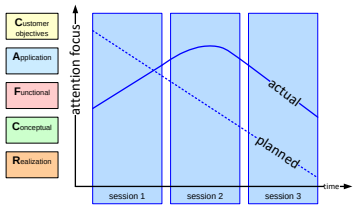
Evaluation from a Wider Context

-



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

Abstract

The architecting method is evaluated by means of experiences in research projects, in workshops and in courses. This evaluation complements the evaluation of the architecting method by means of the medical imaging case.

version: 1.3 status: finished September 1, 2020

1 Introduction

The architecting method has been illustrated by means of the medical imaging workstation case, and this case was used to evaluate the method. Some aspects of the method could not be evaluated, mostly because these aspects have not been made explicit until after the case period.

In Chapter **??** other evaluation possibilities are indicated, such as *research projects*, *workshops* and *courses*. The significance of this wider context is the potential of performing architecting method research with a greater statistical significance. This chapter does not provide this statistical data. More research, with well defined research protocols, is needed to obtain more robust results. Observations from this context are discussed, however, to show the potential of research in this wider context.

Section 2 evaluates the method in the research environment, Section 3 in the workshop settings, and Section 4 in the course setting. In these sections the suppotive information is indicated by an identification tag:

c n for CAFCR and multi-view information

q n for quality cheklist information

s n for story telling information

i n for iteration information

u n for usability information

In Section 5 the findings are summarized.

2 Research Environment

For many years research of architecting methods has been performed at Philips Research in the SwA (Software Architectures) group. The research projects in which this group participated have been using some of the discussed architecting (sub)methods consciously.

A short description of the research projects that applied some of the methods is given below:

Family Asset Management How to manage electronic assets, such as movies, pictures, and music? The publication is: [12] is based on this work.

Project infrastructure platform Electronic and software infrastructure (closed circuit TV, security, public access et cetera) to support project organizations in the domains of industrial buildings, banks, railway stations, airport terminals, and motorways.

Heartcare Image and information integration of all cardio-related information from cathlab to personal monitoring used at home or away.In [1] story telling is used in the scenario approach. In [9] story telling, called scenarios in this article, are used to make architectures better future proof.

Platform for portable multi-media Few or single chip electronics and software platform for the creation of mobile multi-media systems (cellphones, PDAs, personal audio, et cetera).

Software productivity for audiovisual systems How to create the software for integrated and connected audio and video systems (TVs, set-top boxes, personal video recorders, DVD recorders, et cetera) in a limited amount of time?

Composable architectures How to create architectures that support composability? This project consolidated and exchanged experiences over a wide range of products: from televisions to cathlabs. The publications [14], [10], [2], and [11] are the first articles and presentations of the BAPO (Business, Architecture, Process and Organization) and CAFCR models. The PhD thesis [4] by Jürgen Müller zooms in on the *Conceptual* and *Realization* views and provides a method to design components that fulfill multiple qualities. In these design views these qualities are called aspects.

In particular the following (sub)methods have been used:

- the decomposition in the 5 CAFCR views
- story telling
- qualities
- iteration over multiple views

Compared to the medical imaging workstation case these research projects put more emphasis on the *Customer objectives* and *Application views* **(c1)**. This results in more focused research projects and less technology push. Researchers in the platform-oriented projects (infrastructure, multi-media, software productivity), for instance, discovered that solutions were being pushed without any clear need at the customer side **(o1)**.

Story (or scenario) telling has been explicitly researched and will be subject of continuing research. The following benefits **(s1)** of story telling were experienced in the *Family Asset management*, *Heartcare*, *portable multi-media* and *software productivity* projects:

- Communication with the less technical stakeholders is improved.
- Exploration discussions are more to the point: less time is lost on too generic discussions

Explicit attention for qualities (based on the qualities checklist in Figure **??**) also helps to focus the research projects and to find the relevant research issues quicker **(q1)**. Speed of exploration is essential for research projects: identify promising options, and filtering out unattractive options. The speed of exploration is improved by identifying the essential qualities and by identifying the qualities that can be ignored.

