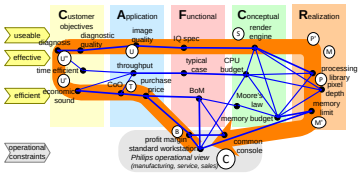
Threads of Reasoning in the Medical Imaging Case

-



cost revisited in context of clinical needs and realization constraints; note: original threads are significantly simplified

Gerrit Muller

University of South-Eastern Norway-NISE

Hasbergsvei 36 P.O. Box 235, NO-3603 Kongsberg Norway

gaudisite@gmail.com

Abstract

A thread of reasoning is build up in steps and the underlying reasoning is explained.

version: 2.4 status: finsihed September 9, 2018

1 Introduction

The thread of reasoning has not been applied consciously during the development of the Medical Imaging Workstation. This chapter describes a reconstruction of the reasoning as it has taken place. In Section 2 the outline of the thread is explained. Section 3 describes the 5 phases as defined in Chapter **??**:

1. Select starting point (3.1)
2. Create insight (3.2)
3. Deepen insight (3.3)
4. Broaden insight (3.4)
5. Define and extend the thread (3.5)

2 Example Thread

Figure 1 shows a set of interrelated customer objectives up to interrelated design decisions. This set of interrelated objectives, specification issues and concepts is a dominant thread of reasoning in the development of the medical imaging workstation.

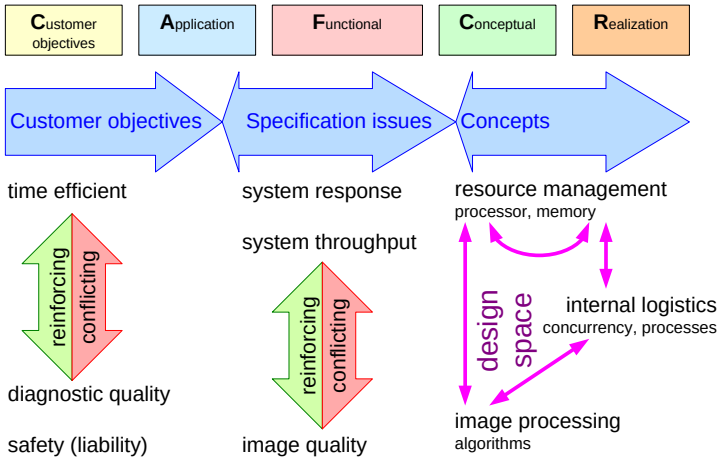


Figure 1: The thread of reasoning about the tension between time efficiency on the one hand and diagnostic quality, safety, and liability on the other hand. In the design space this tension is reflected by many possible design trade-offs.

The objectives of the radiologist are at the same time reenforcing and (somewhat) conflicting. To achieve a good diagnostic quality sufficient time is required or examine and study the results, which can be in conflict with time efficiency. On the

other hand a good diagnostic quality will limit discussions and repeated examinations later on, by which good diagnostic quality can help to achieve time efficiency.

The customer objectives are translated into specifications. The diagnostic quality and safety/liability translate for example into image quality specifications (resolution, contrast, artefact level). A limited image quality is a primary source of a poor diagnostic quality. Artifacts can result in erroneous diagnostics, with its safety and liability consequences.

The time efficiency is achieved by system throughput. The workstation should not be the bottleneck in the total department flow or in the system response time. Waiting for results is clearly not time efficient.

Also at the specification level the reenforcing and the conflicting requirements are present. If the image quality is good, no tricky additional functions are needed to achieve the diagnostic quality. For instance if the image has a good clinical contrast to noise ratio, then no artificial contrast enhancements have to be applied. Function bloating is a primary source of decreased throughput and increased response times. The conflicting aspect is that some image quality functions are inherently time consuming and threaten the throughput and response time.

The design space is full of concepts, where design choices are needed. The concepts of *resource management*, *internal logistics* and *image processing algorithms* have a large impact on the system *response time* and *throughput*. The *image processing algorithms* determine the resulting *image quality*.

The design space is not a simple multi-dimensional space, with orthogonal, independent dimensions. The image processing algorithm has impact on the CPU usage, cache efficiency, memory usage, and image quality. The implementation of these algorithms can be optimized to one or two of these entities, often at the cost of the remaining optimization criteria. For instance: images can be stored completely in memory, which is most efficient for CPU processing time. An alternative is to store and process small parts of the image (lines) at a time, which is more flexible with respect to memory (less fragmentation), but the additional indirection of addressing the image line costs CPU time.

