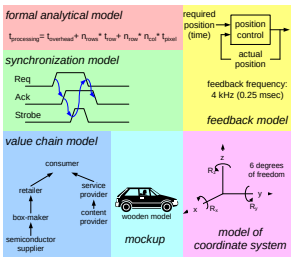
Architectural Fast Viewpoint Hopping

-



Gerrit Muller
USN-SE
Hasbergsvei 36 P.O. Box 235, NO-3603 Kongsberg Norway
gaudisite@gmail.com

This paper has been integrated in the book "Systems Architecting: A Business Perspective", http://www.gaudisite.nl/SABP.html, published by CRC Press in 2011.

Abstract

The challenge for the architect is to cover a wide range of subjects, with many unknowns and uncertainties, while decisions are required all the time.
This paper describes basic working methods, such as viewpoint hopping, quantification, and handling uncertainties.

version: 0 status: concept May 25, 2026

1 Introduction

This section is a selection of [2]. A limited set of generic patterns cover the basic working methods of architects:

- Viewpoint hopping, looking at the problem and (potential) solutions from many points of view.
- Decomposition, breaking up a large problem in smaller problems, introducing interfaces and the need for integration, see section **??**.
- Quantification, building up understanding by quantification, from order of magnitude numbers to specifications with acceptable confidence level.
- Decision making when lots of data is missing, see section 4.
- Modelling, as means of communication, documentation, analysis, simulation, decision making and verification.
- Asking Why, What, How, Who, When, Where questions, see section.
- Problem solving approach.

2 Viewpoint hopping

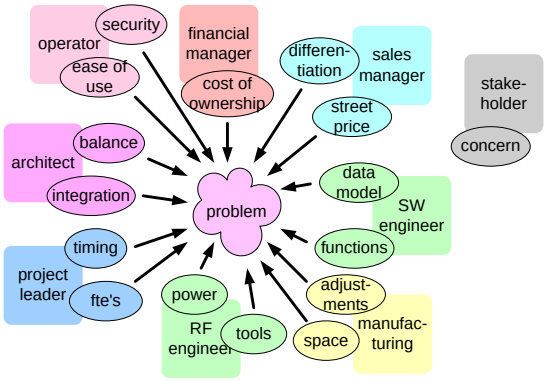


Figure 1: Small subset of viewpoints

The architect is looking towards problems and (potential) solutions from many different viewpoints. A small subset of viewpoints is visualized in Figure 1, where the viewpoints are shown as stakeholders with their concerns.

The architect is interested in an overall view on the problem, where all these viewpoints are present simultaneously. The limitations of the human brains force

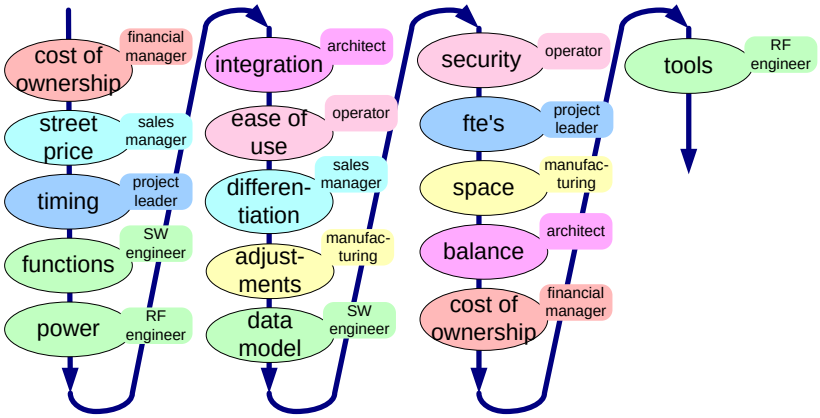


Figure 2: Viewpoint Hopping

the architect to create an overall view by quickly alternating the individual viewpoints. The order in which the viewpoints are alternated is chaotic: problems or opportunities in one viewpoint trigger the switch to a related viewpoint. Figure 2 shows a very short example of viewpoint hopping. This example sequence can take anywhere from minutes to weeks. In a complete product creation project the architect makes thousands[1] of these viewpoint changes.

