A Pattern-Based Method for Safe Control Systems Exemplified within Nuclear Power Production

André Alexandersen Hauge[1,3] and Ketil Stølen[2,3]

[1] Department of Software Engineering,
Institute for Energy Technology, Halden, Norway
`andre.hauge@hrp.no`
[2] Department of Networked Systems and Services,
SINTEF ICT, Oslo, Norway
`ketil.stolen@sintef.no`
[3] Department of Informatics, University of Oslo, Norway

Abstract. This article exemplifies the application of a pattern-based method, called SaCS (Safe Control Systems), on a case taken from the nuclear domain. The method is supported by a pattern language and provides guidance on the development of design concepts for safety critical systems. The SaCS language offers six different kinds of basic patterns as well as operators for composition.

Keywords: conceptual design, pattern language, development process, safety.

1 Introduction

This article presents a pattern-based method, referred to as SaCS (Safe Control Systems), facilitating development of conceptual designs for safety critical systems. Intended users of SaCS are system developers, safety engineers and HW/SW engineers.

The method interleaves three main activities each of which is divided into sub-activities:

S *Pattern Selection* – The purpose of this activity is to support the conception of a design with respect to a given development case by: a) selecting SaCS patterns for requirement elicitation; b) selecting SaCS patterns for establishing design basis; c) selecting SaCS patterns for establishing safety case.

C *Pattern Composition* – The purpose of this activity is to specify the intended use of the selected patterns by: a) specifying composite patterns; b) specifying composition of composite patterns.

I *Pattern Instantiation* – The purpose of this activity is to instantiate the composite pattern specification by: a) selecting pattern instantiation order; and b) conducting step wise instantiation.

A safety critical system design may be evaluated as suitable and sufficiently safe for its intended purpose only when the necessary evidence supporting this claim has been established. Evidence in the context of safety critical systems development is the documented results of the different process assurance and product assurance activities performed during development. The SaCS method offers six kinds of basic patterns categorised according to two development perspectives: *Process Assurance*; and *Product Assurance*. Both perspectives details patterns according to three aspects: *Requirement*; *Solution*; and *Safety Case*. Each basic pattern contains an instantiation rule that may be used for assessing whether a result is an instantiation of a pattern. A graphical notation is used to explicitly detail a pattern composition and may be used to assess whether a conceptual design is an instantiation of a pattern composition.

To the best of our knowledge, there exists no other pattern-based method that combines diverse kinds of patterns into compositions like SaCS. The supporting language is inspired by classical pattern language literature (e.g. [1,2,3]); the patterns are defined based on safety domain needs as expressed in international safety standards and guidelines (e.g. [6,9]); the graphical notation is inspired by languages for system modelling (e.g. [10]).

This article describes the SaCS method, and its supporting language, in an example-driven manner based on a case taken from the nuclear domain.

The remainder of this article is structured as follows: Section 2 outlines our hypothesis and main prediction. Section 3 describes the nuclear case. Section 4 exemplifies how functional requirements are elicited. Section 5 exemplifies how a design basis is established. Section 6 exemplifies how safety requirements are elicited. Section 7 exemplifies how a safety case is established. Section 8 exemplifies how intermediate pattern compositions are combined into an overall pattern composition. Section 9 concludes.

2 Success Criteria

Success is evaluated based on satisfaction of predictions. The hypothesis (H) and predictions (P) for the application of SaCS is defined below.

H: The SaCS method facilitates effective and efficient development of conceptual designs that are: 1) consistent; 2) comprehensible; 3) reusable; and 4) implementable.

Definition. *A conceptual design is as a triple consisting of: a specification of requirements; a specification of design; and a specification of a safety case. The safety case characterises a strategy for demonstrating that the design is safe with respect to safety requirements.*

We deduce the following prediction from the hypothesis with respect to the application of SaCS on the case described in Section 3:

P: Application of the SaCS method on the load following case described in Section 3 results in a conceptual design that uniquely characterises the load following case and is easily instantiated from a composite SaCS pattern. Furthermore, the conceptual design: 1) is consistent; 2) is expressed in a manner that is easily understood; 3) may be easily extended or detailed; 4) is easy to implement.

