Automated Functional Safety Analysis of Vehicles Based on Qualitative Behavior Models and Spatial Representations

Peter Struss, Sonila Dobi
Tech. Univ. of Munich
{struss, dobi}@in.tum.de

Abstract

For automotive vehicles, the analysis of functional safety has been strongly enforced in recent years and is subject to international standards. This paper presents an application of qualitative model-based and spatial reasoning with the aim of automating a major part of the process. The problem and model is split into two parts: first, a qualitative model of the vehicle subsystem (the drive train of a truck in our case study) is used to predict the effect of a component fault on the behavior of the entire vehicle, such as an unintended acceleration. Secondly, the impact of this effect on the environment of the vehicle has to be determined, e.g. a collision of the vehicle with persons or objects. This requires a model of the environment and the interaction of the vehicle with it and, hence, a spatial representation of positions of the vehicle and other objects relative to the road and their interference under different scenarios.

1 Introduction

Analyzing whether a technical system performs safely even under the occurrence of a fault is of high importance when its failure may cause injury or death of humans or other severe damage to its environment. For automotive vehicles, safety analysis has been strongly enforced in recent years and is subject to international standards. This task, which may have to be carried out repetitively for different versions and variants during the design of a system, is knowledge-intensive and consumes significant efforts of experts. Currently, there are no tools supporting and automating the reasoning part of this task. Achieving this through knowledge-based systems solutions that reduce the labor cost and improve the guaranteed coverage and quality of the results is of high importance under economic, social, and environmental aspects.

We present the realization and evaluation of a prototypical solution to the problem in a case study on the drive train of a truck ([Dobi et al., 2013]). It was carried out together with an industrial partner, who provided the subject, the requirements, and the evaluation criteria (in terms of a manually generated safety analysis).

In our solution, we exploit previous research results on qualitative deviation models ([Struss, 2004]) and automated model-based FMEA ([Price, 2000], [Picardi et al., 2004]). Beyond this, it provides several **scientific contributions** and **novel solutions**:

- A **conceptualization** of and a **systematic approach** to the task of **functional safety** analysis of cyber-physical systems, which does not exist in the literature, so far.
- An application of **qualitative deviation models** to a class of **mechanical systems** combining torques and forces and their control unit software.
- The development of a **spatial representation** of the motion of road vehicles and its environment as the basis for the automated analysis of the impact of a component fault on safety.

The following section describes the application context of the task and the drive train case study. Section 3 discusses the approach to functional safety analysis and its formalization. Modeling the drive train and inferring **hazards**, i.e. abnormal **behavior** of the **vehicle** caused by component **faults** are presented in section 4, while the following section describes how to determine the **impact** of an **abnormal motion** of the vehicle on its **environment**.

2 The Task

2.1 Safety Analysis in the Automotive Industries

The number of accidents, casualties, and injuries caused by automotive vehicles, but also other kinds of impact on the environment, e.g. through pollution, is a big concern and has led to many technical solutions (from anti-lock braking systems to sophisticated driver assistance systems), legal regulations (e.g. OBD2), practices and processes (Failure-modes-and-effects and criticality analysis, FMECA, Fault-tree analysis, FTA), and standards (e.g. IEC 61508).

Through a recent standard, ISO 26262 on Road Vehicle Functional Safety focusing on E/E (electrical and electronic) systems [ISO-26262, 2011], the necessity to carry out thorough and vast analyses of vehicle safety and steps towards preventing unacceptable risks caused by system design or component failure has been greatly emphasized.

In the analysis phase, the causal relationships between faults occurring in the system and hazards, i.e. unintended behavior bearing the risk of damage, has to be determined, as well as scenarios under which this damage may occur, its severity, and whether it can be controlled by the driver. If unacceptable risks are not excluded, effective policies have to be introduced into the design (e.g. in terms of structural

