Module Functional View

logo
TBD

Gerrit Muller
Buskerud University College
Frogs vei 41 P.O. Box 235, NO-3603 Kongsberg Norway
gaudisite@gmail.com

Abstract

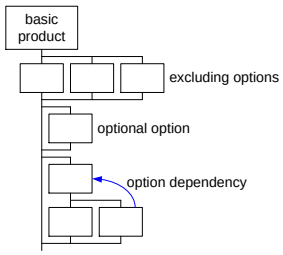
This module addresses the Functional View.

version: 0 status: draft June 23, 2016

Contents

Chapter 1

The functional view



1.1 Introduction

The functional view describes the **what** of the system, or in other words: how is the system perceived from the outside, especially by the customer. The product specification (or requirement specification[1]) covers normally the content of this view. The content of these specs should be observable from outside of the system.

Several methods and models can be used in this view. (Use) Cases, section 1.2, describing the system from user point of view. Commercial, service and goods flow decompositions, section 1.3, describing the product in terms of the commercial packages and options and the other logistics dimensions. Function and feature specifications, section 1.4, focusing on a more functional view or a feature wise view. Performance specification and models, section 1.5, describing performance aspects such as throughput and latency, as a function of the commercial options and the available parameter space.

The information model, described in section 1.6 is especially important when interfacing with other systems. Section 1.7 describes the role of standards in the product specification.

[1] Or any combination of the words: system, product, functional, performance, requirement and specification

1.2 Case descriptions

Use cases are an useful means to describe desired functional behavior in a more cohesive way. An use case describes a set of functions together in typical, worst case or exceptional scenarios. Use cases become really effective if the use case is not limited to the functional behavior, but when the non-functional aspects are described as well.

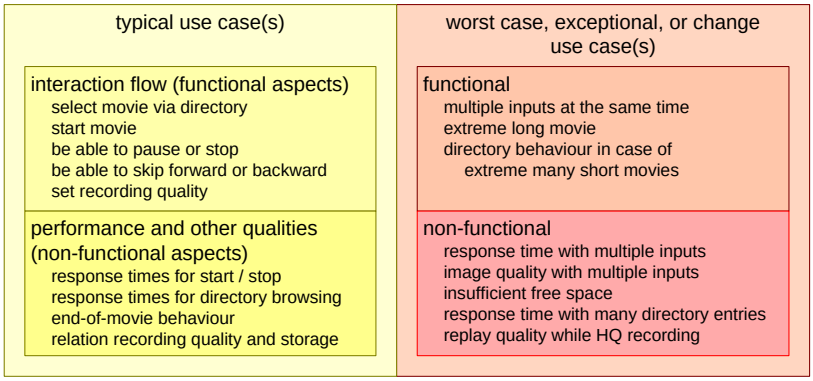


Figure 1.1: Example personal video recorder use case contents for typical use case and worst case or exceptional use case

Figure 1.1 shows the possible content for personal video recorder use cases. The most typical use is to watch movies: find the desired movie and play it. Additional features are the possibility to pause or stop and to skip forward or backward. The use case description itself should describe exactly the required functionality. The required non-functional aspects, such as performance, reliability and exceptional behavior must be described as well.

Typical use cases describe the core requirements of the products. The boundaries of the product must be described as well. These boundaries can be simply specified (maximum amount of video stored is 20 hours standard quality or 10 hours high definition quality) or a set of *worst case* use cases can be used. *Worst case* use cases are especially useful if the boundaries are rather situational dependent, the circumstances can be described in the use case.

The exceptional use case are comparable to the worst case use cases. Exceptions can be described directly (if insufficient storage space is available the recording stops and a message is displayed). Here *exception* use cases are helpful if the exception and the desired exceptional behavior are dependent on the circumstances.

Figure 1.2 summarizes recommendations for working with use cases. Many use case descriptions suffer from fragmentation: every function is described as a separate use case. The overview is lost, and the interaction of functions is missed. The granularity of use cases should match with the external use.

