PANORover: Autonomous Driving System Development Platform

Marc Zeller
Siemens AG, Technology
Munich, Germany
marc.zeller@siemens.com

Olexiy Kupriyanov
Siemens AG, Digital Industry Software
Nuernberg, Germany
olexiy.kupriyanov@siemens.com

Norbert Beck
Siemens AG, Digital Industry Software
Nuernberg, Germany
norbert.beck@siemens.com

***Abstract*—The complexity of heterogeneous embedded systems as well as the introduction of AI-based functions to realize ADAS or autonomous driving features pose new challenges for the safety assessment process. In this demo, we illustrate the analysis of a complex system in terms of function safety (ISO 26262) and Safety Of The Intended Functionality (SOTIF, ISO 21448) with the model-based *Component Fault Tree (CFT)* methodology using a self-driving toy vehicle (the PANORover).**

***Index Terms*—heterogeneous systems, AI, safety analysis, SOTIF, modeling, fault tree analysis, CFT**

I. Introduction

The main objective of the *PANORover* called demonstrator is to provide real-world application cases for the assessment, optimization and final validation of the model-based methodology developed in PANORAMA project[1] in the context of Siemens' related modeling and design software tools. In order to be able to address the main engineering challenges related to complex heterogeneous hardware/software systems, an exemplary development of a self-driving toy vehicle (the PANORover) shall be shown. As the development of modern complex systems is usually undertaken in the context of an embedded development process with multiple levels of modeling abstraction and various development stages, the demonstrator shall also cover the integration chain of various design artifacts such as requirements, safety models, various types of architecture models, analysis results, and implementations. Thus, the most important aspects to be highlighted during the development of the PANORover can be represented in terms of the following areas: requirements management, embedded system architecture modeling, software architecture, heterogeneous hardware, timing and safety analyses.

In this demonstration, we will focus on the safety analysis of the heterogeneous hardware/software platform of the PANORover to show how complex, heterogenous systems incorporating AI-based functions can be assessed in terms of safety.

II. Heterogeneous HW/SW Platform of the PANORover

The rover vehicle (see Fig. 1) realizes an automated braking and collision avoidance ADAS (Advanced Driver Assistance Systems) use-case with some elements of Autonomous Driving (AD). A driver can control the vehicle remotely by issuing steering, acceleration, and deceleration commands. Whereas the rover vehicle detects approaching obstacles and traffic sings in the direction of its current movement and it reacts autonomously either by automated braking or by adjusting its current speed accordingly in a reasonable time. Such an autonomous reaction of the rover overrides any contradicting control commands of the driver.

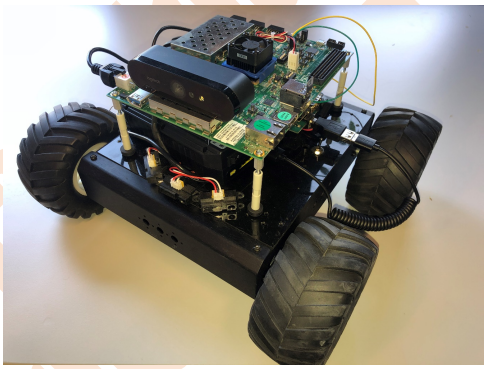

Fig. 1. Hardware setup of self-driving rover platform

Along with an appropriate mechanical and electrical setup the rover vehicle comprises a composite embedded hardware architecture based on two control units (CUs). The CUs are implemented on separate processor boards that are interconnected via CAN bus as shown in Fig. 2: *Low-Level Control Unit* (LLCU) and *Collision Avoidance Electronic Control Unit* (CA-ECU) implementing the ADAS/AD scenario.

The LLCU realizes rudimental control functions to interpret the incoming driver's commands and accordingly to generate pulse-width modulated signals to independently drive two pairs of connected DC motors. One pair of DC motors is equipped with incremental encoders that allow the LLCU software to measure the actual speed of the rover. Furthermore, the distance to approaching objects can be measured using two triples of infrared sensors (IRSs) each placed on the front and on the back sides of the vehicle. The LLCU captures the IRSs signals, combines the values together with the received driver's commands and the estimated actual speed, and transmits then the aggregated setpoint data packed into respective CAN messages to the CA-ECU.