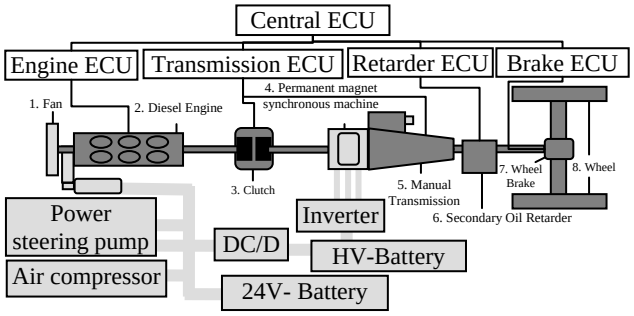

Figure 1. Drive Train Model

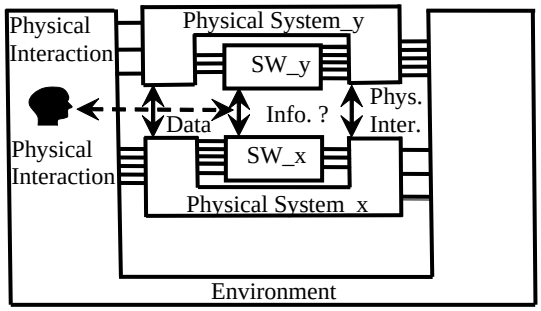

Figure 2. Cyber-physical Systems

changes and redundancy, additional sensors, or modified software functions).

2.2 The Drive Train Case Study

Our industrial partner selected a drive train of a truck as the subject of a case study. Its structure is sketched in Figure 1. The main part (in dark gray) comprises the engine, which produces torque for acceleration, but also for braking, the clutch, which may interrupt the propagation of torque, the transmission allowing to switch between forward and reverse torque (and idling), the retarder, a braking device that, when applied, counteracts the rotational motion through a propeller moving in oil, and the axle with the wheel, which transforms rotational acceleration into translational acceleration (and vice versa), and the wheel brakes. Components are controlled by specialized Electronic Control Units (ECU), which communicate with a central ECU that processes, for instance. the driver demands. The light-gray components are related to electrical aspects and are not discussed in this paper.

The industrial partner also supplied us with documents on exemplary problems and manually generated safety analysis tables. The core of an entry in such a table links a component fault (e.g. "erroneous CLOSE command to the clutch"), a special driving situation ("engine running, vehicle standing"), and a type of scenario ("vehicle in front of pedestrian crossing") with a hazard ("unintended forward acceleration") and its impact on the environment ("injury of persons"). Relevant impacts are typically hitting objects or persons, where, obviously, the severity is influenced by the type of object. More details are provided in [Dobi et al., 2013].

3 Safety of Cyber-Physical Systems

As mentioned before, there is no lack of standards and current practices. However, they do not provide a formal foundation for a computer-based solution. Hence, a systematic and structured approach to functional safety analysis of systems with embedded software had to be developed and mapped to formalized solutions in model-based problems solving. We present the solution using our case study as an illustration. Its background is illustrated by Figure 2: a cyber-physical system (CPS) comprises a number of subsystems, which are systems composed of physical (mechanical, electrical, hydraulic, etc.) components and software components, whose interaction happens exclusively through a usually relatively small set of sensor signals as the input to the software components and actuator signals as their output. Different subsystems interact both via connections between their physical components and via communication between their software components. In a vehicle, the components of the drive train with their individual ECUs are examples for such subsystems. At a higher level, the drive train itself can be considered as a subsystem. The top level system is the entire vehicle.

From the perspective of safety analysis, it is important to note that it is **only** the vehicle as a **physical** system that interacts with the environment. The embedded software never directly interferes with the environment. As a consequence, **hazards**, misbehaviors that bear the potential of damage in the environment, are defined **exclusively** at the intersection of the **physical system** and the **physical** environment, in our example, an unintended motion of the vehicle relative to its environment. As an important consequence, buggy software behavior matters if and only if it may cause the physical system to create a hazard.

In turn, hazards create risks only through their impact on the environment. Obviously, this environment is much more diversified and dynamically changing compared to the designed artifact, the vehicle. It cannot be explored exhaustively, but only through certain abstract types of scenarios and driving situations as illustrated by the example mentioned in 2.2.

In consequence, we approach the task of building a tool for safety analysis by dividing it (conceptually) into two steps (Figure 3):

- **hazard analysis**: a **behavior model** of the CPS (i.e. the relevant subsystems of the vehicle) is used to determine whether assumed faults of (software or physical) components may result in (pre-defined) hazards for a set of specified scenarios, in our case **driving situations** (in terms of speed, driver actions, etc.) and **road conditions** (slope and surface friction),
- **impact analysis**: a **model of the environment**, relating positions and motions of the vehicle and other objects and agents, determines whether the fault/hazard may have a dangerous **impact** (in our case, a collision) under certain **environmental conditions**, in the example specified by the driving

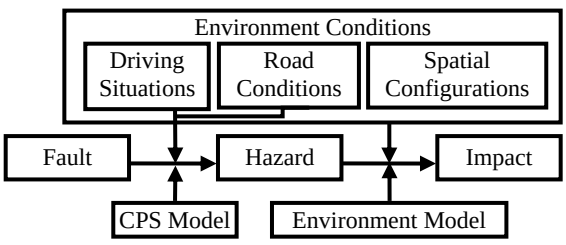

Figure 3. Hazard and Impact Analysis and their Inputs

situation, road conditions (including curvature) and the spatial configuration of other objects.

