Computer and Information Science; Vol. 6, No. 4; 2013
ISSN 1913-8989 E-ISSN 1913-8997
Published by Canadian Center of Science and Education

Handling Safety-Related Requirements in Critical Systems Product Lines

Ahmad Khader Haboush[1]

[1] Computer Science and Network Department, Jerash University, Jordan

Correspondence: Ahmad Khader Haboush, Computer Science and Network Department, Jerash University, Jordan. E-mail: ahmad_ram2001@yahoo.com

Received: August 18, 2013 Accepted: September 11, 2013 Online Published: September 29, 2013

doi:10.5539/cis.v6n4p132 URL: http://dx.doi.org/10.5539/cis.v6n4p132

Abstract

Companies currently use product lines to enhance products qualities and to lower the development and marketing costs. Many approaches available to support product lines, but most of them fail to address safety critical requirements in their approaches. In this paper we present DOPLER^{+}, an approach to handle safety critical system requirements and the addition of new safety critical requirements, and their influences on other features and requirement within the product line. To illustrate our approach, we apply DOPLER^{+} to mobile robot case study.

Keywords: product line, SFTA, critical systems, requirement engineering, FMEA

1. Introduction

The main goal of software Product line (SPL) is to decrease the efforts by increasing the degree of reuse in software engineering. Software product lines is defined as "a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way" (Bergey et al., 2005). Product line engineering requires two stages; domain engineering which defines the commonalities and variability features of the products, and application engineering which defines the features uniquely defines each product (Liu et al., 2007). One of the major research issues in PLE is how to handle safety critical features in product lines; and how to add new safety critical features to the product line (Heider et al., 2012).

In this paper we present DOPLER^{+}, an expansion of DOPLER (Decision-Oriented Product Line Engineering for Effective Reuse) approach (Dhungana et al., 2011) and DOPLER^{Ucon} (Rabiser & Dhungana, 2007) to deal with the effects of adding a safety critical safety requirement to a PLE. Current version of DOPLER doesn't handle safety requirements and rely on the engineers to do it manually. We use the work of (Dehlinger & Lutz, 2004), SFTA analysis for product lines to modify the current version DOPLER^{Ucon}.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 gives an overview of DOPLER^{Ucon}. Section 4 describes the modifications that lead to DOPLER^{+}. Section 5 discusses mobile robot product line- a case study. Section 6 offers some concluding remarks

2. Related Work

Researchers provide us with many models and approaches to handle product lines to make most benefits of the reuse of components, features, codes … etc. but most of them don't address safety in critical systems in their approaches, so many of them failed to handle safety-critical features. To handle Safety-related Feature Interaction in Safety-Critical Product Lines, Liu (2007) propose a four steps solution where she combines the traditional feature modeling with the safety analysis in safety-critical product lines to handle safety-related feature interactions. Dehlinger and Lutz (2005) shows a product line requirements Approach to Safe Reuse in Multi-Agent Systems, they extend the role of the multi-agent systems used in distributed systems to handle safe reuse. Liu et al. (2007) discuss the safety analysis of product lines by using state-based modeling, they had a scenario-guided executions of state model over variant to explore the relationships between behavioral variations and potential hazardous states. Braga et al. (2012) try to adapt software product line engineering process for certifying safety critical embedded systems. They establish an infrastructure of product line engineering for certified critical safety products. Faulk (2001) presents a systematic approach to developing a Product-line

Requirements Specification (PRS). The PRS explicitly represents the family's common requirements as well as the allowed variations that distinguish family members. When completed, the PRS definition also supports generation of well-formed Sofware Requirements Specifcations (SRS) for members of the product line. Braga et al. (2012) presents ProLiCES, an approach for the development of safety-critical embedded applications and its usage to develop a product line for unmanned aerial vehicles. They shows a two-path parallel life cycle, where Domain Engineering and Application Engineering can beconducted simultaneously or separately, depending on the current scenario faced by the software enterprise. In Saratxaga et al. (2012) a product line tool chain is presented based on the analysis of current SPL tools and approaches in order to fit the specific needs within industry partners in the CESAR project. There main goal was to show the benefits of a combination of SPL tools in an industrial scenario.

