Patterns in Safety System Development

Jari Rauhamäki, Timo Vepsäläinen and Seppo Kuikka
Department of Automation Science and Engineering
Tampere University of Technology
Tampere, Finland
jari.rauhamaki|timo.vepsalainen|seppo.kuikka@tut.fi

***Abstract*—Development of safety systems for modern industrial control applications is challenged on the one hand by ever growing systems and on the other hand by increasing cost pressures. That is, design process efficiency is a crucial aspect. How to efficiently utilize existing engineering knowledge and document suitable approaches to the common problems of the domain? Design patterns provide a design process with solutions. Design patterns can represent existing knowledge from past projects or illustrate solution blueprints inspired indirectly, e.g., by safety standards. Thus, they provide a designer with support for design decisions during a development process.**

Keywords-design pattern; safety; control system; engineering

I. INTRODUCTION

Safety awareness is constantly increasing across engineering disciplines, regulative governing bodies as well as customers. This trend results in an increasing demand for higher safety integrity levels as well as broadens the product spectrum in which safety systems are deployed. On the other hand, safety system engineering has a constantly increasing need to make the design process efficient in terms of schedule and cost. These issues lead to pressure to increase the efficiency of the safety system development process.

Engineering industry produces vast amounts of tacit and explicit knowledge during customer and R&D projects. This knowledge is a valuable resource that can be used to increase efficiency when available in a suitable format. Explicit project knowledge is typically left as is, i.e., produced knowledge is archived, but it is not indexed or otherwise edited to be easily accessible. Engineers can access the information, but they need to know exactly the project id, subsystem, diagram etc. to locate the existing solution to the problem they are working with. In the context of safety system development explicit existing knowledge could be, for example, a solution to arrange communication between safety-critical and non-safety-critical subsystems according to a safety standard. Tacit knowledge is another source of valuable engineering knowledge. Tacit knowledge is knowledge of individuals or organizations, not available in explicit documented format. In a context of safety system development tacit knowledge could be for instance a solution model of an engineer to a certain problem.

Development of a safety system is bureaucratic and costly, typically regulated by legislation, regulations and standards, which set requirements for the development process. Typical requirements are sets of certain safety functions that need to be implemented in the system and collections of methods and techniques that need to be utilized to achieve sufficient safety integrity levels of the safety functions to reduce risks into a tolerable level. The standards, legislation and regulations require various matters, but give little to no solutions on how these requirements can be fulfilled not to mention guidance for practical safety function implementation.

Our proposed solution for the problems above is application of design patterns in the field of safety system development. Design patterns document solutions to problems commonly encountered and they have proven their value in engineering disciplines such as software engineering [1]. In software engineering large amounts of patterns have been identified and documented.

The contribution of this article is to show how design patterns could benefit the engineering process also the in domain of safety system development. We indicate the rationale to use design patterns in the safety system engineering domain, which is not similar to traditional software engineering though some reasons of use are obviously the same. Problematic issues related to patterns in context of safety system engineering are discussed to provide a broader viewpoint. This also provides a premise and rationale for further studies considering the topic.

The article is organized as follows. In section II we provide background information on design patterns and the domain. Section III presents related work and positions the research. In section IV we present a generalized model of a development process in which safety related aspects are involved and illustrate pattern usage in such a process. In section V the justification for the usage of design patterns in context of the safety system development is discussed in detail. In section VI, the challenging issues of design pattern usage in the domain are pointed out. Sections VII and VIII discuss future work and conclude the article respectively.

Copyright (c) IARIA, 2013. ISBN: 978-1-61208-268-4

II. BACKGROUND

A. Patterns in engineering

The concept of design patterns originates from Christopher Alexander's book: A Pattern Language: Towns, Buildings, Construction [2]. The book illustrates 253 patterns considering architecture, urban design and community habitability. Thus, the roots of design patterns are originated in a domain that has been studied and used for hundreds of years. This illustrates the original nature of design patterns, which is to document solutions identified from real world applications.