Only a very limited improvement in exploration speed has been observed **(i1)**. The highly individual nature of researchers appears to be a bottleneck. Also the diversity and fragmentation of the group of stakeholders, with their individual interests, is a bottleneck. Both bottlenecks hamper the sharing of objectives and the identification of most important qualities. Improvement in exploration speed is certainly possible, but this requires an interaction of architecting with the context of business, processes, and people. New projects at the Embedded Systems Institute, which are set up outside the Philips processes and organization, show promising results. See for instance the Boderc project [3].

No validating or invalidating evidence about the *threads of reasoning* is obtained from the research environment. The *threads of reasoning* did only exist as a vague notion [8].

3 Workshops

The architecting methods have also been used to structure many different kinds of workshops. The subjects of these workshops covered areas such as: strategy, roadmapping, project definition exploration of problem and solution domain, cross fertilization, and architecture assessment. The domains that were investigated were quite varied, for example: MR, X-ray, semiconductors, displays, storage, motorway management, and printers.

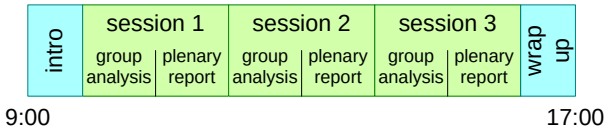


Figure 1: Typical workshop program template

Figure 1 shows the typical program template of these workshops. Most time is used to stimulate interaction among the participants focused on the subject. This interaction takes place in small teams based on a few predefined questions. The result of these discussions is presented and discussed plenary. A session with the interaction from one team and the plenary presentation typically takes two hours. In a one-day workshop about three successive sessions can be scheduled. The remaining time is needed for introduction and wrap up.

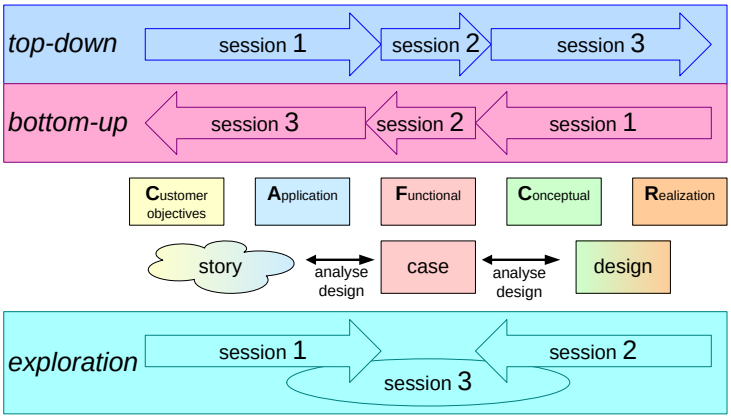


Figure 2: Workshop approaches

Figure 2 shows several approaches to structure the questions for the three sessions: *top-down*, *bottom-up*, and *exploration*. The basis for all these approaches is the *CAFCR* model **(c2)**, complemented with *story telling* **(s2)**. All approaches have been used with small variations.

The *top-down* approach requires participants that are open and sufficiently customer aware. **What** and **how** questions help the participants to move the investigation from customer towards realization.

In the *bottom-up* approach the link to the customer world is created by repeating **why** questions. The bottom-up approach works well, but should be followed by a top-down question: "Did we start with the *valid* solution?". The improved understanding of the customer often results in adjustments to the original solution. In some cases the participants conclude that the solution is invalid: the solution is not addressing the need of the customer. In that case more appropriate solution directions are generated during the workshop.

The *exploration* approach is more open:

- What is needed?
- What is possible?
- So what are going to create?

The exploration approach is appropriate if sufficient freedom of choice is available; it works less well in very constrained situations.

Also the level of abstraction must be chosen: very generic, for instance identifying key drivers, or very specific via a story. Both generic and specific approaches have been used, as well as combinations of the two. The more generic approaches work well if the participants already have a good shared understanding. If the understanding is limited or not shared specific approaches work better.

The formulation of the questions for the sessions is critical. The questions must be specific to trigger a concrete discussion. The questions must be open to prevent too much bias in the solutions. The CAFCR submethods **(c3)** and the qualities **(q2)** are useful sources of inspiration to articulate the questions. Examples in the *Customer objectives* view are:

- How does the value chain for digital televisions look in 2006?
- What are the key drivers for neuro radiology?

