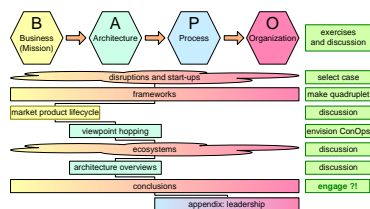


Tutorial How to Engineer Systems as Start-up or in Disruptive Circumstances?



Gerrit Muller

USN-SE and TNO-ESI

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

gaudisite@gmail.com

Abstract

Start-up companies large organizations may have similar needs, when disruptive developments take place. This tutorial guides the participants to a highly iterative systems engineering process with the continuous challenge to do just enough. The purpose is to facilitate the required level of responsiveness, while at the same time bring sufficient anticipation of the many enabling systems that need development.

Copyright © 2026 by Gerrit Muller. Published and used by INCOSE with permission.

Distribution

This article or presentation is written as part of the Gaudí project. The Gaudí project philosophy is to improve by obtaining frequent feedback. Frequent feedback is pursued by an open creation process. This document is published as intermediate or nearly mature version to get feedback. Further distribution is allowed as long as the document remains complete and unchanged.

All Gaudí documents are available at:
<http://www.gaudisite.nl/>

Contents

1	Introduction	1
2	Introduction How to Engineer Systems as Start-up or in Disruptive Circumstances?	3
2.1	Introduction	3
2.2	Examples of Disruptive Events	5
2.3	Light-Weight Architecting	6
3	What Is Innovation?	8
3.1	Innovation	8
4	Frameworks for Architecting	11
4.1	Introduction to frameworks	11
4.2	The BAPO framework	11
4.3	The quadruplet framework for architecture descriptions	12
4.4	A framework covering the spectrum social-technical	13
5	Market Product Life Cycle Consequences for Architecting	15
5.1	Introduction	15
5.2	Observed Lifecycle Curve in Practice	16
5.3	Life Cycle Model	17
5.4	Acknowledgements	19
6	Architectural Fast Viewpoint Hopping	20
6.1	Introduction	20
6.2	Viewpoint hopping	21
6.3	Quantification	24
6.4	Coping with uncertainty	26
6.5	Acknowledgements	27
7	Introduction to Ecosystems	28
7.1	Introduction	28

7.2	Enabling systems	30
7.3	Scope of Control	31
7.4	Layers of Ecosystems	31
8	Stakeholders, from Abstract to Individual Humans	33
8.1	Introduction to Stakeholders from abstract to humans	33
8.2	Abstracting Stakeholders	34
8.3	Introduction to Stakeholders from abstract to humans	36
8.4	An approach to engage stakeholders	39
8.5	Acknowledgements	39

Chapter 1

Introduction

Start-up companies operate in an environment, where their market is not clear, nor their business model. At the same time, the contours of their solutions are still shaping up and the technology, production and delivery process and the supply are all under development. In other words, everything is moving simultaneously. Start-ups need persistence and focus to spend their limited resources well, while they also need to be open minded and responsive to market and business needs. How can start-ups use the systems engineering knowledge in a way that it fits this context?

Interestingly, large organizations may have similar needs, when disruptive developments take place. A clear example is defense, where drones and electronic warfare disrupt existing doctrines. This tutorial guides the participants to a highly iterative systems engineering process with the continuous challenge to do just enough. The purpose is to facilitate the required level of responsiveness, while at the same time bring sufficient anticipation of the many enabling systems that need development.

Figure 1.1 shows the flow of the tutorial.

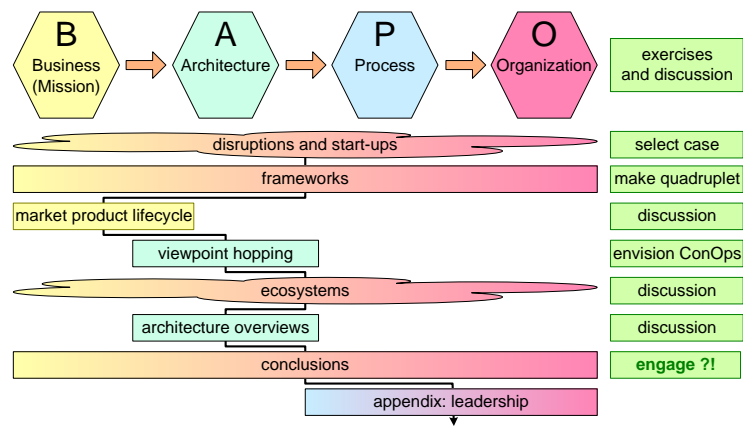
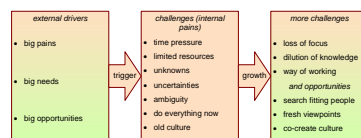


Figure 1.1: Flow of the tutorial

Chapter 2

Introduction How to Engineer Systems as Start-up or in Disruptive Circumstances?



2.1 Introduction

The common drivers for start-ups and large organizations facing disruptive events is that big pains form external drivers. When there are big pains, there are big needs; where there are big needs, there are big opportunities. Figure 2.1 shows the external drivers, e.g. pains, needs, and opportunities.

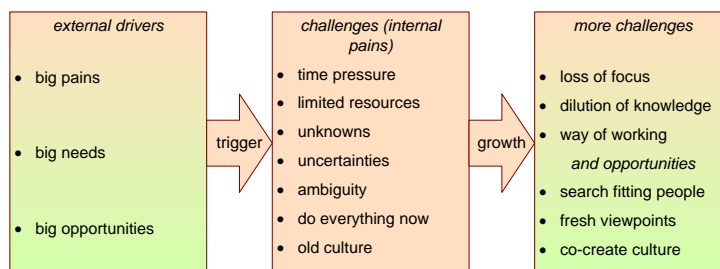


Figure 2.1: Disruptive Change Consequences

The drivers from the external world cause challenges for the organization(s)

that must respond to the drivers. These challenges are internal pains:

Time pressure caused by the disruptive event. Note that disruptive events may have a long lead-time that become acute after decision makers ignoring it too long.

Limited resources ; in large organizations, the resources are occupied with the ongoing operational work. In start-ups, the resources aren't available yet

Unknowns caused by entering new territory, somewhere where humanity hasn't been yet

Uncertainties with the same cause as the unknowns

Ambiguity due to the pluriformity of the stakeholders, and the stakeholders themselves being in this new territory

Do everything now demands; the circumstances demand that a new capability needs to be available now with at the same time all enabling capabilities (e.g. volume production, secure and robust supply chains, training internal and external, maintenance, ...), while we also need to cope with feasibility risks.

Old culture , especially in large organizations, which may not fit the new needs.

Addressing many of the internal pains tend to require a fast growth of the organization and its ecosystem. Growth brings more challenges, however, also creates opportunities. Major challenges are:

Loss of focus on top of the problem of doing everything now for the system-of-interest, for example, recruiting resources, arranging housing and infrastructure, and adapting organization and process.

Dilution of knowledge since new resources lack the application knowledge.

Way of working that continuously needs adaptation to fit the growing organization

Opportunities are:

Search fitting people that probably will differ from the people that the organization already has.

Fresh viewpoints from new people.

Co-create culture rather than unfreeze a population that is settled.

The problems, challenges, and opportunities raise questions that Figure 2.2 shows.

- How to keep **focus**?
- How to cope with **chaos** and to maintain **overview**?
- How to **discover unknowns**?
- How to **reduce uncertainties** and ambiguity?
using **time** and **resources wisely**
- How to **lead others** in the organization?
- How to **engage external stakeholders**?
- How to **orchestrate** the **ecosystem**?

Figure 2.2: Questions that the problems, challenges, and opportunities raise

2.2 Examples of Disruptive Events

Figure 2.3 shows several examples of disruptive events in a few domains: defense (neglect and dependence), health care (infarct) and the climate and environment (emergency). There are many more, e.g. housing, food, and democracy, to name a few more.

