Safety Analysis and Requirements Allocation for Software Product Lines

André L. de Oliveira[1]

[1]Dept. of Computer Science, Universidade Federal de Juiz de Fora, Brasil
andre.oliveira@ice.ufjf.br

Abstract. A Software Product Line (SPL) is a set of software intensive systems that share a set of capabilities (features) that satisfy the specific needs of a particular application domain. Variant-intensive safety-critical systems, developed as part of a software product line, must still comply with safety standards. Standards use the concept of Safety Integrity Levels (SILs) to drive the assignment of system safety requirements to components of a system under design. However, for a SPL, the safety requirements that need to be allocated to a component may vary in different products. Variation in the design can indeed change the possible hazards incurred in each product, their causes, and can alter the safety requirements placed on individual components in different products. Establishing common SILs for components of a large-scale SPL by considering all possible products and their usage context (scenarios), is desirable for economies of scale, but it also poses challenges to safety engineering. In this tutorial, we present an overview of DEPendable Software Product Line Engineering (SPLE) approach, which extends traditional SPL process to support Safety Analysis, allocation and decomposition of safety requirements, in the form of SILs, to variant-intensive components. Performing such kind of allocation enables the early identification of safety requirements to ensure the safe use of components across the SPL. We describe the DEPendable-SPLE safety analysis and allocation of safety requirements approach steps in the automotive domain.

Keywords: Safety-Critical Software Product Line, Safety Requirements, Safety Integrity Levels, Requirements Allocation and Decomposition.

1 Introduction

Safety-critical software is becoming more complex due to the greater possibilities offered by inter-connectivity and the increased computing power. The demand for building ecosystems that integrate software from multiple vendors while still fulfilling the safety goals of an application domain is rising [12]. In the development of the latest automotive systems, newer functionalities are generated by integrating independently OEM (Original Equipment Manufacturer). The mass customization in the automotive industry leads to a higher variation within a single product with thousands of variations points [7]. In the automotive domain, electronic control units (ECUs) [9]

and power-train controllers [11] are highly variant-intensive. Automotive ECUs are used in airbag, electronic window lifter, and driver assistant systems.

A Software Product Line (SPL) approach allows developing a set of products that have common features and variable parts, meeting the demands of highly cost intensive market segments such as automotive and aerospace sectors [7]. SPL approaches have been successfully adopted to manage the complexity of increasingly number of variants and reducing the development costs of automotive systems. Product line design maximizes software reuse across products but must still yield safe individual product configurations (variants) and this poses research challenges to safety engineering. Companies that adopt an SPL approach to build their products should analyze/demonstrate safety properties across multiple products to identify potential threats that may lead the system to unsafe states, classify their risks, and assign safety requirements to mitigate failure effects.

Standards use the concept of Safety Integrity Levels (SILs) to assign safety requirements of different stringencies to components of a system under design. They take the form of Automotive Safety Integrity Levels (ASILs) in the ISO 26262 [3] standard for passenger cars. The ASILs range from ASIL A, the least stringent, to ASIL D, the most stringent, with parts that don't have safety requirements assigned to normal quality management (QM). Different ASILs may impose different certification requirements, i.e., in terms of safety objectives to be addressed to achieve safety approval. ASILs are assigned early in the system design process at system level, after the identification of system hazards. These hazards are given more or less stringent ASILs depending on how high are the risks they pose. As the system architecture is being refined, ASILs assigned to hazards are iteratively allocated to subsystems and components. ASIL decomposition allows efficient allocation of safety requirements to achieve compliance with standards without being unnecessarily expensive.

Model-Based Safety Assessment (MSBA) techniques can potentially provide frameworks for safety analysis and SILs allocation, by allowing the automatic identification of combinations of component failures that lead to hazards, and therefore by locating opportunities for ASIL decomposition. HiP-HOPS (Hierarchically Performed Hazard Origin & Propagation Studies) [7] is an advanced MBSA technique that already provides such an approach for ISO 26262. HiP-HOPS implements a combination of model-based, automated Fault Tree Analysis (FTA) process and a Tabu Search (TS) [1] meta-heuristic optimization algorithm, allowing optimal ASIL allocation and has shown to scale up to complex systems.

