Module Customer Side

logo
TBD

Gerrit Muller
University of South-Eastern Norway-NISE
Hasbergsvei 36 P.O. Box 235, NO-3603 Kongsberg Norway
gaudisite@gmail.com

Abstract

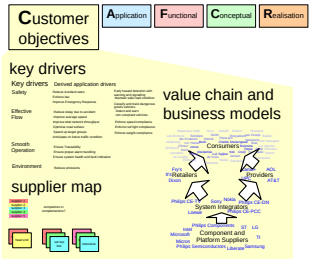
This module addresses The Customer Objectives and Application Views:

version: 0 status: draft September 1, 2020

Contents

Chapter 1

The customer objectives view



1.1 Introduction

The customer objectives view describes the goals of the customer, the **what**. The goal of articulating these objectives is to better understand the needs and therefore to be able to design a better product.

In searching the objectives some focus on the product is needed, although the architect must keep an open mind. The architect must prevent a circular reasoning, starting from the product functionality and, blinded by the product focus, finding only objectives matching with this same functionality.

Ideally the trade-offs in the customer domain become clear. For instance what is the trade-off between performance and cost, or size and performance or size and cost. The key driver method articulates the essence of the customer needs in a limited set of drivers.

The customer is often driven by his context. Some of the models and methods described here address ways to understand the customer context, such as value chains and business models. Value chains and business models are used to address the customer's customer. The supplier map addresses the supplying side of the customer.

Figure 1.1 shows an overview of the methods in the customer objectives view.

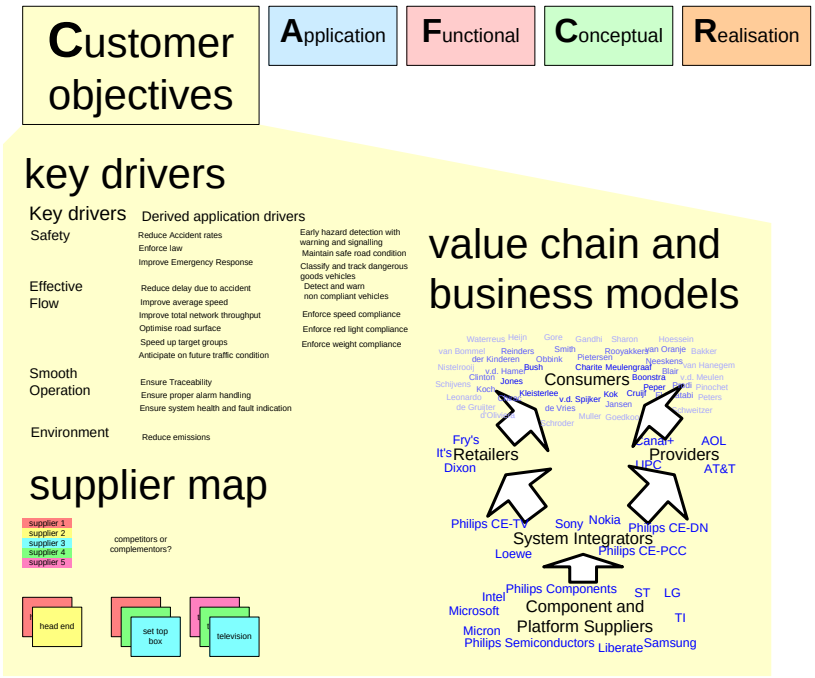


Figure 1.1: Overview of Customer Objectives View methods

1.2 Key drivers

The essence of the objectives of the customers can be captured in terms of customer key drivers. The key drivers provide direction to capture requirements and to focus the development. The key drivers in the customer objectives view will be linked with requirements and design choices in the other views. The key driver submethod gains its value from relating a few sharp articulated key drivers to a much longer list of requirements. By capturing these relations a much better understanding of customer and product requirements is achieved.

Figure 1.2 shows an example of key drivers for a motorway management system, an analysis performed at Philips Projects in 1999.