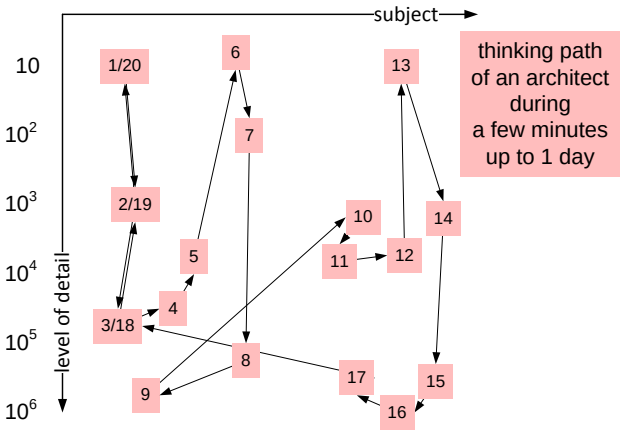


Figure 3: The seemingly random exploration path

Viewpoint hopping is happening quite fast in the head of the architect. Besides changing the viewpoint the architect is also zooming in and out with respect to

[1] Based on observations of other architects and own experience.

the level of detail. The dynamic range of the details taken into account is many orders of magnitude. Exploring different subjects and different levels of detail together can be viewed as an exploration path. The exploration path followed by the architect (in the architect's head) appears to be quite random. Figure 3 shows an example of an exploration path happening inside the architects head.

The plane used to show the exploration path has one axis with *subjects*, which can be stakeholders, concerns, functions, qualities, design aspects, et cetera, while the other axis is *the level of detail*. A very coarse (low level of detail) is for example the customer key driver level (for instance cost per placement is 0.1 milli-cent/placement). Examples at the very detailed level are lines of code, cycle accurate simulation data, or bolt type, material and size.

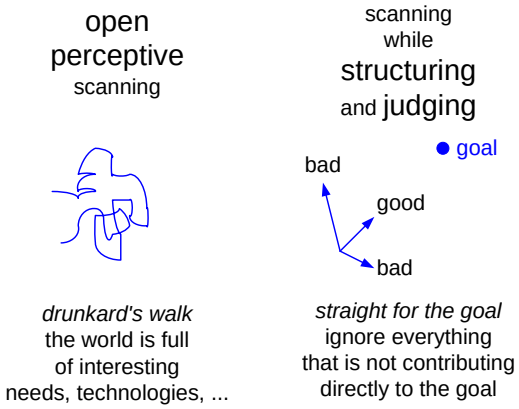


Figure 4: Two modes of scanning by an architect

Both axis span a tremendous dynamic range, creating a huge space for exploration. Systematic scanning of this space is way too slow. An architect is using two techniques to scan this space, that are quite difficult to combine: open perceptive scanning and scanning while structuring and judging. The open perceptive mode is needed to build understanding and insight. Early structuring and judging is dangerous because it might become a self-fulfilling prophecy. The structuring and judging is required to reach a result in a limited amount of time and effort. See figure 4 for these 2 modes of scanning.

The scanning approach taken by the architect can be compared with *simulated annealing methods* for optimization[3]. An interesting quote from this book, comparing optimization methods:

> Although the analogy is not perfect, there is a sense in which all of the minimization algorithms thus far in this chapter correspond to rapid cooling or quenching. In all cases, we have gone greedily

for the quick, nearby solution: From the starting point, go immediately downhill as far as you can go. This, as often remarked above, leads to a local, but not necessarily a global, minimum. Nature's own minimization algorithm is based on a quite different procedure...

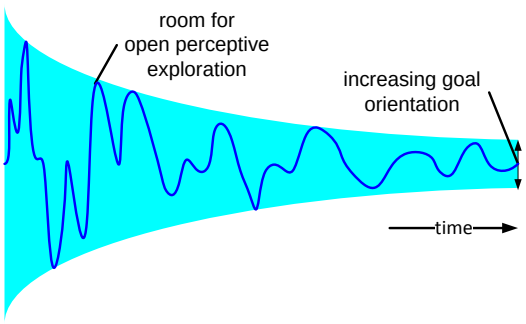


Figure 5: Combined open perceptive scanning and goal-oriented scanning

See also Figure 5 for the combined scanning path. The perceptive mode is used more early in the project, while at the end of the project the goal oriented mode is dominant.

