Documenting Assumptions about the Operational Context of Long-Living Collaborative Embedded Systems

Marian Daun, Bastian Tenbergen, Jennifer Brings, Thorsten Weyer

paluno – The Ruhr Institute for Software Technology
University of Duisburg-Essen, Germany
marian.daun, bastian.tenbergen, jennifer.brings, thorsten.weyer@paluno.uni-due.de

Abstract: Today's embedded systems operate in highly interactive collaborative system networks to fullfill an overall purpose within a complex technical system (e.g., motor vehicles, aircrafts, industrial plants). The lifespan of such complex technical systems typically covers several decades in which a modernization or replacement of individual embedded systems is accompanied by high efforts and costs. Consequently, collaborative embedded systems need to be designed to cope with changes in their operational context throughout the prospective lifespan. In this paper, we outline the advantage of explicitly documenting assumptions about the operational context of long-living collaborative embedded systems. Documenting assumptions about the operational context fosters the engineering of collaborative embedded systems insofar that these systems are able to cope with specific changes in their operational context throughout their lifespan.

1 Introduction

In the automotive or avionics industries, embedded systems typically serve very long lifespans. Aircraft are designed to be in service for 30 years or longer and must be supplied with spare parts over several decades, as is the case with automobiles. Moreover, embedded systems in these domains are typically closely integrated in their operational contexts, i.e. the external actors the systems interact with during operation (see [DTW12]): Embedded systems monitor context measurements using sensors, exchange instructions with external actors, compute necessary control commands, and exert influence onto their context by means of actuators. In many cases, these embedded systems are part of a collaborative system network in order to achieve a common goal. However, as such long-living and collaborative embedded systems age, their contexts inevitably change. For example, aircraft routinely undergo several major overhauls, in which systems are replaced with more modern equivalents, upgraded to offer additional functionality, or updated to fix deprecated or suboptimal behavior. Therefore, during development, it is necessary to account for possible changes in the operational context over the long years of operation.

In the engineering of embedded systems, the focus of development typically lies on the specification of behavioral requirements and the definition of a functional design, which defines the system's functions and specifies the interplay between functions to fulfill the behavioral requirements (cf. [DHW14]). By doing so, the development process typically does not account for possible changes in the operational context during the operational phase (e.g., a new sensor technology or the replacement of a neighboring system, which impairs the functional interplay within the collaborative system network). Therefore, the long-living nature of collaborative embedded systems makes it necessary to explicitly document assumptions about the operational context (cf. [DB+14]). These assumptions must be documented during the development phase (i.e. during requirements engineering or development of the functional design, see [DTW12]) such that changes in the operational context can be monitored and acted upon during the operational phase.

2 Explicit Documentation of Assumed Context Configurations

To assist the development of long-living collaborative embedded systems, we suggest that assumption about the operational context be documented explicitly during the development phase. By doing so, engineers make assumptions about the nature of the interaction between the system and it's operational context and can anticipate what happens when changes occur in the context during the operational phase. When explicitly documenting these assumptions alongside the regular engineering artifacts, permissable context configurations at different stages of the operational phase can be captured, which are known to retain adequate functionality. For long-living systems, explicit documentation of such context assumptions means that alternative context configurations, which specify permissable changes in the operational context, can be predicted. This enables the use of automated validation and verification techniques during the development phase during the operational phase, e.g., to check if a proposed upgrade to one system will result in safe behavior of the entire collaborative system network. Figure 1 illustrates the relation between assumed context configurations during the development phase and actual configurations during the operational phase.

