Course System Architecting (SARCH/SESA)

by *Gerrit Muller*

University of South-Eastern Norway-NISE

Abstract

The Course System Architecture SARCH is a course for system architects, potential system architects and immediate stakeholders of the system architect, such as project leaders, designers and marketing managers.

February 15, 2024
status: draft
version: 0.5

SARCH/SESA Module 00 Course Information

by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

Course System Architecting Introduction

February 15, 2024
status: draft
version: 1.3

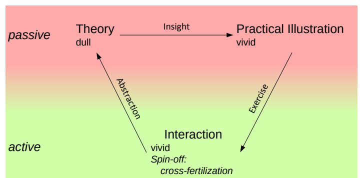
SESA Course Systems Architecting

by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

This article describes the course Systems Architecting. The course is set up to make the art of system architecting more accessible. The course will address a wide spectrum of issues in relation with system architecture, such as: Processes, Business, Role and task of the system architect (team), Roadmapping, System Architect toolkit, Technical, Skills, and Psycho Social

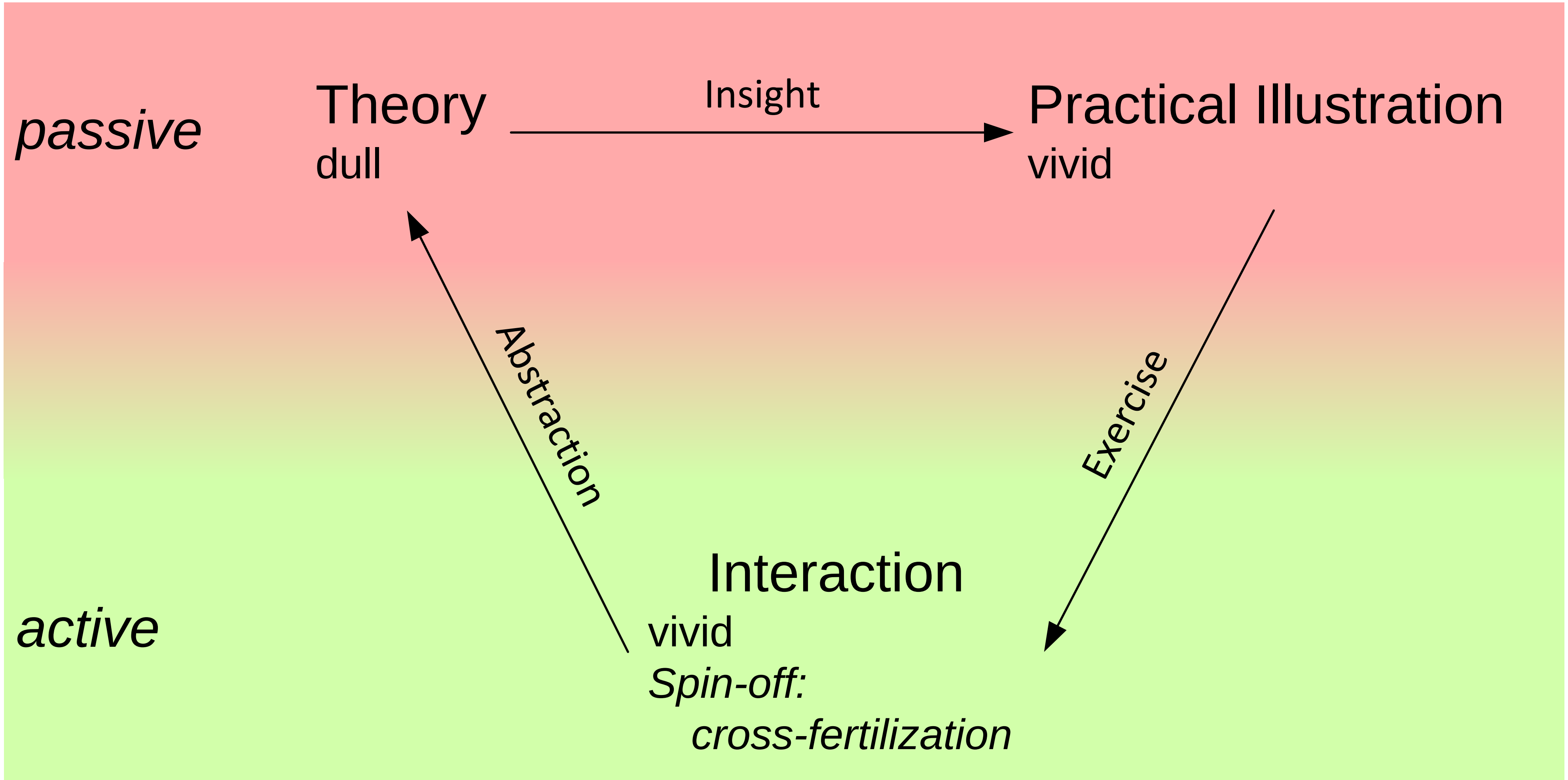
February 15, 2024
status: concept
version: 2.4



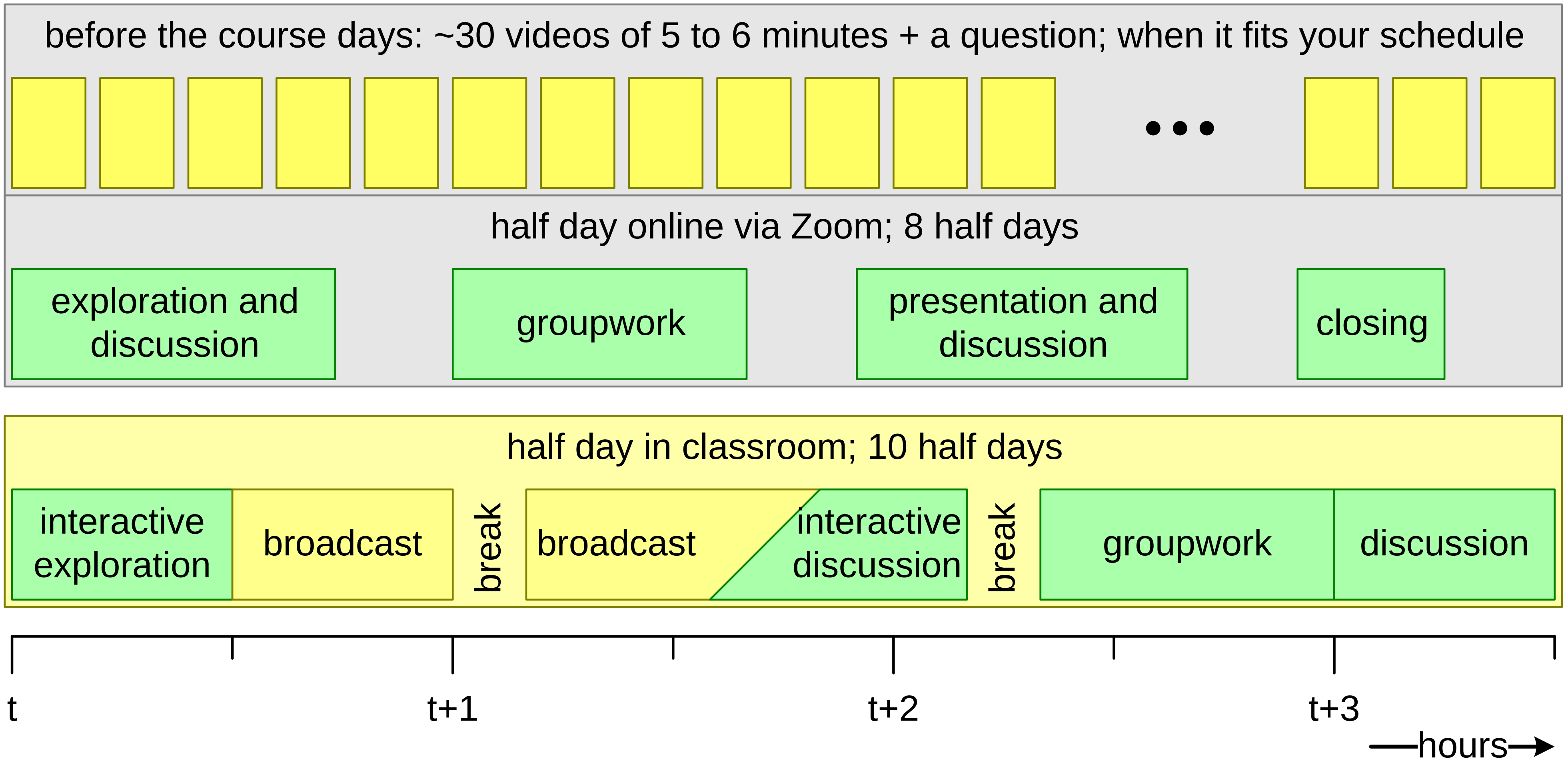
Program

Session 1 Positioning the System Architecture Process, Product Creation Process	
Session 2 Role and Task of the System Architect	
Session 3 Requirements Capturing	
Session 4 System Architect Toolkit	
Session 5 Roadmapping	online combined in half day
Session 6 Product Families, generic developments	
Session 7 Documentation, reviewing and other supportive processes; The role of Software in complex products	
Session 8 BoM presentation	
Session 9 Psycho Social side	online combined in half day
Session 10 Wrap up, Expectations, How to continue, Evaluation	

Structure


passive
Theory
dull
Insight
Practical Illustration
vivid
Abstraction
Exercise
active
Interaction
vivid
Spin-off:
cross-fertilization

Timing Template of one subject



Rules of the Interactive Parts

- Your contribution is essential.
- Don't monopolize the time, everyone also the quiet people should have the opportunity to contribute;
  The facilitator will intervene if the contribution is limited to a small group of participants.
- Respect the contribution of others;
  Opinions can't be wrong, difference of opinion is normal and called pluriformity.
- The course format is highly experimental and based on improvisation, constructive proposals are welcome;
  it is your course! Regular evaluations will give the opportunity to influence the rest of the course.

Rules of the Broadcast Parts

- Please write your questions/remarks/statements on yellow stickers and attach them at the end on the P-flip.
  These will be used in the interactive section for discussion and to increase insight.
- Short clarification questions are welcome,
  discussion will take place in the interactive part.
- Stupid questions don't exist. Learning is based on **safe** and **open** interaction.
  Very individual oriented questions can be referred to a break or after the session.

The Gaudí Project

by *Gerrit Muller* University of South-Eastern Norway-NISE

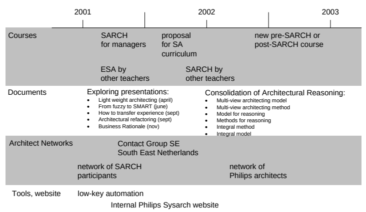
e-mail: `gaudisite@gmail.com`

`www.gaudisite.nl`

Abstract

The Gaudí project is described. The goals of the project, the way of working, and an outline for the period 2001 to 2003. The deliverables in terms of documents are positioned by means of a two-dimensional map. Courses based on the Gaudí material are described. The current status of the courses is given.

February 15, 2024
status: draft
version: 3.1



Goals of the Gaudí Project

- Consolidate existing Systems Architecting Methods

  evaluate, reflect, generalize

- Make the Systems Architecting art more accessible

  case descriptions

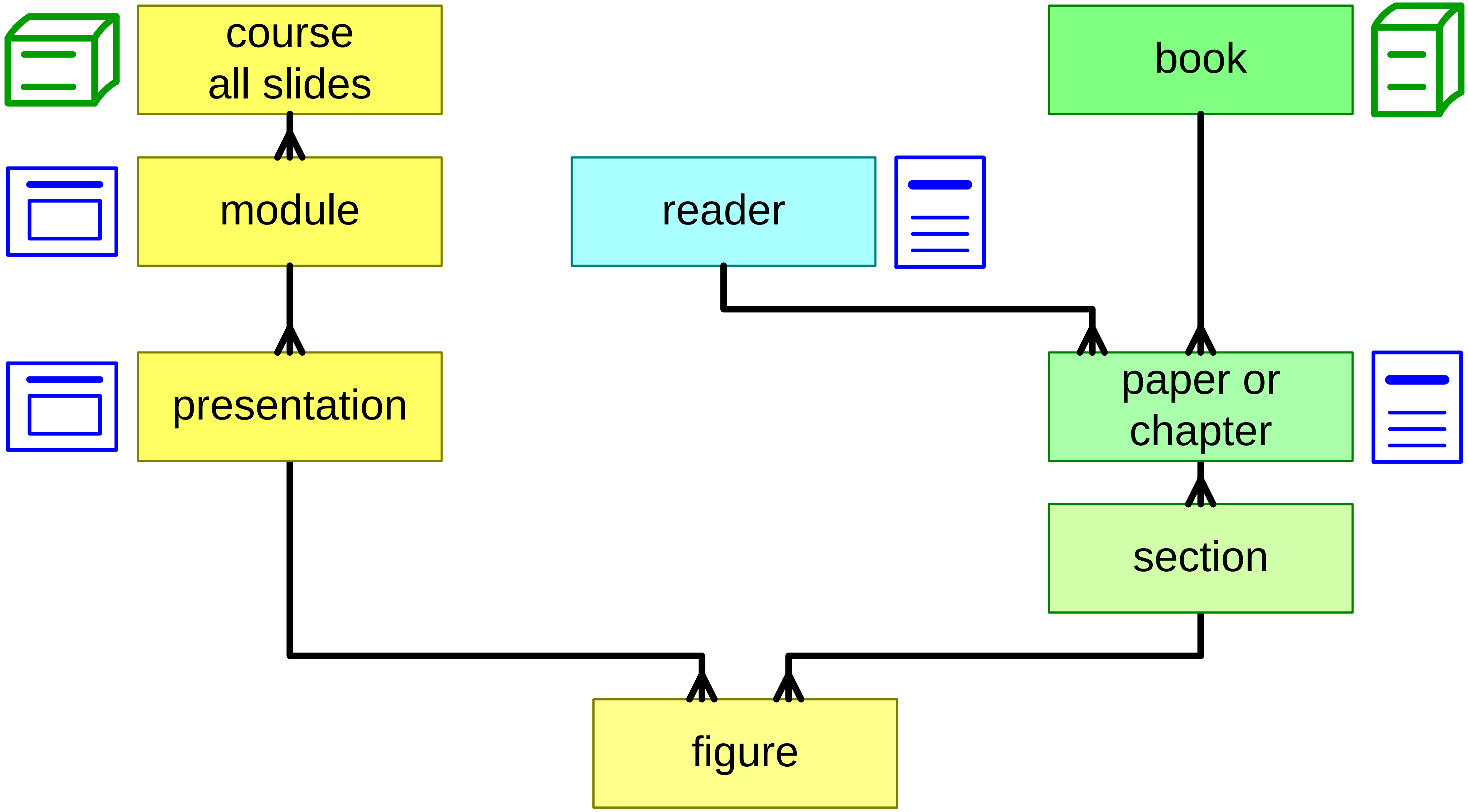
- Enable the education of (future) System Architects

  curriculum, course material

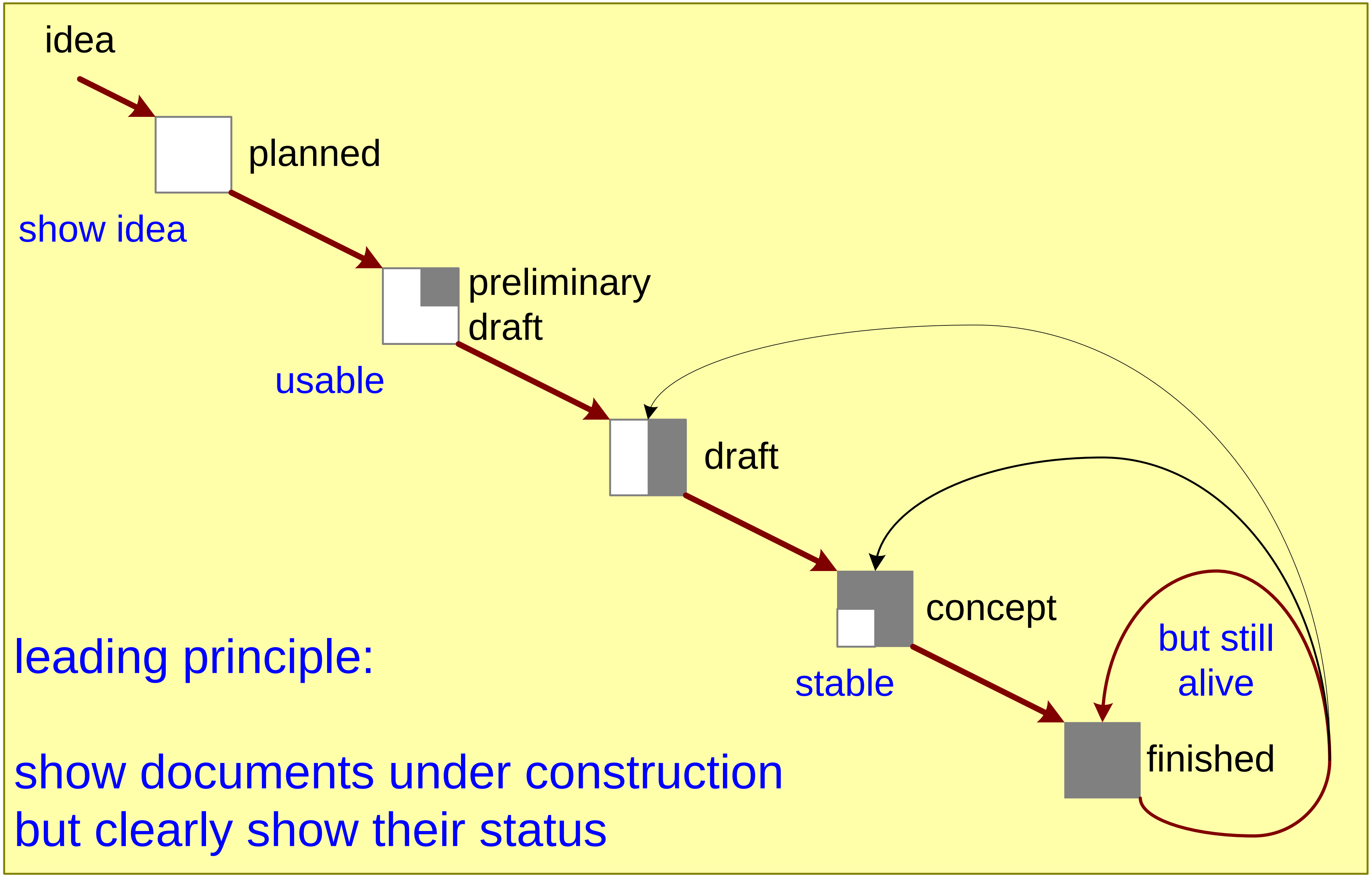
- Research new or improved Systems Architecting Methods

  industry as laboratory

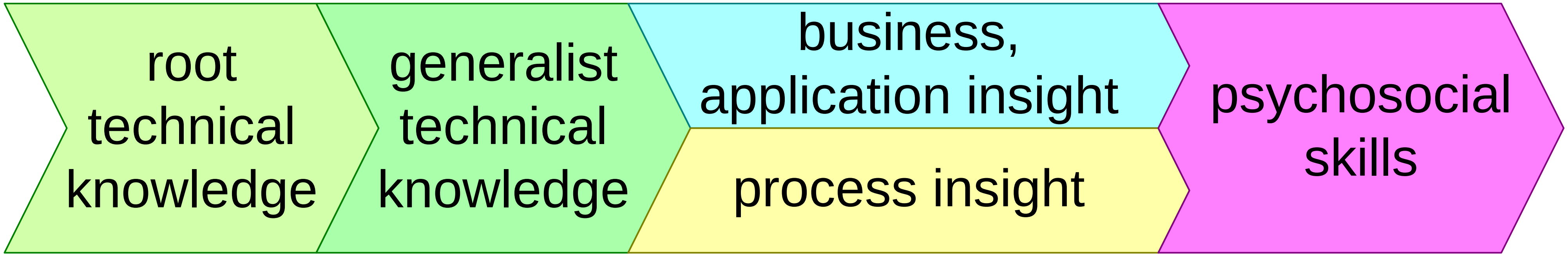
Modular approach



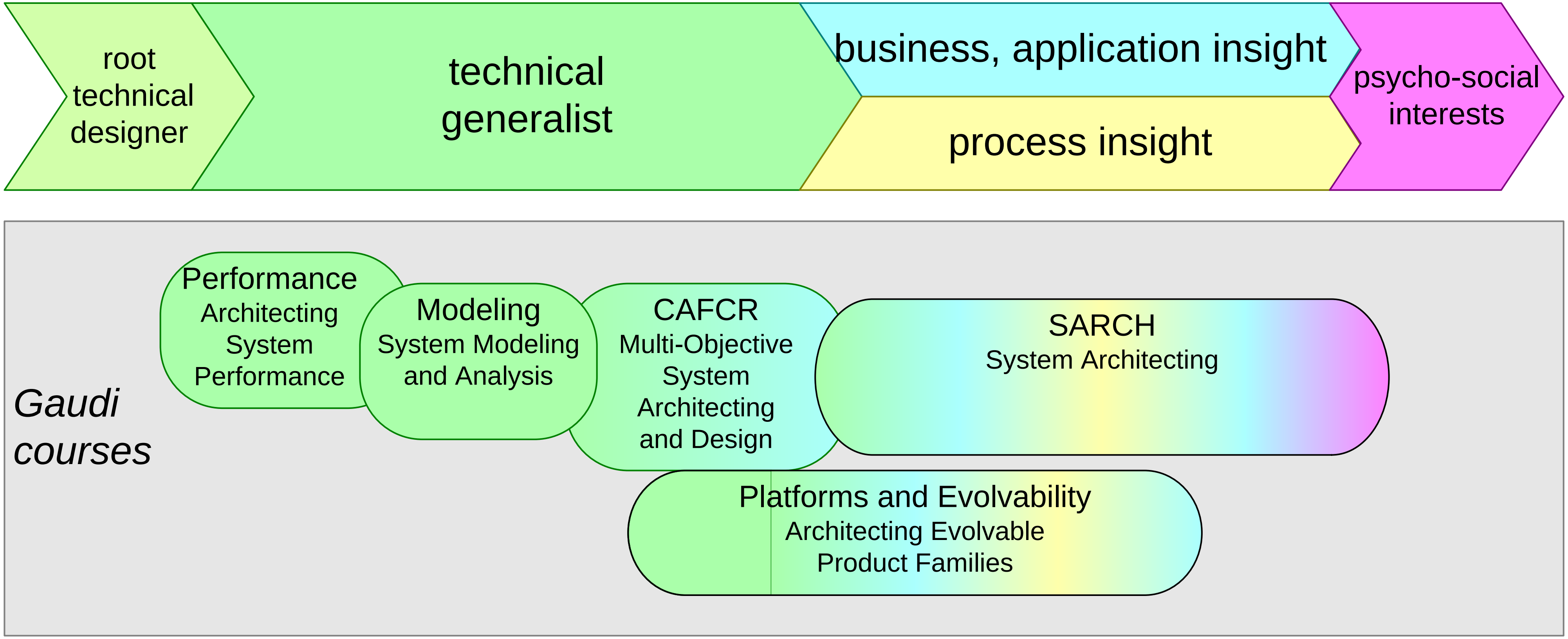
Show Early to Get Feedback



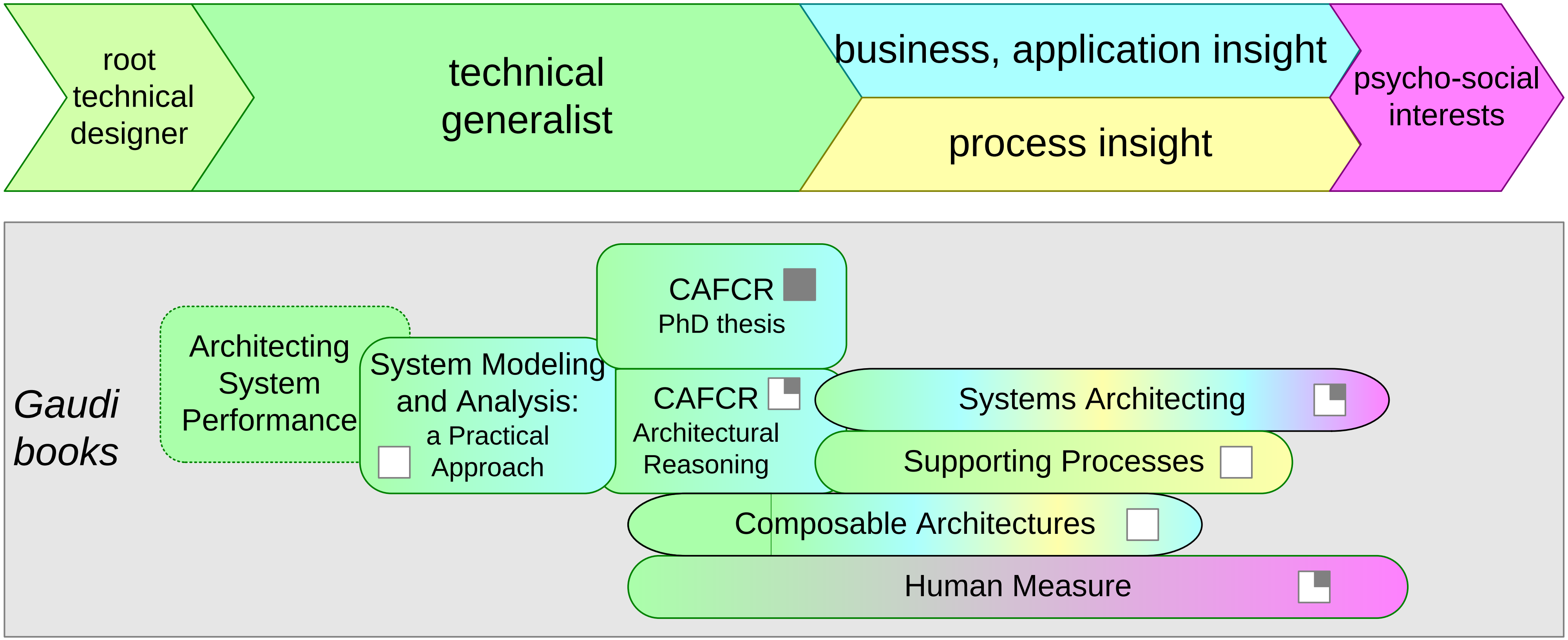
Growth of the System Architect



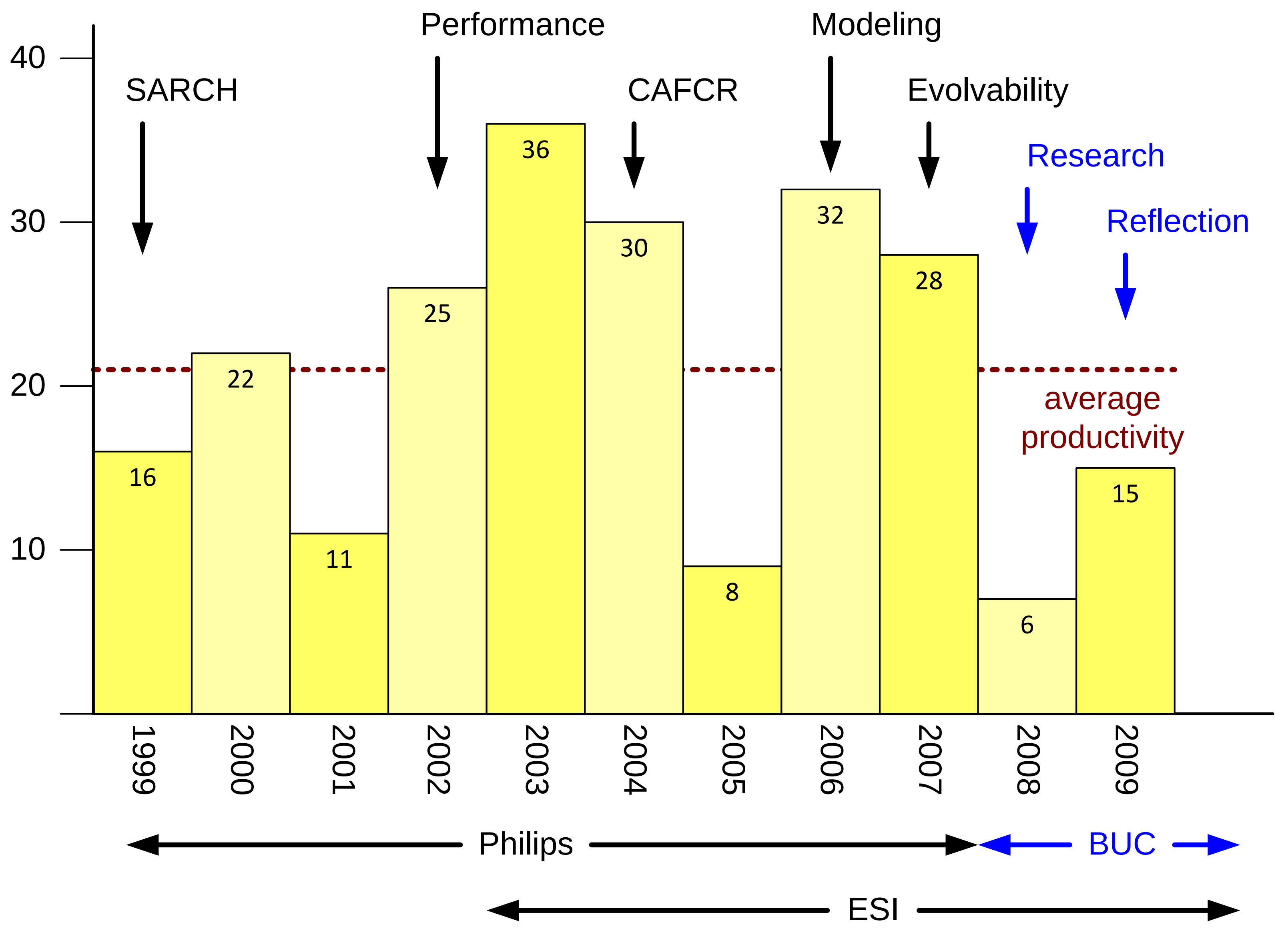
Positioning Courses



Positioning Books



Productivity: number of new entries



Module System Architecture Context

by *Gerrit Muller* Buskerud University College and Buskerud University College
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

The system architecture process is positioned in a wider context: First in the business context, then in the Product Creation Process context.

