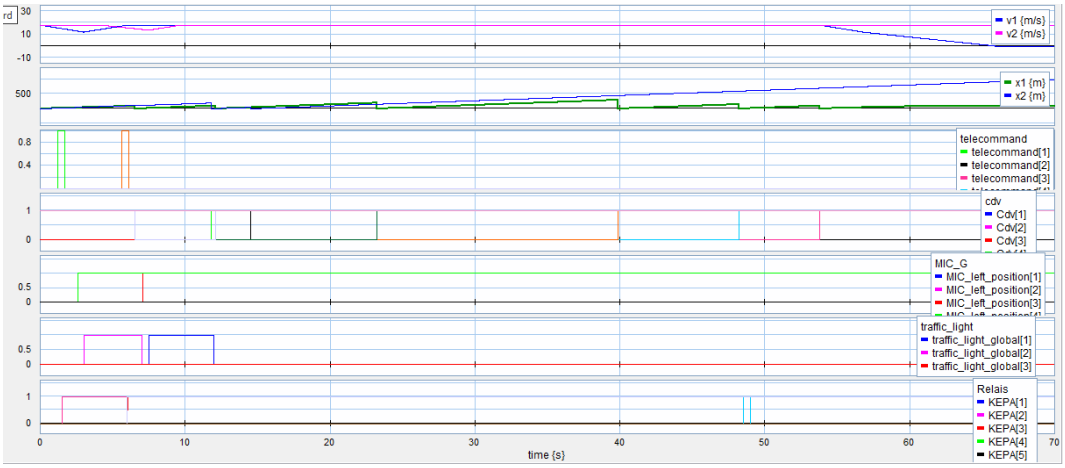

Figure 11: two unrelated Trains simulation (Q2V2vsV1Q1)

duration of physical components such as the commutation of relays and the state changes of the motorized switches. Also, it must be possible to set the time cycle of a micro controller.

- **Related Baseline Tools Requirements**
  [LPH+15] Requirements 0003-0005 are related because the modelling of time step could be done at the cosimulation level.

- **Method of verification**
  While modelling the Railway signalling system using 20-SIM, the duration of state changes of the relays and switches were modeled using the 20-SIM operator "tdelay" (Indicators: succeed: yes/no, rate of success over cases). Cycle-time of interlocking software were also modeled using a Clock operation that consumes a duration (using the VDM-RT operator "duration").

- **Assessment: achieved**
  In all our sub case studies, the modelling of delays, and cycles times were achieved in VDM-RT and 20 SIM. Indicator: 100%

2.5.2 Cle_2: ClearSy_time_Simulation

- **Description**
  The baseline tool should simulate delays/cycle time of system's actions/steps. The baseline tool should enable the description of computational delays and duration of commutation of relay in a signalling

system.

- **Related Baseline Tools Requirements**
  [LPH+15] Requirements 0003-0005 is related because the simulation of time could be done at the cosimulation level. Requirement 0024 - is important to be as precise as possible.

- **Verification Method**
  ClearSy Attempted to simulate the Railway signalling system using 20-SIM and VDM-RT. The switching of relay should be simulated. Cycle-time should be simulated (Indicators: succeed: yes/no, rate of success over cases).

- **Assessment: achieved**
  As already shown in the above co-simulation part (see graphs 9, 10, 11), the PLC time cycle and the delays of state changes for relays and motorized switched have been successfully simulated and also cosimulated. Indicator: 100%

2.5.3 Cle_3: ClearSy_time_Trigger_Modelling

- **Description**
  The baseline tools must contain, at least in one of their language, a trigger artifact based on time or delay.

- **Related Baseline Tools Requirements**
  [LPH+15] Requirements 0003-0005 is related because the simulation of time could be done at the cosimulation level.

- **Verification Method**
  VDM-RT were tested for coding time based trigger. (Indicators: succeed: yes/no, rate of yes over cases).

- **Assessment: achieved**
  It has been possible to set the duration of a computation unit using the VDM-RT CPU class. Cycle-time of interlocking software has also been modeled using a Clock operation that consumes a duration (using the VDM-RT operator "duration"). The "TON" and "TOF" functions - from the LADDER code that enable to handle delay before triggering a signal or to hold on a signal during a moment- have been handled into VDM-RT by computing explicitely at each micro-step duration, the total elapsed time. IN function of the elapsed time the TON/TOF operators decide the triggering. Indicator: 100%

2.5.4 Cle_4: ClearSy_time_Trigger_Simulation

- **Description**
  The baseline tool should simulate the trigger based on time or delay. In order to be able to synchronize with relay, and to express cycle time, it should be possible to simulate in the software logic the clock or delay enabling to postpone the triggering of operations.
- **Related Baseline Tools Requirements**
  [LPH+15] Requirements 0003-0005 is related because the simulation of time could be done at the cosimulation level. Requirement 0024 is important to be as precise as possible.
- **Verification Method**
  VDM-RT will be tested for simulating the code of time based trigger. (Indicators: succeed: yes/no, rate of yes over cases).
- **Assessment: achieved**
  As already shown in the above co-simulation part (see graphs 9, 10, 11), the PLC time cycle and the delays of order have been simulated and co-simulated using VDM-RT. It has been possible to simulate the duration of a computation unit using the VDM-RT CPU class. Cycle-time of interlocking software has also been simulated using a Clock operation that consumes a duration (using the VDM-RT operator "duration"). The "TON" and "TOF" functions - from the LADDER code which enable to handle delay before triggering a signal or to hold on a signal during a moment- have been simulated. Indicator: 100%

2.5.5 Cle_5: ClearSy_checking

- **Description**
  The baseline tool should enable to check logical consistency at the level of co-simulation.
- **Related Baseline Tools Requirements**
  [LPH+15] Requirements 00032 to 00035 are Model-checking based requirements(discrete Model-checking, continuous Model-checking and global(co-simulation) Model-checking). Their achievements are important for Cle_5 requirement industrial achievement.
- **Verification Method**
  Non collision invariant or Overall delay should be checked. For instance

the duration of switching of relay which was missing in ClearSy's first prototype and which caused an error at the testing phase on the industrial site, could be earlier found out with the help of the model checker. Indicator: succeed yes/no, rate of yes over cases
This should be done by model-checking by VDM-RT/RT-Tester/20-SIM. (Indicator: number logical of consistencies checked/ all logical consistencies)

- **Assessment: not achieved**
  At the first year, it has not been possible to use model-checking technique at the co-simulation level. However, it is possible to check invariant at the level of VDM-RT, or to inject error warning in the 20-SIM model when continuous invariant is falsified. Indicator: 25%

2.5.6 Cle_6: ClearSy_Simulation_Scalability_1

- **Description**
  The baseline tool **could** simulate real size Railway map evolution and trains movements.
- **Related Baseline Tools Requirements**
  [LPH+15] Requirement 0024, since it is important to control Simulator in order to avoid side effects from computation latency.
- **Verification Method**
  Indicator: Number of simulated tracks (and track circuits sensors), cross/joins (and join sensors) and related equations of simulation of train movement (20-SIM).
- **Assessment: achieved**
  The railway case study provided several csv files that store track map data (joins, traffic light, track circuit...) . For this year only few hundred lines of data were used. At the level of the 20-SIM model of the train movement and relays and switches, there are 183 variables and 177 equations. For this first year, the co-simulation from baseline tools (Crescendo + VDM-RT + 20-SIM) successfully handled the continuous model. Indicator: 100%

2.5.7 Cle_7: ClearSy_Simulation_Scalability_2

- **Description**
  The baseline tool **could** simulate real size railway signalling variables evolution.
- **Related Baseline Tools Requirements**
  [LPH+15] Requirement 0024, since it is important to control Simulator in order to avoid side effects from computation latency.
- **Verification Method**
  Indicator: number of signalling variables and logic are simulated into VDM-RT.
- **Assessment: achieved**
  For the first year, the VDM-RT model of interlocking provided toward 250 variables. The simulation and cosimulation of such variables handling has been successfully achieved. Indicator: 100%

2.5.8 Cle_8: ClearSy_Simulation_Exploration_Scalability

- **Description**
  The baseline tools should integrate several heterogenous models seamlessly.
- **Related Baseline Tools Requirements**
  [LPH+15] Requirements 0018-0020 about Design Space Exploration could be necessary in order to assess the maximal/minimal value from a range of parameters. The guidance requirements 0073, 0076 would be welcome.
- **Verification Method**
  Indicators: VDM-20SIM-Crescendo elapsed Time that the simulator consumes in order to assess the maximal/minimal value from a range of test/ a state space; number of parameters/equations. Use case : rollback case maximal value assessment, availability maximal value assessment, maximum duration assessment of train movement.
- **Assessment: not achieved**
  The starting INTO-CPS baseline tools do not enable to sweep a range of parameters at the level of cosimulation. Indicator: not assessable

2.5.9 Cle_9: ClearSy_Simulation_Accuracy_Confidence

- **Description**
  The baseline tool **could** make clear the mechanisms, the accuracy and confidence of the simulation. It could be possible to handle and make clear the simulation of ordinary differential equation, with discontinue acceleration. It could be possible to model, explain and simulate multi-masses movement.
- **Related Baseline Tools Requirements**
  [LPH+15] Requirements 0045, 0047, 0055, 0058, 0061, 0065 : Semantics are necessary in order to keep accuracy and confidence. Quantifiable simulation tolerance at the INTO-CPS co simulation are necessary to keep accuracy.
- **Verification Method**
  -20-SIM discontinuity handling (yes/no, accuracy/explanations)
  -20-SIM-Crescendo maximal/minimal value assessment for a range of test case at the cosimulation level, margin error (rollback, is there margin error)
  - is there a clear semantic for VDM-RT, Crescendo and 20-SIM or open modelica ?
- **Assessment: not achieved**
  The 20-SIM simulation successfully handles the discontinuity of the acceleration because of the change of the track map (and so because the slope may change), or because of an emergency braking. However, there is no margin error provided at the end of the simulation or cosimulation. Is the simulation robust ? Because the need Cle_8 is not achieved, it is not possible to assess the accuracy of the maximal/minimal value exploration for a range of test case. Indicator: 25%

2.5.10 Cle_10: ClearSy_Seamless_integration

- **Description**
  The baseline tool should easily enable integrating several heterogenous models. The co-modelling level should enable modelling a continuous train movement, model the track map (with discrete information), model the interlocking signalling software and model the electrical relays/switches.

- **Related Baseline Tools Requirements**
  The [LPH+15] guidance requirements 0067, 0071, 0076 is welcome. Help for modelling at the co-simulation level is welcome: Requirements 0049 and 0050, 0051, 0052.
- **Verification Method**
  The Crescendo gluing/orchestration engine tool has been tested.
  Indicator: co-simulation: yes/no/partially achieved, rate
  duration to "develop co-simulation"
  Is it FMI compliant ?
- **Assessment: partially achieved**
  The co-simulation of VDM-RT and 20-SIM has been successfully achieved (see above part of cosimulation). The duration of tunning the cosimulation, without including the elapsed time of debugging the individual VDM-RT and 20-SIM models was toward 40 days of work. More precisly, we have to distinguish the duration to learn the ad-hoc cosimulation machinary from Crescendo for 20 days, to the creation of the tests 5 days, to the debugging of the tests 15 days. Compared to our past project, any replay of a set of tests was manual and spent 5 days. As soon as the cosimulation technology is known (no 20 days of learning cosimulation machinary), the co-simulation is very interesting when there are multiple test replays because for the INTO-CPS cosimulation, the replay are automatic. Finally, at this first year, the baseline tools cannot cosimulate VDM-RT and Open Modelica using FMI technologies. The crescendo contracts that enable the glue between the VDM-RT and 20-SIM models are still ad-hoc. There is no FMI compatible cosimulation INTO-CPS baseline tool. Indicator: 65%

2.5.11 Cle_11: ClearSy_Distributed_Modelling

- **Description**
  The baseline tool should enable the modelling of distributed hardware, whith communication delay.
- **Related Baseline Tools Requirements**
  Is the requirements 0024 is related depends wether delays are modelled at the cosimulation level
- **Verification Method**

The 20-SIM-VDM-RT-Crescendo will be assessed against the use case of distributed interlocking (distributed communicating Hardware)

- **Assessment: Postponed to Y2**
  This assessment has been postponed for year 2, because in year 2, the distributed interlocking will be developped.

2.5.12 Cle_12: ClearSy_Code_generation

- **Description**
  The baseline tool should easily enable generating code (C) or binary (HEX) with compatible facilities, complying with safety critical standards without too much need for manual patch.
- **Related Baseline Tools Requirements**
  [LPH+15] requirements 0037, 0042, 0044
- **Verification Method**
  VDM-RT Interlocking software generation (for code execution) on Pic 32 micro controllers for simulating interlocking.
  Indicator: achieved or not, duration to set the generation
- **Assessment: not achieved**
  For the first year, the baseline tool does not enable to generate C code from the VDM-RT model but only from the VDM-PP models. Indicator: 25%

2.5.13 Cle_13: ClearSy_certification_Safety

- **Description**
  The INTO-CPS tool chain **should** provide quality arguments for a possible certification kit (or any means to ease safety case) or redundant validation chain.
  - what is the global level of confidence ? what elements are available to be used for safety case ? For each formalized modelling language (such as Open Modelica and VDM-RT) the language provider should also provide evidence that the corresponding simulator adhere to the formal semantic of their language.
- **Related Baseline Tools Requirements**
  The requirements 091 and 092 from [LPH+15] are critical for justification of well foundedness and safety handling. [LPH+15] Requirements

0045, 0047, 0055, 0058, 0061, 0065 : Semantics are necessary in order to keep accuracy and confidence. Quantifiable simulation tolerance at the INTO-CPS co simulation are necessary to keep accuracy.

- **Verification Method**
  Indicator: Safety certification kit/method: yes/no
- **Assessment: not achieved**
  At the first year, there is no available clear (formal) semantic of VDM-RT, 20-SIM or Open Modelica and Crescendo. Moreover, there is no certification that the baseline tools behave as their specified semantic. Neither the cosimulation engine, or simulation engine, do not provide tolerance margin of the resulted simulations. There is no redundant validation chain for safety purpose. There is no certification kit. Indicator: 0%

2.5.14 Cle_14: CLearSy_Failure_Modelling

- **Description**
  The baseline tool could enable modelling degraded mode at the cosimulation level.
- **Related Baseline Tools Requirements**
  [LPH+15] guidance requirement 0082 is critical. The interrupts mechanisms are also important at requirement 0056.
- **Verification Method**
  VDM-RT/20-SIM. Model Emergency braking phase. Indicator: yes achieved /no/ partially
- **Assessment: not achieved**
  It is not possible to model faulty behaviour in a CT model at the cosimulation level yet. Indicator: 0%

2.5.15 Cle_15: ClearSy_Traceability

- **Description**
  The baseline tool should enable to coherently organize requirements and systematically to warn the user about missing checking of requirements against simulation or automatic checking tools.

- **Related Baseline Tools Requirements**
  From the deliverable [LPH+15], the requirements 0089 and 0090 are critical for traceability and impact analysis. [LPH+15] guidance requirement 0074 is welcome. Requirements 0012-0017 are important.
- **Verification Method**
  Modelio testing of requirements handling (Indicator: duration to set a requirement),
  traceability (indicator : yes/no)
  easy checking (indicator duration to Set/launch checking)
  dealing with versions , indicator : yes /no
- **Assessment: Partially achieved**
  indicators:
  duration to set a requirement w.r.t. internal ClearSy tools: writing a few Excel requirements: 10 min, writing the same Modelio requirements: 12 min.
  There is traceability facility, but not between a requirement and a piece of code (some area in the code), or a document (system, Hardware) and not documented in the INTO-CPS project yet.
  An easy checking is possible in theory but not documented in the INTO-CPS project yet.
  Indicator: (50%) .

2.5.16 Cle_16: ClearSy_3D animation

- **Description**
  In the baseline tool 20-SIM it is possible to have a 3D animation of the progress while being simulated. It would be essential to keep this kind of functionality in the multi-model FMI based co-simulation as well. The main goal is to visualize track slope but in any direction. The usual track slope description is in the train movement direction, but it can be also interesting to have the slope between two opposite [4] wheels on a same bogie. This may provide technical details in order to face centrifugal force by using such slope. Having such 3D modelling and animator should also be available for visualizing other simulation results from other tool. This should be done, by providing a 3d animator interface using FMI/FMU standard.

