Stream Runtime Verification of Changing Requirements

Motivation and Background

Different verification techniques are used throughout the engineering process of systems. For many systems, this process of verification must not end with the system's deployment, but needs to continue at runtime. One technique to provide formal assurance at runtime is stream runtime verification (SRV) [1]. In SRV, a stream of system events (e.g., from a log or from message exchange in a distributed system) is analysed w.r.t. a requirement expressed in a formal language such as Linear Temporal Logic (LTL). This formal requirement(a so called property) can automatically be transformed into an automaton [2] to check if a system execution conforms to the requirement, i.e., solve the word problem (is the execution (word) successful or failing (going to good states or bad states)). In some systems, these requirements are not static throughout the system's runtime, but might be changed, e.g., by the stakeholders. In this case, the verification should adapt instead of being restarted to prevent information loss. In previous work [3], we have introduced a notion of property adaptation patterns (PAPs) that describe a handful of possible changes in the requirement.

Goals

The goal of this thesis is to extend this preliminary catalogue of PAPs by finding relevant changes of requirements in the literature and formalise them to changes in LTL. Additionally, one practical aspect would be to encode the new PAPs into our existing implementation that can be used to evaluate the thesis.

Description of the Task

The specific tasks are:

- Find changes of requirements in the literature
- Formalise them in LTL
- And encode them in the existing C++ framework

Research Type

Theoretical Aspects:	****✱
Industrial Relevance:	****✱
Implementation	**✱✱✱

Prerequisite

The student should be enrolled in the master of computer science program, and has completed the required course modules to start a master thesis.

Skills required

Programming skills in C++, understanding of, or willingness to learn, formal methods such as needed for the project. The student should be able to read and comprehend research articles in English.

Contacts

Marc Carwehl (m.carwehl@hu-berlin.de)
Software Engineering Group, Institut für Informatik, Humboldt-Universität zu Berlin

References

[1] Falcone, Yliès, Klaus Havelund, and Giles Reger. "A tutorial on runtime verification." *Engineering dependable software systems*(2013): 141-175.

[2] Havelund, Klaus, and Grigore Roşu. "Synthesizing monitors for safety properties." *Tools and Algorithms for the Construction and Analysis of Systems: 8th International Conference, TACAS 2002 Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2002 Grenoble, France, April 8–12, 2002 Proceedings 8*. Springer Berlin Heidelberg, 2002.
[3] Carwehl, Marc, et al. "Runtime verification of self-adaptive systems with changing requirements." *2023 IEEE/ACM 18th Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2023