February 15, 2024
status: concept
version: 1.1

Process Decomposition of a Business

by *Gerrit Muller* USN-SE

e-mail: gaudisite@gmail.com

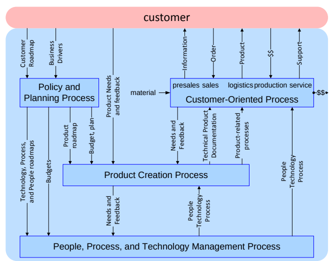
www.gaudisite.nl

Abstract

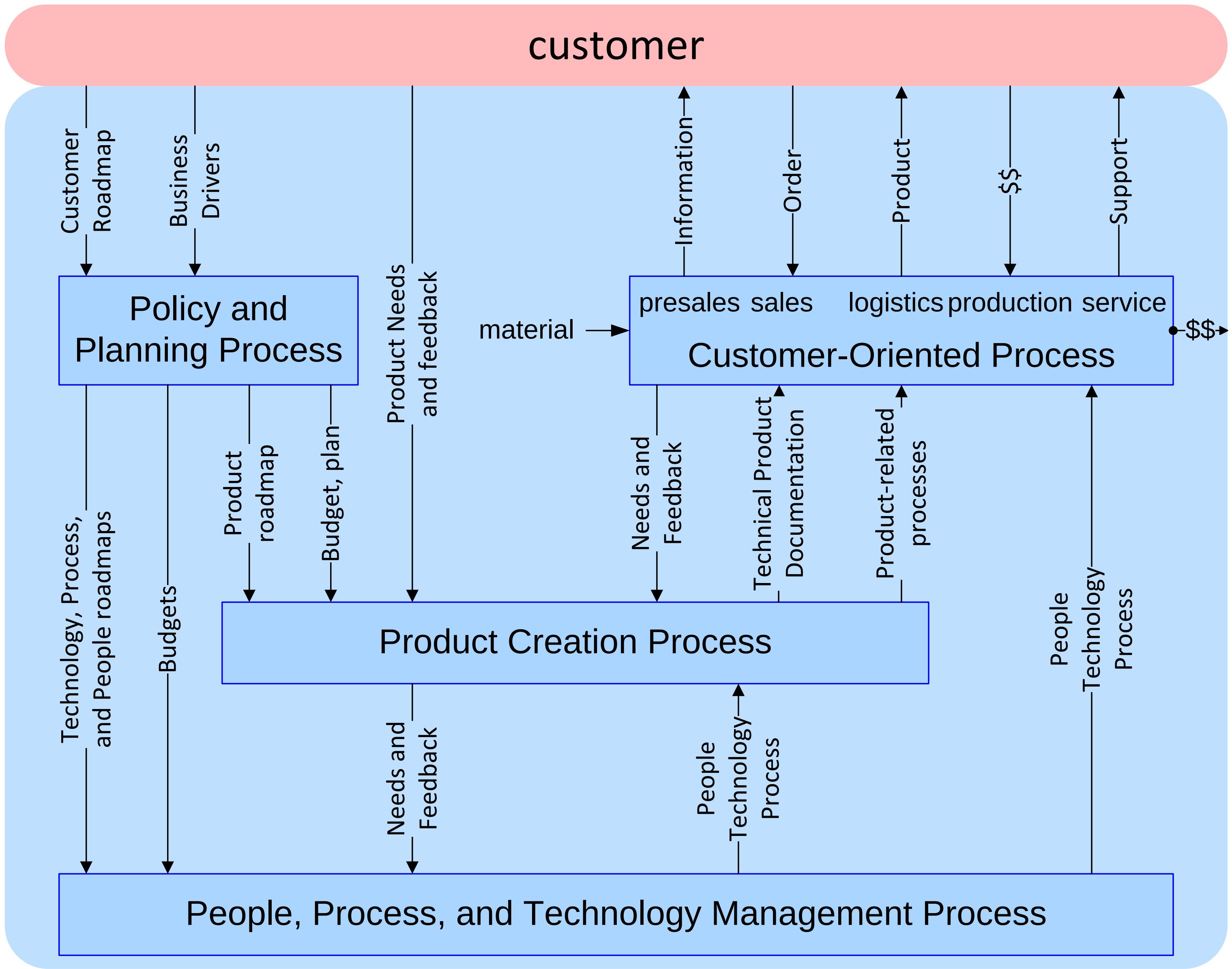
This article positions the system architecture process in a wider business scope. This positioning is intended to help understanding the processes in which the system architect (or team of system architects) is involved.

It focuses on an organization that creates and builds systems consisting of hardware and software. Although other product areas such as solution providers, services, courseware, et cetera also need system architects, the process structure will deviate from the structure as presented here.

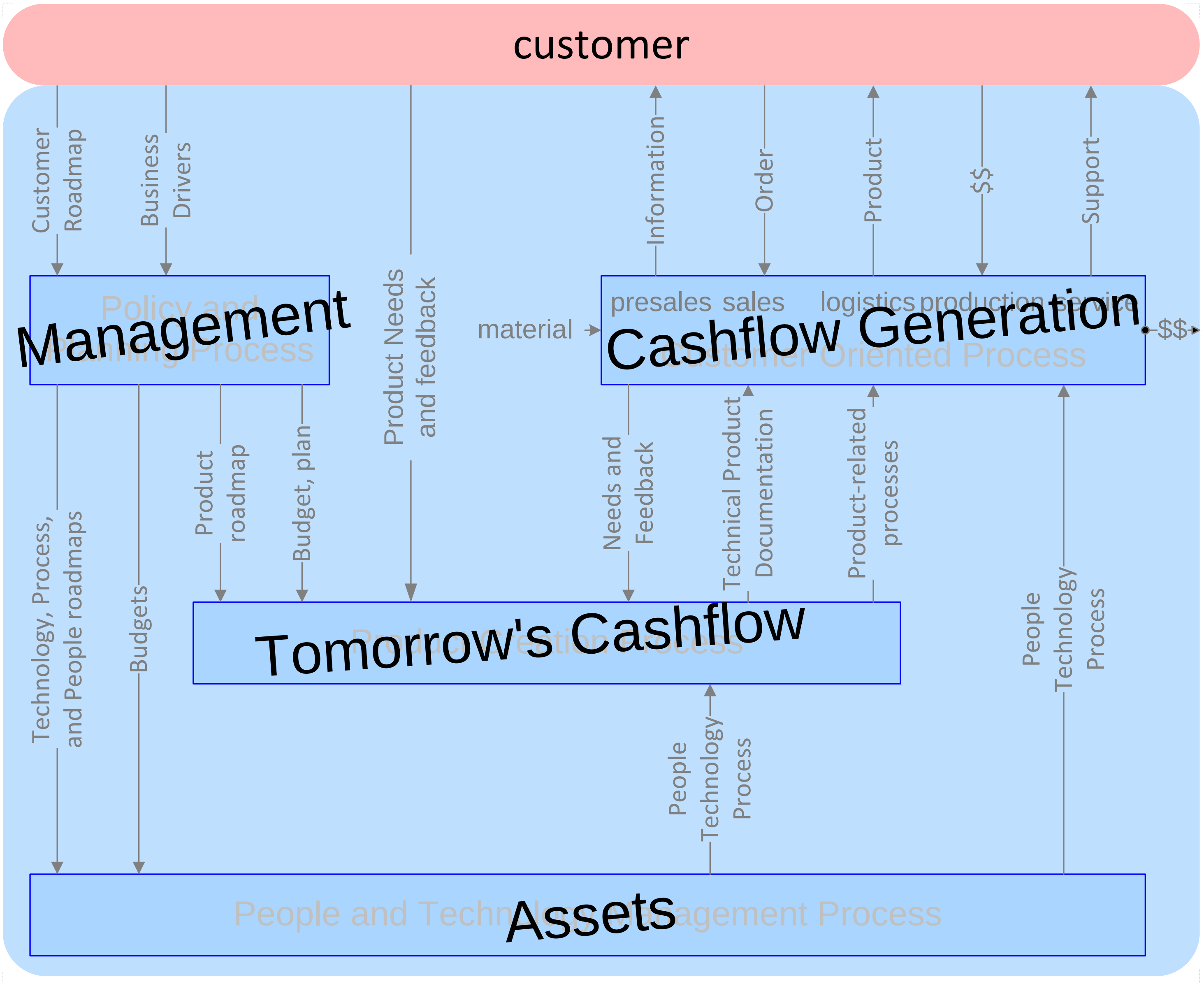
February 15, 2024
status: concept
version: 1.1



Simplified Decomposition of the Business



Financial Characterization of Decomposition



Multiple Instances per Process

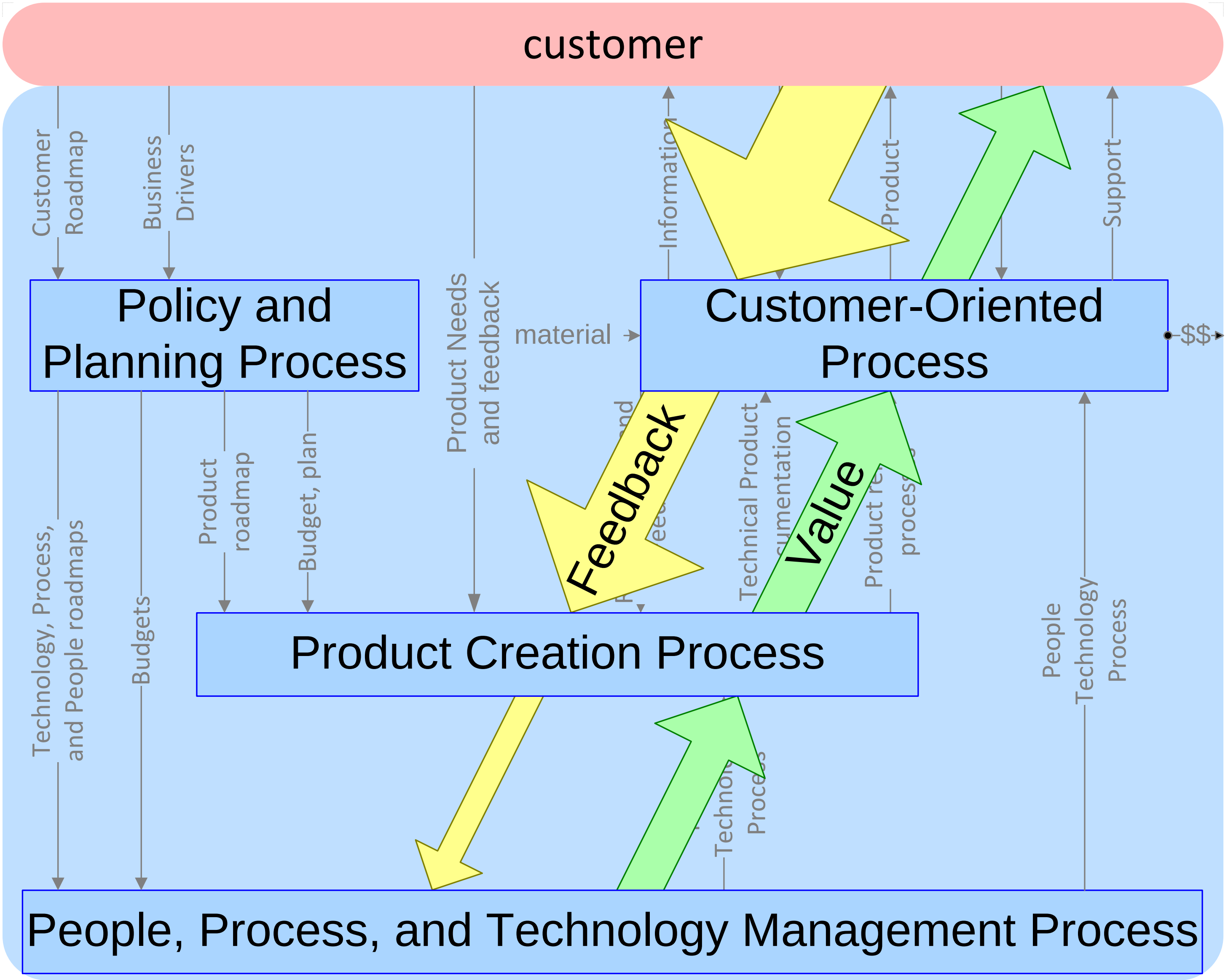
Customer Oriented Process: Depends on geography, customer base, and supply chain.

Product Creation Process: One per entity to be developed, where such an entity can be a product family, a product, or a subsystem.

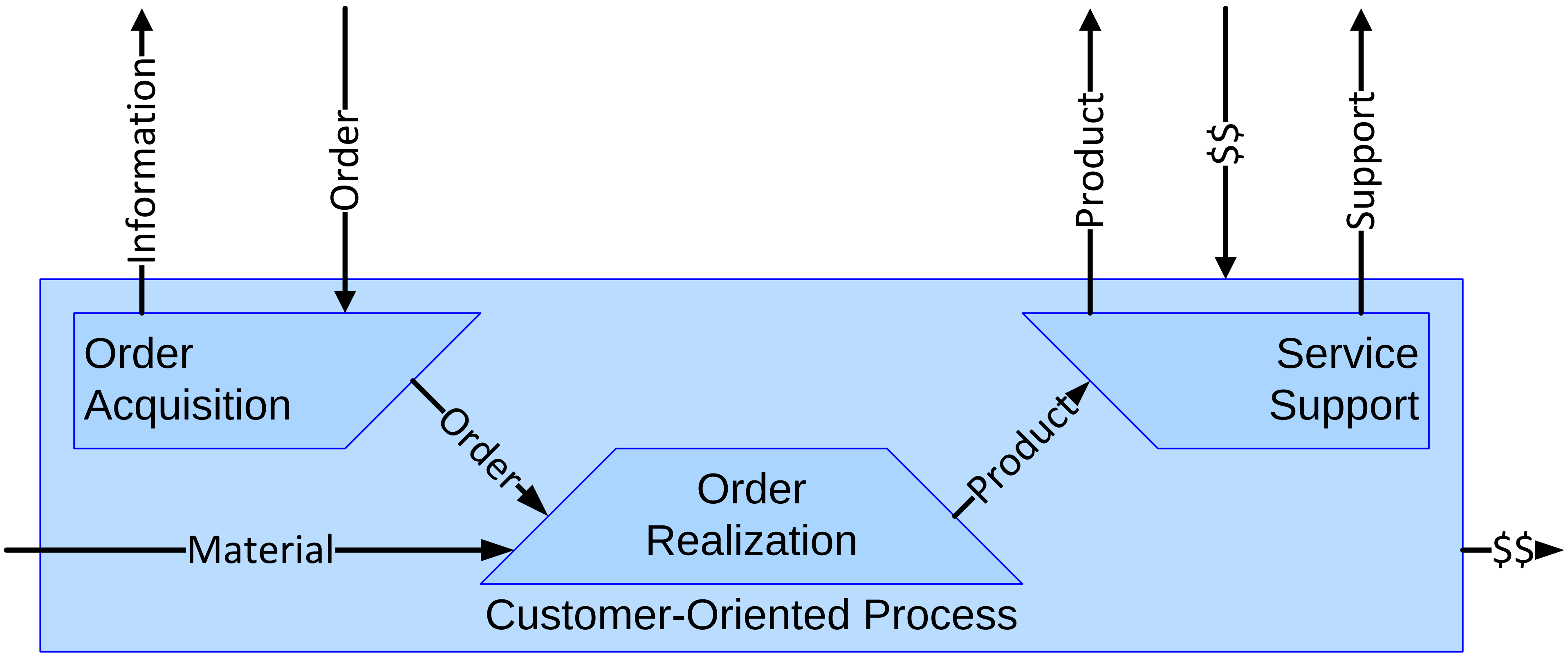
People and Technology Management Process: One per “competence”, where a competence is a cohesive set of technologies and methods.

Policy and Planning Process: One per business. This is the pro-active integrating process.

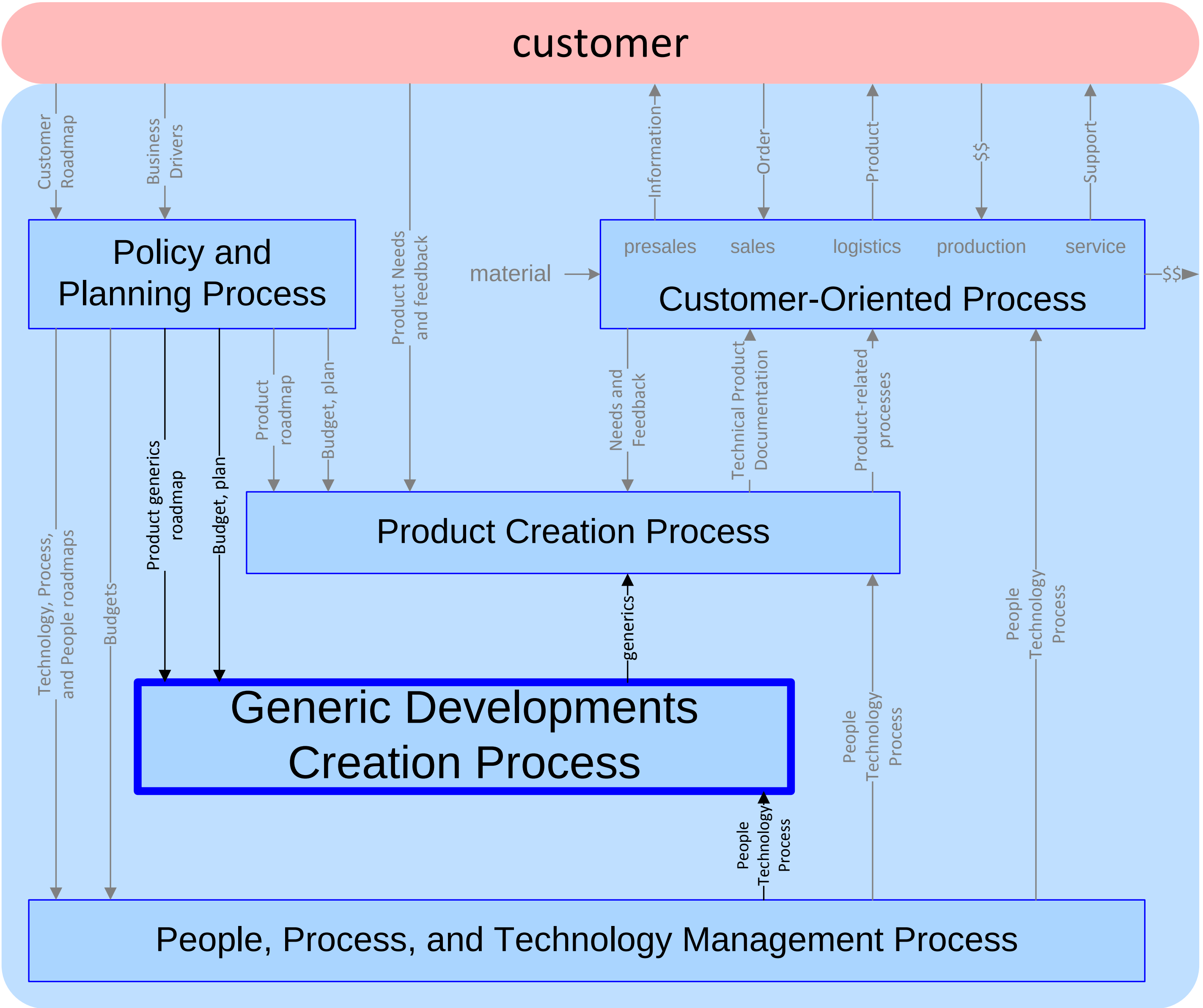
The Value Chain and the Opposite Feedback Flow



Decomposition of the Customer Oriented Process



Extended with Generic Developments



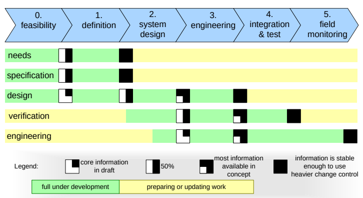
The Product Creation Process

by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

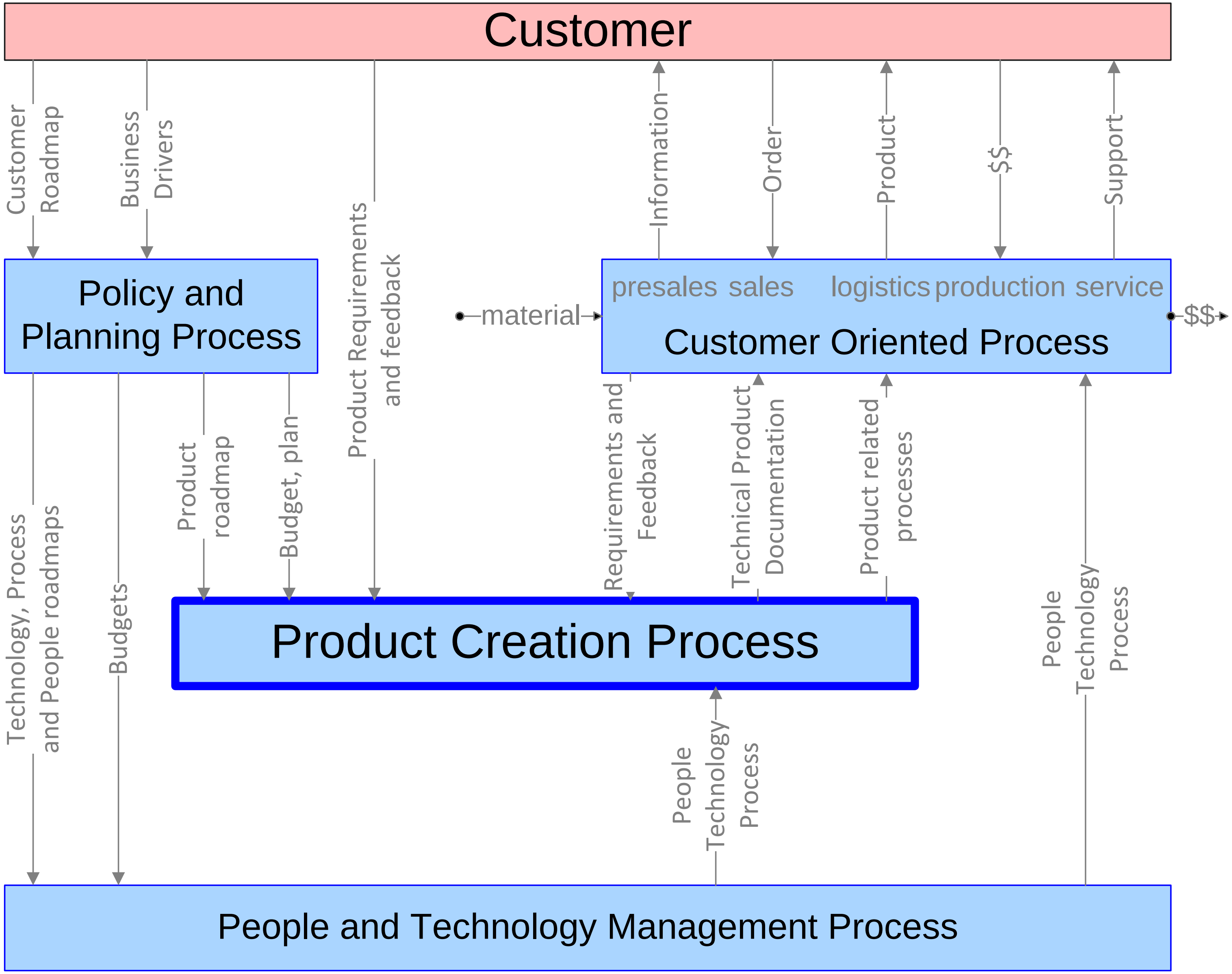
Abstract

The Product Creation Process is described in its context. A phased model for Product Creation is shown. Many organizations use a phased model as blueprint for the way of working. The operational organization of the product creation process is discussed, especially the role of the operational leader.

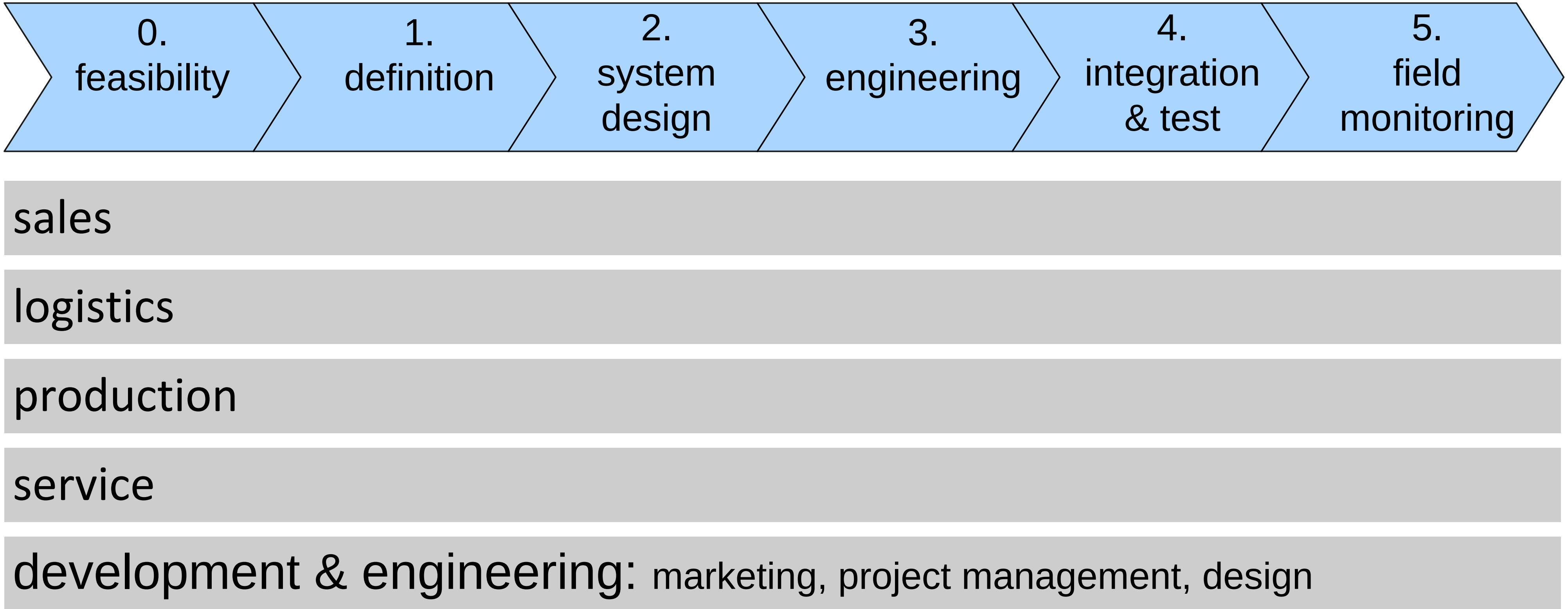
February 15, 2024
status: concept
version: 2.2



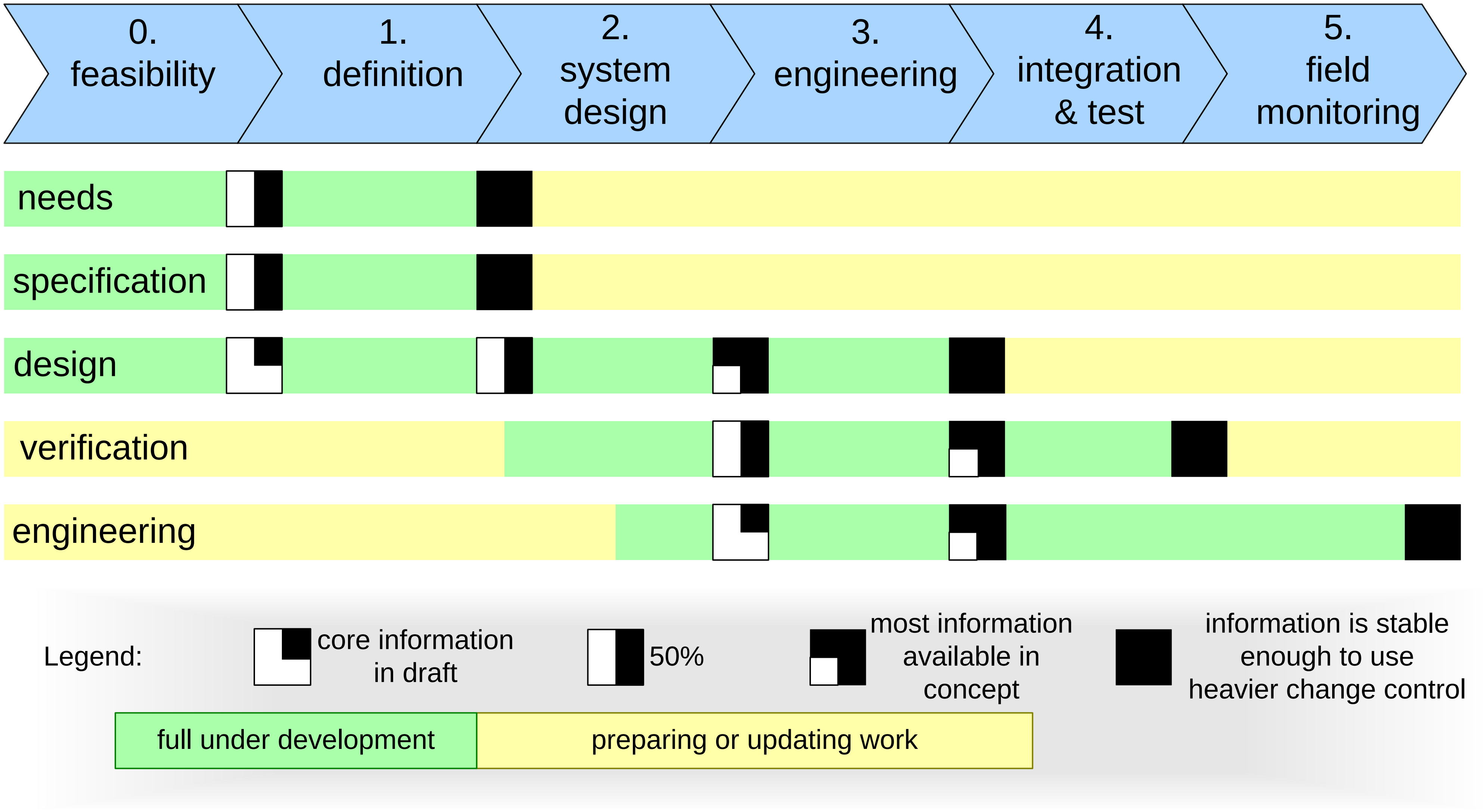
The Product Creation Process in Business Context



Phasing of the PCP at Business Level



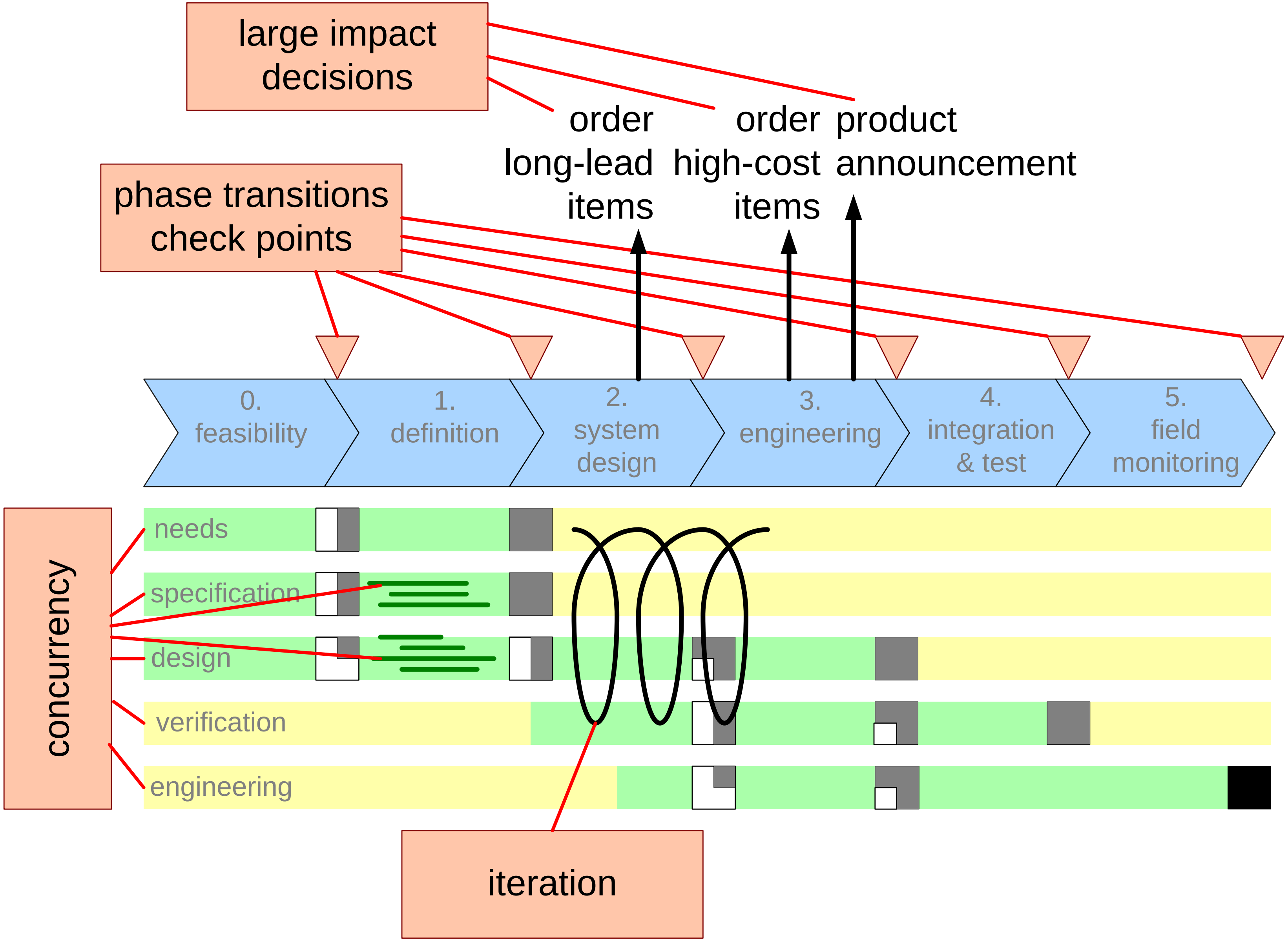
Phasing the Design Control Process



Advantages and Disadvantages of a Phased Process

benefits	*disadvantages*
blueprint: how to work	following blueprint blindly
reuse of experience	too bureaucratic
employees know *what* and *when*	transitions treated black and white
reference for management	

Characteristics of a Phase Model



Decisions and Phase Transitions

Define a minimal set of *large-impact* decisions.

