4-2012

A Safety Case Pattern for Model-Based Development Approach

Anaheed Ayoub
University of Pennsylvania, anaheed@seas.upenn.edu

BaekGyu Kim
University of Pennsylvania, baekgyu@seas.upenn.edu

Insup Lee
University of Pennsylvania, lee@cis.upenn.edu

Oleg Sokolsky
University of Pennsylvania, sokolsky@seas.upenn.edu

A Safety Case Pattern for Model-Based Development Approach

Abstract
In this paper, a safety case pattern is introduced to facilitate the presentation of a correctness argument for a system implemented using formal methods in the development process. We took advantage of our experience in constructing a safety case for the Patient Controlled Analgesic (PCA) infusion pump, to define this safety case pattern. The proposed pattern is appropriate to be instantiated within the safety cases constructed for systems that are developed by applying model-based approaches.

Keywords
safety cases, safety case patterns, model-based development approach, PCA infusion pump

Comments
NASA Formal Methods Symposium (NFM), Norfolk, VA, April 2012.

A Safety Case Pattern for Model-Based Development Approach *

Anaheed Ayoub, BaekGyu Kim, Insup Lee, and Oleg Sokolsky

Department of Computer and Information Science
University of Pennsylvania
anaheed,baekgyu,lee,sokolsky@seas.upenn.edu

Abstract. In this paper, a safety case pattern is introduced to facilitate the presentation of a correctness argument for a system implemented using formal methods in the development process. We took advantage of our experience in constructing a safety case for the Patient Controlled Analgesic (PCA) infusion pump, to define this safety case pattern. The proposed pattern is appropriate to be instantiated within the safety cases constructed for systems that are developed by applying model-based approaches.
Keywords: safety cases, safety case patterns, model-based development approach, PCA infusion pump

1 Introduction

A Patient Controlled Analgesic (PCA) infusion pump is one type of infusion pump that primarily delivers pain relievers, and is equipped with a feature that allows for additional limited delivery of medication, called bolus, upon patient demand. We are developing a PCA implementation software by using the model-based approach based on the Generic PCA model [2] and the Generic PCA safety requirements [1] provided by the U.S. Food and Drug Administration (FDA) as shown in [14].

According to FDAs Infusion Pump Improvement Initiative [16], the FDA has received over 56,000 reports of adverse events associated with the use of infusion pumps from 2005 through 2009. The FDA structured 510k guidance document [15] to assist industry in preparing premarket notification submissions for infusion pumps. These recommendations are intended to improve the quality of infusion pumps in order to reduce the number of recalls and infusion pump Medical Device Reports (MDRs). In 510k submissions, the FDA recommends device manufacturers to submit infusion pump information (i.e., what beneficial properties the manufacturer claims for the infusion pump and how those properties are supported by the provided evidences) through a framework known as an assurance case [5]. This recommendation is the main motivation for our work.

* This research was supported in part by NSF CNS-0930647, NSF CNS-1035715, and NSF CNS-1042829.

An assurance case is a way to demonstrate the validity of a claim by providing a convincing argument together with supporting evidence (e.g., testing results, analysis results, etc.). The 510k guidance document specifically mentions the safety case [11] that is a special form of the assurance case that addresses safety. There is often commonality among the structures of the argument used in safety cases. This commonality motivates the definition for the concept of safety case patterns [11], which is an approach to support the reuse of safety arguments between safety cases. For example, patterns extracted from a safety case built for a specific product can be reused in constructing safety cases for other products that are developed via similar processes.

We are constructing a safety case for the PCA implementation software. The term "PCA implementation software" means the software code that is automatically generated from the GPCA reference model, and then extended to interface with the target platform [14]. The ultimate goal of this safety case construction is to show that the PCA implementation software we developed is acceptably safe, with the intention of providing a guiding example of safety cases for other infusion pumps. We are constructing the PCA safety case concurrently with the PCA implementation development. This concurrent development enables assurance needs to drive development decisions [5].

The main contribution of this paper is to define a safety case pattern that allows the incorporation of the belief in the model correctness obtained by using formal methods in the development process. This pattern is appropriate in constructing safety cases for infusion pumps those are developed using the model-based approach. The paper is organized as follows: we start by briefly giving background information in Section 2. Section 3 describes the main contribution of the paper which proposes a safety case pattern. Related work is discussed in Section 4. Finally, conclusions and ongoing work are given in Section 5.