Definition. *A conceptual design instantiates a SaCS composite pattern if: each element of the triple can be instantiated from the SaCS composite pattern according to the instantiation rules of the individual patterns and according to the rules for composition.*

3 The Case: Load Following Mode Control

In France, approximately 75% of the total electricity production is generated by nuclear power which requires the ability to scale production according to demand. This is called load following [8]. The electricity production generated by a PWR (Pressurised Water Reactor) [8] is typically controlled using:

- *Control rods*: Control rods are inserted into the core, leading to the control rods absorbing neutrons and thereby reducing the fission process.
- *Coolant as moderator*: Boron acid is added to the primary cooling water, leading to the coolant absorbing neutrons and thereby reducing the fission process.

Control rods may be efficiently used to adjust reactivity in the core; several percentage change in effect may be experienced within minutes as the core will react immediately upon insertion or retraction. When using boron acid as moderator there is a time delay of several hours to reach destined reactivity level; reversing the process requires filtering out the boron from the moderator which is a slow and costly process.

When using Boron acid as moderator, fuel is consumed evenly in the reactor as the coolant circulates in the core. When using the control rods as moderator, the fuel is consumed unevenly in the reactor as the control rods are inserted at specific sections of the core and normally would not be fully inserted.

A successful introduction of load following mode control requires satisfying the following goals:

G1 *Produce according to demand*: assure high manoeuvrability so that production may be easily scaled and assure precision by compensating for fuel burn up.
G2 *Cost optimisation*: assure optimal balance of control means with respect to cost associated with the use of boron acid versus control rods.
G3 *Fuel utilisation*: assure optimal fuel utilisation.

The SaCS method is applied for deriving an adaptable load following mode control system intended as an upgrade of an existing nuclear power plant control system. The adaptable feature is introduced as a means to calibrate the controller performing control rod control during operation in order to accommodate fuel burn up. The system will be referred to as *ALF* (Adaptable Load Following). The scope is limited to goal *G1* only.

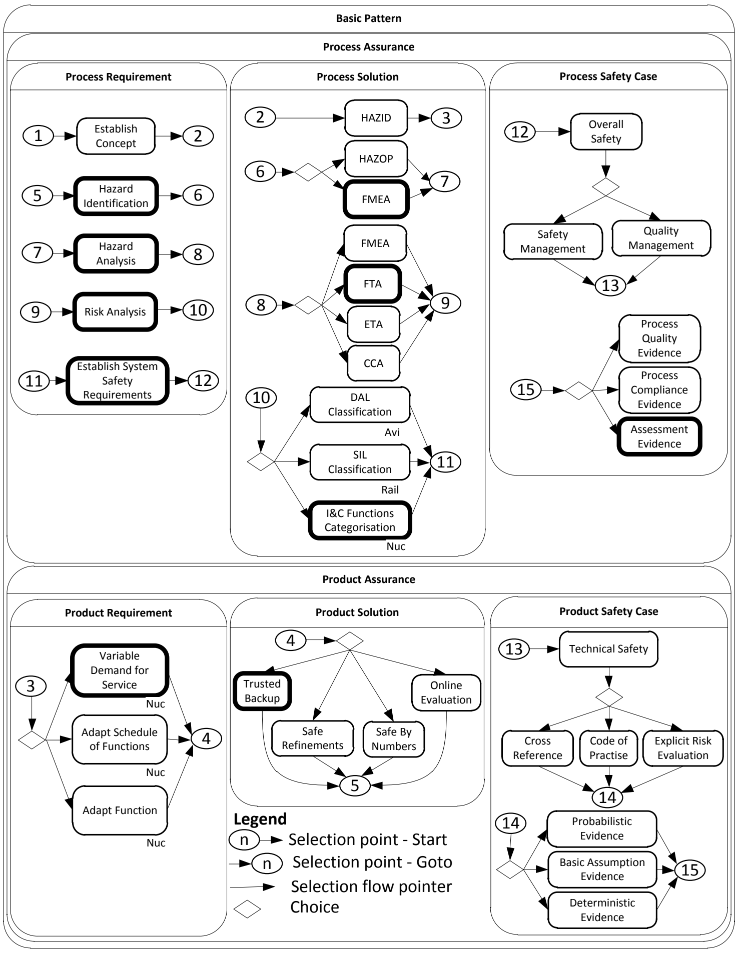

Fig. 1. Pattern Selection Activity Map

4 Elicit Functional Requirements

4.1 Pattern Selection

The selection of SaCS basic patterns is performed by the use of the pattern selection map illustrated in Figure 1[1]. Selection starts at selection point (1). Arrows provide the direction of flow through the selection map. Pattern selection ends when all selection points have been explored. A *choice* specifies alternatives where more than one alternative may be chosen. The patterns emphasized with a thick line in Figure 1 are used in this article.

