Capturing and Documentation of Decisions in Security Requirements Engineering through Heuristics

Stefan Gärtner[1], Tom-Michael Hesse[2], Kurt Schneider[1], and Barbara Paech[2]

[1]Software Engineering Group, Leibniz University Hanover, Germany
{stefan.gaertner, kurt.schneider}@inf.uni-hannover.de
[2]Institute of Computer Science, University of Heidelberg, Germany
{hesse, paech}@informatik.uni-heidelberg.de

1 Introduction

Security of software systems is critical to business because detected security flaws such as the loss of sensitive information or damages can decrease revenue significantly [8]. One reason for security problems is the lack of security awareness in software development. If software engineers are not aware of security concerns as early as in requirements engineering, they cannot appropriately address them in their design decisions [4]. As a consequence, security requirements and derived decisions for the design of a software system have to be identified and documented.

The first challenge is to identify security-related aspects in natural language requirements. They are often described implicitly. Our approach aims to make those aspects explicit through heuristics. The second challenge is to document design decisions based on identified security concerns. Significant portions of knowledge explaining decisions remain implicit during development, so that developing and maintaining the system becomes increasingly difficult over time [2]. For this purpose, our approach is intended to capture and document decisions and their rationale explicitly as a resource for future decision-making. In this paper, we present our tool-supported approach and how it is used in requirements engineering to document decisions.

Running Example. To illustrate concepts of our approach, we use a trading system as it can also be found in most supermarkets. It consists of various cash desk PCs connected to a central store server. In our example, a credit card reader with pin pad is plugged into the cash desk PC, which enables customers to pay their goods by credit card.

2 Tool-Supported Approach

The principal idea of our approach is to provide heuristic feedback on security-related aspects of requirements in order to document decisions. An explicit decision model is used to capture and document requirements, design decisions, and related rationale systematically.

The tool for supporting our approach is based on UNICASE [1]. It is an Eclipse-based knowledge management tool for project and system knowledge in software development. Knowledge is modeled through EMF models for artifacts such as use cases, design diagrams, work items and others.

We extend UNICASE with the heuristic requirements assistant (HeRA) which we developed in our earlier work [5, 6, 7]. HeRA is used to improve natural language requirements. Therefore, heuristics are used to analyze requirements and to provide helpful feedback to requirements editors. For this purpose, heuristics search for imperfect formulations and erroneous properties of requirements. They are derived from an experience base manually.

Identification of Security-related Aspects through Heuristics. In our previous project *SecReq* [4], we extended HeRA to detect security-related requirements through heuristics successfully. To document decisions and capture rationale systemtically, requirements engineers need additional information of recognized security issues such as involved entry points, affected assets, identified vulnerability, and effective countermeasures. For this purpose, we extended our heuristics to incorporate explicitly modeled security knowledge that is retrieved from various sources such as security-relevant requirements of related projects, security standards, laws, and many others. Regarding our example, a heuristic may provide feedback that paying by credit card using a card reader with pin pad is prone to loss of sensitive information. Due to insufficient visual cover, the pin a customer enters using the keyboard of the card reader can be spied out by other customers or the cashier. To mitigate the identified vulnerability, heuristic feedback presents countermeasures such as to improve visual cover or to establish a safety distance. In summary, provided feedback highlights security-related aspects and sketchs out how to mitigate the problem.

Documenting Decisions with Models and Heuristic Feedback. To document decisions and their rationale, UNICASE currently uses the Questions, Options and Criteria approach. Whereas this allows to document argumentations in general, a decision-specific model for knowledge is integrated to

capture relevant information efficiently. This model allows to document decisions and their rationale in a cooperative way by all involved stakeholder. Major benefits are the ability to model many first-class entities such as assumptions, constraints or implications of decisions and to refine given knowledge in an iterative way [3].

The feedback provided by HeRA and the extended decision knowledge model complement each other, as we use heuristic feedback to provide starting points for all following documentation activities. For this purpose, heuristic feedback is directly presented in UNICASE. Thus, requirements engineers can seamlessly transfer relevant feedback to decision-specific model elements such as *Decisions*, which are attached to requirements. Furthermore, *Decisions* are initially created with information provided by heuristic feedback. If the provided feedback is not sufficient or not detailed enough, requirements engineers are able to enrich *Decisions* with further decision-specific model elements such as *Constraint*, *Implication*, *Assumption* and others.