The two models together associate component faults directly with safety violations and risks.

For characterizing and implementing the required inferences, hazard analysis can be formalized as the task of determining whether a hazard HAZ_k may occur under a model of the system with an assumed fault, $MODEL_f$, in a scenario $SCEN_j$:

$$MODEL_f \cup SCEN_j \cup HAZ_k \not\models \bot,$$

or, stronger, is entailed by them:

$$MODEL_f \cup SCEN_j \models HAZ_k \ .$$

Hazard analysis is basically identical to failure-modes-and-effects analysis (FMEA) (where hazards correspond to effects) and iterates over the Cartesian product of scenarios and faults and performs the above checks for each defined hazard. Under qualitative abstraction, each of the three elements can be represented as a set of constraints or, if preferred, as first order formulas, and the analysis can be carried out using a constraint solver. In our case study, we used the FMEA engine of Raz'r [Raz'r, 2ß13]. Under a different view, a hazard is associated with a fault, if $MODEL_f$ is a consistency-based diagnosis of the (hypothetical) observations $SCEN_j \cup HAZ_k$.

Impact analysis is formalized as checking whether $MODEL_f$, the environment model, $MODEL_{env}$, and a scenario, $SCEN_{env,j}$, are consistent with an impact, $IMPACT_l$:

$$MODEL_f \cup MODEL_{env} \cup SCEN_{env,j} \cup IMPACT_l \not\models \bot,$$

where $SCEN_{env,j}$ is usually an extension of $SCEN_j$ by environmental conditions.

If hazards are determined as an intermediate result, the criterion becomes

$$HAZ_k \cup MODEL_{env} \cup SCEN_{env,j} \cup IMPACT_l \not\models \bot.$$

In either case, it is apparent that impact analysis can be implemented using the FMEA engine, as well.

In order to understand that the consistency criterion introduced above is not too weak, one has to consider that we formalize and implement an inherently qualitative and worst-case analysis. Firstly, the analysis is performed at design time, and parameters may not yet have numerical values. Beyond this, the faults are qualitative: decreased friction of a brake, a leakage of a pipe, and a high sensor signal cannot be described by numerical values. Hazards are qualitative: too high or too low acceleration are not specified more precisely than in this qualitative way. Scenarios are qualitative: "a vehicle approaching a pedestrian crossing with medium speed" or "going downhill a winding road". With regard to the required inferences, the worst-case analysis is not expected (and, given the qualitative input, not able) to firmly conclude the impact. What needs to be determined is the **potential** of a collision, e.g. given a reduced deceleration of the vehicle and, hence, a longer brake path – after all, for the instances of the pedestrian crossing scenario, it is not even clear whether there will be any pedestrians present. Finally, determining that a brake with reduced friction causes a reduced deceleration suffices to consider it as a reason for a risk in the respective scenario.

4 Model-based Hazard Analysis

In the following subsections, the elements needed for hazard analysis are described for the drive train case study:

- a **model** of the respective **system** (physical and software components), in which models of the component fault can be injected,
- a definition of relevant **driving situations and road conditions**, for which the analysis has to be carried out,
- a definition of the relevant **hazards**,

and we discuss the results obtained.

4.1 Drive Train Model

Reflecting the basic requirements of the analysis task, the models we developed are

- **behavior models** (as opposed to functional models as proposed e.g. in [Kurtoglu-Tumer-Jensen, 2010]), because the analysis needs to be based on determining the system performance objectively (based on 1st principles models), and completely, (not reduced by the expected function of intentions of the designers). Those aspects are captured by the scenarios and hazards considered,
- **qualitative models**, according to the nature of the analysis, as discussed in the previous section,
- **deviation models**, since faults, faulty behaviors, and hazards express (significant) deviations from a reference (nominal or intended state),
- **component-oriented, compositional models**, firstly because the analysis is highly repetitive, in having to be performed several times during the design phase, applied to alternative designs, and subsequently to different versions and variants, and a model-based solution is economically beneficial only if building the model is cheap. Secondly, components are the building blocks of the system and also the entities introducing the faults, and the same holds for their models. Note that the qualitative level of modeling increases the re-usability of models, because many intricate, irrelevant distinctions are abstracted away.