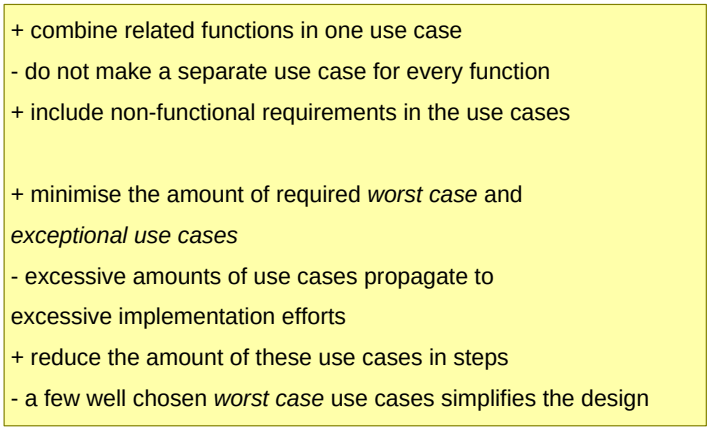


Figure 1.2: Recommendations for working with use cases

Another problem is that too many use cases are described, again with the consequence of losing the overview and worse spending too much time at not relevant specification issues. The problem is that up front the knowledge is insufficient to select the most relevant use cases. A somewhat more extensive exploration phase is recommended, where afterwards a reduction of use cases is applied.

1.3 Commercial, service and goods flow decomposition

The commercial granularity of sellable features and the allowed configurations can be visualized in a commercial configuration tree, as shown in figure 1.3. All items in such a tree will appear in brochures, folders, catalogues. Note that the commercial granularity is often somewhat more coarse than the design decomposition. The commercial packaging is optimized to enable the sales process and to the margin. In some businesses the highest margin is in the add-ons, the accessories. In that case the add-ons are not part of the standard product to protect the margin.

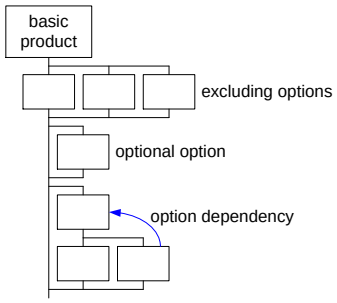


Figure 1.3: Commercial tree as means to describe commercial available variations and packaging

The commercial tree makes clear what the relations between commercial options are. Options might be exclusive (either this printer or that printer can be connected, not both at the same time). Options can also be dependent on other options (High definition video requires the memory extension to be present. The decomposition model chosen is a commercial decision, at least as long as the technical implications are feasible and acceptable in cost.

The same strategy can be used to define and visualize the decompositions needed for service (customer support, maintenance) and goods flow (ordering, storage and manufacturing of goods). Figure 1.4 shows the decompositions with their main decomposition drivers. These decompositions are not identical, but they are related. The goods flow decomposition must support the commercial as well as the service decomposition. The goods flow decomposition has a big impact on the costs side of the goods flow (goods=costs!) and must be sufficiently optimized for cost efficiency.

The service decomposition is driven by the need to maintain systems efficient, which often means that minimal parts should be replaced. The granularity of the service decomposition is finer than the commercial decomposition.

The goods flow decomposition, which needs to support both the commercial as well as the service decomposition, has a finer granularity than both these decom-

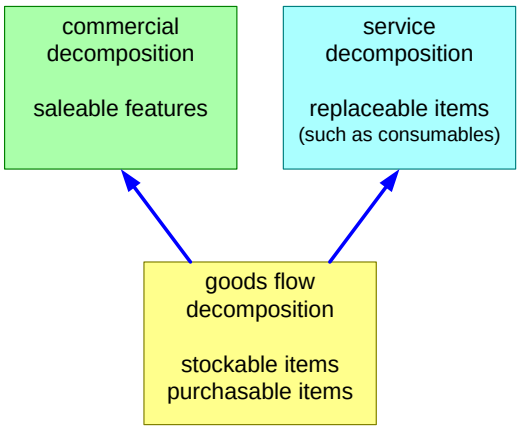


Figure 1.4: Logistic decompositions for a product

positions. At the input side is the goods flow decomposition determined by the granularity of the supply chain.

In Philips all three decompositions used to fit in the so-called 12NC system, a logistics identification scheme deployed in the *Technical Product Documentation (TPD)*. The TPD is the formal output of the product creation process. These decompositions are used in logistics information systems (MRP).

