Safe Design Real-time Embedded Systems with Security Patterns

Ahmed Ben Mansour, Mohamed Naija and Samir Ben Ahmed
El Manar University, Faculty of Mathematical, Physical and Natural Sciences of Tunis Manar, Tunisia

Keywords: Real Time Embedded Systems, security Pattern, Ontology, Adaptation.

Abstract: Security is a fundamental property in the modeling of real-time embedded systems. Unfortunately, integrating this property is a hard task for a designer due to their small background concerning this feature. Thankfully, the design pattern can provide a practical solution to integrate security through an abstraction mode. However, Nowadays the number of design pattern is increasing, for that reason, the selection of suitable pattern is a fundamental challenge for designers. In this context, we propose in this position paper an approach to integrating security pattern in the phase of modeling of real-time embedded systems. To solve the problem of selection pattern we propose in our approach to use the ontology-based solution, and thus, we propose some methods to guarantee the performance of the systems after integrating.

1 INTRODUCTION

Pattern passed development is becoming increasingly recognized in software engineering. The use of patterns is one of the means by which reusability can be ensured in the systems development because it saves time during the development and as a result reducing the time to market. However, integrating these patterns remain a challenge for developers, which is why it is a critical research topic these days. In fact, patterns are applied in the modeling of modern software architectures, distributed systems, and real-time embedded systems (RTES).

Nowadays, the modeling of real-time embedded systems may be stated as a crucial problem with the current lack of design models and tools. This is why, it is necessary to provide a solution that manages the coexistence of the different hardware and software requirement and at the same time assist non-functional properties (NFPs) (e.g., Time requirement and security specialists). It is also advantageous to provide them with building blocks that meet their needs, of course, methods will guide them throughout the systems life-cycle. A solution based on model-driven engineering (MDE) (Dobson et al., 2006) and patterns seem to meet these expectations.

Besides, design patterns (Gamma, 1995) can provide a practical solution to meet specific needs such as, the security concern that can be addressed using security patterns that encapsulate security solutions by an abstraction mode and allows designers to integrate it in the phase of co-design real-time embedded systems. These types of patterns ensure reusability thanks to their construction advantage which is independent of all environments.

The use of security patterns in real-time embedded systems co-design saves much time and reduces the development cost in RTES systems, but the choice of suitable patterns is a challenge for the designer because of its vast number. Besides, the integration of design patterns can affect the performance of the systems and the architecture quality, for example, the integration of the pattern can increase the time response of the system. The designer must take into consideration the different effects when he chooses the correct pattern.

In this position paper, we propose an approach to select the right security pattern that is suitable for solving security problems and respecting the different time constraints of the system. It represents a generic illustration backing into the co-modeling of features to be used in the context of modeling security and analysis time constraint in the RTES.

Our approach aims to guide RTES designers in building and verification of safe RTES models. It facilitates complex systems modeling, reduces the development time and improves software process quality.

The remainder of this position paper is organized as follows. In section II, we give background on related concepts. In section III, we present an overview of the proposed approach. Finally, we conclude with some outlined directions for future works.

Ben Mansour, A., Naija, M. and Ben Ahmed, S.
Safe Design Real-time Embedded Systems with Security Patterns.
DOI: 10.5220/0007749104590463
In *Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2019)*, pages 459-463
ISBN: 978-989-758-375-9

2 THEORETICAL BACKGROUND

The increase in the importance of security in organizations leads to the birth of several research-oriented levels for the integration of security concept in various phases of software development lifecycle. In this section, we present some works on the integration of security design patterns, and then we present some existing ontologies in the same field. Finally, we present different works about adaptation mechanism.

2.1 Review of Security Pattern Selection

The authors proposed in (Alam et al., 2007), (Fernandez et al., 2007a) an approach for integrating security patterns at all phases of systems development. These guidelines affect the phase of design, implementation, testing, and deployment of the systems. In fact, the authors presented a recommendation for follow-up without giving details on the modeling or the concrete solution.

In (Fernandez et al., 2007b) and (Toahchoodee et al., 2009) the authors used the oriented programming paradigm to integrate security patterns into the modeling of the system. The security aspects are modeled independently by using the security patterns to be woven after that with the functional models.

The authors in (Bouaziz et al., 2015) proposed a process to integrate security pattern in the different phases of system development. This process provides separation between domain expertise and application security expertise. The primary goal in this work is to provide a semi-automatic integration of security pattern into component-based models. This integration is performed through a set of information rules. The result of this integration is a new model supporting security concepts.