The components of the drive train determine the **acceleration or deceleration** of the vehicle.

These considerations indicate that the modeling task is non-trivial. The issues to be addressed are

- The overall (deviation of the) torque applied cannot be determined locally, but only as the **combined impact** of several components.
- The transformation of **torque** into an **accelerating force** and vice versa
- The modeling of software components and, especially, **software faults**, which seems to be in the complexity class of clearing out the Augean stables.

We discuss these aspects in the following.

Deviation Models

We use deviation models ([Struss, 2004]) in the same way as in a case study of FMEA of a braking system [Struss-Fraracci, 2012]: the qualitative deviation of a variable x is defined as

$$\Delta x := \text{sign}\,(x_{act} - x_{nom})$$

which captures whether an actual (observed, assumed, or inferred) value is greater, less or equal to the nominal value. The latter is the value to be expected under nominal behavior, technically: the value implied by the model in which all components are in OK mode.

Faults may introduce non-zero deviations, e.g. the model of a worn brake would result in a deviating braking torque, which depends on the direction of the rotation (static friction)

$\Delta T_{brake} = \omega$

or the applied torque in case of kinetic friction

$\Delta T_{brake} = T_{wheel}$

Models of OK and faulty behavior are stated in terms of constraints on the deviations. For instance, a correctly closed clutch simply propagates a deviating torque coming on the left from the engine to the right (flipping the sign):

$\Delta T_{right} = -\Delta T_{left}$.

Most models variables and all deviations have values from the domain Sign = {-, 0, +}: torques and forces, T and F, rotational and translational speeds, ω and v. The commands and states explicitly discussed here have Boolean values {0, 1}.

In the following, we outline the key ideas and illustrate them by selected component models. The models were manually created. How to automate their generation from existing numerical models is important, but not subject to this paper (see e.g. our previous work in [Struss-Fraracci-Nyga, 2011]).

Drive Train Modeling: Combining Torques

The core purpose of the drive train component models is to determine the **combined** (deviations of the) **torques** of the various components acting on the axle, the transformation of **forces and torques** into each other through the interaction of the wheel with the road surface dependent on friction, and the relation to translational **acceleration** and speed of the entire vehicle. Things get even more complicated, when the road has a non-zero slope and **gravity** adds a force that accelerates (or decelerates) the vehicle – again, dependent on friction: with sufficient friction, the gravity component along the road will add another torque to the axle (which may be overcome by other torques), otherwise, it will directly contribute to the translational acceleration of the vehicle (sliding downhill). In this paper, we ignore the impact of slope and friction, in assuming that friction suffices to strictly link torque and force and that the road is level.

The overall torque results from the interaction of all components, which potentially contribute to it. The engine can produce a driving torque, the braking elements (wheel brake, retarder, and engine) may generate a torque opposite to the rotation, and the clutch and transmission may interrupt or reverse the propagated torque.

Our current model is based on assuming that there are no cyclic structures among the mechanically connected components, which is the case in our application, but certainly also in a much broader class of systems. The component models link the torque (deviation) on the right-hand side to the one on the left-hand side, possibly adding a torque (deviation) generated by the respective component. Hence, at each location in the drive-train model, the torque (deviations) represent the sum of all torques collected on its left-hand side.

Whenever a terminal component (in our case the wheel) or a component in an open state (the clutch and the transmission) is reached, the arriving torque is the total one for the section left, and for the open components, the torque on the right-hand side is zero, as exemplified by the clutch (state=0 means open):

$state=1 \Rightarrow T_{right} = T_{left}$

$state=0 \Rightarrow T_{total} = T_{left} \wedge T_{right} = 0$.