Adding concurrency partially helps to improve response times. Waiting times, for instance for disk reads, can then be used to do other useful processing. On the other hand additional overhead in context switching, and locking is caused by the concurrency.

The essence of the thread of reasoning is to have sufficient insight in the customer and application needs, so that the problem space becomes sharply defined and understood. This understanding is used to select the *sweet* spots of the design space, that satisfy the needs. Understanding of the design space is needed to sharpen the understanding of the problem space; in other words iteration between problem and solution space is required.

3 Exploration of Problems and Solutions

In this section the thread of reasoning is shown as it emerges over time. For every phase the CAFCR views are annotated with relevant subjects in that phase and the relations between the subjects.

Figures 2 to 6, described in Subsections 3.1 to 3.5, show the phases as described in Chapter **??**. The figures show the main issues under discussion as dots. The relations between the issues are shown as lines between the issues, where the thickness of the line indicates the relative weight of the relationship. The core of the reasoning is indicated as a thick arrow. The cluster of issues at the start point and at the finish are shown as letter in a white ellipse. Some clusters of issues at turning points in the reasoning are also indicated as white ellipse.

3.1 Phase 1: Introvert View

At the moment that the architect (me) joined the product development a lot of technology exploration had been transformed into a working prototype, the so-called *basic application*. Main ingredients were the use of Object-Oriented (OO) technology and the vision that a "software only" product was feasible en beneficial.

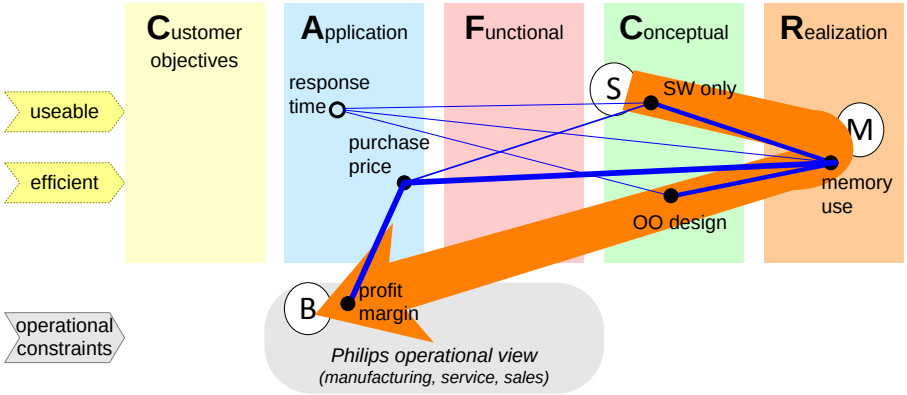


Figure 2: Thread of reasoning; introvert phase. The starting point (S) is the a priori design choice for a SW only solution based on Object Orientation. The consequence for resource usage, especially memory (M) and the business (B), especially product margin are explored.

Experienced architects will address two major concerns immediately: will the design with these new technologies fit in the technical constraints, especially memory in this case, and will the product fit in the business constraints (do we make sufficient margin and profit)?

The response time has been touched only very lightly. The system was only capable of viewing, an activity for which response time is crucial. The prototype showed acceptable performance, so not much time was spent on this issue. Design changes to eventually solve cost or memory issues potentially lower the performance, in which case response time can suddenly become important.

Figure 2 shows the thread of reasoning in this early stage. Striking is the introvert nature of this reasoning: internal design choices and Philips internal needs dominate. The implicitly addressed qualities are useability and efficiency. Most attention was for the operational constraints. The direction of the reasoning during this phase has been from the Conceptual and Realization views towards the operational consequences: starting at the designers choice for OO and software only (S), via concerns over memory constraints (M) towards the business (B) constraints margin and profit. The figure indicates that more issues have been touched in the reasoning, such as response time from user point of view. In the day to day situation many more related issues have been touched, but these issues had less impact on the overall reasoning.

3.2 Phase 2: Exploring Memory Needs

The first phase indicated that the memory use was unknown and unpredictable. It was decided to extend the implementation with measurement provisions, such as memory usage. The OIT in the dynamic run time environment enabled a very elegant way of tracing object instantiations. At the same time a new concern popped up: what is the overhead cost induced by the run time environment?

The object instantiation tracing could easily be extended to show the amount of memory allocated for the object structures. The large data elements, such as images, are allocated on the heap and required additional instrumentation. Via the bulkdata concept this type of memory use was instrumented. Bottom up the insight in memory use was emerging.