[1] Not all patterns in Figure 1 are yet available in the SaCS language but has been indicated for illustration purpose.

The labelled frames in Figure 1 represent selection activities denoted as UML [10] activity diagrams. The hierarchy of selection activities may also be used as an indication of the categorisation of patterns into types. All patterns that may be selected is of type *Basic Pattern* (indicated by the outermost frame). The type *Basic Pattern* is specialised into two pattern types: *Process Assurance*; and *Product Assurance*. These two are both specialised into three pattern types: *Requirement*; *Solution*; and *Safety Case*. The *Solution* type within *Process Assurance* is for patterns on methods supporting the process of developing the product. The *Solution* type within *Product Assurance* is for patterns on design of the product to be developed.

All patterns indicated in Figure 1 should be understood as generally applicable unless otherwise specified. General patterns represent domain independent and thus common safety practices. Domain specific patterns are annotated by a tag below the pattern reference. In Figure 1, the tag: "Nuc" is short for nuclear; "Avi" short for aviation; and "Rail" short for railway. Domain specific patterns reflect practices that are dependent on domain.

In selection point (3) of Figure 1 a set of product assurance requirement patterns may be reached. We assume in this article that the information provided in Section 3 sufficiently details the development objectives and context such that the patterns reached from selection point (1) and (2) may be passed. The pattern *Variable Demand for Service* reached from selection point (3) captures the problem of specifying requirements for a system that shall accommodate changes arising in a nuclear power production environment. The pattern is regarded as suitable for elicitation of requirements related to the goal *G1* (see Section 3).

4.2 Pattern Instantiation

The pattern *Variable Demand for Service* referred to in Figure 1 is a product oriented requirement pattern. Pattern descriptions is not given in this article (see [5] for the full details) but an excerpt of the pattern is given in Figure 2.

Figure 2 defines a parametrised problem frame annotated by a SaCS adapted version of the problem frames notation [7]. It provides the analyst with a means for elaborating upon the problem of change in a nuclear power production environment in order to derive requirements (represented by *Req*) for the system under construction (represented by *Machine*) that control a plant (represented by *Plant*) such that a given objective (represented by *Obj*) is fulfilled.

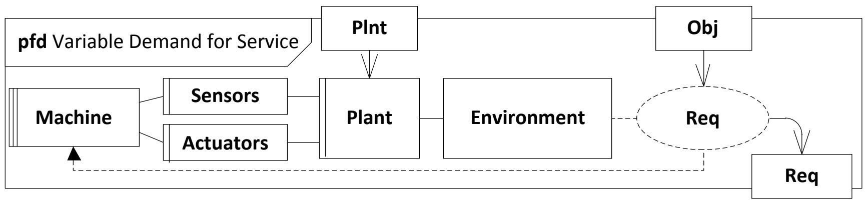

Fig. 2. Excerpt (simplified) of "Variable Demand for Service" Pattern

When *Variable Demand for Service* is instantiated, the *Req* artefact indicated in Figure 2 is produced with respect to the context given by *Obj* and *Plnt*. In Section 4.1 we selected the pattern as support for eliciting requirements for a *PWR* system upgrade with respect to goal *G1*. The parameter *Obj* is then bound to *G1*, and the parameter *Plnt* is bound to the specification of the *PWR* system that the *ALF* upgrade is meant for.

Assume that the instantiation of *Variable Demand for Service* according to its instantiation rule provides a set of requirements where one of these is defined as: *"FR.1: ALF system shall activate calibration of the control rod patterns when the need to calibrate is indicated"*.

