Rapid Innovation Toolkit for the development of dependable cooperative applications

Dominique Seydel, Gereon Weiss
Application Architecture Design & Validation
Fraunhofer ESK
Munich, Germany
{dominique.seydel, gereon.weiss}@esk.fraunhofer.de

***Abstract*— Cooperative applications have an enormous potential to improve future mobility systems. Though, special challenges regarding safety and security arise out of the connectivity and the distribution of the application among heterogeneous systems. These include expensive and time-consuming development and test phases. Especially, the debugging of an application, whose sub-functions are located on heterogeneous and partially mobile systems, requires a new kind of testing environment. The test and validation of the overall application is complex, as the wireless link implies varying timing behaviour and less data confidence. For this purpose, the proposed testbed integrates the DANA ("Description and Analysis of Networked Applications") Framework to achieve a central overview of the overall application and the behaviour of all systems involved. This software tool kit is able to find deviations from the specified behaviour and also it can instantly locate and identify erroneous functions. In this paper, we present a solution for the complete development cycle of cooperative automotive systems together with an exemplary development flow for safety and security testing.**

Keywords—automotive safety; cooperative applications; security testing; validation; autonomous systems; ITS

I. INTRODUCTION

In comparison to the development of traditional ADAS functions, testing and simulation of dependable connected applications have to consider the interaction of heterogeneous systems that are distributed within a wirelessly networked architecture. As the communication link is more unreliable in contrast to common input sensors, the application has to cope with varying timing behaviour and less data confidence. However, the higher complexity in the development process of cooperative applications is justified by several advantages. They result from the aspect that foreign road users are no longer observed only from the outside in order to predict their behaviour, but they provide insights into their status, their intentions and their participation in cooperative maneuvers. This results in an increased reliability of predicted vehicle movements, which in turn can be used for safety functions and allow an increased reaction time of dependable safety functions. In terms of driving comfort, the traffic participant's cooperation allows a foresighted maneuver planning. As depicted in Fig. 1, the achievable set of driving maneuvers decreases when a possible impact approaches. The shorter the time to a possible impact, the less safe driving maneuvers are executable, as each maneuver has a required execution time. The inclusion of cooperative applications has the advantage that important information is available earlier and the maneuvering space is increased. Therefore, one of the questions that can be evaluated within the testbed is how much data confidence of the received information is required for a time-critical decision.

Another current question is whether parts of the application functionality can be outsourced from the automotive platform to enable a faster development and update cycle, reduce hardware complexity and to group functions across vehicles. The presented testbed can be used to evaluate whether a function can be moved to the cloud or whether it is more advantageous to deploy the function to an edge or fog component. The architectural diversity resulting from this outsourcing of functions raises new questions regarding the dependability and security of the application, the end-to-end

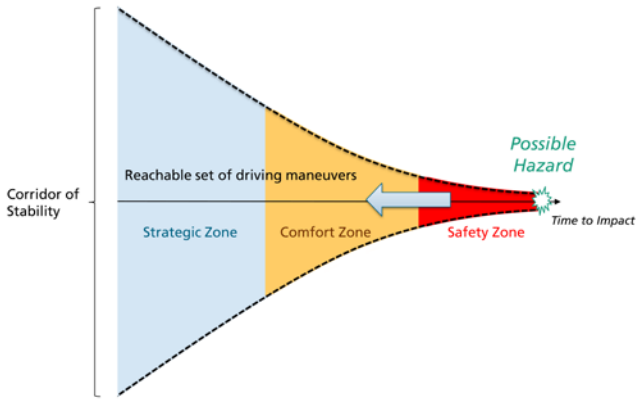

Fig. 1 Connected applications enable an increased reaction time for safety mechanisms

quality of the service, scalability and the comprehensive debugging process. The evaluation of the distributed application within the proposed testbed, gives objective statements on these questions.

By the current state of available tools, the development, testing and certification of autonomous systems is complex, costly in terms of time and equipment, potentially hazardous and often incomplete. For instance, if it comes to complex applications that require a distributed consensus, e.g. Merging Assistance [1], an application distributed among various foreign entities must be validated. Thus, it appears that development and simulation environments are not yet ready to rapidly develop prototypes of cooperative driving functions.

Therefore, we provide an approach for an integrated testing environment that can cover the whole innovation cycle for prototype development of cooperative automotive systems. Incorporating safety and security aspects, it starts from the design of applications over simulation to integrating and validating the respective prototypes.

The following Chapter II gives an overview on an efficient approach for the development of cooperative applications. Further aspects of the simulation and testing phase are discussed in Chapter III. The current scope of software analysis is presented in Chapter IV. We conclude our work with Chapter V and provide a brief outlook to next steps.

