Available online at www.sciencedirect.com
SCIENCE DIRECT®
The Journal of Systems and Software 78 (2005) 111–127
The Journal of Systems and Software
www.elsevier.com/locate/jss

# Bi-directional safety analysis of product lines

Qian Feng [a], Robyn R. Lutz [b,*]

[a] Department of Computer Science, Iowa State University, USA
[b] Department of Computer Science, Iowa State University and Jet Propulsion Laboratory, Caltech, USA

Received 15 July 2004; received in revised form 9 February 2005; accepted 10 February 2005
Available online 3 May 2005

**Abstract**

As product-line engineering becomes more widespread, more safety-critical software product lines are being built. This paper describes a structured method for performing safety analysis on a software product line, building on standard product-line assets: product-line requirements, architecture, and scenarios. The safety-analysis method is bi-directional in that it combines a forward analysis (from failure modes to effects) with a backward analysis (from hazards to contributing causes). Safety-analysis results are converted to XML files to allow automated consistency checking between the forward and backward analysis results and to support reuse of the safety-analysis results throughout the product line. The paper demonstrates and evaluates the method on a safety-critical product-line subsystem, the Door Control System. Results show that the bi-directional safety-analysis method found both missing and incorrect software safety requirements. Some of the new safety requirements affected all the systems in the product line while others affected only some of the systems in the product line. The results demonstrate that the proposed method can handle the challenges to safety analysis posed by variations within a product line.


*Keywords:* Product lines; Software safety; Software architecture; XML; Reuse

## 1. Introduction

As product-line engineering becomes more common, more safety-critical product lines are being built. A product line is a set of systems developed from a common set of core requirements and sharing a suite of common traits among the members (Ardis and Weiss, 1997; Weiss and Lai, 1999). Examples of safety-critical product lines include embedded medical devices such as pacemakers, space telescopes, power-plant control systems, and some industrial robots.

The potential for reuse among the systems in a software product line extends beyond code reuse. Reuse of software assets currently includes product-line requirements specifications, product-line core architecture, product-line test suites and product-line performance analyses.

This paper describes results from an investigation into how, and to what extent, product-line safety analyses can be performed and reused as a product-line asset. That is, we are interested in the potential for reuse of the safety analysis among the members of a safety-critical product line. The motivation for this research is to improve the safety-analysis techniques available to developers of commercial, safety-critical product lines.

It is important to note that safety is a property of a single system, not of a set of systems. Thus, any safety analysis done during the early domain engineering of the product line (i.e., when the entire product line is being defined) must be re-evaluated, adjusted, and completed during application engineering (i.e., when each individual system is built). Some preliminary results regarding the reuse of safety analyses during application engineering have appeared in (Dehlinger and Lutz, to

* Corresponding author. Tel.: +1 5152943654; fax: +1 5152940258.
*E-mail addresses:* qianfeng@cs.iastate.edu (Q. Feng), rlutz@cs.iastate.edu (R.R. Lutz).

0164-1212/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2005.02.028

appear; Lu and Lutz, 2002). In this paper we focus instead on the process of developing a product-line safety analysis for the domain engineering phase of safety-critical software product lines.

The paper extends the Bi-Directional Safety Analysis (BDSA) method (Lutz and Woodhouse, 1997) to product lines. The BDSA method combines a forward search from potential failure modes to their effects with a backward search from feasible hazards to the contributing causes of each hazard. The forward search is similar to a Software Failure Modes and Effects Analysis (SFMEA); the backward search is similar to a Software Fault Tree Analysis (SFTA). The combination of the forward and backward search has proven effective in discovering latent safety requirements.

The work described in this paper investigates two major challenges to extending the BDSA method to product lines: how to adequately understand and specify the safety consequences of the variations among the members of the product line, and how to structure the process such that the safety analysis is derived from, and traceable to, the product-line requirements and design.

In order to address these challenges in a way that is likely to be used by industry, the safety-analysis method presented in this paper is grounded in the standard artifacts of the product-line domain-engineering process. These domain-engineering assets are: (1) the Commonality and Variability Analysis that specifies both the requirements shared by all the systems in the product line and the variations among the systems' requirements; (2) the product-line architecture that forms the shared, core software architecture for all the systems and supports the required variations; and (3) the product-line use cases and scenarios that specify the range of uses and the sequences of events that some or all of the systems in the product line may experience.

Grounding the safety analysis in the domain-engineering products has several benefits. First, it supports documented traceability from the extended commonality analysis to the safety analysis and is requisite for future automated updating of the safety analysis as the product line evolves. Second, linking the safety analysis to the products that capture the subtleties of the domain provides more complete handling of variations, the rationales for the variations and the consequences of the variations in the safety analysis. Third, using standard domain-engineering assets promotes readier adoption of the safety-analysis method by companies building safety-critical, software product lines and can lower the cost of performing enhanced safety analyses on these product lines. The first two benefits are demonstrated in the paper by application of the safety-analysis method to the Door Control System, a safety-critical subsystem of the Smart Home product line.

Fig. 1 shows an overview of the analysis method developed in this paper with the Extended Commonality Analysis driving the bi-directional Safety Analysis in the lower half of the figure. Our method consists of seven steps:

*Step* 1: Performed Commonality and Variability Analysis to specify the requirements for the given product line.
*Step* 2: Developed the architecture design and sequence diagrams from the product-line requirements.
*Step* 3: Extended the Commonality and Variability Analysis based on the results of the Commonality and Variability Analysis and the Architecture Design diagrams.

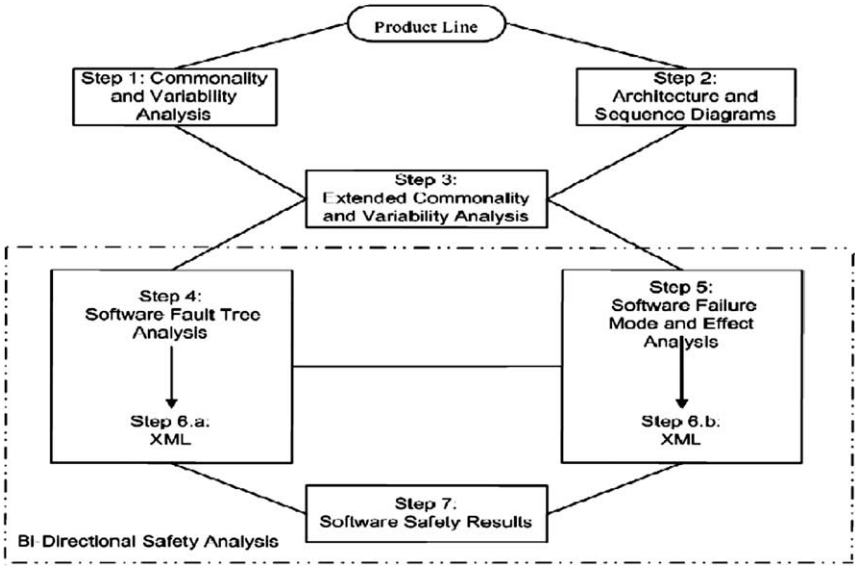


Fig. 1. An overview of the safety-analysis method.