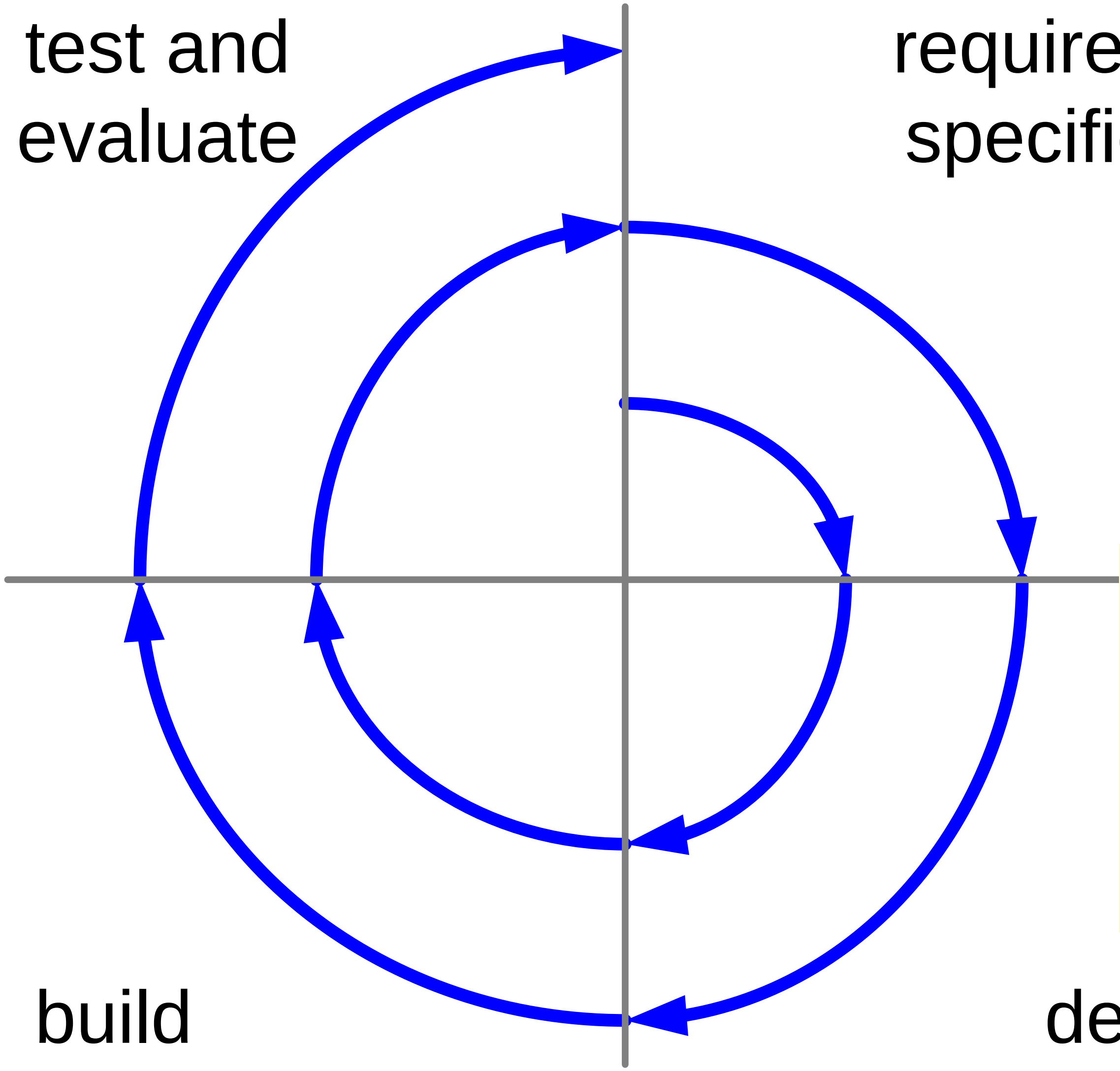
Define the mandatory and supporting information required for the decision.

Schedule a decision after the appropriate phase transition.

Decide explicitly.

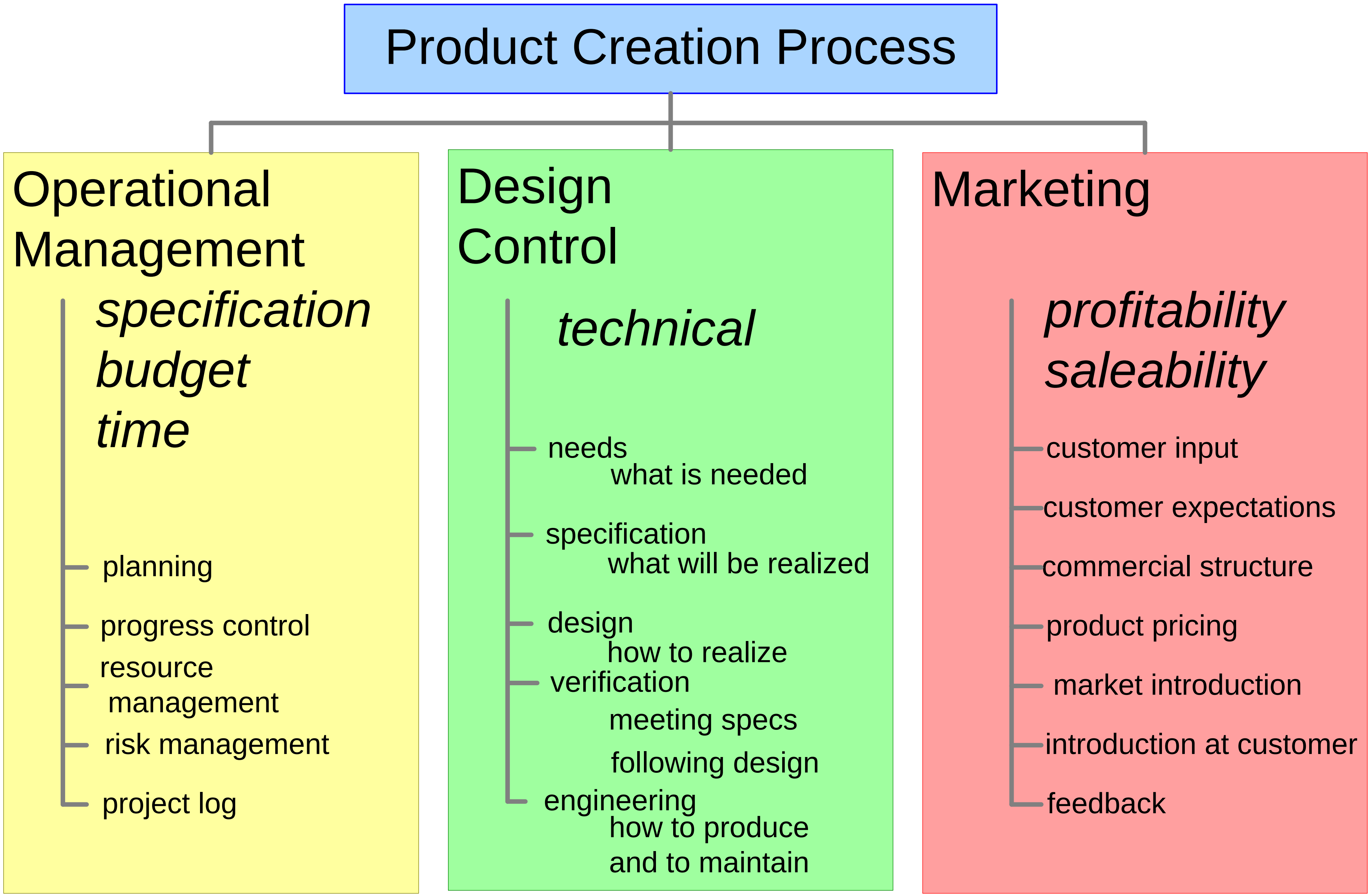
Communicate the decision clearly and widely.

Evolutionary PCP model

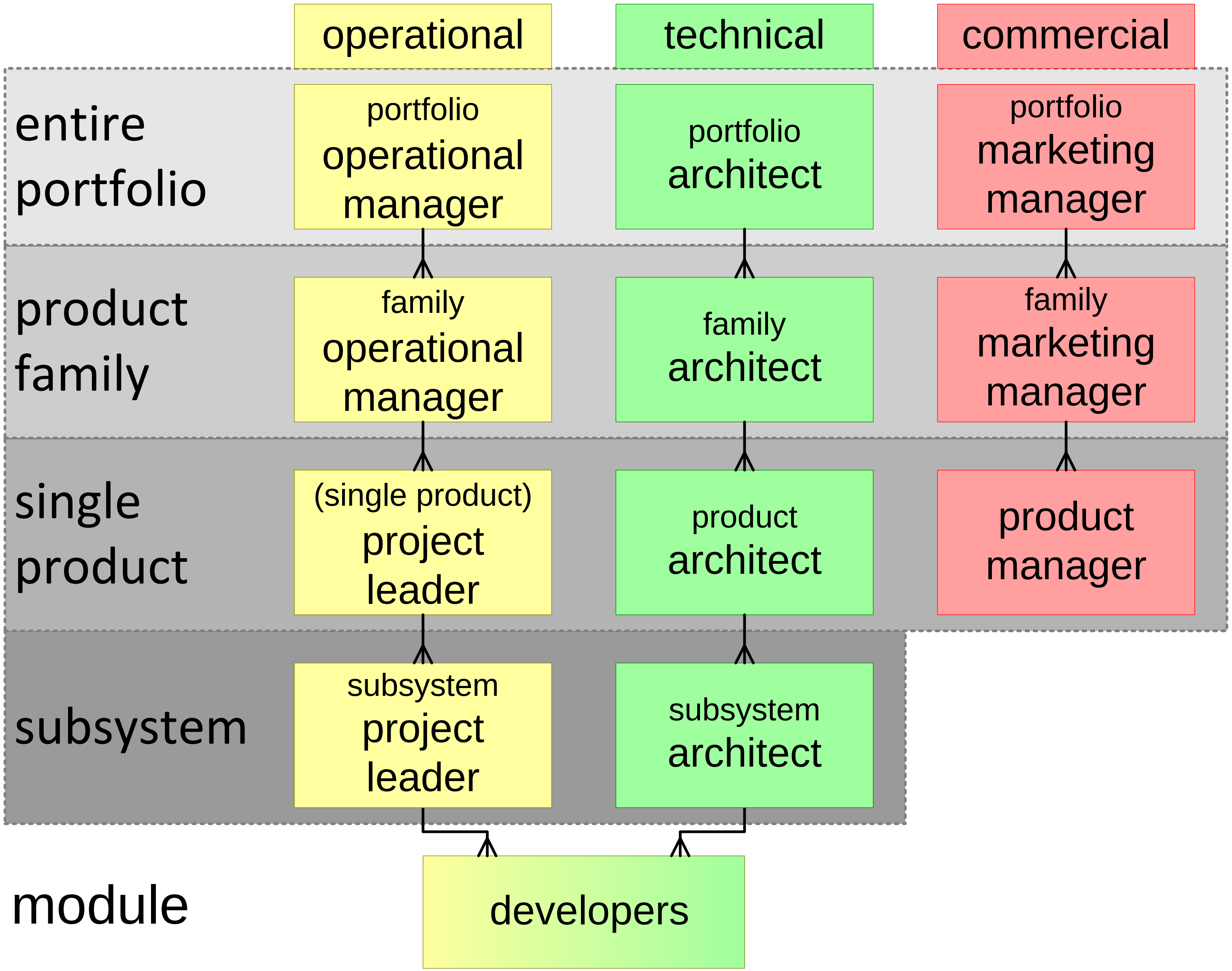


2% of budget (EVO)
2 weeks (XP)
up to 2 months
per cyclus

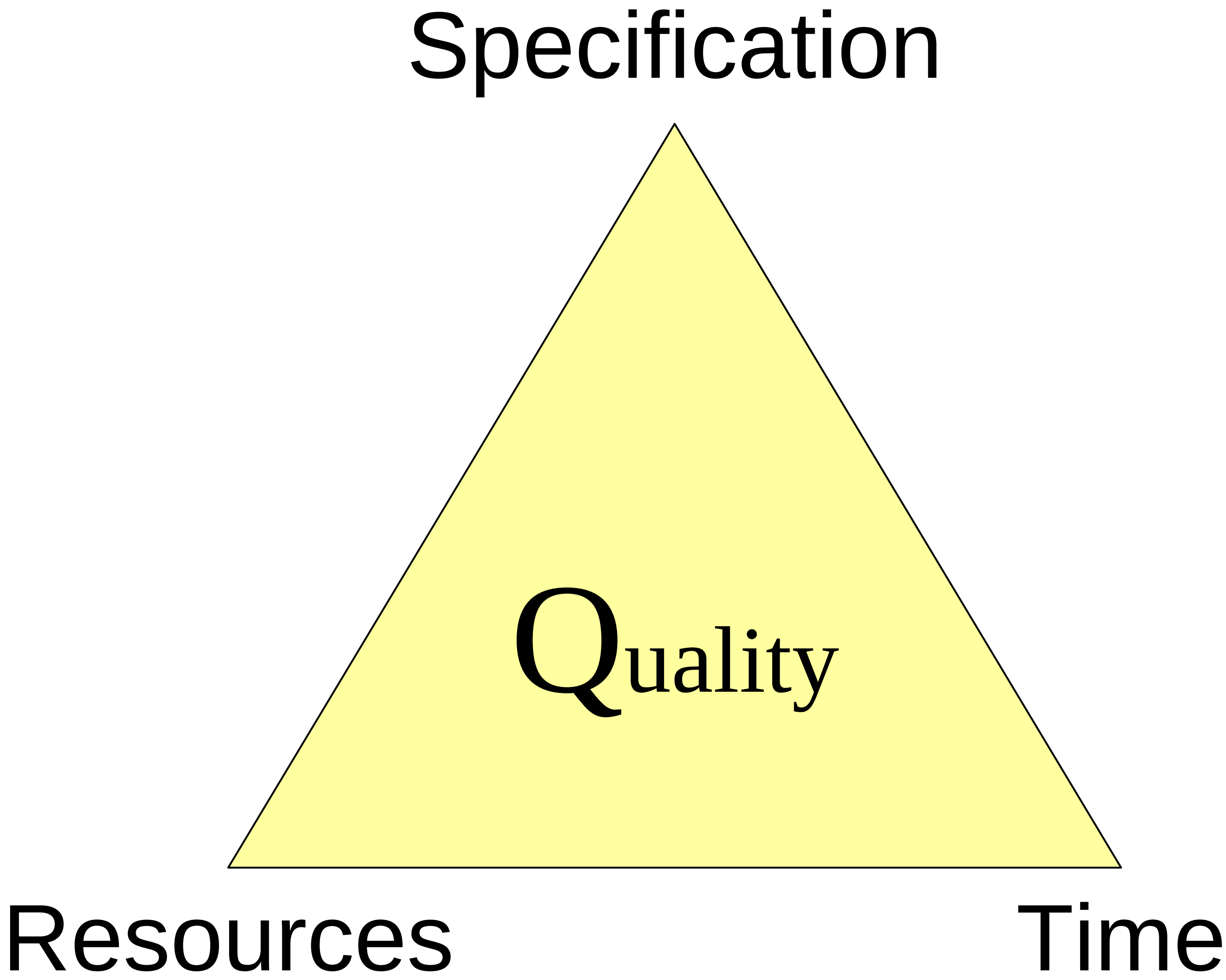
Decomposition of the Product Creation Process



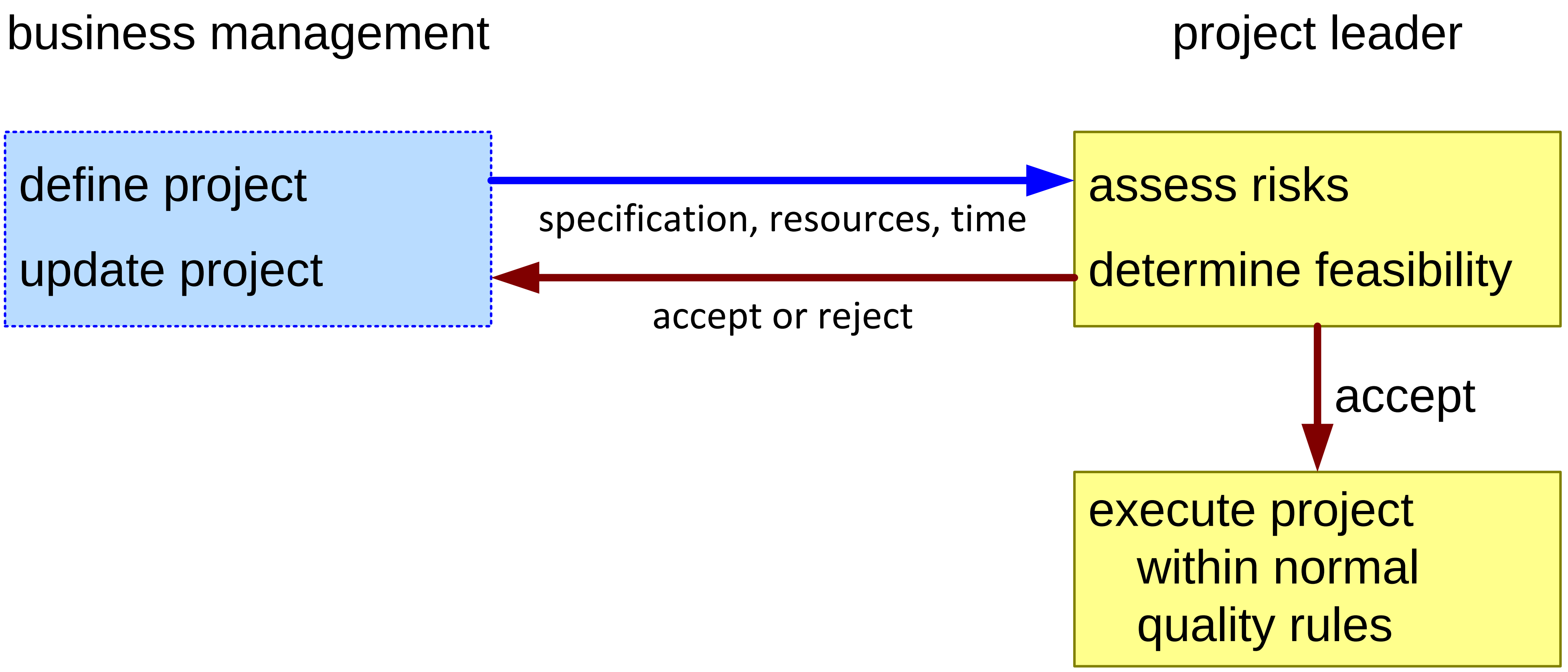
Operational Organization of the PCP



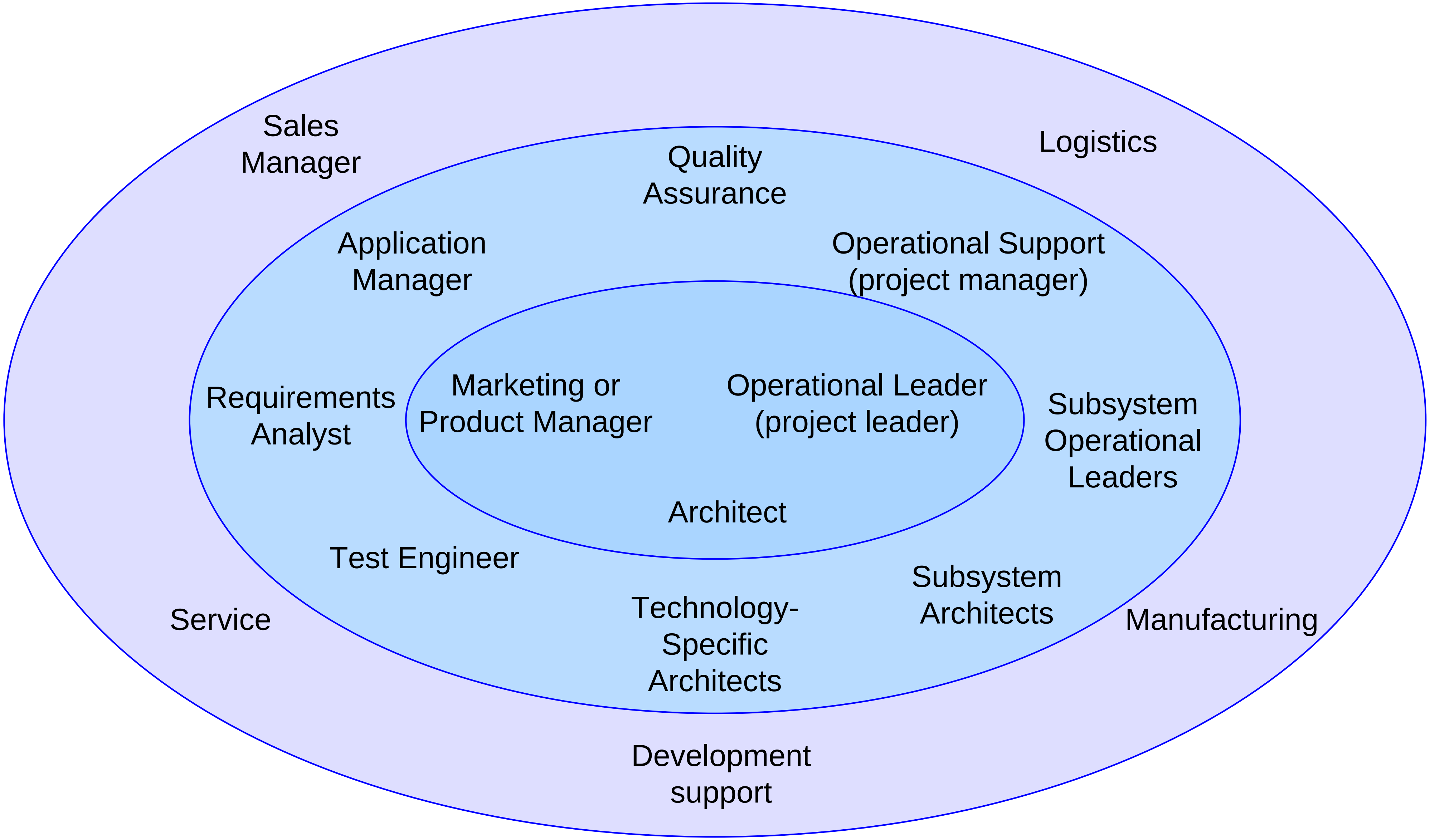
Prime Responsibilities of the Operational Leader



The Rules of the Operational Game



Operational Teams



The System Architecture Process

by *Gerrit Muller* USN-SE

e-mail: gaudisite@gmail.com

www.gaudisite.nl

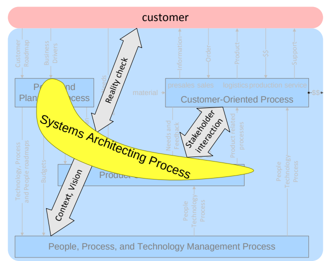
Abstract

The System Architecture Process is positioned in the business context. This process bridges the gap between the Policy and Planning Process and the Product Creation Process.

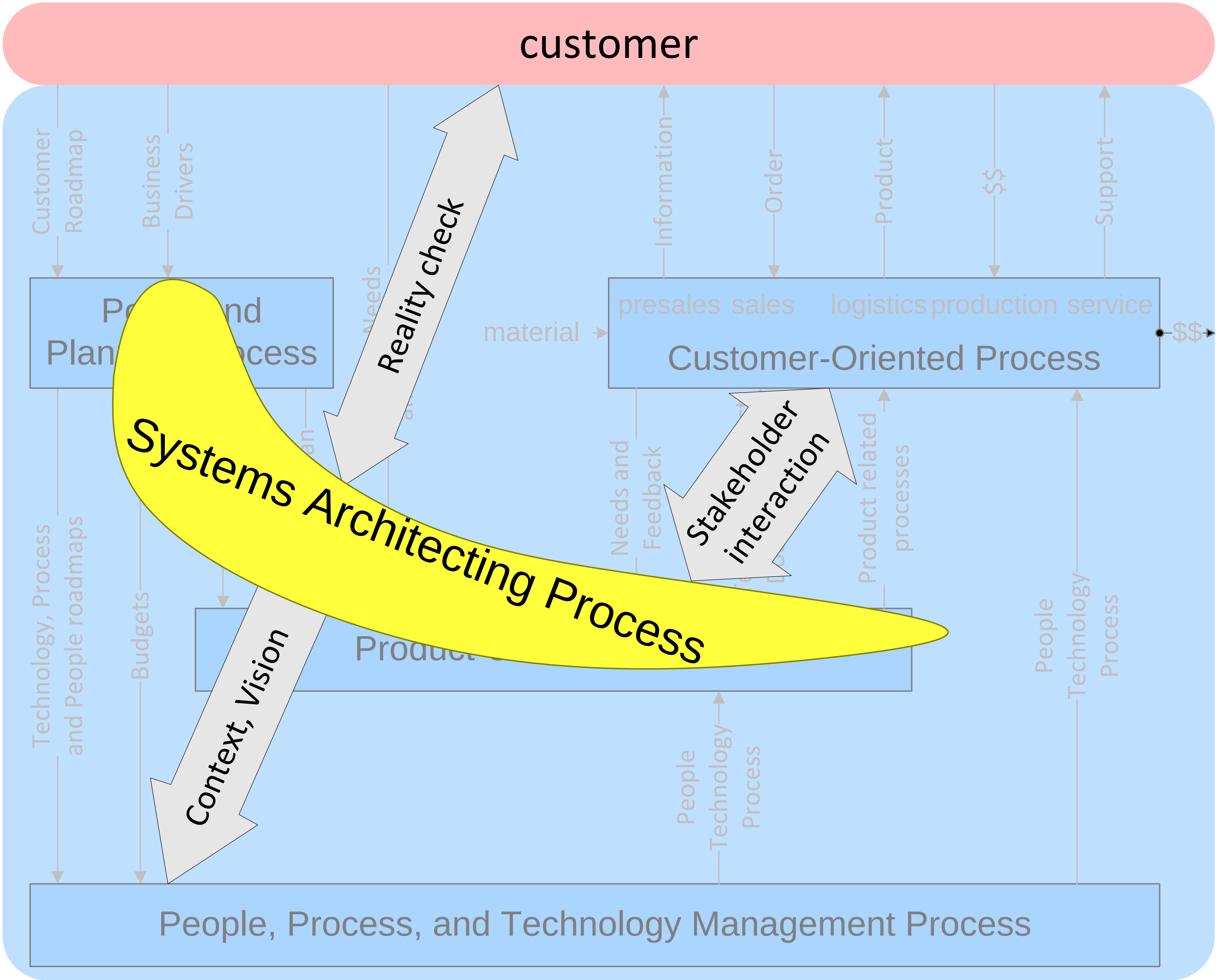
The purpose of the System Architecture Process is to provide the Integral Technical overview and consistency, and to maintain the integrity over time. Subjective characteristics as elegance and simplicity are key elements of a good architecture.

The scope of the system architecture process is illustrated by showing 5 views used in a reference architecture, ranging from Customer Business to Realization.