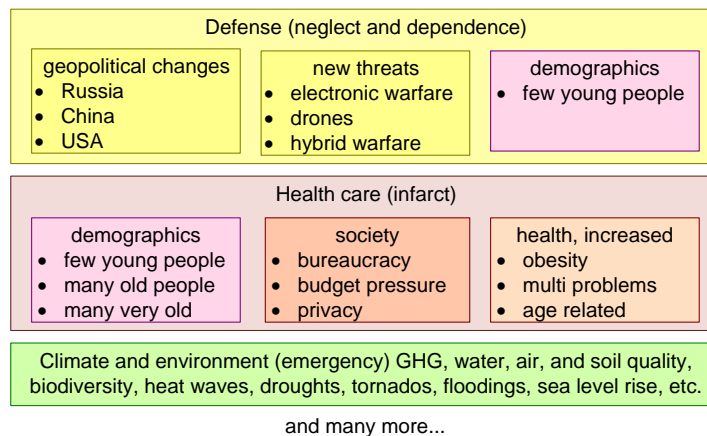


Figure 2.3: Examples of disruptive events

In defense geopolitical changes, e.g. Russia's aggression, USA withdrawal, and China's overwhelming development change the geopolitical landscape much. Technical developments in materials, sensors, effectors, computing power, data processing and analysis (wrongly named AI), control, and communication enable remotely

controlled operation of drones in air, under water, above water, and on land. These systems come with new capabilities and weaknesses, where electronic warfare palys dominantly, influencing communication, and positoning. The defense organization needs to revise most of iots doctrines, and consequently, needs to retrain and reorganize itself completely, Meanwhile, most countries have much less young people available that will staff the defense organziation.

In Healthcare, demograohics plays a dominating role too. The population will have much less young people, while there are many old and very old people. At the same time, the population is less healthy with many more obese people. The older people tend to have multiple health problems, complicatin the care. They also suffer from age-related health issues, e.g, cardio-vascular, cancer, and dementia. Society amd thegovernemnt have increased the bureaucracy in an attempt to control cost and quality (and to cover asses, when there are problems). Most western countries spend a major part of their GDP on hekath care. Increasing privacy regulations and expectations complicate the care further.

Climate and environment can fill a series of books, I will nto elaborate them here.

2.3 Light-Weight Architecting

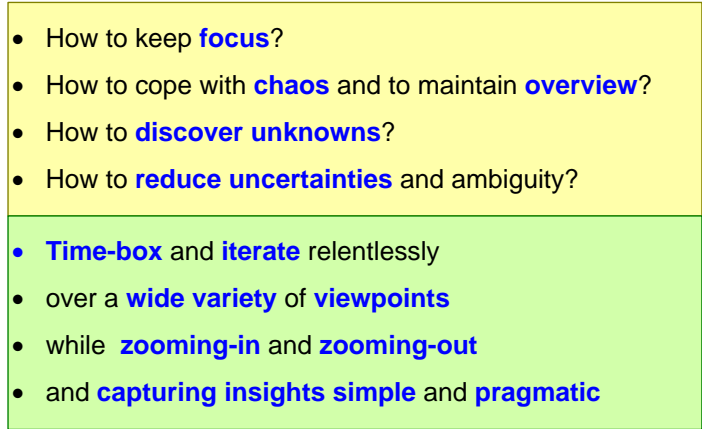
- 
- How to keep **focus**?
 - How to cope with **chaos** and to maintain **overview**?
 - How to **discover unknowns**?
 - How to **reduce uncertainties** and ambiguity?
 - **Time-box** and **iterate** relentlessly
 - over a **wide variety** of **viewpoints**
 - while **zooming-in** and **zooming-out**
 - and **capturing insights simple** and **pragmatic**

Figure 2.4: First Answer: Use Light-Weight Architecting

Figure 2.4 shows my assertion that li8ght-weight architecting is an answer to the first set of questions to keep focus, maintain overview, discover unknwons, and to reduce uncertainties and ambiguity. Light-weight architecting requires time-boxing and iterating relentlessly over a wide variety of viewpoints, while zooming-in and zooming-out and capturing insights simple and pragmatic.

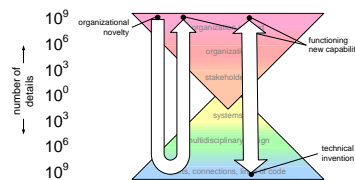
- How to **lead others** in the organization?
 - How to **engage external stakeholders**?
 - How to **orchestrate** the **ecosystem**?
- Using **leadership** skills bridging **ratio** and **feelings**
 - to **manage self** (which requires knowing yourself well)
 - and to **connect to others** emotionally as well as rationally
 - such that interaction happens in **learning mode**

Figure 2.5: Answer Part 2: Use Light-Weight Archiecting

Figure 2.5 shows leadership as the answers to the remaining questions. Using leadership skills bridging ratio and feelings to manage self (which requires knowing yourself well) and to connect to others emotionally as well as rationally such that interaction happens in learning mode

Chapter 3

What Is Innovation?



3.1 Innovation

In many organizations the holy grail of strategy is *innovation*. *Innovation* is a fundamental way to increase the value proposition to the market. Companies have a continuous need for a better value proposition in a world with constant pressure on the margin. The alternative to maintain the margin at a healthy level is to reduce cost levels.

Most (mature) organizations achieve the desired improvement of the value proposition by repetitive small improvement steps. However, many small steps often do not open new markets, or create new applications. *Innovation* is the result of a creative effort both in the technology side, as well as the application and marketing side.

Figure ??WII diabolos shows that I see innovation as a fully functioning capability in the real world. It should function in the real world in a wide variety of circumstances. [?]www:DynamicRangeAbstractionLevels introduces the levels of details that form the background for two types of innovation:

organizational novelty that with appropriate technical support enables an organizational innovation

technical inventions that when implemented in a system and deployed in an organization results in a functioning new capability.

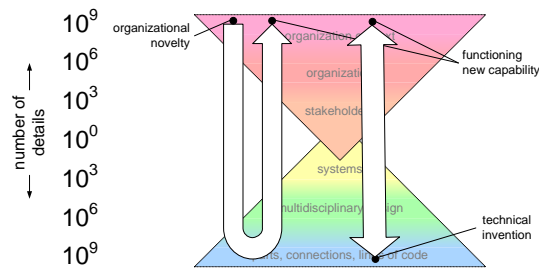


Figure 3.1: Innovations should function in the real world

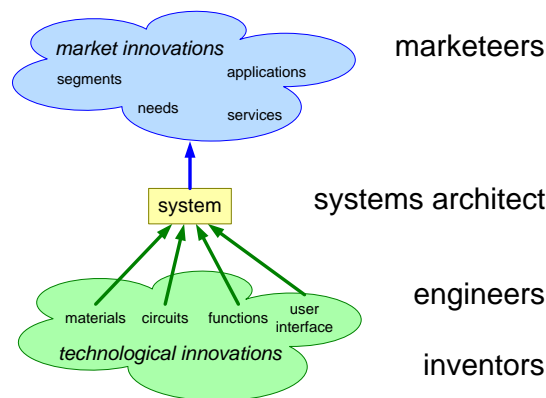


Figure 3.2: Innovations require all major contributors

Figure 3.2 shows that truly innovative technology people (“inventors”), engineers, architects, organizers, and marketers need to collaborate in a concerted effort.

New fully functioning capabilities require both fully functioning enabling and complementing systems to be fully scalable and sustainable. For example, enabling systems are a functioning supply chain, logistics support, maintenance staffing, tools, procedures, et cetera, while complementing systems provide the capability with required energy, data, and communication infrastructure and operation.

There is a tension between processes and management and innovation. The inherent nature of innovation is to go beyond today’s limitations, while processes and management also tend to enforce limitations. Innovation requires inspiration rather than control. This same tension can also be observed in the architecting role. Many architects are used to identify and mitigate risks, a valuable contribution to product creation. However, the risk based focus can be a severe limitation when searching for innovative solutions.

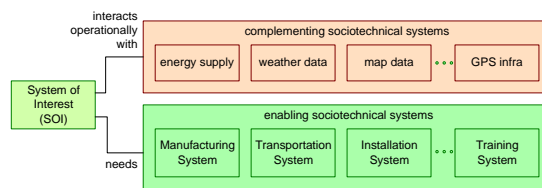
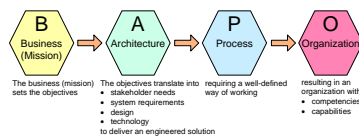


Figure 3.3: A functioning new capability requires both fully functioning enabling and complementing systems

Chapter 4

Frameworks for Architecting



4.1 Introduction to frameworks

Figure 4.1 shows the frameworks this paper is using.