4.3 Pattern Composition

Figure 3 illustrates a *Composite Pattern Diagram* that is a means for a user to specify a composite pattern. A composite pattern describes an intended use, or the integration of, a set of patterns. A specific pattern is referred to by a *Pattern Reference*. A pattern reference is illustrated by an oval shape with two compartments. The letter "R" given in the lower compartment denotes that this is a requirement pattern; the prefix "Nuc-" indicates that this is a pattern for the nuclear domain. A solid-drawn oval line indicates a product assurance pattern. A dotted-drawn oval line indicates a process assurance pattern.

A small square on the oval represent a *Port*. A port is used to represent a connection point to *Pattern Artefacts*. A socket represents *Required Pattern Artefact* and the lollipop represents *Provided Pattern Artefact*. Patterns are integrated by the use of *Combinators*. A combinator (e.g. the solid-drawn lines annotated with "delegates" in Figure 3) specifies a relationship between two patterns in terms of a pattern matching of the content of two *Artefact Lists*, one bound to a source pattern and one bound to a target pattern. An artefact list is an ordered list of *Pattern Artefacts* (A user may give names to lists).

Figure 3 specifies that the *Variable Demand for Service* pattern delegates its parameters to the *Functional Requirements* composite. The binding of parameter *Obj* and *Plnt* to the informal information provided on goal *G1* and the *PWR* specification is denoted by two *Comment* elements. A comment is drawn similar to a comment in UML [10].

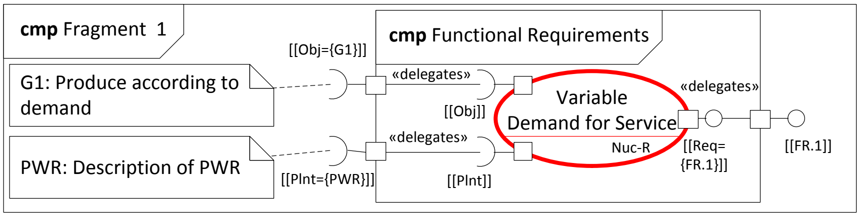

Fig. 3. Fragment showing use of "Functional Requirements" composite

5 Establish Design Basis

5.1 Pattern Selection

In selection point (4) of Figure 1, a set of alternative design patterns may be selected. All design patterns describe adaptable control concepts. The patterns differ in how adaptable control is approached and how negative effects due to potential erroneous adaptation are mitigated.

The *Trusted Backup* pattern describes a system concept where an adaptable controller may operate freely in a delimited operational state space. Safety is assured by a redundant non-adaptable controller that operates in a broader state space and in parallel with the adaptable controller. Control privileges are granted by a control delegator to the most suitable controller at any given time on the basis of switching rules and information from safety monitoring.

The *Trusted Backup* is selected as design basis for the ALF system on the basis of an evaluation of the strengths and weaknesses of the different design patterns with respect to functional requirements, e.g. *FR.1* (see Section 4.2).

5.2 Pattern Instantiation

Requirements may be associated with the system described by the *Trusted Backup* pattern. No excerpt of the pattern is provided here due to space restrictions (fully described in [5]). Assume a design specification identified as *ALF Dgn* is provided upon instantiation of *Trusted Backup* according to its instantiation rule. The design specification describes the structure and behaviour of the *ALF* system and consists of component diagrams and sequence diagrams specified in UML as well as textual descriptions.

5.3 Pattern Composition

The referenced basic pattern *Trusted Backup* in Figure 4 is contained in a composite named *Design*. Requirements may be associated with the system (denoted *S* for short) described by the pattern *Trusted Backup*. In SaCS this is done by associating requirements (here *FR.1*) to the respective artefact (here *S*) as illustrated in Figure 4.

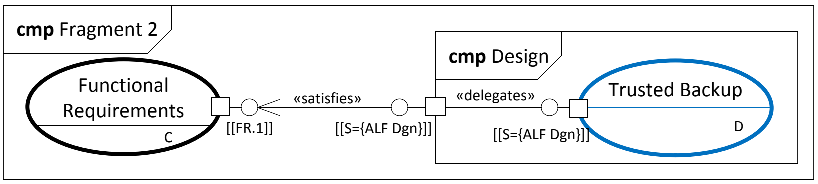

Fig. 4. Fragment showing use of "Design" composite

The *satisfies* combinator in Figure 4 indicates that *ALF Dgn* (that is the instantiation of *S*) satisfies the requirement *FR.1* provided as output from instantiation of the *Functional Requirements* composite. The *Functional Requirements* composite is detailed in Figure 3. A pattern reference to a composite is indicated by the letter "C" in the lower compartment of a solid-drawn oval line.

