

How to relate design decisions to stakeholder satisfaction.

Bridging the broad stakeholder universe and the detailed technology world.

Gerrit Muller

The satisfaction of stakeholders depends on “hard” factors (e.g. economical, financial and legal) and “soft” factors (e.g. psychological, social, political and cultural). We will discuss the broadness of the stakeholder universe and its relation to the depth of technical design world. In practice the stakeholder world seems to be disconnected from the technical design world. These two worlds should be connected to the degree that the realized product appropriately fulfills the needs of all stakeholders. We position the architect as the team member that has to connect those entities. We will zoom in on those characteristics of the architect that seem to be missing in the current architect profile and that have large impact on (human) stakeholder satisfaction.

Introduction

If we look at the steady stream of new products and applications that are created, then we see that many of them suffer from usability problems. Most well known examples are consumer products such as Personal Video Recorders or DVD-players, where often only one family member is sufficiently capable to operate the product. Only a few products and applications seem to be intuitive and inspiring. Positive examples are Apple products, where lots of attention has been paid to these human factors. Also the *Sense and Simplicity* drive of Philips acknowledges the problems in usability in the past and importance of these human factors for the future. The first question is why so little products are satisfactory. The next question is what we can do to improve this situation.

Current practice is that products are created by multi-disciplinary teams staffed by many experts. These teams are often complemented with a few cross-cutting roles, such as marketing, project management and architecting. In this article we will focus on the architecting role, where we assume that one or more architects perform most of the architecting work.

Our starting point is today's architect: an experienced autodidact whose personality fitted the job at hand. Many studies and books describe the profile of today's architect. However, we observe that this profile is weak in the humane aspects. We will discuss the role of the architect to facilitate an extension of the desired architect profile. In the extended profile we focus on the additional capabilities needed for human-oriented system design.

We shortly discuss the role of patterns in capturing experience. We propose to enrich the pattern collection with human-oriented patterns as a practical way to achieve our ideal of systems that satisfy their human stakeholders.

The role of the architect

In this section we discuss the role of the architect and we discuss how today's architects have emerged. This insight is later used to discuss the extension of the profile of the architect to improve the effectiveness of the architect in the humane aspects.

We pose here that the architect is responsible for the quality of the product and its design. The quality of the product is mostly measured by the fit of the product in its operating context. If the product fits well then the customers¹ are satisfied, while customer as well as producer and maintainer can run a healthy business. The quality of the design facilitates the quality of the product. However, a good design also facilitates evolution of the product and its derivatives. A good design will enable the creation of products at the right combination of qualities such as performance, cost, effort, reliability et cetera.

The architect fulfils this responsibility by:

- Decomposing the system in subsystems, functions, components, et cetera, in such a way that product integration is easy and efficient in time, effort and cost
- Balancing conflicting product qualities and design choices
- Maintaining and communicating the overview of the product and its design
- Maintaining the integrity and consistency of the design
- Striving for the most simple solution
- Striving for the most appropriate solution from stakeholder perspective

The daily work of the architect is to communicate with stakeholders and experts in the team, while exploring (and reading, thinking, analyzing) design details, design concepts and options, specification options, application needs, and other context information.

The role of the architect as posed here clearly indicates that the architect is heavily involved in the human aspects of products in the customer context. A poorly usable product indicates that the related experts and the architect were not able to fulfill the architectural goal satisfactorily.

How do architects grow?

System architects need a wide range of knowledge, skills and experience to be effective. Figure 2 shows a typical development of a system architect, based on observations of experienced architects. Understanding how architects have grown in the past may help us foster new architects faster. For example, teach and coach potential architects to make them aware of relevant know how and skills.

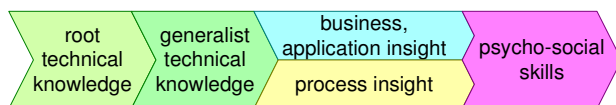


Figure 2 Typical Development of a System Architect

The system architect is rooted in technology. A thorough understanding of a single technological subject is an essential underpinning. The next step is a broadening of the technical scope.