Alexander defines design patterns as abstracted solutions to recurring design problems in a given context [2]. This definition is also adopted by the Design Patterns: Elements of Reusable Object-Oriented Software [3], which considers design patterns in the domain of software engineering. The definition includes the three main elements of a design pattern: context, problem and solution. Patterns illustrate solutions to problems that can be applied, in a suitable context, many times but never end up with completely identical solutions.

An analogy for pattern solution application can be found in interior furnishing. An apartment building may have dozens of apartments with the same floor plan, but none of the apartments is similar in interior decoration. When residents move in an apartment, they furnish it, i.e., let us assume they apply an imaginary "Furnish for habitability" pattern. The context of the pattern is an unfurnished and empty apartment, the problem is the low habitability of an unfurnished apartment and the solution is to furnish the apartment with furniture, textiles and other decoration elements to improve habitability. As none of the apartments have identical furnishing the "Furnish for habitability" pattern has been applied multiple times but ending up with a distinct outcome each time.

Process patterns illustrate processes used to complete a task. The purpose is to divide the execution of a task into steps and provide instructions how to execute the steps to complete the whole task. [4]. Process patterns also represent the context, problem, solution paradigm.

B. Two kinds of control systems

Safety systems often, though not always, co-exist and sometimes also co-operate with ordinary control systems. A control system is a system consisting of sensor(s), logic(s) and actuator(s). In this sense, a control system is similar to an E/E/PE (Electrical/Electronic/Programmable Electronic) safety system.

The purpose of a control system is, however, different from the purpose of a safety system. The main purpose of a control system is to control a machine or a process to produce a desired output, e.g., rolls of paper or printed circuit boards. The purpose of a safety system is to ensure the safety of humans, environment and machinery itself, i.e., the main concern of a safety system is to prevent the realization of hazards.

The problem is that the tasks of these two systems differentiate in purpose. A safety system tries to retain the system in a safe operation state whereas a control system tries to maximize the output of the system. To carry out the tasks the same process variables need to be considered. The situation is even worse as the systems often have opposite preferences considering the state of the system.

C. Some patterns for functional safety system development

In our recent research projects, we have focused on safety system principles and architectures. It was noted that patterns for safety systems are not available although patterns for the related domains are considered (see section III). However, we see potential in patterns in the domain of safety system development. During the recent projects, we have identified and developed patterns for the development of safety systems. Table I summarizes the patterns published in VikingPlop'12 [5].

The patterns consider various aspects of safety systems. The Separated safety and the Productive safety patterns consider the co-existence and distribution of liabilities between the safety and main control systems. The Separated override, De-energized override and Safety limiter patterns illustrate approaches to override the main control system with a safety system. The approaches have distinct redeeming features and downsides. For instance, the Separated override pattern emphasizes separation between the systems whereas the Safety limiter pattern allows cooperation between the systems and reduces the amount of needed hardware. The Hardwired safety pattern proposes usage of a hardwired safety system instead of a software based solution in a suitable context.

TABLE I. FUNCTIONAL SAFETY SYSTEM PATTERNS [5]

Pattern	Description
Separated safety	Development of a complete system according to safety regulations is a bureaucratic and slow process. Therefore, divide the system into basic control and safety systems and develop only the safety system according to safety regulations.
Productive safety	A control system utilizes advanced and complex corrective functions to keep the controlled process in the operational state. These functions are very hard to implement in a safety system. Therefore, implement the corrective functions in a basic control system and use simple(st) approach for the safety system.
Separated override	A safety system must be able to override a basic control system whenever systems control same process quantities. Therefore, provide the safety system with a separate actuator to obtain a safe state.
De-energized override	A safety system must be able to override a basic control system whenever systems control same process quantities. Therefore, let the safety system use de-energization of the basic control system's actuator(s) to obtain a safe state.
Safety limiter	A safety system must be able to override basic control system whenever systems control same process quantities. Therefore, disengage the basic control system completely from the actuator and let the safety system control the actuator. Route the output of the basic control system to the safety system and let the safety system treat the control value so that safe operation is ensured.
Hardwired safety	Development of safety-related application software for simple safety function is bureaucratic, time consuming and costly. Therefore, instead of a software-based solution, use a hardware-based safety system.