Determining the deviation models is not as straightforward, as it may appear, as we will explain using the model of the retarder as an example. If engaged (state=1), it will generate a torque opposite to the rotation (zero, if there is no rotation) and add it to the left-hand one. The base model is obvious:

$T_{right} = T_{left} \oplus T_{brake}$

$state = 1 \Rightarrow T_{brake} = -\omega$

$state = 0 \Rightarrow T_{brake} = 0$,

where $\oplus$ denotes addition of signs. The first line directly translates into a constraint on the deviations:

$\Delta T_{right} = \Delta T_{left} \oplus \Delta T_{brake}$

However, determining ΔT_{brake} requires consideration of how the actual state is related to the nominal one, which depends on the control command to the component, and, to complicate matters, not on the actual command, but the **command that corresponds to the nominal situation**. This means we have to model possibly deviating commands, and we apply the concept and even the definition of a deviation also to Boolean variables. For instance, in the retarder model, $\Delta state$ = - means $state = 0$ (i.e. it is not engaged) although it should be 1, and $\Delta state$ = + expresses that it is erroneously engaged. Such deviations could be caused by retarder faults, e.g. stuck-engaged. However, in the context of our analysis, we must consider the possibility that the commands to the retarder are not the nominal ones (caused by a software fault or the response of the correct software to a deviating sensor value). Under multiple faults, a component fault may even mask the effect of a wrong command (the retarder stuck engaged compensates for Δcmd = -). In the OK model of the retarder, the actual state corresponds to the command, and the deviations of the command and state (i.e. the real, physical state) are identical:

$\Delta state = \Delta cmd$.

For a stuck engaged fault, however, Table 1 captures the constraint on the deviations:

Table 1. Retarder stuck engaged - Deviation constraint

cmd	Δcmd	Δstate
1	0	0
0	0	+
0	-	0
1	+	+

Here, the third row represents the masking case mentioned above, the first one reflects that the physical state coincides with the command, while in the second one, it does not.

From $\Delta state$, ΔT_{brake} is determined by

$\Delta T_{brake} = -\omega \otimes \Delta state$,

where $\otimes$ denotes multiplication of signs. This completes the model of the retarder.

Software Models

Since the drive train contains a number of ECUs, we also need to include models of software **and its faults** in our library. Remember: all that matters about software faults is their impact on the physical system, more precisely, on the controlled actuators. For the Boolean commands in our model, this means the only fault types to be considered are

- **Missing** (or late) **command**: $\Delta cmd = -$
- **Untimely** (or early) **command**: $\Delta cmd = +$.

The same applies to continuous actuator signals, where the faults represent signal too low and too high, respectively.

This provides evidence for the claim that putting safety analysis back on its feet and the physical model in the center, greatly simplifies the modeling and analysis of the embedded software. In particular, for the purpose of hazard analysis, we obtain a small set of reusable software models for our library (see [Struss, 2013]). Of course, if we do have a more detailed model of the software, also the fault models can be more specific.

4.2 Driving Situations and Hazards

While the model captures the objective physical behavior of the system, the **functional** perspective is introduced by the selection and definition of the relevant and meaningful scenarios (driving situations) and hazard (violations of the desired functionality) for which the fault analysis is carried out. Whether a component fault causes a hazard is usually dependent of the context: if the retarder is stuck and, hence, applies a braking torque does not lead to a risk if the driver pushes the brake pedal, anyway. Therefore, hazard analysis (and FMEA) is carried out for certain different **driving situations**. From the material in our case study, we observed that there are relatively few of them. They can be characterized by the **vehicle velocity** and the **driver demand**, which is expressed by pushing the accelerator pedal or the brake pedal and selecting the gear.

The considered scenarios are normal **driving** (with or without intended acceleration), **starting**, and **braking** for both forward and backward motion. The forward ones are defined according to Table 2. Based on the documentation of the manual analysis and interviews, we introduced a distinction between "low speed" ("+") and "high speed" ("++"), which are both mapped to "+" in the physical model.

Table 2. Selected Definitions of Driving Situations

	Accelerator pedal pushed	Brake pedal pushed	Gear			Clutch pedal not pushed	v
			F	N	R		
F-start	x		x			x	0
Drive, high speed	x		x			x	++
Drive, low speed	x		x			x	+
F-brake, high speed		x	x			(x)	++
F-brake, low speed		x	x			(x)	+

Also the hazards are predefined. For the drive train, the hazards are given by deviating acceleration of the vehicle (resulting from deviating torques). Hence, the **basic hazards** are Δa=+ and Δa=-. From the perspective of FMEA, they may have a different intuitive meaning for different scenarios. For instance, Δa = + means for the (forward) drive situation that the vehicle becomes faster than intended, while for the braking scenario, the braking torque is reduced or even zero. For supporting an intuitive interpretation of the hazards, we defined them in a scenario-specific way, as illustrated by Table 3, which shows only the definitions relevant for forward situations (several values in a cell represent a disjunction).