When the awakening system architect has reached a technological breadth, it will become obvious that most problems have a root cause outside of technology. Two main parallel streams are opened:

- The business side: the market, customers, value, competition, logistics, service aspects
- The process side: who is doing what and why

¹ Note that customers represent a broad term, from end-users and operators, to decision makers and purchasers

During this phase the system architect will broaden in these two dimensions. The system architect will view these dimensions from a technological perspective. Again when a sufficient level of understanding is attained awareness starts to grow that people behave much less rationally than technical designs. The growing awareness of the psychological and the sociological aspects of both developers and other colleagues, as well as customers is the next phase of growth.

Why is it difficult to architect for user satisfaction?

In this section we discuss the lack of user satisfaction by introducing two simple models for breadth ("CAFPCR") [America 2000] and depth (the exponential pyramid). We position the architect as the person who connects breadth and depth.

Observation of a large set of projects, ranging from high volume products such as semiconductor development, video products, and car navigation to low volume professional products, such as printers, electron microscopes, medical equipment or semiconductor equipment, showed us that quite often the design team including the architect focus entirely on the internals of the product. Their starting point is a set of product requirements. They try to satisfy this set of requirements as good as possible. Many members of the design team are attracted to the details of the solution, and have difficulty to come back to the essence of the product.

Stepping outside the product

The design of satisfactory products requires understanding of the customer context where the product will be used. We use the CAFPCR model as a means to design and assess architectures. Figure 2 shows the CAFPCR model, existing of 5 views. The *Customer Objectives* view captures the **what** of the customer: What does the customer want to achieve? The *Application* view captures **how** the customer achieves these objectives. These two views help the architect to step outside the product. The *Functional* view captures the **what** of the product as black box: What does the product do? The *Functional* view also captures the desired product qualities, such as performance and reliability, sometimes called the non-functional requirements. The design of the product, the **how** of the product, is divided in two views: the *conceptual* view and the *realization* view. The *conceptual* view captures the essentials of the designs in concepts. This view is reasonably stable over time, while the *realization* tends to evolve quickly caused by fast technological developments. The well known 4+1 model from Kruchten [Kruchten 1995] also slightly enters the customer context by adding use cases in the +1, however the CAFPCR model extends much farther in the customer context. Use cases in the CAFPCR model fit in the *functional* view.

Note that specialized experts are often active in a subset of these views. Engineers typically focus on the Conceptual and Realization views, while designers of Human factors, information ergonomics, and user centered design focus much more on the application and functional views. Business analysts, marketing and product managers have their focus on customer objectives and functional view. In many organizations the horizontal links are decoupled by documents: requirements analysis reports, requirements specifications, functional specifications et cetera. We position the architect as being secondary responsible to all these aspects, where the primary responsibility is owned by the specialized expert.

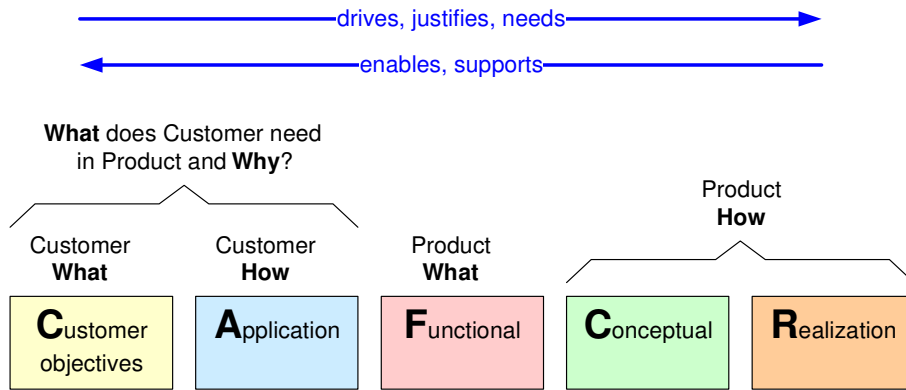


Figure 2 CAFCR a simple model relating the product design to the customer context.