2.2 Review of Security Ontology

The advantages of applying ontology technology into the information security domain are specific in (Guan et al., 2016) from three viewpoints: 1) ontologies can eliminate the ambiguity of items to a properties list and systematically organize information at a detailed level. 2) ontological technology can induce the modularity which can be used by other approaches. 3) an ontological approach can forecast security problems by providing inference mechanisms.

Schumacher proposed in (Schumacher, 2003) a security ontology to maintain the security pattern repository with a general security pattern search engine. However, only top-level concepts were introduced in their ontology, which is too abstract to apply to the specific context.

In (Denker et al., 2005), Denker had mainly addressed the knowledge representation and some of the reasoning issue for trust and security in the semantic web. Enforcement mechanisms, automatic negotiation, and tools for semantic web security are much less developed and provide ample opportunities. They summarized an ontological approach to enhancing the semantic web with security using ontology.

In (Souag et al., 2015) the authors presented a core and general security ontology for security requirements engineering. Its core and general status are attained thanks to its coverage of broad and high-level security concepts and relationships.

2.3 Review of Adaptation

The design of adaptive RTES is relatively old domain and thus very rich. Unfortunately, these tasks present a challenge due to the complexity of the problem it handles (Said et al., 2014). A typical fundamental challenge is optimizing system non-functional properties while meeting internal and external constraints (real-time constraint). Such as, high quality of service may require a high utilization of system resources (CPU cycles and memory space).

In (Said et al., 2014), the authors have proposed five patterns representing generic modeling of the adaptation loop modules. The patterns take into consideration the real-time features of adaptation operation. These design patterns are presented in the static view through class diagrams annotated with MARTE (OMG, 2008) profile stereotyped. The adaptation is then considered as a dynamic and partial change of the operation mode without supporting platform resources.

In (Yousef and Khaled, 2015) the authors proposed a feedback loop design process for self-adaptation. Feedback loops provide the generic mechanism for self-adaptation. It is a control loop where the output of the controlled system is fed back to the input. Typically, it involves four key activities:

- Monitor: it is responsible for collecting data from a set of sensors, filtering and aggregating this data and sending it to the analyzer component for any possible symptoms of system goals violation.
- Analyzer: its responsibility is to receive collected and logged data from monitor representing the system state history and analyze them for any possible symptoms of system goals and requirements violation.
- Plan: is responsible for the construction of the change plan in response to an adaptation request received from the analyzer.

- Execute: is responsible for sending the corrective actions to the effectors in a specific order.

3 SECURITY PATTERN INTEGRATION

In this section, we present the general idea of our approach for the integration and selection of security patterns in the modeling of RTES. The approach is based on the scenario that was defined in (Motii et al., 2015) and (Motii et al., 2016), we modify some steps to guarantee the automation and to solve several problems that are not solved, and we redefine some phases in the scenario.

The figure shows the different phases of the approach. Our approach is based on the separation between the functional model and the non-functional properties. For that, the first step in our approach is the modeling of the functional model with architecture design. This model describes the functional aspect of the system and contains the several operations of the software system.

The second step is the functional model sent to the risk analysis. This phase for listing the threats and deriving the security requirements. This analysis is carried out by a safety risk analyzer. The security requirements describe what should be provided by the security mechanisms to stop some threats.

Based on security requirements, a method to search a security pattern must be described to ensure the identification of the suitable patterns that can solve the described security problem automatically. To achieve this goal, we will seek to define an ontology to look for the appropriate security pattern, for that we will describe and store the security requirements and the security patterns semantically.

For the semantic representation of the security requirements, we will use OWL (Web Ontology Language) to guarantee the automatic mapping of requirements to its solution. For the semantic representation must contain all the necessary information of a security pattern, its level of abstraction, type of solution, the context and its integration method. Two libraries will be defined: a library of security pattern and a library of the security ontologies.

The operation of the search algorithm will be to look for semantic representation of security requirement suitable for a semantic pattern presentation if it exists then the ontology will send the proper representation of the pattern to extract the proper security pattern that can solve the security problem. Finally, the security pattern will be selected and prepared for integration into a functional model by the system designer. A secure model will be the result of the integration.