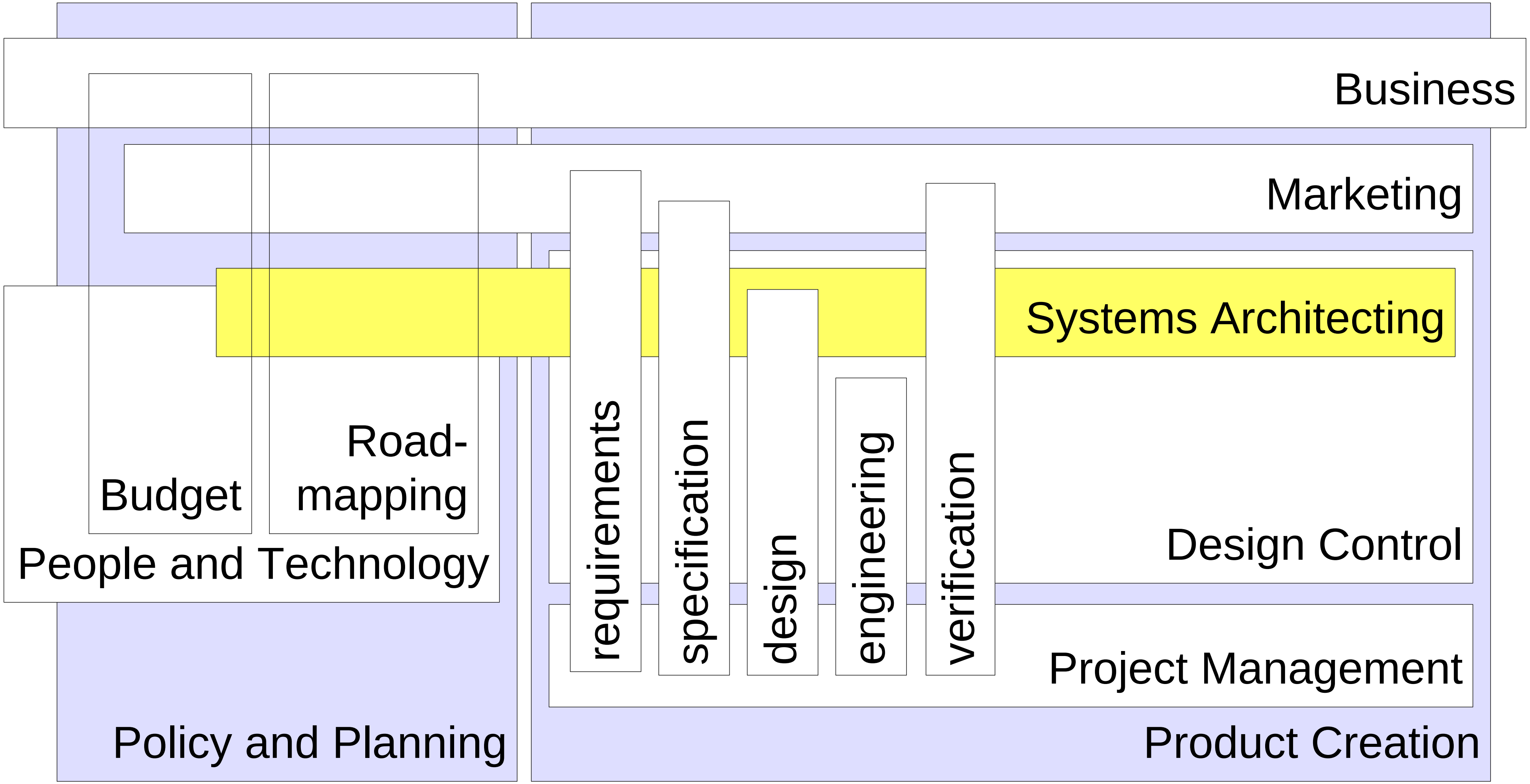
February 15, 2024
status: concept
version: 2.3



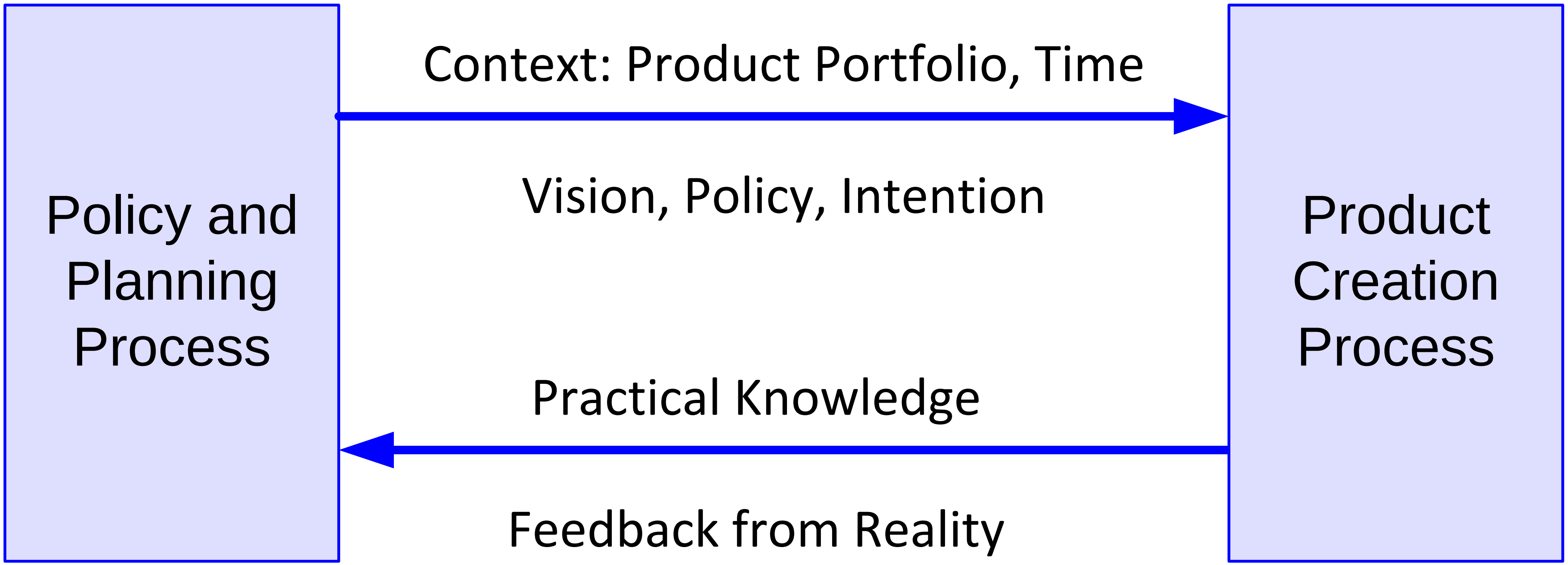
System Architecting Process in Business Context



Map of System Architecting Process and Neighborhood



System Architecting Relation between PPP and PCP



System Architecting Key Issues

key words

balance

consistency

integrity

simplicity

elegance

stakeholder satisfaction

balancing acts

External ⟷ internal requirements

Short term needs ⟷ long term interests

Efforts ⟷ risks from requirements to verification

Mutual influence of detailed designs

Value ⟷ costs

example trade-offs

performance

functionality

qualities

synergy

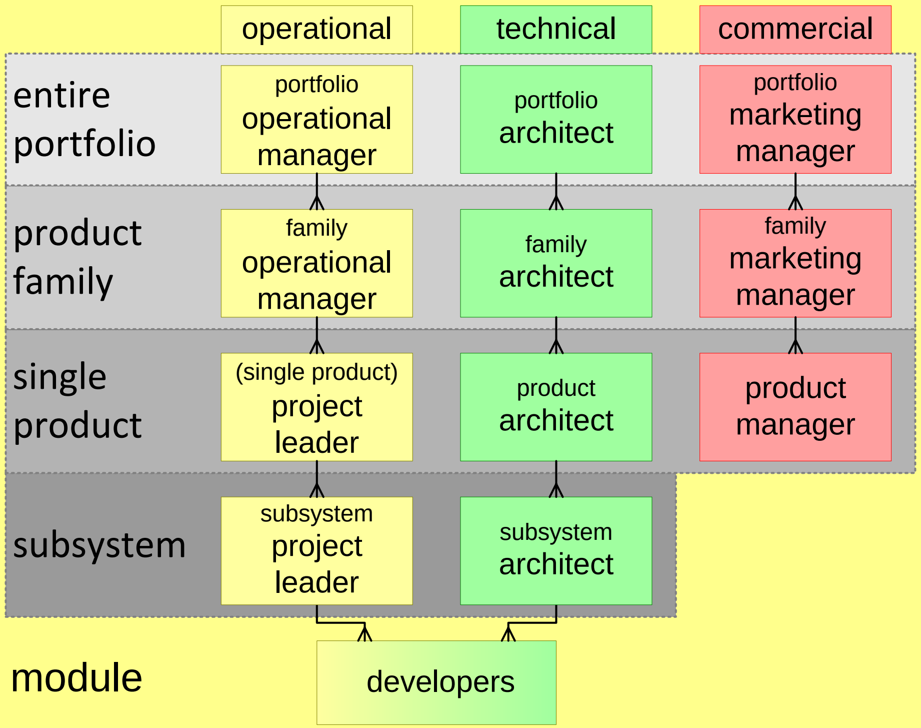
specific solution

Exercise Product Creation Process

Make a map with names of individuals in the **operational organization** of one project and its context

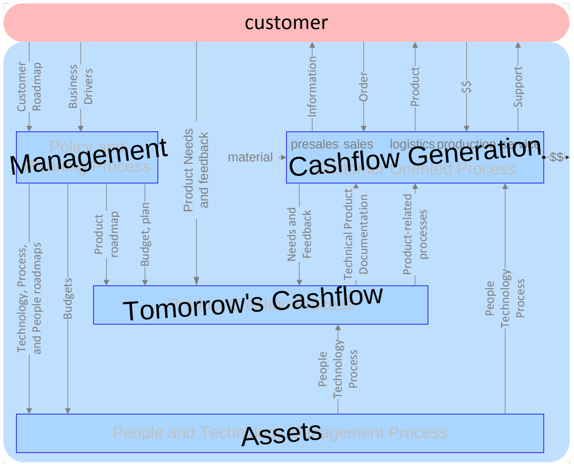
Identify the **relationships** of the **project core team**:

- **geographical**
- **organizational**
- **psychological**

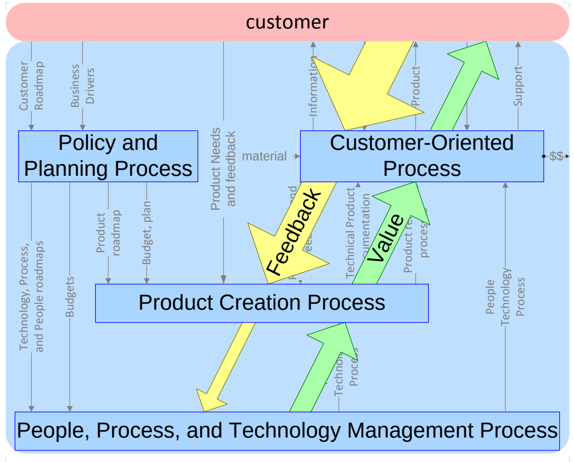


Process Decomposition of a Business

Importance in Financial terms



Value Chain and Feedback Flow

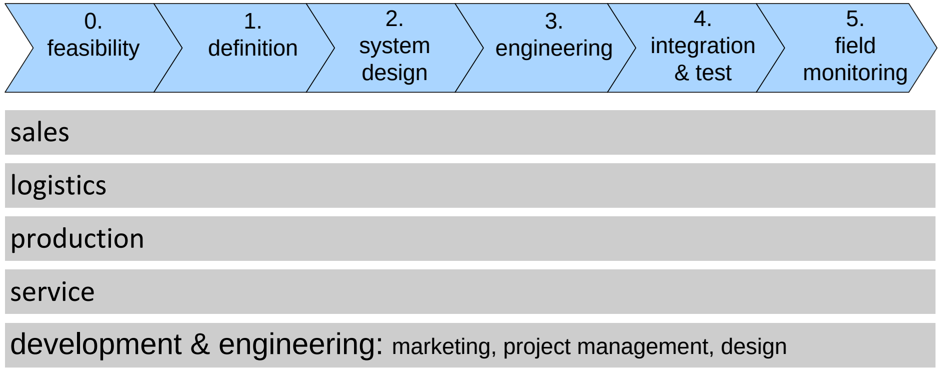


intentionally left blank

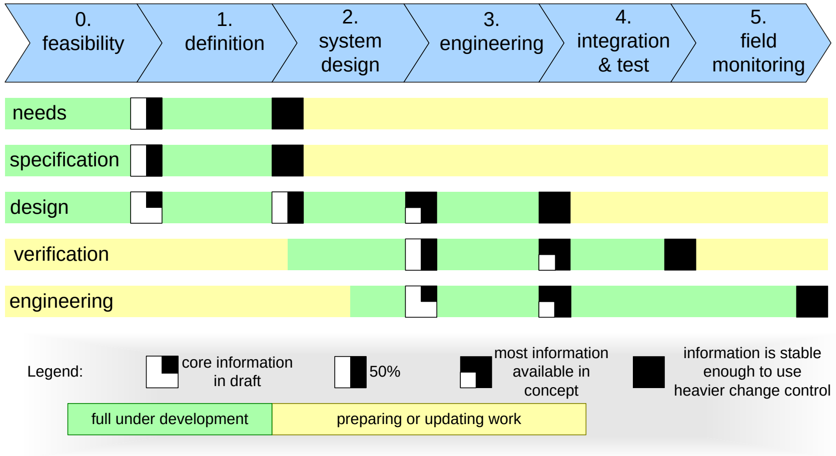
intentionally left blank

Product Creation Process

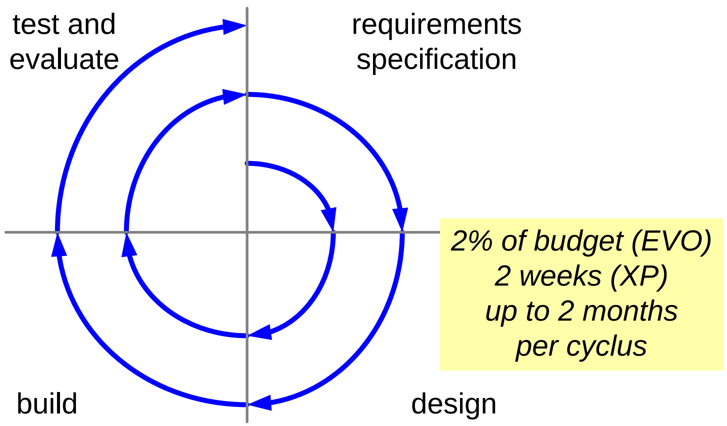
PCP involves **all** disciplines, much more than D&E

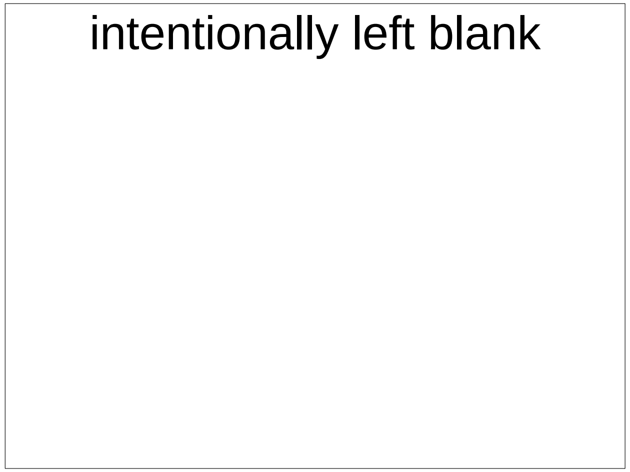


Phased Process



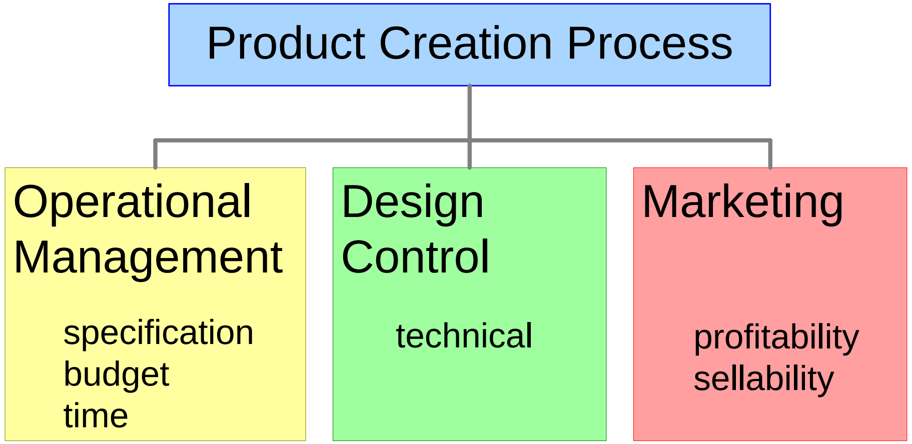
Incremental Development



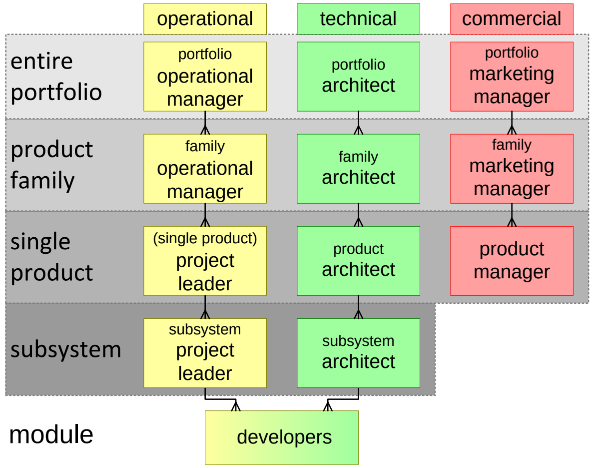


PCP Decomposition and Operational Management

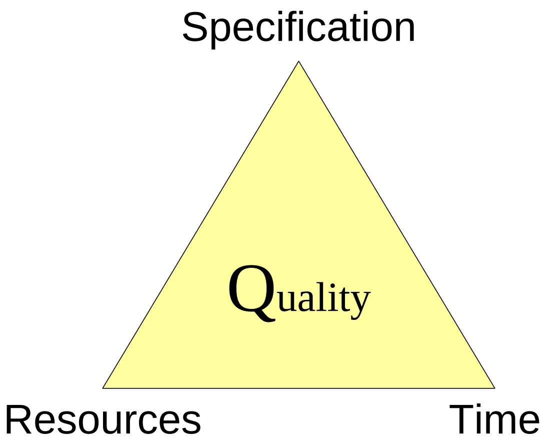
PCP decomposition



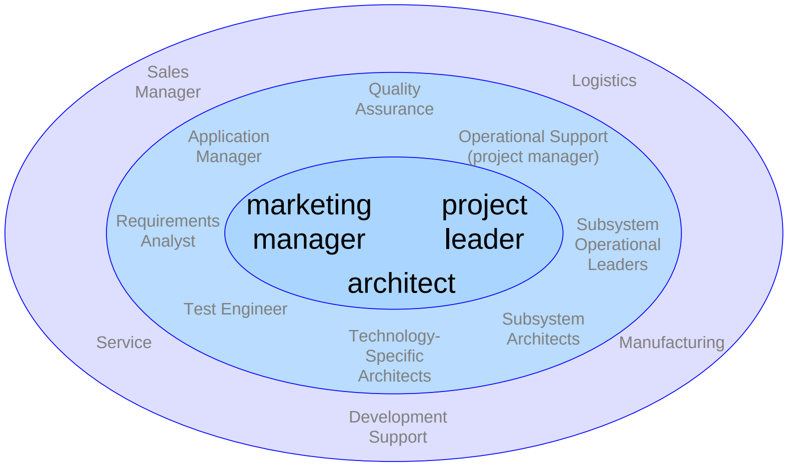
Architecture at all levels; From portfolio to subsystem



Operational Commitment

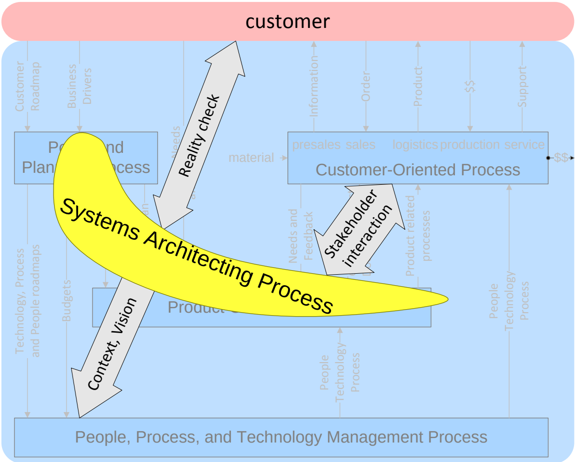


Core: Operational + Technical + Commercial



System Architecture Process

In Business Context



Key Issues

key words

balance

consistency

integrity

simplicity

elegance

stakeholder satisfaction

balancing acts

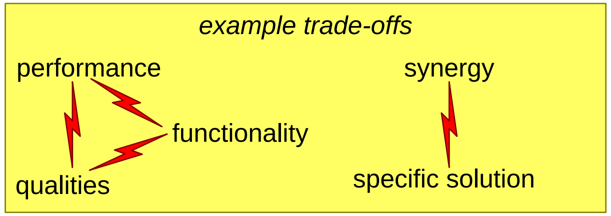
External ↔ internal requirements

Short term needs ↔ long term interests

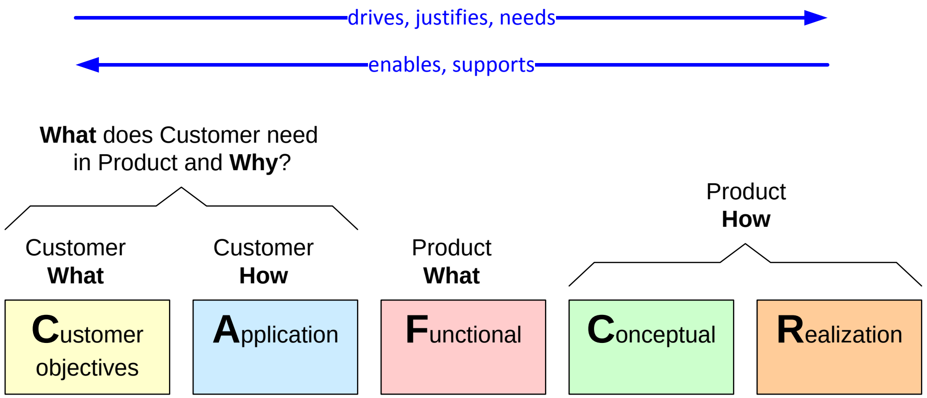
Efforts ↔ risks from requirements to verification

Mutual influence of detailed designs

Value ↔ costs



5 Views



intentionally left blank

Role and Task of the System Architect

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

The role and the task of the system architect are described in this module.

February 15, 2024
status: preliminary draft
version: 1.0

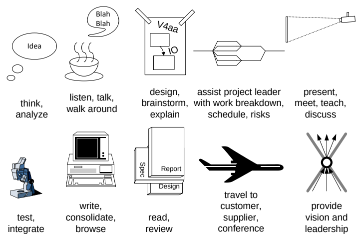
The Role and Task of the System Architect

by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

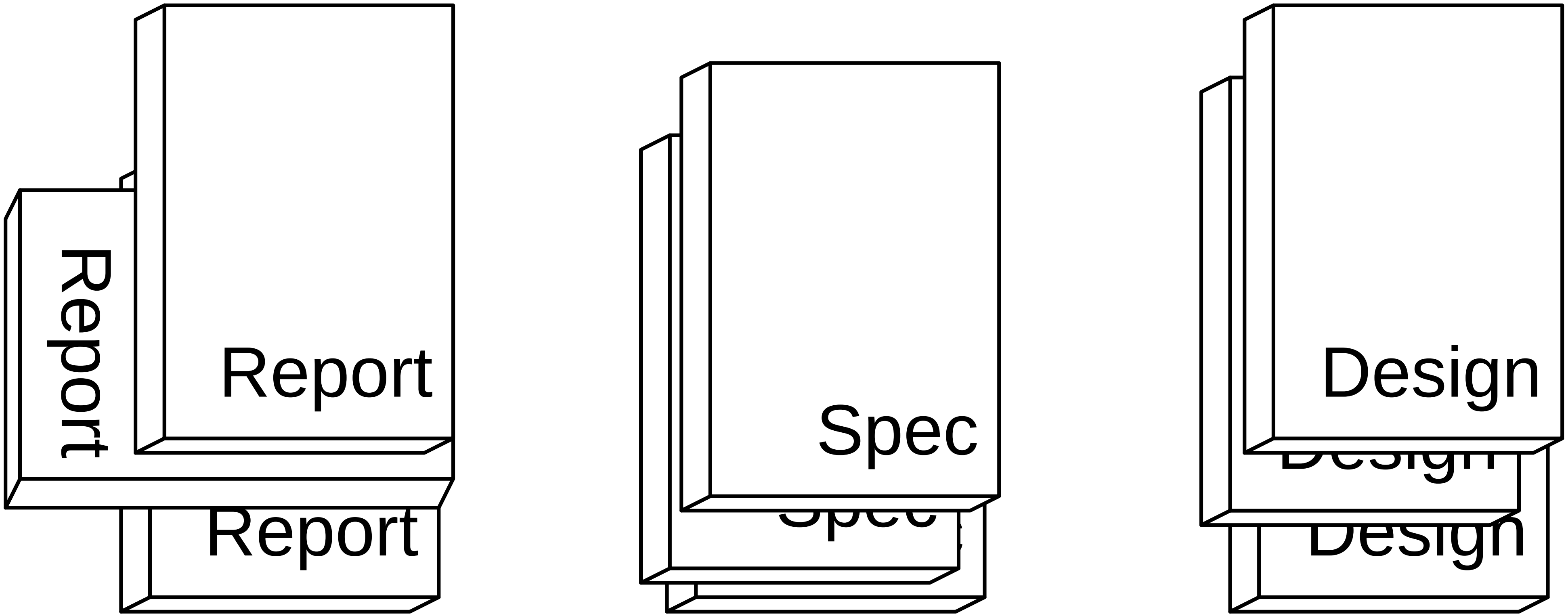
Abstract

The role of the system architect is described from three viewpoints: deliverables, responsibilities and activities. This description shows the inherent tension in this role: a small set of hard deliverables, covering a fuzzy set of responsibilities, hiding an enormous amount of barely visible day-to-day work.

February 15, 2024
status: concept
version: 2.0



Deliverables of the System Architect



List of Deliverables

Customer and Life-Cycle Needs *(what is needed)*

System Specification *(what will be realized)*

Design Specification *(how the system will be realized)*

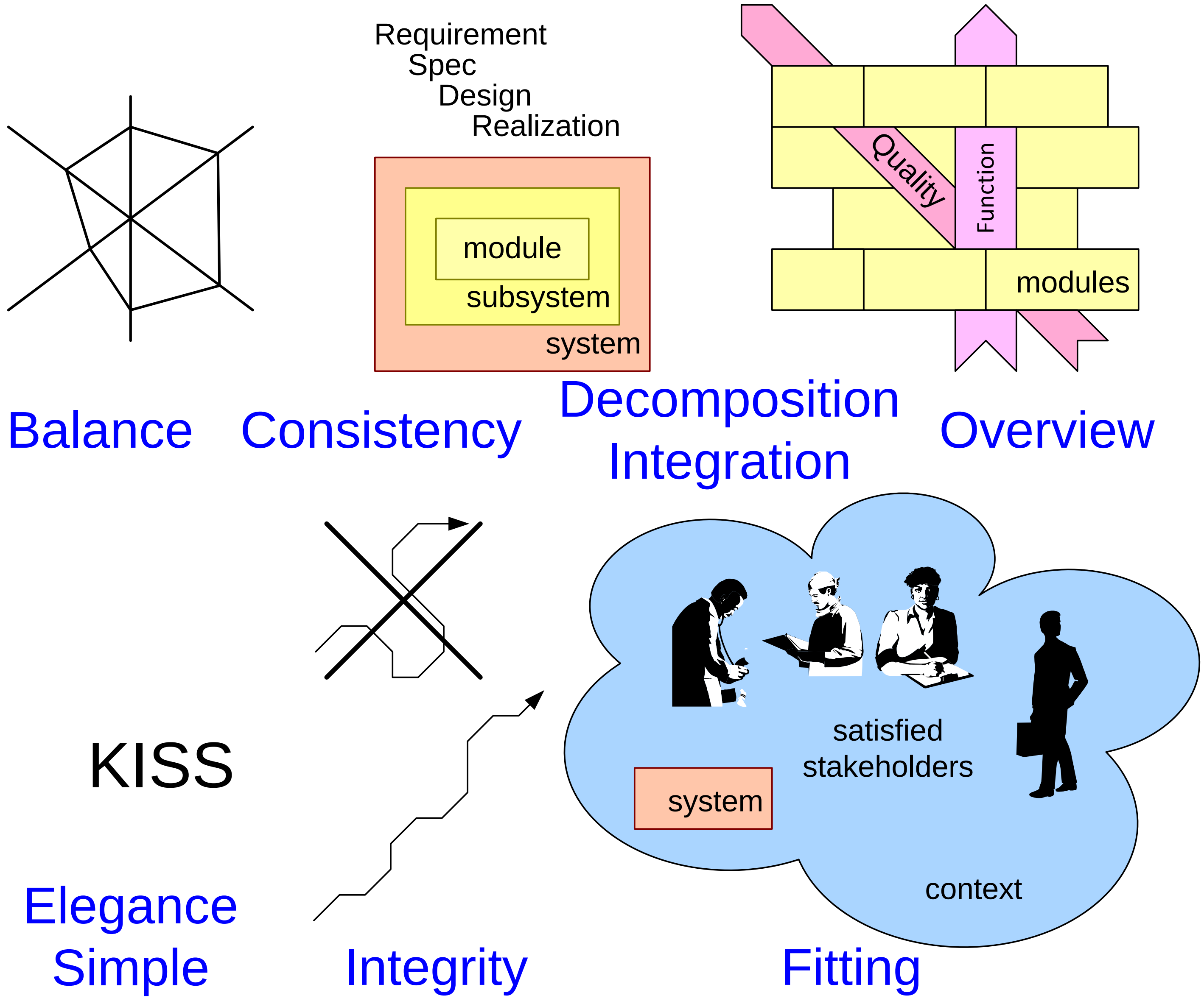
Verification Specification *(how the system will be verified)*

Verification Report *(the result of the verification)*

Feasibility Report *(the results of a feasibility study)*

Roadmap

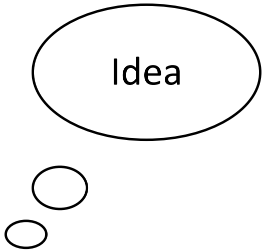
Responsibilities of the System Architect



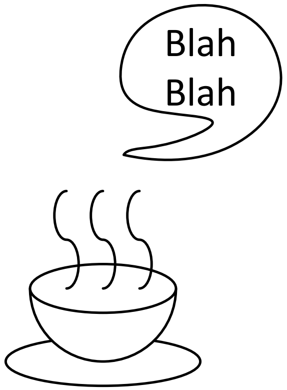
Examples of Secondary Responsibilities

responsibility	primary owner
business plan, profit	business manager
schedule, resources	project leader
market, saleability	marketing manager
technology	technology manager
process, people	line manager
detailed designs	engineers

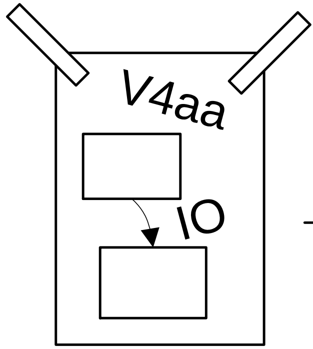
What does the System Architect do?



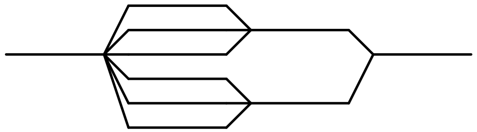
think,
analyze



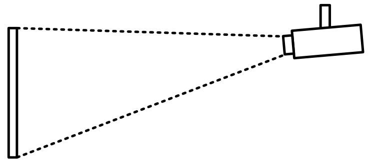
listen, talk,
walk around



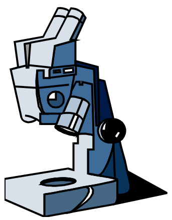
design,
brainstorm,
explain



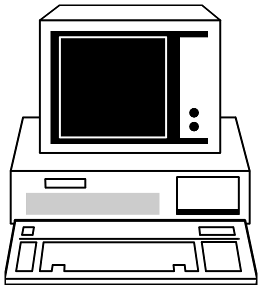
assist project leader
with work breakdown,
schedule, risks



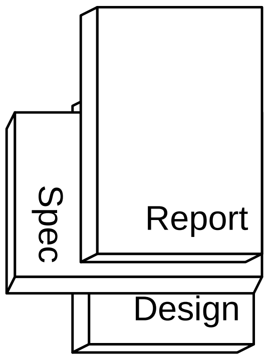
present,
meet, teach,
discuss



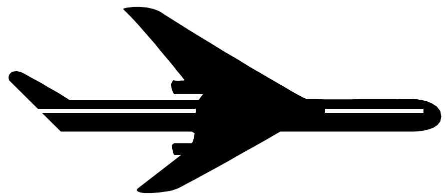
test,
integrate



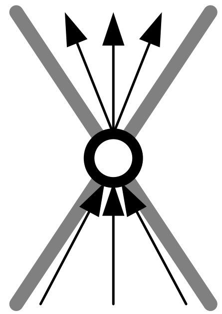
write,
consolidate,
browse



read,
review



travel to
customer,
supplier,
conference



provide
vision and
leadership

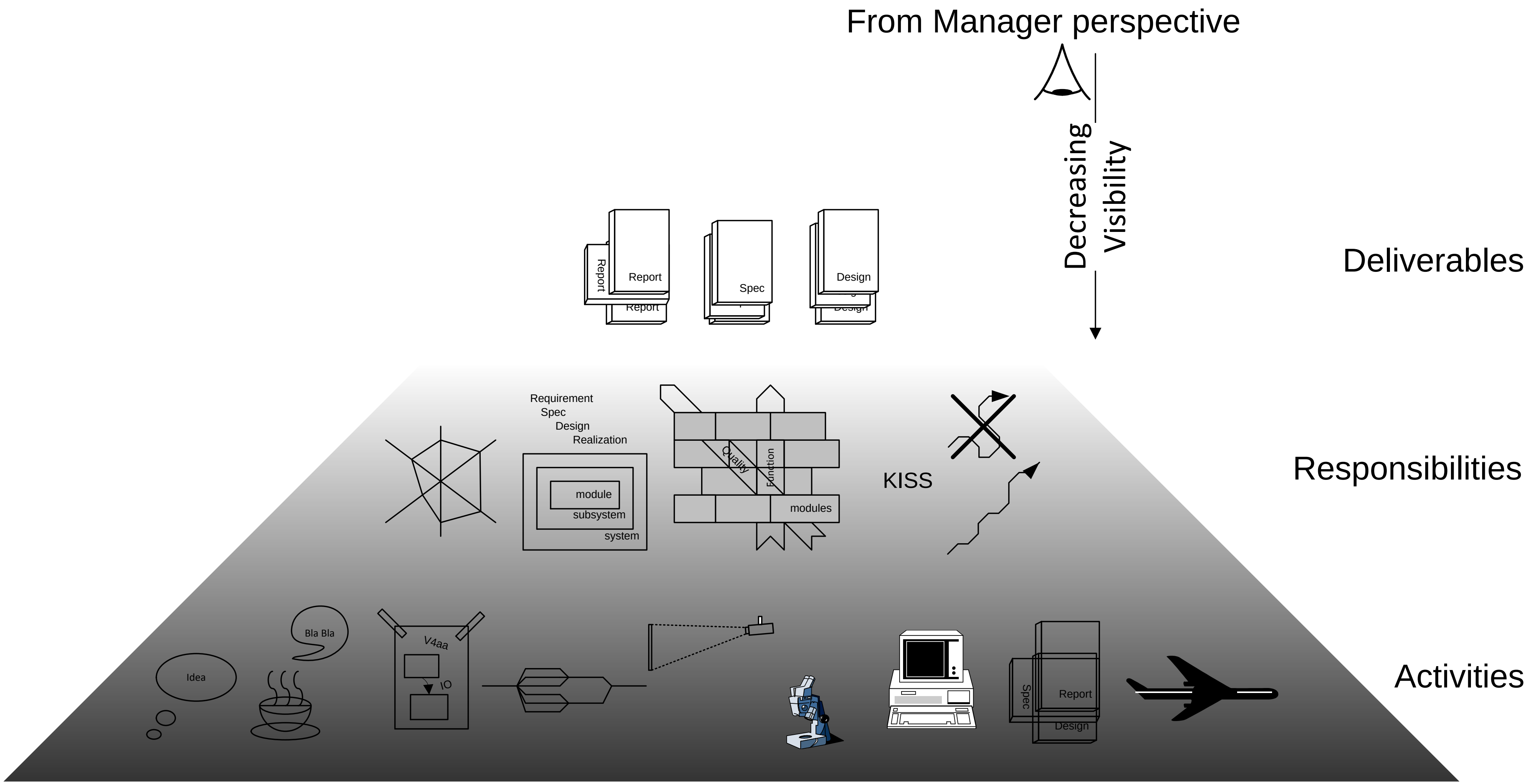
From Detail to Overview

		Quantity per year (order-of-magnitude)	architect time per item
consolidation in deliverables	driving views	10	100 h
meetings	shared issues	10^2	1 h
informal contacts	touched details	10^4	0.5 - 10 min
sampling scanning	seen details	$10^5 - 10^6$	0.1 - 1 sec
	product details	$10^7 - 10^{10}$	
	real-world facts	infinite	

Reality or Virtuality?