Drowning in the details

Modern products are internally determined by tens of millions of details. For example the mechanical, electronics and software repositories contain each millions of definitions, parameters or lines of code. This is visualized in Figure 3, where the mono-disciplinary level at the bottom is at the level of 10^7 details. However, the system can also be viewed at many higher levels of abstractions: highly abstract at system level with a few hundred details or as multi-disciplinary design with thousands of details. A system can be expressed in its functions and key performance parameters, without any technological detail. The multi-disciplinary design decomposes the system in functions and subsystems and starts to allocate functions and properties to mono-disciplinary components.

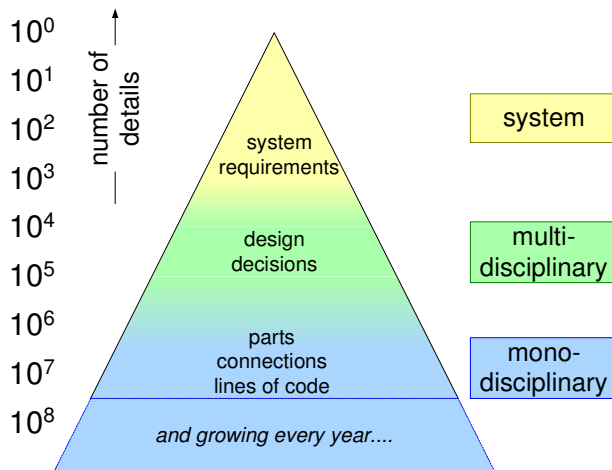


Figure 3 Connecting System Design to Detailed Design

The architect should maintain the overview in this complex technology world with millions of details. The translation of system requirements into detailed mono-disciplinary design decisions spans many orders of magnitude. The few statements of performance, cost and size in the system requirements specification ultimately result in millions of details in the technical product description: million(s) of lines of code, connections, and parts. The technical product description is the accumulation of mono-disciplinary formalizations. Figure 7 shows this dynamic range as a pyramid with the system at the top and the millions of technical details at the bottom.

Connecting the broad system context to the detailed design

The combination of Figures 2 and 3 brings us to a very common organizational problem: the disconnect between customer oriented reasoning (breadth, CAF) and technical expertise (depth, the mono-disciplinary area in the pyramid). Figure 4 shows this disconnect. Breadth

and depth can also be understood by looking at the Zachman framework [Zachman 1987] that defines 36 (6 by 6) views to capture breadth and depth.

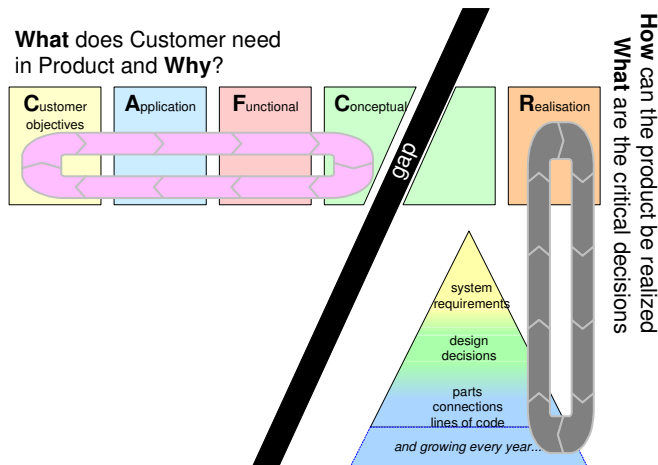


Figure 4 Organizational Problem: Disconnect

Our definition of the work of an architect places this role as a bridge between these two worlds. The architect has to stretch the horizontal iterative loop and connect it to the vertical loop. In essence the architect must combine and balance breadth and depth iterations.

We should realize that this architect role is quite a stretching proposition. The architect is stretched in customer, application and business direction and at the same time the same architect is expected to be able to discuss technological details at nuts and bolts level. By necessity the architect will function most of the time at higher abstraction levels, time and brain capacity don't allow the architect to spend all time at detailed design level. People fill different spots in the abstraction hierarchies of the design; system architects work at system level, senior designers make the connection from system design to mono-disciplinary engineering, and engineers focus on mono-disciplinary engineering. For communication purposes and to get a healthy system design the roles must have sufficient overlap. This means that all players need to be stretched regularly beyond their natural realm of comfort.

The architect's profile

In this section we will focus on the profile of the architect and especially what extensions to this profile are required to improve the effectiveness of the architect for the humane aspects.