II. Rapid Application Development

A. Application Development Cycle

The testbed supports all steps of the Vehicle-to-X (V2X) application development life cycle, beginning from application design, continuous integration into simulation environments, testing environments over tools performing functional and security analyses, up to a secure deployment and update process. The application development flow of the innovation- and testbed concept is shown in Fig. 2. It is also designed to only use single aspects in a building-block style when developing innovative applications.

The testbed provides the ezCar2x® framework, described in [2], that allows testing connected applications within a simulation environment, using network and traffic simulation as well as within a field test environment, deployed on in-vehicle communication hardware. The usage of the ezCar2x® framework enables a continuous development cycle implementing the DevOps approach, described in [3]. The DevOps approach is designed to integrate the application under development into the field test environment to evaluate its behaviour under real conditions. The DevOps approach also enables an iterative refinement of the application incorporating results, e.g. erratic behaviour, and a diagnosis of the underlying cause of the failure.

Additional analysis tools enable to examine the developed application. On the one hand, our DANA tool for functional validation can be used iteratively in every testing step [3]. For the integration testing of the application, the analysis toolbox provides application security testing methods in order to detect software vulnerabilities in an early development stage. Using

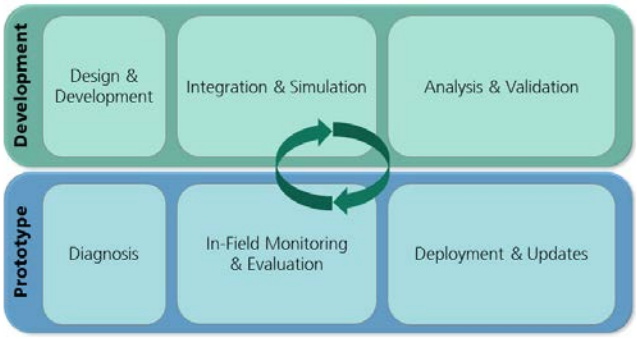

Fig. 2 Application Development Cycle using the Testbed Services

static application security testing (SAST) as well as dynamic application security testing (DAST) allows quick analyses during integration testing in order to detect potential software vulnerabilities.

One main feature of the testbed is a combined simulation and field testing approach, where virtual and real traffic participants can be tested in a synchronized environment. This feature is detailed in Chapter III.B.

Finally, the application is built and subsequently signed within the software repository and pushed to the update server, which is part of the back end. The update server again signs the application and deploys it to V2X devices.

B. Application Design

For the initial development step of designing a cooperative driving function, the testing environment comprises interfaces to common automotive modelling tools, like Matlab Simulink or ADTF. The deployed application uses the ezCar2x® framework, an ETSI ITS (Intelligent Transport Systems) compliant communication stack, which can either run on real communication hardware or on a virtual node within a network simulation. Furthermore, application security testing can be conducted with static and dynamic methods.

If enhanced safety mechanisms are required for the intended application, state-of-the-art software methods, e.g. graceful degradation strategies [4] or model-based communication [5], can be incorporated into the application model within this development step as well. For example, a connected application with safety-critical functionality, as Platooning, strongly depends on Quality of Service (QoS) parameters of the communication link. Our safety function for Resilient Control uses these QoS parameters, such as the current Packet Loss Rate (PLR), to decide which degradation mode is sufficient, e.g. readjusting the distance to the vehicle ahead. The safety mechanisms for resilient control are developed as generic component and can be integrated into common automotive software architectures, as AUTOSAR, AUTOSAR Adaptive and further concepts. Also existing architectures from non-safety domains like infotainment, as developed by the GENIVI Alliance, can be integrated to handle the unreliability of the communication link.

Another aspect that is getting more relevant for application design and validation are so-called Plastic Architectures. Parts

of an application can be distributed over several entities. For example, in the case of a Collision Warning application [6] this includes the interaction of the originating, the warning and optionally edge or cloud components form the overall function. As the specific architecture may frequently change, the architectures change depending on the context, thus becoming formable or plastic. Also, in future parts of the application can be dynamically relocated during runtime, e.g. from cloud over edge to in-vehicle components. Thereby, the system boundaries dynamically change depending on the current communication relations. Although, there are concepts to solve the underlying network aspects [7], these runtime conditions have already to be covered within the design phase of the specific applications.

III. Simulation & Testing

One goal of simulation and testing for cooperative automated driving is to achieve a fail-operational behaviour of the application, even if the context information is unconfident. Therefore, the coverage of the test cases used within the simulation environment and during virtual testing should be as realistic and as comprehensive as possible. This is achieved by (automatically) defining reference scenarios and generating variations from them, e.g. by stochastic variations [8].