Table 3. Selected Hazard Definitions

Hazard	Driving Situation	a	Δa
Increased acceleration	Drive, F-start	+	+
Reduced or no acceleration	Drive, F-Start	+, 0	-
Unintended deceleration	Drive	-	-
Unintended backward acceleration	F-start	-	-
Reduced or no deceleration	F-brake	-, 0	+
Increased deceleration	F-brake	-	-
Unintended acceleration	F-brake	+	+

4.4 Results of Automated Hazard Analysis

Raz'r performs the checks described in section 3 for all faults, scenarios, and hazards, and summarizes them in a table, which represents a key result of safety analysis. Table 4 shows a part of it, including software faults. With respect to the modeled component faults and the defined driving situations and hazards, the table is complete and correct. None of the entries in the table is surprising or difficult to obtain manually – but it is not the objective of this work to generate results the engineers could not produce. Instead, the goal is to automate the mechanistic part of their work. The manual production of the table costs at least tens of person hours, while the tool needs minutes. And the algorithm does neither omit scenarios or faults nor miss a hazard.

Table 4. Partial Results of Automatic Hazard Analysis for the Driving Situation "Drive"

Scenario	Part	Failure Mode	Hazard / Impact
DriveSituation	CrankShaft1	Broken	:Reduced_or_no_acceleration
DriveSituation	Clutch1	ClutchStuckOpened	:Reduced_or_no_acceleration
DriveSituation	Clutch1	ClutchStuckClosed	:>>no system level effects<<
DriveSituation	GearBox1	StuckReverse	:Unintended_deceleration
DriveSituation	GearBox1	StuckNeutral	:Reduced_or_no_acceleration
DriveSituation	GearBox1	StuckForward	:>>no system level effects<<
DriveSituation	Retarder1	RetarderStuckNotEngaged	:>>no system level effects<<
DriveSituation	Retarder1	RetarderStuckEngaged	
DriveSituation	Retarder1	RetarderStuckEngaged	:Reduced_or_no_acceleration
DriveSituation	Retarder1	RetarderStuckEngaged	:Unintended_deceleration
DriveSituation	Brakes1	StuckNotEngaged	:>>no system level effects<<
DriveSituation	BrakesECU1	UntimelyCommand	:Unintended_deceleration
DriveSituation	RetarderECU1	MissingCommand	:>>no system level effects<<
DriveSituation	RetarderECU1	UntimelyCommand	
DriveSituation	RetarderECU1	UntimelyCommand	:Reduced_or_no_acceleration
DriveSituation	RetarderECU1	UntimelyCommand	:Unintended_deceleration

5 Model-based Impact Analysis

The hazard analysis described above yields the consequence of faults in terms of the behavior of the CPS, in our case deviations in the motion of the vehicle, more specifically,

deviation of its acceleration. Determining the **impact** requires a model that captures the interaction of the CPS with its environment, which means in our case study, it has to represent the location and motion of the vehicle as well as other objects and to infer potential collisions. Again, this analysis is carried out for different scenarios, where scenarios in this phase need to capture different **spatial configurations** of the vehicle and other objects. Besides their (potential) spatial extension, objects have an associated type (which influences the severity of the impact). The various spatial configurations represent classes of specific real situations, such as "street with persons on sidewalk" and "approaching exit on a freeway". As a consequence, the required spatial representation has to be very abstract and qualitative, as described in the following section.

5.1 Environment Model

As opposed to extensive other work (e.g. in robot navigation) that exploits spatial reasoning for exploring trajectories of moving objects and their spatial relations and predicting collisions based on **specific** situations we need to represent archetypes of situations, possible ranges of motions, and the potential of collisions. [Dylla et al., 2007] deals with a similar task in order to represent sea navigation rules. While this domain implies degrees of freedom in 2D, our solution exploits the (trajectory on the) road as a reference, which greatly simplifies the spatial representation and reasoning.

To approach this and derive a simplified representation, we first abstract from the road as a 3D object:

- Although it may go uphill and downhill, the 3rd dimension is eliminated and only expressed as an attribute **slope** of the road, which influences the motion of the vehicle through gravitational force, which is already covered by the vehicle model.
- Although the road (or, more generally, the intended trajectory of the vehicle, as in "exiting from a freeway") may have **curves**, which influence the impact (e.g. at high speeds), we also turn this into a (Boolean) attribute of the road, indicating whether the **curvature** is significant or not
- Then we apply a **transformation** $\Re^2 \rightarrow \Re^2$ by turning the vehicle trajectory into one coordinate axis, σ, and the orthogonal distance from the road into the other coordinate, δ, with the initial location of the vehicle in the origin, as illustrated by Figure 4. Unless the trajectory is a straight line, this mapping is well-defined only for a certain envelope of the trajectory which depends on its curvature. Outside this envelope, an object on the concave side of the trajectory covers an extended area in the (σ,δ)-space, which is a feature, rather than a deficiency: in reality, it is adjacent to the trajectory for a longer section and could be hit by the vehicle when it deviates from the trajectory at several positions.