The job of an architect is quite demanding and requires quite some capabilities. [Muller 2001] provides an experience based profile for an "ideal" architect. In [Frampton 2005] and [Moti 2006] competences or skills are analyzed that architects should possess. We propose to add the following desired characteristics to the profile to improve the hit rate of products that are satisfactory to customers:

- Observation skills
- Imagination
- Rich collection of technical, organizational and human-oriented patterns

And more in general, the discussion emphasized the need for *reflection* as kind of meta capability: *reflection* is the critical evaluation of processes and activities, and/or the related artefacts. *Reflection* is needed to develop skills, and especially to develop the above mentioned additional skills.

Observation skills

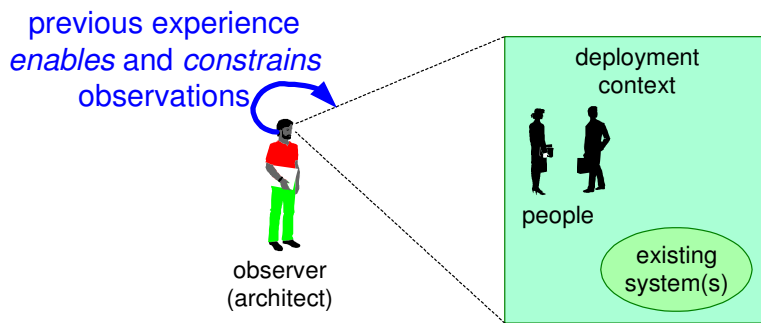


Figure 5 Observation skills determine the capability to understand the existing situation.

Observation is a highly subjective action. Nevertheless, we need an architect to be able to observe an existing situation in such a way that we understand people, system behavior, and interaction. Note that this includes the purely human interaction, person-to-person. One of the difficult aspects of observing is that observations are always influenced (filtered) by previous experience, see Figure 5. Previous experience is a reference for observation, without it we observe chaos. Unfortunately, previous experience also heavily triggers many interpretations that pollute the pure observation. Architects need to be aware of these factors; Observations have to be sufficiently open or perceptive.

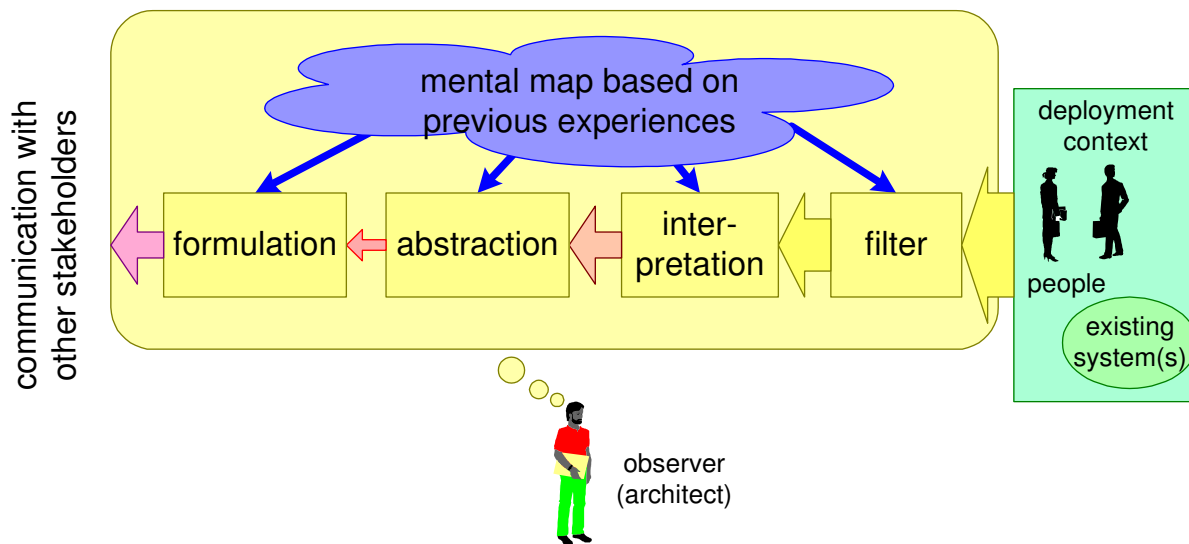


Figure 6 We make many (un)conscious steps from *observation* to *communicated idea*.