6 Elicit Safety Requirements

6.1 Pattern Selection

Once a design is selected in selection point (4) of Figure 1, further traversal leads to selection point (5) and the pattern *Hazard Identification.* This pattern defines the process of identifying potential hazards and may be used to identify hazards associated with the ALF system.

In selection point (6), a set of method patterns supporting hazard identification are provided. The *FMEA* pattern is selected under the assumption that a FMEA (Failure Modes Effects Analysis) is suitable for identifying potential failure modes of the ALF system and hazards associates with these.

Once a hazard identification method is decided, further traversal leads to selection point (7) and *Hazard Analysis.* In selection point (8) process solution patterns supporting hazard analysis may be selected. The *FTA* is selected as support for *Hazard Analysis* under the assumption that a top-down FTA (Fault Tree Analysis) assessment is a suitable complement to the bottom-up assessment provided by FMEA.

Selection point (9) leads to the pattern *Risk Analysis.* The pattern provides guidance on how to address identified hazards with respect to their potential severity and likelihood and establish a notion of risk.

In selection point (10), domain specific patterns capturing different methods for criticality classification are indicated. The *I&C Functions Classification* is selected as the ALF system is developed within a nuclear context.

In selection point (11) the pattern *Establish System Safety Requirements* is reached. The pattern describes the process of eliciting requirements on the basis of identified risks.

6.2 Pattern Instantiation

Safety requirements are defined on the basis of risk assessment. The process requirement patterns selected in Section 6.1 support the process of eliciting safety requirements and may be applied subsequently in the following order:

1. *Hazard Identification* – used to identify hazards.
2. *Hazard Analysis* – used to identify potential causes of hazards.
3. *Risk Analysis* – used for addressing hazards with respect to their severity and likelihood of occurring combined into a notion of risk.
4. *Establish System Safety Requirements* – used for defining requirements on the basis of identified risks.

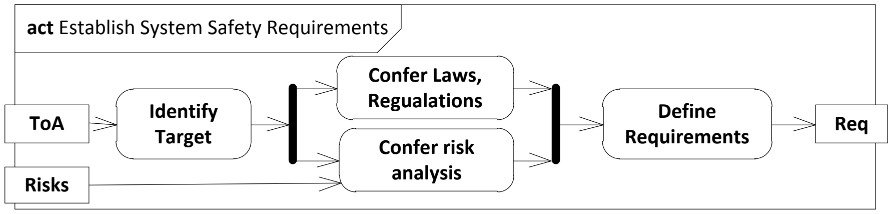
act Establish System Safety Requirements
ToA
Risks
Identify Target
Confer Laws, Regualations
Confer risk analysis
Define Requirements
Req

Fig. 5. Excerpt (simplified) of "Establish System Safety Requirements" pattern

Assume that when *Risk analysis* is instantiated on the basis of inputs provided by the instantiation of its successors, the following risk is identified: *"R.1: Erroneously adapted control function"*. The different process requirement patterns follow the same format; details on how they are instantiated are only given with respect to the pattern *Establish System Safety Requirements.*

Figure 5 is an excerpt of *Establish System Safety Requirements* pattern. It describes a UML activity diagram with some SaCS specific annotations. The pattern provides the analyst a means for elaborating upon the problem of establishing safety requirements (represented by *Req*) based on inputs on the risks (represented by *Risks*) associated with a given target (represented by *ToA*).

Assume that *Establish System Safety Requirements* is instantiated with the parameter *Risks* bound to the risk *R.1*, and the parameter *ToA* bound to the *ALF Dgn* design (see Section 5.2). The instantiation according to the instantiation rule of the pattern might then give the safety requirements: *"SR.1: ALF shall disable the adaptive controller during the time period when controller parameters are configured"* and *"SR.2: ALF shall assure that configured parameters are correctly modified before enabling adaptable control"*.