One of the parameter variations is the realistic behaviour of the communication channel during a certain driving scenario. Therefore, a network simulation tool, e.g. ns-3 or OMNET++, and a traffic simulation tool, e.g. SUMO, VTD or CarMaker, are integrated into the simulation environment. Our testbed could also be integrated with other microscopic and macroscopic traffic simulation tools, as each of them has advantages when testing a specific connected application.

A. Simulation Environment

The suggested concept combines three different simulation aspects into one integrated simulation environment.

The first component is a traffic simulator that is used to model and run driving test cases on a realistic road network.

The second component is a network simulation tool for evaluating applications under real communication conditions. For the heterogeneous use of common vehicular communication technologies, e.g. 802.11p, 4G or LTE, the ezCar2x® framework provides additional network layer components. The network simulation tool also facilitates interfaces to control traffic simulation and integration hardware-in-the-loop tests or vehicle-in-the-loop tests (as for Virtual Platooning), e. g. including RSUs.

The third component of the testbed is for test control. Traces from all simulation components are monitored and analyzed within the test control component. For ensuring the security of cooperative systems, testing covers white-, gray-, and black-box approaches (e.g. Data-Flow Analysis, Fuzzing or Penetration Testing). In order to validate the applications, test cases have to reach full coverage and should therefore be generated (semi-)automatically for each application.

B. Integrated and Hybrid Simulation

As already described in Chapter II.A, the application implementation is deployed on each virtual V2X node within the network simulation environment. Together with the ezCar2x® Framework each virtual node can be equipped with developed applications and also with V2X communication ability. Hence, its interaction with other nodes can be simulated as realistically as possible.

The interaction between all virtual nodes is realized using a virtual wireless channel. Thereby, we consider the specific characteristics of each communication technology by using individual channel models, e.g. dedicated models for ITSG5, LTE or 5G. This virtual wireless channel can also be used to integrate real hardware into the simulation by using a channel proxy and creating a mirror node for each hardware component within the network simulation.

The network simulation is coupled with a macroscopic traffic simulation for large scale traffic scenarios, e.g., to test security mechanisms for V2X messages, and with a microscopic traffic simulation for smaller driving scenarios, e.g. Cooperative Merging [6] or Platooning. The coupling via a control interface is needed to synchronize the behaviour of communication nodes and traffic participants in each simulation for the given driving scenario.

The environment can be extended with hybrid simulation capabilities by including hardware-in-the-loop. A RSU comprising an application, e.g. Smart Lighting, can be integrated into the simulation loop, by connecting it to the wireless channel interface of the simulation environment. The RSU can again interact with further communication hardware, e.g. test vehicles that are in communication range. The RSU can also be connected with sensors, that are integrated into the testing environment and which can provide status data to generate event messages, e.g. Decentralized Environmental Notification Messages (DENMs).

C. Sensor Integration

The effectiveness of a connected applications' simulation depends on how realistic the input data for a certain driving scenario is. The input data from distributed sources, e.g. the status data within Cooperative Awareness Messages (CAMs) from other vehicles or roadside sensors, have to be synchronized during recording and replay phase. Synchronization is required to establish the intended driving conditions for the application under test.

Within our simulation environment, vehicle sensors and infrastructure sensors are integrated as components in ezCar2x® via generic sensor interfaces. When recording test data, the real sensors can be easily integrated into the synchronized recording process. The same setup can be used in field tests, where infrastructure sensors are usually integrated into RSUs to provide their environment data. Thus, virtual, hybrid and integration tests can be carried out with little effort.

IV. Software Analysis

In each step of the development process it is advantageous to perform additional analyses to obtain detailed knowledge of the overall system and the application behaviour for debugging, security and validation purposes. This chapter gives an overview of our methods and tools for software analysis.

An exemplary flow of the development process for an application is shown in Fig. 2. The application under development can be prototyped as implemented source code or as a software model to be further developed and optimized within the testbed.

A. *Monitoring and Functional Validation*

For software validation and verification, model-based techniques are advantageous during the design and integration phase. Our DANA platform [9], an open and modular environment based on Eclipse, is a tool built for specifying and analyzing networked applications. For this purpose, the specified valid behaviour of the application is described as a layered reference model. This model provides a basis for further model-based development steps. On the one hand, it can be used for various transformations of behaviour models, e.g., for generating test cases or code for running simulations. On the other hand, it can be used for static analyses to check conformance to modeling guidelines, metrics for interfaces, and the compatibility of behaviour models. The model-based approach also allows a quick integration of new message sources, e.g. additional communication protocols or wireless channels. Furthermore, the DANA tool can be used to verify and validate software interface behaviour, as messages in these interfaces can contain complex data and complex interactions.