Figure 1.3 shows a submethod how to obtain a graph linking key drivers to requirements. The first step is to define the scope of the key driver graph. For Figure 1.2 the customer is the motorway management operator. The next step is to acquire facts, for example by extracting functionality and performance figures out of the product specification. Analysis of these facts recovers implicit facts. The requirements of an existing system can be analyzed by repeating *why* questions. For example: "Why does the system need *automatic upstream accident detection*?". The third step is to bring more structure in the facts, by building a graph, which

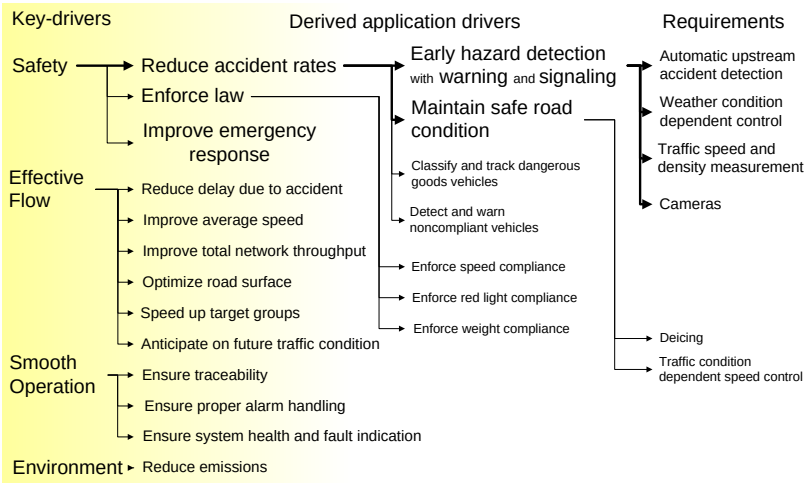


Figure 1.2: Example of the four key drivers in a motorway management system

connects requirements to key drivers. A workshop with brainstorms and discussions is an effective way to obtain the graph. The last step is to obtain feedback from customers. The total graph can have many n:m relations, i.e. requirements that serve many drivers and drivers that are supported by many requirements. The graph is good if the customers are enthusiastic about the key drivers and the derived application drivers. If a lot of explaining is required then the understanding of the customer is far from complete. Frequent iterations over these steps improves the quality of the understanding of the customer's viewpoint. Every iteration causes moves of elements in the graph in driver or requirement direction and also causes rephrasing of elements in the graph.

Figure 1.4 shows an additional set of recommendations for applying the key driver submethod. The most important goals of the customer are obtained by limiting the number of key drivers. In this way the participants in the discussion are forced to make choices. The focus in product innovation is often on differentiating features, or unique selling points. As a consequence, the core functionality from the customer's point of view may get insufficient attention. An example of this are cell phones that are overloaded with features, but that have a poor user interface to make connections. The core functionality must be dominantly present in the graph. The naming used in the graph must fit in the customer world and be as specific as possible. Very generic names tend to be true, but they do not help to really understand the customer's viewpoint. The boundary between the Customer Objectives view and the Application view is not very sharp. When creating the

• Define the scope specific.	in terms of stakeholder or market segments
• Acquire and analyze facts	extract facts from the product specification and ask why questions about the specification of existing products.
• Build a graph of relations between drivers and requirements by means of brainstorming and discussions	where requirements may have multiple drivers
• Obtain feedback	discuss with customers, observe their reactions
• Iterate many times	increased understanding often triggers the move of issues from driver to requirement or vice versa and rephrasing

Figure 1.3: Submethod to link key drivers to requirements, existing of the iteration over four steps

• Limit the number of key-drivers	minimal 3, maximal 6
• Don't leave out the obvious key-drivers	for instance the well-known main function of the product
• Use short names, recognized by the customer.	
• Use market-/customer- specific names, no generic names	for instance replace "ease of use" by "minimal number of actions for experienced users", or "efficiency" by "integral cost per patient"
• Do not worry about the exact boundary between Customer Objective and Application	create clear goal means relations

Figure 1.4: Recommendations for applying the key driver submethod

graph that relates *key drivers* to *requirements* one frequently experiences that a key driver is phrased in terms of a (partial) solution. If this happens either the key driver has to be rephrased or the solution should be moved to the requirement (or even realization) side of the graph. A repetition of this kind of iterations increases the insight in the needs of the customer in relation to the characteristics of the product. The **why**, **what** and **how** questions can help to rephrase drivers and requirements. The graph is good if the relations between goals and means are clear for all stakeholders.

1.3 Value chain and business models

The position of the customer in the value chain and the business models deployed by the players in the value chain are important factors in understanding the goals of this customer.

Figure 1.5 shows an example value chain from the Consumer Electronics Domain. At the start of the chain are the component suppliers, making chips and other elementary components such as optical drives, displays, et cetera. These compo-

nents are used by system integrators, building the consumer appliances, such as televisions, set top boxes and cellphones. Note that this value chain is often longer than shown here, where components are aggregated in larger components into subassemblies and finally into systems.