3. DOPLER^{UCON}

Before we talk about the activities of DOPLER^{Ucon} and how it handles the addition of new requirement to an existing product in a product line we need to show how requirements are modeled in DOPLER. All the requirements commonalities and variability's and their structural and functional dependencies are stored in core assets. Stockholders and engineers decide which of these requirements that satisfies inclusion conditions would be included in the product. The selected requirements are called decisions. Decisions are related with each other's as one decision can lead to and affect another decision, so these decisions and their effects are especially documented in the variability model. Each decision has hierarchal and logical dependencies; hierarchal dependencies state the conditions that should be satisfied for the decision to be taken, while logical dependencies are the known consequences of taking this decision to other decisions (Dhungana et al., 2007).

3.1 DOPLER^{Ucon} Activities

(1) Variability model adaptation and sales knowledge acquisition. It starts by capturing requirements from assets by engineers and stockholders. Product/sales manager drops irrelevant requirements. The output of this activity is called derivation model (Activity 1 in Figure 1).

(2) Product configuration and simulation. Taking the derivation model as input, this activity maps the product derivation decision and variability model to allow automatic selection of required assets. The selected assets can be used to simulate the product (Activity 2 in Figure 1).

(3) Requirement elicitation and negotiations. There are two types of negotiations; sales to customer, where customers provide their wishes and features to sales persons. And sales to engineer, where engineer's checks to decide which of these wishes should be considered as new requirement and which of them are provided by the current product line assets (Activity 3 in Figure 1).

(4) Product development. Where requirement provided by the previous activity are available to the engineers to start planning and developing the new features (Activity 4 in Figure 1).

(5) Product deployment. There are different deployment mechanisms used by companies, the use of the existing assets documentation can reduce the effort for the manual product deployment. However, there's a chance that the development of new feature is not suitable. So, there's a feedback loop to the simulation activity in such situation (Activity 5 in Figure 1).

(6) Product line evolution. Finally, the newly developed assets are analyzed by engineers to check their suitability to be included to the product line. This is done by the feedback loop to the model (Activity 6 in Figure 1).

As a new requirement captured, particular interest is: the origin and the influence. The origin is the list of decisions that lead to particular requirement, and the influence; which is the list of decisions that likely to be affected (Dhungana et al., 2007).

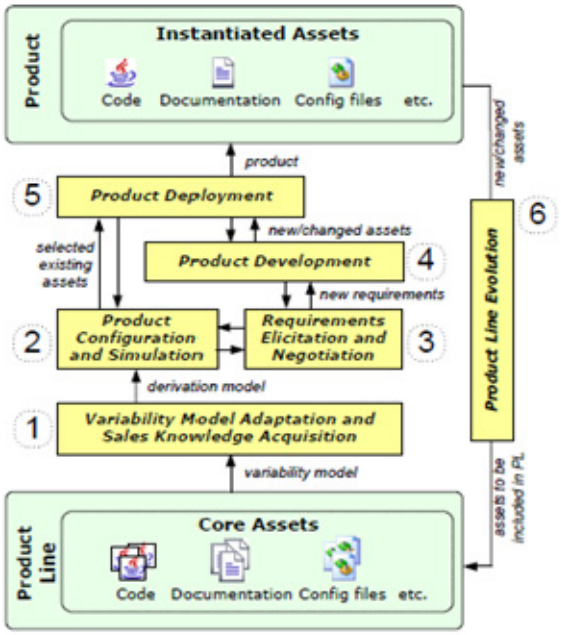

Figure 1. Requirement engineering activities in DOPLER^{Ucon} (Dhungana et al., 2007)

The problem of DOPLER^{Ucon} is that it doesn't handle requirements for critical safety product line. For example, considering robot product line, the visibility sensor has two options; sonar sensor - noisy and not accurate- and laser finder –high accurate and small distance coverage- and robot speed –legs movement (slow) or wheels (high) - . If the stockholder chooses sonar for visibility and wheels for movement, then the chance of avoiding obstacles during its mission is very low. Using current DOPLER^{Ucon}, it will fail to address that there is a need for recovery feature to be provided, unless it's added manually by the engineers.

4. DOPLER^{+}

Figure 2 shows the additional activities needed to handle safety-critical requirements in DOPLER product line. It is mainly the same as DOPLER^{Ucon}, before creating the product line core assets a preliminary hazard investigation need to be established. The main results of this investigation are Hazard analysis and Failure Mode and Effect analysis (FMEA). Using FMEA and the product line assets (code, features, configuration … etc.), we can create SFTA for the current product line using the algorithm provided by (Dehlinger & Lutz, 2004).