Abstractions only exist for concrete facts.

Visible Output versus Invisible Work



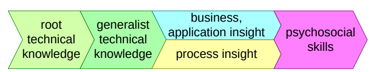
The Awakening of a System Architect

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

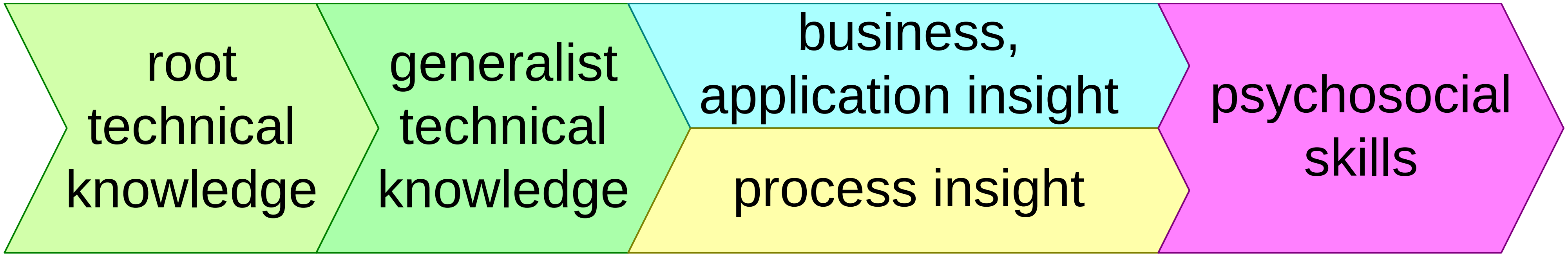
Abstract

The typical phases of a system architect development are described, beginning at the fundamental technology knowledge, with a later broadening in technology and in business aspects. Finally the subtlety of individual human beings is taken into account.

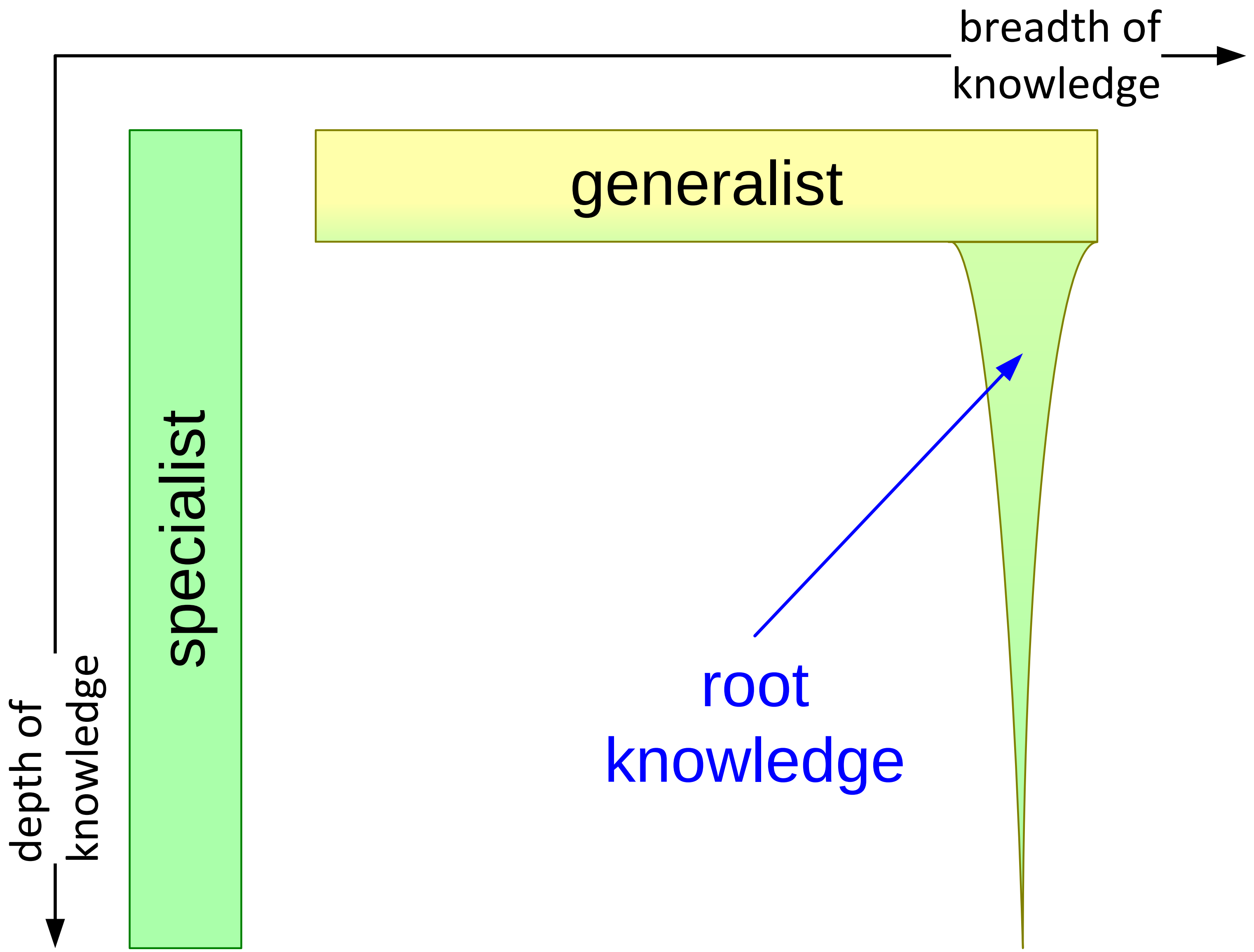
February 15, 2024
status: concept
version: 1.1



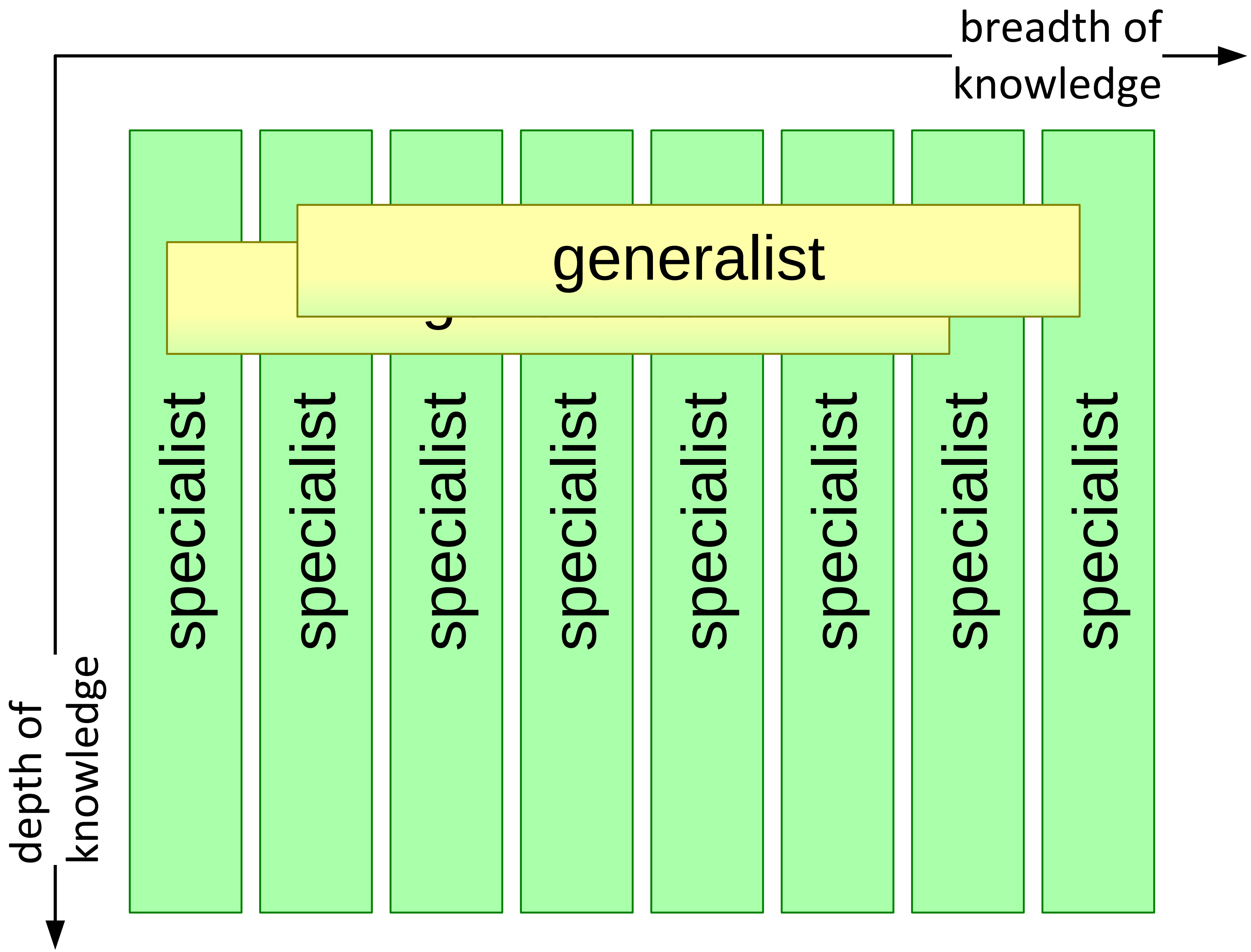
Typical Growth of a System Architect



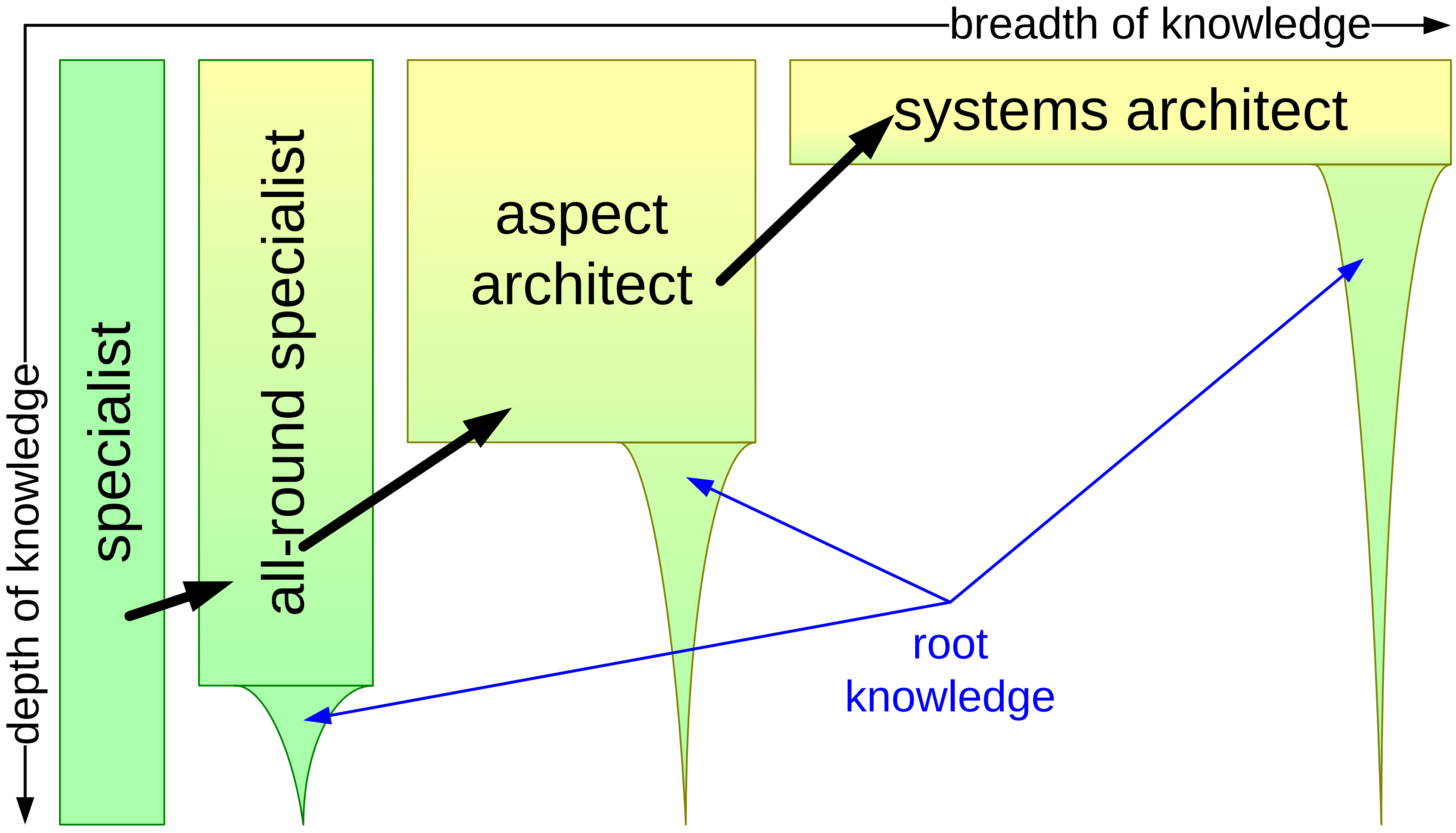
Generalist versus Specialist



Generalists and Specialists are Complementary



Spectrum from Specialist to System Architect



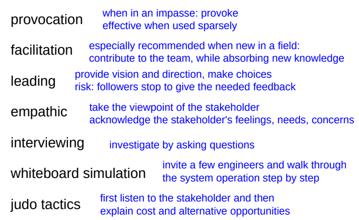
Architecting Interaction Styles

by *Gerrit Muller* USN-SE
e-mail: `gaudisite@gmail.com`
`www.gaudisite.nl`

Abstract

A system architects needs skills to apply different interactions styles, depending on the circumstances. This document discusses the following interaction styles: provocation, facilitation, leading, empathic, interviewing, white board simulation, and judo tactics.

February 15, 2024
status: draft
version: 0.2



Architecting Styles

provocation	when in an impasse: provoke effective when used sparsely
facilitation	especially recommended when new in a field: contribute to the team, while absorbing new knowledge
leading	provide vision and direction, make choices risk: followers stop to give the needed feedback
empathic	take the viewpoint of the stakeholder acknowledge the stakeholder's feelings, needs, concerns
interviewing	investigate by asking questions
whiteboard simulation	invite a few engineers and walk through the system operation step by step
judo tactics	first listen to the stakeholder and then explain cost and alternative opportunities

Exercise Role and Task of the System Architect

Role play with 3 roles and optional observer:

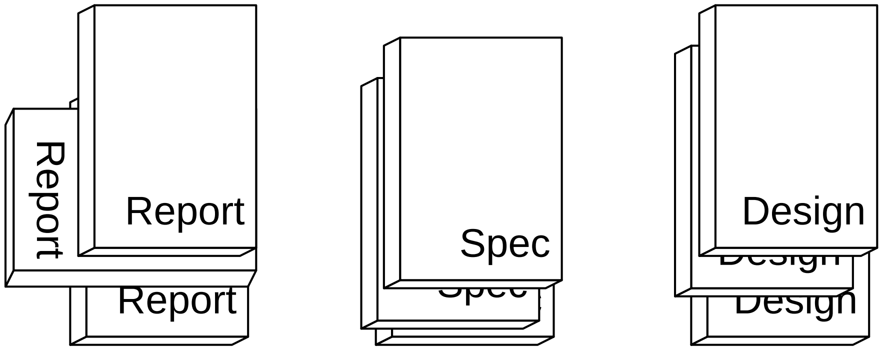
- 1 operational leader (project leader)
- 1 system architect
- 1 marketing manager
- 1 observer (optional)

Discuss the definition (business relevance, specification, and planning) of a travel e-mail mate.

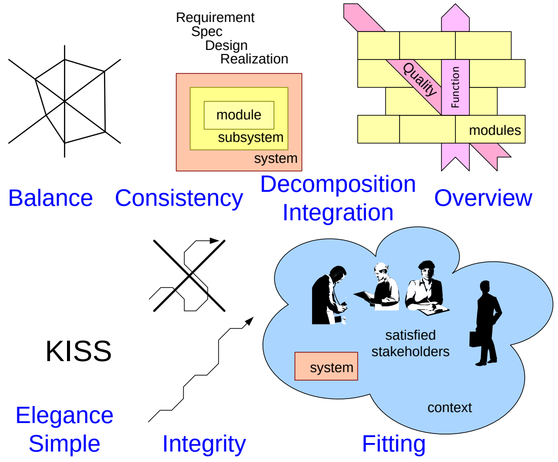
Present (max. 2 flips) the result and the process (the relation and interaction of the three roles).

Role and Task of a System Architect

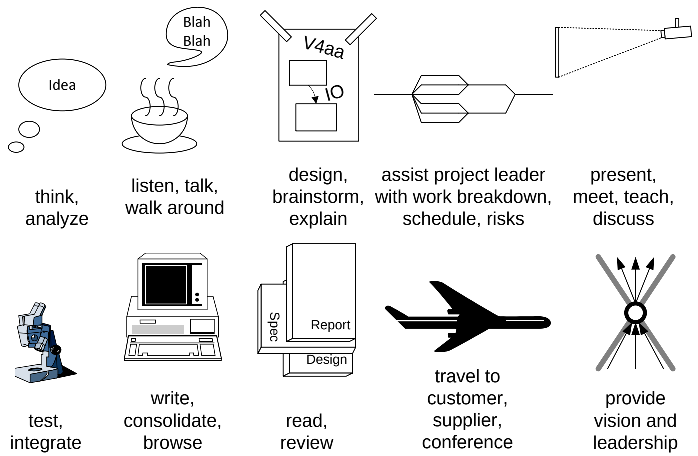
Deliverables



Responsibilities



Daily Activities

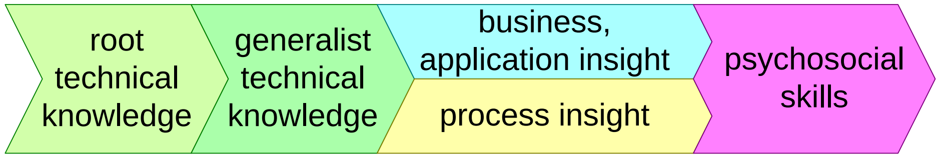


From detail to overview

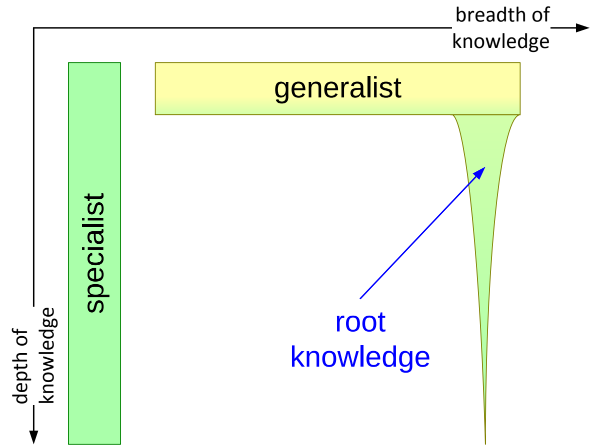
		Quantity per year (order-of-magnitude)	architect time per item
consolidation in deliverables	driving views	10	100 h
meetings	shared issues	10^2	1 h
informal contacts	touched details	10^4	0.5 - 10 min
sampling scanning	seen details	$10^5 - 10^6$	0.1 - 1 sec
	product details	$10^7 - 10^{10}$	
	real-world facts	infinite	

Personal characteristics of a System Architect

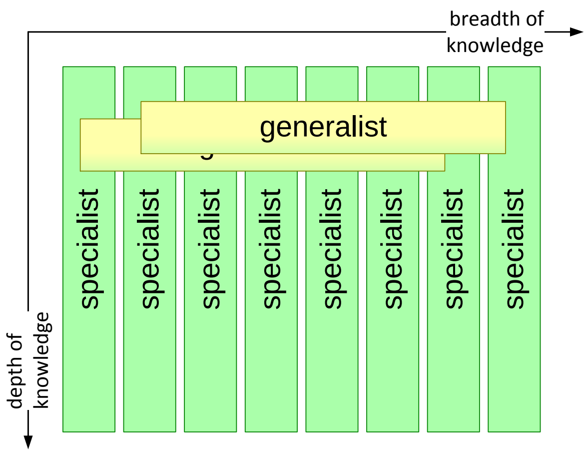
Typical growth of a Architect



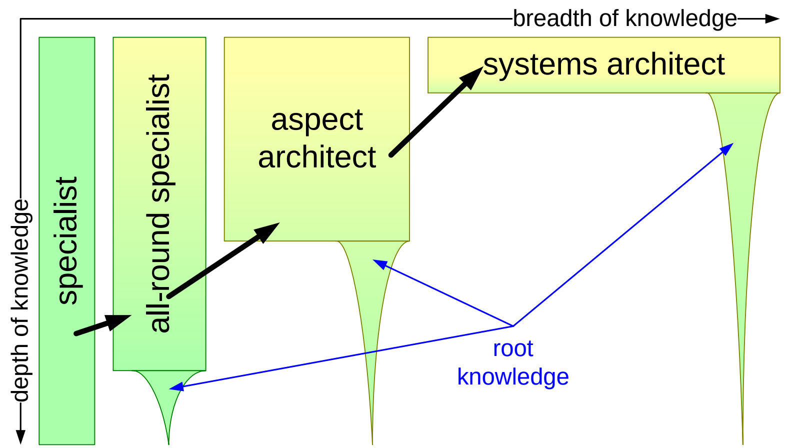
Generalist vs Specialist



Complementary Roles



Role Spectrum



Module Requirements

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

This module addresses requirements: What are requirements? How to find, select, and consolidate requirements?

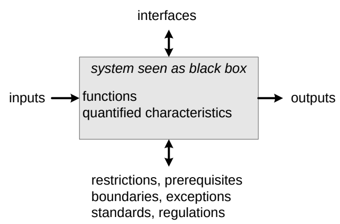
February 15, 2024
status: concept
version: 1.4

Fundamentals of Requirements Engineering

by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

Requirements engineering is one of the systems engineering pillars. In this document we discuss the fundamentals of systems engineering, such as the transformation of needs into specification, the need to prescribe *what* rather than *how*, and the requirements when writing requirements.

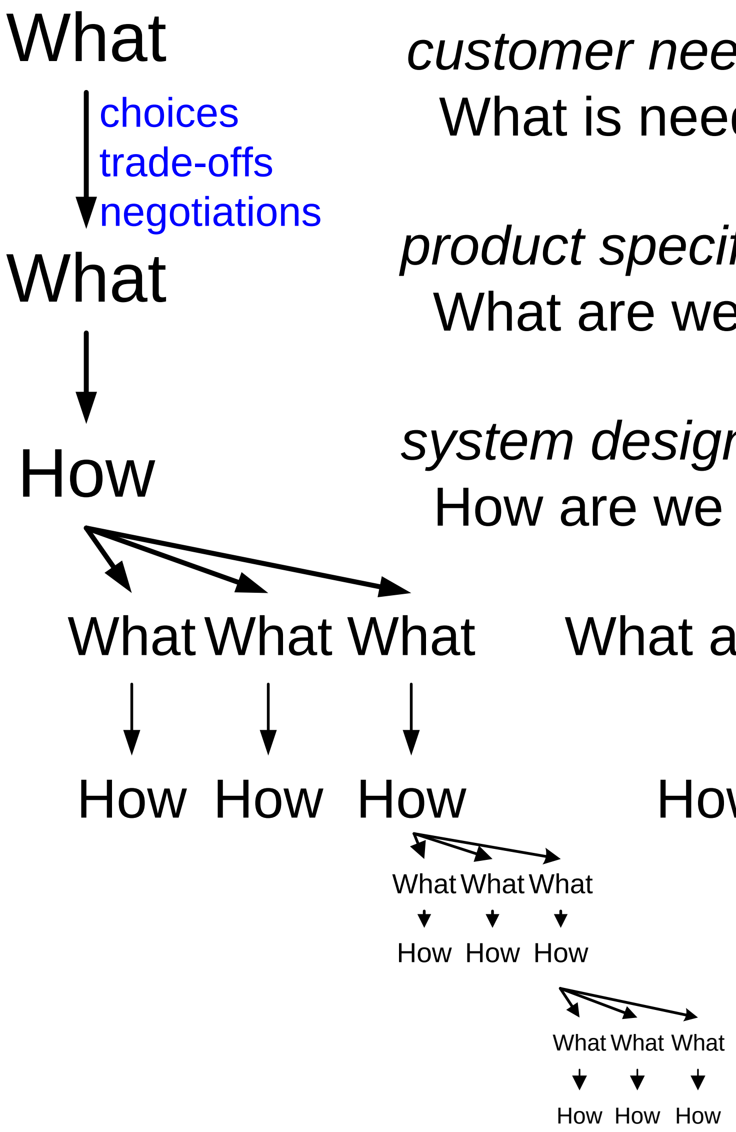


February 15, 2024
status: concept
version: 0.1

Definition of "Requirement"

Requirements describing the needs of the customer: ***Customer Needs***
Requirements describing the characteristics of the final resulting system (product): ***System (Product) Specification*** The ***requirements management process*** recursively applies this definition for every level of decomposition.
Requirements describing the needs of the company itself over the life cycle: ***Life Cycle Needs***

Flow of Requirements



customer needs:
What is needed by the customer?

product specification:
What are we going to realize?

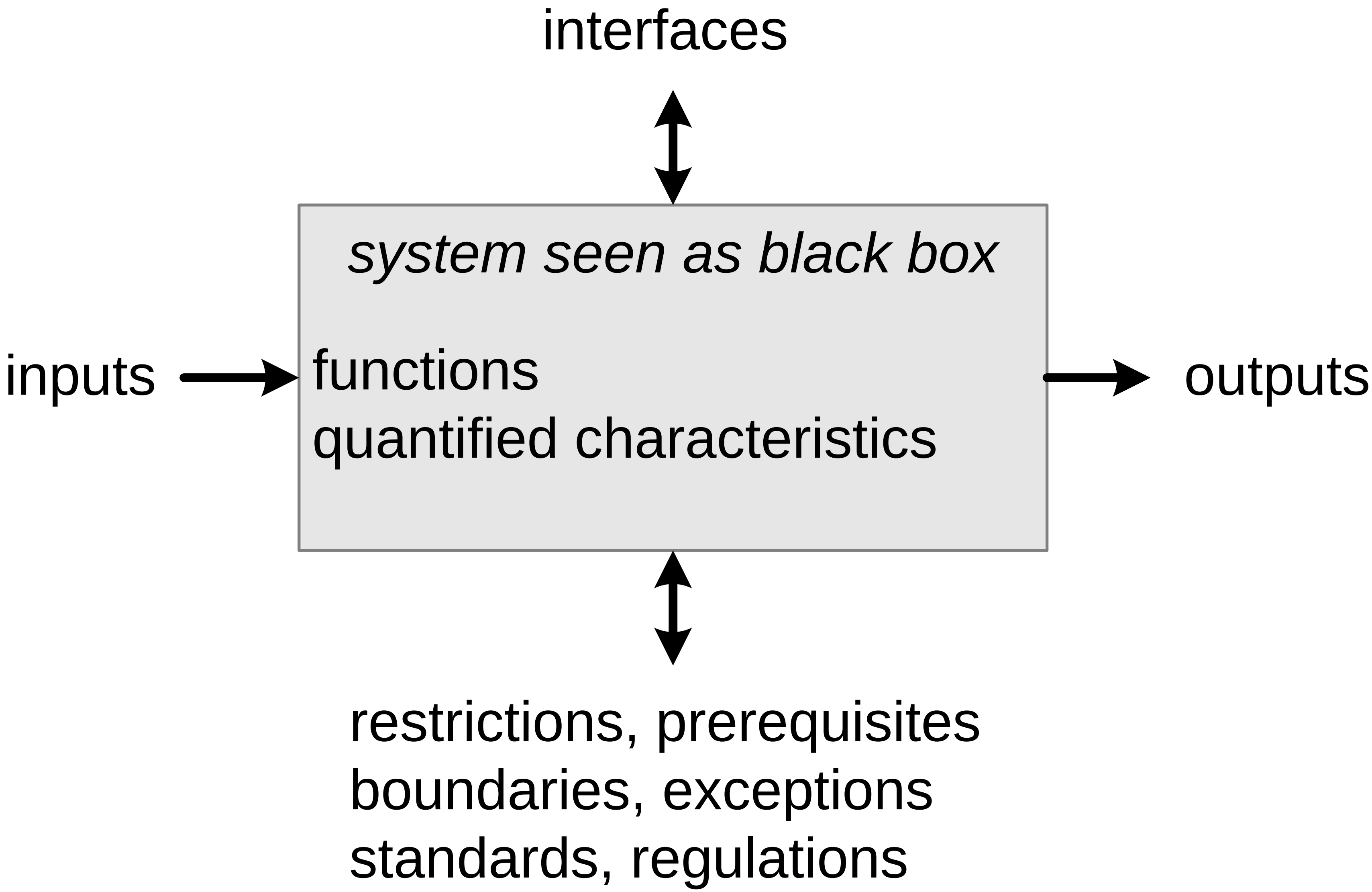
system design:
How are we going to realize the product?

What are the subsystems we will realize?