6.3 Pattern Composition

The composite *Identify Risk* illustrated in Figure 6 is not detailed here but it may be assumed to provide data on risks by the use of the patterns *Hazard Identification*, *Hazard Analysis* and *Risk Analysis* as outlined in Sections 6.1 and 6.2. The pattern *I&C Functions Categorisation* is supposed to reflect the method for risk classification used within a nuclear context as defined in [6].

Semantics associated with the *address* combinator assures that the parameter *ToA* of *Establish System Safety Requirements* is inherited from *ToA* of the pattern *Identify Risk.*

7 Establish Safety Case

7.1 Pattern Selection

From selection point (12) in Figure 1 and onwards, patterns supporting a safety demonstration is provided. We select *Assessment Evidence*, reached from selection point (15), as support for deriving a safety case demonstrating that the safety requirements *SR.1* and *SR.2* (defined in Section 6.2) are satisfied.

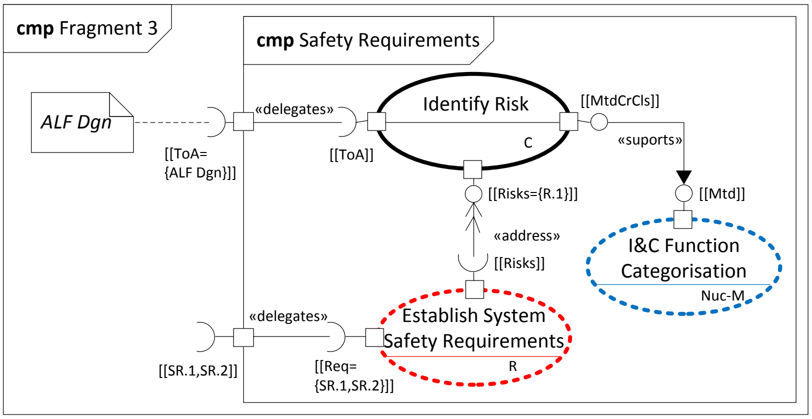

Fig. 6. Fragment showing use of "Safety Requirements" composite

7.2 Pattern Instantiation

Figure 7 represents an excerpt (fully described in [5]) of the pattern *Assessment Evidence* and defines a parametrised argument structure annotated by a SaCS adapted version of the GSN notation [4]. When *Assessment Evidence* is instantiated a safety case is produced, represented by the output *Case*. The parameters of the argument structure is bound such that the target of demonstration is set by *ToD*, the condition that is argued satisfied is set by *Cond*. The argument structure decomposes an overall claim via sub-claims down to evidences. The FMEA assessment identified as *ALF FMEA* of the design *ALF Dgn* performed during the assessment phase described in Section 6 provides a suitable evidence that may be bound to the evidence obligation *Ev*.

7.3 Pattern Composition

Figure 8 specifies that the *Assessment Evidence* pattern delegates its parameters to the *Safety Case* composite. The binding of parameters *ToD*, *Cond*, and *Ev* is set informally by three comment elements referencing the respective artefacts that shall be interpreted as the assignments. Instantiation of the pattern provides the safety case artefact identified as *ALF Case*.

