Bernard, Alain; Coatanea, Eric; Christophe, François; Laroche, Florent

Design

Published in:
Procedia CIRP

DOI:
10.1016/j.procir.2014.06.146

Published: 01/01/2014

Document Version
Publisher's PDF, also known as Version of record

Please cite the original version:
Design: a key stage of product lifecycle

Alain Bernarda*, Eric Coataneab, François Christopheb, Florent Larochea

a Ecole Centrale de Nantes, IRCCyN UMR CNRS 6597, 1, rue de la Noë, BP 92101 – 44321 Nantes Cedex 3 - France
b Aalto University School of Engineering, P.O. Box 14100 – 00076 Aalto – Finland

* Corresponding author. Tel.: + 33 2 40 37 68 08; fax: +33 2 40 37 69 30. E-mail address: alain.bernard@irccyn.ec-nantes.fr

Abstract

Design appears to be a key and critical stage of product lifecycle. Different models have been introduced in previous research to describe the conceptual design process. The RFBS model is one of them extending and deepening the existing FBS models. A previous paper was presenting the model and assuming the possible execution of the process tasks automatically. The present paper provides an overview of the progresses that have been made in this direction during the past years. The model-driven engineering philosophy underlying the RFBS model of knowledge is concretely exemplified in this paper. The implementation through ontology and language such as SysML that was part of the model-driven engineering philosophy is concretely described in this paper in form of computer-aided tools dedicated to the conceptual design stages.

1. Introduction

As stated by many researchers, conceptual design decisions are constraining more than 80% of the final cost and performances of a system along its lifecycle [4], [42]. In fact, design processes are time constrained [29, 21] and do not follow in practice a purely iterative process [35, 12] but instead a trade-off between iteration needs and stage gate approach is required for fulfilling the time constrains [36, 39]. Consequently, a poorly conceived artefact cannot be saved by the excellence of the later phases of the design activity. In addition, variety is increasing [11] [15] and need of information management is more and more important. In such context, and in close relation with integrated methods of product development [3], it is necessary to develop generic and adaptable conceptual models to face these issues.

The RFBS model [8] is providing a way to focus on the conceptual design stages and it adds the concept of requirement to the models of Gero or Umeda [16] [36] [37] [17] but it is also proposing to go beyond the guidelines and methodologies for conceptual design proposed in the design and engineering design communities [19] [20] [25] [33] [34] by proposing a model-based vision of the system development process. The RFBS model is covering areas not fully covered by systems engineering standards such as IEEE 1220-1998 [30] or ANSI/EIA-632-1999 [27].

Firstly, this paper presents an overview of some existing models for knowledge representation of conceptual design. Secondly, the stages that have been supported in the RFBS model during the past years are presented through the use of input models and generation of output models based on the semi-formal language SysML [31]. Thirdly, the necessary future developments are discussed and the future work program is introduced.

2. State-of-the-art of knowledge representation

The present section presents a brief overview of different viewpoints about knowledge representation in engineering.

2.1. Dialectal thinking

The design activity is an activity where iteration is playing a fundamental role. From an initial imperfect solution, new iterations are proposed over time to improve design performances. The progresses are made through a dialectal
thinking process commonly used in science. The dialectical thinking is a response to conflicting or contradictory objectives that need to be associated for answering a problem.

The key feature of the dialectical reasoning is the principle of integration, starting with recognition of contradiction, followed by reconciliation of opposing perspectives [26].

This dialectical and historical perspective is a central element in the vision developed in TRIZ by Altshuller [1] and it is not a surprise knowing the history of Altshuller in Soviet Union. In contrast with other methods of solving problems, TRIZ emphasizes the contradictions and recommends solving them instead of making the usual engineering trade-offs. TRIZ is especially interesting because its literature provides an analysis of the types of contradictions and the nature of problems. Savransky in his introduction to TRIZ methodology [28] distinguished different problem structures and different types of contradictions namely administrative, technical and physical contradictions.

2.2. Inferential design theory

Developed by Arciszewski and Michalski in 1994, the inferential design theory (IDT) proposes a framework for the integration of multiple conceptual design methods [2]. The main idea of IDT is to propose logical operators on the knowledge parts contained in the system’s knowledge base. These operators are called transmutations. IDT defines eleven pairs of transmutations, each pair being composed of opposite transmutations.