Finally, we abstract this space according to the distinctions that appeared in the natural language descriptions supplied by the industrial partner, i.e. we discretize $\Re^2$ to a level that captures the qualitative distinctions needed to characterize locations and is able to infer a potential collision due to the (qualitatively) deviating motion of the vehicle. We chose the grid depicted in **Fehler! Verweisquelle konnte nicht gefunden werden.**. The grid is defined by qualitative positions 0 (at the vehicle), **c**lose, **m**edium, **f**ar for σ, i.e. along the vehicle trajectory, and **s**traight, **r**ight-of, **m**edium-**r**ight-of, **f**ar-**r**ight-of (and the same for left) for δ, the distance from the trajectory. The vehicle's initial position will always be in (0, s), while, for instance, pedestrians may cover the r-strip, or a median be located in the l-strip.

The environment model is specific to the analysis task. In our case study, it is a "component" that is connected to the road (retrieving its curvature attribute), the vehicle (to access its speed and acceleration (deviation)), and other objects, which have a location and type. Right now, we include obstacles (immobile objects), other vehicles (with the potential to move fast, and persons (moving slowly).

The spatial interaction is modeled by **impact range constraints** that determine the potential positions of the vehicle and the other objects after the initial situation. For instance, a slowly driving vehicle with Δa=+ may reach positions (s, c) and (s, m). A fast, braking vehicle with Δa=+ on a road with non-zero curvature covers {(s, c), (s, m), (s, f), (l, m), (l, f), (ml, f), (r, m), (r, f), (mr, f)}, i.e. it may deviated from the trajectory.

5.2 Spatial Configurations and Impacts

While the environmental model represents the physics (of motion), the focus of the analysis and the relevant aspects of safety are expressed by the scenarios and effects to be considered. The former are given by combining the driving situations and road conditions (including curvature) used in hazard analysis with spatial configurations of other objects (relative to the vehicle position), which correspond to different classes of traffic situations. For instance, "pedestrian crossing medium" is represented by

(vehicle.speed=+,
object.type="persons",
object.location={(m, s), (m, l), (m, r)},
road.curvature=0),

and "approaching freeway exit" by

(vehicle.speed=++,
object.type="obstacle",
object.location={(m, l), (f, l)},
road.curvature=1).

Finally, the impacts, i.e. effects, to be determined for this

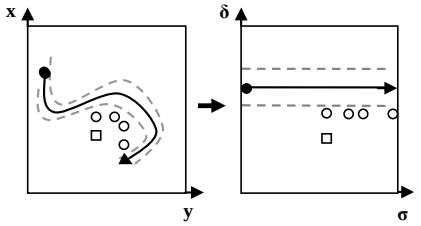

Figure 4. Transformation of the Coordinate System

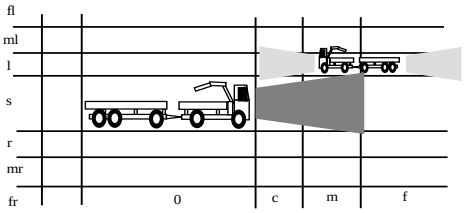

Figure 5. The Qualitative Spatial Representation

part of the analysis are modeled by **collision constraints** in the environment model: they can simply be encoded as

Equal (vehicle.position, object.position),

on the potential locations of the vehicle and of another object. If these effects are consistent with a scenario, this means the impact ranges have a non-empty intersection and, hence, a collision is possible. Impacts can be defined specific to the type of objects and, thus, determine the severity of the impact.

5.3 Results of Impact Analysis

Impact analysis is also carried out by the Raz'r FMEA engine in the same way as hazard analysis, with the impacts as relevant effects.

An example for the automatically generated results is shown in Table 5. The results have been successfully evaluated based on a set of standard situations supplied by our industrial partners. Such a gold standard for evaluation is limited and may appear subjective and vague. However, due to the nature of the task, this is the only criterion for evaluation, and we cannot expect a more rigorous and objective reference. Of course, other examples and different practice may require a revision and refinement of the current distinctions, esp. in the physical model and the spatial representation. However, so far, no evidence occurred that the current foundation for our tool would need substantial revisions.