Copyright (c) IARIA, 2013. ISBN: 978-1-61208-268-4

III. RELATED WORK

Design patterns have been studied and documented in the field of software engineering extensively covering for example object-oriented software [3] and [6], Pattern-oriented architecture [7] and [8], enterprise applications [9], [10], and service-oriented architecture [11]. These books concentrate on software engineering for desktop and server-side applications and architectures. Though these patterns may be usable in safety system development they are not focused on safety aspects.

Fault tolerance is a part of safety system design as safety systems should preferably be fault-tolerant to be able to operate under fault conditions and ensure safety. However, fault-tolerance is not a sufficient condition for safety. Fault-tolerant software can be hazardous in a safety system if the functionality of the software is hazardous, e.g., due to erroneously set requirements. Design patterns for fault-tolerant software systems have been introduced, for example, by Hanmer [12].

E/E/PE safety systems include both hardware and software components. Armoush [13] and Douglass [14] introduce design patterns covering software and hardware aspects of safety systems. The presented patterns are focused on redundancy, which, again, is an approach to increase reliability and fault-tolerance of a system.

Eloranta, Koskinen, Leppänen and Reijonen [15] have studied distributed machine control systems and documented patterns for the design of such systems. Some of the patterns are also related to functional safety aspects. The application domain of the above patterns is closely related to our design patterns considering safety system development and architecture [5].

Koskinen, Vuori and Katara have studied and developed process patterns for the application of the IEC 61508-3 standard. In their article [4] they stated that process patterns can speed up the training of inexperienced engineers and remove ambiguities typically related to safety standard application. This provides additional support for the usage of patterns in the domain of safety systems.

Riehle [1] properly points out three main usage areas of design patterns in current software industry practice. These areas are communication, implementation and documentation. In this article, we consider how these usage areas are transferable into safety system engineering.

IV. GENERALIZED SAFETY SYSTEM DEVELOPMENT PROCESS

In this section, a generalized process for the development of safety-related E/E/PE systems is illustrated. The purpose is to provide an idea about how pattern usage can relate to such a process. The illustrated process is inspired by the IEC 61508-1 overall safety lifecycle [16] and eight steps to safety [14].

Development of a safety system begins with the definition of the overall *scope* of the EUC (Equipment Under Control) and the concept of the system. In this phase understanding about the system and its environment is built. In this context process patterns can be used to identify EUC related aspects (e.g., what are typical characteristics of, for example, bending machines) and typical machinery concepts.

Hazard and risk analysis follows the scope definition. Hazard and risk analysis forms a significant part of the development process as the results directly impact on the coverage of the safety system and selection of safety measures and safety integrity levels. Patterns can be used to identify typical hazards related to specific systems (machinery type) and processes (operations executed by the machinery). Process patterns can be used to describe and interpret the phases of the hazard and risk analysis as required by the followed standard.

When the risks are defined, the *requirements* for the risk mitigation methods are documented in the requirement specification phase. This includes the definition of the risk mitigation methods, safety functions, and the non-functional requirements and safety integrity levels related to them. Patterns can be used to document typical approaches to mitigate risks with the positive and negative effects related to the approaches thus providing support for decision making. The requirement specification phase can also be supported with process patterns. For instance, the Software Safety Requirements Specification pattern in [4] illustrates a requirement specification process mined from the IEC 61508 to provide help and document the sub phases of this development phase.