[4] at each side of the track

- **Related Baseline Tools Requirements**
  From the deliverable [LPH+15], the requirements 0093 is critical.
- **Verification Method**
  There is or not such available 3D-animator using FMU (yes or no)
- **Assessment: Partially achieved**
  indicators:
  There is embedded 3D animator in 20-SIM. However, no 3D-animator compatible with FMI is provided by the "starting" INTO-CPS baseline tools yet. Indicator: (50%) .

2.6 Discussion

Point about needs at the end of the first Year
According to the above industrial need assessments summarized at table 1, basic needs are already fulfilled since they are mainly achieved by individual mature baseline tools for simple cases. However, as soon as needs are toward distributed and real time concerns (for instance VDM-RT code generation) the future INTO-CPS tool chain is necessary. Similarly, the following problems should be faced using future INTO-CPS tool chain:

(1) an industrial scalable Design Space Exploration, testing or model checking mechanisms,
(2) the co-simulation standardization by using FMI standard, by providing co-simulation method guideline,
(3) providing quality/certification argument.

Having the above assessment results in mind, it appears that there are mainly two benefits for ClearSy to use the INTO-CPS tools.

First benefit of using INTO-CPS technology: Automatize and render the tests robust, realistic and early

Second benefit of using INTO-CPS technology: Optimizing and ensuring safety
The INTO-CPS tool chain will also help ClearSy to assess best parameters for optimizing availability but still ensuring safety. More precisely, the second benefit is to enhance our past interlocking product with non proprietary

Table 1: ClearSy's Needs

Needs.	Category	Sub Cat.	Insight	Priority	Status	Vers.	Accept.
Cle_1	Functional	System	express delays	M	Year 1	1	Achieved
Cle_2	Non-Functional	System	simulate delays	S	Year 1	1	Achieved
Cle_3	Functional	Software	model trigger time	M	Year 1	1	Achieved
Cle_4	Functional	Software	Simulate trigger time	S	Year 1	1	Achieved
Cle_5	Non-Functional	Safety	checking logical reqs.	S	Year 1	1	Not Achieved
Cle_6	Non-Functional	Efficiency	Simulate track map	C	Year 1	1	Achieved
Cle_7	Non-Functional	Efficiency	Simulate signal logic	C	Year 1	1	Achieved
Cle_8	Non-Functional	Efficiency	Easy Space exploration	C	Year 1	1	Not achieved
Cle_9	Non-Functional	Safety	simulator Accuracy	C	Year 1	1	Not Achieved
Cle_10	Non-Functional	Tools	Integrability of Models	S	Year 1	1	Part. Achieved
Cle_11	Functional	Tools	Distributed Systems	S	Year 1	1	to-do Year2
Cle_12	Functional	Hardware	Generate (C or Hex)	S	Year 1	1	Not Achieved
Cle_13	Non-Functional	Safety	Certification kit	S	Year 1	1	Not Achieved
Cle_14	Functional	Safety	Degraded mode	C	Year 1	1	Not Achieved
Cle_15	Non-Functional	Safety	Organize reqs.	S	Year 1	1	Part. achieved
Cle_16	Non-Functional	Visual.	3D	S	Year 1	1	Part. achieved

PLC (pic 32) and in a distributed approach. At the middle of the INTO-CPS project, the INTO-CPS tool chain will enable to assess optimal interlocking distribution, optimal parameters for emergency braking, including manual rollback case.

2.7 Conclusion

At the end of this first year, using INTO-CPS baseline tools, ClearSy achieved the modelling of the interlocking algorithm together with a co-simulation of the whole hybrid system: Interlocking "plus" the real movement of the trains, the changes of the wayside and tracks equipments. This has been done by using VDM-RT for the interlocking algorithm (the discrete part of the model), by using 20-SIM for the movement of the train and Crescendo for co-simulation. Clearsy assessed INTO-CPS baseline tools with respect to its own industrial needs. According to ClearSy 's assessments, although the "starting" INTO-CPS baseline tools appear ad-hoc and not fully "synchronized", the future INTO-CPS tool chain should pave the way for a complete, scalable and mature co-modelling and development tool for hybrid system. Next year INTO-CPS results appear very promising (FMI compatible co-engine and simulators, space exploration). With the help of the next year (Y2) INTO-CPS Tool chain, ClearSy will terminate the distributed version of interlocking, and will try to assess optimal distribution of the interlocking, and optimal parameters of rollback and forth and finally will try to model fault from the emergency braking. Mainly Clearsy envision two main benefits of using the INTO-CPS technology: (1) Automatize and render the tests robust, realistic and early of its software product for critical hybrid systems, and (2) Optimizing and ensuring safety by having a precise modelling and simulation of the physical system and environment.

3 Agriculture case study

3.1 Introduction

This case study is provided by the Danish company Agro Intelligence (AI) and it is focused on the evaluation and development of an agriculture robotic platform. Figure 12 shows the first generation of the robot, a track-based

prototype. The work presented in here will be focused on a wheel-based version currently under development.

3.2 Case Study

This case study is focused on the agricultural robotic platform Robotti. An early version of this robot can be seen in Figure 12. Robotti is designed as a low cost, semi-autonomous machine to apply different kinds of soil treatments through agricultural tools called implements. Examples of implements could be weeding or row cleaning tools.

Figure 12: Front view of the first generation Robotti.

The robot was initially conceived to be deployed and operated in fields structured in rows. An overview of the working environment is shown in Figure 13. This figure shows the start position where the robot is normally deployed and it represents with arrows the trajectory it follows. The robot transits from one row to the adjacent one in the turning areas. The field in which the robot is deployed is not necessarily flat and it could present different kinds of slope changes. In addition to the plants and the soil that has to be treated, there are other external elements in the environment that can hamper the normal operation of the robot. These are obstacles of different sizes that require different kinds, such as birdnests or humans in the way. Small obstacles can be dealt with autonomously by the robot by handling the implements height, while big obstacles demand a machine full stop.

Figure 14 shows a specific kind of implement for row cleaning purposes. This implement has a linear actuator mounted in the diagonal of a parallelogram.

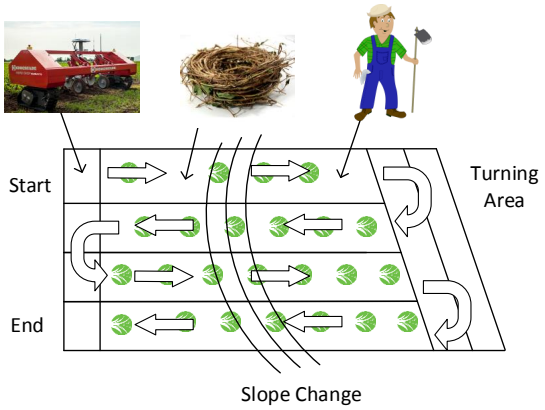

Figure 13: Example of a field where Robotti could be deployed.

When the linear actuator expands as a result the implement is lifted and it is not in contact with the soil. Figure 15 shows the same kind of implement when the robot is deployed in the field and the implements lowered. In the agricultural domain there are many different kinds of implements for different purposes. The ones considered in this case study are row cleaning implements.

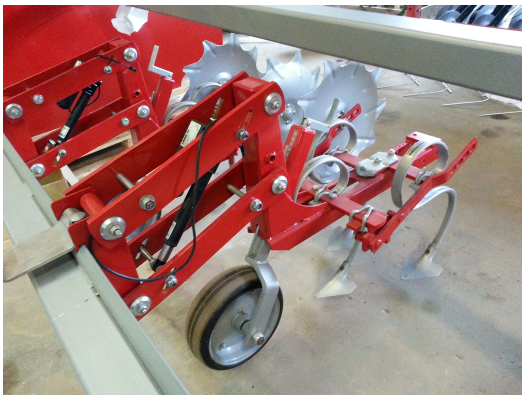

Figure 14: Implement in a workshop setup. The black bar is the linear actuator responsible for lowering and lifting it.

3.3 Holistic Architecture

In order to deliver the services specified in the use cases the robot needs a control logic infrastructure that handles both the system and the interaction with the environment. Robotti features a two-layer control architecture. An

Figure 15: Two different row cleaning implements mounted in Robotti and processing the soil.

overview of this architecture is presented in the SysML [Sys12] Internal Block Definition (IBD) diagram shown in Figure 16. The blocks presented in there are:

High-Level Controller: is responsible for the overall control of the robot when it is deployed in the field. This must address the navigation through preloaded routes as well as processing the relevant events when obstacles are detected in front of the robot. It also determines whether the implements should be lifted or lowered.

Real-Time Kinematics GPS: is shown in the diagram as RTK-GPS and provides a high-accuracy position estimate (within error margin of 1 cm). Positioning information is used by the guidance algorithm running in the high-level controller in order to navigate the field.

Obstacle Detection: is responsible for locating obstacles placed on the robot trajectory. Once an obstacle is detected, this is classified by the high-level controller according to size and placement.

Low-Level Controller: is responsible for handling the robot motor speeds and implements position. It provides an interface for the high-level control so the robot can be steered and operated according to the working plan.

Motor Control Unit: is responsible for setting the target rotational speeds in the motors. It implements at the hardware level PID [Wes00] controllers that provide the proper motor signals based on the current speed and the target one.

Implement: is used for soil processing and it can be in two different posi-

tions: lifted or lowered. The implements are moved between those two by the means of a linear actuator controlled by the Low-Level Controller.

Safety Hardware: incorporates a safety layer implemented via hardware that allows to cut the electrical supply to the motors in order to perform an emergency stop. Even though this can be conducted independently of the software, such an event is notified to the low and high level controllers.

The robot realization incorporates additional hardware and software components. However, some of them have been removed from the diagram shown in Figure 16 in order to provide a more communicative view.

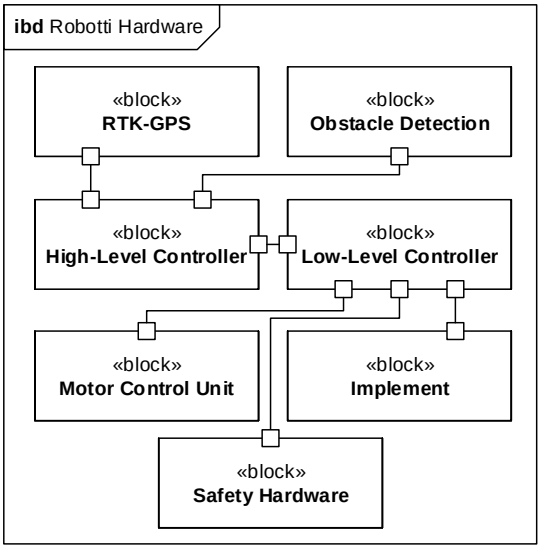

Figure 16: SysML Internal Block Definition Diagram showing the main Robotti components.

The High-Level controller is currently provided by a subcontractor. This implies that AI will not be delving into implementation details, yet it would be beneficial to have a high-level understanding of the controller architecture. An overview of the main constituent blocks of the High-Level Controller is presented in Figure 17. Additionally, it is necessary to formalize and specify the interactions between the High-Level controller and the Low-level controller that is designed and implemented in-house. The low-level controller is composed of the following blocks:

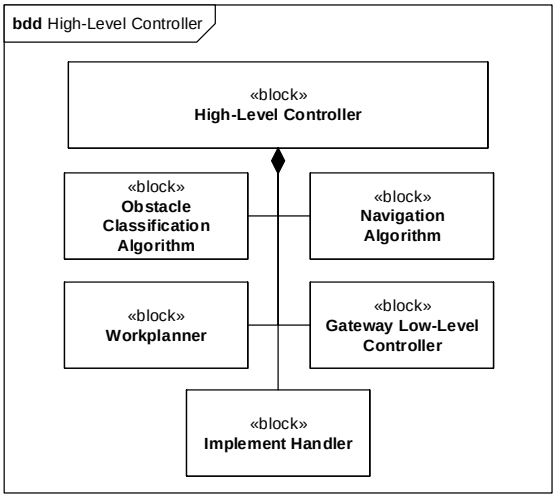

Figure 17: SysML Block Definition Diagram showing the composition of the High-Level Controller.

Safety Hardware I/O: integrates the necessary hardware circuitry in order to perform an emergency stop.

Motor Interface Handler: translates the received rotational speeds from the high-level control to the target motor rotational speeds. It interfaces the motor controller and provides this target speed as inputs.

Implement Height Regulator: receives the target implement height from the high-level controller and executes the appropriate regulation algorithm to reach it.

Gateway High-Level Controller: connects the low-level controller to the high-level controller. It mostly receives and passes on commands from the High-Level controller in order to use the motors and the implements.

3.4 Design Modelling

The models developed as part of this case study have been documented using the SysML modelling language. This notation has been used to provide different kinds of views of the system. Examples are the architectural views presented in the internal block diagrams and block definition diagrams pre-

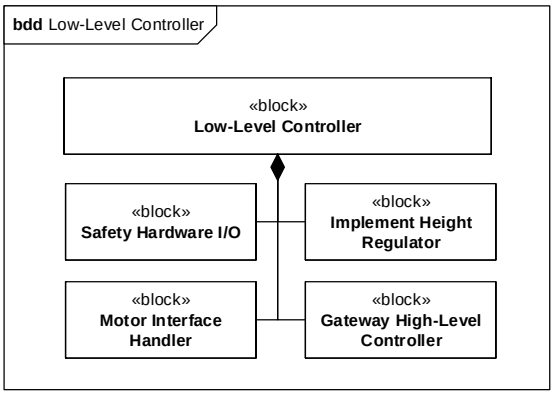

Figure 18: SysML Block Definition Diagram showing the composition of the Low-Level Controller.

sented in Figures 16, 17 and 18. Finally, state diagrams are used in Figures 19 and 20 to represent how the robot reacts to different external events toggling between different operational modes.

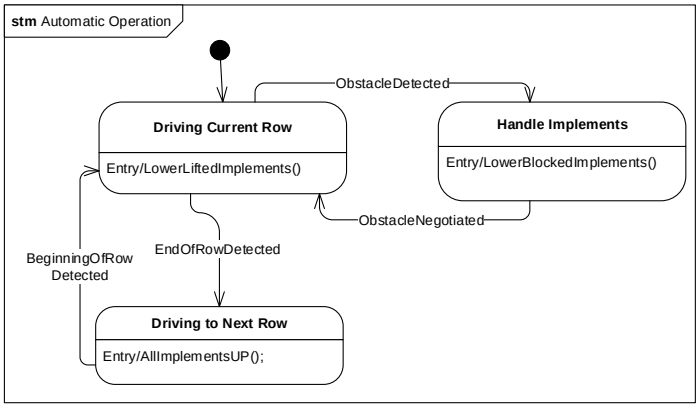

Figure 20: SysML State Machine showing the states when executing the UC Operate Automatically. These are nested to the state Automatic Operation.

3.5 Assessment of the baseline technologies

The application of the baseline technologies to the modelling of the case study has allowed to determine their suitability and their performance. This assessment led to a number of conclusions:

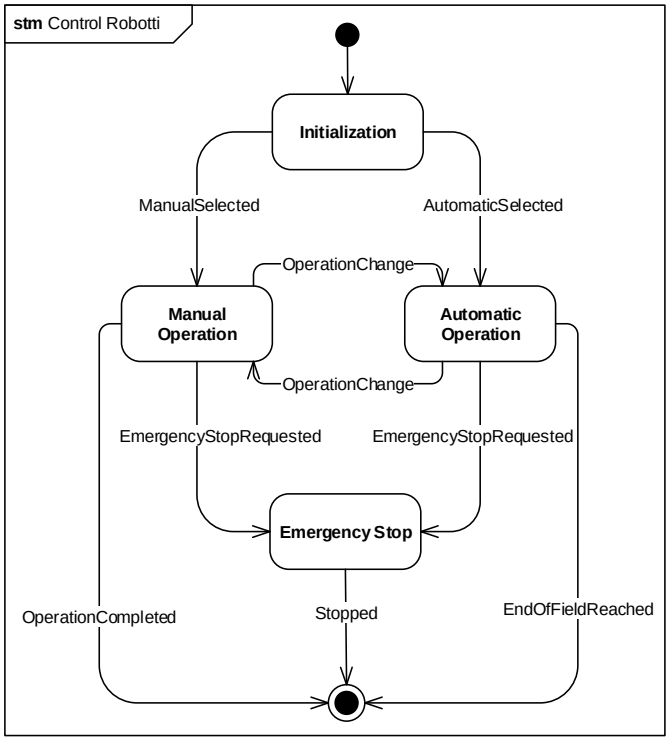

Figure 19: SysML State Machine showing the states that are part of UC Control Robotti.

