A comparative case study of functional models to support system architecture design

Sonia Ben Hamidaabc, Antoine Grandoub, Marija Jankovicb, Claudia Eckertd, Alain Hueta, Jean-Claude Bocquetb *

aAIRBUS Defence & Space, Les Mureaux, France
bEcole Centrale Paris, Chatenay-Malabry, France
cInstitut d Recherche Technologique SystemX, Palaiseau, France
dOpen University, Milton Keynes, England

Abstract

Methods supporting designers’ cognition in early stages are critical. Function modelling in that can be important because it allows understanding the design perimeter as well as communication between engineers. Moreover, in complex system design, no one designer or design team is able to explore and define the design perimeter alone. In order to understand the advantages as well as the limits of modeling approaches we have chosen three approaches Function-Behavior-State (FBS) [1], Functional representation to support idea generation [3] and Affordances [2] and we look at them also in relation with a process on problem definition of complex systems used in industry [3]. Some of the advantages as well as needs for future developments in the case of complex system design are discussed.

1 Introduction

During system architecture the arrangement of functional elements and their mapping to physical components needs to be defined [4]. Therefore, it is critical to identify and describe a complete set of functions for the products to define the scope of the product and communicate between different experts in an organization. While many different

* Corresponding author. Tel.: +3-362-377-9329.
E-mail address: sonia.benhamida@outlook.com
functional modelling approaches exist, their uptake in industry has been limited and designers find it difficult to think about functions in a consistent and coherent way [5,6].

In this paper, we investigate different approaches to assess their usability for complex system design and in particular system architecture design. We selected four approaches: Function-Behavior-State (FBS) [1], Functional representation to support idea generation [3], Problem definition process [3], and Affordances [2]. These approaches have been regarded with their diversity in mind: 1) FBS as a somewhat standard approach that is process oriented and modelling different states, 2) Functional representation to support idea generation because the idea is also to support idea generation in function modelling in complex system design and 3) Affordances because the approach has never been applied in this company. Models of an intelligent refrigerator for each approach illustrate the approaches and discuss possible advantages and challenges in an industrial setting.

This paper is organized in 6 sections. The section 2 of the paper addresses the state of the art on function modelling, system architecture design and provides initial analysis of potential challenges and difficulties. In section 3, we give an overview of the case study used and detail our methodology to select the four approaches and apply them. Section 4 presents the models developed for each approach. In section 5, we discuss the advantages and the limits and in section 6 we give some preliminary conclusions.

2 Background

2.1 Function Modelling

Functions allow designers to translate needs into physical structures that will meet those needs, and function modelling is a formal way to define and model functions. The term function is used by engineers with many different meanings which hampers the use of functional description [7]. Umeda [1] states “there is no clear and uniform definition of a function. And moreover, it seems impossible to describe function objectively”. Several literature reviews on functional modelling show a diversity of approaches. Erden [8] identifies eighteen different function modelling approaches highlighting the underlining ambiguity related to functional description in engineering. Szykman [9] states “a single designer or design team can no longer manage the complete product development effort”; function modelling provides a framework for overall system description. Muller [10] developed a scheme representing the design process aiming at showing the gap between the high-level requirements and the low level details. Erden [8] completes the scheme stating that “function modelling serves as a mean of linking the upper and lower levels of system design and description”. Chandrasekaran and Josephson [11] identify two viewpoints to describe a function. The environment-centric view considers function as its effects. The device-centric view considers function in term of internal parameters of the object. The difference between those two views resides in the level of abstraction that they consider, and also their stage in the design process. The device-centric functions are the outcome of the deployment of the environment centric functions. Crilly [12] highlights the necessity of these two views to think of system’s, in the context of nested systems. Tomiyama [5] identifies four major purposes of functional descriptions: (1) To represent the purpose of the artefact, (2) To explain the behavior, structure, (3) To capture customer requirements and (4) To illustrate the overview of the system. Functional description exists in various engineering contexts, such as Requirement descriptions, Systems architecture and Value engineering. However the concept of function modelling appears to be difficult to apply in industry environment [5,6,13]. Tomiyama [5] focuses on this lack of practicality and identifies three syndromes: (1) “Never used it”: Practitioners have never received any formal training beyond “transformational boxes” or “to do something” verb noun pairs. (2) “No added value”: Practitioners stay at the same level of knowledge and detail in function modelling without going deeper. (3) “Not practical”: Practitioners think that function modelling tools are only for academics since they were developed by them. Facing this lack of use in industry, Eckert [6] proposes to focus on user-centered tools. Engineers need methods with immediate benefits, easy to follow with clear and intuitive training materials, show; also the outcomes of the models should be easy to explain to others.
2.2 System Architecture