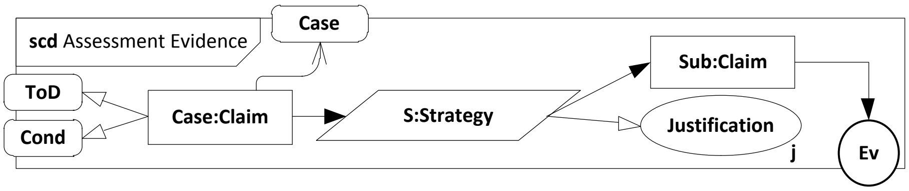

Fig. 7. Excerpt (simplified) of "Assessment Evidence" pattern

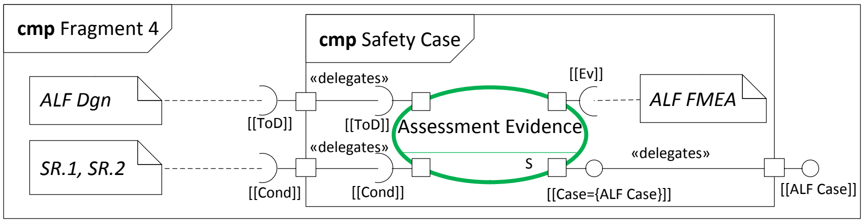

Fig. 8. Fragment showing use of "Safety Case" composite

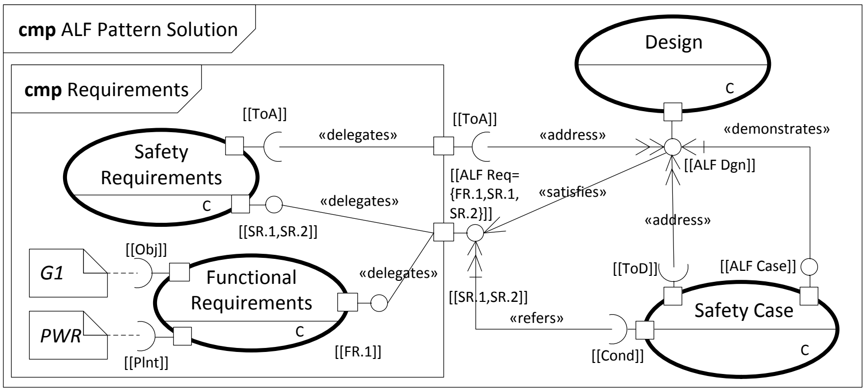

Fig. 9. The "ALF Pattern Solution" composite

8 Combine Fragments

The composite *ALF Pattern Solution* illustrated in Figure 9 specifies how the different composite patterns defined in the previous sections are combined.

Figure 9 specifies the patterns used; the artefacts provided as a result of pattern instantiation; the relationships between patterns and pattern artefacts by the use of operators. The composite specification of Figure 9 may be refined by successive steps of the SaCS method, e.g. by extending the different constituent composite patterns with respect to the goals *G2-G3* of Section 3.

9 Conclusions

In this paper we have exemplified the application of the SaCS-method on the load following mode control application.

We claim that the conceptual design is easily instantiated from several SaCS basic patterns within a case specific SaCS composite (Figure 9). Each basic pattern has clearly defined inputs and outputs and provides guidance on instantiation through defined instantiation rules. Combination of instantiation results from several patterns is defined by composition operators. The conceptual design is built systematically in manageable steps (exemplified in Section 4 to Section 8) by instantiating pieces (basic patterns) of the whole (composite pattern) and merge results. The conceptual design (fully described in [5]) is consistent with definition, the required triple is provided by the artefacts *ALF Req, ALF Dgn,* and *ALF Case* (as indicated in Figure 9) and uniquely specifies the load following case.

Future work includes expanding the set of basic patterns, detailing the syntax of the pattern language and evaluation of the SaCS-method on further cases from other domains.

Acknowledgments. This work has been conducted and funded within the OECD Halden Reactor Project, Institute for energy technology (IFE), Halden, Norway.

References

1. Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., Angel, S.: A Pattern Language: Towns, Buildings, Construction. Oxford University Press (1977)
2. Buschmann, F., Henney, K., Schmidt, D.C.: Pattern-Oriented Software Architecture: On Patterns and Pattern Languages, vol. 5. Wiley (2007)
3. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley (1995)
4. GSN Working Group: GSN Community Standard, version 1.0 (2011)
5. Hauge, A.A.: Stølen, K.: A Pattern Based Method for Safe Control Conceptualisation Exemplified Within Nuclear Power Production, HWR-1029, Institute for energy technology, OECD Halden Reactor Project, Halden, Norway (to appear)
6. IEC: Nuclear Power Plants – Instrumentation and Control Important to Safety – Classification of Instrumentation and Control Functions. IEC-61226, International Electrotechnical Commission (2009)
7. Jackson, M.: Problem Frames: Analyzing and Structuring Software Development Problems. Addison-Wesley (2001)
8. Lokhov, A.: Technical and Economic Aspects of Load Following with Nuclear Power Plants. Nuclear Development Division, OECD NEA (2011)
9. The Commission of the European Communities: Commission Regulation (EC) No 352/2009 on the Adoption of Common Safety Method on Risk Evaluation and Assessment, 352/2009/EC (2009)
10. Object Management Group: Unified Modeling Language Specification, version 2.4.1 (2011)