Regarding our example, a requirements engineer uses the provided feedback to look for proper solution in order to improve visual cover. As a concrete solution, the card reader can be rotatably mounted on the cash desk. The requirements engineer documents this solution in the decision-specific model as *Decision*. Provided heuristic feedback is documented as *Assumption* to explain the decision. Due to restrictions by law, the mounted card reader must be fixed permanently, so that customers are not able to steal the reader. The requirements engineer documents this aspect as *Constraint*, which is assigned to the made decision. It is documented as *Implication* that some mechanical fastenings are not compatible with certain card readers. Thus, not all card readers can be used in order to fulfill the given constraint.

In this way, awareness for requirements-related security issues is raised in the following project phases by documenting them explicitly. The used decision model elements and the principal idea of using heuristics for documenting decision is illustrated in Figure 1. As our approach supports creating decision elements as reminders for decision points, we believe it lowers documentation effort for decisions.

Open Issues. The success of our approach highly depends on the quality of the modeled security knowledge. To use our tool-supported approach in an industrial setting, we need a methodological approach to create heuristics from security knowledge systematically. Whereas detailed knowledge leads to more helpful feedback, it is expensive to retrieve and to model. Therefore, we have to evaluate a manageable level of detail. Moreover, not all knowledge is made explicit in practice due to insufficient time or high documentation effort. Consequently, we need to investigate ways to manage the portion of implicit knowledge within decisions.

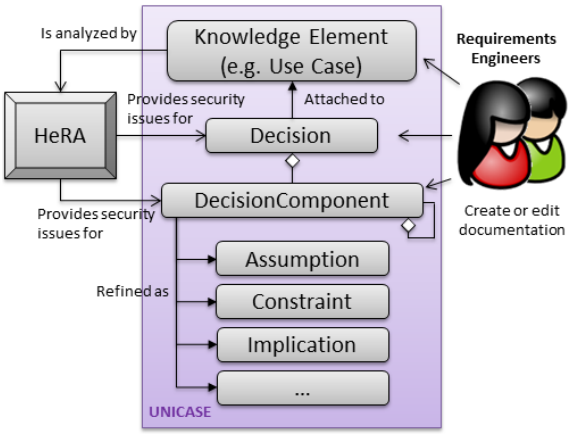

Figure 1: Documenting decisions supported by heuristics and the decision documentation model.

3 Conclusion

In this paper, we explained our tool-based approach to incorporate security knowledge into decision making through heuristics. Our approach supports documentation of decisions and their rationale in security requirements engineering. We therefore extended our tool UNICASE by integrating heuristics.

As a next step, we plan to extend our approach according to the open issues. For this purpose, we seek for industrial case studies.

Acknowledgement

This work was partially supported by the DFG (German Research Foundation) under the Priority Programme SPP1593: Design For Future — Managed Software Evolution.

References

[1] Unicase. http://www.unicase.org/ (Retrieved in 09-2013).

[2] J. E. Burge, J. M. Carroll, R. McCall, and I. Mistrik. *Rationale-Based Software Engineering*. Springer, 2008.

[3] T.-M. Hesse and B. Paech. Supporting the collaborative development of requirements and architecture documentation. In *3rd Int. Workshop on the Twin Peaks of Requirements and Architecture*, pages 22 – 26. IEEE, 2013.

[4] S. Houmb, S. Islam, E. Knauss, J. Juerjens, and K. Schneider. Eliciting security requirements and tracing them to design: an integration of common criteria, heuristics, and umlsec. *Requirements Engineering*, 15(1):63–93, 2010.

[5] E. Knauss and S. Meyer. Experience-based requirements engineering tools. In W. Maalej and A. K. Thurimella, editors, *Managing Requirements Knowledge*, pages 333–351. Springer, 2013.

[6] E. Knauss and K. Schneider. Supporting learning organisations in writing better requirements documents based on heuristic critiques. In B. Regnell and D. Damian, editors, *Requirements Engineering: Foundation for Software Quality*, volume 7195 of *LNCS*, pages 165–171. Springer, 2012.

[7] E. Knauss, K. Schneider, and K. Stapel. Learning to write better requirements through heuristic critiques. In *17th IEEE Int. Requirements Engineering Conference*, pages 387–388, 2009.

[8] US Federal Trade Commision. In the Matter of CardSystems Solutions, Inc., 2006.