1.4 Function and feature specifications

The product specification needs to define the functions and features of the product. The decomposition for this description is again another decomposition than the commercial decomposition. The commercial decomposition is too coarse to use it as basis for the product specification. The technical decomposition in functions and features is kind of a building box to compose commercial products and packages.

technical functions	*products* home cinema system	flat screen cinema TV	bedroom TV
HD display	+	+	-
SD->HD up conversion	+	+	-
HD->SD down conversion	+	+	o
HD storage	o	-	-
SD storage	o	-	o
HD IQ improvement	+	+	-
SD IQ improvement	+	+	+
HD digital input	+	+	o
SD digital input	+	+	o
SD analog input	o	+	+
6 HQ channel audio	+	o	-
2 channel audio	-	+	+

legend	
+	present
o	optional
-	absent

Figure 1.5: Mapping technical functions on products

Figure 1.5 shows a mapping of technical functions and features onto products. The technical functions and features should still be oriented towards the *what* of the product. In practice this view emerges slowly after many iterations between design decompositions and commercial and logistics oriented decompositions.

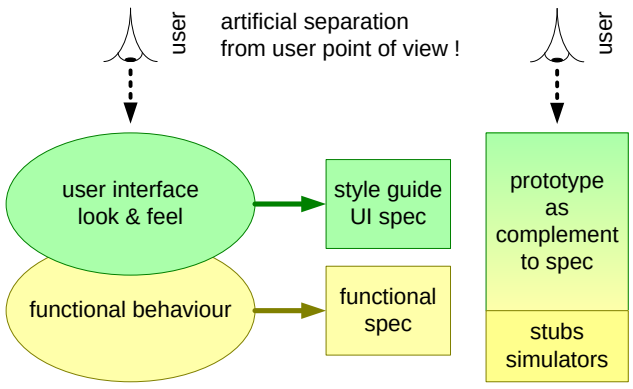


Figure 1.6: Relation between user interface and functional specification

The struggle in nailing down the functional specification is the degree in which user interface and functional specification are decoupled and separated. Separation eases the delivery of look and feel variants. However this separation from user point of view is rather artificial, see figure 1.6, which shows that the user experiences the system behavior via the user interface. As design team we create then artifacts as style guides, user interface specifications and functional specifications.

Another consideration is the high dynamics of user interface details versus the relative stability of the functions itself. Hard coupling of user interface description and functional specification propagates the dynamics of user interface details into the entire functional specifications.

Figure 1.6 offers an alternative solution for this dilemma by using a prototype as complement to the specification for the user interface details. Such an approach allows the team to limit the functional specification, style guide and user interface specification to the essentials. A clear description of the way of working is required for quality assurance purposes: the specification is leading and is verified, is the prototype archived and a formal part of the specification?

1.5 Performance

The performance need to be specified quantitatively and verifiable in the functional view. This means that the performance needs to be specified in conjunction with the circumstances in which this performance specification is valid. In easy cases a simple maximum value is sufficient, which is valid under all circumstances. In many systems the performance specification is more complicated: the system performance depends on the user settings of the system.

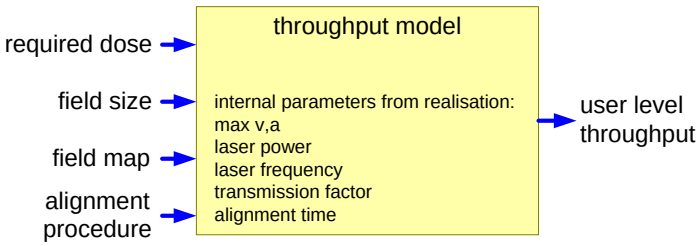


Figure 1.7: Example of performance modelling: throughput as function of user controlled values

In not too complicated systems it is sufficient to define a limited set of performance points in the parameter space. For more performance critical and complex systems an external performance model might be required, which describes the required relation between performance and user settings. Figure 1.7 shows an example of such a performance model for waferstepper throughput.

Throughput models are the result of several iterations between problem and solution space. Sufficient understanding of the solution space is needed to know which user parameters are relevant in the throughput model.

From the functional view *(the what perspective)* the internal design parameters are not relevant. In the iteration and decision process this model with external and internal parameters is a means to understand the consequences of design choices and to understand the consequences (cost) of customer needs.

The notion of *internal* and *external* is also somewhat artificial. In this example many customers measure the dose and do expect a certain relationship between dose and throughput. These customers perceive dose as externally known parameter.