Examples from the *Realization* and *Conceptual* views are:

- What are the most critical system resources for this story? Please quantify.
- What functionality is provided by the Microsoft COM framework? What functionality do we actually use?

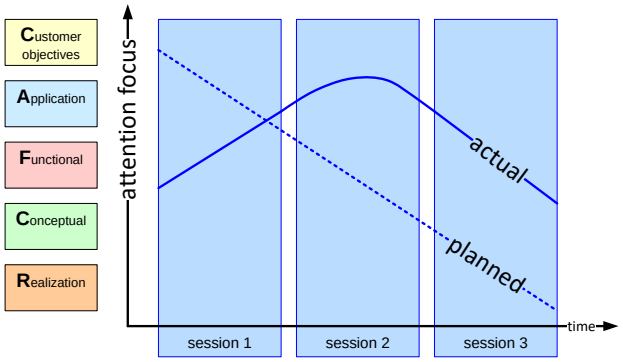


Figure 3: Hysteresis due to latency in viewpoint change

The basis of all these approaches is to stimulate the participants to perform a rapid shared iteration. In facilitating more than 30 of these workshops I have observed that the iteration speed in these workshops is limited **(i2)**. Many participants need time to make the context switch. The consequence of this context switch time is that an hysteresis occurs in the goal of the workshop program and the actual execution, as shown in Figure 3.

This observation has implications for the usability and efficiency of the architecting method:

- The iteration speed is limited by the capabilities of the architect using them
- The architect must be well aware of the limited iteration speed of his stakeholders. Iterating too quickly in interaction with the stakeholders causes a phase difference between architect and stakeholders. The phase difference has a negative impact on the communication.

System specification and design problems are often caused by the missing links between the *CAFCR* views. Iteration over the *CAFCR* views makes it possible to identify important and critical issues and their relations earlier. Faster iterations bring problems quicker to the surface. In zeroth order[1]: the efficiency of a method is proportional with the iteration speed. The iteration speed is not directly dependent on the method. The speed of iteration is determined by the capabilities of the workshop participants. This again is an area where architecting method and the process and people interfere: the usability of the architecting method depends on the skills and the capabilities of the people and the organization.

4 Courses

Some of the submethods are being trained as part of a System Architecting Course. The experiences of teaching this course are described in [7]. As part of the course the participants have to do exercises, some of them using the submethods. The course is mostly focused on the non-technical aspects of system architecting. Five of the course modules, shown in Figure 4, have relevant exercises for the evaluation. In the figure is indicated what submethod is used per exercise. The course has been given 20 times between November 1999 and February 2003, with a total amount of participants of about 300.

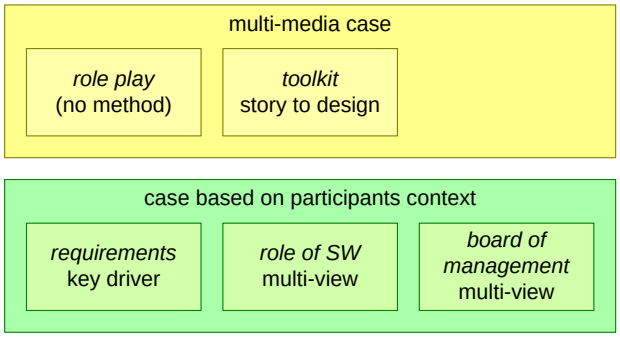


Figure 4: Submethods used in course exercises

The *role play*, which is not yet using any method, is relevant because it functions as a kind of zero measurement. The participants play the roles of product manager, project leader, and architect. Together they have to define a new multi-media product, including an indication of business relevance and potential schedule. At

[1] If the iteration speed is too high, no practical fact finding and analysis can be applied. A higher order model will show a drop in efficiency for too high iteration speeds. In small (circa 4 people) teams, with a shared background, I have observed useful results in iterations of less than 1 hour **(i3)**.

the beginning of the course no methods have been provided yet to cope with this kind of problem.

In the *toolkit* exercise newly mixed teams have to use the *story telling* technique to discuss the same product as used in the *role play* exercise. They have to create a story and to make a start with the analysis and design. Often participants remark that the method would have helped them greatly in the earlier *role play* exercise **(s3)**. The teacher can observe the difference between defining a product without method (the zero measurement) and defining a product with a story telling as method. This order of exercises makes the participants aware of the value of methods. The learning effect increases by experiencing both situations: without and with methods.