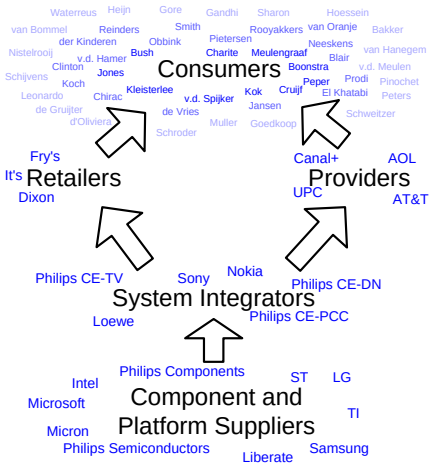


Figure 1.5: Example value chain

The consumer appliances itself are distributed through 2 different channels: the retailers and the service providers. Retailers sell appliances directly to the consumers, earning their money with this appliance sales and sometimes also with maintenance contracts for these appliances. Providers sell services (for instance telecom, internet), where the appliance is the means to access these services. The providers earn their money via the recurring revenues of the services.

Retailers and service providers have entirely different business models, which will be reflected by differences in the key drivers for both parties.

Reality is even much more complicated. For instance adding the *content* providers to the value chain adds an additional set of business models, with a lot of conflicting interests (especially Digital Rights Management, which is of high importance for the content providers, but is often highly conflicting with (legal) consumer interests).

1.4 Suppliers

The value chain must be described from the point of view of the customer. The customer sees your company as one of the (potential) suppliers. From the customer point of view products from many suppliers have to be integrated to create the total solution for his needs.

In terms of your own company this means that you have to make a map of

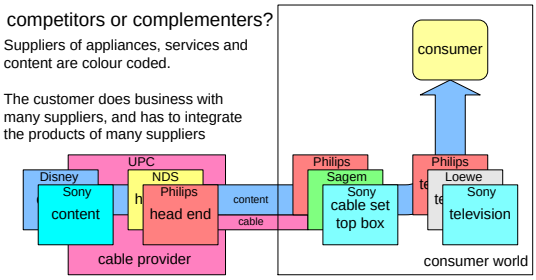
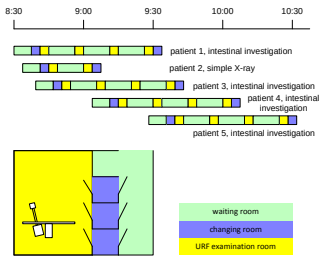


Figure 1.6: Example of simple supplier map for a cable provider

competitors and complementers, which together will supply the solution to the customer. Figure 1.6 shows an example of a simple supplier map for a cable provider. If your company is delivering set top boxes, then some companies can be viewed as competitor and complementer at the same time.

Chapter 2

The application view



2.1 Introduction

The application view is used to understand how the customer is achieving his objectives. The methods and models used in the application view should discuss the customer's world. Figure 2.1 shows an overview of the methods discussed here.

The customer is a gross generalization, which can be made more specific by identifying the customer stakeholders and their concerns, see section 2.2.

The customer is operating in a wider world, which he only partially controls. A context diagram shows the context of the customer, see section 2.3. Note that part of this context may interface actively with the product, while most of this context simply exists as neighboring entities. The fact that no interface exists is no reason not to take these entities into account, for instance to prevent unwanted duplication of functionality.

The customer domain can be modelled in static and dynamic models. Entity relationship models (section 2.4) show a static view on the domain, which can be complemented by dynamic models (section 2.5).

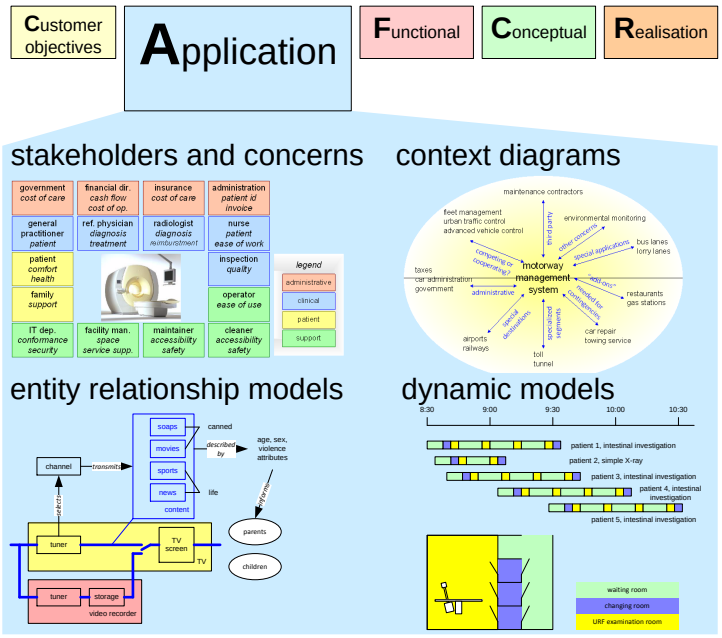


Figure 2.1: Overview of methods and models that can be used in the application view