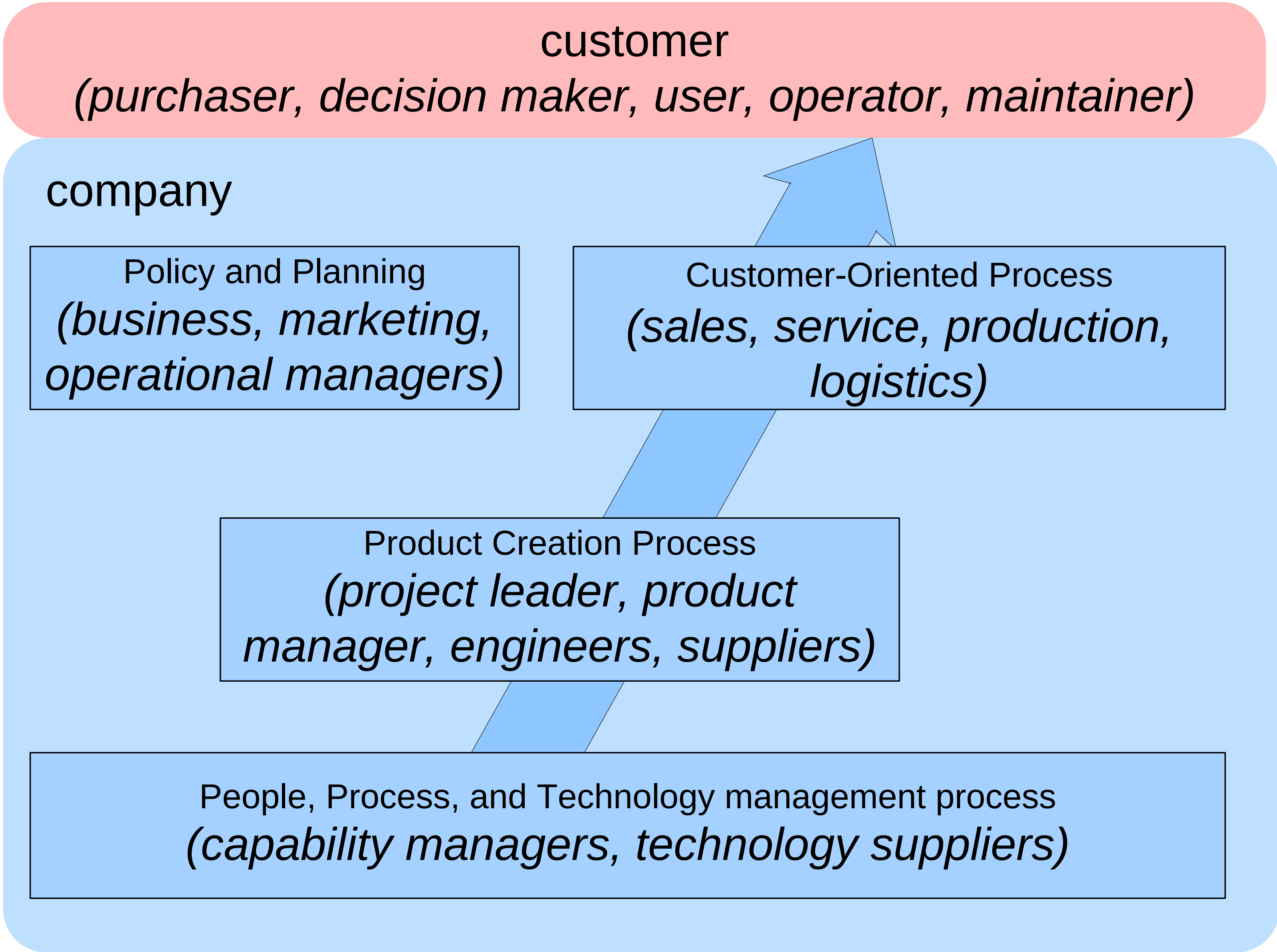
How will the subsystems be realized?

up to "atomic" components

System as a Black Box



Stakeholders w.r.t. Requirements



The “Formal” Requirements for Requirements

Specific

Unambiguous

Verifiable

Quantifiable

Measurable

Complete

Traceable

The Requirements to Enable Human Use

Accessible

Understandable

Low threshold

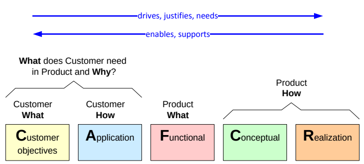
Short introduction to basic "CAFCR" model

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

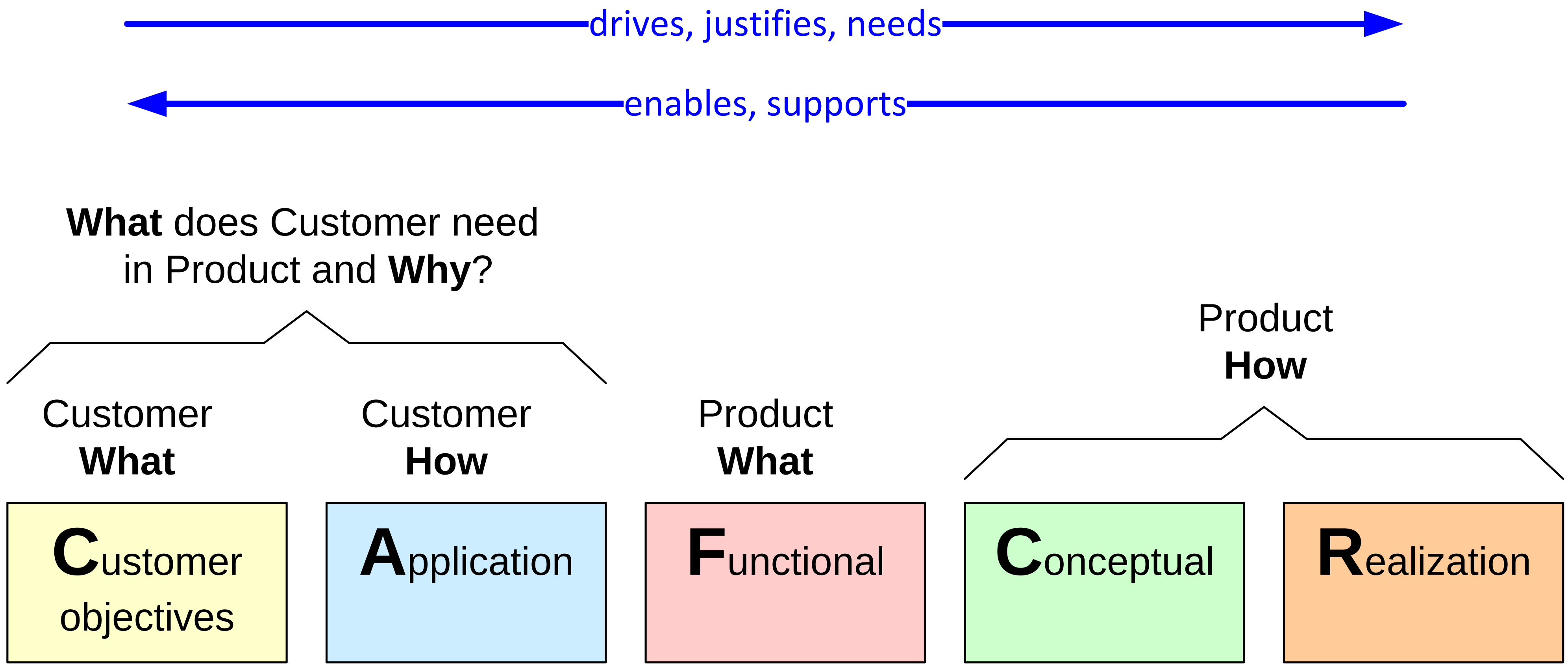
Abstract

The basic "CAFCR" reference model is described, which is used to describe a system in relation to its context. The main stakeholder in the context is the customer. The question "Who is the customer?" is addressed.

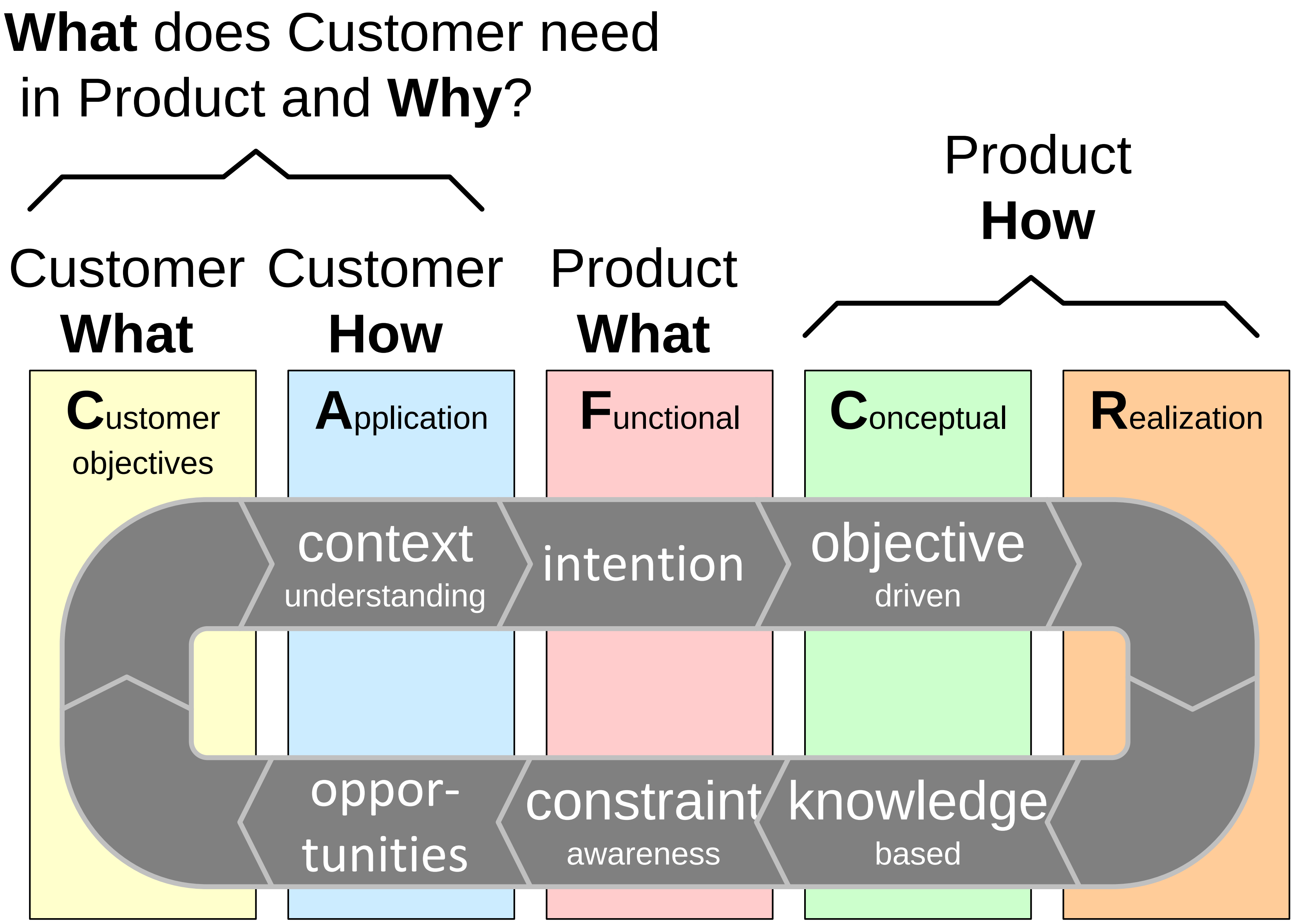
February 15, 2024
status: draft
version: 0.4



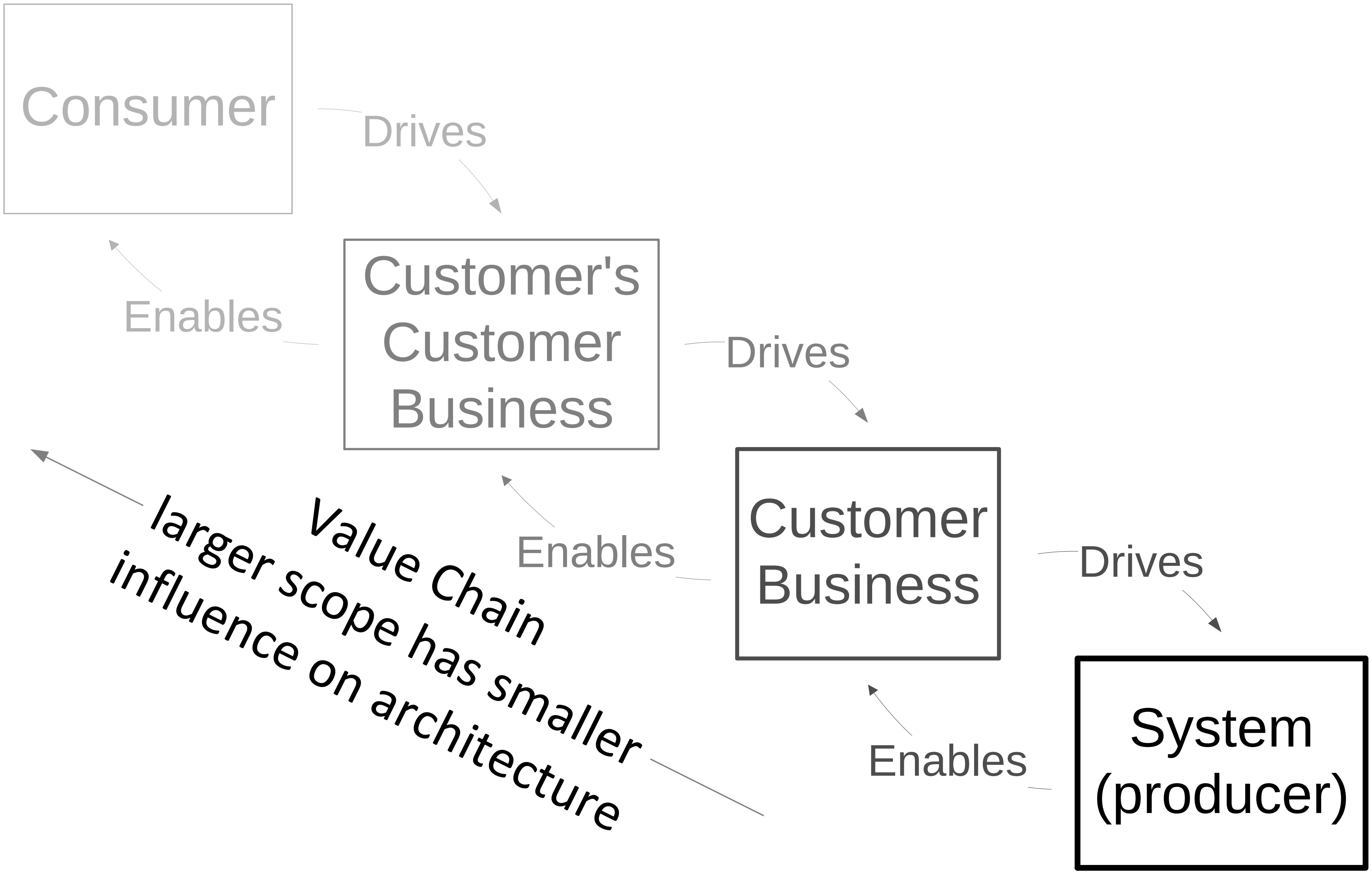
The "CAFCR" model



Integrating CAFCR



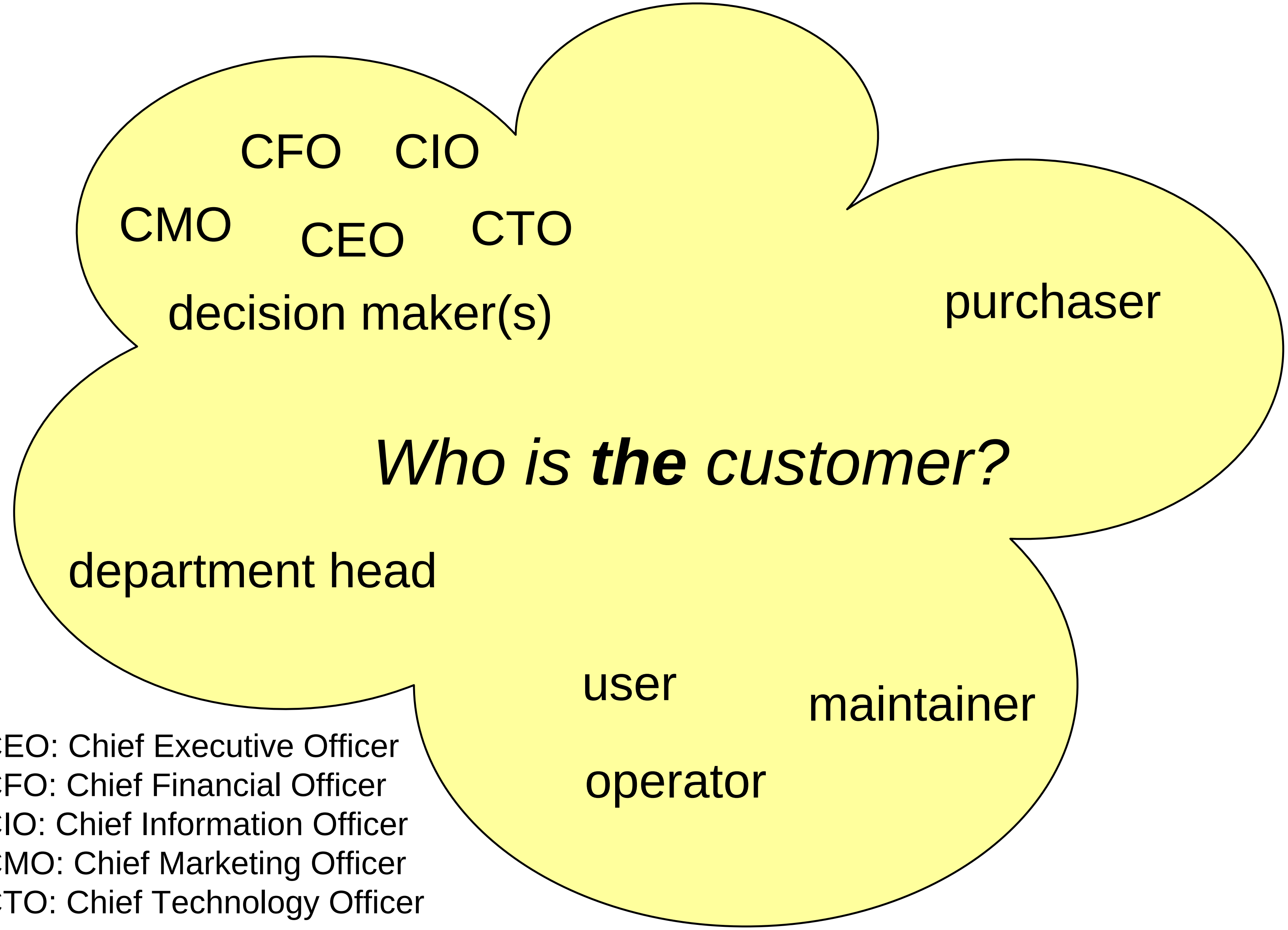
CAFCR can be applied recursively



Market segmentation

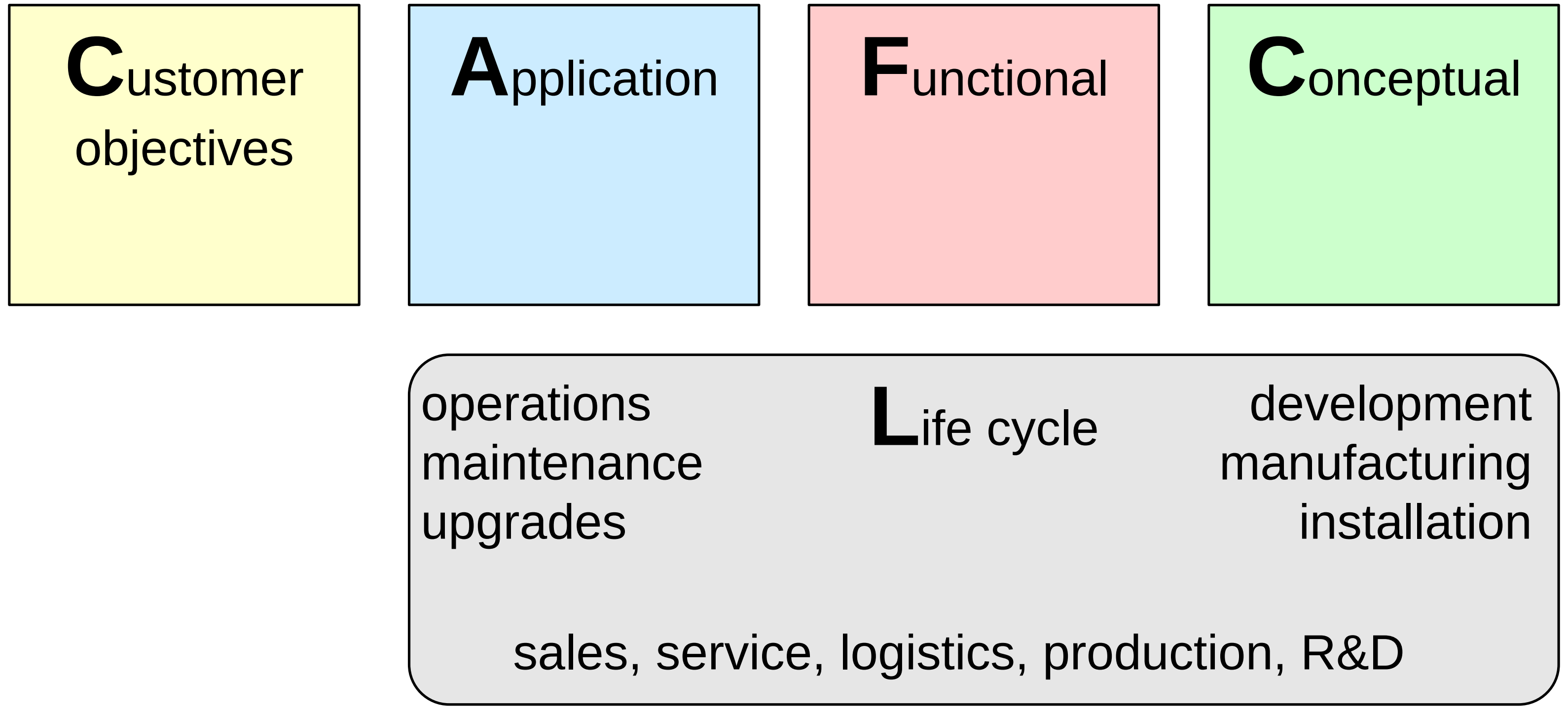
segmentation axis	examples
geographical	USA, UK, Germany, Japan, China
business model	profit, non profit
economics	high end versus cost constrained
consumers	youth, elderly
outlet	retailer, provider, OEM, consumer direct

Example of a small buying organization



CEO: Chief Executive Officer
CFO: Chief Financial Officer
CIO: Chief Information Officer
CMO: Chief Marketing Officer
CTO: Chief Technology Officer

CAFCR+ model; Life Cycle View

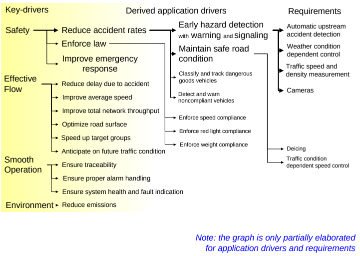


Key Drivers How To

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

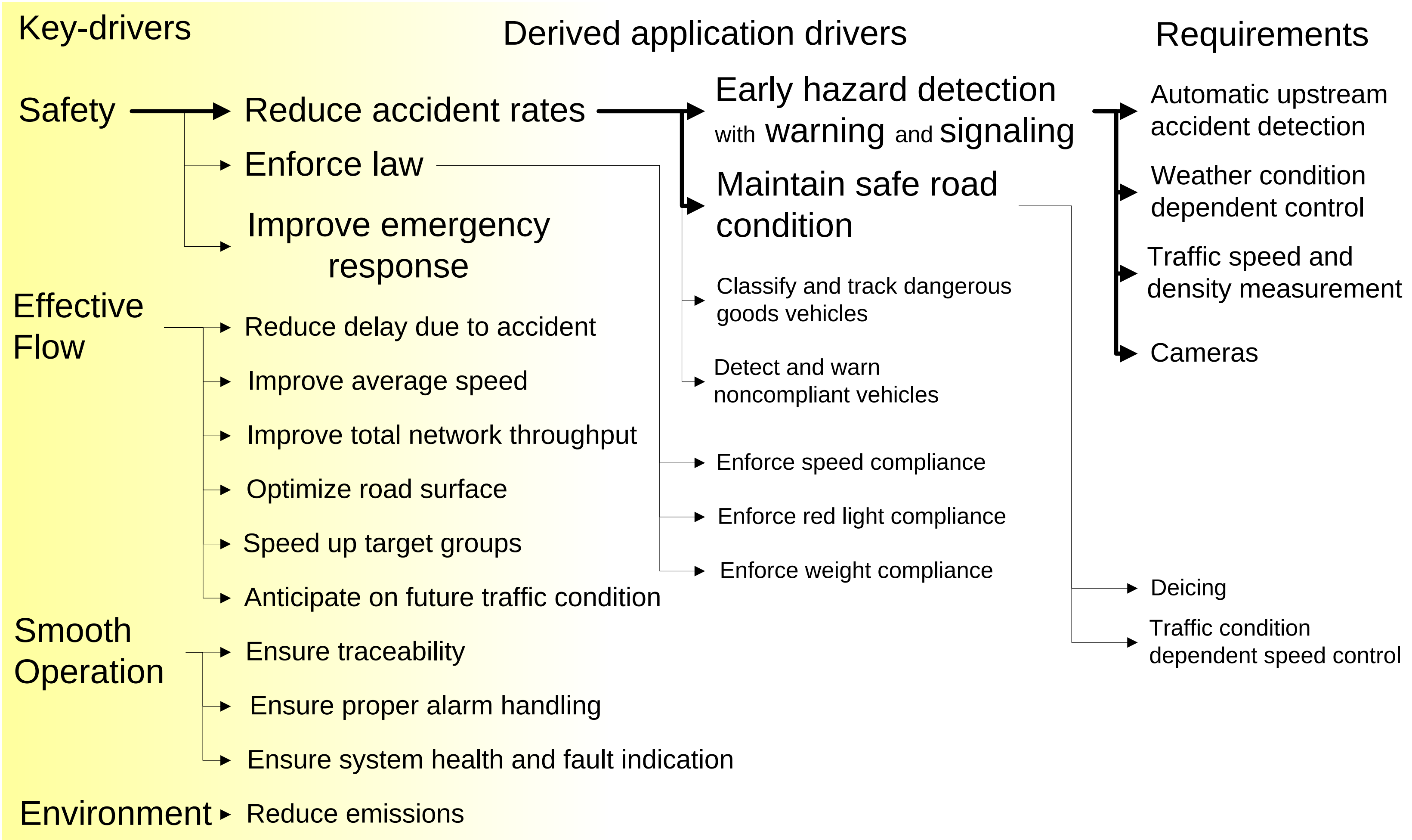
The notion of ”business key drivers” is introduced and a method is described to link these key drivers to the product specification.



Distribution

February 15, 2024
status: draft
version: 0.2

Example Motorway Management Analysis



Note: the graph is only partially elaborated for application drivers and requirements

Method to create Key Driver Graph

• Define the scope specific.	in terms of stakeholder or market segments
• Acquire and analyze facts	extract facts from the product specification and ask why questions about the specification of existing products.
• Build a graph of relations between drivers and requirements by means of brainstorming and discussions	where requirements may have multiple drivers
• Obtain feedback	discuss with customers, observe their reactions
• Iterate many times	increased understanding often triggers the move of issues from driver to requirement or vice versa and rephrasing

Recommendation for the Definition of Key Drivers

• Limit the number of key-drivers	minimal 3, maximal 6
• Don't leave out the obvious key-drivers	for instance the well-known main function of the product
• Use short names, recognized by the customer.	
• Use market-/customer- specific names, no generic names	for instance replace "ease of use" by "minimal number of actions for experienced users", or "efficiency" by "integral cost per patient"
• Do not worry about the exact boundary between Customer Objective and Application	create clear goal means relations

Transformation of Key Drivers into Requirements

Customer **What**	Customer **How**	Product **What**
Customer objectives	**A**pplication	**F**unctional
Key (Customer) Drivers →	Derived Application Drivers →	Requirements
goal	means may be skipped or articulated by several intermediate steps	functions interfaces performance figures

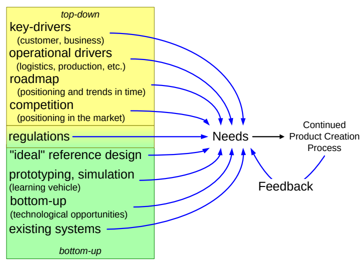
Requirements Elicitation and Selection

by *Gerrit Muller*   University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

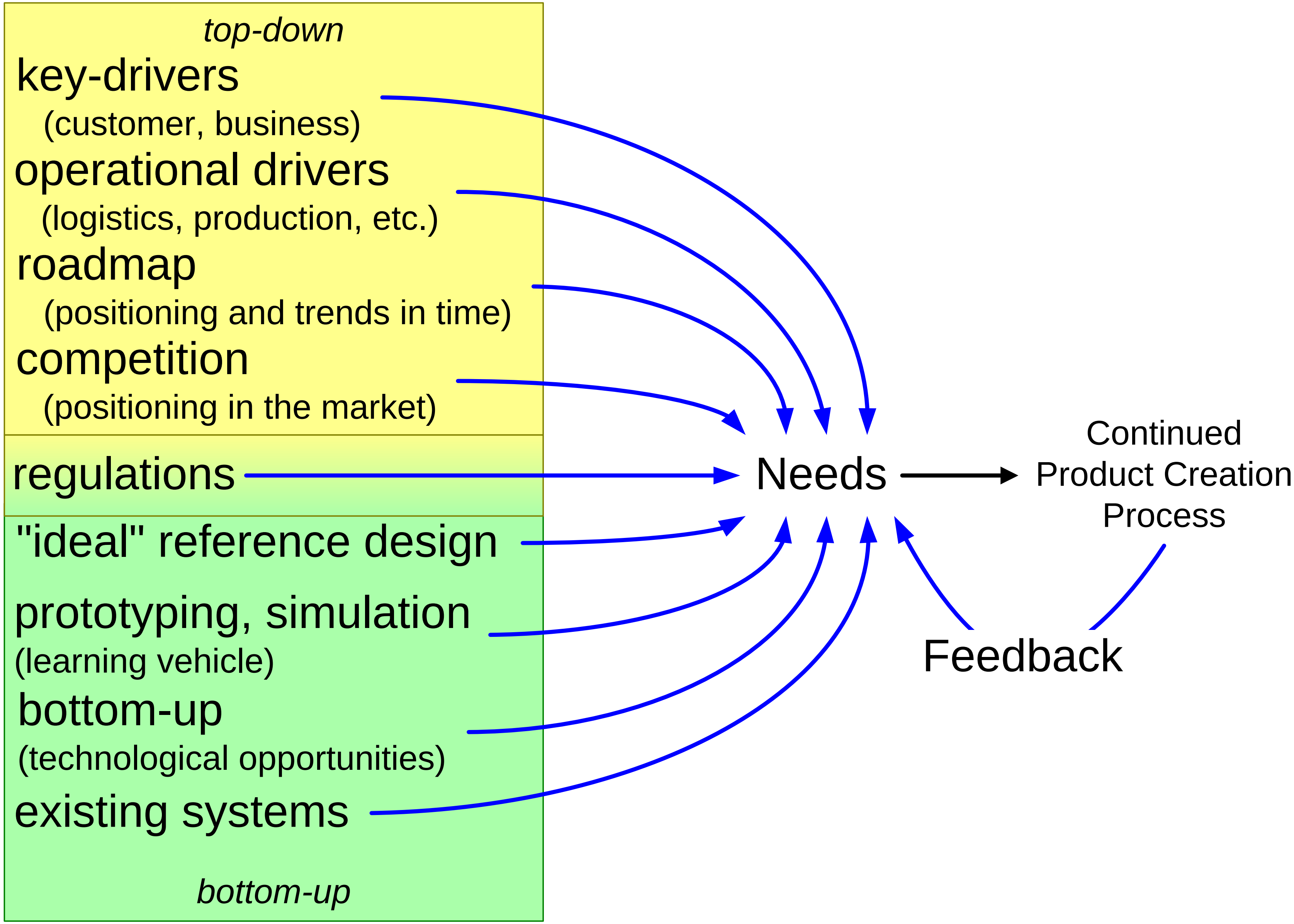
Abstract

An elicitation method for needs is described using many different viewpoints. A selection process with a coarse and a fine selection is described to reduce the specification to an acceptable and feasible subset.