BAPO is the overarching framework, connecting Business, Architecture, Process, and Organization

Quadruplet elaborating Business and Architecture

The social to technical aspects elaborating Process and Organization

4.2 The BAPO framework

BAPO stands for Business, Architecture, Process, and Organization. The order of these views is relevant:

- The Business drives the architecture that guides the engineered solution.
- The architecture requires a suitable way of working, which is captured in the processes.
- The processes fit in an organization with capabilities and requires individuals with specific competences.

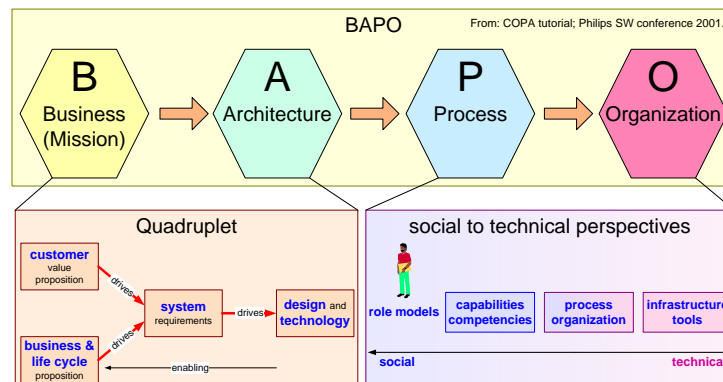


Figure 4.1: Frameworks for Architecting

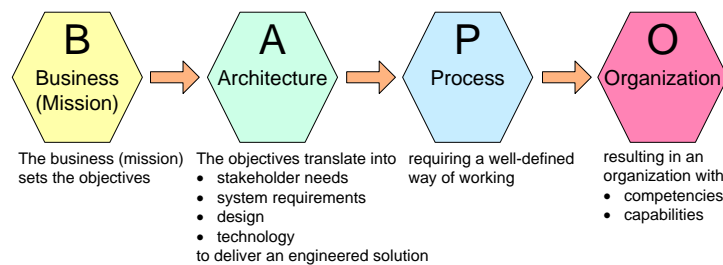


Figure 4.2: The Encompassing BAPO Framework

Figure 4.2 shows the BAPO framework.

Obbink et al. [7] created the BAPO framework. Later Henk Obbink suggested extending the framework to BAPOC by adding Culture as additional view. The thinking behind adding culture is that culture has much impact on individual and organizational behavior. Culture is challenging to change, which takes energy and time. Changing process and organization may seem easier, however, culture may limit its effect.

When using the framework in the defense domain, the term business is ill-fitting. In defense a term like mission or main task, capturing the main objective of defense is more suitable.

4.3 The quadruplet framework for architecture descriptions

The quadruplet is a framework to guide the creation and capturing of architecture descriptions. The core of the framework is that the customer value and the business

and lifecycle propositions drive the definition of the system, which in turn drives the design and technology choices. Figure 4.3 shows the quadruplet framework.

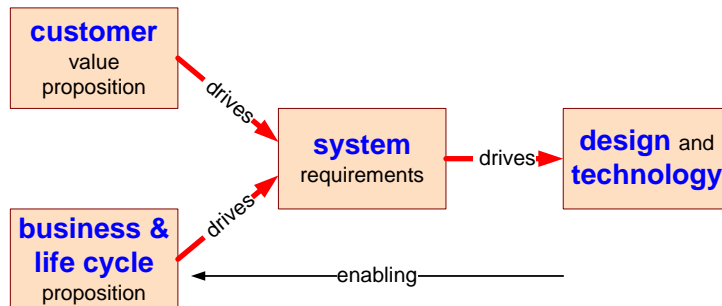


Figure 4.3: The Customer and Business Propositions are Driving

The quadruplet framework is building on the CAFCR+ framework; see[4]:

- The CA-views map on the customer value proposition.
- The F-view maps on the system requirements.
- The CR-views map on the design and technology
- The +-view maps on the business and life cycle.

The quadruplet simplifies CAFCR+ by combining C and A, and C and R views. The simplification helps in some of the ways of using the framework. The adapted visualization order expresses the driving logic better. This improved order balances the customer and business and lifecycle propositions better.

4.4 A framework covering the spectrum social-technical

Figure 4.4 shows a spectrum from social to technical with 4 main views:

Role models are essential to help people copying desired behavior

Capabilities and competencies that need development. Organizational capabilities build upon individual competencies. A capable team is more than a group of competent individuals.

Process and organization is the structure that we agree upon to work together. Processes are the dynamic behavior, while the organization is the static structure, e.g. what groups with what relations.

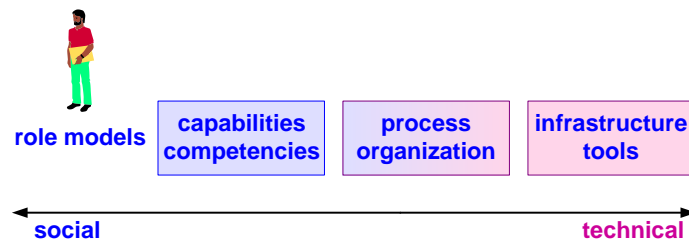


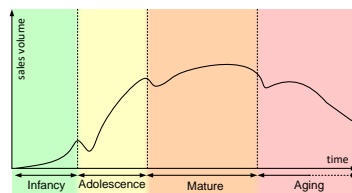
Figure 4.4: Change Requires Addressing the Full Spectrum

Infrastructure and tools Infrastructure and tools form the most technical components. They support the processes and organization to enable the desired capabilities.

The framework showing a spectrum from social to technical finds its origin in change and transformation programs. Organizational transformations require attention over the full spectrum from social to technical. This may seem obvious; however, I have seen amazingly many transformations addressing only one of these views.

Chapter 5

Market Product Life Cycle Consequences for Architecting



5.1 Introduction

A class of products serving a specific market evolves over time. The sales volume of these products reflecting this evolution. The systems architecting approach depends where products are in this evolution.

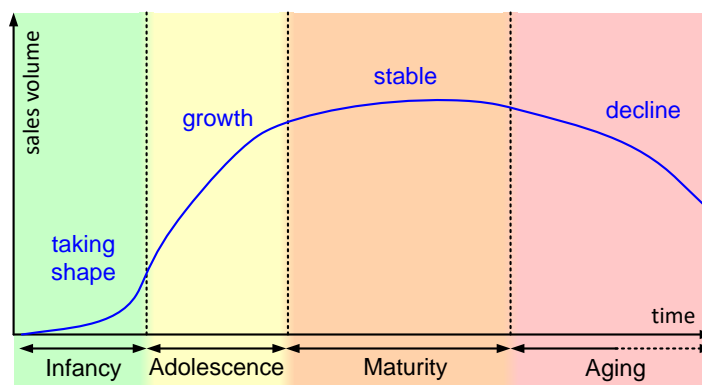


Figure 5.1: Compared with ideal bathtub curve

The life cycle of a product market combination can be visualized by showing the sales volume as a function of the time. In literature the form of the curve of the sales volume as function of the time is described as bathtub, see Figure 5.1. It is customary to recognize four phases in this curve:

- The life cycle starts with very small sales in the **infancy** phase, where the product finds its shape.
- A fast-increasing sales volume in the **adolescent phase**.
- A more or less stable sales volume in the **mature** phase.
- A decreasing sales volume in the **aging** phase.

The curve and its phases represent the theoretical evolution. In the next paragraphs we will discuss observations in practice and an explanation, and we will show that the class of products and the market themselves also evolve on a macro scale.

5.2 Observed Lifecycle Curve in Practice

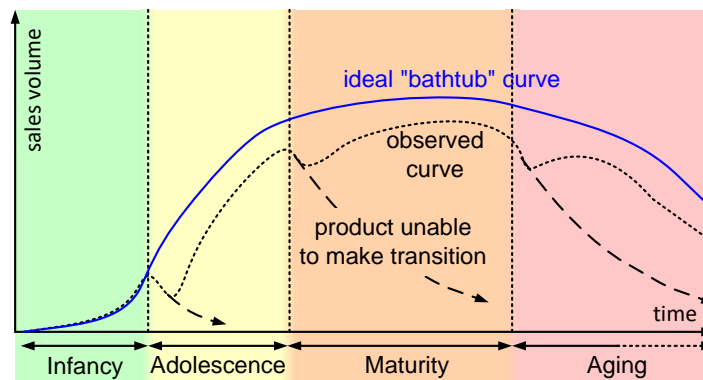


Figure 5.2: Market product lifecycle phases

Henk Obbink (Philips Research) observed dips in the sales volume, as shown in figure 5.2. The transition from one phase to the next does not seem to happen smoothly. In some cases, the sales drops further and the product does not make the transition at all.