Table 5. Partial Results of Automatic Hazard Analysis for "Approaching Freeway Exit, High Speed, Braking"

Part	Failure Mode	Hazard / Impact
CrankShaft1	Broken	:collision_with_object
Clutch1	ClutchStuckOpened	:collision_with_object
Clutch1	ClutchStuckClosed	:>>no system level effects<<
GearBox1	StuckReverse	:collision_with_object
GearBox1	StuckNeutral	:collision_with_object
GearBox1	StuckForward	:>>no system level effects<<
Retarder1	RetarderStuckNotEngaged	:collision_with_object
Retarder1	RetarderStuckEngaged	:>>no system level effects<<
Brakes1	StuckNotEngaged	:collision_with_object
Brakes1	StuckEngaged	:>>no system level effects<<

6 Outlook

The results obtained have increased industrial interest in pursuing this line of research. We are currently preparing a collaborative project involving an automotive manufacturer, an automotive software engineering company, a certification company, and academic partners (representing model-based approaches from AI and software engineering) that aims at providing foundations for tools for functional safety that are compliant with the standards and processes. This will require embedding the analytic part covered here with higher-level models from design and also feeding back its results to the process of responding to severe shortcomings by developing appropriate safety functions. Steps towards formal foundations for an integration of the model-based systems and software engineering technologies will be required for this.

Acknowledgements

We would like to thank our partners from ITK for providing their domain knowledge and their patience and Alessandro Fraracci for his support. Special thanks to Oskar Dressler (OCC'M Software) for providing a very efficient implementation of the FMEA algorithm.

References

[Dobi et al., 2013] Dobi, S., Gleirscher, M., Spichkova, M., Struss, P. Model-based Hazard Analysis and Risk Assessment Technical Report TUM-I1333, Technische Universität München, 2013.

[Dylla et al., 2007] Frank Dylla, Lutz Frommberger, Jan Oliver Wallgrün, Diedrich Wolter, Berhard Nebel and Stefan Wölfl. SailAway: Formalizing Navigation Rules. In: *Proceedings of the Artificial and Ambient Intelligence Symposium on Spatial Reasoning and Communication*, AISB, p. 470-474, Newcastle upon Tyne, UK, 2007.

[Kurtoglu-Tumer-Jensen, 2010] Kurtoglu, T., Tumer, I.Y., Jensen, C.: A functional failure reasoning methodology for evaluation of conceptual system architectures. Research in Engineering Design 21.4 (2010): 209-234.

[ISO-26262, 2011] ISO, "ISO 26262", international Standard ISO/FDIS 26262, 2011. http://www.iso.org/

[Picardi et al., 2004] C. Picardi, L. Console, F. Berger, J. Breeman, T. Kanakis, J. Moelands, S. Collas, E. Arbaretier, N. De Domenico, E. Girardelli, O. Dressler, P. Struss, B. Zilbermann: AUTAS: a tool for supporting FMECA generation in aeronautic systems. In: Proceeding of the 16th European Conference on Artificial Intelligence August 22nd - 27th 2004 Valencia, Spain, pp. 750-754

[Price, 2000] Price, C. Autosteve: automated electrical design analysis. In Proceedings ECAI-2000, p.721-725, 2000

[Raz'r, 2013] http://www.occm.de/

[Struss, 2004] Struss, P.: Models of Behavior Deviations in Model-based Systems. In: Proceeding of the 16th European Conference on Artificial Intelligence August 22nd - 27th 2004 Valencia, Spain, pp. 883-887

[Struss-Fraracci-Nyga, 2011] Struss, P., Fraracci, A., Nyga, D.: An Automated Model Abstraction Operator Implemented in the Multiple Modeling Environment MOM. In: 25th International Workshop on Qualitative Reasoning, Barcelona, Spain, 2011.

[Struss-Fraracci, 2012] StrussP., Fraracci, A.: Automated Model-based FMEA of a Braking System. 8th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes (Safeprocess 2012), Mexico City, 2012.

[Struss, 2013] Struss, P.: Model-based Analysis of Embedded Systems: Placing it upon its Feet instead of on its Head - An Outsider's View - In: 8th International Conference on Software Engineering and Applications (ICSOFT-EA 2013), Reykjavik, Iceland, July 29-31 2013.