Urlich [4] defines System Architecture as “(1) the arrangement of functional elements; (2) the mapping from functional elements to physical components; (3) the specification of the interfaces among interacting physical components”. For Crawley [14], the system architecture is “the embodiment of concept and the allocation of physical/informational function to elements of form, and definition of interfaces among elements and with the surrounding context”. Eppinger and Browning [15] define system architecture as “the structure of the system, embodied in its elements, their relationships to each other (and to the system’s environment), and the principles guiding its design and evolution — that give rise to its functions and behaviors”. Hence the functional architecture represents one facet of the system architecture and links system design intentions to the physical world.

2.3 Challenges of using functional models in Systems Architecture design

The importance and utility of using functional description can be seen through (this is by no means an exhaustive list): 1) the understanding gained through the process of the model building, 2) the results gained through building the model and 3) the knowledge that is captured in the model. The way designers analyze a product is strongly influenced by their notion of function. However, there are several difficulties in using the function modelling. First, it is very hard to identify the distance between the model and the system that is designed. It is very hard to identify if elements have been missed and if all perimeter is covered. Moreover, models are produced from a particular viewpoint for a particular purpose. Without these being explicit, the user of a model does not understand the rationale of the person who built the model. For example whether unwanted functions (e.g. vibration) are included in a functional model depends on the view of function. Models can be generated at different levels of abstraction, which influences the range of objects they denote.

3 Case study: Design of an intelligent refrigerator

In order to investigate the potential applicability of the four approaches in an industry environment, we used a case study of a next-generation refrigerator [16]. The idea is to design refrigerator with the aim of going in the direction of connected objects and collecting customer data. Therefore, the objectives of this design is to define new services that will enhance customer satisfaction but also use the data collected to enhance user experience. Although the illustration case study can be considered as simple, the discussions and tests have been also conducted with regard to complex system design within industry context.

3.1 Selection of the four modelling approaches

Erden [6] compares eighteen function modelling approaches through six main domains. The ontology, the semantic definition of function, the function representation formalism, the function–context relation, the decomposition and verification and the implementation in a programming environment and application. However, these domains do not take into account industry concerns in terms of practicality and usefulness. We analyzed all the eighteen approaches with the following practical questions in mind: Is the approach well explained and illustrated in order to reproduce it? Is the function modelling representation defined? Is the approach supported by a tool? How much is the approach promising regarding industry’s challenges such as systems complexity, reusability, design innovation, etc.? These questions have been decided with 2 senior experts in system architecture design. Moreover, a diversity of approaches was also of importance. We selected two approaches from Erden’s review with different ontology and definition of function: The Function-Behavior-State approach, because it is the only process-centered approach. And the Functional representation to support idea generation approach because it takes into account existing solutions. We also wanted to apply a more classical industrial approach. We selected a Problem definition process [3] which illustrates concrete applications and building on activities and techniques described for example in NASA handbook [17] and the Defense Acquisition Guidebook [18], in order to engineer complex systems and systems of systems. Finally, we selected the
affordance-based design approach because it incorporates the user directly in the modelling approach rather than mitigated through a set of explicit requirements [19]. It is a user-centric approach, “in which the needs, wants, and limitations of end users of a product are given extensive attention at each stage of the design process” [20].

4  Function and affordance modelling approaches: (1) Problem definition process, (2) Function-Behavior-State, (3) Functional representation to support idea generation, (4) Affordances

4.1  Problem definition process

The Problem definition process [3] has been selected as one of the reference processes used in industry. The system engineer first focuses on the system, its purpose and mission, its perimeter and its context. Then the system lifecycle is analyzed from its conception to its retirement. A complete analysis of the system’s stakeholders is then conducted to see what actors the system involves. A working framework is created, followed by different methods and tools to gather information from the stakeholders of the system. Later the model of the system context is realized to understand and define the system’s goals. Then modelling the domain and defining stakeholders’ requirements and constraints allows to express service requirements, functional interface requirements, and physical interfaces.

Therefore, here we present just one step of the process. In Figure 1: Problem definition process in the case study of intelligent refrigerator, the main missions of the “intelligent refrigerator” are listed and different interactions of the system. After identifying different stakeholders and using Ishikawa diagram to identify major problem through interviews and data collection, a definition of service requirements is used to concentrate this data (Figure 1).