The hypothesis for the dips in the curve is that characteristics of all stakeholders are different for the different life cycle phases. If an organization is not adopting the way of working to these changes, then a mismatch with the changed circumstances results in decreasing sales. Figure 5.2 also indicates that, if no adaptation to the

change takes place, that the sales might even drop to zero. Zero sales effectively is killing the business, while still plenty of market opportunity is present.

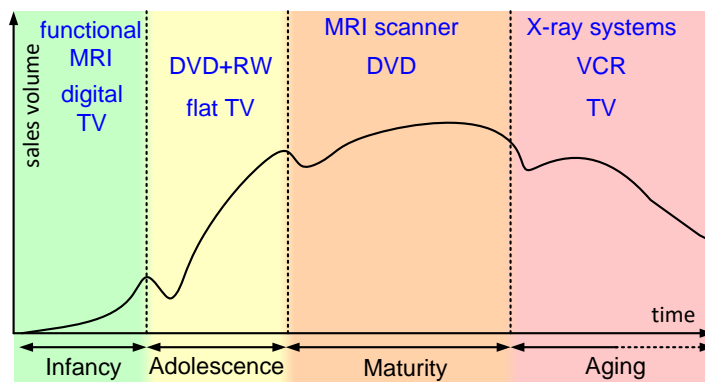


Figure 5.3: Examples of product classes on the curve

Figure 5.3 annotates the life cycle graph with a number of products and their positioning in the life cycle. As can be seen products can move backwards in the phases (i.e. become “younger”) by the addition of innovative features. For instance MRS scanners moved backwards when *functional imaging* was added, an innovative way to visualize the activity of specific tissues. Similarly, conventional televisions rejuvenated multiple times by adding digital processing, flat screens, and digital interfaces.

5.3 Life Cycle Model

Figure 5.4 shows typical attributes of the life cycle phases.

Uncertainty about the customer needs, and therefore the product requirements is characterizing the *infancy* phase. Essential is that the creator/producer is responsive to the customer needs, which will provide insight in needs and requirements. The way of working in this phase reflects the inherent uncertainty, the chaotic development, and the innovative and pioneering mind set. Product cost is still less of an issue; the risk related to the uncertainty is the dominant concern. The design copes with the uncertainty by over-dimensioning those aspects which are perceived to be the most uncertain.

The *adolescent* phase is characterized by strong (exponential) growth of the sales volume, concurrent with an increase in performance, features and product variants. The challenge is to cope with this strong growth in many dimensions. With respect to the requirements a strategic selection is needed, to serve the growing customer base, without drowning in an exploding complexity. The technical and process challenge is to scale up in all dimensions at the same time. Up-scaling

	Infancy	Adolescence	Mature	Ageing
Driving factor	Business vision		Stable business model	Harvesting of assets
Value from	Responsiveness	Features	Refinements / service	Refining existing assets
Requirements	Discovery	Select strategic	Prioritize	Low effort high value only
Dominant technical concerns	Feasibility	Scaling	Legacy Obsolescence	Lack of product knowledge Low effort for obsolete technologies
Type of people	Inventors & pioneers	Few inventors & pioneers "designers"	"Engineers"	"Maintainers"
Process	Chaotic		Bureaucratic	Budget driven
Dominant pattern	Overdimensioning	Conservative expansion	Midlife refactoring	UI gadgets

Figure 5.4: Attributes per phase

the Customer Oriented Processes and the Product Creation Process requires more shared structure between the participants. This involves a mind set change: less inventors, more designers. The design pattern used frequently in this phase is conservative extension of a base design.

The *mature* phase is characterized by more stability of the business model and the market, while the market has become much more cost sensitive. Instead of running along in the feature race more attention is required to optimize the specification and development choices. The value can be shifting from the core product itself to services and complements of the product, while the features of the product are mostly refined. The age of the product starts to interfere with the business, obsolescence problems occur, as well as legacy problems. Innovative contributions become counterproductive, more rigid engineers are preferred above creative designers. The cost optimization is obtained by process optimization, where the processes also become much more rigid, but also more predictable, controllable and executable by a large community of less educated engineers. The design copes with the aging technology by performing limited refactoring activities in areas where return on investment is still likely.

The *aging* phase is often the phase where the product is entirely seen as cash cow, maximize the return on (low) investments. This is done by searching all the low effort high value requirements, resulting mostly in small refinements to the existing product. Often the integral product know how and even specialist know how has been lost. Only very important obsolescence problems are tackled. Again the mind set of the people working on the product is changing to become more maintenance oriented. Cost is a very dominating concern, budgets are used to

control the cost. Many changes are cosmetic or superficial, taking place in the most visible parts of the product: the user interface and the outer packaging.

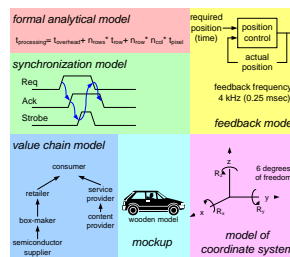
5.4 Acknowledgements

Henk Obbink observed the discontinuity of market success at the phase transitions. The analysis of this phenomenon was carried out by Jürgen Müller, Henk Obbink and Gerrit Muller.

Pierre America improved the layout of the diagrams.

Chapter 6

Architectural Fast Viewpoint Hopping



6.1 Introduction

This section is a selection of [6]. 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 6.4.
- 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.

6.2 Viewpoint hopping

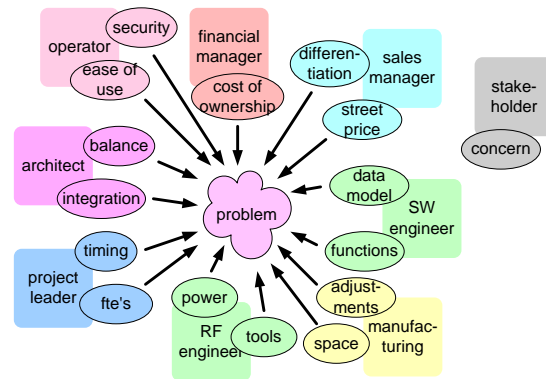


Figure 6.1: Small subset of viewpoints

The architect is looking towards problems and (potential) solutions from many different viewpoints. A small subset of viewpoints is visualized in Figure 6.1, where the viewpoints are shown as stakeholders with their concerns.

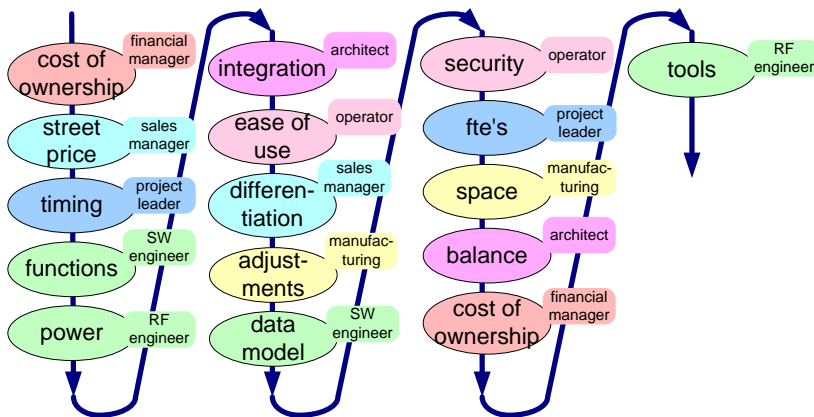


Figure 6.2: Viewpoint Hopping

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 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 6.2 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 thousands¹ of these viewpoint changes.