- The modelling abstractions provided by VDM are very adequate to represent high-level logical control aspects. On the other hand, we believe that VDM is not the best choice when it comes to representing low-level PID controls even though these are of a DE nature. The main limitation here comes from the co-simulation low speed when PID controllers are run in VDM and the physical plant under control is run in 20-SIM.
- VDM-PP is strong when it comes to represent logical problems using the OO paradigm. This has facilitated the creation of state machine representations of the robot operation. This can be beneficial since it allows to formalize in an executable specification how the robot should operate. On the other hand, such a representation is very low level and close to the kind of implementation that could be conducted directly using a programming language. The INTO-CPS tool chain is going to incorporate the necessary facilities to make software in the loop a possibility. In that case the modeller would have to face a choice: modelling vs. direct implementation. The toolchain should incorporate the methodological framework needed in order to adopt a sound strategy in these cases.
- The Overture environment has been used to code generate the DE models into a Java executable, that could potentially be deployed already in the High-Level controller. However, we would like to use a programming language with wider support in embedded platforms such as C or C++. This means that the code generation has to be improved in order to be applicable in this case study.
- Hardware in the Loop possibilities in the VDM tool Overture have been previously prototyped [IJL14] but it is of limited use. This capability has to be further extended in order to have it applied in an industrial setting.
- The 20-SIM tool is perfectly suited to analyse the mechanical aspects of the robot under study. On the other hand, since 20-SIM is a multipurpose tool for electromechanical/physical systems modelling, it lacks specialized toolboxes for robotics modelling. Such toolboxes of ready made components could speed up the creation of robotic models and the evaluation of different kinds of sensors.
- The 20-SIM tool together with 20-SIM 4C can generate C and C++ code for different platforms. However, B&R PLCs is not among them at the moment. These PLCs are central to this case study since they

are used in the low-level controller.

- Both VDM and 20-SIM can be used to represent system aspects at various levels of abstractions. However, both modelling notations are domain specific and are designed to be used by specialized engineers. It would be advantageous to introduce the application of domain independent modelling notations such as SysML to describe how models developed either in 20-SIM or VDM are constructed. However, there are no official guidelines from the tool manufacturers on how to map the different abstractions to specific SysML constructs.
- Both VDM and 20-SIM can potentially be used to implement a model-driven engineering approach to Cyber-Physical System design. However, both tools lack at the moment the traceability structures that must be in place before implementing such a process.

3.6 Industrial needs and assessment

In order to approach the limitations presented above a number of industrial needs have been formulated. These will be addressed throughout the different Work Packages that compose the project over years two and three. An overview of these needs is presented in Table 2.

The following sections provide a description of the industrial needs for the agricultural case study. These industrial needs are formulated as requirements. Additionally, references to the Requirements IDs as identified in Deliverable 7.3 are provided where appropriate.

3.6.1 AI_1: Code generation from VDM-RT

- **Description:** The tool must facilitate code generation from VDM-RT models to C++ state machines using the GoF state pattern. The target platform for this is embedded linux and it must run either as a standalone executable or within a ROS node.
- **Method of Verification:** Code generation from models of different control logic of varying complexity. Testing of the generated code in a controlled setup and comparison with expected results based on models. Measurement: percentage of VDM-RT moduls translated; suitability as control software

Table 2: Summary of the industrial needs proposed by Agro Intelligence.

Req.	Category	Sub Cat.	Description	Priority	Status	Ver.
AI_1	Functional	Tools	Code generation from VDM-RT	M	Not Started	1
AI_2	Functional	Tools	Code generation from 20-sim	H	Not Started	1
AI_3	Functional	Tools	20-sim and SIL simulation with B&R PLCs	H	Not Started	1
AI_4	Functional	Tools	VDM-RT H/SIL simulation with B&R PLCs	H	Not Started	1
AI_5	Functional	Tools	Requirements traceability down to models	M	Not Started	1
AI_6	Functional	Tools	Requirements traceability down to realization	M	Not Started	1
AI_7	Non-Functional	Safety	Safety Assurance	M	Not Started	1
AI_8	Non-Functional	Process	Methodology embedded development	M	Not Started	1
AI_9	Functional	Tools	Model snapshot and version control	L	Not Started	1
AI_10	Functional	Tools	Simulation results snapshot and version control	L	Not Started	1
AI_11	Functional	Tools	Gazeebo integration	L	Not Started	1
AI_12	Functional	Tools	3D animation/visualisation	M	Not Started	1

- **Assessment:** Not started.
- **Related baseline tools requirements:** 0038, 0039, 0040
- **Degree of achievement:** 0%

3.6.2 AI_2: Code generation from 20-sim

- **Description:** The tool must facilitate code generation from 20-sim to C/C++ software to be deployed in B&R PLCs. The generated code from 20-sim should follow the PackML standard implemented in the library BRDK_MU[5].
- **Method of Verification:** Code generation from models of different control logic of varying complexity. Testing of the generated code in a controlled setup and comparison with expected results based on models. Measurement: percentage of VDM-RT moduls translated; suitability as control software
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0038, 0039, 0040
- **Degree of achievement:** 0%

3.6.3 AI_3: 20-Sim HIL and SIL simuation with B&R PLCs

- **Description:** The tool must facilitate Hardware-in-the-Loop and Software-in-the-Loop simulation of 20-Sim models with B&R PLCs.
- **Method of Verification:** Co-execution of different models of increasing complexity with Hardware and Software realizations. Measurement: Yes/No for HiL and SiL; Percentage of simulated cases (from total of "pure" model simulations)
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0042, 0043, 0084, 0086, 0087, 0088
- **Degree of achievement:** 0%

[5]Additional details on this library are available upon request.

3.6.4 AI_4: VDM-RT HIL and SIL simulation with B&R PLCs

- **Description:** The tool must facilitate Hardware-in-the-Loop and Software-in-the-Loop simulation of VDM models with B&R PLCs.
- **Method of Verification:** Co-execution of different models of increasing complexity with Hardware and Software realizations. Measurement: Yes/No for HiL and SiL; Percentage of simulated cases (from total of "pure" model simulations)
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0042, 0043, 0084, 0086, 0087, 0088
- **Degree of achievement:** 0%

3.6.5 AI_5: Requirements traceability within models

- **Description:** The tool must enable the traceability from requirements captured in SysML models or in an Excel Requirements Traceability Matrix (RTM) to intermediate models developed in SysML, VDM or 20-sim.
- **Method of Verification:** Traceability provided/not provided. Applicability assessed by using the tools in the case study and evaluating whether the requirements can be traced to final realizations or not. Measurement: Percentage of requirements traceable.
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0070, 0074, 0075
- **Degree of achievement:** 0%

3.6.6 AI_6: Requirements traceability down to implementation

- **Description:** The tool must enable the traceability from requirements captured in SysML models or in an Excel RTM down to system realizations.
- **Method of Verification:** Traceability provided/not provided. Applicability assessed by using the tools in the case study. Measurement: Percentage of requirements traceable.

- **Assessment:** Not started.
- **Related baseline tools requirements:** 0070, 0074, 0075
- **Degree of achievement:** 0%

3.6.7 AI_7: Safety assurance.

- **Description:** The project must provide methodological guidelines on how to use the model-based engineering approach in order to support safety-related claims.
- **Method of Verification:** Provided/Not provided. Applicability assessed by using the tools in the case study. Measurement: Percentage of safety cases supported. Completeness of safety cases.
- **Assessment:** Not started.
- **Related baseline tools requirements:** none.
- **Degree of achievement:** 0%

3.6.8 AI_8: Methodology for the development of embedded real-time systems.

- **Description:** The project must provide methodological guidelines to apply a model-based engineering approach using the tools from INTO-CPS to the development of embedded real-time systems that constitute Cyber-Physical Systems.
- **Method of Verification:** Provided/Not provided. Applicability assessed by using the tools in the case study. Measurement: Weighed percentage of activities not applicable and activities missing in proposed methodology.
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0070 - 0076
- **Degree of achievement:** 0%

3.6.9 AI_9: Model snapshot and version control

- **Description:** The tool should provide a way to keep track of the different versions of the models that compose a co-model.
- **Method of Verification:** Provided/Not provided. Applicability assessed by using the tools in the case study. Measurement: Yes/No. Degree of support in multi-simulation setup.
- **Assessment:** Not started.
- **Related baseline tools requirements:** none.
- **Degree of achievement:** 0%

3.6.10 AI_10: Simulation results snapshot and version control

- **Description:** The tool should provide a way to keep track of the different simulation results of a co-model. This could enable that at any point of time a certain set of simulation results can be linked to a concrete version of the models as well as the parameters used for the simulation.
- **Method of Verification:** Provided/Not provided. Applicability assessed by using the tools in the case study. Measurement: Yes/No. Degree of support in multi-simulation setup.
- **Assessment:** Not started.
- **Related baseline tools requirements:** none.
- **Degree of achievement:** 0%

3.6.11 AI_11: Gazeebo integration

- **Description:** The project could enable the co-simulation of high-level control models represented in VDM-RT with Robot models running in the Gazeebo environment.
- **Method of Verification:** Co-simulation of models of different complexity. Reading of sensors and control of actuators that are deployed in Gazeebo. Interaction with the Gazebo simulated world. Measurement: Yes/No. Degree of support in simulation/multi-simulation setup.

- **Assessment:** Not started.
- **Related baseline tools requirements:** 0001 - 0006
- **Degree of achievement:** 0%

3.6.12 AI_12: 3D animation/visualisation

- **Description:** The project should enable the visualisation of Robotti models similar to what is offered by the baseline tool 20-sim. This will help to understand complex behaviour expressed in the model and facilitate communication in multi-disciplinary engineering teams.
- **Method of Verification:** Provided/Not provided. Visualisation of co-simulation models. Ease of use. Quality of visualisation. Measurement: Yes/No. Weighed: Degree of support in simulation/multi-simulation setup; Time to create visualisation compared to "conventional" tool; Acceptance by engineers.
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0093
- **Degree of achievement:** 0%

3.7 Discussion

3.7.1 Development of multimodels

The modelling approach presented here could benefit from the application of multimodelling. Certain aspects of the system are too complex to have them represented in a single model. Some of these aspects could include modelling of the environment. A candidate structure for a multimodel is presented in figure 21. This multimodel is composed of several views, among them SysML representations of the systems architecture as well as the system requirements. These requirements are satisfied through cyber-physical components that are represented using different paradigms. VDM could be used as presented here to model control aspects at different levels of abstraction. 20-SIM will continue being used to represent physical aspects of the system. The SysML views on top of these models would help to implement traceability and to give an overall view of the system and models architecture. Finally, more complex models of the environment could be added in order

to better represent how the system interact with it. These models could be context models created using a Discrete Event formalism and describing how the system interact with other computer based systems. They could also be CT models of the environment that represent external factors to the system that vary over time (more detailed friction models, moving obstacles, etc). However, in order to produce such a multi-model the INTO-CPS toolchain and the integration of the differing modelling tools used in the project should be ready first.

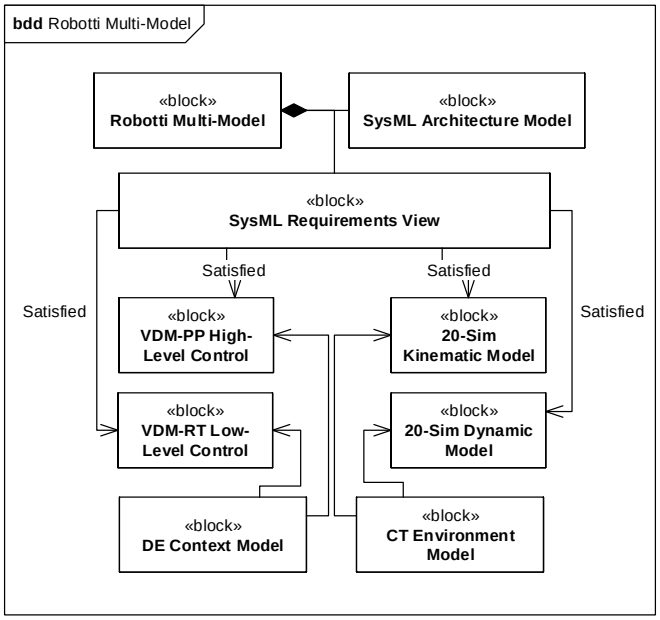

Figure 21: Structure of a possible multimodel.

3.7.2 Model deployment and Hw/Sw-in-the-Loop

The model-based engineering approach to be adopted, after the INTO-CPS toolchain has been completed, aims ultimately at the realization of a prototype of the Cyber-Physical System. This means that during the engineering lifecycle at some point the models have to be progressively transformed into hardware and software realizations. Such a transformation is conducted typically in an incremental manner, hence requiring hardware and software in-the-loop capabilities. This would facilitate the creation of different kinds of experimental and test setups. One of them and particularly relevant to

this case study would be the combination of controller implementations with physical models of the robot. This would facilitate the evaluation of the control software against a simulated environment without requiring expensive prototypes and lengthy validation and verification processes.

3.8 Conclusion

This report has presented the application of modelling to the industrial case study provided by the company Agro Intelligence. This has contributed to fulfil the objectives for year one: (a) getting trained and familiar with the baseline technologies, (b) assessing these baseline technologies and (c) proposing a number of industrial needs that can be addressed during the remainder of the project in order to improve the tools.

After an initial application of the baseline technologies to model the case study it has become clear that the INTO-CPS technology will contribute to lower the high prototyping costs, it will enable to explore the design space in a more cost effective manner and it will also facilitate the production of evidence to aid the certification process. This vision regarding the applicability of INTO-CPS has shaped and helped to formulate the industrial needs. Industrial needs related to hardware and software in the loop have the potential to facilitate the progressive realization and testing of cyber physical components. Industrial needs related to code generation have the potential to shorten development times as well as ensuring that the required level of quality is maintained.

4 Building use case

4.1 Introduction

The building case study will focus on modeling and analysis of energy and comfort systems that control the temperature of an area inside a building premise.The main objective for the building case study is to respect high level requirements that span from temperature control of rooms, zone levels and floors to energy consumption and safety operation of the complete system. One of the main reasons why the aforementioned fact remains a challenge, is the product line engineering approach currently followed by building automation industries. In the current workflow, different types of engineers contributing to the creation of the same device are involved and affecting respectively the system design. Verification of the generated models or code is enabled in stages of the product lifecycle that leave the system open to delays due to late error discovery. To this end, INTO-CPS and co-simulation solutions will not only bridge the identified gap between engineers but also, enable verification and validation of the output models early in the design phase, thus, rapidly decreasing the product lifecycle timeline while respecting system requirements.

4.2 Case Study

Current section contains a description of the building use case provided by UTRC, the modeling and simulation of it, using INTO-CPS baseline tools and the assessment of the baseline tools according to their requirements. The specific case study focus on the fan coil unit (FCU) paradigm as a proof of concept example influenced by the Heating Ventilation and Air Conditioning (HVAC) domain. The FCU device aims to control the air temperature in a room through the use of several physical components and software controllers. For the specific use-case, water is heated or cooled in a Heat Pump and flows to the Coil. A Fan directs air through the coil which is heated or cooled depending upon the coil temperature and flows back into the room. A controller is able to alter the fan speed and the rate of the water flow from the Heat Pump to the coil. In addition, the room temperature is affected by the walls and windows, which constitute the environment of the FCU.

