An Experimental Comparison of System Diagrams and Textual Use Cases for the Identification of Safety Hazards

Tor Stålhane, Norwegian University of Science and Technology, Trondheim, Norway
Guttorm Sindre, Norwegian University of Science and Technology, Trondheim, Norway

ABSTRACT

Requirement defects are more costly to correct the later in the development process they are discovered. The same applies to safety requirements, and defects that remain in the fielded system are then not only costly, but potentially life-threatening. It is important to discover safety hazards as early in the process as possible, and it is thus interesting to integrate safety analysis with techniques used in the early stage of requirements engineering. This paper describes an experiment comparing how well two system diagrams and textual use cases support non-experts in identifying hazards in a simple control system. Results show that system diagrams were better for finding hazards related to peripheral equipment, while for all other kinds of hazards textual use cases were as good or better. Not only the type of representation matters, but also how the information is brought into focus for the analyst, as this might steer the analyst towards noting some hazards but ignoring others.

Keywords: Hazard Identification, Requirements Engineering, Safety, Systems Development, Use Case

INTRODUCTION

Software has been used in safety-critical systems for a long time, but a number of trends, such as massive system integration within and across enterprises, increasing computerization of consumer goods, and the internet of things, mean that number of IT systems that can have safety implications is rapidly increasing. In projects where software needs to be developed with short lead times, safety will often appear as a bottleneck (Croxford & Sutton, 1996). An international developer of embedded systems that we have collaborated with reports that as much as 25% of the man-hours in safety-critical software projects are spent on safety certification (Wien, 2010). There are two unfortunate approaches that companies may choose here. One is to delay all safety analysis until late development, the other to do safety analysis only in

DOI: 10.4018/ijismd.2014010101
a superficial manner. There are several problems with both these. Either safety hazards may be discovered only late in the development process, e.g. in system testing, when they will be costly to correct because a number of design decisions have to be reverted, or the safety hazards will not be discovered at all by the developer. In the latter case, they may be discovered by a relevant certifying authority, in which case the system will not be certified - meaning that the entire development effort is wasted. Or, even worse, the hazards are still overlooked and allowed into a production system, which is outright dangerous (Jürjens, 2003). This will also hold for other types of non-functional requirements but safety is special since safety requirements are not about the system but about the system’s relationship to its operating environment.

Previous work has already made some investigations into candidate representations for hazard and risk identification – HazId. In Stålhane and Sindre (2007) there was an experimental comparison of two approaches, one using a combination of use case diagrams and Failure Mode and Effect Analysis (FMEA) tables (Stamatis, 1995), and another using a combination of use and misuse case diagrams (Sindre & Opdahl, 2005). The experiment in Stålhane and Sindre (2008) compared the use of textual use (and misuse) cases with diagrams. The objective in each experiment was to see which technique found the most hazards, and the results indicated that misuse cases did better than FMEA, and textual misuse cases did better than use case diagrams. Finally, in Stålhane, Sindre and Bousquet (2010) there was a comparison between textual (mis)use cases and sequence diagrams, indicating that use cases were better for identifying hazards resulting from the operation of the system (e.g., human operator or user errors), while sequence diagrams were better for identifying system-internal hazards. Sequence diagrams, however, are typically a design-level technique, and do usually not exist in the concept or requirements stage. In all our experiments involving textual uses cases, the documentation has included a system diagram in order to provide a system overview. Hence, an interesting candidate for comparison with textual use cases might be higher level diagrams showing an overview of the system, rather than internal message passing. This will show us which hazards can be identified by the system diagram alone and what a textual use case can add for the HazId process. This is important since many system developers in the concept phase only have a system diagram and want to identify as many hazards as possible at this stage.

This paper is an extension of a previous conference paper (Stålhane & Sindre, 2012). The current paper extends the previous one by reanalyzing the data to also understand why some hazards were never or only seldom identified. This provided a new insight related to what kind of information that needs to be supplied for efficient hazard identification. In addition, we have extended and strengthened the discussion related to lay people versus professionals when it comes to hazard identification.

The objective of the paper is two-fold: (1) to argue that developers – including requirements engineers – and customers can contribute in the early hazard identification and (2) to present an experimental comparison between textual use cases enhanced with a system diagram and, on the other hand, the system diagrams alone, showing a high level view of the system components. The rest of the paper is structured as follows: the second section presents the research questions and hypotheses, the third section discusses related work, and the fourth section explains the experiment design, whereupon the fifth section presents the results. The sixth section discusses threats to validity, and finally the seventh section concludes the paper.

HAZARD IDENTIFICATION AND THE NON-EXPERTS

It is generally agreed that the more information about a system that is available, the better are we able to identify all related hazards. On the other hand, the earlier we are able to do hazard identification, the better are our opportunities to get a safe system. This is generally known as the
Related Content

**A Method for Eliciting Goals for Business Process Models based on Non-Functional Requirements Catalogues**
[www.igi-global.com/article/method-eliciting-goals-business-process/53203?camid=4v1a](www.igi-global.com/article/method-eliciting-goals-business-process/53203?camid=4v1a)

**Accountability for Service Compliance: A Survey**
Jinhui Yao, Shiping Chen and David Levy (2012). *International Journal of Systems and Service-Oriented Engineering* (pp. 16-43).
[www.igi-global.com/article/accountability-service-compliance/64197?camid=4v1a](www.igi-global.com/article/accountability-service-compliance/64197?camid=4v1a)

**High Level Definition of Event-Based Applications for Pervasive Systems**
[www.igi-global.com/chapter/high-level-definition-event-based/55439?camid=4v1a](www.igi-global.com/chapter/high-level-definition-event-based/55439?camid=4v1a)

[www.igi-global.com/chapter/supporting-data-intensive-analysis-processes/115440?camid=4v1a](www.igi-global.com/chapter/supporting-data-intensive-analysis-processes/115440?camid=4v1a)