Reliability and Security in Wellbeing Monitoring Embedded Systems *

José Santos[1,2,3][0009−0009−8982−181X], André Lourenço[1,3][0000−0001−8935−9578], and Tiago Dias[1,2][0000−0001−7445−5823]

[1] ISEL – IPL, Rua Conselheiro Emídio Navarro 1, 1959-007 Lisbon, Portugal
[2] INESC-ID, Rua Alves Redol 9, 1000-029 Lisbon, Portugal
[3] CardioID Technologies, Lisbon, Portugal

Abstract. The development of software for critical embedded systems is an effort-demanding task. The use of Validation and Verification (V&V) methods enables the automation of testing, ensuring enhanced reliability and security. In this work, we present the approach used for the formal verification of the new software architecture of CardioWheel, an Advanced Driver Assistance System (ADAS) that uses physiological signals for driver monitoring. Additionally we describe the setup that was used for the experimental verification step and discuss the resulting advantages, showing how V&V workflows can be implemented in this context.

Keywords: Cyber-Physical System · Critical System · Reliability · Validation and Verification · Uppaal · CardioWheel.

1 Introduction

In the last few years, the reliability and cyber security aspects of Cyber-Physical Systems (CPS) for critical domains, such as health and automotive, have been growing increasing awareness and interest in the research community due to their current applicability in multiple situations.

In the automotive domain, one key element that has been recognized as of the utmost importance to guarantee the correctness of the driving task, and thus improve road safety, is making sure that the driver of a vehicle is in a cognitive state complacent with the skill, readiness, and responsibility such task demands. Accordingly, the European Union has regulated the use of Driver Drowsiness and Attention Warning (DDAW) systems from 6 July 2022 for new types and from 7 July 2024 for all new vehicles.

CardioWheel is an Advanced Driver Assistance System (ADAS) developed by the company CardioID that uses the Electrocardiogram (ECG) acquired from the driver's hands to continuously detect drowsiness and perform biometric identity recognition [5, 4].This system is composed of three computing elements. A CPS performs acquisition (and preprocessing) of the ECG signals using two dry

* This work was partially supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) under project UIDB/50021/2020 and CardioID.

electrodes, seamlessly integrated into a steering wheel cover using conductive materials. A gateway collects the resulting sensitive data and securely sends it to a cloud server that can process it for several different applications, such as healthcare, automotive, biometrics and cyber security, or human factors engineering. This includes using the data also to re-train Machine Learning (ML) models. Furthermore, the CPS is managed by a service from the cloud server through the gateway. CardioWheel is protected under the patent WO2013109154A1 [8].

The software development for the initial version of the CPS component of CardioWheel followed a quite straightforward approach based on unit tests, with further integration tests carried out on a custom-built driving simulator [7]. However, due to the significant amount of involved sensors and the complexity of the event-triggering logic, the task of fully testing and validating the whole system has become quite complex and error-prone, resulting in substantial time investment. Recognizing that improvements could be made through the use of Validation and Verification (V&V) tools [2, 3, 9], CardioWheel entered the H2020 ESCEL VALU3S project[4], aiming to improve its reliability and security aspects using systematic and automated testing and verification workflows.

In this work, we describe the steps taken for the verification and formal validation of the new software architecture of the CPS component of CardioWheel. Starting at the design stage, the implementation of such development workflow brings several advantages, including improved robustness of system requirement's coverage, decreased time and cost spent on manual testing, and increased guarantees of system stability when integrating new features.

2 Formal verification of the CardioWheel architecture

2.1 Background

A verification process determines the quality of an architecture, including aspects of design verification and software quality analysis. Typically, this involves using formal methods and tools to verify the abstract architecture and conducting unit and integration testing. Conversely, the validation process establishes if the software meets the requirements, including usability and performance tests.

At the start of an effective verification workflow is the definition of system requirements and intended functionality, which can be expressed in use cases and requirement tables. This information is then leveraged to generate a state machine using Unified Modeling Language (UML), which aids in the verification process. While this methodology is useful to provide an abstract definition of a system, its real potential is truly unlocked when it is translated to be used with formal verification tools capable of evaluating if the system's design is compatible with the established requirements, such as Uppaal [1].

Uppaal uses Timed Automata (TA) models to represent the system's tasks. TAs are directed graphs representing system states in their nodes and conditional transitions in their edges. Uppaal exploits this structure to include real-time and

[4] https://valu3s.eu/

concurrency properties in the system's representation. Furthermore, the definition of several of these TAs, one for each task, connected through trigger nodes, allows the tool to explore timing-related requirement coverage and concurrency related faults before any line of code is written.

2.2 Formal verification using Uppaal

The functionality and requirements of the CPS component of CardioWheel were first modeled in two concurrent use cases, one regarding the acquisition, processing, and transmission of the ECG signal, and the other for the deployment of Over-The-Air (OTA) firmware updates. These models guided the development process of the new software architecture of the CPS to identify the several tasks that the software must execute to implement the use cases, as well as the definition of a requirements table listing the system properties that must be observed through the correct interplay between all the tasks.

The first step of this process consisted in the definition of the architecture using the state machine UML model. Then, this model was converted into a TA model, giving us easy conversion to the Uppaal tool and expanding the architecture possibilities, as TAs can model real-time systems with concurrency. To support the complete system verification, further specification of data acquisition, data transmission, and update management was also conducted. In addition, to emulate external interaction and constrain the acquisition task to follow a specific sampling frequency, the concepts of interrupt/event generators and timers were created in Uppaal. As a result, the actual functionality of the state machine was modeled into several separate TAs.