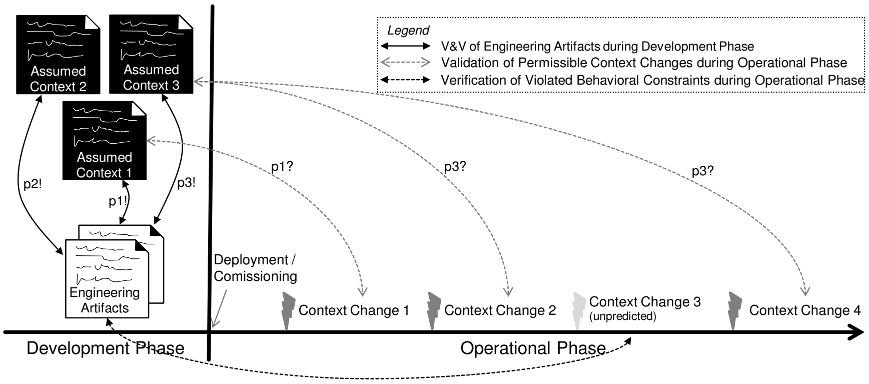
Assumed Context 2
Assumed Context 3
Assumed Context 1
Engineering Artifacts
p2!
p1!
p3!
p1?
p3?
p3?
Legend
V&V of Engineering Artifacts during Development Phase
Validation of Permissible Context Changes during Operational Phase
Verification of Violated Behavioral Constraints during Operational Phase
Deployment / Comissioning
Context Change 1
Context Change 2
Context Change 3 (unpredicted)
Context Change 4
Development Phase
Operational Phase

Figure 1: Checking assumed context configurations against actual context configurations

As can be seen in Figure 1, the assumed context configurations can be used for verification purposes against the system's engineering artifacts, as is commonly done during development. Multiple assumed context configurations can be documented, in which, through verification, it becomes certain that adequate functionality of the system is maintained. As can be further seen, these multiple assumed context configurations allow for runtime verification during the operaitonal phase of a long-living collaborative system: Changes in the operational context (e.g., a major overhaul or system upgrade) can be validated by checking the system's actual, perceived context against the assumed context configurations. In case the system's acutal context has not been assumed to be a permissable context configuration, runtime verification of the system's behavior with regard to the unforeseen context change must be conducted. This can, for example, be done during system maintenance.

3 Towards an Integrated Methodology

Explicit documentation of context information has also been considered a prerequisite for various quality assurance and analysis approaches such as model checking of static properties of engineering artifacts (e.g., [DP+09]) as well as checking of behavioral properties (e.g., [AH01]). Ontology-based approaches have also been proposed for context documentation in the past (e.g., [SLF03]), yet they focus on non-collaborative systems. In prior work, we investigated documenting engineering artifacts with regard to context interactions [DTW12]. Furthermore, we proposed an ontology-centric approach to document and analyse knowledge sources, which impact the engineering process [DB+14]. Currently, our work is focused on an ontology-centric approach, which allows documenting static-structural, functional, and behavioral context properties of collaborative embedded systems in accordance with [ISO11].

Acknowledgements. This research was funded by the *German Federal Ministry of Education and Research* (grant no. 01IS12005C).

References

[AH01] Alfaro, L. de; Henzinger, T.: Interface automata. In: Proc. ESEC/FSE, 2001; 109–120.

[DB+14] Daun, M.; Brings, J.; Tenbergen, B; Weyer, T.: On the Model-based Documentation of Knowledge Sources in the Engineering of Embedded Systems. In: Proc. ENVISION 2020, 2014; 67-76.

[DHW14]Daun, M.; Höfflinger, J.; Weyer, T.: Function-Centered Engineering of Embedded Systems – Evaluating Industry Needs and Possible Solutions. In: Proc. ENASE, 2014; 226-234.

[DP+09] Dhaussy, P.; Pillain, P.; Creff, S.; Raji, A.; Traon, Y.; Baudry, B.: Evaluating Context Descriptions and Property Definition Patterns for Software Formal Validation. In: Model Driven Engineering Languages and Systems, Springer, 2009; 438–452.

[DTW12]Daun, M.; Tenbergen, B.; Weyer, T.: Requirements Viewpoint. In: Model-Based Engineering of Embedded Systems, Springer, 2012; 51-68.

[ISO11] ISO 42010: Systems and software engineering - Architecture description, 2011

[SLF03] Strang, T.; Linnhoff-Popien, C.; Frank, K.: CoOL: A Context Ontology Language to Enable Contextual Interoperability. In: Distributed Applications and Interoperable Systems, Springer, 2003; 236–247.