![Figure 1: Problem definition process in the case study of intelligent refrigerator](image)

4.2  Function-Behavior-State

Shimomura [1], in the Function-Behavior-State approach, defines the terms function as the “description of behavior abstracted by human through recognition of the behavior for utilization”, behavior as the “sequential changes of state”,

and state as the “combination of entities, attributes of entities and relations among entities”. Shimomura introduces four types of relations among functions. The first relation type decomposed-into decomposes a function into sub-functions. The second type conditioned-by is a causal relationship: The first function will not occur if the behavior linked to the second function is not observed. The third type enhanced-by adds a function in order to fulfil of function modifier, where a function modifier is an adverb or a group of words to give more information about the function and the expected behavior. During the evaluation of a function, if the designer finds out that a function modifier is not fully satisfied by a function, he will add a new function to the first function in order to improve the level of satisfaction. Finally the relation type described-as details function modifiers. In this case study, “To allow food conversation” function has been identified with 4 different key performance indicator KPIs (Economically, Comfortably, Efficiently, Quickly). This function is further decomposed (marked as “D” relationship) into three sub functions. This process is repeated until we are able to map function to structure (blue relationships in the diagram). In this case, two enhancing-functions are identified to improve comfort and time saving.

![Function-Behavior-State approach for the case of an intelligent fridge](image)

**4.3 Functional representation to support idea generation**

The design of familiar products, as categorized by Summers [21], does not require function level design, leading to less usage of function modelling. To benefit from knowledge of existing solutions, Chakrabarti [22] proposes to represent structures and solutions in terms of provided and required functions to ensure the generation of solutions to a problem. First, the design problem is defined by a set of functions to fulfil. A function is the “description of the
action or effect required by a design problem, or that is supplied by a solution”. A set of partial alternative solutions is synthesized using the knowledge base of solutions in order to fulfil each function. That partial alternative set of solutions is evaluated with regard to the whole problem to see if all functions have been fulfilled. If not, the problem is revised considering the remaining functions, until all the functions are fulfilled.

With regard to the main function of the refrigerator “To preserve and store food” and with regard to the main Key Performance Indicators (KPIs), we identified the Cooling system as partly covering the problem in terms of cost and comfort. This is indicated in such a way that on the left side functions and KPIs are defined and on the right side potential solutions that can entirely or partially satisfy a given function. In this process, the KPIs are marked in each step (Input KPI and Output KPI) to ensure that the functions can be satisfied.

Figure 3: Functional representation to support idea generation for the case of an intelligent fridge.
4.4 Affordances

Maier [23] defines the affordance as “a behavior that can be exhibited when the two subsystems interact”. In order to develop a relational model of design, Maier [24] defines three entities: the designer of the artifact, the artifact being designed and the user of the artifact. And two relationships: The designer designs that artifact, the user uses the artifact. Maier [25–27] identifies two different categories of affordances, when the interacting subsystems are an artifact or a user: (1) The “artifact-user affordance expresses an interactive relationship between an artifact and a user where a behavior may occur between the artifact and user that neither the artifact nor the user could manifest alone”. (2) The “artifact-artifact affordance is a potential behavior that may be exhibited by the two artifacts together, that could not be manifested by either artifact alone”. Developing the work of Maier [27], Cormier [2] details the classification of the user categories, the artifact-artifact relationships and the artifact-user relationships, and formalizes an affordance-based method for capturing users’ needs. Cormier uses the concept of Desired Affordance Model (DAM) which is a structure for organizing users, artifacts and affordances. It enables engineers to proceed from the identification and development of general requirements to move on specification, following four steps defined by Maier [27]: Step 1: Understanding, Gathering and Expressing User Needs in term of Affordances. Step 2: Applying Generic Affordance Structure Template. Step 3: Prioritizing Affordances. Step 4: Organizing the Affordances into a Structure. Cormier [2] defines a standard and specific vocabulary for describing the different stakeholders. We first identified the needs linked to the KPI: Afford low energy consumption, Afford low economic cost, Afford low food waste, Afford comfort of use, Afford time saving. With regard to this methodology Figure 4 shows only the “Food consumption” part identifying the 2 types of affordances. For each user type, i.e. the owner, the food consumer, the repairman, the food buyer, we modeled the desired affordance of the refrigerator. Overall, 33 affordances have been identified.

