Design and Verification of a Health-Monitoring Driver Assistance System

Helena Gruhn*, Daniel Stöhr*, Mehmet Gövercin†, and Sabine Glesner*
* Software Engineering for Embedded Systems, Technical University of Berlin,
† Geriatrics Research Group, Charité - Universitätsmedizin Berlin,
{helena.gruhn, daniel.stoehr, sabine.glesner}@tu-berlin.de, mehmet.goevercin@charite.de

***Abstract*—Health-monitoring driver assistance systems support an independent and self-determined lifestyle enhancing the driver's safety. These systems are health-critical and need to guarantee correct behavior in emergency situations such as heart attacks. Furthermore, they have to be adjustable and extendable with respect to integrated functionalities to fit individual and changing needs. We present a concept for a mobile, service-oriented driver assistance system with dynamic network behavior. Additionally, we introduce a verification approach to ensure correct behavior.**

I. Introduction

Mobility is a central aspect of a self-determined lifestyle. A rising number of assistive technologies have been developed to support traffic participants, in particular mobile elderly persons, in hazardous situations, e.g., distance-warning or autonomous emergency stop systems [1]. In particular, the number of elderly causing traffic accidents with personal injuries increased in Germany by 4.5% between 1990 and 2005 [2]. A driver assistance system has to be modular and extendable to fit the individual needs of each person. In particular, elderly with developing multimorbidities need a support network that is easily adjustable to their changing demands. Furthermore, assistance systems have to be trustworthy and reliable. In particular, in life-endangering situations their correct behavior has to be guaranteed. It must be ensured that emergency reactions are triggered in time and that all relevant messages within the system and to the surrounding environment reach their destination eventually. To reach this goal, it is necessary to verify the correct behavior of such systems.

In this paper, we present our *service-oriented health-monitoring driver assistance system (DAS)* combined with our approach for the verification of its correct functionality. Our main contribution is to provide a design method for these assistance systems such that current verification techniques can be applied. Hence, we significantly increase the safety and reliability of theses systems. Hereby, we focus on correctness w.r.t. network topology, functional, and timing behavior.

II. A health-monitoring driver assistance system

Our DAS increases driving safety by reacting to specific emergency situations, e.g., heart attacks, by triggering an emergency stop. To assess the driver's health status, vital parameters, e.g., the heart beat, must be observed [3]. As elderly people often show a highly individual multimorbidity, we need our DAS to be adaptable by adjusting the choice of observed vital parameters to individual needs.

To this end, we design our DAS as a *service-oriented sensor-actuator network* [4]. Each node has sensing and/or actuation capabilities. Vital parameters are registered using the respective *sensors*. *Actuators*, i.e., a speaker and the autonomous emergency stop system (ES system), interact with the environment. The controller is a special, central network node that processes the readings of all connected sensors and triggers emergency actions. For this purpose, it implements a set of rules specifying which specific sensor readings imply emergency situations. The network is *service-oriented*, requiring all nodes to offer a unified and platform-independent interface to ensure interoperability. This allows for a dynamic interaction of *mobile nodes*, e.g., sensors directly carried by the driver, with the network.