When we see something we first of all unconsciously filter what we see. Next we interpret, and then we tend to make abstractions of the filtered and interpreted observations. To communicate again with other humans we package the information by formulating sentences. All these steps, as shown in Figure 6, are influenced by our previous experiences. We (unconsciously) grow a mental map that captures our understanding of the world around us. We use the metaphor "map" because maps are the result of years of observations and interpretations. The mental map influences the way we look, interpret, abstract and formulate: past experiences set expectations. We tend to see (something close to) what we expect.

For example let's assume that Sara and Pieter are talking near the gantry at the office, while both of them have a cup in their hands. Our subconscious filtering might have removed anything from dust on the floor or the texture of the walls, to details of Pieter's hair dressing. The interpretation might be that they have social interaction during their coffee break. The abstraction might be that office workers have social interaction while drinking coffee. The architect might propose to have a gantry at every floor to stimulate social interaction.

All steps from filtering to formulation are sensible steps. However, the architect should be aware of the steps that have been taken. The architect has to consider alternatives taking into account the credibility of the information and the their own mental map. In the example seeing cups in the hands of Sara and Pieter is interpreted as *drinking coffee*. They might also be heading for the garbage bin, or cleaning their cups. The step from *talking* to *social interaction* is even bigger. In general we observe humans from the outside and we interpret their verbal and nonverbal communication. Conclusions about mood and emotions are inevitably the result of an interpretation. Hence, evaluation of someone's satisfaction is a tricky activity. A common pitfall is to mistake the generalization as truth. In the generalization step we go from two individuals, Sara and Pieter, to the abstraction *office workers*. Many individuals are abstracted into few generalized stakeholders. In the generalization a lot of specific information is lost. The danger is that we forget the natural variation that is present in a population of individuals. We then design for some "grey", averaged, set of stakeholders.

The final step to communicate about our observation is also a step where a lot of information is deformed or lost. One of the most common forms of communication is written text. The idea to be communicated is packaged in words and sentences. Every reader of this text reads the text with their own personal background and experiences. If only written text is used then no feedback loop is present to calibrate the meaning of the words, see [Muller2002]. Direct interaction between architect and stakeholder allows for much more calibration of the language and concepts that are used. Nevertheless, also in direct interaction information is deformed and lost.

Imagination

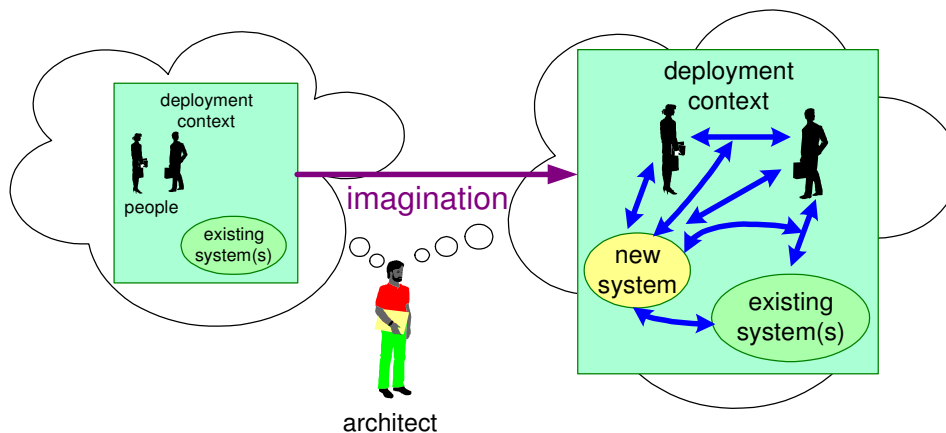


Figure 7 Imagination is needed to reason about the impact of the new system.

Figure 7 shows that imagination is used to think about the consequences of introducing a new (or updated) system. Imagination is the capability to create mental images of artifacts and situations that might occur as consequence of change. When we introduce a new system, then the new system has to interact with the existing systems and the people in the deployment context. However, the introduction will also impact the interaction of people, people and existing systems, and existing systems mutually. The imagination of the architect helps the architect to reason about the consequences of changes.