[1]https://www.panorama-research.org/

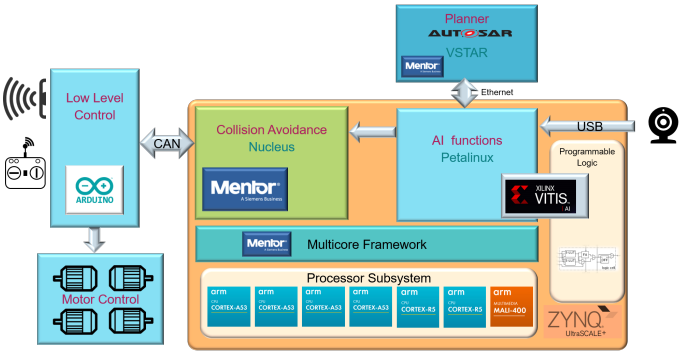

Fig. 2. Rover system architecture and allocation of functions to operating systems

The CA-ECU emulates behavior of a centralized ECU with a heterogeneous hardware architecture. Its main task is to continuously derive an appropriate autonomous reaction based on the continuously sensed environment model of the rover and to communicate the reaction commands back to the LLCU to be safely executed there. Additionally to the setpoint data prepared by the LLCU, a dedicated (AI-based) object detection module shall use visual data captured by the front camera in order to recognize traffic signs and to detect other types of larger distance objects, that hardly can be detected by the IRSs from the low-level control. The perception tasks are realized as AI functions using a Deep Neutral Network (DNN). The multiple sources of sensed information are sampled with various periodicities, that shall be merged by means of sensor fusion functionality to obtain a more confident representation model of the environment. Based on the outputs of the object detection and sensor fusion functions the planner module then computes the actual reaction command which is then translated into lower-level speed and turn limits that can be directly realized by low-level control.

III. Safety Assurance

In order to assess complex, heterogeneous embedded systems such as the PANORover in terms of safety, we use a modular and hierarchical approach to specify and analyze the failure behavior of the system.

With *Component Fault Trees (CFTs)* there is a model- and component-based methodology for fault tree analysis [1], [2], which supports reuse by a modular and compositional safety analysis strategy. CFTs are Boolean models associated with system development elements such as components [3], [4]. It has the same expressive power as classic fault trees. Like classic fault trees, CFTs are used to model failure behavior of a system. This failure behavior, including their appearance rate, is used to document the absence of unreasonable risk of the overall system. In CFTs, a separate so-called *CFT element* is related to a component [3]. Failures that are visible at the outport of a component are modeled using Output Failure Modes which are related to the specific outport. To model how specific failures propagate from an inport of a component to the outport, Input Failure Modes are used. The internal failure behavior that also influences the output failure modes is modeled using Boolean gates such as OR, AND and M-out-of-N as well as so-called Basic Events. Basic Events model failure modes that originate within a component. Each Basic Event can be assigned a failure rate, e.g. the *Mean Time Between Failures (MTBF)* or the *Failure In Time (FIT)*.

Furthermore, to analyze heterogeneous embedded systems incorporating AI functions the *Safety Of The Intended Functionality (SOTIF)* (ISO 21448 [5]) must be considered in addition to functional safety (accoding to ISO 26262 [6]). SOTIF is defined as the absence of unreasonable risk due to hazards resulting from functional insufficiencies of the intended functionality. In order to assess the safety of systems incorporating AI, hazards coming from failures as well as hazards resulting from functional insufficiencies of the intended functionality must be mitigated sufficiently. Therefore, we need to extend safety analysis techniques in order to create cause-effect-relationships for individual failures as well as for functional insufficiencies and system hazards for the specified system. In a CFT model both failures and functional insufficiencies can be represented. Hence, with the CFT methodology we are able to describe cause-effect-relationships and mitigation schemes on different system levels for individual failures as well as functional insufficiencies and system hazards for the specified heterogeneous embedded system. Hence, we can analyze a system qualitatively in order to show that all hazards are mitigated sufficiently.

We demonstrate this safety analysis approach in the PANORover demo scenario, in which a Capella[2] model of the system architecture of the PANORover is enriched with a CFT model to conduct an analysis of the system in terms of safety and to check, if all safety requirements are fulfilled by the system design.

Acknowledgment

The research leading to these results has received funding from the Federal Ministry for Education and Research (BMBF) under grant agreement 01IS18047D and by Vinnova under registration number 2018-02228 in the context of the ITEA3 EU-Project PANORAMA.

References

[1] International Electrotechnical Commission (IEC), "IEC 61025: Fault Tree Analysis (FTA)," 2006.
[2] E. Ruijters and M. Stoelinga, "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools," *Computer Science Review*, vol. 15–16, pp. 29 – 62, 2015.
[3] B. Kaiser, P. Liggesmeyer, and O. Mäckel, "A new component concept for fault trees," in *Proceedings of the 8th Australian Workshop on Safety Critical Systems and Software*, 2003, pp. 37–46.
[4] B. Kaiser, D. Schneider, R. Adler, D. Domis, F. Möhrle, A. Berres, M. Zeller, K. Höfig, and M. Rothfelder, "Advances in component fault trees," in *Proceedings of the 28th European Safety and Reliability Conference (ESREL)*. Taylor & Francis (CRC Press), 2018.
[5] International Organization for Standardization (ISO), "ISO/PAS 21448 – road vehicles-safety of the intended functionality," 2019.
[6] ——, "ISO 26262: Road vehicles – Functional safety," 2018.

[2]https://www.eclipse.org/capella/