The main process TA is the one that effectively controls the operation flow, which comprehends all the channel synchronizations needed to start the execution of the current state. The ADC TA takes the responsibility of starting a timer and, following its synchronization signals, saving the data read from the ADC peripheral at the specified sampling rate. After a predetermined number of samples is acquired, the samples are sent and the sender TA is activated. The update TA purpose is to simulate an update on the device and, after the completion of such procedure, prompt the system reset. The reset needs to be propagated through the whole state machine and for that we use a synchronization channel of the broadcast type. The external event TA enables dealing with acquisition start and stop signals and the update requests that change the system's internal state, which are also received as external inputs. Finally, timer TA has the important role of ensuring the readings are performed at fixed time intervals, therefore guaranteeing the precision of the sampling rate. These two components were modeled into two TAs that generate events and interact with the overall state machine: the external event TA and the timer TA.

Having these definitions, the Uppaal tool can explore the TA graphs starting from different trigger nodes, i.e. starting from the various possible system states, and verify that certain system properties are kept throughout those runs. By defining heuristic rules representing such properties, it is possible to have guarantees that the system will not enter any deadlock states, or that the signal

acquisition and transmission tasks are properly canceled when starting an OTA update. Also, it is possible to identify dead branches in the system design, i.e. execution steps that cannot be reached through the system's expected behavior.

3 System Validation

Besides the formal verification, the full validation of a system requires tools capable of testing the major aspects of its implementation, as well as the communication with other end-nodes. To accomplish these goals for the CPS component of CardioWheel, an automatic validation station based on a Raspberry Pi small single-board computer was developed in the scope of the VALU3S project.

The developed validation station manages the installation of test firmware and provides a local network and a Bluetooth Low Energy master to test the communication capabilities of the CPS. Furthermore, it coordinates the injection of a synthetic ECG signal, which allows the validation of hardware properties of the analogue front-end. To enable a robust validation of timing-related properties, the station also manages the creation and installation of monitors that perform run-time verification based on formal requirements [6], a formal validation method that observes expected system properties as it runs in real-time.

Encapsulated in a compact format and using a simple and intuitive touch-based interface, this validation station provides two major advantages for the CardioWheel production. Firstly, the automation of the whole process ensures that a thought-out V&V process is planned and its application is made systematically and thoroughly. Secondly, the expertise requirements to perform V&V of such a complex system becomes concentrated on the design phase of the V&V routines, while the procedure itself can be conducted by any non-specialized operator, greatly reducing the costs associated with system validation.

4 Conclusion

This work focuses on the application of Uppaal in the design of the new software architecture of the CPS component of CardioWheel, as well as the validation of such system using a custom validation station specially developed for the CardioWheel ecosystem.

The application of formal system verification brings several benefits to the design, implementation, and deployment of CardioWheel. By specifying the system use cases and requirements and systematizing a system architecture, it was possible to employ formal tools like Uppaal to devise a software architecture, with the added benefit of studying and priming the system without any investment being made into writing code. Any faults or shortcomings detected in this phase can be fixed without having to sacrifice implementation efforts. Furthermore, maintaining an updated model of the system allows for its expansion regarding new potential features, whose implementation strategy can be tested within this model to ensure that its inclusion will not destabilize the system.

References

1. Behrmann, G., David, A., Larsen, K.G., Håkansson, J., Pettersson, P., Yi, W., Hendriks, M.: Uppaal 4.0. Los Alamitos, CA: IEEE Computer Society (2006)
2. Daw, Z., Cleaveland, R., Vetter, M.: Formal verification of software-based medical devices considering medical guidelines. International Journal of Computer Assisted Radiology and Surgery **9** (2014)
3. Kumar, P., Singh, L.K., Kumar, C.: Suitability analysis of software reliability models for its applicability on NPP systems. Quality and Reliability Engineering International **34**(8) (2018)
4. Lourenço, A., Alves, A.P., Carreiras, C., Duarte, R.P., Fred, A.: Cardiowheel: Ecg biometrics on the steering wheel. In: Michael, Bianca, Z., Ricard, G., Dino, P., Francesco, B., Jaime, C., Albert, S.M.B., May (eds.) Machine Learning and Knowledge Discovery in Databases. pp. 267–270. Springer International Publishing (2015)
5. Lourenço, A., Silveira, S., Cardoso, J.S.: Cardiowheel: Physiological driver monitoring. In: Sixth International Symposium on Naturalistic Driving Research (2017), publications/meetingAbstract/2017ALourencoISNDR.pdf
6. Nandi, G.S., Pereira, D., Proença, J., Santos, J., Rodrigues, L.A., Lourenço, A., Tovar, E.: MARS: A toolset for the safe and secure deployment of heterogeneous distributed systems. In: 1st International Workshop on Explainability of Real-time Systems and their Analysis at the IEEE Real-Time Systems Symposium (RTSS 2022) in Houston, USA (2022), https://micro.ros.org/docs/overview/hardware/
7. Raimundo, D., Lourenço, A., Abrantes, A.: Driving simulator for performance monitoring with physiological sensors. In: 2018 19th IEEE Mediterranean Electrotechnical Conference (MELECON). pp. 119–124 (2018). https://doi.org/10.1109/MELCON.2018.8379079
8. silva, H., Lourenço, A., Fred, A.: Device and method for continuous biometric recognition based on electrocardiographic signals (WO2013109154A1, May 25, 2013)
9. Tuan, L.A., Zheng, M.C., Tho, Q.T.: Modeling and verification of safety critical systems: A case study on pacemaker. In: 2010 Fourth International Conference on Secure Software Integration and Reliability Improvement (2010). https://doi.org/10.1109/SSIRI.2010.28