Note that this imagination works within, or starts from, the same mental map shown in Figure 6. As architects we project the consequences of our ideas guided by our mental map to imagine how the new system will be perceived. We should, however, as architect be aware that the particular individual that will be confronted with our new system perceives the world based on his or her own mental map. In other words we have to take into account the invisible, highly implicit, mental map of our individual stakeholders. One of the means to achieve this is to strive for more explicit modeling and to frequently communicate and share models.

Rich collection of technical, organizational and human-oriented patterns

Patterns are a possible way to capture experience and to communicate with stakeholders. Christopher Alexander uses the notion of patterns to capture solutions, see for example <http://www.patternlanguage.com/> and [Alexander 1979]. The pattern describes the question (what do we want), the context (where, when) and the possible solution with its characteristics.

Someone who is actively observing and reflecting collects in due time a rich collection of patterns for the specific domains of interest. This collection of patterns is a reference to quickly understand situations and to imagine the impact of changes. Note also that this works positive as well as constraining. An architect must be well aware of the limitations of this reference framework to be able to look for creative (new, still unknown) solutions when appropriate.

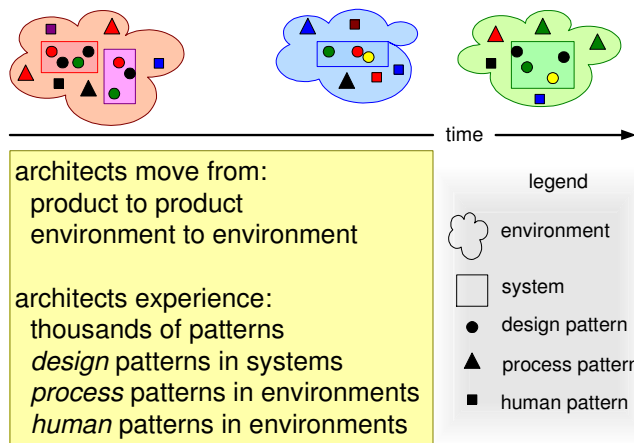


Figure 7 Architects build a rich collection of patterns in due time, taken from [Muller 2006].

Figure 7 shows that the architect builds a reference in terms of many different patterns in due time. Some design oriented patterns are common within a system or across systems. A subset of these design patterns also reappear in different environments. Some of the patterns are more abstract or organizational. Note that the environment as shown in this figure is the environment of the architect: the development organization.

Evert van de Waal [Waal 2005] proposed system level patterns as research subject in the Boderc project. This proposal triggered a discussions, where the conclusion of the Boderc participants at that moment is that the human brain has an amazing capacity. This brain capacity allows an experienced architect to have a collection of tens of thousands of patterns at different levels of abstraction. This rich collection is one of the key success factors in architecting, because it provides a broad framework for design reflection. It takes quite some time and reflection to collect and grasp this amount of patterns. Later, when teaching a course based on the CAFCR model of Figure 2, we observed that the effectiveness of the participants is directly correlated to their experience level and the contents of the pattern collection.

Let's look at user interfaces for an illustration of patterns. User interfaces can be "moded", the meaning of buttons depends on the mode of the system, or "mode-less", the meaning of buttons remains constant. *Moded* user interfaces tend to be more complicated, because history plays a role in the user action, while *mode-less* user interfaces have a easy one-to-one relation between button and related action. However, in systems with lots of interaction, the user interface of *modeless* user interfaces tends to grow beyond acceptable limits: there are too many buttons simultaneously present; a problem that is solved in *moded* user interfaces. Another pattern in user interfacing is the queuing of actions when buttons are pressed quicker than the system handles them. Such queuing is friendly for fast typing users, but can have unexpected side effects, especially when the meaning of buttons depends on previous actions, as in *moded* user interfaces. This paragraph explained very shortly two rather trivial patterns (which, nevertheless, are misapplied frequently). This particular pattern is normally applied

and owned by user interface experts. Typically architects "share" many patterns with other experts. Architects are dealing with several of these patterns daily, building the rich collection of tens of thousands of patterns in decades of work.