Other examples of performance data are: standby time of a cell phone, gas consumption of a car, average monthly cost of a lease car. Note the increasing need in these examples for specification of the context.

1.6 Information Model

The information model is layered, as shown in figure 1.8. The highest layer is the understanding of the humans using the information model. This understanding is always biased by the individual human knowledge, emotional state and many other human factors, see [2]. The real meaning of information for human beings is never completely defined, humans always add interpretation to the definition.

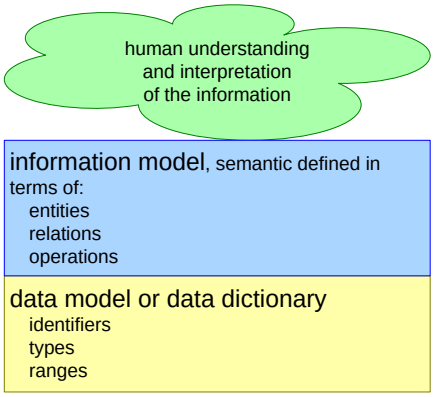


Figure 1.8: Layering of information definitions

The information model itself describes the semantics of the information. The syntax and representation aspects are described in the data model or data dictionary.

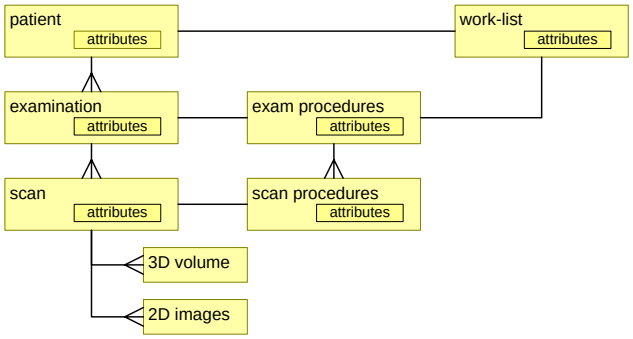


Figure 1.9: Example of a partial information model

The information model describes the information as seen outside of the system. It should not contain internal design choices. The information model is an important means to decouple interoperating systems. The functional behavior of the systems is predictable as long as all systems adhere to the information model. Figure 1.9 shows an example of a part of an information model.

The ingredients of an internal information model are:

- entities
- relations between entities
- operations on entities

The most difficult part of the information model is to capture the semantics of the information. The information model defines the intended meaning of the information, in terms of entities, their meaning, the relation with other entities and possible operations which can be applied on these entities.

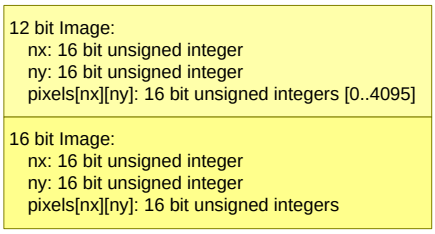


Figure 1.10: Small part of a datamodel

The technical details of the information model, such as exact identifiers, data types and ranges is defined in the datamodel. Figure 1.10 shows a small part of a datamodel defining 12 and 16 bit images. The term data dictionary is also often used for this lower level of definitions.

1.7 Standards

Compliance with standards is part of the product specification. The level of compliance and eventual exceptions need to be specified. Duplication of information in the standard must be avoided (minimize redundancy). The nice characteristic of standards in general is that the standards are extensively described and well defined. Most standard related implementation effort is straight forward engineering work, without the uncertainty of most other parts of the product specification.

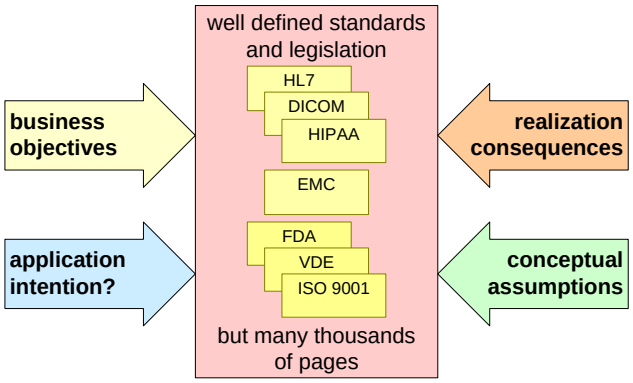


Figure 1.11: The standards compliance in the *functional view* in a broader force field.