The need arose to define relevant cases to be measured and to be used as the basis for a memory budget. An URF examination was used to define a typical case. Now the application knowledge starts to enter the reasoning process, and the reasoning starts to become more extrovert. Efficiency and usability are the main qualities addressed.

Figure 3 shows the thread of reasoning for Phase 2. The reasoning is still bottom-up from Realization towards Application View. The realization concerns about speed and memory consumption (M') result into concepts for resource management and measurement support. For analysis and validation a use case description in the Functional view is needed. The use case is based on insight in a URF examination (U) from application viewpoint.

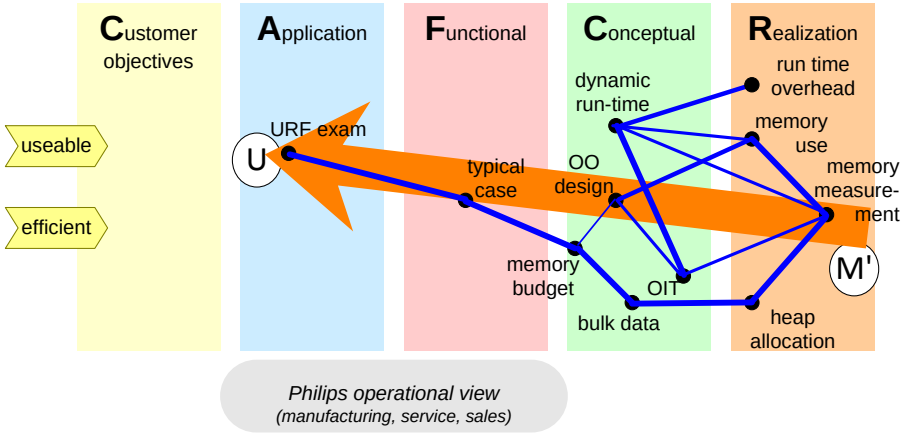


Figure 3: Thread of reasoning; memory needs. Create insight by zooming in on memory management (M'). Requirements for the memory management design are needed, resulting in an exploration of the typical URF examination (U).

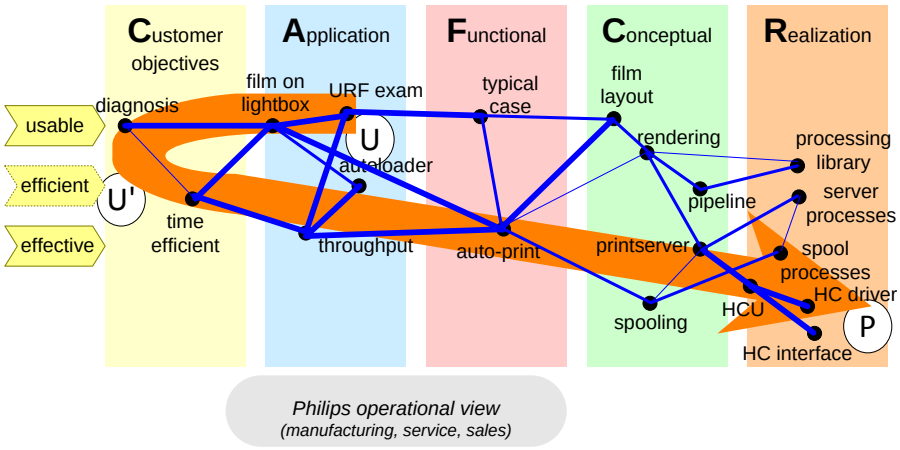
3.3 Phase 3: Extrovert View Uncovers Gaps in Conceptual and Realization Views

The discussion about the URF examination and the typical case made it very clear that radiologists perform their diagnoses by looking at the film on the lightbox. This is for them very efficient in time. Their speed of working is further increased by the autoloader, which quickly shows all films of the next examination.

To support this typical workflow the production of filmsheets and the throughput of films and examinations is important. Interactive viewing on the other hand is from the radiologist's point of view much less efficient. Diagnosis on the basis of film takes seconds, diagnosis by interactive viewing takes minutes. The auto-print functionality enables the production of films directly from the examination room.

auto-print functionality requires lots of new functions and concepts in the system, such as background printing (spooling), defining and using film layouts, using the right rendering, et cetera. The processing library must support these functions. Also an execution architecture is required to support the concurrency: server processes and spool processes are introduced. Last but not least, hardcopy units (HCU), for example laser printers, need to be interfaced to the system. A new set of components is introduced in the system to do the printing: hardcopy interface hardware, hardcopy driver, and the hardcopy units themselves.