To handle adding new requirements that is not presented in the product line or dropping other requirements, we add a new step called trim product line SFTA. Mainly, large number of modifications came from the negotiations between customers and sales persons, and sales persons and engineers. So after the end of this activity, most of the requirements needed for the new product are available, resulting in more accurate SFTA. Another number of modifications can be founded during the product development activity, if founded, the new requirements should be handled in the requirements elicitation and negotiations and the product SFTA should be modified. We use the process of reusing product line SFTA presented by (Dehlinger & Lutz, 2004), the process is explained in depth in the next section.

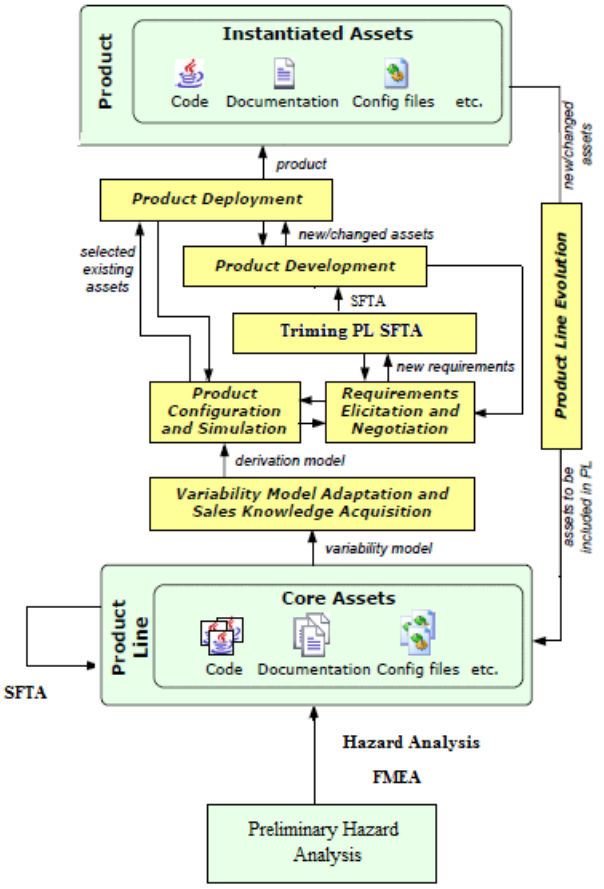

Figure 2. Requirement engineering activities in DOPLER^{+}

5. Mobile Robot Product Line - A Case Study

Mobile robots have many basic specifications; it can move forward and backward, turn right or lift. Robots also may vary in their behavior such as scouting area, constructing a map, collecting object with a specific color and many more. Based on (Thompson & Heimdahl, 2001), the mobile robot product line can be divided into: hardware dimensions and behavior dimensions.

Hardware dimensions specify the commonalities and variability of hardware features. These dimensions might include basic platform; a commonality feature is that robots have a means of locomotion, move forward and backward, turn number of degrees to the right or left. A variability features for it like: the mean of the platform may vary; it might be treads, legs, wheels. Moreover, the maximum speed may also vary.

Other hardware features available for robots, for example, for detecting obstacles the robot should be provided by a range finder. There are different types for range finder; sonar sensor for example covers a wide area with less accuracy. And laser sensor, which has high accuracy and small coverage area. There are many other hardware features available.

For the behavioral dimension which specifies what the robot will do, we will present two behaviors: random exploration and door investigation. We will refer to the commonality features by the letter C, and the variability features with the letter V.

For Random Exploration,

C1. The robot attempts to avoid collision.

V1. An object may not be considered as an obstacle based on the robot's mode of operation.

C2. If the robot collides, it should have a procedure to recover from collision.

V2. The number of collisions that should happen before a robot shut down.

C3. If there are no obstacles, the robot should have other behavior as a subfamily.

V3. If there are no obstacles, the robot should move at maximum speed.

For door navigation,

C4. The robot should be able to locate a door.

C4.a. Once the robot found an object that believes it's a door, the robot treats the door sides as obstacles as in C1 and V1.

V4. The width of the door may vary depends on the width of the robot.

In DOPLER, the requirements and features are presented as questions to be answered by the stockholder and engineers. For example, one feature is what is the locomotion means used for this robot? Or does the robot have a camera attached to it.

Now to show how DOPLER^{+} handles safety requirements, a preliminary hazard analysis activity is executed, the output of this activity is a Failure Mode Effect and Analysis (FMEA) for current product line is provided in Table 1.