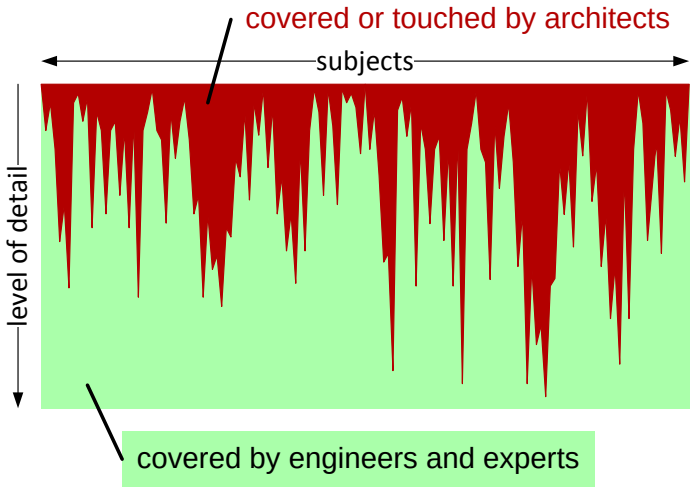


Figure 6: The final coverage of the problem and solution space by architect and engineers

The coverage of the problem and solution space is visualized in Figure 6. Note that the area covered or touched by the architect(s) is not exclusively covered,

engineers will also cover or touch that area partially. The architect needs experience to learn when to dig deeper and when to move on to next subjects. Balancing depth and breadth is still largely an art.

3 Quantification

The architect is continuously trying to improve understanding of problem and solution. This understanding is based on many different interacting insights, such as functionality, behavior, relationships et cetera. An important factor in understanding is the **quantification**. Quantification helps to get grip on the many vague aspects of problem and solution. Many aspects can be quantified, much more than most designers are willing to quantify.

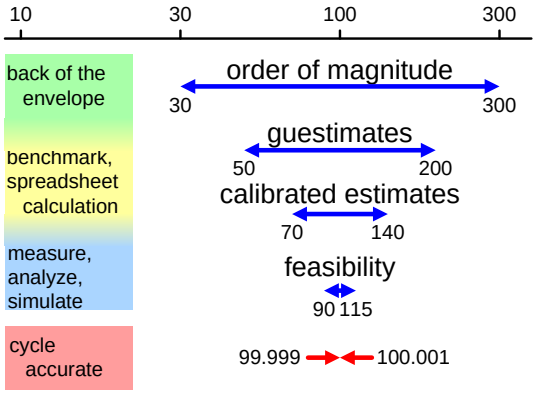


Figure 7: Successive quantification refined

The precision of the quantification increases during the project. Figure 7 shows the stepwise refinement of the quantification. In first instance it is important to get a feeling for the problem by quantifying orders of magnitude. For example:

- How large is the targeted customer population?
- What is the amount of money they are willing and able to spend?
- How many pictures/movies do they want to store?
- How much storage and bandwidth is needed?

The order of magnitude numbers can be refined by making back of the envelop calculations, making simple models and making assumptions and estimates. From this work it becomes clear where the major uncertainties are and what measurements or other data acquisitions will help to refine the numbers further.

At the bottom of figure 7 the other extreme of the spectrum of quantification is shown, in this example cycle accurate simulation of video frame processing results in very accurate numbers. It is a challenge for an architect to bridge these worlds.

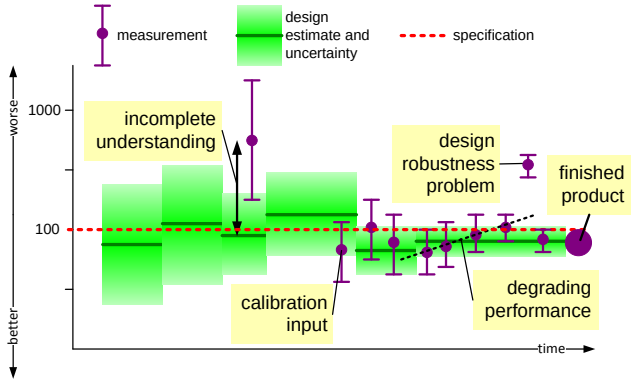


Figure 8: Example of the evolution of quantification in time

Figure 8 shows an example how the quantification evolves in time. The dotted red line represents the required performance as defined in the specification. The shaded area indicates the "paper" value, with its accuracy. The measurements are shown as dots with a range bar. A large difference between paper value and measurement is a clear indication of missing understanding. Later during the implementation continuous measurements monitor the expected outcome, in this example a clear degradation is visible. Large jumps in the measurements are an indication of a design which is not robust (small implementation changes cause large performance deviations).