During this phase the focus shifted from efficiency to effectiveness. Efficiency



Radiologists diagnose from film, throughput is important
Extrovert view shows conceptual and realization gaps!

Figure 4: Thread of reasoning; uncovering gaps. The insight is deepened by further exploration of the URF examination (U) and the underlying objectives (U') of the radiologist. The auto-print functionality is specified as response for the radiologist needs. The technical consequences of the auto-print are explored, in this case the need for printing concepts and realization (P).

is mostly an introvert concern about resource constraints. Effectiveness is a more extrovert concern about the quality of the result. Hitchins clearly explains in [1] efficiency and effectiveness, and points out that the focus on efficiency alone creates vulnerable and sub-optimal systems. Usability remains important during this phase, for example auto-print.

Figure 4 shows the thread of reasoning of Phase 3. The insights obtained during the previous phase trigger a further exploration of the Customer Objectives and Application View. The insight that an efficient diagnosis (U') is performed by means of film sheets on a lightbox (U) triggers the addition of the auto-print function to the Functional View. New concepts and software functions are needed to realize the auto-print function (P). The direction of reasoning is now top-down over all the CAFCR views.

3.4 Phase 4: from Diagnosis to Throughput

The discussion about URF examinations and the diagnostic process triggers another thread, a thread about the desired diagnostic quality. The high brightness and resolution of films on a lightbox ensures that the actual viewing is not degrading the diagnostic quality. The inherent image quality of the acquired and printed image is

critical for the final diagnostic quality.

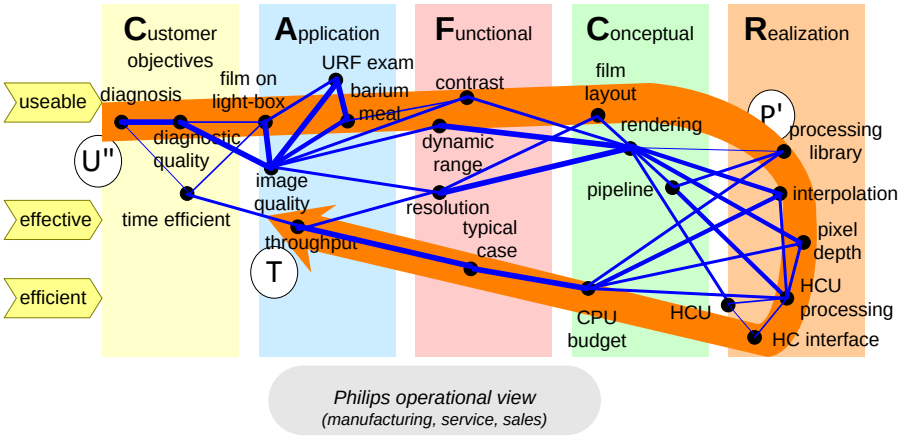


Figure 5: Thread of reasoning; phase 4. The insight is broadened. Starting at the objective to perform *diagnosis* efficient in time (U"), the application is further explored: type of examination and type of images. The specification of the imaging needs (contrast, dynamic range, resolution) is improved. The consequences for rendering and film layout on a large set of realization aspects (P') is elaborated. The rendering implementation has impact on CPU usage and the throughput (T) of the typical case.

At specification level the image quality is specified in terms of resolution, contrast and dynamic range. At application level the contrast is increased by the use of barium meal, which takes the contrast to the required level in these soft (for X-ray low contrast) tissues. At the same time the combination of X-ray settings and barium meals increases the dynamic range of the produced images.

The size of the images depends on the required resolution, which also determines the film layout. The rendering algorithms must fulfil the image quality specifications. The rendering is implemented as a pipeline of processing steps from an optimized processing library.

One of the costly operations is the interpolation. One of the design options was to use the processing in the hardcopy unit. This would greatly relieve the resource (processor and memory) needs, but it would at the same time be much less flexible with respect to rendering. It was decided not to use the hardcopy unit processing.