As the requirements for safety measures and functions are defined the process can continue on to the *realization* of safety system. The phase consists of design and implementation of the safety system. In this phase of development process, patterns have value as the level of abstraction suits well to describe solutions to design and implementation problems. The patterns provide designers with documented solutions to commonly encountered safety design problems. However, the patterns also provide information about consequences related to application of them. This enables an engineer to select the most suitable solution by justifying the consequences. For instance, the three override patterns described in Table I illustrate different approaches to a design problem where a safety system should be able to override a control system. Each of the solutions has their own consequences and the designer can choose the one that is the best fit for the system under development. Process patterns can support the realization process by, e.g., providing support to carry out the recurring phases of development such as the modification or architectural design of the software [4].

The implementation part of the *realization* phase can be supported with design patterns as in this phase engineers encounter a large number of common problems where design patterns are able to provide solutions. The patterns applied in the implementation phase often represent a lower level of abstraction and provide focused solution models to lower level implementation problems.

The rest of the development process relate to *validation, verification, testing, installation and maintenance* aspects. Process patterns for validation and verification document and help to follow the processes. For instance, patterns for

Copyright (c) IARIA, 2013. ISBN: 978-1-61208-268-4

validation and verification in context of the IEC 61508 are provided in [4]. Maintenance of long life cycle systems benefits from the usage of design patterns as known solution models are used.

V. RATIONALE FOR DESIGN PATTERN USAGE IN SAFETY SYSTEM ENGINEERING

As illustrated in Section II/A design patterns have redeeming features in context of the software engineering domain. However, the software engineering domain, at least desktop software engineering, is different from safety system engineering. Of course, software systems are a part of modern safety systems, but the nature of a pure software system is distinct from a safety system. In the following subsections rationale for the usage of design patterns in safety system engineering is discussed.

A. Ability to avoid physical damage

Normal application/desktop software, run on a personal computer, mobile device, server or similar device, has limited possibilities to interact with its environment. Potential physical risks associated with such devices and applications are, for example, overheating, electric shock, battery malfunctions and fans none of which are directly controllable by the application software ran in the system. That is not to say that application software cannot be critical. For example, a failure of banking, insurance or other large scale business system may inflict massive losses to its owners in form of revenue or work contribution losses and is thus considered critical. However, no (direct) human, environmental or machinery related hazards exist in such cases.

Systems in which safety systems are deployed are able to cause hazardous situations for humans, environment and hardware by their nature (if not, no safety control system would be required). Industrial and machinery control systems operate actuators (e.g., fans, valves, and heaters) process devices (e.g., conveyors, robots, and guillotines) and substances (e.g., toxic chemicals or hot fluids) that are hazardous for humans, environment and the systems itself. As the safety systems are dedicated to mitigate risk related to such machinery they are expected and required to have certain level integrity to carry out the safety functions.

Design patterns document good approaches, practices and solutions common in safety system development. This provides designers with tried solutions to problems as well as removes the need to reinvent the wheel thus resulting in a more productive development process as well as solutions with a justified approach. The development burden is decreased and the designers can focus on details as patterns describe the main solution model.

B. Experience as a valuable resource in safety system development

In the field of safety system engineering, well-tried solutions are welcome as they have additional empirical data to back up applicability. By identifying patterns from existing projects and designs and making the solutions explicit in patterns, experience can be transferred from one engineer to another. Design patterns support the illustration of experience in explicit format by requiring the pattern writer to consider different aspects of the solution. This work is carried out in consideration on the context in which the solution can be used, consequences and the resulting context related to the solution. Patterns document (or at least they should document) also negative consequences, preconditions and assumptions related to pattern application. This provides engineers with a foundation to use or not to use certain solutions and compare them against each other to select the best approach for the problem under consideration.

A good approach is to document the proven solutions of past projects into patterns to be used in forthcoming projects. In this way, the patterns are directly related to the domain, they can be written to solve a dedicated problem and the consequences are known. That is not to say one should limit to such patterns only. Third-party patterns may provide fruitful insight into other kinds of solution models and open new kinds of approach possibilities to solve a certain kind of problem with more desirable consequences.