Considering for example the single room area, the two interacting components of the room will be a) the Fan Coil Unit (FCU) and b) the room

Figure 22: Fan Coil Unit

thermostat, as shown in Fig. 23. The FCU is connected with and Air Handling Unit (AHU) for managing fresh air circulation of the building itself, as well as, a Chiller unit that provide heat or cool the water pipes of each FCU. For the specific use-case we consider the AHU and the Chiller as constants in our design that provide fixed values of water temperature and air flow parameters. Fig. 23 sketches the main components involved in controlling the temperature in a single room area. They are as follows

- an FCU is placed inside the room in order to control the heating and the flow of the air inside the room. The FCU device contains a fan for bringing fresh air inside the room which is provided by the Air Handling Unit (AHU) of the building.
- Temperature of the air is managed by two water valves each of which has varied flow controlled by the FCU in order to provide the exact set point of temperature requested by the user.
- Water heating/cooling is provided by the Chiller Unit and the Heat Pump attached to the building.
- FCU is composed by an FCU controller, a fan , an air-damper, temperature sensors and a coil.
- FCU controller is taking inputs-commands from the room thermostat that the end-user has access to as well as sensors attached to the FCU device.
- Room thermostat is the device that interacts with the end-user in order to provide room temperature preferences.

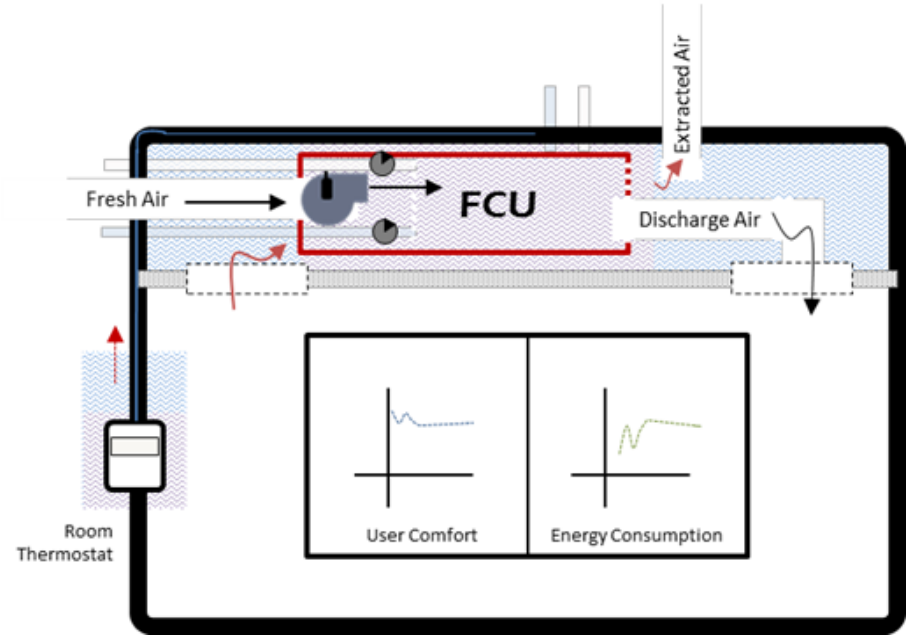

Figure 23: Room Level schematic for controlling air temperature with respect to use-comfort and energy consumption of the Fan coil unit attached to the room

- Set-points are provided as input to the FCU which accordingly regulates the valves positions of the cooled/heated water pumps and the speed of the fan in order to reach the requested temperature inside the room.
- The complete system is surrounded by the AHU and the Chiller for providing the FCU device with air and heated or cooled water.

Main objectives will be to respect user-comfort requirements in the room level while keeping energy consumption at minimum level. The user-comfort requirements will be bounded by building management policies related to time schedules, forming high level requirements of the expected system behavior.

4.3 Holistic Architecture

Current control strategy will compose a core model that initially will regulate fan speed, water valves' position and air flow speed to successfully respect user inputs provided by the thermostat, while maintaining total energy con-

sumption according to high level requirements. A specific control strategy needs to be devised in order to respect the overall objectives of the case study, namely user-comfort and energy consumption. While in current use case control complexity is low composing one control strategy to be implemented, we argue that in the building level, multiple control strategies need to be in place and synchronize their operation with user requirements.

UTRC aims to study an hierarchical control architecture in order to control and monitor the whole building performance during the INTO-CPS project. The supervisory controller *ME-LGR line* is responsible to control multiple pieces of equipment (e.g. Air Handling Unit or Heat Pump) at the site, integrate third-party equipment on a proprietary network based on BACNet communications and serve as a router to controllers on an ARCNET 156 Kbps or MS/TP network. At the lower level, there is *SE Line* controller which is suitable for single equipment applications. It operates in a wide range of environmental conditions, such as FCU or rooftop units, mechanical rooms, equipment closets, or almost at any other weather-tight location. Then, at the room/zone level *ZN* another control-unit is used to manage equipment such as FCU only for a single room case.

4.4 Design Modeling

Within the INTO-CPS baseline technologies we aim at co-modeling the discrete part of the FCU (FCU controller) and the continuous behavior of the collaborative equipment (valves, fan, FCU sensing) for specific user-comfort and energy consumption requirements. We aim at exploring modeling of control functions of the involved FCU equipment with different technologies/tools provided by the INTO-CPS partners in order to extract/develop new model results. INTO-CPS tools will also be used for describing the core functionality of the room thermostat at the design level, that is crucial part for providing inputs to the room HVAC system from the user perspective. For the specific use case, mainly we focus in co-modelling 20-Sim models that describes FCU functionality with Overture models that depict the behavior of the FCU controller.

4.4.1 Model specifications and Implementation using Modelio

Modelio tool will be used in order to develop *SysML/UML* models of components of the HVAC system for the single room case. Within Modelio, use-case

diagrams, high and low level block definition diagram (BDD), internal block diagram (IBD) and state machines are going to be created in order to adequately depict the system's architecture in a top-down behavior of all the HVAC entities.

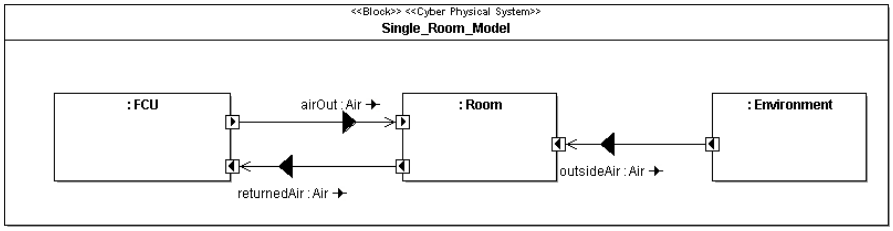

Figure 24: High level block diagram for Single room use case

Basic interactions between the actors are shown in Figure 23. From a high level modelling perspective, we consider as core components the FCU which provides air to the room, which respectively is depending upon thermal affects of the environment, as shown in Figure 24. In the internal block diagram of the single room model, the FCU is divided in the FCU controller that performs the control decisions of the FCU operation, the coil which is the main energy carrier with external power components (Heat Pump), and the fan who regulates the air flow inside the room.

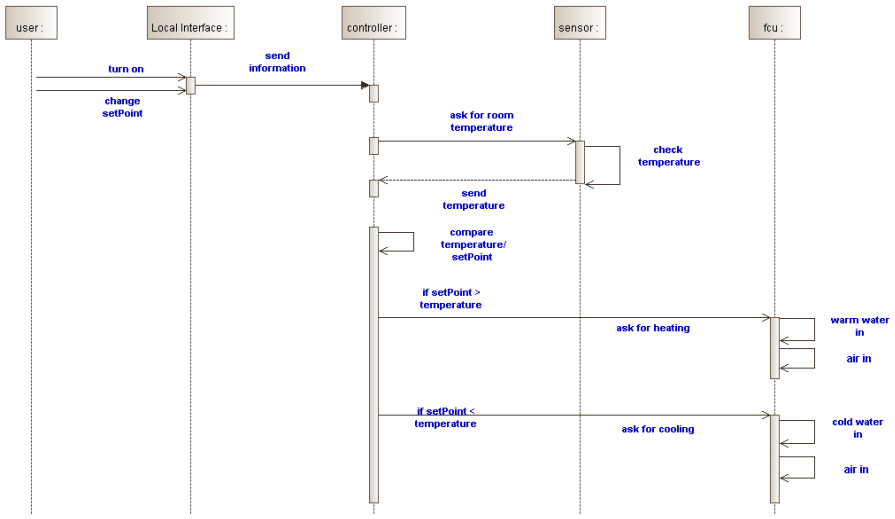

Figure 25: Initial sequence diagram for communication between user, local interface, FCU, sensor and controller

In Fig. 25 we also depict an initial sequence diagram for the communications

executed between the different actors of the use case. The controller is considered as the main communication recipient, as it will manipulate the FCU operation depending on values received, both from the user, the environment temperature and the internal parts of the FCU. At the init state, as soon as use provides inputs of the desired temperature, the controller calculates the Room Air Temperature (RAT) and checks the Temperature up or low threshold that regulated the FCU operation. Based on this comparison the controller will turn the FCU on or off, checking periodically whether the corresponding thresholds (tolerance) are respected.

4.4.2 CT Model specifications and Implementation using 20-Sim

The 20-Sim tool will be evaluated and reviewed for modeling continuous behavior of the FCU components.At this point, FCU components include valves that control the flow of the water, the air damper of the FCU unit for controlling the air flow, the coil that provides the heat or cool to the water pipes and the heat exchange between the environment and the FCU. Certain thermal functions that describe heat balance in the single room case environment needs to be taken into account inside the model. In 20-Sim the model was built based on continuity of mass and heat balance of various components. The variables and parameters as well as governing equations are defined.

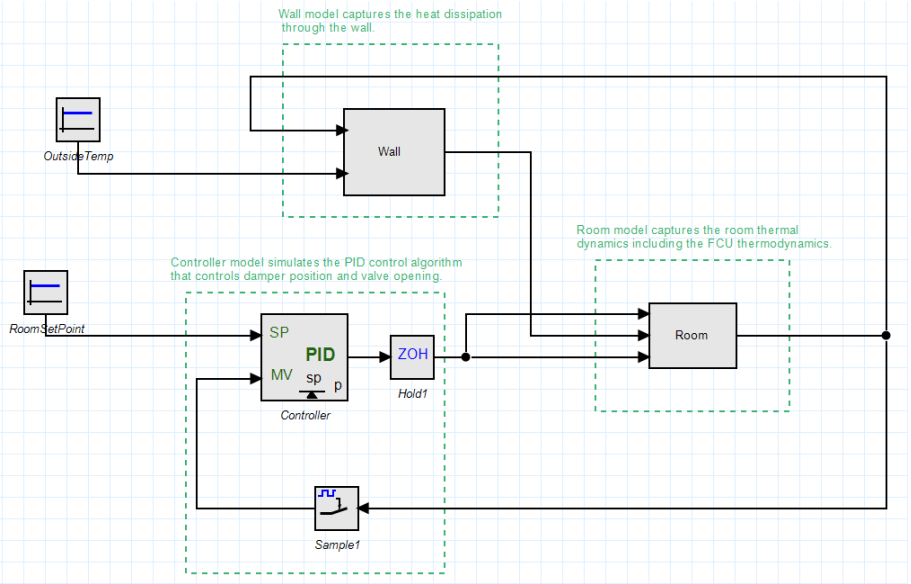

Figure 26: 20-Sim model example of the building use case study

As shown in Fig. 26, the 20-Sim model describes a single room case with temperature controlled by an FCU. In this model, the room model captures temperature dynamics including the FCU thermal effects using the coil, fan and damper. The wall model captures the heat dissipation between the room air temperature (RAT) and outside temperature. The FCU components are controlled using the PID control algorithm, where sensor is modeled to sample the continuous signal of RAT and pass it to the PID controller. Fig. 27 shows the behavior of different controlled variables of the model, such as room air temperature (RAT), room air temperature set-point (RATsp), entering water temperature (EWT) and supply air temperature (SAT).

As shown in Fig. 27, there is a high correlation between EWT, SAT and RAT due to the thermal coupling between them. In our case, the PID controller requires a settling time of 1 hour to reach to the set-point, which is acceptable for the building automation case study. However, this settling time is part of a user comfort requirement, which can be changed by retuning the PID parameters.

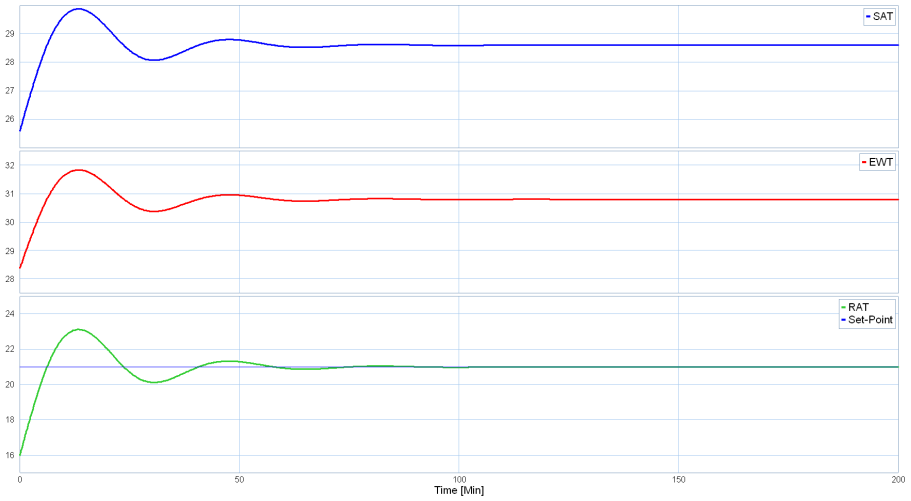

Figure 27: Simulation results of Single room 20-Sim model for SAT, EWT, RAT Temperature versus time (min)

4.4.3 CT Model specifications and Implementation using OpenModelica

In this section we describe the model created in OpenModelica for evaluating the aforementioned FCU use case. In our case, a Modelica model is a set of differential and algebraic equations, and/or discrete events. A user can

build a Modelica model by defining equations or extending existing models primarily for representation of their continuous behavior. In current, there are several commercial Modelica-based programs, similar to OpenModelica, that provide a graphical interface for users to build, manage, and execute models (such as Dymola, SimulationX, and AMESim).

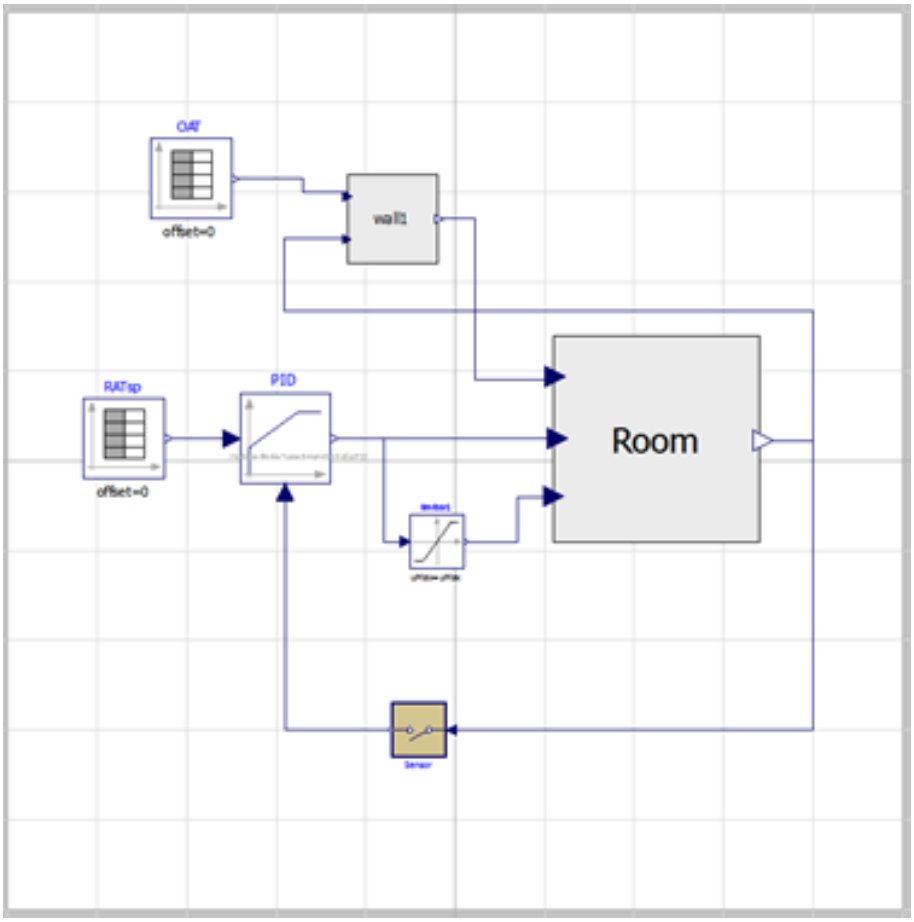

Figure 28: Graphical view of room model in OpenModelica

A first version of an OpenModelica model is developed, related to a room equiped with an FCU in which room air, walls, FCU and FCU controller along with their interactions are modelled. In the OpenModelica model, thermal affects are based on the same thermal equations shown in 4.4.2. Similarly to the 20-Sim case, in OpenModelica these equations are expressed internally as a set of differential algebraic equations by Modelica during the compilation.