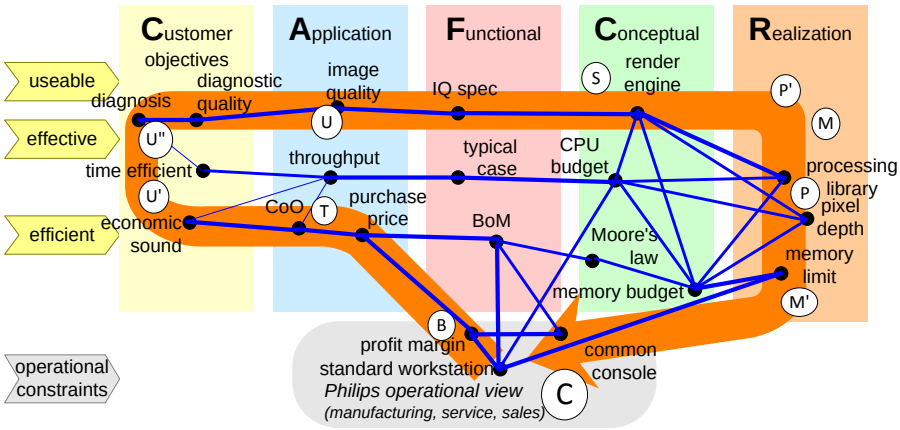
A CPU budget was created, based on the typical case and taking into account all previous design know-how. This CPU budget did fit in the required throughput needs.

Usability, effectiveness and efficiency are more or less balanced at this moment.

Figure 5 shows the thread of reasoning for Phase 4. During this phase the reasoning iterates over all the CAFCR views. The diagnostic quality (U") in the Customer Objectives View results via the clinical acquisition methods in the Application view in image quality requirements in the Functional View. The layout and rendering in the Conceptual view result in a large set of processing functions (P') in the Realization view. The specific know how of the processing in the Realization is used for the CPU and memory budgets in the conceptual view, to validate the feasibility of supporting the typical case in the Functional view. The typical case is a translation of the throughput (T) needs in the Application View.

3.5 Phase 5: Cost Revisited

At this moment much more information was available about the relation between resource needs and system performance. The business policy was to use standard of-the-shelf workstations. The purchase price by the customer could only be met by using the lowest cost version of the workstation. Another policy was to use a Philips medical console, which was to be common among all products. This console was about half of the material cost of the Medical Imaging workstation.



cost revisited in context of clinical needs and realization constraints; note: original threads are significantly simplified

Figure 6: Thread of reasoning; cost revisited. The entire scope of the thread of reasoning is now visible. Sufficient insight is obtained to return to the original business concern of margin and cost (C). In the mean time additional assumptions have surfaced: a common console and standard workstation to reduce costs. From this starting point all other viewpoints are revisited: via time efficient diagnosis to image quality to rendering and processing and back to the memory design.

The real customer interest is to have a system that is economically sound, and where throughput and cost of ownership (CoO) are balanced. Of course the main clinical function, diagnosis, must not suffer from cost optimizations. A detailed and deep understanding of the image quality needs of the customer is needed to make an optimized design.

Note that at this moment in time many of the considerations discussed in the previous steps are still valid and present. However Figure 6 is simplified by leaving out many of these considerations.

Besides efficiency, effectiveness, and usability, the operational constraint is back in the main reasoning thread. At this moment in time that makes a lot of sense, because problem and solution space are sufficiently understood. These constraints never disappeared completely, but the other qualities were more dominant in the intermediate phases.

Figure 6 shows the thread of reasoning in Phase 5. The original business viewpoint is revisited: do we have a commercial feasible product? A full iteration over all CAFCR views relates product costs (C) to the key drivers in the Customer Objectives. The main tensions in the product specification are balanced: image quality, throughput of the typical case and product cost. To do this balancing the main design choices in the Conceptual and Realization views have to be reviewed.