One important type of pattern is missing in the discussion above: the *human-oriented patterns*:

- What system interactions are inspiring to humans?
- What system interactions are difficult, confusing or irritating?
- How does system functionality interfere with the social interaction between humans?
- How do systems support communication between humans?
- How does the system appreciate differences between individuals?

Reflection of observations of actual humans and systems can be captured in human-oriented patterns.

How to develop these architecting competences?

In the previous sections we discussed the role of the architect, the emergence of today's architect, and the desired extension of the profile of the architect. In this section we discuss how to achieve such an extension by coaching potential architects in the way that they build up their experience. We discuss patterns as one of the specific ways to capture experience.

We explained that observation skills and imagination are required to create humane systems. Both observation and imagination are talents (the nature factor) that need to be developed and fed to reach the desired level (the experience factor).

Development of these talents is more than doing it often. On top of doing it often we recommend to:

- Offer a lot of variation in business, applications, systems and roles. The use of time-boxes ranging from a few months to few years is recommended. At ESI we have good experiences with internships of 3 months for this purpose. The *Architecture School* at Philips Research used time-boxes of two years per domain.
- Provide coaching by someone who is close by and who can provide immediate feedback.
- Stimulate reflection about product and process; an external coach can be instrumental or peer intervention. This worked well at the *Architecture School* at Philips Research.
- Challenge and stimulate.

Note that the combination of these activities is important. Seeing a variety of businesses, applications, systems and roles will only shape the observation skills if the architect reflects on observations, conclusions and consequences. Critical questions from an outsider are needed to step sufficiently *out of the box*. Both *observation* and *imagination* are skills that require quite some nourishment to develop.

Summary

We started with the observation that really satisfactory and inspiring products and applications are scarce. We posed that the architect should play a dominant role to achieve human oriented products. We discussed that it is quite challenging to bridge the breadth ("CAF") of the customer context and the (technological) depth of the design, the exponential pyramid of details. We identified *observation, imagination, and a rich collection of patterns* as additional necessary characteristics for architects. We pose that cooperation of experts with depth knowledge and architects with the above mentioned characteristics facilitate the connection of the broad stakeholder needs to the detailed world of technical decisions

Acknowledgements

Fruitful discussions in the Dutch working group “De menselijke maat in de IT” (Human Measure in IT) inspired this article. Especially the contributions from Dieter Hammer, Erik Philippus and Hans Oesterholt-Dijkema have improved the content of this article.

Gerrit Muller

**Embedded Systems
Institute**

Gerrit.muller@esi.nl

References

- [Alexander 1979] Christopher Alexander: *The timeless way of building*, 1979
- [America 2000] 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.
- [Frampton 2005] K. Frampton, J. M. Carroll, and J. A. Thom: *What capabilities do IT architects say they need?* In 10th United Kingdom Academy for Information Systems (UKAIS) Proceedings, 2005.
- [Kruchten 1995] Philippe B. Kruchten: *The 4+1 view model of architecture*. IEEE Software, pages 42–50, November 1995.
- [Moti 2006] F. Moti: *Knowledge, abilities, cognitive characteristics and behavioral competences of engineers with high capacity for engineering systems thinking (CEST)*, Systems Engineering 9 (2006), no. 2.
- [Muller 2006] G. Muller: *Decomposing the Architect; What are Critical Success Factors?* Presented at the INCOSE chapter meeting March 2006 in Washington <http://www.gaudisite.nl/DecomposingTheArchitectPaper.pdf>
- [Muller, 2001] G. Muller: *Function Profiles; the Sheep with Seven Legs*, <http://www.gaudisite.nl/FunctionProfilesPaper.pdf> 2001
- [Muller 2002] G. Muller: *Communicating via CAFCR; illustrated by security example*. Presented at LAC 2002, Nieuwegein November 2002 <http://www.gaudisite.nl/CommunicatingViaCAFCRPaper.pdf>
- [Waal 2005] Evert van de Waal: *Report Incose symposium 2005 and plans*, internal Boderc presentation, 2005
- [Zachman 1987] John Zachman. *The Zachman framework for enterprise architecture*. <http://www.zifa.com/>, 1987.