Although application of the MBSA process to software product line design would be beneficial, this is not straightforward. Extensions of SPL approaches to critical systems must consider safety engineering and certification issues [4, 5, 9]. In a safety-critical software product line, variation in the design choices and usage context may impact hazard analysis, assignment of safety integrity levels (SILs) to hazards and their decomposition throughout software components [2, 4, 5, 6]. Thereby, establishing safety requirements for SPL components requires finding the ASILs that should be assigned to those components to ensure their safe use across the SPL. How can we transfer the top-down thinking about safety to a Software Product Line design? In this tutorial, we provide an answer to this question by presenting the novel DEPendable-

SPLE [4] approach to support model-based safety analysis and SIL Allocation for product lines and an extension to HiP-HOPS design optimization capability [6].

This tutorial is organized as follows. We firstly provide an overview of Software Product Line Engineering, feature modeling, and variant management. We further describe the automotive ISO 26262 standard and ASIL decomposition. We introduce the DEPendable-SPLE approach, the concept of Product Line SIL Allocation and Decomposition and HiP-HOPS optimization extension. Finally, we illustrate the application of product line safety analysis and requirements allocation approach and tooling in the automotive domain.

References

1. Azevedo, L. S., Parker, D., Walker, M. Papadopoulos, Y. Automatic Decomposition of Safety Integrity Levels : Optimization by Tabu Search. In: 2nd Workshop on Critical Automotive applications : Robustness & Safety (CARS), Safecomp (2013).
2. Habli, I., Kelly, T. P., Hopkins, I. Challenges of Establishing a Software Product Line for an Aerospace Engine Monitoring System. In: 11th International Software Product Line Conference (SPLC), pp. 193–202, ACM, Japan (2007).
3. ISO. ISO 26262: Road vehicles - Functional safety, (2011).
4. Oliveira, A. L., Braga, R. T. V., Masiero, P. C., Parker, D., Papadopoulos, Y., Habli, I., Kelly, T.. Variability management in safety-critical systems design and dependability analysis. Journal of Software: Evolution and Process, v. 31 n. (8), Wiley, (2019).
5. Oliveira, A. L., R. T. V., Masiero, P. C., Papadopoulos, Habli, I., Kelly, T Variability Management in Safety-Critical Software Product Line Engineering. In: R. Capilla, C. Cetina, and B. Gallina (Eds.), ICSR 2018, LNCS 10826, pp. 1–20 (2018).
6. Oliveira, A. L., Braga, R. T. V., Masiero, P. C., Papadopoulos, Y., Azevedo, L., Parker, D., Habli, I., Kelly, T. Automatic allocation of safety requirements to components of a software product line. In: 9th IFAC Symp. on Fault Detection, Supervision and Safety for Technical Processes, Paris, France, Elsevier, v. 48, i. 41, pp. 1309-1314 (2015).
7. Papadopoulos Y., Walker M., Parker D., Rüde, E., Hamann, R., Uhlig, A., Grätz, U., Lien, R. Engineering failure analysis and design optimization with HiP-HOPS. Journal of Engineering Failure Analysis, Elsevier. 18 (2), 590-608, (2011).
8. Pohl, P., Höchsmann, M., Wohlgemuth, P., Tischer, C. Variant management solution for large scale software product lines. In: 40th International Conference on Software Engineering: Software Engineering in Practice, pp. 85-94, ACM, Gothenburg, Sweden (2018).
9. Schulze, M., Mauersberger, J., Beuche, D. Functional safety and variability: can it be brought together? In: 17th Int. Software Product Line Conf., pp. 236-243; ACM (2013).
10. SPLC.net, SPLC hall of the fame: General Motors Powertrain (GMPW), https://splc.net/fame/general-motors-powertrain, last accessed 2020/03/03.
11. Tischer, C., Muller, A., Mandl, T., Krause, R. Experiences from a Large Scale Software Product Line Merger in the Automotive Domain. In: 15th International Software Product Line Conference, pp. 267-276, ACM, Munich, Germany (2011).
12. Wolschke, C., Becker, M., Schneickert, S., Adler, R., and MacGregor, J. Industrial Perspective on Reuse of Safety Artifacts in Software Product Lines. In: 23rd International Systems and Software Product Line Conference, Sept. 9–13, ACM, Paris, France (2019).