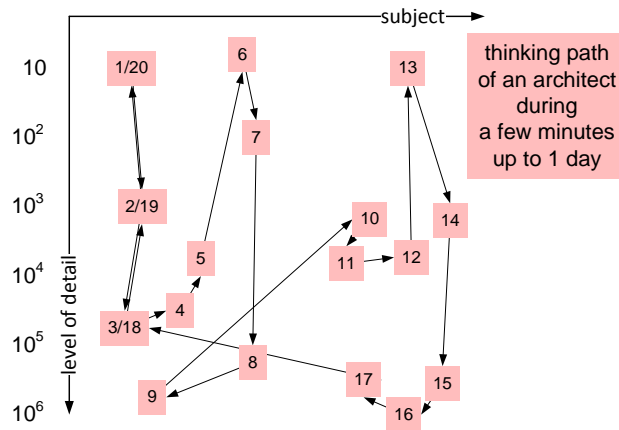


Figure 6.3: The seemingly random exploration path

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 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 6.3 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.

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

¹Based on observations of other architects and own experience.

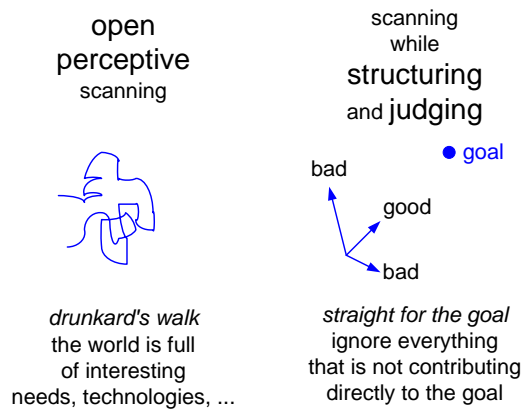


Figure 6.4: Two modes of scanning by an architect

judging is required to reach a result in a limited amount of time and effort. See figure 6.4 for these 2 modes of scanning.

The scanning approach taken by the architect can be compared with *simulated annealing methods* for optimization[8]. 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...

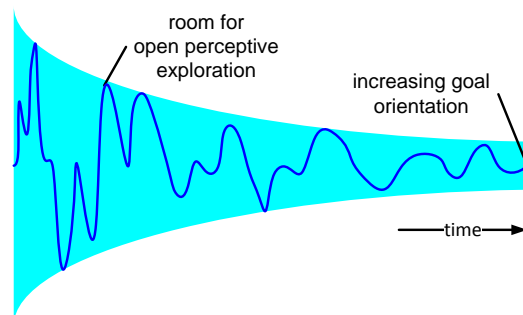


Figure 6.5: Combined open perceptive scanning and goal-oriented scanning

See also Figure 6.5 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.

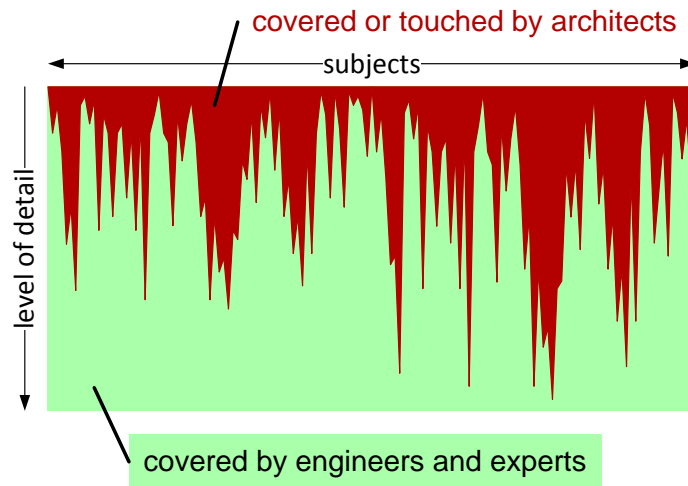


Figure 6.6: The final coverage of the problem and solution space by architect and engineers

The coverage of the problem and solution space is visualized in Figure 6.6. 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.

6.3 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.

The precision of the quantification increases during the project. Figure 6.7 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?

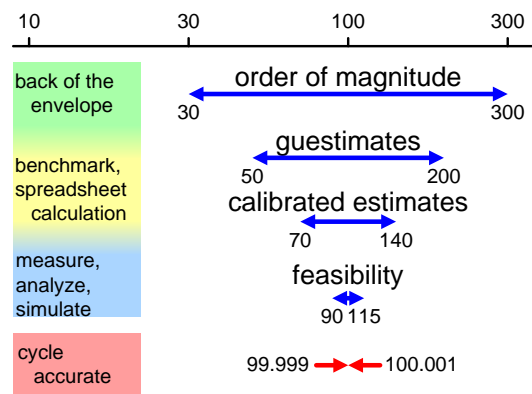


Figure 6.7: Successive quantification refined

- 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 6.7 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.

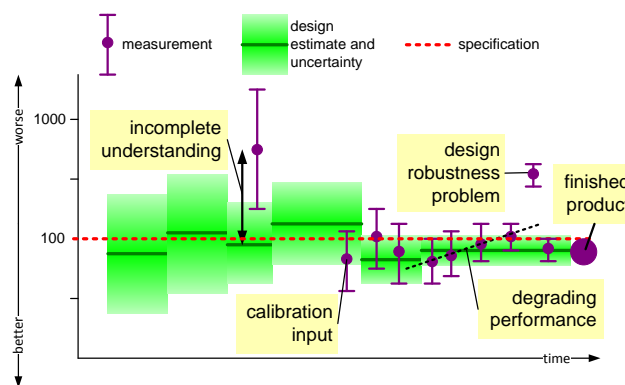


Figure 6.8: Example of the evolution of quantification in time

Figure 6.8 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).

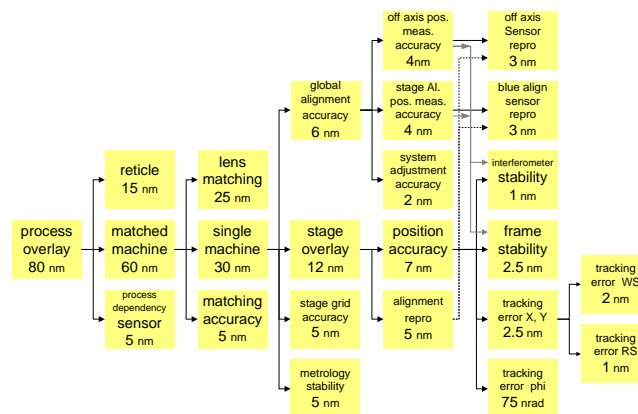


Figure 6.9: Example of a quantified understanding of overlay in a wafer stepper

Figure 6.9 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.

6.4 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.

An important means in making decisions is building up insight, understanding and overview, by means of structuring the problems. The understanding is used

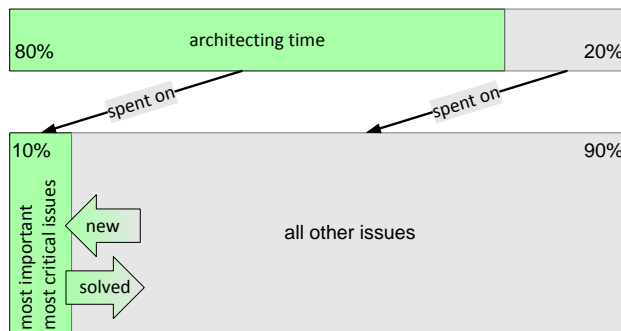


Figure 6.10: The architect focuses on important and critical issues, while monitoring the other issues

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 6.10 visualizes the time distribution of the architect: 80% of the time is spent on 10% of the issues.

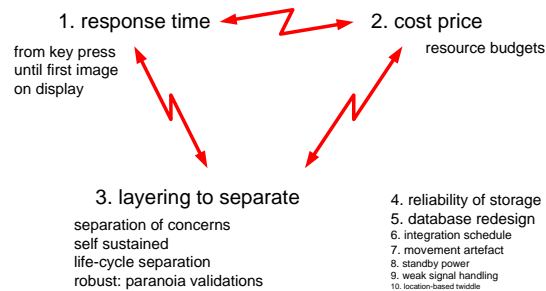


Figure 6.11: Example worry list of an architect

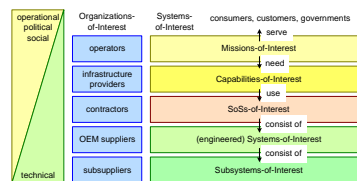
The architect will, often implicitly, work on the basis of a top 10 issue list, the ten most relevant (important, urgent, critical) issues. Figure 6.11 shows an example of such a “worry”-list.