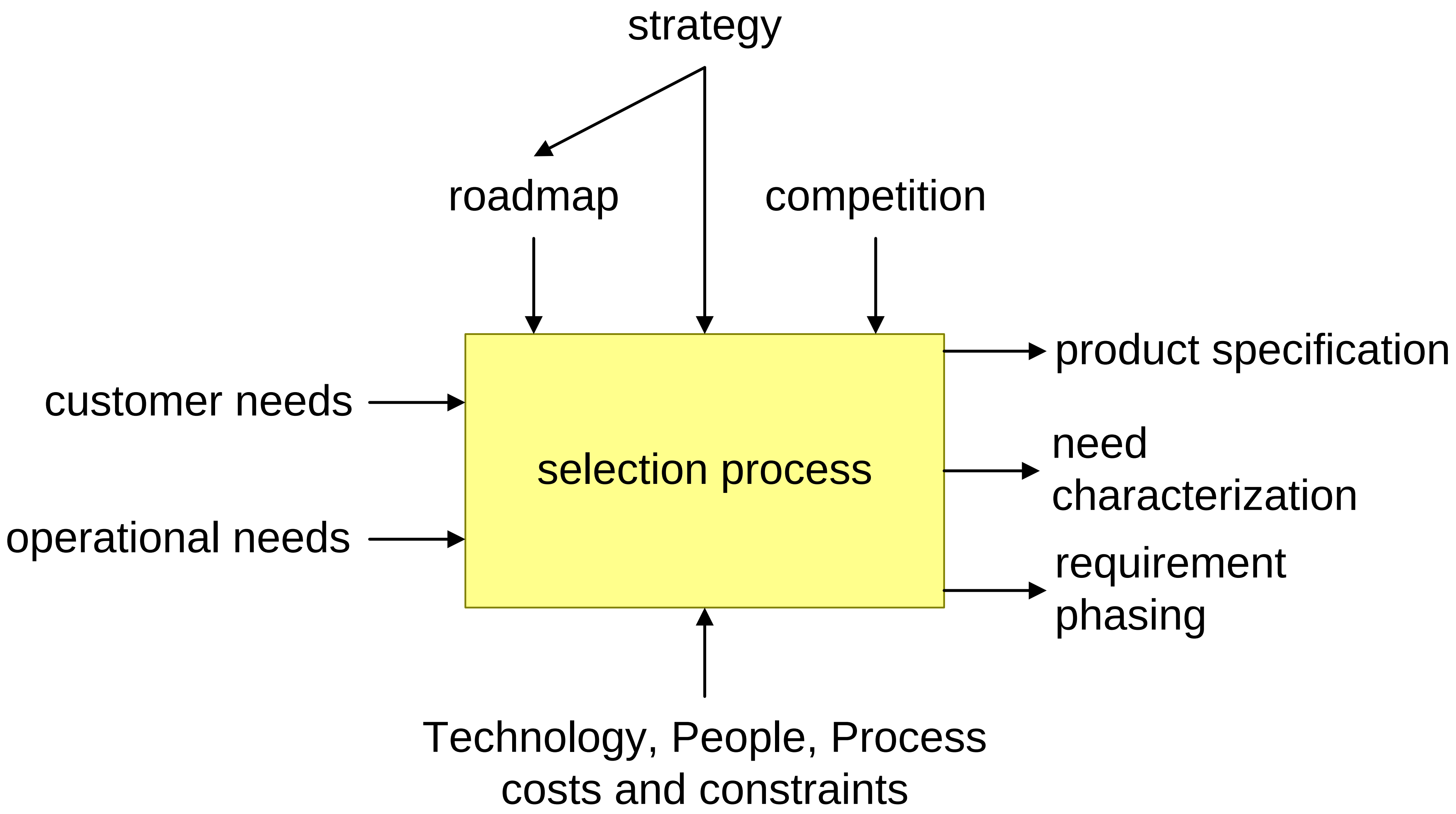
February 15, 2024
status: draft
version: 0



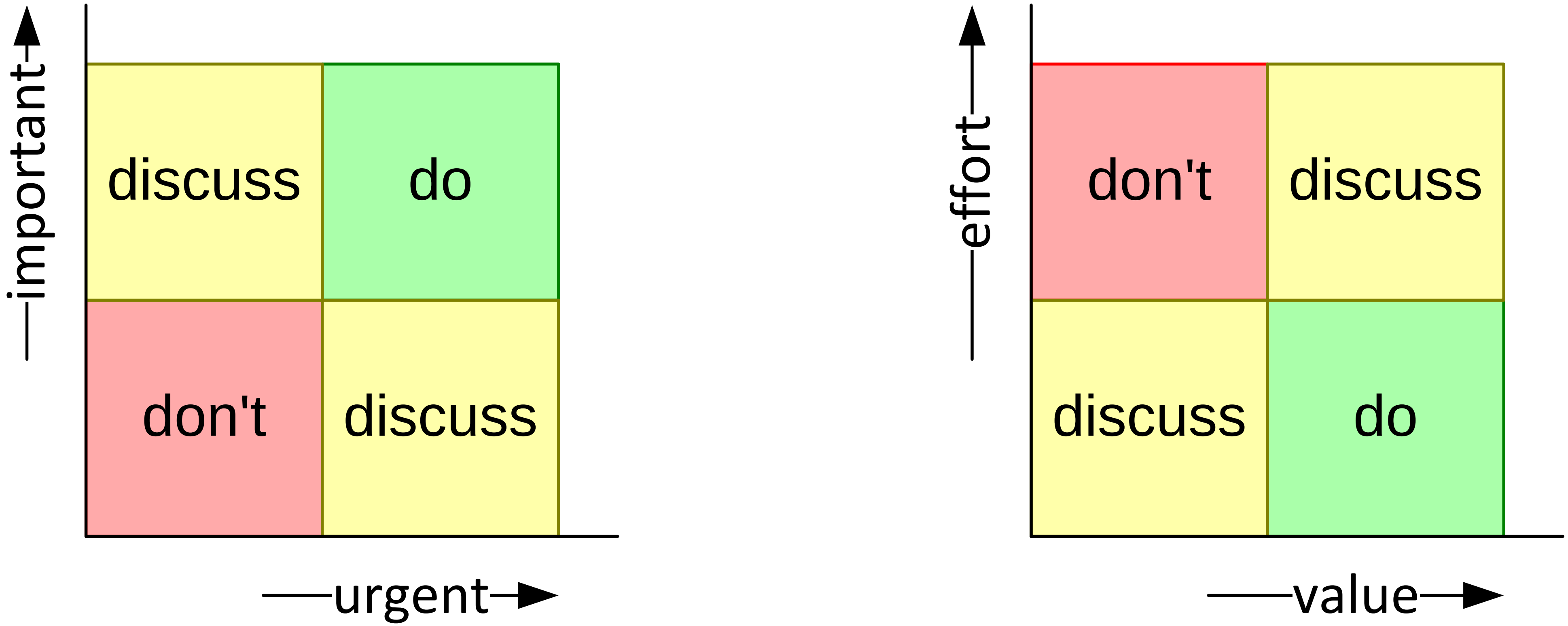
Complementary Viewpoints to Capture Requirements



Requirement Selection Process



Simple Qualification Method



Examples of Quantifiable Aspects

- Value for the customer
- (dis)satisfaction level for the customer
- Selling value (How much is the customer willing to pay?)
- Level of differentiation w.r.t. the competition
- Impact on the market share
- Impact on the profit margin

Use relative scale, e.g. 1..5 1=low value, 5 -high value

Ask several knowledgeable people to score

Discussion provides insight (don't fall in spreadsheet trap)

Exercise Requirements Capturing

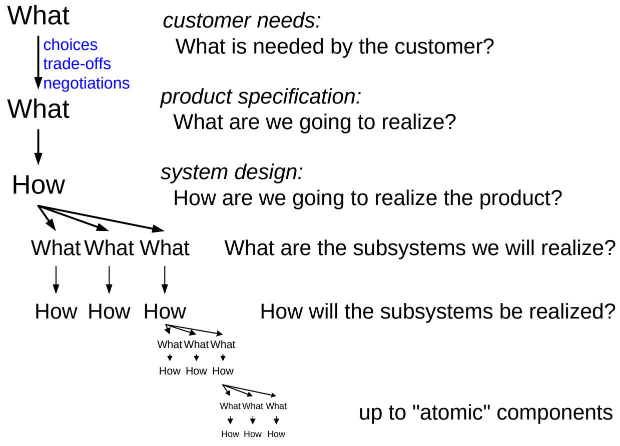
- Determine the key drivers for one particular product family.
- Translate these drivers into application drivers and derive from them the requirements.

Needs and Requirements

Needs, Specification, Requirements

Requirements describing the needs of the customer: ***Customer Needs***
Requirements describing the characteristics of the final resulting system (product): ***System (Product) Specification*** The ***requirements management process*** recursively applies this definition for every level of decomposition.
Requirements describing the needs of the company itself over the life cycle: ***Life Cycle Needs***

Flow of Requirements



Requirements for Requirements

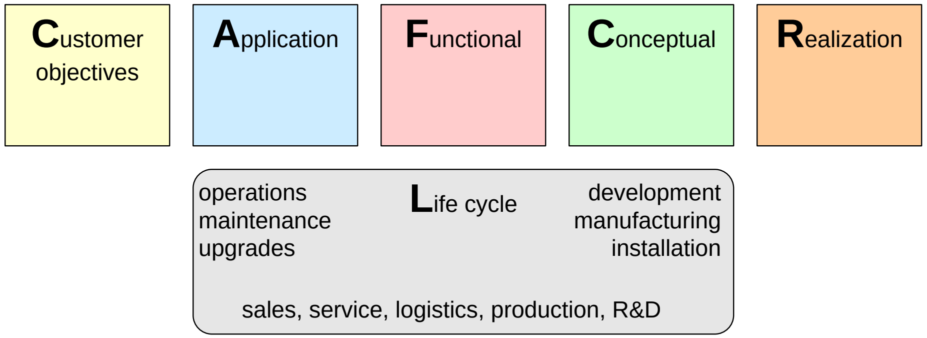
- Specific
- Unambiguous
- Verifiable
- Quantifiable
- Measurable
- Complete
- Traceable

Enable Human Use

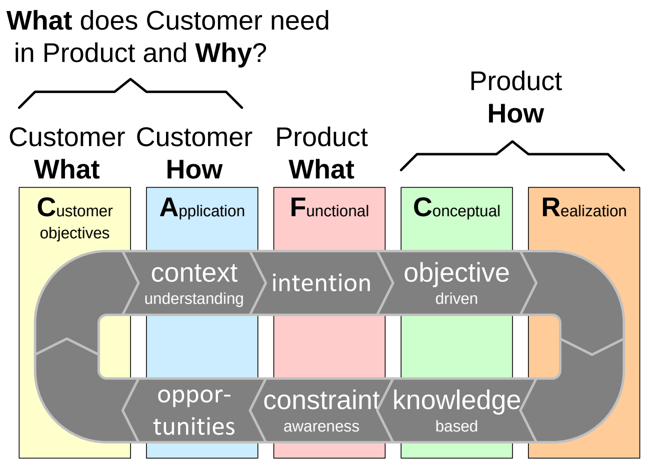
- Accessible
- Understandable
- Low threshold

CAFCR, Customer Key Driver Graph

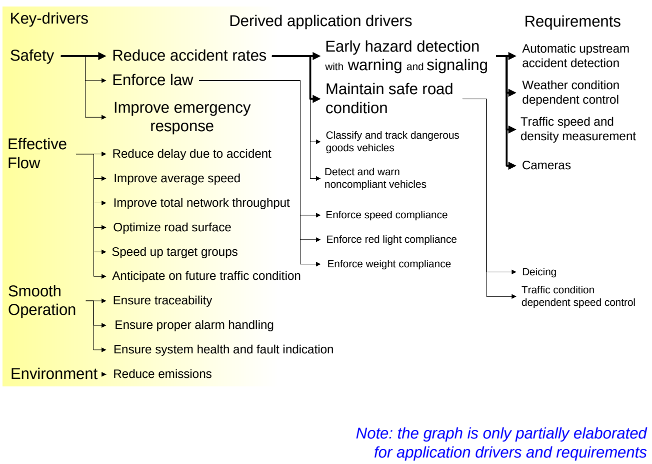
CAFCR+ Model



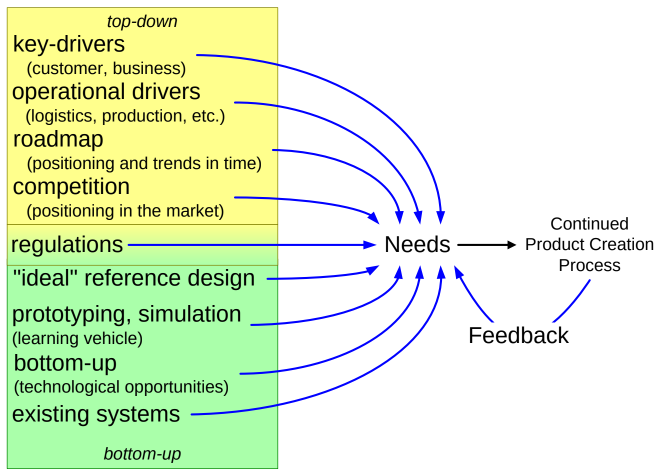
Iterate over Views



Example Key Driver Graph



Complementary Viewpoints



Module System Architect Toolkit

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

This module addresses tools and techniques available to the System Architect. It explains the basic CAFCR method and addresses story telling as method.

February 15, 2024
status: concept
version: 1.4

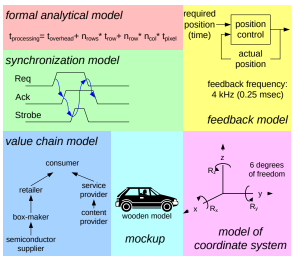
Basic Working Methods of a System Architect

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

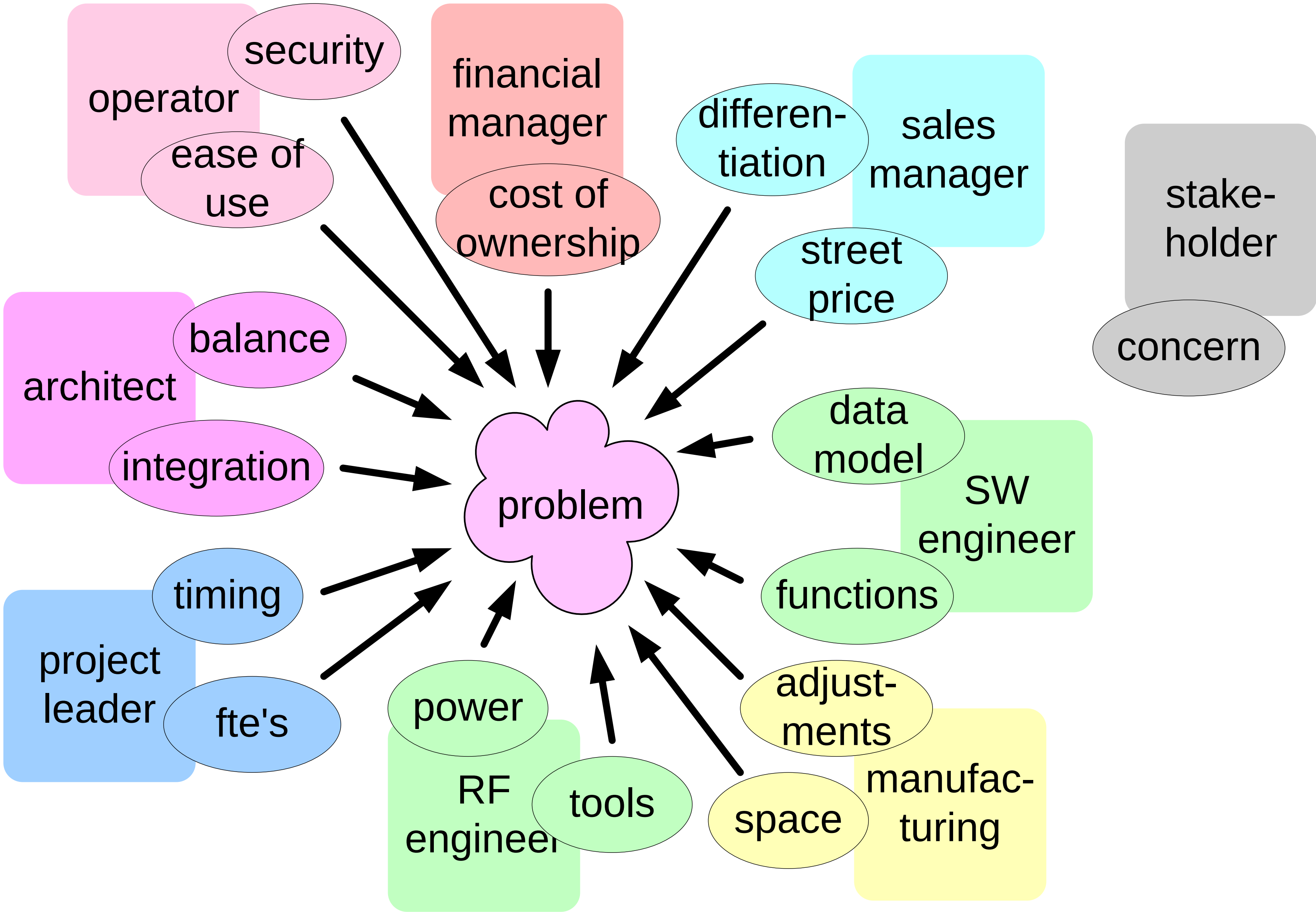
Abstract

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.
The basic working methods, such as viewpoint hopping, modelling, handling uncertainties and WWHWWW questions are described.

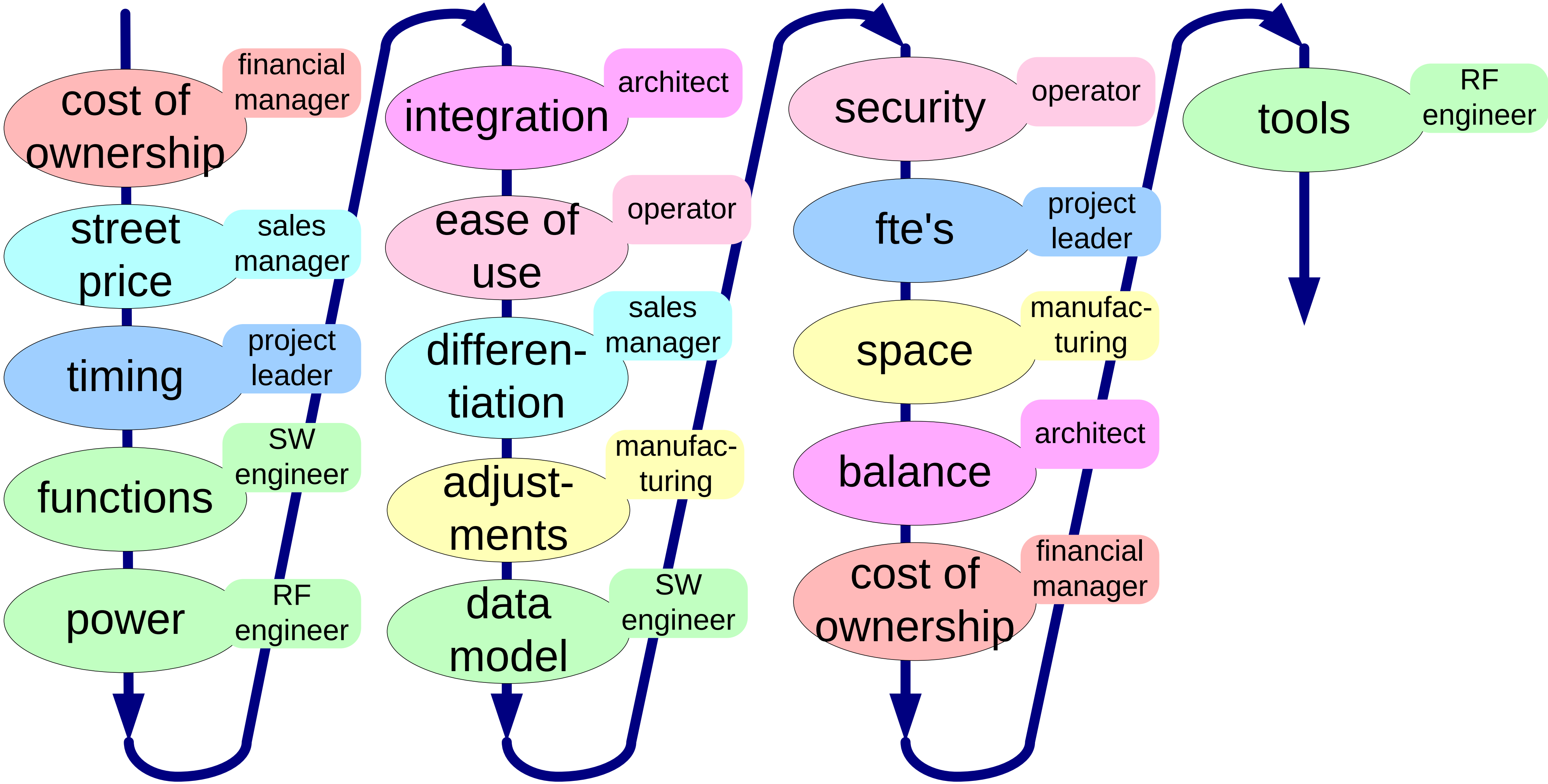
February 15, 2024
status: concept
version: 1.5



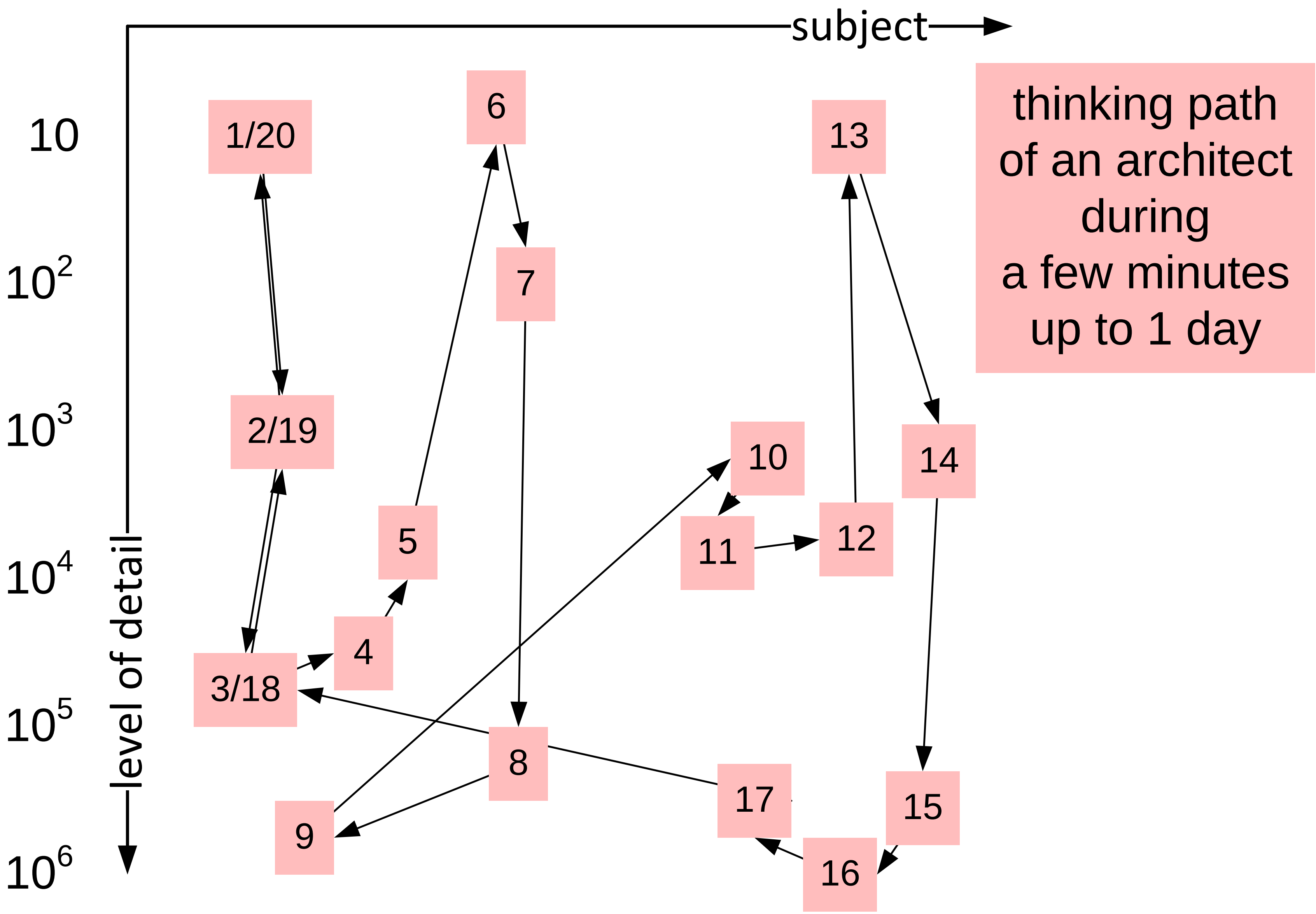
Many viewpoints



Viewpoint Hopping

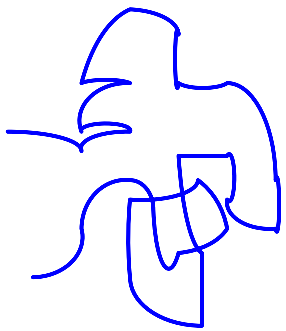


The seemingly random exploration path



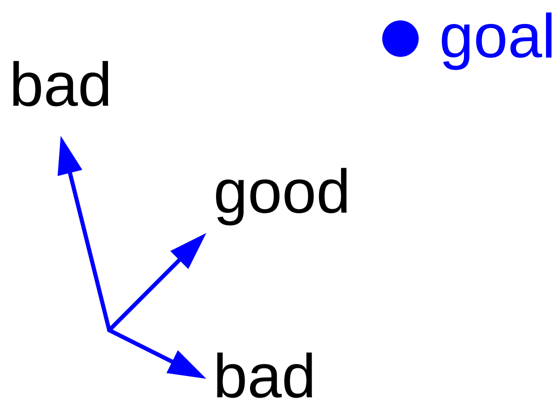
Scanning modes of the architect

open
perceptive
scanning



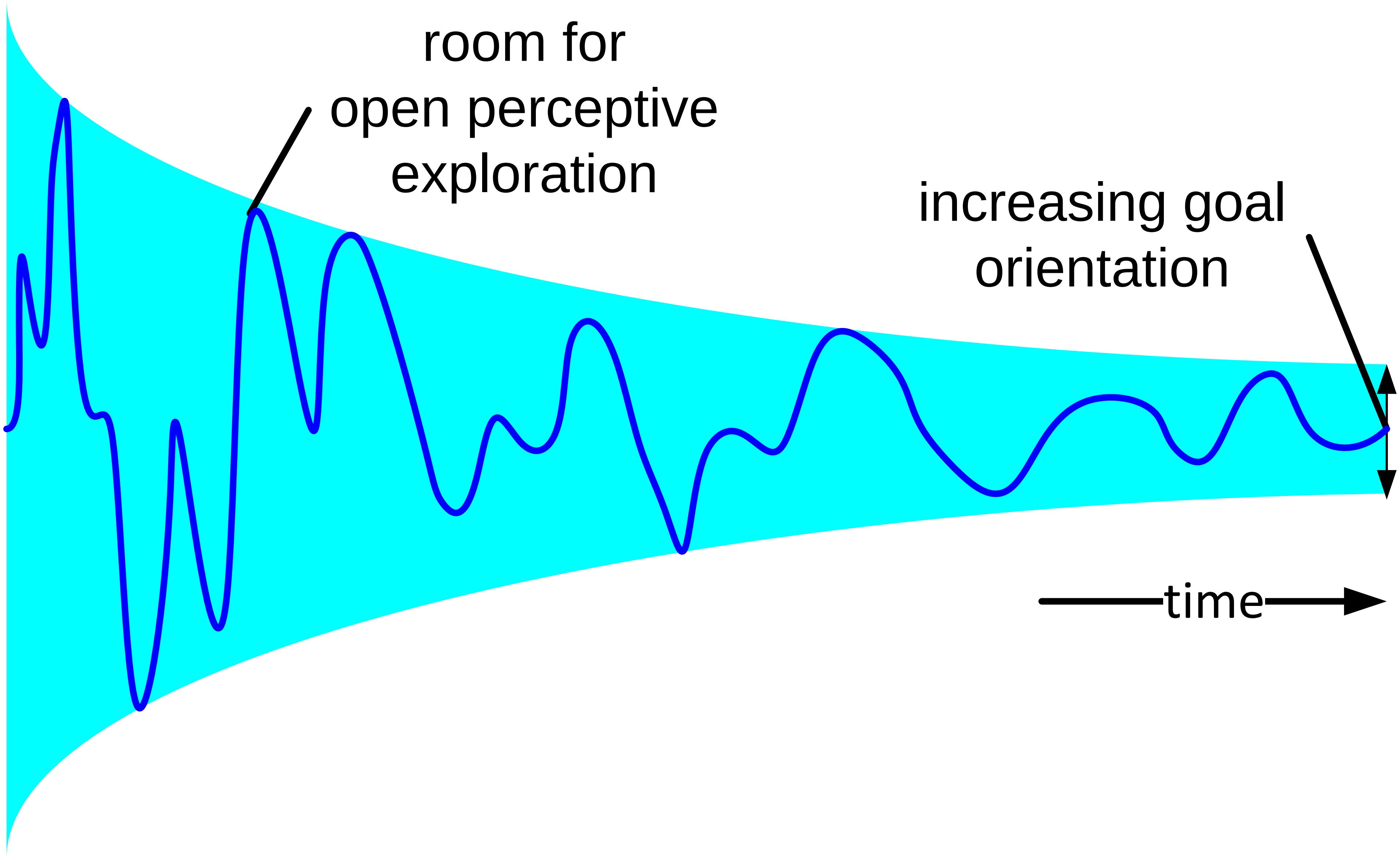
drunkard's walk
the world is full
of interesting
needs, technologies, ...

scanning
while
structuring
and judging

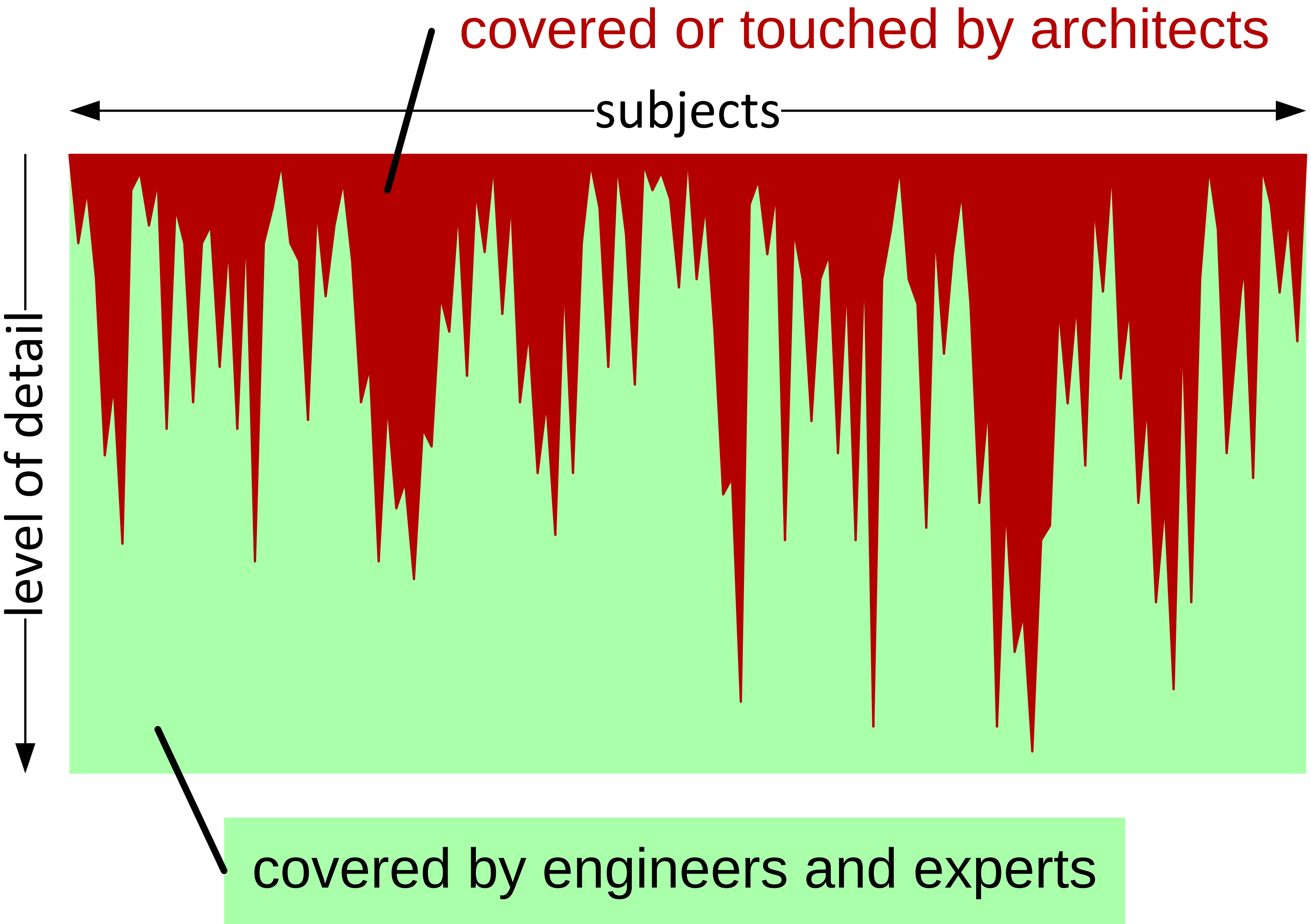


straight for the goal
ignore everything
that is not contributing
directly to the goal