Nevertheless architecting work is required in deciding on standards and in designing the implementation. Figure 1.11 shows the forces working upon the standards selection. The market and business environment more or less dictate a set of standards, if the product not comply the system is not viable. Some of these standards are mandatory due to legislation (for instance VDE or FDA related), some are de facto musts (for instance DICOM, the medical imaging communication standard).

The use of the standard and the compliance level depend on the intended use. A key question for the architect is: *What is the intention of the standard?* At the other hand standards are created by domain experts, which make all kinds of conceptual assumptions. If the standard is used in a way which does not correspond well with these assumptions, then it creates many specification and design problems. Good understanding of the underlying conceptual assumptions is a must for the architect.

The standard can have significant implementation consequences, for instance in the amount of effort needed or the amount of license costs involved in creating the implementation. These costs must be balanced with the created customer value.

A major problem with standards compliance is the massive amount of documentation and know how which is involved. The architect must find out the essence in terms of *objectives*, *intention*, *assumptions* and *consequences* of standards. In fact

the architect must have a *CAFCR* mental model per standard[2]. For communication purposes the architect can make this model explicit.

[2] the CAFCR model is in fact the architecture of the standard itself.

1.8 Summary

The functional view is concerned with all the required externally observable characteristics of the system. The CAFCR model puts a lot of emphasis on the customer. The operational viewpoint, from the producer point of view, determines also part of the system. Figure 1.12 summarizes the content of the functional view, where the left hand side shows the customer specifications and the right hand side the company operational specifications.

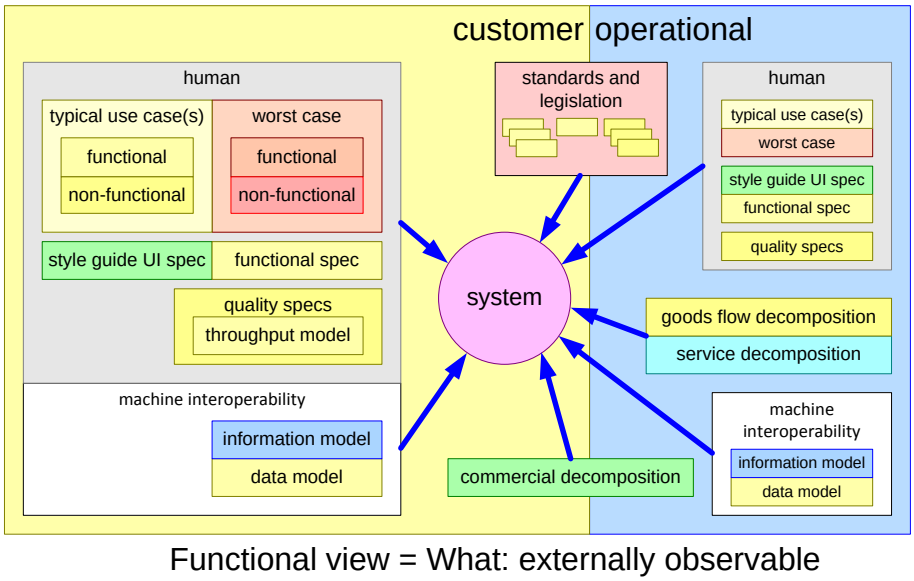


Figure 1.12: Summary of functional view

In the previous chapters we discussed the use cases, user interface, functional specification, quality specifications, and information model from customer point of view. As shown in this figure the same aspects need to addressed from the operational point of view, for example:

- typical use case for service and/or production
- functional specification and user interface for service
- performance of adjustment and verification measurements
- information interface for SPC (Statistical Process Control) purposes

Another classification used in figure 1.12 is human oriented or machine interoperability oriented. Again such a classification is artificial. For some products with a lot of human user interaction this is a useful separation. Other products, for

instance electronic or software components to be used in other systems, don't have immediate human users.

1.9 Acknowledgements

William van der Sterren was very fast in providing relevant feedback.

Bibliography

[1] Gerrit Muller. The system architecture homepage. http://www.gaudisite.nl/index.html, 1999.

[2] Gerrit Muller. Communicating via CAFCR; illustrated by security example. http://www.gaudisite.nl/CommunicatingViaCAFCRPaper.pdf, 2002.

History
Version: 0, date: July 2, 2004 changed by: Gerrit Muller

- created module