After the steps of integration, the operation of the system will be modified; for that, an evaluation must be provided to validate this integration. A model containing the operation of the system must be related to the hardware resources. We will use the profile UML/MARTE for the allocation of the software task in the hardware resources. In the evaluation test, the time constraints for each task must be taken into consideration; for example, when we add the security requirements to the task corresponding to its execution time, the test must verify that change does not exceed to task deadline.

If the evaluation test is positive then the integration of security pattern will be accepted, if not another solution will be provided to validate the integration of pattern, the adaptation solution.

The adaptation solution consists of the changing of the structure or the system behavior to respond to the change of its functioning, in our case, the adaptation is the reconfiguring of the allocation between the resources and the appropriate tasks after the modifying of the execution time. For this solution, we will use the same pattern from the MAPE loop for the self-adaptation after we modify some operations. We will use only three patterns from the loop:

- DecisionMaker: for generating of adaptation decision that specifies the elements to be modified and how to modify them to meet the requirements of the system after pattern integration better.
- Actor: for applying the decision. It maps the actions to the effector interfaces.
- Effectors: it is responsible for applying the change in the allocation of tasks in different resources.

If the validation test (the same scheduling analysis) is valid, then the secured model will be validated, if not then the pattern chosen was not the suitable for the system operation, and we must return to the search patterns phase to find another pattern that responds to the same requirements.

4 CONCLUSION

In this position paper, we proposed an approach for the integration of security patterns in a real-time system model. This approach is based on ontology to automatically select the appropriate security pattern to the security requirement that was extracted by the security risk analysis. Then we dealt with the problem of system performance according to the time constraints validation, and we proposed an alternative so-