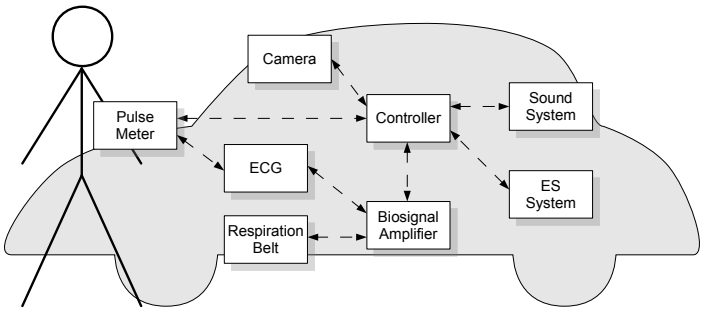

Fig. 1. An example for sensors used within the Driver Assistance System

A. An Assistance System for Cardiac Patients

In Figure 1, we illustrate a DAS designed to assist people with a high risk of heart attacks. The network consists of different nodes which observe the driver's vital parameters. An *electrocardiogram* (*ECG*) monitors the heart activity and the pulse. A *respiration belt* captures the breathing. Furthermore, the network includes a *camera* to detect microsleep and tunnel vision by observing eye and head movements, a *biosignal amplifier*, and a *controller* that provokes emergency actions. We assume that the driver also has an assistance system securing his health status at home requiring the continual application of a portable *pulse meter*. This device dynamically joins or exits the network when the driver enters or leaves the car, respectively. While in the network the mobile pulse meter remains into a standby mode to avoid redundant measurements of the pulse and to reduce energy consumption. It restarts after exiting the network or when a malfunction of the ECG is detected to avoid data loss. The system is able, e.g., to react to a heart attack by triggering a safe emergency brake or to voice an alarm to alert the driver after detecting microsleep.

III. Verification Framework

Within the DAS several *safety-critical* processes occur. We must guarantee that neither the driver, nor other road users

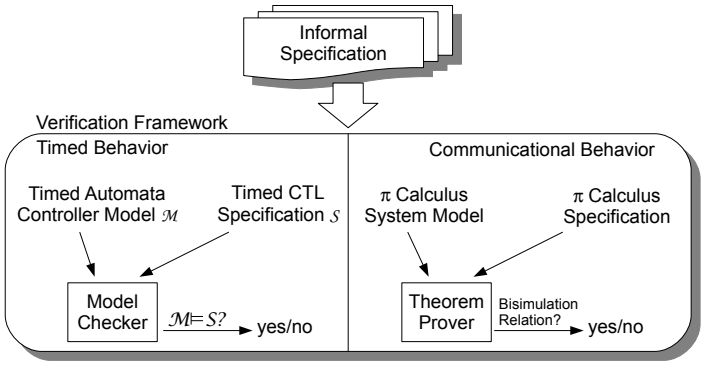

Fig. 2. Our Framework for verifying the Driver Assistance System

are harmed due to incorrect system behavior, e.g., through a delayed triggering of an alarm signal. As testing only shows the presence of errors, but not their absence, we propose a formal verification framework to prove the correctness of the DAS. To the best of our knowledge no formal framework for service-oriented networks exists. Existing work on wireless networks either does not cover topology changes [5], [6] or is not platform-independent [7].

We visualize our verification framework in Figure 2. The starting point is an *informal requirements specification* created by physicians and other domain experts. The informal specification consists of (1) a set of medical (and time-related) rules describing, e.g., which combination of vital parameters implies a heart attack, and (2) a set of network requirements stating which communication paths have to be available on which point of execution. We use two specialized verification methods to handle both types of requirements separately. This is necessary as no actual formalism is able to cover time-related controller behavior on the one hand, and highly dynamic network behavior on the other hand. In the following, we shortly outline both verification approaches and how we integrate them into our framework.

A. Functionalities and Timing Behavior

We verify functional and time-related mechanisms implemented within the controller by relying on the well-established concept of *model checking*. Firstly, we construct a model of the controller's behavior as a *timed automaton* [8] (*TA*). The model describes how the controller reacts to events occurring within the network and in which situations events are triggered by the controller itself. Additionally, timing behavior, e.g., timeouts or delays, can be expressed within the model. Secondly, we construct the corresponding behavioral requirements from the informal specification using the temporal logic *Timed CTL* [9] (*TCTL*). We perform the verification process automatically by using the model checker *UPPAAL* [10]. UPPAAL takes a TA model M and a TCTL specification S as input and automatically calculates if M fulfills S, in short $M \models S$. If M does not fulfill S a counter example is generated so that conclusions on the error source of the implementation can be directly drawed.