2 Background

Two important concepts are used in this paper: "safety case patterns" and "the model-based development".

Safety case patterns [12] are defined to capture successful (i.e., convincing, sound, etc.) arguments that are used within the safety case. Whenever a safety case pattern is found to be appropriate to apply in a new safety case development, then it is instantiated within this new safety case. Therefore, safety case patterns allow reusing successful arguments among different safety cases. In essence, the patterns concept attempts to encourage best practice in creating and reviewing safety cases [6]. The Goal Structuring Notation (GSN) is one of description techniques that has proven to be useful for constructing safety cases. Details about GSN can be found in [11]. A number of extensions have been made to GSN to define a safety case pattern language. Those extensions are given in [12].

Model-based development is the notion that we can build systems by constructing abstract representations of the system's behavior and translating them into something that executes on a target platform. A typical model-based ap-

proach includes the following steps: 1) modeling the system, 2) analyzing and verifying this model against the system requirements, 3) systematic transformation of the model into an implementation, and 4) validating the implementation against the system requirements. We applied such model-based approach in developing the PCA implementation software. Therefore, one of the safety case patterns we suggest (described in the Section 3) is an argument about the correctness of the implementation developed using the model-based approach.

3 The Proposed Safety Case Pattern

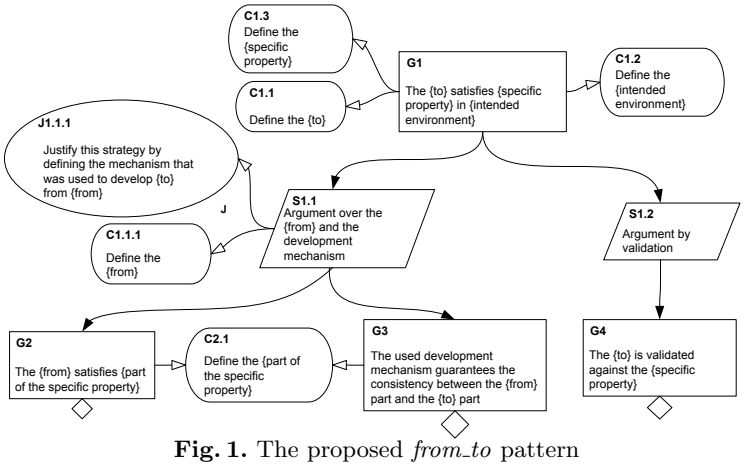

Fig. 1. The proposed *from_to* pattern

We are constructing a safety case for the PCA implementation software we are developing. Due to the page limit, description for the entire PCA safety case is not given. Instead, we concentrate on the safety case pattern extracted from the PCA safety case. The proposed safety case pattern allows one to incorporate the confidence in the model correctness obtained by using formal methods and the confidence in the development process gained by using a well-established development approach. This pattern is appropriate to be used when the system is developed from the formal model using the model-based approach.

Figure 1 shows the GSN structure of the proposed *from_to* pattern. Here, {to} refers to the system implementation and {from} refers to a model of this system. The claim (G1) about the implementation correctness (i.e., satisfaction of some property (referenced in C1.3)) is justified not only by validation (G4 through S1.2) but also by arguing over the model correctness (G2 through S1.1), and the consistency between the model and the implementation created based on it (G3 through S1.1). The model correctness (i.e., further development for G2) is guaranteed through the model verification (i.e., the second step of the model-based approach). The consistency between the model and the implementation (i.e., further development for G3) is supported by the code generation from the verified model (i.e., the third step of the model-based approach). Only part of the property of concern (referenced in C2.1) can be verified at the model level due

to the different abstraction levels between the model and the implementation. However, the validation argument (S1.2) covers the entire property of concern (referenced in C1.3). The additional justification given in (S1.1) increases the assurance in the top-level claim (G1).