6.5 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.

Chapter 7

Introduction to Ecosystems



7.1 Introduction

Capabilities and services that we use at societal level are the result the work of many organizations. We simplify the classification of the organizations contributing to the capabilities and services to a 5 layer model, as Figure 7.1 shows. The five layers are

- operators delivering the actual capabilities
- infrastructure providers that manage the assets that the operators are using
- contractors that build, adapt, and integrate the systems of system that together form the infrastructure
- Original Equipment Manufacturers (OEM) suppliers that develop individual systems that the contractors further integrate
- sub-suppliers developing and delivering products and components to the OEM suppliers

The way of procuring from layer to layer shifts from an acquisition of solutions gradually into procurement of standardized catalogue products and components.

The focus per layer is shifting. Figure 7.2 shows the focus per layer. The higher the layers, the more operational, the deeper in the stack, the more technical

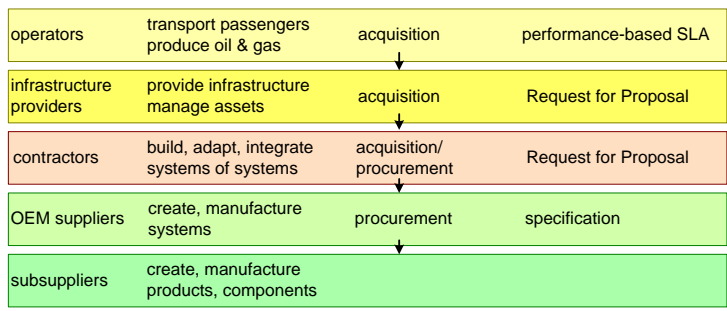


Figure 7.1: Characterization of Domain Layers

the focus becomes. The figure shows a few examples from the rail transportation domain. Operators are inherently operationally oriented. Examples are operators in The Netherlands, such as NS and Arriva. Infrastructure operators are asset oriented. ProRail is in the Netherlands responsible for the rail infrastructure with a worth of many Billions with a yearly turnover close to 2 billion Euro. Contractors are construction project oriented. OEM suppliers have a development project or product focus, while sub-suppliers have a product or component focus. Companies may operate at multiple levels, although these tend to organizationally fit in different silos in that company.

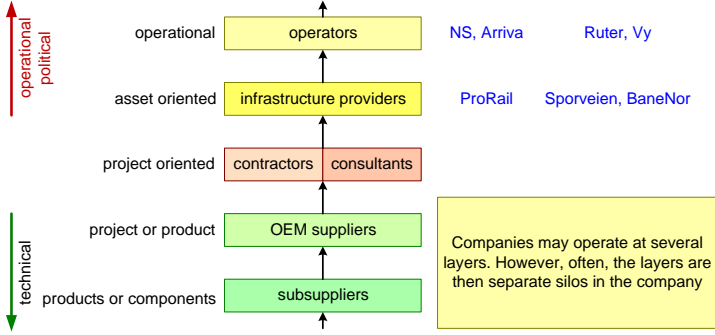


Figure 7.2: Positions in the Value Network Differ in Nature

Figure 7.3 summarizes these layers in terms of the organizations-of-interest and systems-of-interest: Operators serve consumers, customers, and governments with missions-of-interest. Operators need capabilities-of-interest that infrastructure providers provide, using SoSs-of-interest that operators deliver. The SoSs consist of (engineered) systems and SoSs that OEM suppliers develop and deliver. Sub-suppliers develop and deliver the subsystems-of-interest. High in the layers, the main concerns are

operational, political, and social, while deep in the layers, the main concern is technical. Economic and legal concerns play in all layers.

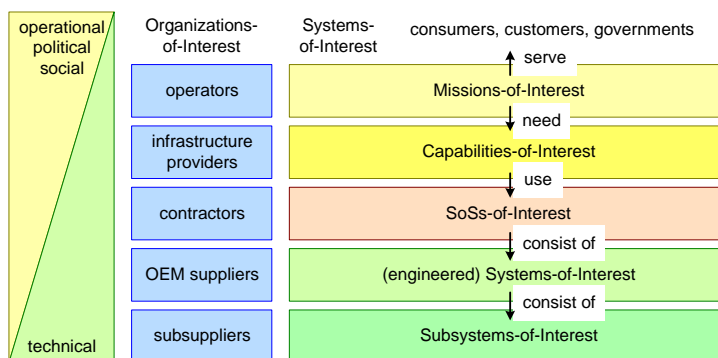
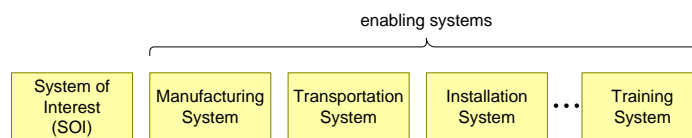


Figure 7.3: Perspective Changes from Layer to Layer

7.2 Enabling systems

Systems engineering teaches us that when we develop a system-of-interest (SoI), that we then also have to develop the related enabling systems. Figure 7.4 shows some examples of enabling systems, such as the manufacturing, transportation, installation and training systems. This list is far from complete. An essential enabling system is the entire supply chain, including its supporting structures, such as contracts, second suppliers, et cetera.



Development and Engineering must develop the System of Interest and all Enabling Systems

Figure 7.4: Development Has to Develop the System of Interest and its Enabling Systems

This applies recursively for each system when going down in the layers of the ecosystem.

7.3 Scope of Control

When developing products and components, the developing organization has more or less full control over the specification and design of the product. This changes fundamentally when we enter the Systems-of-Systems domain. SoSs inherently are independent, so, the SoS constructor and users don't have full control. The less cohesion there is between constituent systems in managerial sense, the lower the influence is anyone within the ecosystem has.

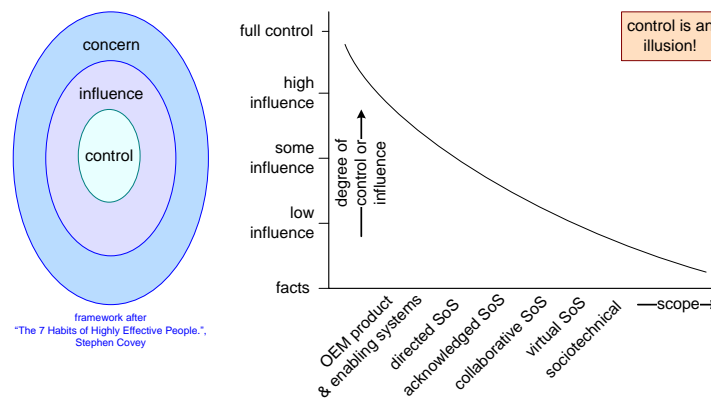


Figure 7.5: The Level of Influence Decreases with Scope

[2] provides a simple framework with a scope of control, a scope of influence, and a sphere of concern. Figure 7.5 shows the simple framework. The graph uses a more continuous scale on the vertical axis for the degree of influence. Be aware that full control is an illusion. The horizontal axis defines the system scope from small to large, also as continuum.

7.4 Layers of Ecosystems

Each layer of Figure 7.3 consists of an ecosystem itself. Figure 7.6 shows an elaboration of these layers. For example, an OEM company has suppliers (with their suppliers), partners offering complementary products and complementing services and systems. The OEM company will have competitors. Most of these organizations will interact with the customers of the OEM company, such as the contractor.