Experience illustrated in format of patterns, also provides a name for the solutions and approaches. This enables the usage of patterns as a part of communication [1], but requires that the patterns have reached awareness of the engineering community using them. When this point is achieved, patterns can be used in communication to illustrate the solutions and approaches described in design patterns. For example, safety system engineers could discuss about how to override a control system with a safety system: "I think separated override [5] would be a good approach in this situation.", "I disagree; I find separated override an excessive action as it would require an additional safety actuator. Maybe we should consider de-energized override [5] instead", "That is true, de-energized override is a more cost-effective approach in this case."

C. Alleviating bureaucracy

Development of safety systems is regulated by directives, legislation and standards such as [17], [18], [19]. Such documents are written partly from a legislative point of view, are too generic to cover various applications and domains, and do not (want to) strictly enforce a certain approach. These aspects restrict the documents from providing solution models. Rather such documents require various techniques, methods, and processes to be used in the development of safety systems, but give minor importance on examples or other guidelines for any specific implementation. In addition, the documents are massive, often hundreds of pages long, which makes finding solutions difficult. This does not mean standards etc. are useless; they just have a different view to safety systems compared with patterns. The standards provide a framework that is applied in a certain way to develop the system. The framework describes methods and techniques to develop safety systems and, e.g., define what to verify and validate when a safety system is being developed. This is certainly a valuable aspect in safety system development.

The purpose of patterns in this context is to supplement the standards and document the solution models and

Copyright (c) IARIA, 2013. ISBN: 978-1-61208-268-4

approaches compliant with the given requirements. The IEC 61508-3 [20], for instance, illustrates a number of techniques and measures to be used in the development of safety-critical software. However, little information on how these techniques shall be used and what kind of solution models they (may) produce is given. Especially safety-related standards can prove hard for a person with limited experience in the development of safety systems [4]. With design patterns, solutions and approaches to implement techniques and measures required in standards can be documented, which illustrates the usage of design patterns as a source of implementation [1]. Process patterns can be used to capture the recurring tasks in the development of a safety system [4].

In context of safety system development the value of patterns is fully established when patterns are mined from a system that has already found compliant with a safety standard. This adds confidence in the solution model validity in context of the considered safety standard. Such patterns can increase development process efficiency as the solution model can be used in other systems with a fairly good confidence as long as the context described in the pattern matches the context in which the pattern is applied. The solution, approach or method once approved in a certification process or assessment for standard compliance, for instance, is useful as it provides at least the main solution framework for the problem under consideration.

Development of a safety system also requires extensive documentation. This is required, e.g., to illustrate compliance with a standard considering development of a safety system or an informal document illustrating the safety foundation of a system, which can be used as a part of safety assessment of a system. Design patterns can be used for documentation purposes [1]. The applied patterns and roles of the patterns can be marked in a document (e.g., in a diagram). For an experienced pattern user this quickly indicates the type of solution used (described by the pattern). The need for reading textual representations decreases as the reader can obtain the information on the roles of the system elements directly from the diagram. In an informal supplementary documentation usage of well-known safety related patterns can be justified. The reader is able to identify the patterns applied and assess their suitability in context of the safety system under consideration. However, in context of the legal safety system documentation, the usage of patterns in documentation does not remove the need for textual representations as the usage of pattern notation in the documentation does not cover the whole functionality and all the aspects of the applied solution.

D. Co-existence of control and safety systems

A safety system often co-exists with a main control system as stated in section II. Although safety and control systems are designed to be separated, they often need to be connected some way (e.g., to share state and operation information). This aspect further increases the amount of work needed to design an operational entity consisting of safety and control systems.