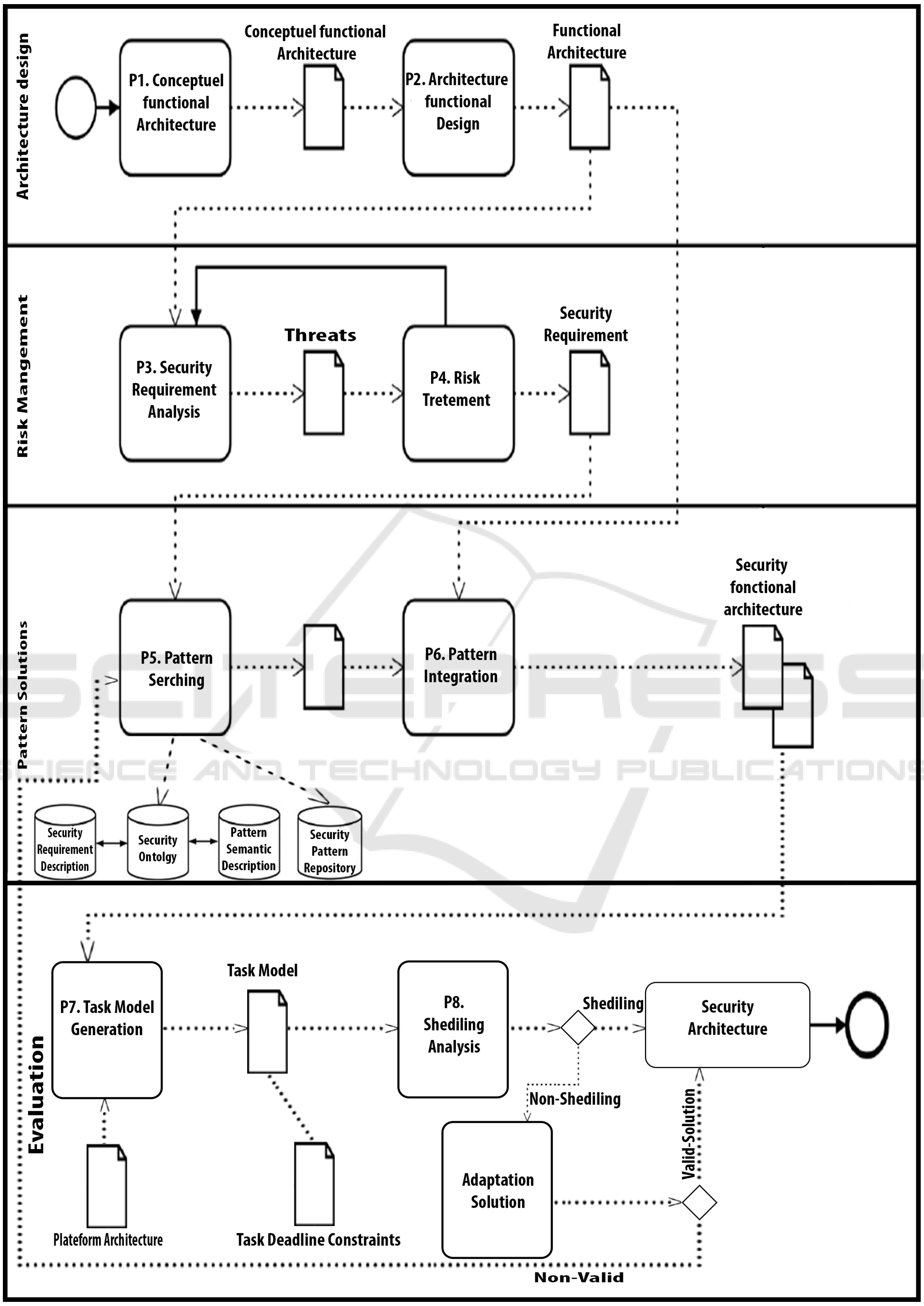

Figure 1: Integration Patterns phases.

lution to the scheduling test which is based on the system adaptation to guarantee the correct functioning of the system with the integrated patterns.

Our future work, we focus on two main areas. One is the extension of the requirement ontology. The other area is the Automatic benchmark integration in the real-time system model; one takes into consideration the other criteria that perform to system performance such as runtime and energy.

REFERENCES

Alam, M., Breu, R., and Hafner, M. (2007). Model-driven security engineering for trust management in sectet. *JSW*, 2(1):47–59.

Bouaziz, R., Krichen, F., and Coulette, B. (2015). C-scrip: collaborative security pattern integration process. *International Journal of Information Technology and Web Engineering (IJITWE)*, 10(1):31–46.

Denker, G., Kagal, L., and Finin, T. (2005). Security in the semantic web using owl. *Information Security Technical Report*, 10(1):51–58.

Dobson, S., Denazis, S., Fernández, A., Gaïti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N., and Zambonelli, F. (2006). A survey of autonomic communications. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 1(2):223–259.

Fernandez, E. B., Ballesteros, J., Desouza-Doucet, A. C., and Larrondo-Petrie, M. M. (2007a). Security patterns for physical access control systems. In *IFIP Annual Conference on Data and Applications Security and Privacy*, pages 259–274. Springer.

Fernandez, E. B., Pelaez, J. C., and Larrondo-Petrie, M. M. (2007b). Security patterns for voice over ip networks. In *Computing in the Global Information Technology, 2007. ICCGI 2007. International Multi-Conference on*, pages 33–33. IEEE.

Gamma, E. (1995). *Design patterns: elements of reusable object-oriented software*. Pearson Education India.

Guan, H., Yang, H., and Wang, J. (2016). An ontology-based approach to security pattern selection. *International Journal of Automation and Computing*, 13(2):168–182.

Motii, A., Hamid, B., Lanusse, A., and Bruel, J.-M. (2015). Guiding the selection of security patterns based on security requirements and pattern classification. In *Proceedings of the 20th European Conference on Pattern Languages of Programs*, page 10. ACM.

Motii, A., Hamid, B., Lanusse, A., and Bruel, J.-M. (2016). Guiding the selection of security patterns for real-time systems. In *Engineering of Complex Computer Systems (ICECCS), 2016 21st International Conference on*, pages 155–164. IEEE.

OMG (2008). Object management group. a uml profile for marte: Modeling and analysis of real-time embedded systems. In *Beta2, Object Management Group.*

Said, M. B., Kacem, Y. H., Kerboeuf, M., Amor, N. B., and Abid, M. (2014). Design patterns for self-adaptive rte systems specification. *International Journal of Reconfigurable Computing*, 2014:8.

Schumacher, M. (2003). *Security engineering with patterns: origins, theoretical models, and new applications*, volume 2754. Springer Science & Business Media.

Souag, A., Salinesi, C., Mazo, R., and Comyn-Wattiau, I. (2015). A security ontology for security requirements elicitation. In *International symposium on engineering secure software and systems*, pages 157–177. Springer.

Toahchoodee, M., Abdunabi, R., Ray, I., and Ray, I. (2009). A trust-based access control model for pervasive computing applications. In *IFIP Annual Conference on Data and Applications Security and Privacy*, pages 307–314. Springer.

Yousef, A. and Khaled, S. (2015). The impact of architectural styles on self adaptive systems engineering. *The Impact of Architectural Styles on Self Adaptive Systems Engineering*.