B. Dynamic Network Behavior

We also need to verify that our DAS functions correctly after modifications of the network topology. These could be the extension of the network through the addition of a new node or a breakdown of a previously established wireless connection. Here, we only consider the ability of a node to receive and send messages by examining the existing communication links. We assume that the internal node behavior is correct. Our idea is to extract all information concerning communication channels and node reachability from the informal specification by building a π-calculus [11] specification. Then, we model our concrete implementation using a π-calculus variant as described in [12]. We choose the π-calculus as basis for our verification approach because it is a formally defined, compositional and well-established formalism. We will prove the correctness of our system by showing a bisimulation relation between the specification and the model. For a semi-automatic proof we will use a theorem prover.

IV. Conclusion & Future Work

In this paper, we presented an approach for the design and verification of a DAS. With that, we are able to develop individual and safe assistance systems supporting elderly to preserve their mobility. We plan to continue our work by deriving concrete rules for our heart attack scenario and by evaluating our framework with these.

Furthermore, we plan to improve the capabilities of our framework by including a method we developed for automatically generating correct controller models [13]. With that, we will be able to to decrease development cost and time by directly generating correct models out of given specifications.

References

[1] P. Waldmann et al., "Der Nothalteassistent-abgesichertes Anhalten bei plötzlicher Fahrunfähigkeit des Fahrzeugführers," in *Ambient Assisted Living-AAL*, 2010.

[2] J. Schade and A. Engeln, *Fortschritte der Verkehrspsychologie Beiträge vom 45. Kongress der Deutschen Gesellschaft für Psychologie*. VS Verlag für Sozialwissenschaften, 2008.

[3] H. Ehmen, M. Gövercin, M. Haesner, J. Kiselev, and E. Steinhagen-Thiessen, "Fahrleistungsrelevante Parameter im Alter," in *Ambient Assisted Living-AAL*. VDE VERLAG GmbH, 2010.

[4] A. Rezgui and M. Eltoweissy, "Service-oriented sensor-actuator networks: Promises, challenges, and the road ahead," *Computer Communications*, vol. 30, no. 13, pp. 2627–2648, 2007.

[5] D. Martinez et al., "Formal specification and design techniques for wireless sensor and actuator networks," *Sensors*, vol. 11, no. 1, pp. 1059–1077, 2011.

[6] M. Diaz et al., "A component framework for wireless sensor and actor networks," in *Emerging Technologies and Factory Automation, 2006. ETFA '06.*, sept. 2006, pp. 300 –307.

[7] O. Sharma et al., "Towards verifying correctness of wireless sensor network applications using insense and spin," *Model Checking Software*, vol. 5578, pp. 223–240, 2009.

[8] R. Alur and D. L. Dill, "A theory of timed automata," *Theoretical Computer Science*, vol. 126, pp. 183–235, 1994.

[9] R. Alur, "Techniques for Automatic Verification of Real-Time Systems," Ph.D. dissertation, Stanford University, 1991.

[10] G. Behrmann, A. David, and K. G. Larsen, "A tutorial on UPPAAL," in *Formal Methods for the Design of Real-Time Systems: 4th International School on Formal Methods for the Design of Computer, Communication, and Software Systems*, no. 3185. Springer-Verlag, 2004.

[11] R. Milner, *Communicating and Mobile Systems: The pi-Calculus*. Cambridge University Press, 1999.

[12] H. Gruhn and S. Glesner, "Towards a Formal Framework for Mobile, Service-Oriented Sensor-Actuator Networks," in *Formal Engineering approaches to Software Components and Architectures (FESCA@ETAPS2013)*, 2013.

[13] D. Stöhr and S. Glesner, "Planning in Real-Time Domains with Timed CTL Goals via Symbolic Model Checking," in *7th International Symposium on Theoretical Aspects of Software Engineering*, to appear.