Table 1. Part of FMEA for robot product line

Item	Failure Mode	Cause of Failure	Possible Effects
Range Finder - rf	Collision	Inaccurate calculation	Shut down
Door investigation sensor- dis	Collision	wrong data or Inaccurate calculation	Shut down
	Avoiding a door	Wrong calculation	N/A
Locomotion sensor - ls	Collision	Missing/Inaccurate data	Shut down

The FMEA shows that an inaccurate calculation of the range finder sensor can make the robot collide, and cause it to shut down. Also, door investigation sensor may provide wrong data, telling that the obstacle is a door resulting in a collision. Furthermore, even though the door investigation sensor may successfully locate a door, it might fail to calculate if the door width and height is suitable for the robot which result a collision. The sensor may also provide a wrong data treated a door as an obstacle, but this failure doesn't have any possible effect. But, missing or providing inaccurate data by the locomotion sensor may result a collision, as the robot will not be able to locate obstacles. This collision may result the robot to shut down.

By using the FMEA table and the product line core assets, now we can create SFTA using SFTA create algorithm provided by (Dehlinger & Lutz, 2004). They state that the root node of the SFTA is mainly provided by a pre-exist hazard list, and the causal conditions can be derived from the FMEA's cause of failure column. To determine intermediate nodes, they use algorithm that searches the effect column for an event, if found, they take the cause of failure for that event as a new event and run the algorithm in a recursive manner. The problem in such algorithm is that it needs refinement from a domain expertise. Another problem is that FMEA allow only the use of OR logical gates, and in some rare cases the use of AND gates as well. The final step in creating SFTA for the product line is to consider the effect of variability of the product line on the SFTA. Figure 3 shows the final SFTA.

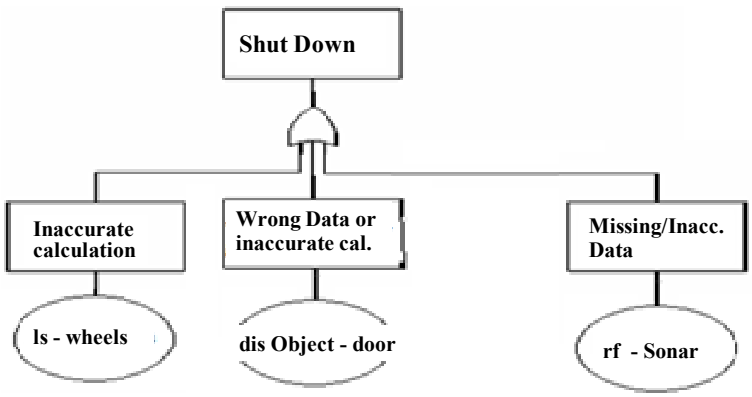

Figure 3. Initial SFTA for the current product line

If a new product is needed, stockholders and engineers will choose the features to be added to the new product. Suppose we want to limit the robot behavior by scout area only. This called pruning the product line SFTA. PLSFTA_search algorithm is called to mark the sub-trees that don't have any impact on the new product. The marked sub-trees are removed and the result SFTA represents the SFTA for the new product. Figure 4 shows the SFTA for the new product. If the new product requires new requirements that are not available by the product line, hazard analysis can be performed and new rows in the FMEA table may appear. So we run SFTA create one more time to have a required SFTA.

If a new product is needed, stockholders and engineers will choose the features to be added to the new product. Suppose we want to limit the robot behavior by scout area only. This called pruning the product line SFTA. PLSFTA_search algorithm is called to mark the sub-trees that don't have any impact on the new product. The marked sub-trees are removed and the result SFTA represents the SFTA for the new product. Figure 4 shows the SFTA for the new product. If the new product requires new requirements that are not available by the product line, hazard analysis can be performed and new rows in the FMEA table may appear. So we run SFTA create one more time to have a required SFTA.