Through OpenModelica simulation results, we review the heating mode when

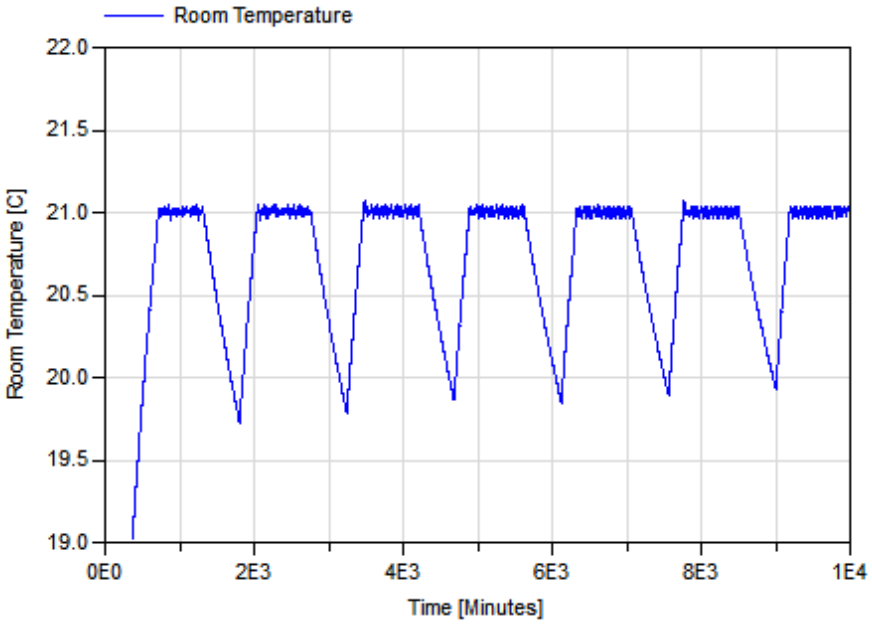

Figure 29: Behavior of room air temperature in seven day period

the FCU is required to provide warm air increment the room's temperature. Figure 29 shows the room air temperature variations in a seven day period. At time zero, it was set to an initial value of 16^0C and the FCU works to bring it to the set point by monitoring the valve opening and the fan speed. The set point was set to 21^0C during office hours and to 16^0C otherwise. Outside of working hours the room temperature drops and increases again the next day.

A one day operation is presented in Figure 30 focusing on the behavior at the FCU level. The water temperature leaving the coil (EWT) oscillates around 30.8^0C depending on the load required to meet the set point. The PID receives feedback from the room in terms of air temperature and based on the difference (set point room air temperature) it generates a signal that translates into fan speed (red profile) and valve opening to bring the difference to zero.

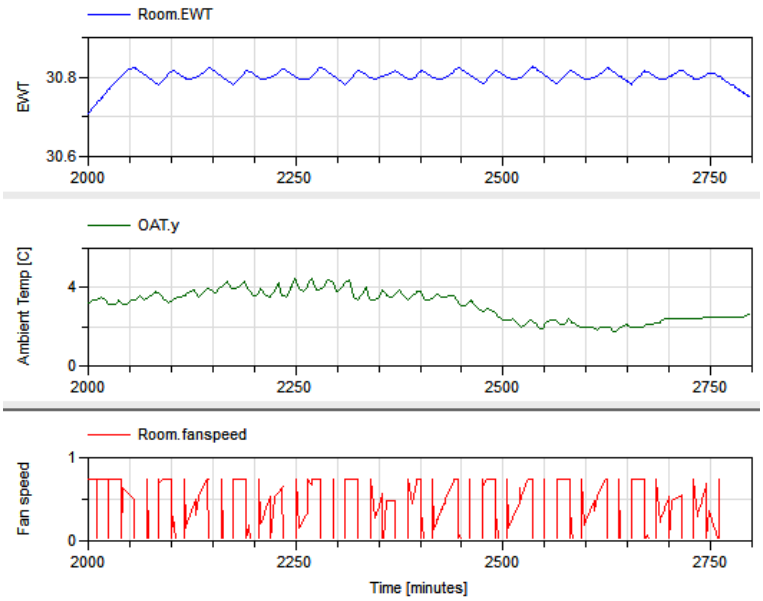

Figure 30: Entering water temperature (top), outside ambient temperature (middle), and fan speed (bottom) during one day operation

4.4.4 DE Model specifications and Implementation using Overture

As already mentioned in previous sections, the discrete part of the co-simulation in the building case study will be based upon the Overture tool. We model the FCU controller of the single room case study using the VDM-RT language. We define the variables of the controller with initial values set as required by the use case (for the set point of the FCU and the tolerance), as well as the instance variables for the rest of the FCU components, namely the damper, the valve and the sensor.

For model initialization, we instantiate the variables with certain values contrary to our use case requirements. The main part of the FCU controller is described by the control loop in the VDM language. The reader can observe the controller's decision for the device operation when the tolerance value (temperature threshold) are met or not.

In Fig. 31 we define the controller's operation with respect to the system wide VDM file. We aim in the next 2 years of INTO-CPS to incorporate real system parameters in the VDM-RT models in order to assess accurately and efficiently the controller's behavior against real data taken from FCU device operation.

```
system System
instance variables
public static controller : [Controller] := nil;

            s : Sensor; -- sensor delcaration
            a : Actuator; -- actuator delcaration
  cpu1 : CPU := new CPU(<FP>, 5E3);
operations
public System : () ==> System
System () ==
(
            s := new Sensor();
            a := new Actuator();
            -- controller delcaration
            controller    := new Controller(s,a);
            -- hardware platform delcaration
            cpu1.deploy(controller, "Controller");
);
```

Figure 31: VDM system file for PID controller

4.4.5 Co-Model specifications and Implementation using Crescendo

```
-- shared design parameters
sdp real HPWaterInTemp;
sdp real HPWaterOutTemp;
sdp real initialCoilTemp;
sdp real AHUSupTemp;
sdp real qLoad;
sdp real OATMin;
sdp real OATMax;
sdp real initialWallTemp;

sdp real setPoint;
sdp real tolerance;


-- Monitored variables (seen from the DE controller)
monitored real roomAirTemperature;

-- Controlled variables (seen from the DE controller)
controlled real damperOpeningValue;
controlled real valveOpeningValue;
```

Figure 32: VDM model interacting with Crescendo co-simulator

The FCU controller (DE) and the Single Room model (CT) were co-simulated using the co-simulation engine provided by Crescendo, as shown in Fig. 32. 20-Sim is used to capture the CT model. Whereas, DE for the FCU controller is captured using a VDM-RT model, as shown in Fig. 31.

In order to communicate between DE and CT model, we have modified the 20-Sim model to identify the sharing input/output signals. In this case the controller model in 20-Sim is replaced with a connection contract to VDM model. In addition, the VDM model captures the PID algorithm, the hardware specifications and sensor/actuator models, as shown in Fig. 31. In the

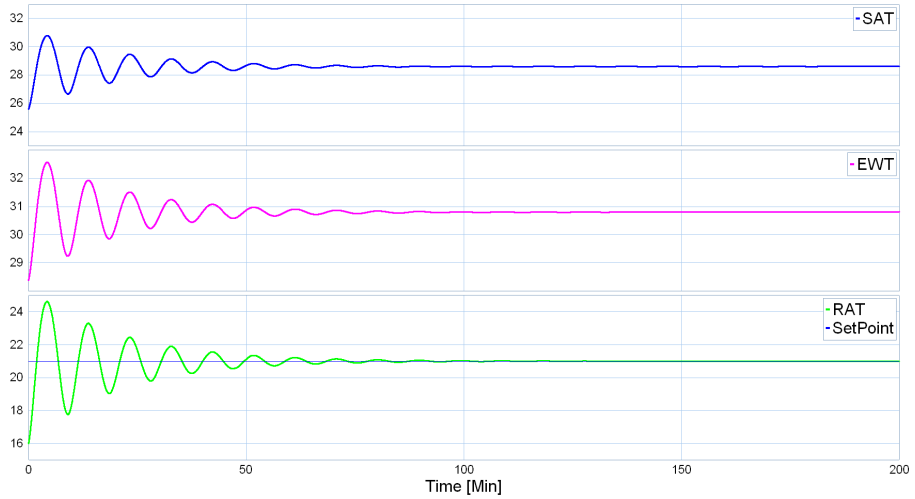

Figure 33: Results of the 20-Sim model interacting with the Crescendo co-simulator

co-simulation contract, we have identified RAT and RATsp as DE inputs, and valve position, and fan speed as DE output signals.

Fig. 33 shows the simulation result for room air temperature similar to the 20-Sim model in Fig. 27. We observe that the settling period (first 1 hour) has more control oscillation due to the discretization that reflects the real hardware specification. Fig. 34 shows the discrete and continuous simulation time with default co-simulation settings.

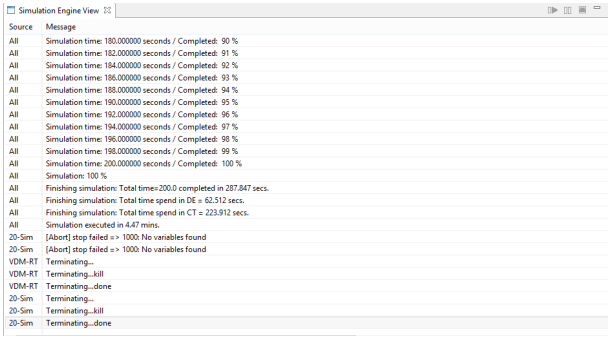

Figure 34: Results of Crescendo co-simulator using the 20-Sim and the VDM-RT model

4.5 Industrial needs assessment

In the current level of specification for the building case study we reviewed the baseline INTO-CPS technologies for modeling continuous, discrete components of the FCU in the single room scenario, while co-simulating the 20-Sim and VDM-RT models using the Crescendo tool. For the first year, we argue that the INTO-CPS baseline technologies have shown a great importance to UTRC as there is clearly a benefit for new results based on co-simulation of HVAC heterogeneous systems. In this section we proceed with a first assessment of the requirements as shown in Table 3 evaluating the INTO-CPS toolchain as described above. Initial requirements are defined according to core functionality for the single room case, where we aim to co-simulate the FCU main component of the room. At this stage requirements are tool-oriented related to the INTO-CPS technologies used to study the use case.

4.5.1 UTRC-Req-001: Simulation Completeness

- **Description**
  INTO-CPS baseline tools must offer simulation completeness (for continuous, discrete and co-models) based on the allowable modeling parameters selected by the end-user.
- **Related to Baseline Tools Requirements**
  Requirements 0003-0005 are related because the modeling of time step could be done at discrete, continuous and co-simulation level
- **Method of Verification**
  Simulation completeness impose the used tools to produce simulation results with respect to user choices of modeling a rich set of discrete or continuous phenomena (at a certain level) while maintaining and updating simulation log files during the simulation execution. This requirement has been validated through the successful continuous, discrete and co-model simulations taken by INTO-CPS tools applied on the building case study.
- **Assessment:** Achieved
  In all of our models in the building case study, the simulation completeness was shown using OpenModelica, 20-Sim, Overture and Crescendo.
  Indicator: 100%

4.5.2 UTRC-Req-002: Simulation Delays

- **Description**
  INTO-CPS baseline tools should execute simulations by providing extra time delays at minimum. With extra time delays we describe simulation executions where simulation solvers might exhibit various delays into acquiring results.
- **Related to Baseline Tools Requirements**
  Requirements 0003-0005 are related because the modeling of time step could be done at discrete, continuous and co-simulation level
- **Method of Verification**
  For the building case study and the current single room FCU models, simulation delays were negligible for all the used tools. As shown in section 4.4.2 and 4.4.3 results related to EWT, RAT and SAT were produced real-time with not extra delays. Specific delay requirements have been assessed successfully by the Overture tool for DE simulation.
- **Assessment:** Achieved
  In all of our models in the building case study, no simulation delays were encountered when executing simulations using OpenModelica, 20-Sim, Overture and co-simulation using Crescendo. Indicator:100%

4.5.3 UTRC-Req-003: Model Parameterization for Validation Testing

- **Description**
  Models imported in the INTO-CPS toolchain should be allowed to be edited by the user with respect to general simulation parameters.
- **Method of verification**
  Requirements are verified through the model experiments and parameterization with different initial values to show validity of the tool's operation. As a requirement to be respected from the majority of model-based design tools, INTO-CPS toolchain offers extended support for model parameterization.
- **Assessment:**Achieved
  In all of our models in the building case study, models could be easily parameterized in order to achieve validation testing during simulations for 20-Sim, OpenModelica and Overture tool. Indicator:100%

4.5.4 UTRC-Req-004: Interactive Simulation based on User inputs

- **Description**
  Ability to execute interactive simulation by incorporating user's inputs. While in simulation the tool is expected to wait for user inputs that are predefined either as constants (e.g. in data sources), user might like to interact with the simulation scenario while observing real-time unwanted behaviors.
- **Method of Verification**
  Currently the user-interactive simulation feature is not supported by any of the INTO-CPS tools. This though can be partially supported offline by defining certain data sources of user values depicting the scenario that the user would like to evaluate.
- **Assessment:** Partially
  In all of our models' simulation executed in the building case study, user interactive simulations are achieved at offline mode. Indicator:60%

4.5.5 UTRC-Req-005: Safety Assertions

- **Description**
  Safety logic and requirements should be enabled to be checked in an assertion style validation method in INTO-CPS. Assertions should describe using a user-friendly format an undesired behavior of the model.
- **Method of Verification**
  In the building use case we partially checked assertions of certain requirements using assertions in the Modelica model provided by the OpenModelica tool. OpenModelica allowed us to define undesired RAT temperatures identifying quickly unwanted behavior of the modelled FCU controller. Currently not all tools support the specific requirement.
- **Assessment:** Partially
  Only in the OpenModelica tool assertions formulas can be defined inside the model in order to raise warning when those are violated. Indicator:25%

4.5.6 UTRC-Req-006: Control Inputs/Outputs Simulation

- **Description**
  Ability to simulate control input and output signals for the majority of signal type values in both continuous and discrete models.
- **Method of Verification**
  Requirement has been addressed successfully as both in simulation and co-simulation the user could evaluate different control signals as inputs to the FCU models while obtaining real-time the output signals of the simulation engines.
- **Assessment:** Achieved
  All tools evaluating the models developed for the building case study were able to simulate continuous and discrete signals. Indicator:100%

4.5.7 UTRC-Req-007: CPU cycle simulation and co-simulation

- **Description**
  The INTO-CPS baseline tool should be able to simulate CPU cycles required to execute instructions imposed by models
- **Related Baseline Tools Requirements**
  Requirements 0003-0005 is related because the simulation of time could be done at the co-simulation level. Requirement 0024 is also important to be as precise as possible.
- **Method of Verification**
  The specific requirement is evaluated through the Overture tool and the VDM-RT language where we have modeled the FCU controller. VDM-RT allows the simulation of CPU cycles when executing VDM-RT instructions defined inside the model as certain operation .
- **Assessment:** Achieved
  Simulation and co-simulation results of sample CPU cycles required to execute instructors of the FCU controller have been successfully obtained. Indicator:100%