Integration of safety and main control systems is sometimes, especially in context of larger processes, a unique design. The operation and responsibilities of the safety and control systems need to be defined and fitted to operate in harmony. If such a system is repeatedly designed from scratch, a great amount of design work needs to be redone. In such situations the design process may greatly benefit from the reuse of templates [21], model libraries and similar ways of reusing existing designs developed in a specific development environment. However, templates and library solutions as such are not a good fit to document solution models and approaches on a generic level. This is due to the fact that solutions are bound to the implementation environment: the solutions are described in terms of the implementation environment/tool. Such an approach complicates the understanding about the solution on a higher level of abstraction.

Contrarily, patterns provide a format to document solutions on a platform independent level. This enables the documentation of solutions, which can be used in different implementation environments as long as the context and other prerequisites are considered. The benefit of a pattern approach is that one is able to take the idea from a pattern and adapt the principle of the solution to solve the problem in hand, thus increasing the efficiency of the design process.

1) A case for pattern usage in design of safety and control system co-existence

This section illustrates a case for usage of design patterns is design of system in which safety and control system operate the system under control. The functional safety system patterns introduced in Table I illustrate solutions for safety and control system co-existence. The patterns describe approaches to arrange the responsibilities of the systems and override of the control system. The idea is to divide the responsibilities so that the development of the safety system is as lightweight as possible, but the safety system still is retaining full control over the machinery.

A potential design decision flow to utilize the patterns is illustrated in Fig. 1. The figure illustrates the pattern relations of the patterns in Table I. The Separated safety pattern is applied first to the system under development. This decision results separated safety and control systems and only the safety system has to be developed according to safety standards, which decreases the development burden considerable as the control system (which is typically a larger entity than the safety system) can now be developed

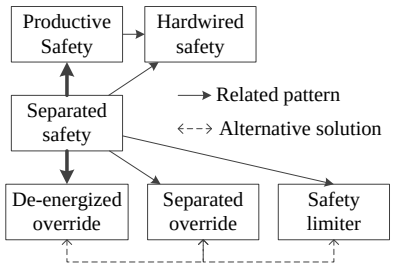

Figure 1. Design flow using functional safety system patterns [5]

without safety standard conformance.

When the system is divided into safety and control system, the Productive safety pattern can be considered. It suggests that the control system implements the basic interlock mechanisms that (try to) keep the system in a normal operational state as far as possible. The interlock mechanisms can be as complex as needed as they do not need to conform to the safety standards. The actual safety functions are implemented on the safety system and they can be rather simple because the control system keeps the machinery in the normal operational state. The safety system can, for instance, only implement an emergency shutdown of the machinery when the control system has failed to retain the normal operational state. This approach simplifies, and thus potentially lowers the cost of, the safety system development and implementation.

As the safety system must be able to drive the system into a safe state regardless of the control system state, the designer needs to implement such functionality. The three override patterns provide three distinct approaches how the safety system can override the control system on the actuator level. The designer can compare the suggested approaches and select the one with most desirable consequences regarding the system under control. For instance, if separation between the safety and control system is the main concern, the Separated or De-energized override pattern is the most appropriate. However, if there is a need to lower the amount of actuators or use advanced safety functionality, the Safety limiter pattern may be a better alternative.

The above workflow illustration also depicts the potential of pattern language utilization. The designer uses a pattern language as a framework and selects the most appropriate patterns to design the system. The pattern language supports the design process by defining relationships between the patterns. The relationships illustrate, e.g., patterns that are applicable after a certain pattern has been applied, conflicting patterns or patterns that solve problems, which may arise when a pattern is applied.

E. Maintainability of common solution models

An important feature of control systems, especially in an industrial domain, is long life-cycles. As safety systems are part of control structures of a system, they also have long life-cycles in similar applications. The maintenance phase of a system may contribute considerably to a large part of system design and development costs when the whole life-cycle costs of the system are considered. Thus the maintainability of a safety system is an important aspect to be ensured during the initial development process of the safety system.