Figure 4: Affordance modelling of an intelligent fridge.
4.5 Criteria for evaluation

To analyze the strengths and weaknesses of the four modelling approaches, we asked an aerospace company to provide us criteria to assess system modelling tools. Among the 29 criteria, we selected 8 of them which focus on functions and are tool-independent. We compared them to the criteria defined by Summers [21] in its benchmarking standard protocol. We also looked at the criteria defined by Wu [19] to compare function-based design and affordance-based design. We selected the following criteria, which cover previous industrial criteria, as shown in Table 1 Mapping of industry criteria to Summers' ones:

Table 1 Mapping of industry criteria to Summers' ones

<table>
<thead>
<tr>
<th>Industry’s criteria</th>
<th>Summers’ criteria [21]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refine operational scenario until elementary operation and function</td>
<td>Scalability, Premature commitment, Construction</td>
</tr>
<tr>
<td>Specifies functional interface between elementary functions</td>
<td>Construction</td>
</tr>
<tr>
<td>Specifies behavior of elementary function</td>
<td>Behavior</td>
</tr>
<tr>
<td>Map Elementary function and flow on physical items and links</td>
<td>Consistency</td>
</tr>
<tr>
<td>Verification capabilities (consistency checks)</td>
<td>Consistency</td>
</tr>
<tr>
<td>Simulate simple behavior</td>
<td>Behavior</td>
</tr>
<tr>
<td>Readability of models (easiness to review, communicate and present to non-specialist)</td>
<td>Visibility</td>
</tr>
<tr>
<td>Ease of use/Ergonomics</td>
<td>Flexibility, Closeness of mapping, Error-proneness</td>
</tr>
</tbody>
</table>

With regard to these criteria, an assessment of these four approaches has been made (see table 2). Approaches are represented in columns and in lines different criteria that have been discussed. For instance, for the problem definition approach we identified this is positive point (“+”) because this approaches does not focus on a specific function modeling approach and several approaches can be used with regard to different project contexts.

Table 2 Comparison of modelling approaches

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Ontology</td>
<td>Device Centered</td>
<td>Process Centered</td>
<td>Device Centered</td>
<td>Environment Centered</td>
</tr>
<tr>
<td>Flexibility ability to modify and adapt the representation to address new problems</td>
<td>+ Different representations can be used. Depends on tool used.</td>
<td>- Each new problem needs to reconsider the whole model. - no filtering available</td>
<td>+ identification of several alternatives solutions + possibility to reuse solutions for new problems</td>
<td>- User oriented modelling. No generic models.</td>
</tr>
<tr>
<td>Consistency enforce physics and other consistency</td>
<td>+ mapping with physical architecture in next steps of the process</td>
<td>+ top-down-bottom-up approach</td>
<td>- not possible to assess</td>
<td>- not available</td>
</tr>
<tr>
<td>Behavior ability of the representation to simulate behavior</td>
<td>+ Simulation of functional sequences with Function-Flow Block Diagram</td>
<td>+ System states, and use of states techniques to simulate behavior. - tool KIEF but no usable versions found</td>
<td>+ possibility to identify KPI related to different sub-systems - not support for causal diagrams between physical parameters</td>
<td>+ description of expected and unexpected behaviors - no possibility to simulate behavior, static</td>
</tr>
<tr>
<td>Scalability support both simple and complex problem types</td>
<td>+ tested on complex systems. See [3]</td>
<td>- all-in-one model representation, model will be too huge to manage</td>
<td>- for complex systems, does not allow to analyze interactions between sub-systems</td>
<td>- As more complex systems are considered, models can become large quickly [2]</td>
</tr>
</tbody>
</table>
Construction Does function modeling support different types of construction approaches?  

<table>
<thead>
<tr>
<th>+ Forward and backward chaining, outside to inside and inside to outside</th>
<th>- No forward and backward chaining + outside to inside (decomposition)</th>
<th>+ Forward chaining (KPI) - no backward chaining, - outside to inside and inside to outside</th>
</tr>
</thead>
</table>

Closeness of mapping What modeling conventions needs to be learned? How intuitive is the resulting model?  

<table>
<thead>
<tr>
<th>- Need to learn Function-Flow Block Diagram notation. + Intuitive model representing functions, flows and sequencing.</th>
<th>- need to learn the different types of relations and the representation of entities and relationships + simple modelling notation</th>
<th>+ intuitive mapping</th>
</tr>
</thead>
</table>

Error-proneness Does the design of the notation induce ‘careless mistakes’?  