4.5.8 UTRC-Req-008: Design Space Exploration

- **Description**
  Design space exploration techniques should be applied to both contin-

uous and discrete models when executing a series of simulations

- **Related Baseline Tools Requirements**
  Requirements 0018-0020 could be necessary in order to assess the maximal/minimal value from a range of parameters.
- **Method of Verification**
  This requirement should be verified using simulations and functional testing. Currently there is not support for design space exploration within the baseline tools.
- **Assessment:** not achieved Currently the design space exploration (DSE) is not support within the INTO-CPS toochain. Models created for the building case study have not been evaluated using DSE in order to determine optimal value parameters based on random simulations (ongoing work). Indicator:10%

4.5.9 UTRC-Req-009: Simulation Accuracy and Precision

- **Description**
  INTO-CPS should be able to provide a varied level of simulation accuracy with a user-selected variable precision. Such a premise will enable increment of results confidence during the simulation executions
- **Related Baseline Tools Requirements**
  Requirements 0045, 0047, 0055, 0058, 0061, 0065 : Semantics are necessary in order to keep accuracy and confidence. Quantifiable simulation tolerance at the INTO-CPS co simulation are necessary to keep accuracy.
- **Method of Verification**
  Having in place semantics for INTO-CPS baseline tools is of paramount importance in order to enable simulation accuracy and precision. Currently INTO-CPS tools have a limited support with respect to simulation and co-simulation manipulation (e.g. rollback or test case evaluation) with accurate results.
- **Assessment:** not achieved
  In the current evaluation study tools have depicted certain levels of precision in the simulation experiments. Expect to increase simulation results accuracy in co-simulation in the next years of the INTO-CPS project. Indicator:25%

4.5.10 UTRC-Req-010: Scalable Simulation

- **Description**
  INTO-CPS toolchain should adequately produce simulation results in large continuous, discrete or co-models where complexity raises.
- **Method of Verification**
  As this verification will depend on the model complexity we argue that the specific requirement will be re-evaluated in the year 2 of INTO-CPS project.
- **Assessment:** not achieved
  In the current evaluation study tools have depicted certain levels of simulation efficiency when tackle a certain level of model complexity. Indicator:not assessable with current models

4.5.11 UTRC-Req-011: Code generation

- **Description**
  INTO-CPS framework should allow code generation with respect to selected hardware platform.
- **Related Baseline Tools Requirements**
  Related to requirements 0037, 0042, 0044.
- **Method of Verification**
  Code generation from the VDM-RT model on ARM processor for executing FCU commands
- **Assessment:**not achieved
  INTO-CPS baseline tools support stand-alone code generation for certain embedded hardware devices. Currently limited support for ARM processors particularly for co-simulation models. Indicator:25%

4.5.12 UTRC-Req-012: Requirements Traceability

- **Description**
  INTO-CPS framework should enable requirements traceability from high level requirements specified within text or translated to properties in certain formats.

- **Related Baseline Tools Requirements**
  Related to requirements 0089 and 0090.
- **Method of Verification**
  Requirements management through Modelio have not yet included in the INTO-CPS framework. Currently this is ongoing work as INTO-CPS is set to enable requirements traceability through Modelio plugin.
- **Assessment:** not achieved
  Currently stand-alone support for requirements traceability within SysML diagrams within Modelio. Indicator:10%

4.5.13 UTRC-Req-013: 3D Animation

- **Description:** FMU 3D Visualization during simulation and co-simulation
- **Method to Verify:** This requirement should be verified using simulation executions of the building case study. We expect to 3D visualize different parts of the FCU device and their actuation while the controller takes decisions for its operation.
- **Assessment:** not achieved
  Currently only 20-Sim provides 3D visualization of simulation. Indicator:10%

4.6 Conclusion

In this section we presented the application of INTO-CPS modeling tools to the industrial case study provided by UTRC. We have shown initial results from continuous, discrete model simulations as well as co-simulation results for the single room use case controlled by an FCU device. Advances in the INTO-CPS technology in the next years will enable additional co-simulation features, more accurate exploration of the simulation results, efficient design space exploration contributing to rapid acceleration of model development and product realizations. Such a fact will definitely support the generation of additional evidences to aid the certification process especially in devices controlled by international standards.

Table 3: UTRC's requirements

Req.	Category	Sub Cat.	Insight	Priority	Status	Version	Assess
UTRC-Req-001	Non-Functional	System	Simulation Completeness	M	Year 1	1	achieved , 100%
UTRC-Req-002	Non-Functional	System	Simulation Delays	S	Year 1	1	achieved , 100%
UTRC-Req-003	Functional	Software	Model Parameterisation for Validation Testing	M	Year 1	1	achieved , 100%
UTRC-Req-004	Functional	Software	Interactive Simulation based on User inputs	S	Year 1	1	Partially achieved , 60%
UTRC-Req-005	Non-Functional	Safety	Safety assertions	S	Year 1	1	Partially, 25%
UTRC-Req-006	Non-Functional	Efficiency	Control Inputs/Outputs Simulation	C	Year 1	1	achieved , 100%
UTRC-Req-007	Non-Functional	Efficiency	CPU cycle simulation and co-simulation	C	Year 1	1	achieved , 100%
UTRC-Req-008	Non-Functional	Efficiency	Design Space Exploration	C	Year 1	1	not achieved, 10%
UTRC-Req-009	Non-Functional	Tools	Simulation Accuracy and Precision	C	Year 1	1	Partially, 25%
UTRC-Req-010	Non-Functional	Tools	Scalable Simulation	S	Year 1	1	not achieved
UTRC-Req-011	Tools	Scalability	Code Generation	S	Year 1	1	not achieved
UTRC-Req-012	Functional	Software	Requirements Traceability	S	Year 1	1	not achieved, 10%
UTRC-Req-013	Functional	Software	3D animation	S	Year 1	1	not achieved, 10%

5 Automotive Case Study

5.1 Introduction

Electric vehicles are becoming more and more important, and various models are offered on the market. While the technology advances, there are concerns about the market breakthrough, as electric vehicles have some drawbacks, compared to conventional vehicles based on internal combustion engines. One of the main drawbacks is the limited range of electric vehicles, which is typically around 120 to 150 km, and the long time required to fully recharge the batteries (up to several hours).

5.2 Case Study

5.2.1 Specifications

The main goal for the automotive case study in INTO-CPS is to create a tool for range prediction of electric vehicles. This shall be done by simulating the drive train of the electric vehicle and other relevant electric auxiliary loads such as the air conditioning, while integrating information from weather, topography or traffic. The range optimization assistant shall enable the driver of an electric vehicle to select a route which offers the maximum range of the vehicle with the given electric battery, in order to alleviate the drawbacks of the limited range.

For the present case study, we distinguish between two scenarios :

- *Offline:* A user gives as input a desired start and end destination. A specific route is simulated and the energy consumption is calculated.
- *Online:* The online mode is activated as soon as a route was chosen and the car starts moving in a certain direction. In this situation, real-time data coming from the vehicle like torque requirements, current speed, available battery power (state of charge) etc can be accessed.

  Here the following factors play an important role in the work flow:

  1. The state of the car (state of charge of the battery) is updated periodically . If the energy level deviates from the prediction (done in the off-line mode) and the energy level isn't enough to reach the destination another route is simulated. If no route is found

under the energy constrains the user is alerted. If a new route was found the user is alerted and has to choose whether the route should be updated.

2. The environment (weather or traffic situation has changed) is seen as a trigger that causes a new simulation to start and generate another route. The user has to actively agree upon the old route being overwritten by the newly generated route.

3. The driver deviates to another route and thus triggers a new route that needs to be generated. Automatically, the old route is overwritten by the new route .

An illustration of the triggers inside the system is shown in figure 35 below.

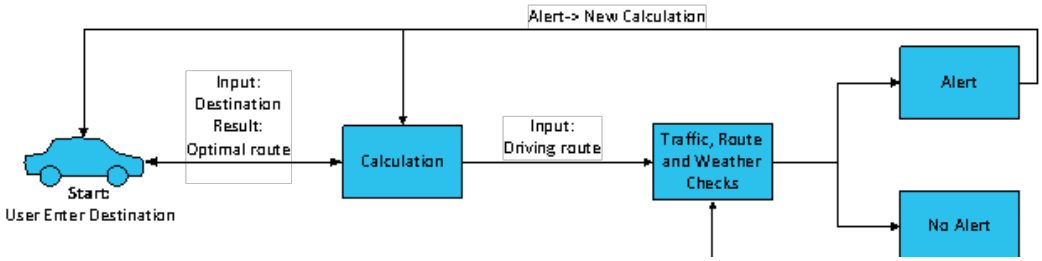

Figure 35: Alert system of the case study.

To demonstrate the alert system the following situation will be shown in the case study. The driver starts the application and enters a destination. The system returns the optimal route. Now there are three possible scenarios:

1. The weather situation changed and a new Route A is the better one.

2. Nothing happens and the driver arrives at the destination.

3. The traffic situation has changed and a new Route C is the better one.

These scenarios are depicted in the following figure 36.

At the end of year 1, the alert system is not yet implemented.

5.3 Architectural Modelling

The system model will initially consist of the following parts:

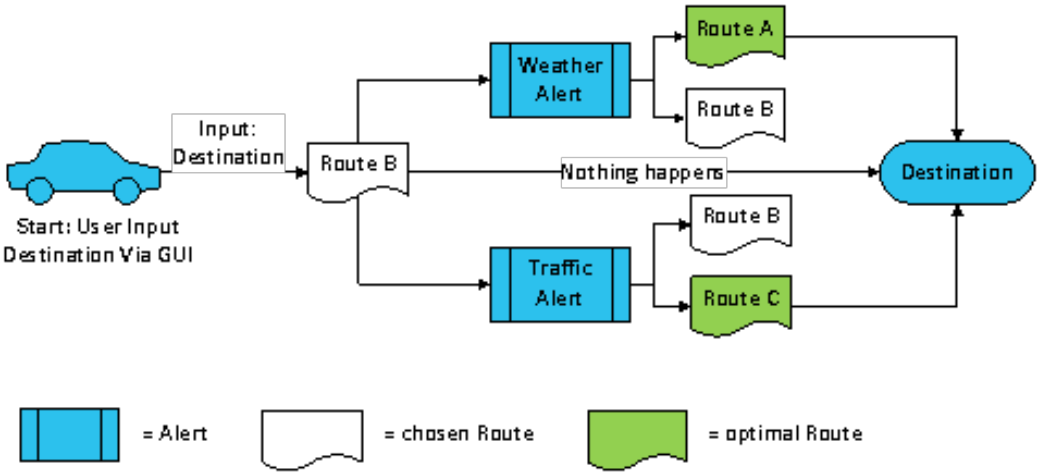

Figure 36: Scenarios for selection of different routes depending on the different alerts.

- Longitudinal dynamics of an electric vehicle, including the battery and air conditioning
- Route planning including traffic information
- Ambient conditions
- Driver profile

These models are coupled with the TWT co-simulation engine. At a later stage, other modules will be added to increase the accuracy of the range prediction. In the following, the different modules will be described.

5.4 Design Modelling

In this section, the single models are described in detail.

5.4.1 Longitudinal dynamics and drive train (CT)

This module calculates the energy consumption of the electric vehicle, depending on vehicle parameters and the route profile (velocity over time $v(t)$, and slope of the route). The module also includes the battery, so that the remaining battery charge (State-of-Charge, SoC) will be calculated for every route. The module also contains the air conditioning (A/C), being a main

power consumer. This module is a continuous time (CT) type and is implemented in Matlab. The Block Definition Diagram (BDD) of the longitudinal dynamics module is shown in the following figure 37.

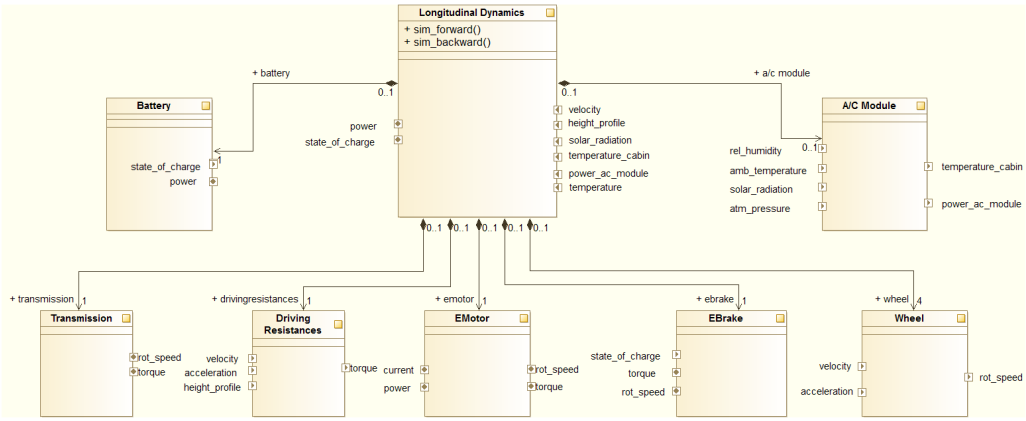

Figure 37: Block Definition Diagram of the longitudinal dynamics module.

The longitudinal dynamics of the vehicle is modeled with the total vehicle force $F_{vehicle}$ opposed by the aerodynamic drag F_{air}, the rolling resistance F_{roll} and the downhill force $F_{incline}$. The total force needed to overcome the resistance forces to reach a desired velocity profile $v_{vehicle}$ is transformed into torque T_{out} and speed ω_{out} demand.

The electrical motor is employed as a permanent magnet AC E-motor. In the E-motor component the mechanical power requirements are converted into an electrical performance requirement $P_{in,el}$ model by the engine map, which also includes the energy losses

The motor power requirements $P_{in,el}$ along with the power requirement coming from the auxiliary components are passed to the battery module.

The battery is modeled as a resistance capacitance model (RC) as described in [Joh02] with the equivalent circuit in Figure 38. The variation in the state of charge of the battery ΔSOC was estimated using the Coulomb counting method which takes the discharging current of a battery $I(t)$ and integrates it over time Δt as

$$\Delta SOC = \frac{I(t)}{Q_n}\Delta t \tag{1}$$

where the nominal capacity Q_n represents the maximum amount of charge that can be stored in the battery. In the equivalent circuit diagram in Figure 38, the capacitances C_b and C_c represent the large capacity of the battery (C_b)

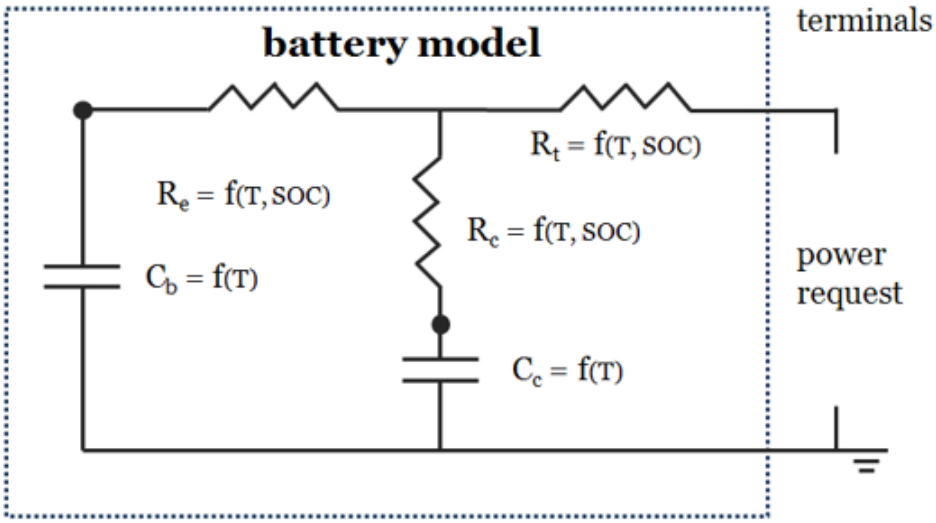

Figure 38: Battery model.

and the small capacity of the surface effects of the cell (C_c), respectively. The resistances R_c, R_e and R_t take the internal resistances into account.