Maintenance of a system is easier if the system is intelligible. Usage of design patterns can improve intelligibility through common vocabulary. If design patterns are used in system development and documentation [1], the maintenance team can more easily understand the system concepts and execute maintenance operations to the system. Naturally this requires that both the developer and the maintenance team know and understand the used patterns. This, in practice, requires either company's internal patterns or widely adopted patterns related to the domain.

VI. CHALLENGES IN DESIGN PATTERN USAGE IN SAFETY SYSTEM ENGINEERING

Patterns have qualities that justify their usage in the development of safety systems. However, some challenging issues can be identified as well. To provide ample insight into patterns the issues of patterns are discussed in this section.

Patterns are not exact. As mentioned, patterns (typically) describe solution on a relatively high abstraction level so that they can be used and implemented in multiple ways. In safety system development exactness and completeness are considered virtues that patterns can, but often do not, provide.

Usage of patterns may lead to *inconsistent understanding* between system developers. A pattern can be implemented in many ways and each person has a unique mindset about a pattern. Thus patterns are not applicable as safety documentation as such. However, when a set of patterns has been used extensively, the patterns may become a part of a communication language that clarifies the ideas shared between individuals [1] and thus may act as a supporting form of documentation.

A developer may *misunderstand pattern solutions* or use them in contexts not suitable for the pattern. A similar issue is naturally related to all situations when documented solutions are applied. One can also misunderstand solutions illustrated in a book, journal article or data of a preceding project.

Patterns are not meant to be detailed illustrations of the solution (though some patterns indeed illustrate details). Instead, they typically provide a *generic framework of the solution*, which the designer can apply in the environment in which the problem is considered. This is one of the strengths of patterns, but it is also a potential issue. A pattern author may have accidentally or intentionally left out some information that would be needed to be fully able to consider all the side-effects of the pattern.

If a pattern reader is *unfamiliar with the domain* the patterns consider, an incorrect overall picture could be adopted. Though patterns consider various aspects of the solution, they cannot take into account all the relevant aspects. In the domain of safety system engineering artefacts relate to each other in complex manners. A single pattern cannot consider all these aspects as it would shift the focus of the pattern. Thus the reader should regard patterns with a healthy sense of criticism when they are applied.

Patterns may encourage designers to *stick with existing solutions*. Often the reuse of solutions is a productive way to go and well-tried solutions are valuable in the field of safety system engineering. However, this should not mean that reuse of solutions is the only way to go. New, more efficient, simpler, and better approaches cannot be developed if old solutions are constantly used. It has to be identified if the design benefits from the reuse of solutions and when one needs to focus on creating a better, novel approach to the problem in hand.

Copyright (c) IARIA, 2013. ISBN: 978-1-61208-268-4

VII. FUTURE WORK

Our future effort is expansion of our safety system development related pattern collection [5] and development of tool support for a semantic search of patterns. The target of the pattern collection expansion is to construct a pattern language that could serve safety system developers. Another aspect is to study the effects of pattern usage in practical development processes. Empirical studies on pattern usage in the development processes of safety systems would provide insight into widening the usage of patterns.

The semantic search for patterns eases pattern discovery. Semantic relations between pattern data are being developed. This enables the search of patterns supported with a semantic deduction engine to identify patterns with similar features and consequences as given in the original search.

VIII. CONCLUSION

In this article, we have illustrated rationale for using design patterns in the development of safety systems. The foundation of usage of patterns lies in the idea of providing a way to document tacit and existing knowledge into an explicit format. When experience is formatted as a design pattern, it can become common knowledge that can serve in documentation, implementation and communication purposes.

Safety systems are parts of critical systems that are able to cause physical damage. The sole purpose of a safety system is to prevent the hazardous situations leading to physical damage. Well-tried solutions and approaches documented in patterns can help in the development of a dependable and cost-effective safety system. Development of safety systems is heavily regulated by standards and legislation, which require methods, techniques and processes to be used, but provide few practical solutions. With design patterns practical solutions can be documented into an intelligible format while providing room for modifiability.