The other relevant exercises are all based on the daily context of the participants. The teams are optimized for domain cohesion. Participants are grouped in such a way that they share more or less the same application area and the same type of problems. For example, group names are: *digital video*, *MR*, *X-Ray*, *automotive*, and *optical storage*.

In the *requirements* exercise they have to make a graph, as described in Section **??**, from key driver to requirements **(c4)**. This is often experienced as an eye-opener: how much more exists than the internal design, how little do we know about the customers!

In the *role of software* the participants have to make a presentation about the software in their system. The explicit recommendation is to do this with multiple diagrams in the Conceptual and Realization view (functional decomposition, layering, flows, size, et cetera) **(c5)**. The presentation should make the intangible software understandable for non-software people. Without this recommendation most engineers tend to explain the software from a single diagram (the class diagram or the layers). Enriching this with other diagrams, such as sizes and other dimensions, helps significantly to make the software more tangible.

The final exercise is a simulated *Board of Management (BoM)* presentation [6], where every team has to give a presentation about an important architectural issue to a management team that is significantly higher up in the hierarchy. They need to deploy a lot of what they learned during the course. To create a successful presentation sufficient customer and business understanding is required (that is the main interest of this higher management team), but it needs to be related to multiple relevant architectural views **(c6)**. The choice of views and submethods is entirely up to the participants.

Many architects struggle in day-to-day life with the perceived lack of understanding of architecting issues by higher management. The *BoM* exercise addresses this problem by improving the presentation content of the (potential) architects. For many participants it is an eye opener to present design issues (Conceptual or Realization views) in relation with the business justification (Customer Objectives, Application, and Functional views).

In the lecture *requirements engineering* for the OOTI-curriculum [13] *CAFCR* and *story telling* are introduced as means to elicit requirements[2]. For most of the postgraduate students customers are far away, they need quite some nudging to take customer needs into consideration. *CAFCR* **(c7)** and *Story telling* **(s4)** clearly help them to think more in customer terms.

5 Conclusion

One of the weak spots of the evaluation by means of the medical imaging workstation was the application of submethods in the *Customer objectives* and *Application* views. These submethods have been used much more in all three categories (research, workshops, and courses). This resulted in a more clear project focus and more attention for the customer needs in research projects, compared to previous research projects. Application of these views in workshops improved the attention for the customer needs and the project focus relative to the situation before the workshop. More focus is for most projects of today a big improvement.

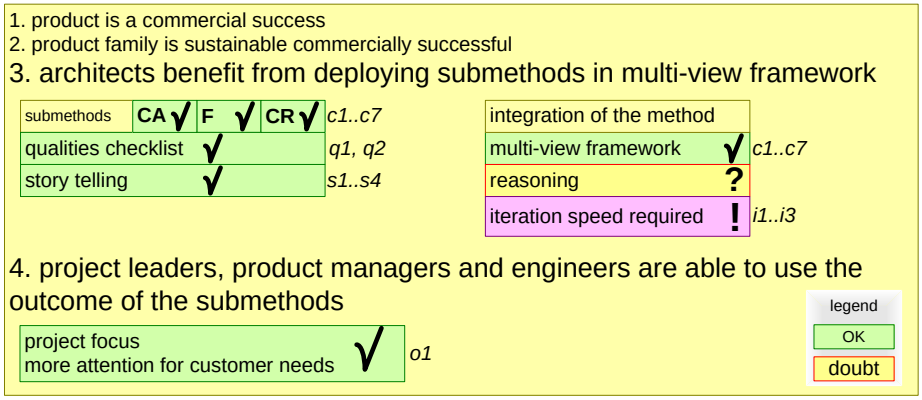


Figure 5: Conclusions of the evaluation in a wider context. The tags are defined in the Sections 2 to 4.

Figure 5 shows the conclusions of this evaluation. These conclusions do not address the product and its future, but only the architect and his stakeholders. The use of the *CAFCR multi-view framework* helps to cope with complex product creation problems. The available *submethods* are successfully used in research environments, workshops, and courses. The qualities from the *quality checklist* helped to bring focus to research projects. No supporting evidence with respect to *threads of reasoning* is obtained from these sources.