If the power requirements on the power train can not be fulfilled by any of its components the simulation will switch from backward mode into forward mode as in Figure 39. In forward simulation mode, the change in state of charge ΔSOC is used as input from which a velocity profile is being calculated. While the forward simulation represents the physical causality (voltage at the electric motor drives the vehicle and therefore generates a velocity profile), the backward simulation can be calculated much faster than the forward simulation path, and is therefore the default mode.

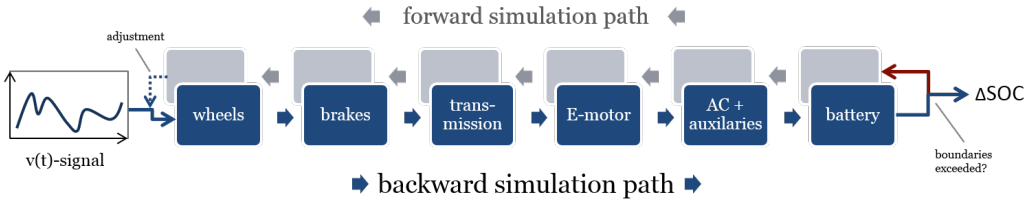

Figure 39: Simulation paths for backward and forward simulation.

In addition to the longitudinal dynamics and the battery, the vehicle's air conditioning (A/C) is also modeled in this module, since it can be a significant consumer of electrical energy.

5.4.2 Route planning (CT)

This module generates a velocity and altitude profile for a given start and end point of a route. The start and end destination are sent as a REST request to the Google Maps API and coupled to the co-simulation via Matlab, and can therefore be characterised as a CT model. In response, a list of alternative routes consisting of multiple road segments characterised by parameters like distance, estimated duration and GPS coordinates are sent. For each of the road segments a velocity profile is then generated. The velocity profile is estimated by ramps and constant functions under the constraint that the segment's distance is traveled in the given time. The slopes of the ramp functions are restrained by the characteristic acceleration curve of the electrical vehicle that will travel along the given route. Further, the altitude profile is calculated based on the GPS coordinates. The altitude for a specific region is interpolated from the data gathered during the space shuttle mission SRTM (Shuttle Radar Topography Mission) [6]. The velocity along with the altitude profile calculated for a certain route can be seen in Figure 40.

5.4.3 Ambient Conditions (CT)

To provide realistic conditions for the A/C control (see section 5.4.1), the most relevant environmental parameters (ambient temperature, air pressure, air humidity and cloudiness) are generated for a given route by the ambient conditions module. Latitude and longitude coordinates previously generated by the route planning (see section 5.4.2) are taken as input and forwarded to the OpenWeatherMap API, and the resulting data is processed with Matlab, and can therefore be considered a CT model. The weather information along the route is then estimated based on the data gathered from the weather station nearest to the coordinate points. The solar radiance Q is then calculated as a function of the solar zenith angle θ_s for given time and location and the air mass M_{air}.

The weather information that is generated by this module for a certain route is plotted in Figure 41. Since the weather information is provided by weather stations that are discretely spaced, the values for pressure, relative humidity and cloudiness are constant in this particular case are constant. If the route was going through an area in which two or more weather stations are placed, these values would vary.

[6] available at $http://glcf.umd.edu/data/$

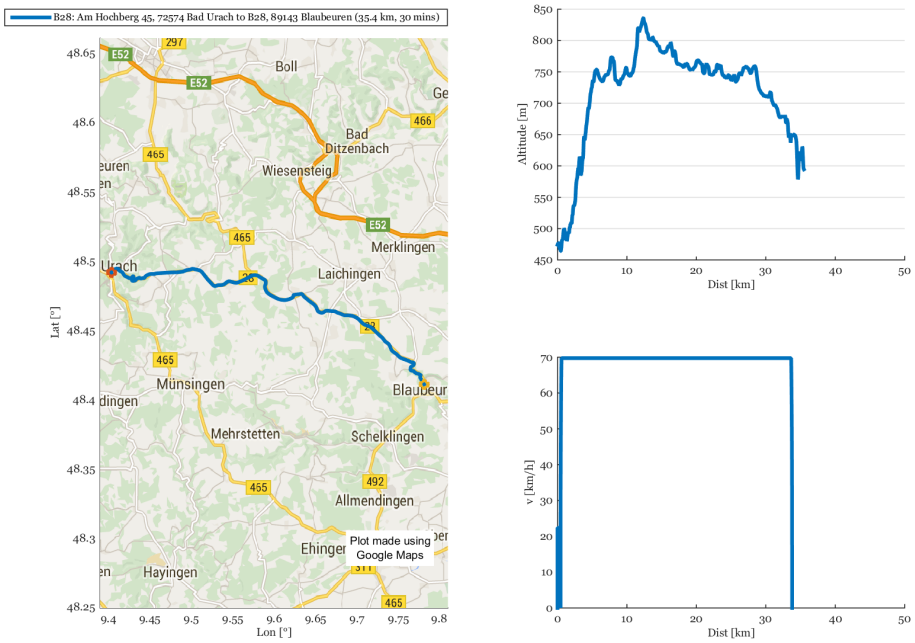

Figure 40: Velocity and altitude profile for a route of 35 km in the vicinity of Stuttgart. The route consists here of a country road only, and thus the velocity is stable at 70 km/h, while the altitude varies between 450m and 850 m above sea level.

5.4.4 Driver profile

The driver profile module takes as input the velocity representing the route profile and the actual velocity simulated with the longitudinal dynamics and drive train module. The driver profile module will try to follow the velocity profile and a new acceleration pedal position is generated according to the driver's profile dynamics. Therefore, this module can be considered a controller that aims at representing the behaviour of a real driver. Further, the acceleration pedal position is translated into power requirements. The driver profile is not yet implemented at the end of year 1.

5.4.5 CAN bus (DE)

The signals of different parts of the vehicle (such as sensors and actors) are transmitted via a bus, such as the CAN bus (CAN: Controller Area Network).

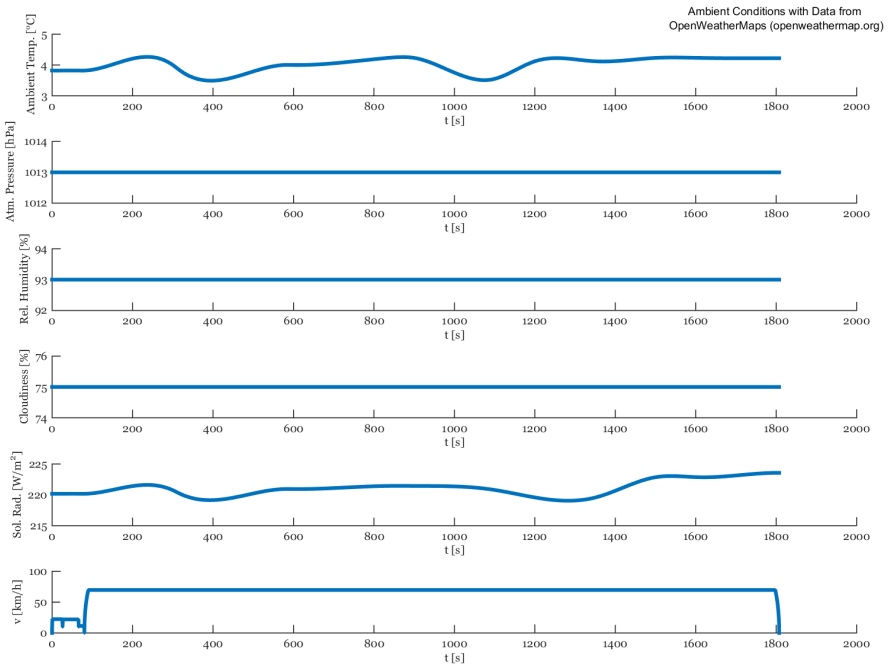

Figure 41: Ambient conditions for a given route plotted over time, including (from top to bottom): ambient temperature in degree centigrade, ambient pressure in hectopascal, relative humidity in percent, cloudliness in percent, solar radiation in watts per square meters, and velocity of the vehicle in kilometers per hour.

The CAN bus was developed in the 1980s by Bosch to reduce the amount of cabling in a vehicle, and therefore save weight and costs. While the CAN bus itself is clocked (i.e. all connected controllers are operating at a certain frequency), it is not a priori known when a signal is transmitted via the bus. For this reason, the CAN bus is modeled in this case study as a discrete event (DE) model using VDM and the Overture tool. The goal of this model is to represent the signal exchange in the online-SIMulation (see Figure 45).

The CAN bus model is described as a UML class diagram consists of three parts, the bus class, representing the state of the bus itself, the controller class, modelling the controller hardware, and the message class. The classes are shown in figure 42 with their most relevant instance variables and operations.

The bus is modeled as a state machine with two states, busy and idle. When

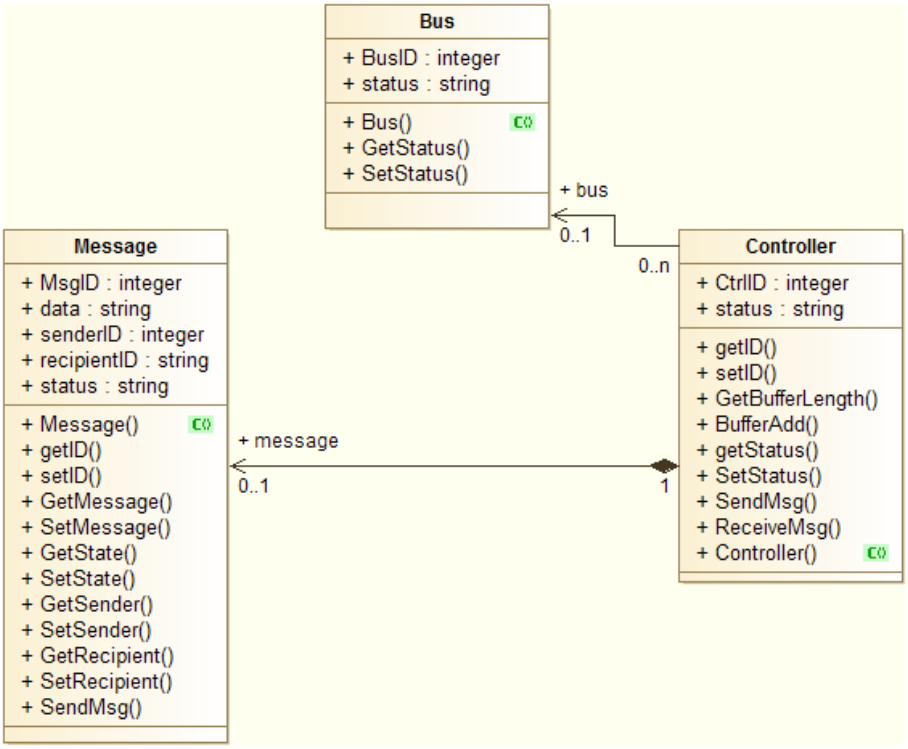

Figure 42: CAN bus model Class Diagram.

a message is transmitted via the bus, the state is busy, when there is no transmission, it is idle. The controller is also modeled with two states, busy and idle. In the future, a third state - the arbitration state - can also be implemented. The state machines are shown in Figure 43. The controller has a buffer, where the messages are stored when they are not transmitted, i.e. when the controller or the bus is busy. The messages are here represented as simple character strings.

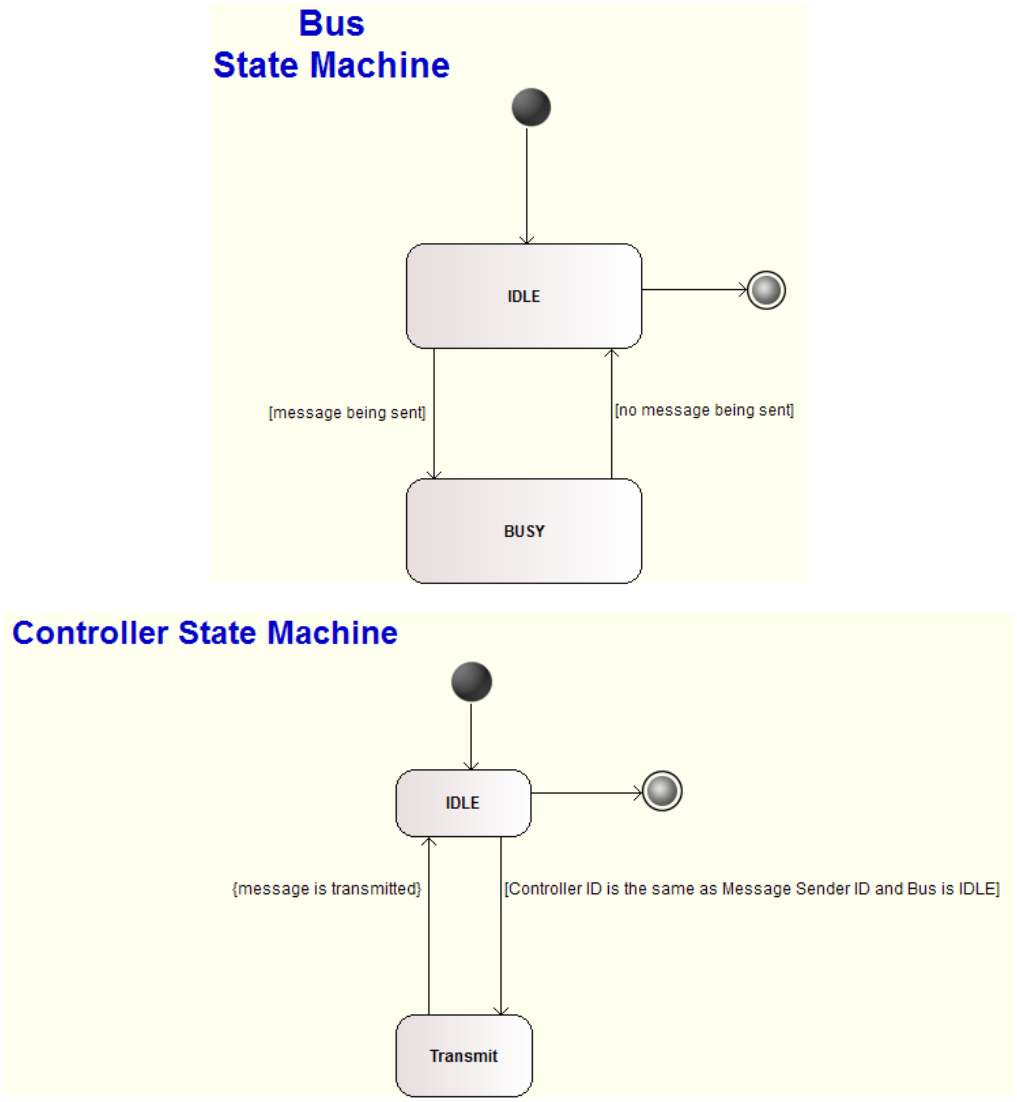

Figure 43: CAN bus state diagrams for the Bus class (top) and the Controller class (bottom)

5.4.6 Co-simulation

The interaction between the different modules described here is depicted in the following. As presented in Section 5.2.1 an off-line and on-line scenario was considered. The off-line scenario is shown in Figure 44. In this scenario, the simulation is performed once, before the actual trip. The route planning module sends the velocity and height profile (see for example Figure 40)

to the longitudinal dynamics module. The environment module sends the signals of the environmental conditions, such as solar radiation, temperature, air pressure, cloudliness and relative humidity (see for example Figure 41), to the longitudinal dynamics module. The longitudinal dynamics sends the required power to the battery module, which in turn sends the required change of charge back.