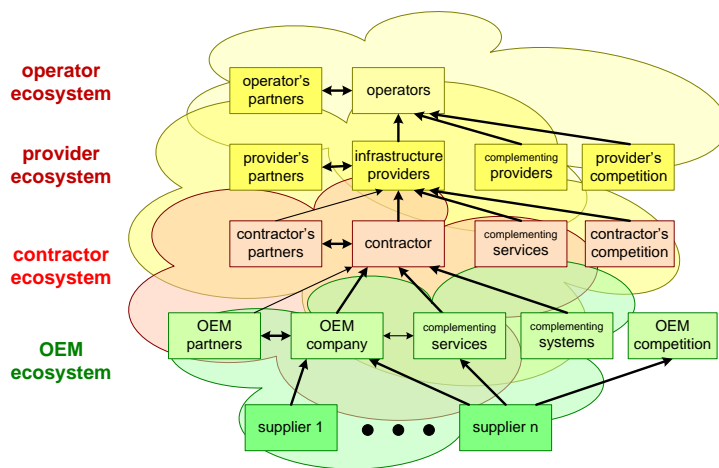
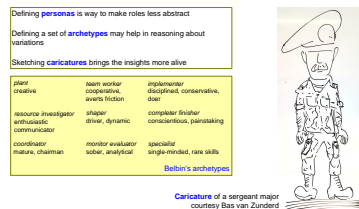


Figure 7.6: Each System and organization is part of its ecosystem

Chapter 8

Stakeholders, from Abstract to Individual Humans



8.1 Introduction to Stakeholders from abstract to humans

Any system of interest is interacting with its context. We distinguish several elements in the context:

- humans and organizations
- natural environment
- man-made artifacts

Figure 8.1 shows that all these elements interact within itself between them, and with the system of interest

Figure 8.2 elaborates the elements further. Stakeholders can be individual humans or organizational entities. Social, political, and psychological characteristics drive their behavior. That means that emotions play a significant role, which triggers a risk of ill-behaving stakeholders. This is in contrast with man-made artifacts, which are often engineered systems. Technical characteristics primarily determine the behavior. We assume rationally designed systems that will behave well when the engineers do their job well. Question is whether complex systems,

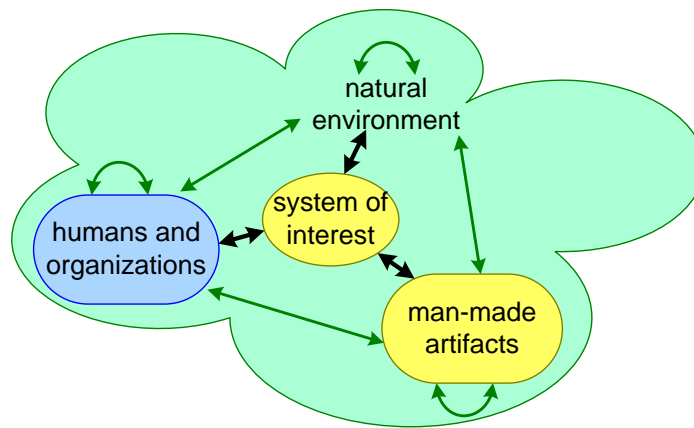


Figure 8.1: The context of a System-of-Interest

systems of systems, and sociotechnical systems will behave well. Do we understand our man-made artifacts well enough?

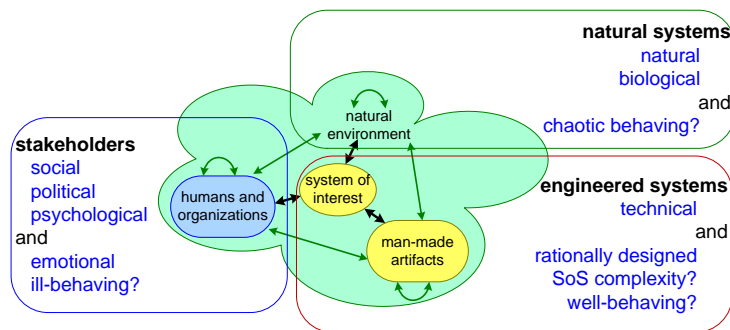


Figure 8.2: The context of a System-of-Interest

Natural systems in the environment may be biological. Nature and biological systems may behave chaotically from our perspective.

The overarching question is how well do we understand the context, so, how well can we understand the behavior of our system of interest in its context? This paper focuses on the stakeholders.

8.2 Abstracting Stakeholders

[5] explains that we can view a system at many levels of abstraction, e.g. without details or with more details. It visualizes this by using an exponential scale. Then

it shows that you can do the same with the system context. Figure 8.3 takes this one step further and shows stakeholder related aspects at the various abstraction levels.

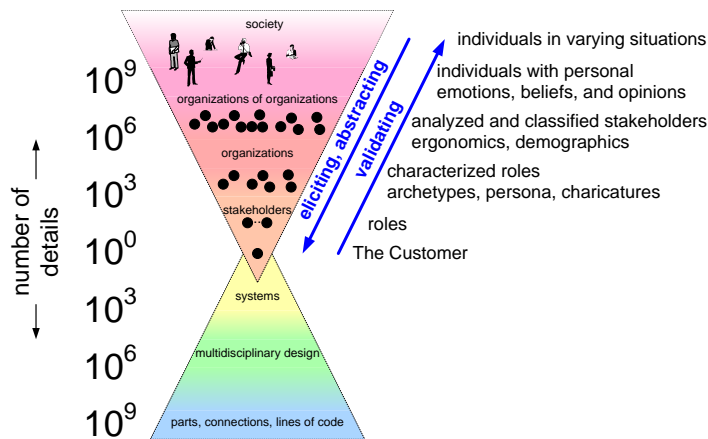


Figure 8.3: Abstraction from humans to roles

Figure 8.4 elaborates the upper pyramid further. When we start at societal level, we have billions of people with many relevant behaviors and characteristics. These are individuals in varying situations, where we can observe behaviors, pains, interests, concerns, patterns, with many variations, and exceptions. In a first abstraction step, we reduce variations and exceptions, by looking at the common patterns. In that way we get individuals with personal emotions, beliefs, and opinions.

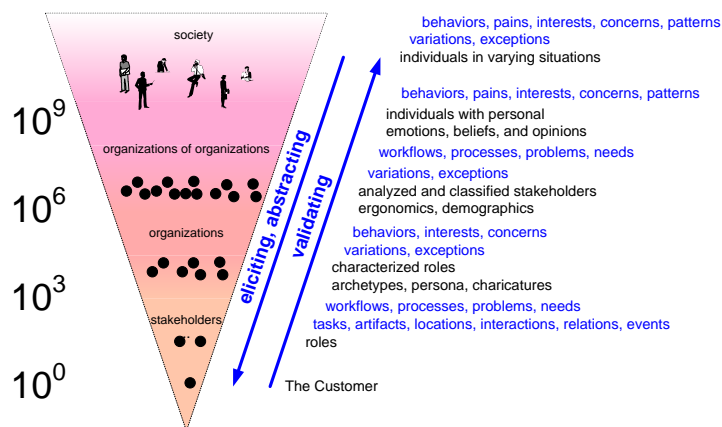


Figure 8.4: Elaborating what to observe

We can abstract the individuals into stakeholders, by observing workflows, processes, problems, needs with its variations and exceptions. The result is analyzed and classified stakeholders with ergonomics and demographics. Further observation of behaviors, interests, concerns with its variations and exceptions, resulting in characterized roles archetypes, persona, caricatures.

Once more abstracting results in the main workflows, processes, problems, needs, tasks, artifacts, locations, interactions, relations, and events. We then have simplified stakeholders in a limited set of roles (such as project leader, electrical engineer, marketing manager, etc.). Figure 8.5 shows an example of roles in acquirer and supplier, relevant for the acquisition and delivery of systems in a business-to-business setting. A common (over)abstraction is “the customer”.

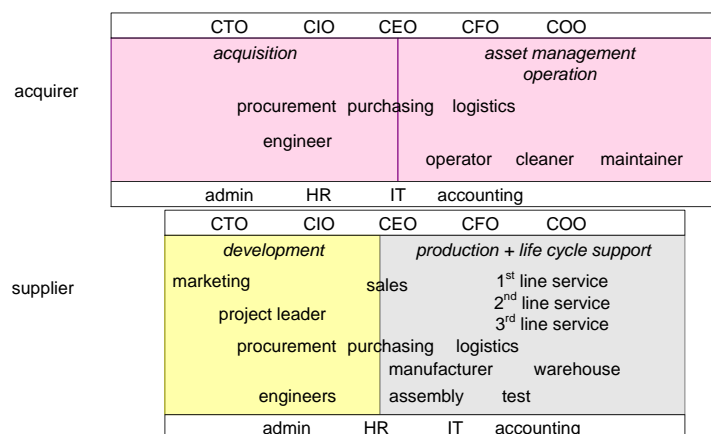


Figure 8.5: Elaborating what to observe

This description is from detail to abstraction, which requires eliciting of insights and capturing abstractions through interaction with stakeholders. Reversely, we need to validate our abstractions by testing them in the real world, also through stakeholder interaction. Once we have an initial set of roles, we can make them more specific again (moving upwards) to increase insight.