[2] Also here the hysteresis effect shown in Figure 3 is present: in the five days of this lecture and execution of a case it is often difficult to do the iteration more than once; the design analysis is sometimes too superficial, due to the attention on the customer needs.

A large majority of people lack the skills to iterate very fast over the CAFCR views. They have, however, no problems in following the reasoning when explained. The conclusion is that those who act as architect and deploy this architecting method must have the capability to iterate quickly. In case of an architecting team at least one of the members of the team must have this iterating capability.

6 Acknowledgements

Pierre America and Jan Gerben Wijnstra provided me with some of the reference data.

References

[1] Pierre America. Making architectures future-proof using scenarios. `http://www.serc.nl/lac/2003/presentaties/Track1/P.America.pdf`, 2003.

[2] Pierre America, Henk Obbink, Rob van Ommering, and Frank van der Linden. COPAM: A component-oriented platform architecting method family for product family engineering. In Patrick Donohoe, editor, *Proceedings of the First Software Product Line Conference*, August 2000.

[3] Embedded Systems Institute. Boderc project. `https://www.embeddedsystems.nl/Boderc`, 2003.

[4] Jï¿½rgen K. Mï¿½ller. *The Building Block Method: Component-Based Architectural Design for Large Software-Intensive Product Families.* Universiteit van Amsterdam, 2003. Ph.D. thesis.

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

[6] Gerrit Muller. How to present architecture issues to higher management. `http://www.gaudisite.nl/ArchitectManagementInteractionPaper.pdf`, 2003.

[7] Gerrit Muller. Experiences of teaching systems architecting. To be published INCOSE 2004 in Toulouse, 2004.

[8] Gerrit Muller, Jï¿½rgen Mï¿½ller, and Jan Gerben Wijnstra. Multi-view architecting. `http://www.gaudisite.nl/IntegratingCAFCRPaper.pdf`, 2001.

[9] Henk Obbink. Scenario-based architecting: Towards architecting the future. http://www.serc.nl/lac/2003/presentaties/Track1/H.Obbink.pdf, 2003.

[10] Henk Obbink, Jürgen Müller, Pierre America, and Rob van Ommering. COPA: A component-oriented platform architecting method for families of software-intensive electronic products. http://www.hitech-projects.com/SAE/COPA/COPA_Tutorial.pdf, 2000.

[11] Henk Obbink, Rob van Ommering, Jan Gerben Wijnstra, and Pierre America. Component oriented platform architecting for software intensive product families. In Mehmet Aksit, editor, *Proceedings of Software Architectures and Component Technology*. Kluwer Enschede, January 2000.

[12] Alexander Sinitsyn. A synchronization framework for personal mobile servers. PerWare '04: Middleware Support for Pervasive Computing; Workshop (at 2nd Conference on Pervasive Computing), March 2004.

[13] Stan Ackermans Institute. Ooti: Post-masters program in software technology. http://wwwooti.win.tue.nl/, 2004.

[14] Jan Gerben Wijnstra. Critical factors for a successful platform-based product family approach. In *Proceedings of the Third Software Product Line Conference*, August 2002.

History

Version: 1.3, date: March 17, 2004 changed by: Gerrit Muller

- added descriptions to the references
- added a clear indication that this type of research is promising, but requires more research to obtain robust results
- linked the conclusions more explicit by using numbered tags
- changed status into finished

Version: 1.2, date: January 19, 2004 changed by: Gerrit Muller

- small textual changes
- changed status into concept

Version: 1.1, date: January 6, 2004 changed by: Gerrit Muller

- added citation to a FAM related project

Version: 1.0, date: December 12, 2003 changed by: Gerrit Muller

- changed chapter title
- changed status into draft
- many small text improvements

Version: 0.1, date: October 1, 2003 changed by: Gerrit Muller

- adapted Figure Conclusions
- removed Introduction figure
- removed Research figure
- moved workshop subject list from figure to text
- clarified text about iteration speed

Version: 0, date: July 1, 2003 changed by: Gerrit Muller

- Created, no changelog yet