In our proposed testbed we use DANA as a central monitoring tool, as visualized in Fig. 3, to have all the status, debug and behaviour information, the error messages and timing data centrally available from each component. This aggregation helps to simplify and to speed up the debugging process during development and runtime. Further validation checks can be applied on this collected data, as described in the previous paragraph. Once the diagnosis of the underlying cause of the failure has been made, the error can be corrected and the new software version is securely deployed to the faulty component.

V. CONCLUSION

We provided an approach for an integrated testing environment that covers the whole development process for prototyping and testing cooperative functions. Incorporating safety and security aspects starting from the design phase, the complex task of simulation and cooperative applications and several testing steps have been described. Finally, a software solution for the validation and deployment of the prototypes was presented, which makes tools available for the whole development cycle. Thereby, a toolkit is provided that is intended to rapidly bring an idea for a connected application into a prototype with a decreased investment risk.

In future, all testbed services described in the previous chapters could also be made available as an online web-service. For this purpose, a next step of the described solution is to enable access to configurable or pre-configured simulations via an online service. On this website, users could remotely control simulation parameters, define new scenarios and get qualified evaluation results. This ongoing development addresses the rapid development of connected applications by abstracting technology know–how and increasing time-to-market speed. Thus, innovators and developers can concentrate on the actual function and idea of the intended application and are able to experience, improve, and validate their solution in early stages prior to competitors.

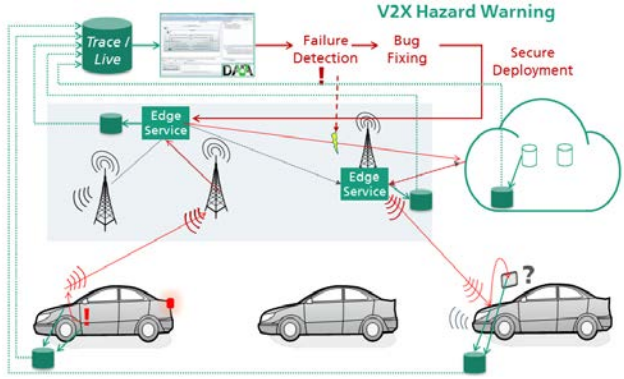

Fig. 3 The DANA Platform speeds up the debugging process of connected applications

ACKNOWLEDGMENT

This project was partially funded by the Bavarian Ministry of Economic Affairs and Media, Energy and Technology within the Fraunhofer High Performance Center “Secure Networked Systems”.

REFERENCES

[1] Ntousakis, I. A., Nikolos, I. K., & Papageorgiou, M. (2017). Cooperative Vehicle Merging on Highways-Model Predictive Control (No. 17-00930).

[2] Roscher, K., Bittl, S., Gonzalez, A. A., Myrtus, M., and Jiru, J. (2014). ezCar2X: Rapid-Prototyping of Communication Technologies and Cooperative ITS Applications on Real Targets and Inside Simulation Environments, In: 11th Conference Wireless Communication and Information. vwh, pp. 51 – 62.

[3] SafeTRANS (2018) Autonome und lernende Cyber-Physical Systems (ACPS): Herausforderungen in Entwicklung, Test und Zertifizierung. http://news.safetrans-de.org/ausgabe-2018-01/Autonome_CPS.html, Access: 17.8.2018.

[4] Schleiss P., Drabek C., Weiss G., Bauer B. (2017) Generic Management of Availability in Fail-Operational Automotive Systems. In: Tonetta S., Schoitsch E., Bitsch F. (eds) Computer Safety, Reliability, and Security. SAFECOMP 2017.

[5] Moradi-Pari, E., Mahjoub, H. N., Kazemi, H., Fallah, Y. P., and Tahmasbi-Sarvestani, A. (2017). Utilizing Model-Based Communication and Control for Cooperative Automated Vehicle Applications. IEEE Transactions on Intelligent Vehicles.

[6] Zhang, R., Cao, L., Bao, S., & Tan, J. (2017). A method for connected vehicle trajectory prediction and collision warning algorithm based on V2V communication. International Journal of Crashworthiness, 22(1), 15-25.

[7] An, X., et al. (2017) On end to end network slicing for 5G communication systems. In: Transactions on Emerging Telecommunications Technologies, 28. Jg., Nr. 4.

[8] Damm W., Heidl P. (Hrsg) (2017) Positionspapier und Roadmap zu „Hochautomatisierte Systeme: Testen, Safety und Entwicklungsprozesse“, SafeTRANS e. V. http://www.safetrans-de.org/de/Aktuelles/?we_objectID=2, Access: 17.8.2018.

[9] Drabek, C., Weiss, G. (2017) DANA - Description and Analysis of Networked Applications. In: International Workshop on Competitions, Usability, Benchmarks, Evaluation, and Standardisation for Runtime Verification Tools (RV-CuBES), pp. 71-80.