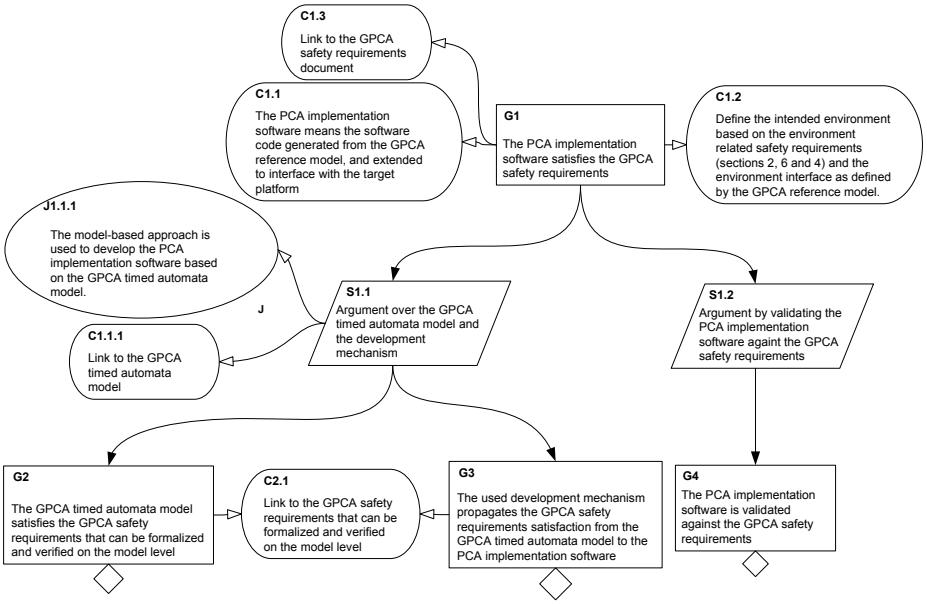

Fig. 2. An instance of the *from_to* pattern

Figure 2 shows an instantiation of this pattern that is part of the PCA safety case. Based on [14], for this pattern instance, the {to} part is the PCA implementation software (referenced in C1.1), the {from} part is the GPCA timed automata model (referenced in C1.1.1) and the GPCA safety requirements (referenced in C1.3) represent the concerned property. In this case, correct PCA implementation means it satisfies the GPCA safety requirements that defined to guarantee the PCA safety. The satisfaction of the GPCA safety requirements in the implementation level (G1) is decomposed by two strategies (S1.1) and (S1.2). The argument in (S1.1) is supported by the correctness of the GPCA timed automata model (G2), and the consistency between the model and the implementation (G3). The correctness of the GPCA timed automata model (i.e., further development for G2) has been proved using the UPPAAL model-checker [4] against the GPCA safety requirements that can be formalized (referenced in C2.1). The consistency between the model and the implementation (i.e., further development for G3) is supported by the code-synthesis from the verified GPCA timed automata model. Not all the GPCA safety requirements (referenced in C1.3) can be verified against the GPCA timed automata model [14]. Only the part referenced in C2.1 can be formalized and verified in the model level (e.g., "*no bolus dose shall be possible during the Power-On Self-Test*"). Other requirements are not formalizable and/or cannot be verification against the model given

its level of details (e.g., "*the flow rate for the bolus dose shall be programmable*" cannot be formalized meaningfully and then verified in the model level).

Generally, using safety case patterns does not necessarily guarantee that the constructed safety case will be sufficiently compelling. So when instantiating the *from_to* pattern, it is necessary to be able to provide justification for each taken instantiation decision to guarantee that the constructed safety case is sufficiently compelling. Guidance for justifying such decisions can be found in [8].

4 Related Work

Assurance cases for medical devices have been discussed in [19]. The work in [19] can be used as staring point for the PCA safety case construction. A safety case given in [10] was constructed for a pacemaker was also developed using the model-based approach. This paper takes a step forward by proposing a safety case pattern for the model-based approach. The concept of safety case patterns was defined in [12]. Many safety case patterns were introduced in [3, 11, 18], but none of them is defined to the model-based approach. Another set of patterns are given in [17]. However, those patterns are introduced only by instantiation examples, limiting their reuse.

The software contribution pattern introduced in [7] is related to the *from_to* pattern. Both concern software development and can be applied iteratively. However, the software contribution pattern is intended to show that the contribution made by the software to the system hazards are acceptably managed. The *from_to* pattern is intended to show the software satisfaction for some concerned property, which can be used to address different aspects. The software contribution pattern is defined to be flexible enough and may be instantiated no matter what development process is used. While the *from_to* pattern is applicable only if the development process guarantees consistency between the developed artifacts. Focusing on a specific development approach (i.e., model-based development) breaks the advantage of the flexibility. The propagation of the correctness between tiers is not part of the software contribution pattern itself [8]. In contrast, the *from_to* pattern argues over the correctness propagation from the {from} artifact to the {to} artifact. This argument strengthens the assurance in the {to} correctness (i.e., the pattern top-level claim).