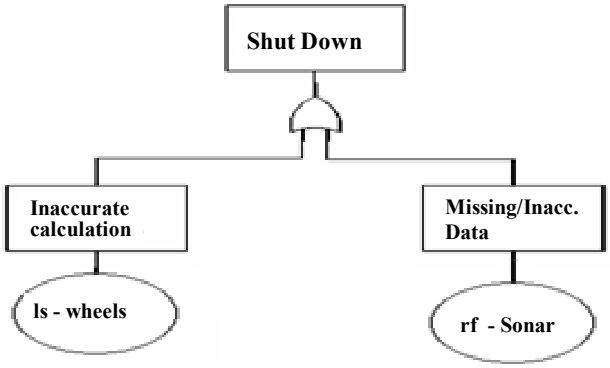

Figure 4. SFTA for the Scout robot product

6. Conclusion

Product line engineering become widely spread due to its advantage of reuse and reduce the time and costs for deriving new products. Dealing with product line features is more complex than traditional features since a change in product line feature means a change in number of products at the same time. In this paper, we illustrate DOPLER$^+$ to handle safety critical requirements. DOPLER$^+$ depends on preliminary hazard analysis and FMEA to create SFTA for the product line. When new product derived, DOPLER$^+$ reuse the product line SFTA to handle the new product. For our future work, we will build a fully automated tool to support DOPLER$^+$.

References

Bergey, J. K., Cohen, S., Donohoe, P., & Jones, L. G. (2005). *Software Product Lines: Experiences from the Eighth DoD Software Product Line Workshop* (p. 1). Carnegie Mellon Software Engineering Institute.

Braga, R. T. V., Junior, O. T., Branco, K. R. C., Neris, L. D. O., & Lee, J. (2012). Adapting a software product line engineering process for certifying safety critical embedded systems. In *Computer Safety, Reliability, and Security* (pp. 352-363). Springer Berlin Heidelberg. http://dx.doi.org/10.1007/978-3-642-33678-2_30

Dehlinger, J., & Lutz, R. (2004). Software Fault Tree Analysis for Product Lines. *Proceedings of the Eighth IEEE International Symposium on High Assurance Systems Engineering* (HASE'04) (pp. 12-21). http://dx.doi.org/10.1109/HASE.2004.1281726

Dehlinger, J., & Lutz, R. (2005). Requirements Approach to Safe Reuse in Multi-Agent Systems. *Proceedings of the fourth international workshop on Software engineering for large-scale multi-agent systems* (SELMAS '05) (pp. 1-7). http://dx.doi.org/10.1145/1082960.1082981

Dhungana, D., Grünbacher, P., & Rabiser, R. (2011). The DOPLER meta-tool for decision-oriented variability modeling: a multiple case study. *Automated Software Engineering, 18*(1), 77-114. http://dx.doi.org/10.1007/s10515-010-0076-6

Dhungana, D., Rabiser, R., & Grunbacher, P. (2007, January). Decision-oriented modeling of product line architectures. In *Software Architecture, 2007. WICSA'07. The Working IEEE/IFIP Conference on* (pp. 22-22). IEEE.

Faulk, S. R. (2001). Product-line requirements specification (PRS): An approach and case study. In *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on* (pp. 48-55). IEEE.

Heider, W., Grünbacher, P., Rabiser, R., & Lehofer, M. (2012). *Software and Systems Traceability* (pp. 195-213). Springer. http://dx.doi.org/10.1007/978-1-4471-2239-5_9

Liu, J. (2007, May). Handling safety-related feature interaction in safety-critical product lines. In *Software Engineering-Companion, 2007. ICSE 2007 Companion. 29th International Conference on* (pp. 85-86). IEEE.

Liu, J., Dehlinger, J., & Lutz, R. (2007). Safety analysis of software product lines using state-based modeling. *The Journal of Systems and Software, 80*, 1879-1892. http://dx.doi.org/10.1016/j.jss.2007.01.047

Rabiser, R., & Dhungana, D. (2007). Integrated Support for Product Configuration and Requirements Engineering in Product Derivation. *Proceedings of the 33rd EUROMICRO Conference on Software Engineering and Advanced Applications* (EUROMICRO '07) (pp. 219-228). http://dx.doi.org/10.1109/EUROMICRO.2007.36

Saratxaga, C. L., Alonso-Montes, C., Haugerr, O., Ekelirr, C., & Mitschke, A. (2012, June). Product line tool-chain: variability in critical systems. In *Product Line Approaches in Software Engineering (PLEASE), 2012 3rd International Workshop on* (pp. 57-60). IEEE.

Thompson, J. M., & Heimdahl, M. P. E. (2001). Extending the product family approach to support n-dimensional and hierarchical product lines. In *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on* (pp. 56-64). IEEE.

Vaccare Braga, R. T., Branco, C., Kalinka, R. L., Trindade Júnior, O., Masiero, P. C., Neris, L. O., & Becker, M. (2012). The prolices approach to develop product lines for safety-critical embedded systems and its application to the unmanned aerial vehicles domain. *CLEI Electronic Journal, 15*(2), 1-13.