Cooperation between a control and a safety system can prove to be a burdensome task especially if it is made from scratch. This may occur in larger control system projects for large scale unique plants. In such cases patterns provide a valuable engineering resource as they describe solution and approaches on an abstract level. This enables a designer to apply the approach in a suitable way considering the system.

Design patterns also have some drawbacks in context of safety system development. They are not exact and accepted as documentation or proof of compliance. Still patterns can help to improve development process and share knowledge.

REFERENCES

[1] D. Riehle, "Lessons Learned from Using Design Patterns in Industry Projects," in Transactions on Pattern Languages of Programming II, vol. 6510, J. Noble, R. Johnson, P. Avgeriou, N. Harrison, and U. Zdun, Eds. Springer Berlin / Heidelberg, 2011, pp. 1-15.

[2] C. Alexander, S. Ishikawa, and M. Silverstein, A pattern language: Towns, buildings, construction. New York: Oxford University Press, 1977.

[3] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design patterns, Elements of reusable object-oriented software. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc, 1995.

[4] J. Koskinen, M. Vuori, and M. Katara, "Safety Process Patterns: Demystifying Safety Standards," Proc. 2012 IEEE International Conference on Software Science, Technology and Engineering (SWSTE), IEEE Computer Society, 2012, pp. 63-71.

[5] J. Rauhamäki, T. Vepsäläinen, and S. Kuikka, "Functional Safety System Patterns," Proc. VikingPLoP 2012 Conference, 2012, pp. 48-68, http://URN.fi/URN:ISBN:978-952-15-2944-3 [retrieved: February, 2013].

[6] E. Freeman, E. Freeman, B. Bates, and K. Sierra, Head first design patterns. O' Reilly & Associates, Inc, 2004.

[7] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, Pattern-oriented software architecture, volume 1: A system of patterns. Chichester, UK: Wiley, 1996.

[8] D. C. Schmidt, M. Stal, H. Rohnert, and F. Buschmann, Pattern-oriented software architecture: Patterns for concurrent and networked objects. 2nd ed.,New York, NY, USA: John Wiley & Sons, Inc, 2000.

[9] M. Fowler, Patterns of enterprise application architecture. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc, 2002.

[10] G. Hohpe and B. Woolf, Enterprise integration patterns: Designing, building, and deploying messaging solutions. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc, 2003.

[11] T. Erl, SOA design patterns. 1st ed.,Upper Saddle River, NJ, USA: Prentice Hall PTR, 2009.

[12] R. S. Hanmer, Patterns for fault tolerant software. Chichester, England ; Hoboken, NJ: John Wiley, 2007.

[13] A. Armoush, Design Patterns for Safety-Critical Embedded Systems. 2010.

[14] B. P. Douglass, Doing hard time: Developing real-time systems with UML, objects, frameworks, and patterns. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc, 1999.

[15] V. Eloranta, J. Koskinen, M. Leppänen, and V. Reijonen, A Pattern Language for Distributed Machine Control Systems. 2010. http://practise.cs.tut.fi/project.php?project=sulake [retrieved: February, 2013].

[16] International Electrotechnical Commission, Functional safety of electrical/electronic/programmable electronic safety-related systems, Part 1: General requirements, IEC, 2010.

[17] European Parliament and of the Council, Directive 2006/42/EC of the european parliament and of the council, vol. L 157/24, 2006.

[18] International Electrotechnical Commission, Functional safety of electrical/electronic/programmable electronic safety-related systems, IEC, 2010.

[19] European Committee for Standardization, Safety of machinery, safety-related parts of control systems, Part 1: General principles for design, 2008.

[20] International Electrotechnical Commission, Functional safety of electrical/electronic/programmable electronic safety-related systems, Part 3: Software requirements, IEC, 2010.

[21] M. Kairaila, Domain-Specific Template-Based Visual Language and Tools for Automation Industry. 2010.

Copyright (c) IARIA, 2013. ISBN: 978-1-61208-268-4