Figure 8.6 shows a few ways to make roles more specific in an insightful ways. A common technique, among others in software engineering, is defining personas. Another technique is defining archetypes. The Belbin roles are actually kind of archetypes. In discussions, it may also help to sketch caricatures.

8.3 Introduction to Stakeholders from abstract to humans

Figure 8.7 shows a supplier company in its ecosystem with many stakeholders of systems engineers. There are many stakeholders within the engineering department.

Defining **personas** is way to make roles less abstract

Defining a set of **archetypes** may help in reasoning about variations

Sketching **caricatures** brings the insights more alive

<i>plant</i> creative	<i>team worker</i> cooperative, averts friction	<i>implementer</i> disciplined, conservative, doer
<i>resource investigator</i> enthusiastic communicator	<i>shaper</i> driver, dynamic	<i>completer finisher</i> conscientious, painstaking
<i>coordinator</i> mature, chairman	<i>monitor evaluator</i> sober, analytical	<i>specialist</i> single-minded, rare skills

Belbin's archetypes



Caricature of a sergeant major
courtesy Bas van Zunderd

Figure 8.6: Making roles more specific using personas, archetypes, and caricatures

ments, in the broader organization including all lifecycle support functions, and the again broader ecosystem, and the customer ecosystem. Many roles in the organization have a responsibility for a subset of stakeholders. The black dots in the figure, denoted as proxies engage such subset. For example, the sector director for service will represent service stakeholders.

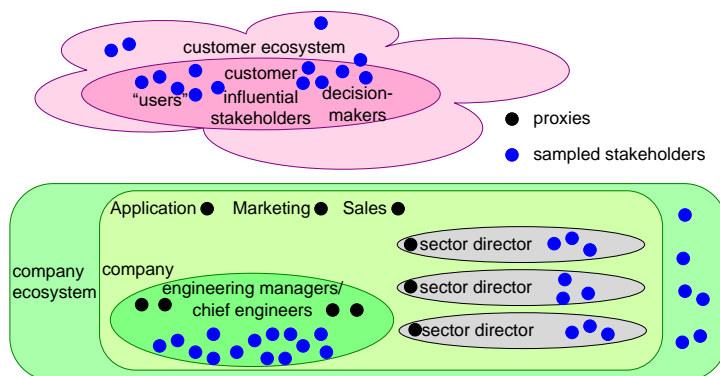


Figure 8.7: Multiple people engage stakeholders; creating context understanding is team effort

Systems engineers will engage these proxies frequently. However, their challenge is to sample the stakeholders these proxies represent as well. The systems engineers need to understand the various stakeholders sufficiently to engage effectively with the proxies. For example, knowing what to look for and what to ask.

Figure 8.8 elaborates for several roles what stakeholders they represent, and hence where they focus when engaging. The challenge for systems engineering is to

- Sales: customer stakeholders with decision power or big influence
 - Marketing: customer stakeholders and the wider customer ecosystem
 - Application: customer stakeholders that actively work with the system
 - Sector directors: (manufacturing, customer support, etc.) life cycle stakeholders and the wider life cycle ecosystem
 - Systems engineers: sampling enough relevant stakeholders to work with their problem and topic of interest
- How can systems engineers know what is enough sampling and what stakeholders are relevant?**

Figure 8.8: Who engages stakeholders?

know how much to sample and who to engage directly.

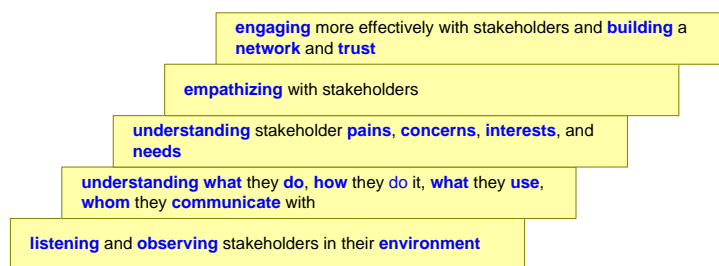


Figure 8.9: Increasing competence levels to achieve a trustful network

We see that systems engineers know that they need to engage stakeholders. However, probably handicapped by limited social skills and introversion, they tend to contact stakeholders way too little. Figure 8.9 shows kind of maturity model for engaging stakeholders.

- The most basic level is listening and observing stakeholders in their environment.
- This allows them to start understanding what they do, how they do it, what they use, whom they communicate with.
- The next step is understanding stakeholder pains, concerns, interests, and needs.
- Which then allows them to empathize with stakeholders in a genuine way.
- Ultimately, systems engineers need the competency to engage more effectively with stakeholders and building a network and trust

8.4 An approach to engage stakeholders

Figure 8.10 shows an approach to engage stakeholders.

- Analyze the field of stakeholders, discuss them and make a map capturing the stakeholders, their positions, and their relations.
- Select a subset of stakeholders to contact them.
- Prepare by reading, searching, discussing, sketching and making diagrams.
- However, engage early and do not postpone that. That requires time-boxing of the initial steps, especially the preparation.
- Capture and communicate insights somewhat more structured, e.g. using A3 architecture overviews [1].
- Keep repeating these steps!

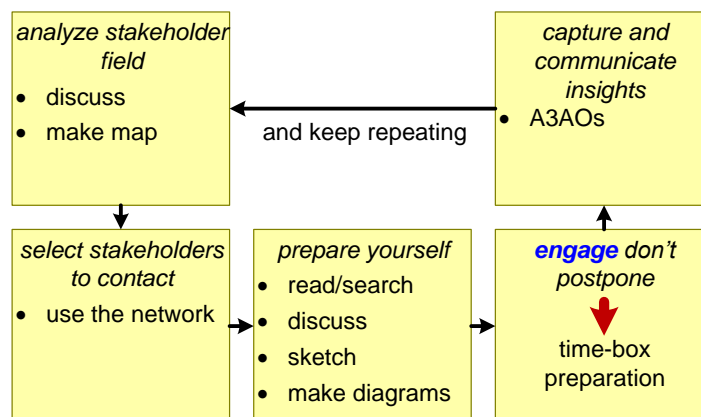


Figure 8.10: An approach to engage stakeholders

8.5 Acknowledgements

Bas van Zunderd provided the caricature of a sergeant major. Natansh Vyas inspired this presentation. Kristin Falk provided feedback on not working diagrams and text. Omid Razbani caught linguistic mistakes.

Bibliography

- [1] Daniel Borches. *A3 Architecture Ovevriews: A Tool for Effective Communication in Product Evolution*. Ph.D. thesis. Woormann Print Service, Enschede, Netherlands, 2010.
- [2] Stephen R. Covey. *The 7 habits of highly effective people: restoring the character ethic*. Free Press, New York, rev. ed. edition, 2004.
- [3] Gerrit Muller. The system architecture homepage. <http://www.gaudisite.nl/index.html>, 1999.
- [4] Gerrit Muller. CAFCR: A multi-view method for embedded systems architecting: Balancing genericity and specificity. <http://www.gaudisite.nl/ThesisBook.pdf>, 2004.
- [5] Gerrit Muller. Dynamic range of abstraction levels in architecting. <https://gaudisite.nl/DynamicRangeAbstractionLevelsPaper.pdf>, 2011.
- [6] Gerrit Muller. <https://gaudisite.nl/basicworkingmethodarchitectpaper.pdf>. <https://gaudisite.nl/BasicWorkingMethodArchitectPaper.pdf>, 2011.
- [7] 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.
- [8] 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: 0.2, date: June 6, 2026 changed by: Gerrit Muller

- added WhatIsInnovation
- added TESDCexampleCase, TESDCexerciseQuadruplet, TESDCworkflowExample, discussion slides, TESDCconclusions, TESDCarchitectureOverviews, and SystemsAndLeadershipProgram to slides
- added Logo-slides for navigation status to draft

Version: 0.1, date: May 31, 2026 changed by: Gerrit Muller

- added TESDCframeworks
- added slides only TESDCstartupExamples, TESDCframeworks, TESDCoverviewExamples
- added logo, which is also the tutorial flow in the introduction
- moved text from abstract to introduction

Version: 0, date: May 9, 2026 changed by: Gerrit Muller

- Created, using material from MarketProductLifecycle, FromSystemToEcosystem, StakeholdersFromAbstractToHumans, SystemModelingAndAnalysis.