5 Conclusions

Our ongoing work is constructing a safety case for the PCA infusion pump system that we are developing. In the development of the PCA implementation, we applied the model-based approach starting from the GPCA model. Here, we suggest a safety case pattern that can be instantiated to argue about the correctness of implementations developed using the model-based approach. Where the correctness (i.e., satisfaction of required properties) of the implementation is justified not only by validation but also by arguing over the model correctness and the preservation of this correctness through the development process.

In addition to constructing a safety argument for the PCA infusion pump, we are also working on constructing confidence arguments that are necessary to increase the confidence in the developed safety argument as suggested in [9].

References

1. Safety Requirements for the Generic Patient Controlled Analgesia Pump. http://rtg.cis.upenn.edu/gip.php3.
2. The Generic Patient Controlled Analgesia Pump Model. http://rtg.cis.upenn.edu/gip.php3.
3. R. Alexander, T. Kelly, Z. Kurd, and J. Mcdermid. Safety Cases for Advanced Control Software: Safety Case Patterns. Technical report, University of York, 2007.
4. G. Behrmann, A. David, and K.G. Larsen. A tutorial on UPPAAL. In *Formal Methods for the Design of Real-Time Systems (revised lectures)*, LNCS, pages 200–237, 2004.
5. P. Graydon, J. Knight, and E. Strunk. Assurance Based Development of Critical Systems. In *the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, DSN '07, pages 347–357, Washington, DC, USA, 2007.
6. R. Hawkins, K. Clegg, R. Alexander, and T. Kelly. Using a Software Safety Argument Pattern Catalogue: Two Case Studies. In *SAFECOMP*, pages 185–198, 2011.
7. R. Hawkins and T. Kelly. A Systematic Approach for Developing Software Safety Arguments. *Journal of System Safety*, 46:25–33, 2009.
8. R. Hawkins and T. Kelly. Software Safety Assurance – What Is Sufficient? In *the 4th IET International Conference of System Safety*, London, 2009.
9. R. Hawkins, T. Kelly, J. Knight, and P. Graydon. A New Approach to creating Clear Safety Arguments. In *the 19th Safety Critical Systems Symposium (SSS'11)*, pages 3–23. Springer London, 2011.
10. E. Jee, I. Lee, and O. Sokolsky. Assurance Cases in Model-Driven Development of the Pacemaker Software. In *Leveraging applications of formal methods, verification, and validation (ISoLA)*, LNCS, pages 343–356, 2010.
11. T. Kelly. *Arguing Safety – A Systematic Approach to Safety Cases Management*. PhD thesis, Department of Computer Science, University of York, 1999.
12. T. Kelly and J. McDermid. Safety Case Construction and Reuse using Patterns. In *SAFECOMP*, pages 55–96. Springer-Verlag, 1997.
13. T. Kelly and J. McDermid. Safety Case Patterns - Reusing Successful Arguments. In *IEE Colloquium on Understanding Patterns and Their Application to System Engineering*, London, U.K., 1998.
14. B. Kim, A. Ayoub, O. Sokolsky, P. Jones, Y. Zhang, R. Jetley, and I. Lee. Safety-Assured Development of the GPCA Infusion Pump Software. In *EMSOFT*, pages 155–164, Taipei, Taiwan, 2011.
15. U.S. Food and Drug Administration, Center for Devices and Radiological Health. *Guidance for Industry and FDA Staff - Total Product Life Cycle: Infusion Pump - Premarket Notification [510(k)] Submissions*, April 2010.
16. U.S. Food and Drug Administration, Center for Devices and Radiological Health. *White Paper: Infusion Pump Improvement Initiative*, April 2010.
17. S. Wagner, B. Schtz, S. Puchner, and P. Kock. A Case Study on Safety Cases in the Automotive Domain: Modules, Patterns, and Models. In *ISSRE*, pages 269–278, 2010.

18. R. Weaver. *The Safety of Software - Constructing and Assuring Arguments*. PhD thesis, Department of Computer Science, University of York, 2003.
19. C. Weinstock and J. Goodenough. Towards an Assurance Case Practice for Medical Device. Technical report, CMU/SEI-2009-TN-018, 2009.