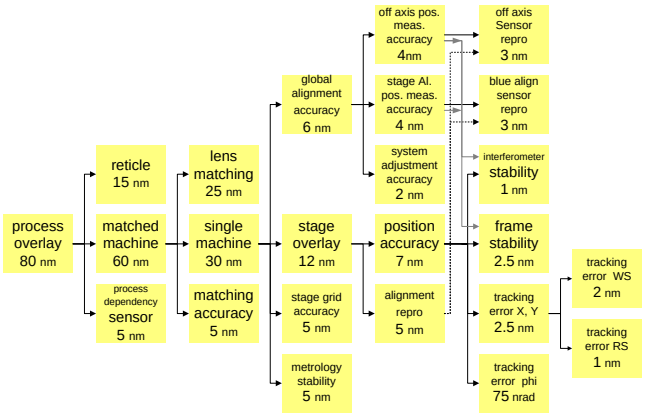


Figure 9: Example of a quantified understanding of overlay in a wafer stepper

Figure 9 shows a graphical example of an “overlay” budget for a wafer stepper. This figure is taken from the *System Design Specification* of the ASML TwinScan system, although for confidentiality reasons some minor modifications have been applied. This budget is based on a model of the overlay functionality in the wafer stepper. The budget is used to provide requirements for subsystems and components. The actual contributions to the overlay are measured during the design and integration process, on functional models or prototypes. These measurements provide early feedback of the overlay design. If needed the budget or the design is changed on the basis of this feedback.

4 Coping with uncertainty

The architect has to make decisions all the time, while most substantiating data is still missing. On top of that some of the available data will be false, inconsistent or interpreted wrong.

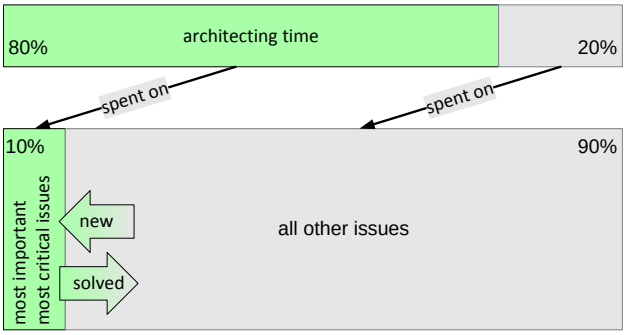


Figure 10: The architect focuses on important and critical issues, while monitoring the other issues

An important means in making decisions is building up insight, understanding and overview, by means of structuring the problems. The understanding is used to determine important (for the product use) and critical (with respect to technical design and implementation) issues. The architect will pay most attention to these *important* and *critical* issues. The other issues are monitored, because sometimes minor details turn out to be important or critical issues. Figure 10 visualizes the time distribution of the architect: 80% of the time is spent on 10% of the issues.

The architect will, often implicitly, work on the basis of a top 10 issue list, the ten most relevant (important, urgent, critical) issues. Figure 11 shows an example of such a “worry”-list.

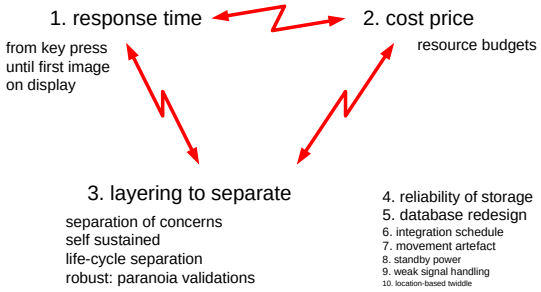


Figure 11: Example worry list of an architect

5 Acknowledgements

The team of composable architectures, with the following members Pierre America, Marcel Bijsterveld, Peter van den Hamer, Jürgen Müller, Henk Obbink, Rob van Ommering, and William van der Sterren within Philips Research provided valuable feedback for the original article.

References

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

[2] Gerrit Muller. https://gaudisite.nl/basicworkingmethodarchitectpaper.pdf. `https://gaudisite.nl/BasicWorkingMethodArchitectPaper.pdf`, 2011.

[3] William H. Press, William T. Vetterling, Saul A. Teulosky, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, England, 1992. Simulated annealing methods page 444 and further.

History

Version: 0, date: May 25, 2026 changed by: Gerrit Muller

- Created as subset from BWMA