This theory is very interesting due to the combination of knowledge parts that it proposes. To a certain extent, this combination implements a degree of creativity in the system. Furthermore, IDT is claimed to be a unified framework due to its multi-strategic aspect. Nevertheless, this theory does not explain the type of necessary knowledge at the conceptual design stage. A very similar approach has been proposed later and it seems independently by Hatchuel et al. [18] to represent the design process. The principles are similar and share intriguing similarities with the work of Arciszewski and Michalski.

2.3. Function-Behaviour-Structure

Gero proposed the FBS model in 1990 [16] in parallel with Umeda and his colleagues proposed also a FBS model [37] [38]. Both models are sharing strong similarities but they diverge on the interpretation of the S letter. In the Gero’s model, the S means Structure when in the Umeda and al. model it means State.

The model of Umeda and al. has been associated with an expert system and it is probably the first attempt to build such type of model-based system. The model of Gero is presented in figure 1 and each of its processes is described in detail in [16].

Since then, this model has been used also as a reference for building expert systems based on the triplet of knowledge with modifications, such as Tian’s TFBBS [32]. The original FBS model of Gero has evolved with time as Gero and Kannengiesser proposed a version of it situated within the creative environment in 2002 [17]. Researchers have also adapted FBS in order to fit different scopes, such as, for example, Labrousse et al. who provided FBS-PPRE (Function / Behaviour / Structure – Process / Product / Resource / External Effects) modelled for managing company knowledge [22].

This overview has presented the bases of the present research. The next chapter is presenting the RFBS model used as the central knowledge representation framework for conceptual design in the present article. As stated before taking this vision of the engineering process at early stages is not excluding the use of complementary perspectives presented in this brief state of the art.

3. The RFBS model

Gero’s FBS model contains few aspects which needed to be updated for a usage of model-driven engineering in the context of Conceptual design. Apart from the fact that this previous model did not include a requirement phase, a major point needs to be reanalysed in the synthesis phase of conceptual design. In fact, Gero stated that the only possible link between function and structure was through the expression of behaviour. The authors have argued in a previous article and demonstrated through a software proof of concept that it was possible to create “embryos” of structures out of functions only, thanks to semantics [9].

They call these preliminary structures generic or abstract structures. As abstract classes in object-oriented modelling, their aim is only to encapsulate each atomic function of the system into one or more of the six families of organs from our previous works [10] [8]. In the RFBS model, presented in figure 2, the model-based approach starts with the transformation of the needs into a model of requirements.

This model of requirements contains several elements described in standards from System Engineering such as the standard ANSI/EIA-632-1999 [27]. Different classes of
requirements are described and have to be populated. In military context different standards exist such as the MODAF and DoDAF frameworks [41]. Those frameworks integrate different categories of requirements. For example the following viewpoints are present: operational, capability, services and system requirements.

The RFBS diagram is separated into stages, processes (or transformation phases) and reformulation (or iteration) processes. A mapping of each RFBS stages with a representation in SysML is provided below and a more detailed exploration of the mapping can be obtained from Christophe et al. [8] [9]. For example, different representation stages can use the following SysML diagrams:

- **R (Requirements)**: requirements and use case diagrams
- **F (Functions)**: represented with internal block diagram (black-box model: function-flow)
- **Be (Expected behaviour)**: activity, sequence and state-machine diagrams
- **Bs (Behaviour of the system)**: parametric and same diagrams as **Be**
- **GS (Generic Structure)**: class diagram (with abstract classes)
- **S (Structure)**: class, components and deployment diagrams

Nevertheless, a specific profile of the language had to be created in order to enable the representation of functional decompositions and architectures [5].

4. Contribution to the integration of RFBS with analysis and simulation tools

4.1. Support tool for stages R

This stage **R (Requirements)** of the RFBS model consists in the extraction, capture, elicitation of requirements from the various stakeholders involved in the project. During that phase, documents written in natural language (NL) are often reviewed among other material. These documents can be large in size and tedious to read as for instance ISO standards documents. It is then a difficult task even for experts to extract requirements from such type of document. The support tool developed for this stage aims at systematically extracting requirements from NL documents and to provide experts with a model of the requirements extracted and their interactions.