<table>
<thead>
<tr>
<th>+ Several tools available allow exploitation - there might be a problem of interoperability.</th>
<th>- easy keystroke errors for the relationship type, no visual distinction between different types of relationships</th>
<th>- No tool tested. Everything is error-prone</th>
</tr>
</thead>
</table>

Premature commitment Do the model require decisions before they have the information needed is available?  

<table>
<thead>
<tr>
<th>- Choice of the high-level function induces early commitment. - top-down approach requires to early define system’s boundaries</th>
<th>- Choice of the high-level function induces early commitment.</th>
<th>+ does allow investigating several alternatives</th>
</tr>
</thead>
</table>

Visibility How easy is it to see all aspects of the model? Can two models be compared?  

<table>
<thead>
<tr>
<th>+ decomposition - tools do not allow to compare two models. Comparison is not straightforward</th>
<th>- not straightforward layout to ease the comparison, - need to check object per object</th>
<th>- KPI input-output might be difficult to compare</th>
</tr>
</thead>
</table>

Visibility How easy is it to see all aspects of the model? Can two models be compared?  

<table>
<thead>
<tr>
<th>+ decomposition - tools do not allow to compare two models. Comparison is not straightforward</th>
<th>- not straightforward layout to ease the comparison, - need to check object per object</th>
<th>- KPI input-output might be difficult to compare</th>
</tr>
</thead>
</table>

5 Discussion

The advantage of the Problem definition process is that it is a defined step-by-step systematic method. However, function modelling occurs in the concept development phase after the problem clarification where other tools are used such as Ishikawa diagrams, or users’ stories.

In the Function-Behavior-State approach, we appreciated to be able to think of functions and their performances (called function-modifier) at the same time. Moreover, the relationship “enhanced by” justifies the introduction of functions enhancing the performance. Another advantage is to discover functions through a top-down–bottom-up approach, i.e. going back and forth between functional, behavioral, and structural domains. However, the readability of the model is affected by representing all the information on the same model. It would be useful to zoom-in on a decomposed function. And finally, the method does not seem to support functional alternatives management.

As for the Functional representation approach, it focuses on the similarities of the problem to tackle with existing systems. The designer focuses on what is not fulfilled, and what makes the problem different from existing solutions. However, to what extent is this approach applicable to complex systems, with an increased number of components and interactions? The analysis of interactions among the selected existing systems does not seem well supported.

When using affordances, we were able to think both of functional and non-functional needs. Cormier’s templates and checklist guided our discovery and capture of users’ needs in a structured manner. The checklist lists a broad range of relationships involving for example spatial or environmental interactions, as well as subjective feelings like comfort, perception and aesthetics. Affordance-based design allows to express complex relationships. We focused our attention on what will be provided to the user. However the vocabulary defined by Cormier needs to be tailored to the industry’s sector, e.g. space, defense, to make it more relevant. Moreover, the transition to the physical architecture is
not always obvious. Also, the lack of tool makes it difficult to use as the number of affordances grows rapidly. Finally, affordance-based design is weakly integrated to industrial development processes.

The three function modelling approaches are use-centric. They are good problem-solving tools focusing “on the goals and tasks associated with the use of the artifact, rather than on the end user” [28]. They only meet the user’s functional needs. On the other hand, affordance-based design also covers functional and non-functional needs. Function and affordance are complementary. Future work is needed to integrate them [29]. Ciavola [30] initiates a framework to use in conjunction functions and affordances, linking high-level affordance information with low-level function information.

6 Conclusion

Using functional description and function modelling is essential in the design process. It allows reflecting upon the design perimeter, constraints, knowledge capitalization etc. However, building and using functional models is not straightforward in early design stages where discovery of functional alternatives and reuse previous development is at stake. In this work, we have identified four approaches with regard to their applicability in industry. We discussed the merit and limits of each of them, and the complementary use of these modelling methods. Additional difficulty is given when considering these approaches in complex system design environment and in designing system architecture. There is a need for somewhat standardized approach allowing data and knowledge sharing, but also certain flexibility to be able to adapt to different design phases and design contexts. For example there is a need for a function modeling on the system level and on the subsystem level. In general, we discuss this as traceability problem. However, these tasks are often done by different teams and sometimes different companies. The question is how can we share and communicate around function modeling and what tools can be used to support these processes in industry environment. Moreover, how this modeling can be integrated in system architecture design process as it is one of its major activities.

Acknowledgements

We would like to thank AIRBUS Group Innovations and AIRBUS Defence and Space for their support and feedback in relation with this study.

References