4 Conclusion

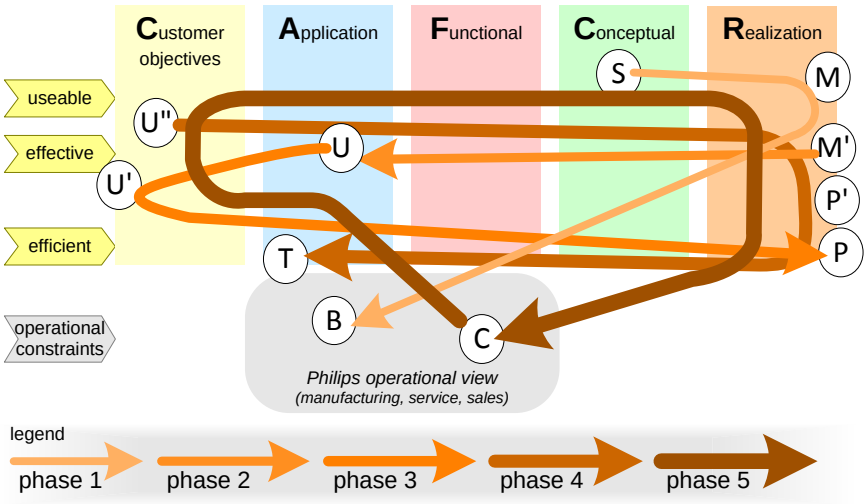


Figure 7: All steps superimposed in one diagram. The iterative nature of the reasoning is visible: the same aspects are explored multiple times, coming from different directions. It also shows that jumps are made during the reasoning.

The know-how at the start of the product creation was limited to a number of nice technology options and a number of business and application opportunities. The designers had the technology know-how, the marketing and application managers had the customer know-how. The product creation team went through several learning phases. Figure 7 shows the many iterations in the five phases. During those phases some of the know-how was shared and a lot of new know-how emerged. The sharing of know-how made it possible to relate customer needs to design and implementation options. The interaction between the team members and the search for relations between needs and designs triggered many new questions. The answers to these questions created new know-how.

The architecting process has been analyzed in retrospect, resulting in this description of *threads of reasoning*. This Chapter *Threads of Reasoning* shows that:

- The specification and design issues that are discussed fit in all CAFCR views or the operational view.
- The positioning of the issues and their relationships in the CAFCR views enable a compact description of the reasoning during the product creation.
- Submethods are used to address one issue or a small cluster of issues.
- Qualities are useful as integrating elements over the CAFCR views.
- The *threads of reasoning* are an explicit way to facilitate the interaction and the search for relations.
- The *threads of reasoning* create an integral overview.
- The *threads of reasoning* facilitate a converging specification and design.

5 Acknowledgments

Discussions with Frans Beenker helped to shape the phased thread diagrams. Feedback of Pieter Hartel triggered the need for definition of the semantics in the figures, and helped to provide a better explanation.

References

[1] Derek K. Hitchins. Putting systems to work. `http://www.hitchins.co.uk/`, 1992. Originally published by John Wiley and Sons, Chichester, UK, in 1992.

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

History

Version: 2.4, date: April 5, 2004 changed by: Gerrit Muller

- updated the 5 phase figures.

Version: 2.3, date: March 16, 2004 changed by: Gerrit Muller
- added short description to Hitchins reference
- updated the 5 phase figures.
- changed status to finished

Version: 2.2, date: March 1, 2004 changed by: Gerrit Muller
- added section “Introduction” and “Conclusion”
- added short description to the caption of the 5 phase figures
- updated the 5 phase figures.
- changed status to concept

Version: 2.1, date: February 10, 2004 changed by: Gerrit Muller
- clarified the semantics of the thread figures.
- added description of the thread of reasoning outline for every phase
- added the use of CAFCR views to the "The Medical Imaging Case in Retrospect"

Version: 2.0, date: November 14, 2003 changed by: Gerrit Muller
- many small text changes
- added subsection "The Medical Imaging Case in Retrospect”
- changed status to draft

Version: 1.2, date: May 27, 2003 changed by: Gerrit Muller
- added qualities to the thread diagrams and to the accompanying text

Version: 1.1, date: May 21, 2003 changed by: Gerrit Muller
- moved chronological description of events to chapter: ”Medical Imaging creation in chronological order”

Version: 1.0, date: May 14, 2003 changed by: Gerrit Muller
- added phases of the thread of reasoning
- added section ”Acknowledgements”
- set status to preliminary draft

Version: 0.5, date: May 9, 2003 changed by: Gerrit Muller
- added CAFCR as reference to figure 1 thread of reasoning
- added axis annotation to the Memory budget
- added the meaning of x and $f(x)$ in the section safety problem.

Version: 0.4, date: April 8, 2003 changed by: Gerrit Muller
- added text to section development
- added text to section Performance problem
- added text to section Safety problem

Version: 0.3, date: April 7, 2003 changed by: Gerrit Muller
- added (empty) sections for safety and exloration

Version: 0.2, date: March 26, 2003 changed by: Gerrit Muller
- added chronology of Easyvision RF R1 development
- Added figures performance problem due to memory shortage

Version: 0.1, date: February 18, 2003 changed by: Gerrit Muller
- Added text to ”Example thread”

Version: 0, date: February 6, 2003 changed by: Gerrit Muller
- Created, no changelog yet