The software tool takes a SysML profile describing a classification of requirements types as input along with a textual document (in this case as pdf format) and returns a SysML model of the requirements extracted. The SysML profiles can take the forms shown as example in Figure 3.

The process handled by this support tool contains three main steps: extraction of requirements, clustering of requirements according to profile and assignment of relations between requirements. Each of these steps is described more in details in the following sub-sections.

Extraction of requirement sentences

Based on the information collected from the Stanford parser on each sentence from the document, sentences containing modal verbs are extracted and considered as requirements.

This extraction technique was tested on two cases of different scales from European Standard IEC 61508-3:2010 (Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 3: Software
requirements) and on one case based on a Finnish official document (Finnish Security and Defence Policy 2012*).

The first case consists in testing this extraction on one page of this standard (i.e. introduction page). In this first case, 24 sentences were listed on that page of which 6 sentences were considered as requirements.

The second case consists in full scale testing and the entire document was analysed. From this test, 848 sentences were recognised as well-formed of which 259 contained modal verbs. It is to be noted that the extraction during the full scale test did not take more than 5 minutes for 103 pages covered.

In the third case, pages 99 to 114 were analysed. In these pages, 407 sentences were considered of which 121 requirements were found, as shown in Figure 4. The following figures and examples of this software application are based on this third case study.

During that phase, each requirement sentence extracted from the document is compared with the categories of requirements given from the SysML profile shown in Figure 5. This comparison is realized with a word for word search from the requirement sentence corresponding to a word from one or more of the profile categories. Basically, if one word from the requirement sentence considered matches a word from a category then this requirement is assigned to this category. In this phase, we deliberately assume that one requirement sentence can belong to several categories as it is often the case practically and requirements should not be omitted when sending sub-sets of requirements for example to subcontractors. This categorization of requirements could also be achieved with the use of Latent Semantic Analysis [13], more precisely Singular Value Decomposition, but this pragmatic approach shows similar results by avoiding calculation times.

This clustering phase is important in this approach because it enables giving precedence, i.e. a direction, and a formal type to the relations between requirement sentences.

Assigning relations between requirements

During this phase of the approach, requirements are compared pair-wise for their similarity.

During this phase, each requirement sentence extracted from the document is compared with the categories of requirements given from the SysML profile shown in Figure 5. This comparison is realized with a word for word search from the requirement sentence corresponding to a word from one or more of the profile categories. Basically, if one word from the requirement sentence considered matches a word from a category then this requirement is assigned to this category. In this phase, we deliberately assume that one requirement sentence can belong to several categories as it is often the case practically and requirements should not be omitted when sending sub-sets of requirements for example to subcontractors. This categorization of requirements could also be achieved with the use of Latent Semantic Analysis [13], more precisely Singular Value Decomposition, but this pragmatic approach shows similar results by avoiding calculation times.

This clustering phase is important in this approach because it enables giving precedence, i.e. a direction, and a formal type to the relations between requirement sentences.

Assigning relations between requirements

During this phase of the approach, requirements are compared pair-wise for their similarity.

Assigning relations between requirements

During this phase, each requirement sentence extracted from the document is compared with the categories of requirements given from the SysML profile shown in Figure 5. This comparison is realized with a word for word search from the requirement sentence corresponding to a word from one or more of the profile categories. Basically, if one word from the requirement sentence considered matches a word from a category then this requirement is assigned to this category. In this phase, we deliberately assume that one requirement sentence can belong to several categories as it is often the case practically and requirements should not be omitted when sending sub-sets of requirements for example to subcontractors. This categorization of requirements could also be achieved with the use of Latent Semantic Analysis [13], more precisely Singular Value Decomposition, but this pragmatic approach shows similar results by avoiding calculation times.

This clustering phase is important in this approach because it enables giving precedence, i.e. a direction, and a formal type to the relations between requirement sentences.

Assigning relations between requirements

During this phase of the approach, requirements are compared pair-wise for their similarity.