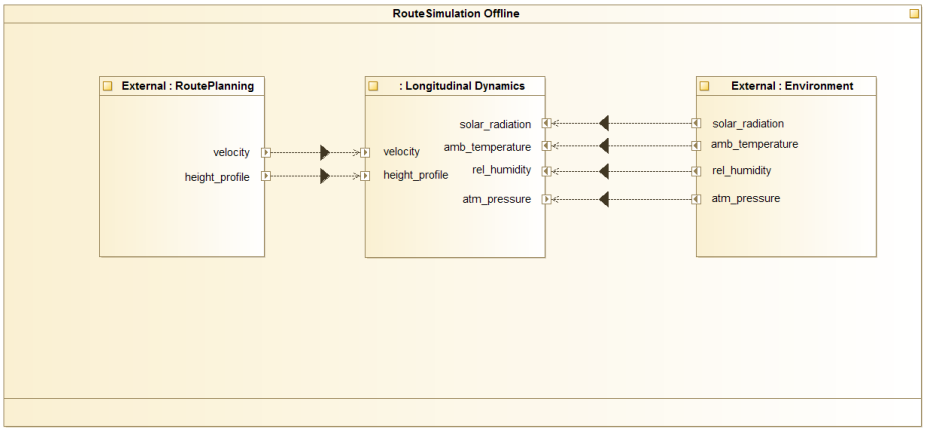

Figure 44: SysML Internal Block Diagram (IBD) for the route simulation offline scenario.

The exchange of real time data during the on-line scenario is seen in Figure 45. In addition, the CAN bus (described above) shall be included to model the signal exchange between the vehicle and the route planning module.

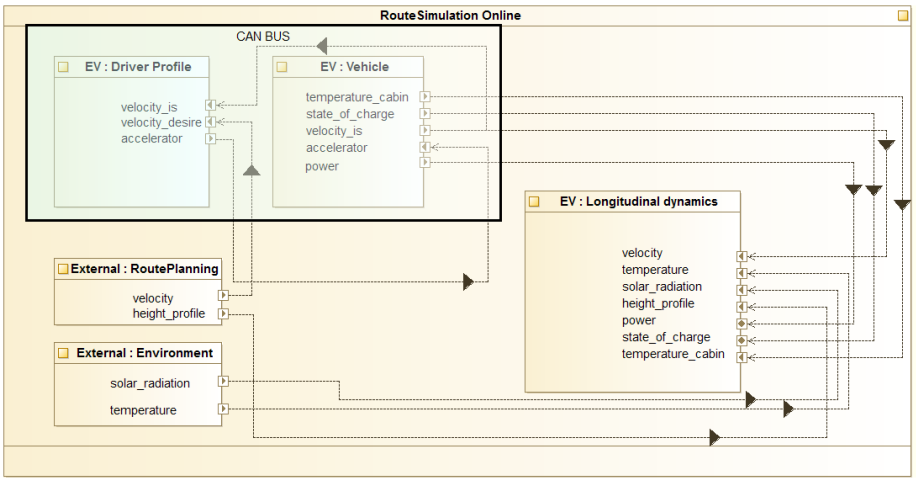

Figure 45: SysML Internal Block Diagram (IBD) for the route simulation online scenario.

5.4.7 Simulation results

For the route seen in Figure 40 the simulation results for the offline simulation are shown in Figure 46. The results are obtained from the TWT co-simulation framework. After a short distance in a town, the vehicle is driving on a country road for the remainder of the trip, at a constant speed of about 70 km/h. Due to the required energy, the battery State of Charge drops from 100 % to about 80 %, and thus the battery voltage drops from about 330 V to about 315 V. The figure also shows quite heavy oscillations in the battery voltage, current and the motor torque, which need to be evaluated in the future, during the assessment.

5.4.8 Evaluation of baseline tools

- Modelio

  Modelio offers a lot of tool support for system development that stretches from defining requirements to design and system modelling. Until now, we have made use of the *SysML* diagrams. The block definition diagram enabled us to define the interface of each module in a quick and elegant manner. The internal block diagram turned out to be very helpful when visualizing the information flow that has to be exchanged between the different models. The ability of linking requirements to systems blocks and code is of great interest to us. However, there is an

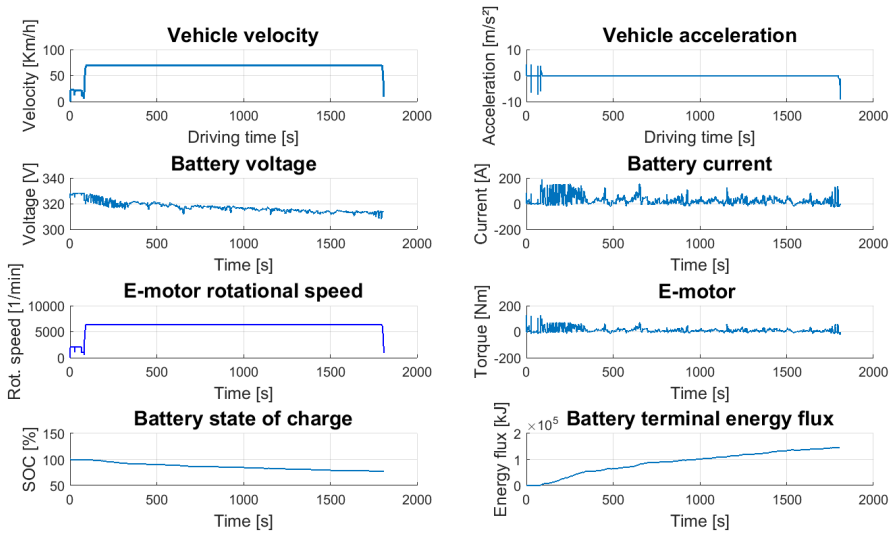

Figure 46: Simulation results for a given route, showing the most relevant signals.

overhead when porting existing documentation and modules to Modelio. Also, some of our models are developed with Matlab and there is no support of code linking in this direction.

- VDM / Overture

  Using the baseline tools (Overture to model the CAN bus, the TWT co-simulation engine to couple the CT models), it is currently not possible to couple the VDM model with the other models, described in Section 5.3. Therefore, at the end of year 1, the CAN bus model is not yet integrated into the co-simulation.

- OpenModelica

  While we only started with the evaluation of OpenModelica, it can already be said that it offers many possiblities for modelling, in particular for physical systems that include different domains (e.g. electrical, mechanical, thermal...). The fact that signals have units makes modelling and error-tracking easier, and it offers a intuitive way of modelling, due to its acausal nature. However, we found that this very acausal modelling makes execution of a simulation comparably slow. So far, we have tested to implement a simple version of the longitudinal dynamics model (see section 5.4.1) in OpenModelica.

- TWT co-simulation Framework

  We were able to couple continuous time models with only little modifications to the TWT co-simulation engine. Still, the TWT co-simulation engine lacks the support for simulating with a variable step time. Currently, it is not possible to couple Overture (for the DE models) to the TWT co-simulation engine, and there is no FMU export option. Future versions of Overture are planned to have an FMU export, and this will allow coupling of Overture models to the co-simulation engine.

5.5 Industrial needs and assessments

In order to give feedback for the tool developers (Workpackages 4 and 5), industrial needs were collected from the automotive case study, which are presented in this section. Some of these needs are already addressed in specific requirements to the tools (see Deliverable D7.3 [LPH+15]), others will be discussed with the partners in year 2 and 3. It is expected that this list of industrial needs will be updated, once the new INTO-CPS tools will be used.

5.5.1 TWT_1: Validity checking

- **Description:** The validity of system models, i.e. the correctness of the connected input and output signals to and from the different models is checked by the tools.
- **Method of Verification:** Manual variation of input and output signals to check if the tools detect the changes. If the tools notify the user of a mismatch, this requirement is fulfilled.
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0035 - 0036
- **Degree of achievement:** 0%

5.5.2 TWT_2: System structure

- **Description:** A system structure, describing the connections between the different models and their hierarchy, is created and saved by the

Table 4: Summary of the needs by the TWT case study to the INTO-CPS tools.

Req.	Category	Sub Cat.	Description	Priority	Status	Version	Review
TWT_1	Functional	SW	Validity checking	M	Not Started	1	to-do
TWT_2	Functional	SW	System structure creation	S	Not Started	1	to-do
TWT_3	Functional	SW	Results tracing	M	Not Started	1	to-do
TWT_4	Functional	SW	Requirements assessment	S	Not Started	1	to-do
TWT_5	Functional	SW	System creation from SysML	S	Not Started	1	to-do
TWT_6	Functional	SW	Version checking	S	Not Started	1	to-do
TWT_7	Functional	Method.	Version selection	S	Not Started	1	to-do
TWT_8	Functional	SW	Parameter variation	S	Not Started	1	to-do
TWT_9	Functional	SW	Parameter selection	S	Not Started	1	to-do

tools, in order to re-import and modify later. This simplifies the generation, distribution and modification of the whole system architecture.

- **Method of Verification:** Manual generation of a system model to import into the tools. If the system structure is correctly imported, this requirement is fulfilled.
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0001, 0005, 0012
- **Degree of achievement:** 0%

5.5.3 TWT_3: Results tracing

- **Description:** Simulation results are traced to the requirements and the models where the results originated from. This makes testing of the requirements easier and thus increases trust in the simulation results.
- **Method of Verification:** Comparison of results and requirements in simple models. For a measurement, the number of requirements that are linked to simulation results can be compared to the total number of requirements.
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0015 - 0017
- **Degree of achievement:** 0%

5.5.4 TWT_4: Requirements assessment

- **Description:** An indicator is provided to determine how well a requirement has been met, not just yes/no.
- **Method of Verification:** Requirements can be varied systematically to assess if the resulting indicator / ranking function reacts accordingly.
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0021
- **Degree of achievement:** 0%

5.5.5 TWT_5: System creation from SysML

- **Description:** Starting from a SysML model of the system, a skeleton system model (in FMI) is created. This will simplify the workflow by connecting the different tools for system modelling and simulation.
- **Method of Verification:** Comparison of an automatically generated skeleton system model with its SysML model.
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0049
- **Degree of achievement:** 0%

5.5.6 TWT_6: Version checking

- **Description:** The tools should be able to check if the interfaces between different models have changed due to different versions. This is relevant for the development process of modelling, especially when different parties are responsible for different models.
- **Method of Verification:** Selection of different model versions with different inputs and outputs. If different versions of the same model can be selected, or the user is notified of differences between model versions, this requirement is fulfilled.
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0090
- **Degree of achievement:** 0%

5.5.7 TWT_7: Version selection

- **Description:** The user selects different versions of the same model (either single model or system model).
- **Method of Verification:** Provided / not provided.
- **Assessment:** Not started.
- **Related baseline tools requirements:** It remains to be discussed if this requirement will be incorporated in the general requirements document, deliverable D7.3 [LPH+15].

- **Degree of achievement:** 0%

5.5.8 TWT_8: Parameter variation

- **Description:** Model parameters are systematically and automatically varied within defined limits. This allows systematic optimization of the models.
- **Method of Verification:** Provided / not provided.
- **Assessment:** Not started.
- **Related baseline tools requirements:** 0020
- **Degree of achievement:** 0%

5.5.9 TWT_9: Parameter selection

- **Description:** If the user wants to parametrize a whole model at once, parameter sets for models are selected from a parameter file. This can be helpful if relatively complex models have large parameter sets, describing for instance different variants of a product.
- **Method of Verification:** Provided / not provided.
- **Assessment:** Not started.
- **Related baseline tools requirements:** It remains to be discussed if this requirement will be incorporated in the general requirements document, deliverable D7.3 [LPH+15].
- **Degree of achievement:** 0%

5.6 Conclusion

In this section, the progress of the automotive case study that was made during year 1 is presented. Based on the know-how that was already available, TWT has created a co-simulation using CT models and have started with DE modeling, to create a cruising range prediction tool for electric vehicles. This tool takes into account a model of the physical behaviour of the vehicle (the longitudinal dynamics, the battery, electric engine and the air conditioning) as well as the topology of the chosen route, and weather

conditions (relevant for the air conditioning). It will therefore allow the user to predict the remaining battery charge, based on on realistic data (vehicle physics, route, weather). It can be used for example for realistic calculation of electric vehicle fleets or for integration in the vehicle's software, as an addition to the navigation system, or for optimizing vehicle settings for maximum range.

The evolving INTO-CPS tools will in the next years allow a further integration of the model-based design process into the engineering workflow, for example by connecting requirements with simulation results. The requirements to the case study will also be refined and extended, and will be assessed, after the modeling is complete.

6 General Conclusion

In this deliverable, the industrial members outlined their case study. They showed which INTO-CPS baseline tools they tried to use for modelling and simulating. Finally, they assess INTO-CPS baseline tools with respect to their industrial needs.

Concerning the Railway case study, ClearSy developped an interlocking algorithm using VDM-RT. Then using the Crescendo tool, Clearsy co-simulated the run of the interlocking algorithm together with the movements of the trains and the interlocking equipment from the wayside and the tracks (Telecomands, traffic lights, motorized switches, track circuits). The movements of the trains were simulated by 20-SIM, the interlocking algorithm was simulated by VDM-RT.

Concerning the Agriculture case study, Agro Intelligence has used the baseline technologies to co-model an agricultural robot taking into account the discrete control and continuous properties of the mechanical equipment and ground and soil properties. This has been co-simulated using the existing tools Crescendo and 20-sim. In the next years modelling will focus on increasing the system boundaries to incorporate safety properties in a more systematic way, explore the use of HiL and SiL, code generation and model-based testing in the product life-cycle as exemplified by Robotti.

UTRC building automation use-case has expressed continuous and discrete modelling, as well as well as co-simulation capability for a single room plant controlled by an FCU device. Our main objective for year 1 was to familiarize the baseline technologies, assess the model-based techniques from several tools and potentially assess the co-simulation platform requirements both from an industrial and a tool perspective. Results have shown a great benefit of the INTO-CPS technology, as it simplifies the workflow effort required for building automation design.

Concerning the Automotive case study, the progress that was made during year 1 is presented in this deliverable. Based on the know-how that was already available, TWT has created a co-simulation using CT models and have started with DE modelling, to create a cruising range prediction tool for electric vehicles. In the next years, the evolving INTO-CPS tools will be used for better integration of model-based design and testing.

According to industrial members assessments of the baseline tools, although the "starting" INTO-CPS baseline tools appear not fully "synchronized" for co-modelling and co-simulation, the future INTO-CPS tool chain should pave the way for a complete, scalable and mature co-modelling and development tool for hybrid system. Next year INTO-CPS releases appear very promising (e.g. FMI compatible co-engine and simulators, design space exploration) and the industrial members are ready to use such new INTO-CPS tool chain for improving their products and competitiveness.

References

[Bas15] Stelios Basagiannis. Case Study 1, Building, (Confidential). Technical report, INTO-CPS Confidential Deliverable, D1.1d, December 2015.

[EGH15] Jose Esparza, Ole Green, and Stefan Hallerstede. Case Study 1, Agriculture, (Confidential). Technical report, INTO-CPS Confidential Deliverable, D1.1c, December 2015.

[HL15] Francois Hantry and Thierry Lecomte. Case Study 1, Railway, (Confidential). Technical report, INTO-CPS Confidential Deliverable, D1.1b, December 2015.

[IJL14] José Antonio Esparza Isasa, Peter W.V. Jørgensen, and Peter Gorm Larsen. Hardware In the Loop for VDM-Real Time Modelling of Embedded Systems. In *MODELSWARD 2014, Second International Conference on Model-Driven Engineering and Software Development*, January 2014.

[Joh02] V.H. Johnson. Battery performance models in advisor. *Journal of Power Sources*, 110:321–329, 2002.

[KB15] Christian König and Natalie Balcu. Case Study 1, Automotive, (Confidential). Technical report, INTO-CPS Confidential Deliverable, D1.1e, December 2015.

[LPH+15] Peter Gorm Larsen, Ken Pierce, Francois Hantry, Joey W. Coleman, Sune Wolff, Kenneth Lausdahl, Marcel Groothuis, Adrian Pop, Miran Hasanagić, Jörg Brauer, Etienne Brosse, Carl Gamble, Simon Foster, and Jim Woodcock. Requirements Report year 1. Technical report, INTO-CPS Deliverable, D7.3, December 2015.

[Sys12] OMG Systems Modeling Language (OMG SysML™). Technical Report Version 1.3, SysML Modelling team, June 2012. http://www.omg.org/spec/SysML/1.3/.

[Wes00] Tim Wescott. PID Without a PhD. *Embedded Systems Design*, (86 - 108), October 2000.