2.2 Customer stakeholders and concerns

In the daily use of the system many human and organizational entities are involved, all of them with their own interests. Of course many of these stakeholders will also appear in the static entity relationship models. However human and organizations are very complex entities, with psychological, social and cultural characteristics, all of them influencing the way the customer is working. These stakeholders have multiple concerns, which determine their needs and behavior. Figure 2.2 shows stakeholders and concerns for an MRI scanner.

The IEEE 1471 standard about architectural descriptions uses stakeholders and concerns as the starting point for an architectural description.

Identification and articulation of the stakeholders and concerns is a first step in understanding the application domain. The next step can be to gain insight in the *informal* relationships. In many cases the formal relationships, such as organization charts and process descriptions are solely used for this view, which is a horrible mistake. Many organizations function thanks to the unwritten information flows of the social system. Insight in the informal side is required to prevent a solution which does only work in theory.

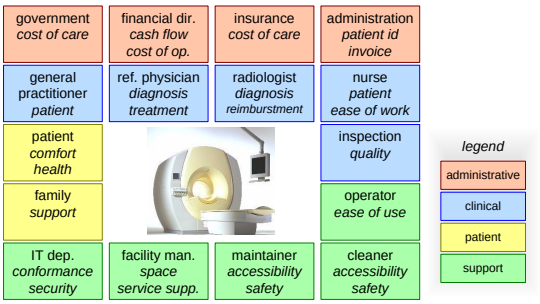


Figure 2.2: Stakeholders and concerns of an MRI scanner

2.3 Context diagram

The system is operating in the customer domain in the context of the customer. In the customer context many systems have some relationship with the system, quite often without having a direct interface.

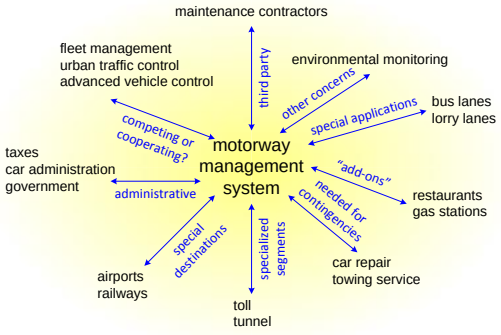


Figure 2.3: Systems in the context of a motorway management system

Figure 2.3 shows a simple context diagram of a motorway management system. Tunnels and toll stations often have their own local management systems, although they are part of the same motorway. The motorway is connecting destinations, such as urban areas. Urban areas have many traffic systems, such as traffic management (traffic lights) and parking systems. For every system in the context questions can be asked, such as:

- is there a need to interface directly (e.g. show parking information to people still on the highway)
- is duplication of functionality required (measuring traffic density and sending it to a central traffic control center)

2.4 Entity relationship model

The OO (Object Oriented software) world is quite used to entity relationship diagrams. These diagrams model the outside world in such a way that the system can interact with the outside world. These models belong in the ”CAFCR” thinking in the conceptual view. The entity relationship models advocated here model the customers world in terms of entities in this world and relations between them. Additionally also the activities performed on the entities can be modelled. The main purpose of this modelling is to gain insight in how the customer is achieving his objectives.

One of the major problems of understanding the customers world is its infinite size and complexity. The art of making an useful entity relationship model is to very carefully select what to include in the model and therefore also what **not** to include. Models in the application view, especially this entity relationship model, are by definition far from complete.

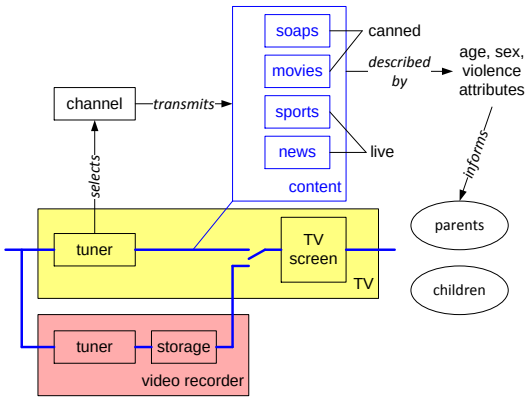


Figure 2.4: Diagram with entities and relationship for a simple TV appliance

Figure 2.4 shows an example of an entity relationship model for a simple TV. Part of the model shows the well recognizable flow of video content (the bottom part of the diagram), while the top part shows a few essential facts about the contents. The layout and semantics of the blocks are not strict, these form-factors are secondary to expressing the essence of the application.