Assigning relations between requirements

During this phase of the approach, requirements are compared pair-wise for their similarity.

Assigning relations between requirements

During this phase of the approach, requirements are compared pair-wise for their similarity.

Assigning relations between requirements

During this phase of the approach, requirements are compared pair-wise for their similarity.

Assigning relations between requirements

During this phase of the approach, requirements are compared pair-wise for their similarity.
dependency relationship directed from the longest requirement to the shortest.

4.2. Support tool for stages Bs and Be

This section presents a software tool developed for supporting the analysis of the stages related to Behaviour, Be and Bs, in the RFBS model. A causal ordering graph of the studied system (e.g. an air bearing) is presented in Figure 6. The causal ordering was obtained by a group work composed of experts. The experts selected an initial set of 17 variables and 23 cause-effect relations between them using a support approach presented by Coatanéa and al. [10]. This graph has been modelled using a parametric diagram from the SysML language modelled using Topcased† modeller, based on Eclipse Modelling Framework [14]. Each design variable was represented in a block containing its dimensional representation in form of a composition of mass (M), length (L), and time (T). Each cause-effect relation is represented by a dependency relation indicating a client depending on a master resource.

The XMI format of this SysML model is used as input for the software support tool presented in this article. Figure 7 shows the transformation of the representation in Figure 6 into a more traditional causal graph of the system variables.

The Directed Acyclic Graph of Figure 7 is clustered using a colouring approach described in Coatanéa et al. [10]. This colouring leads to Figure 8 where performance variables are presented in red, design variables in green (variables easier to control) and in blue (variables harder to control), and exogenous variables in black (variables outside the system).

Four variables, the, air bearing stiffness (K), the lifting force (Fc), the output airflow (F0) and the output pressure (Pc) have been classified by the algorithm in the performance category.

The following step consists in defining qualitative objectives for those four performance variables. Then, these objectives can be back propagated from these nodes to the rest of the network. The propagation algorithm is not described in this article.

The contradicting nodes resulting from the propagation mechanism are represented in yellow in Figure 9. For example the injection nozzle diameter d should be simultaneously as small as possible and also as big as possible.

This approach is in line with the dialectal thinking approach described for Knowledge representation in section 2.1. This type of support tool can be used at conceptual stage to analyse expected behaviour Be and behaviour resulting from the structure of the system (Bs).

4. Conclusion

This paper relates new propositions to improve one of the most critical phases of product lifecycle, Design, and

† http://www.topcased.org/
especially, Conceptual design. More precisely, the approach developed in this article presents the concrete developments made in order to implement computer-aided tools for different stages of the RFBS model.

As conclusion, let us summarize the stages of the RFBS model so far covered by software support tools developed by the authors.

In previous work [9], it was shown that the synthesis process can be partially automated and that computers can suggest preliminary solutions to the designing team (F → GS process).

This paper first presented the development achieved for refining Requirements written in Natural Language into a more formal representation; therefore this support tool is a strong assistant for representing R and during the process of functional analysis (R → F process).

The second achievement presented in this paper corresponds to the effort of representing behaviours Be and Bs in a formal and common manner; in that sense, this development provides the necessary ground for the evaluation process (Be ↔ Bs process). Moreover, the causal ordering and analysis of contradictions proposed with this support tool provide some support for the extraction of Be from R for one (R → Be process) and for the discovery of main design issues for the latter (Be → S).

Nevertheless, the present article presents a causal ordering obtained by a group of experts applying manually the method described in the article. An automatic causal ordering algorithm has also been developed and is currently being tested. The types of contradictions addressed directly in the case study of the article are the physical contradictions which are the ones visible at the most detailed level of representation of system architecture.

This summary shows the state of software support for conceptual design and enables a vision on the developments to be achieved as future work. At this point, it is noticeable that the focus has been placed on guiding and providing strong mathematical support for designing teams’ decision during the conceptual design process. Second, however, it is to be noticed that the synthesis process (GS → S process) still relies on the expertise and know-how of the designing team. In this work, the authors have tried to demonstrate that TRIZ, graph theory and dimensional analysis can form parts of a coherent holistic approach supporting the design process.

References