2.5 Dynamic models

Many models, such as entity relationship models, make the static relationships explicit, but don’t address the dynamics of the system. Many different models can be used to model the dynamics, or in other words to model the behavior in time. Examples are of dynamic models are shown in figure 2.5

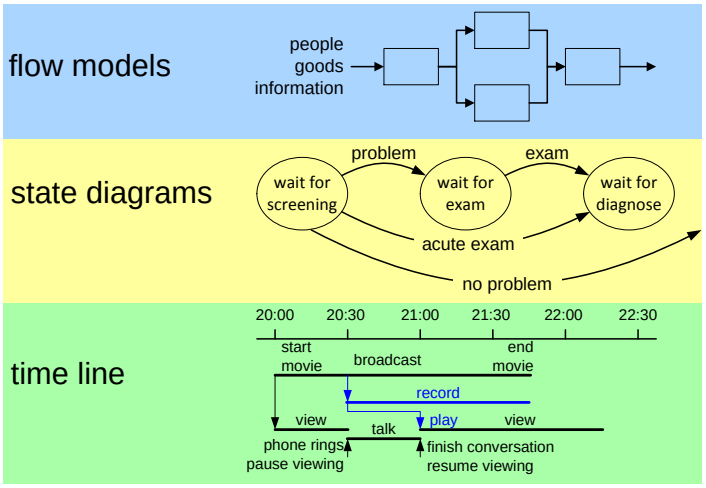


Figure 2.5: Examples of dynamic models

Productivity and Cost of ownership models are internally based on dynamic models, although the result is often a more simplified parameterized model, see figure 2.6.

Figure 2.7 shows an example of a time-line model for an URF examination room. The involved rooms play an important role in this model, therefore an example geographical layout is shown to explain the essence of the time-line model.

The patient must have been fasting for an intestine investigation. In the beginning of the examination the patient gets a barium meal, which slowly moves through the intestines. About every quarter of an hour a few X-ray images-images are made of the intestines filled with barium. This type of examination is interleaving multiple patients to efficiently use the expensive equipment and clinical personnel operating it.

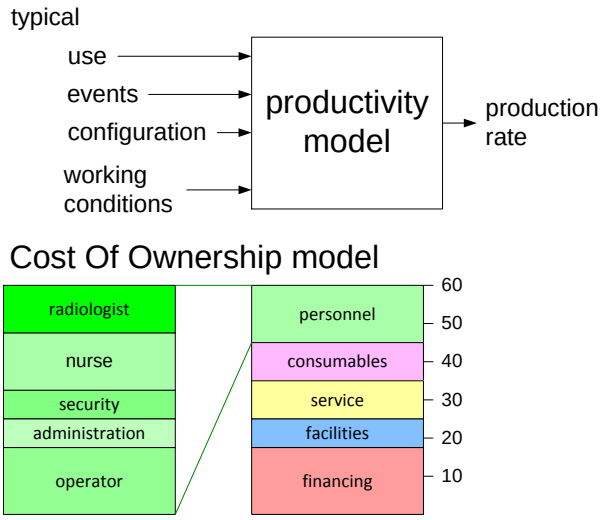


Figure 2.6: Productivity and cost models

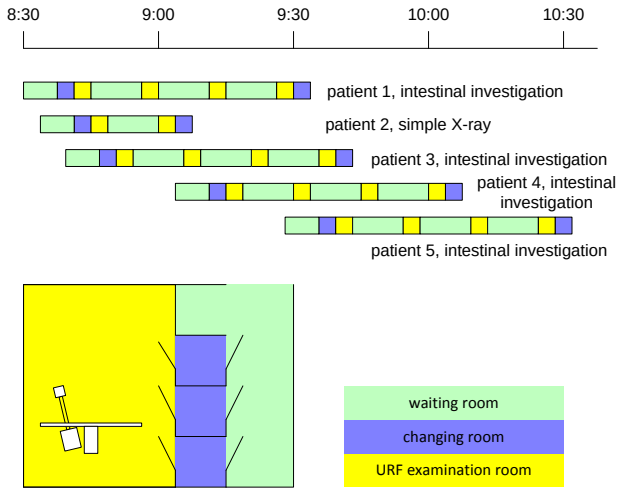


Figure 2.7: Dynamics of an URF examination room

Bibliography

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

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

- created module