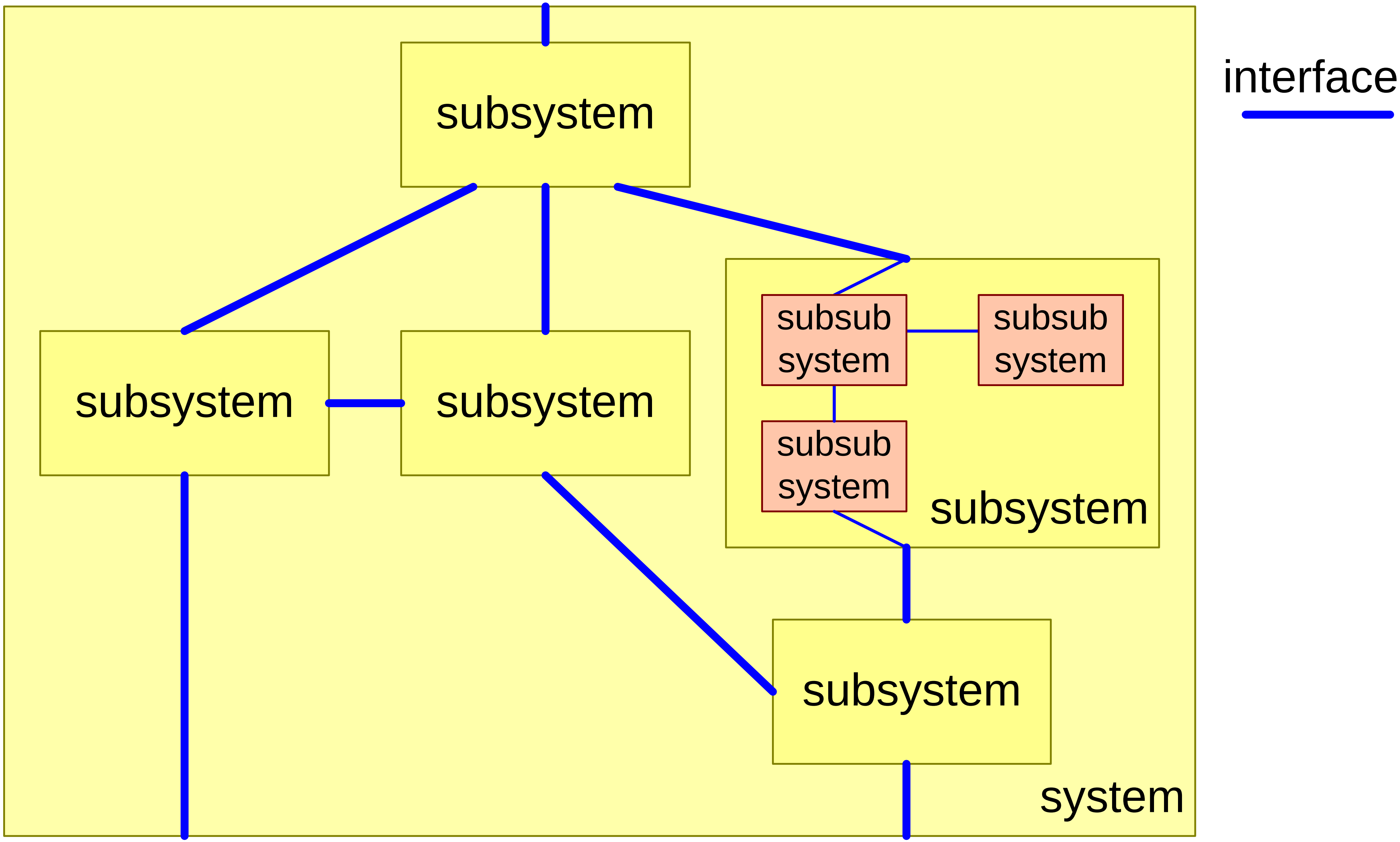
Combined open perceptive and goal oriented scanning



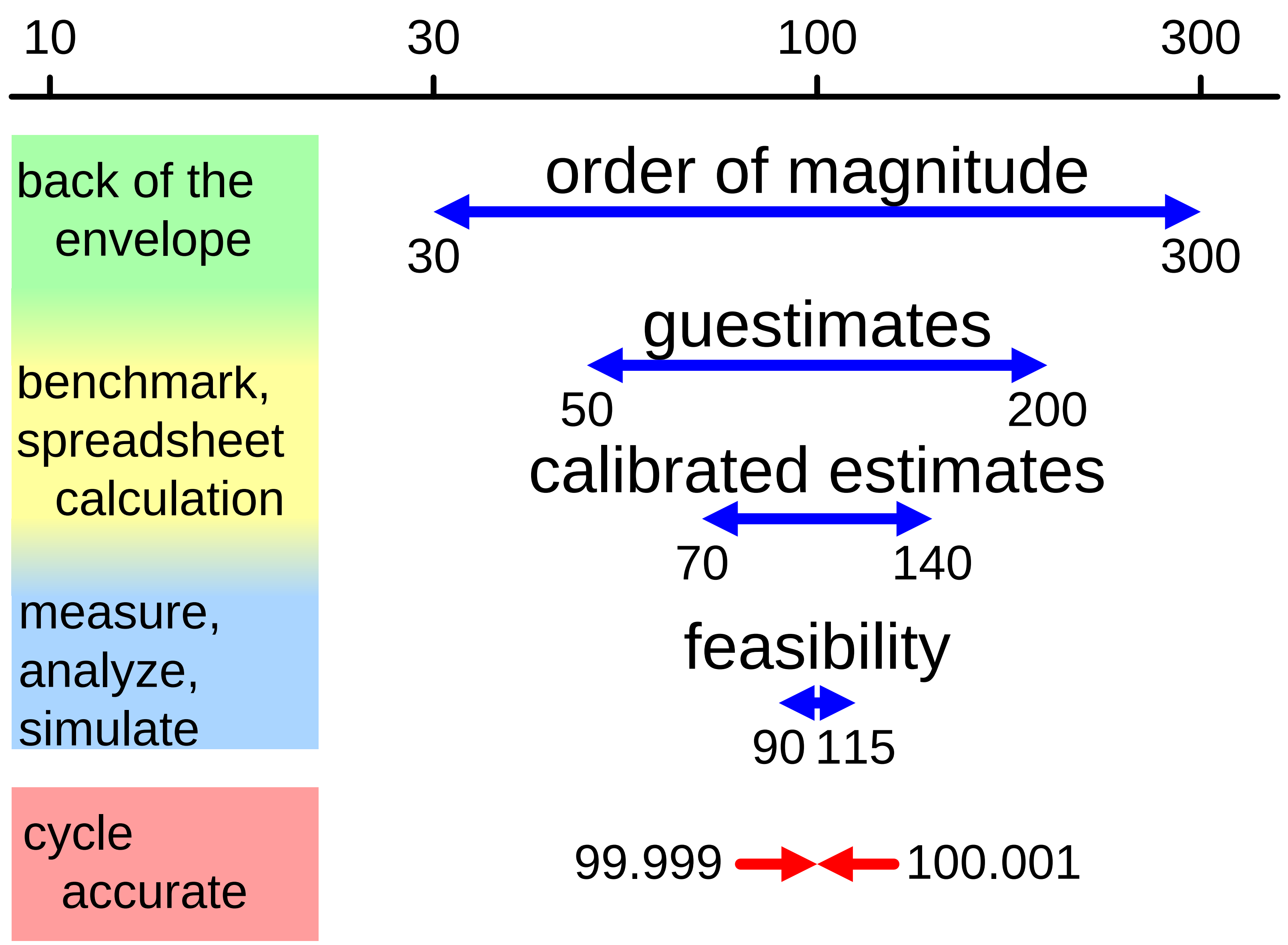
Coverage of problem and solution space



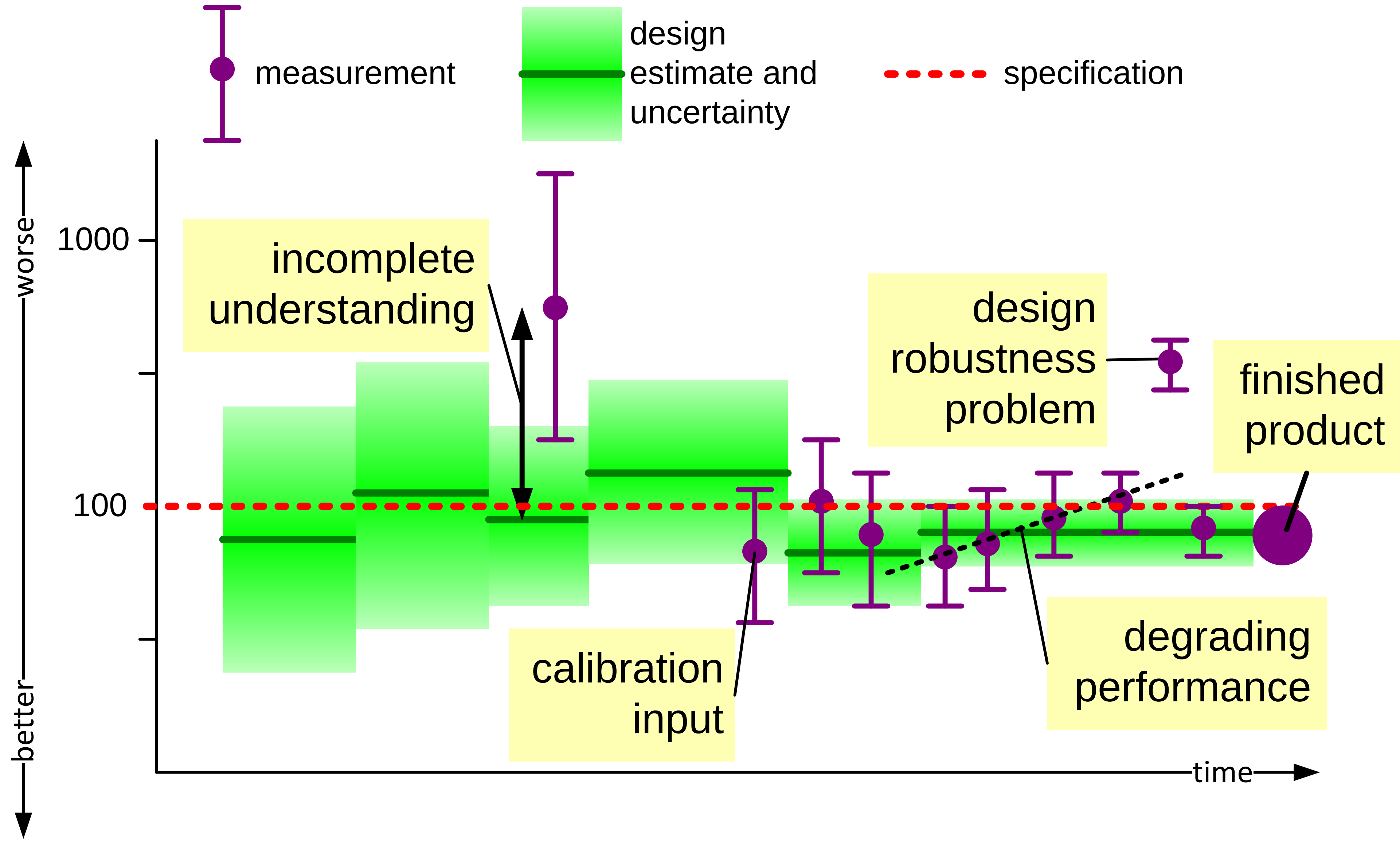
Decomposition, interfaces and integration



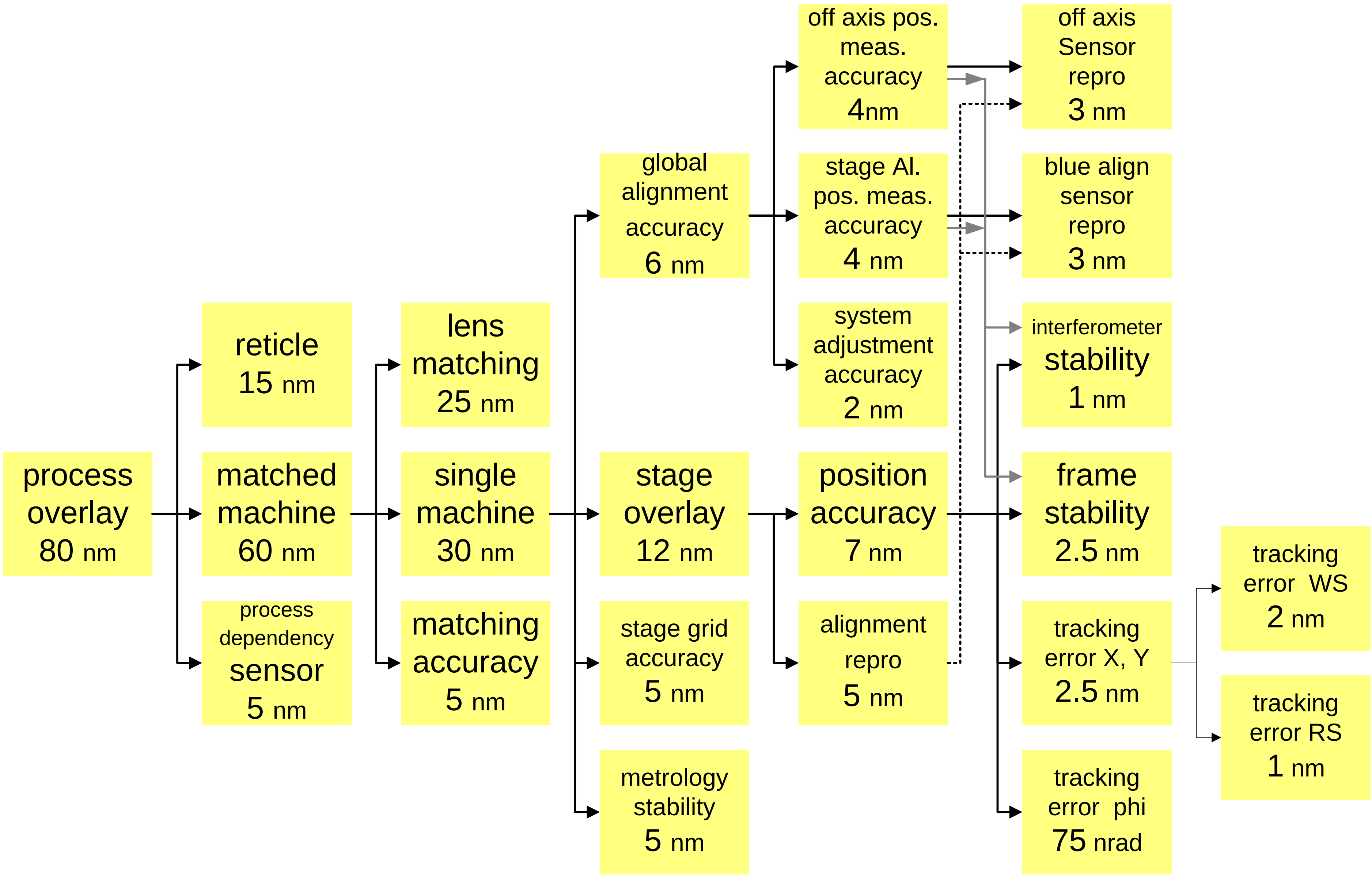
Successive quantification refinement



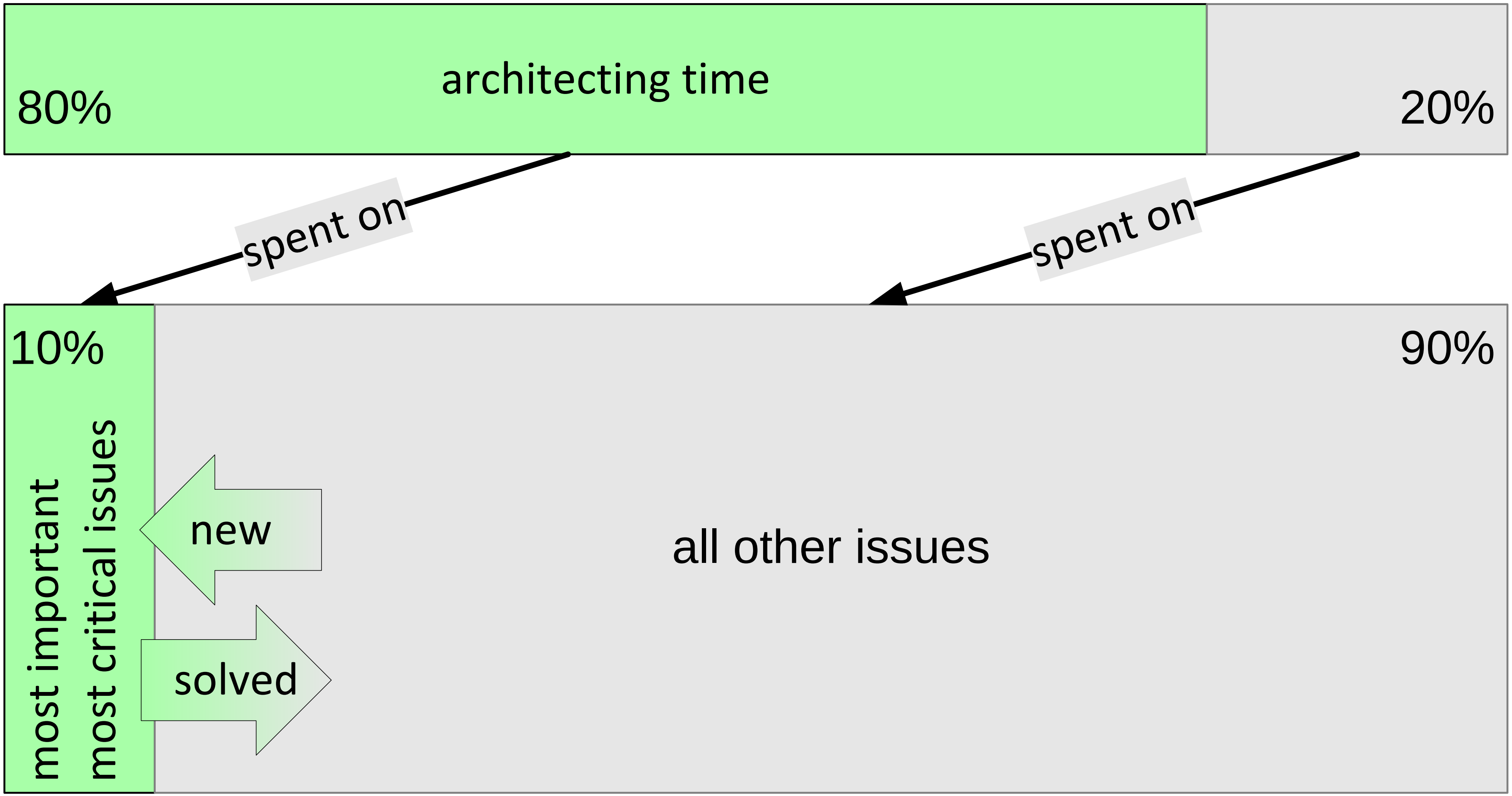
Example evolution of quantification



Quantified understanding of waferstepper overlay



Architect focus on important issues



Architect “worry” list

1. response time

from key press
until first image
on display

2. cost price

resource budgets

3. layering to separate

separation of concerns
self sustained
life-cycle separation
robust: paranoia validations

4. reliability of storage
5. database redesign
6. integration schedule
7. movement artefact
8. standby power
9. weak signal handling
10. location-based twiddle

A **model** is

a **simplified** representation of

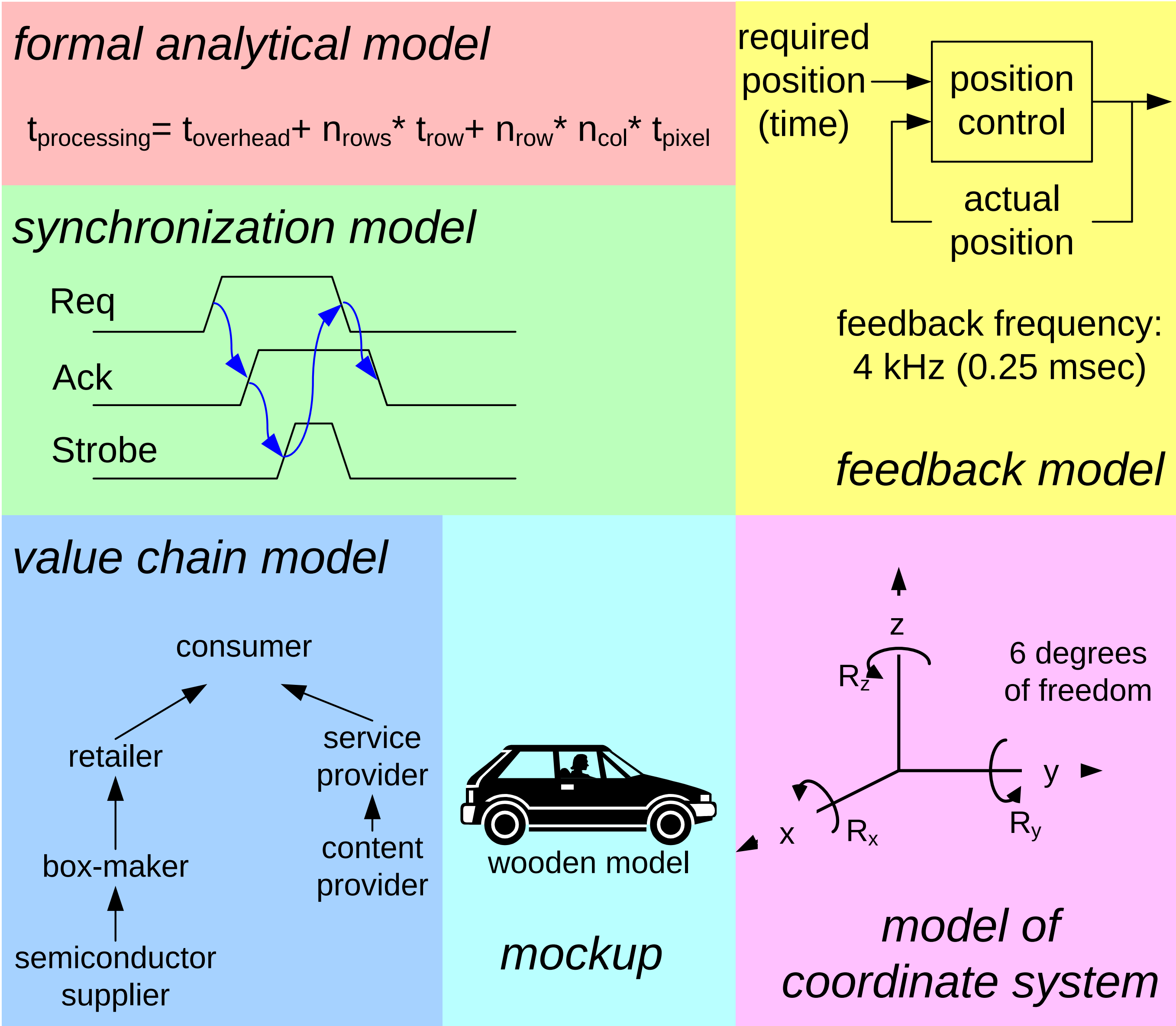
part of the **real world** used for:

communication, documentation

analysis, simulation,

decision making, verification

Some examples of models



Types of models

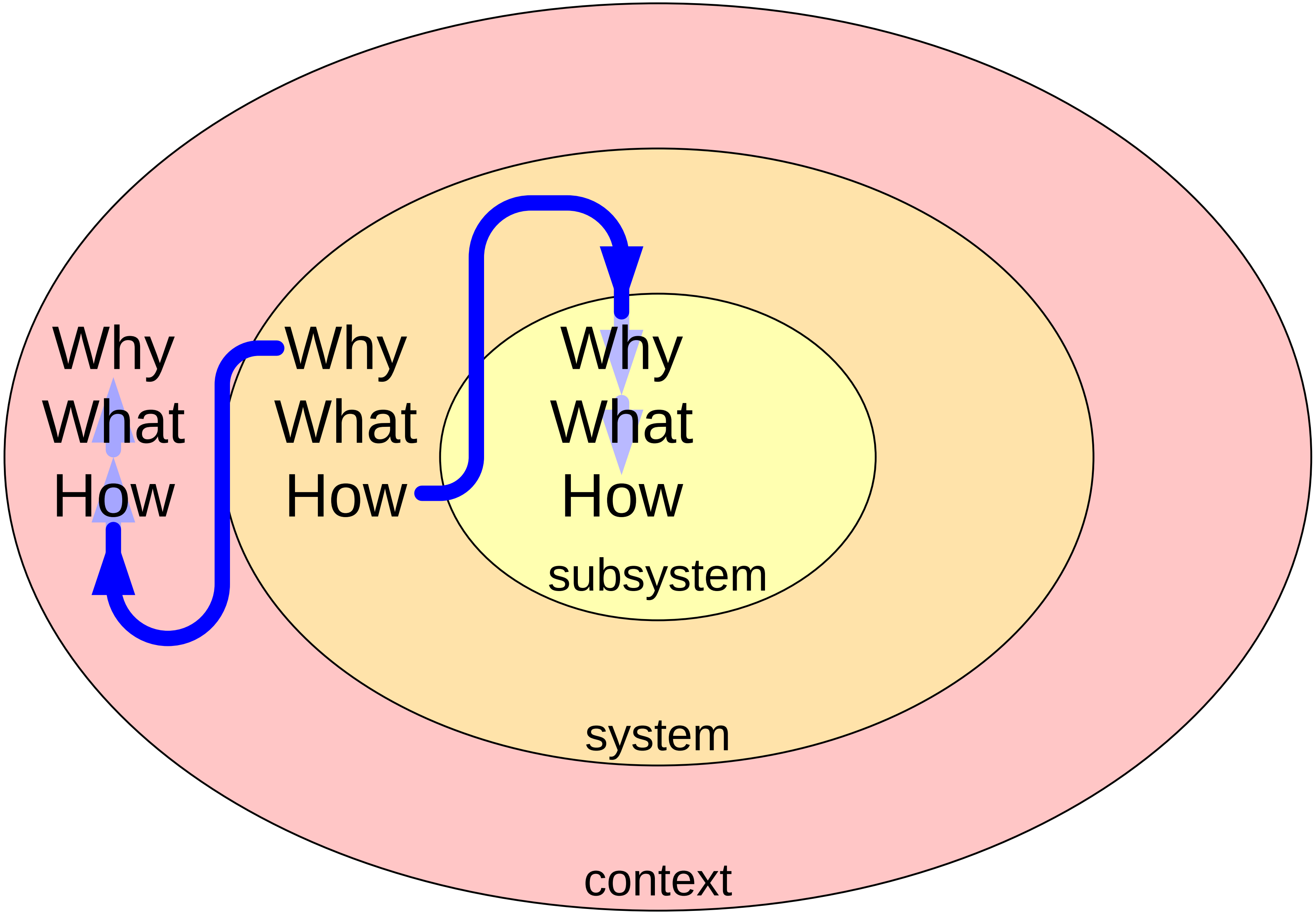
mathematical linguistic	visual
formal	informal
quantitative	qualitative
detailed	global
concrete	abstract
accurate	approximate
executable	read only

←rational—— ——intuitive→

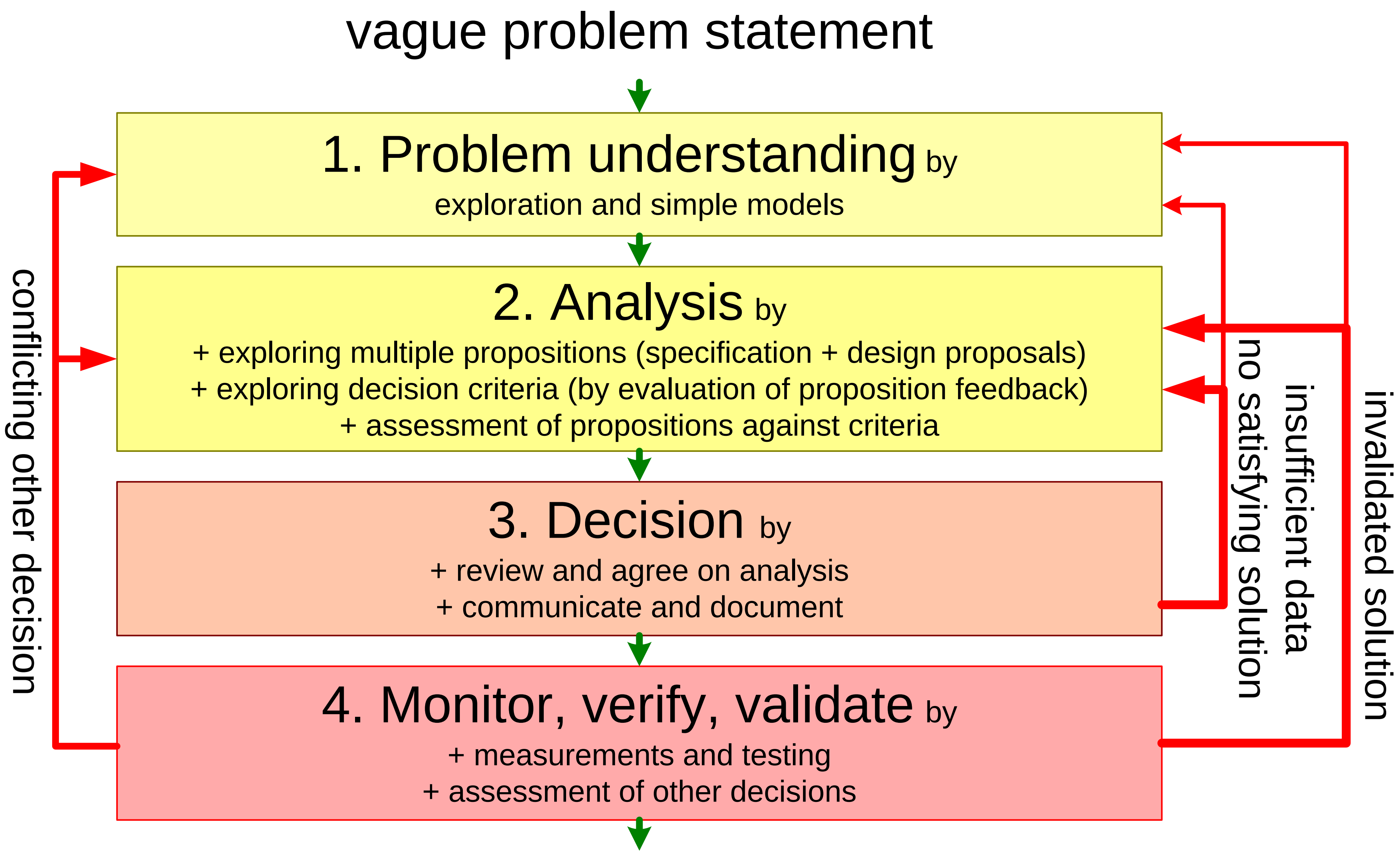
Questions

Why	Who
What	When
How	Where

Why broadens scope, How opens details



Flow from problem to solution



Multiple propositions

throughput	20 p/m	high-performance sensor	350 ns
cost	5 k$	high-speed moves	9 m/s
safety		additional pipelining	

low cost and performance 1

throughput	20 p/m	high-performance sensor	300 ns
cost	5 k$	high-speed moves	10 m/s
safety			

low cost and performance 2

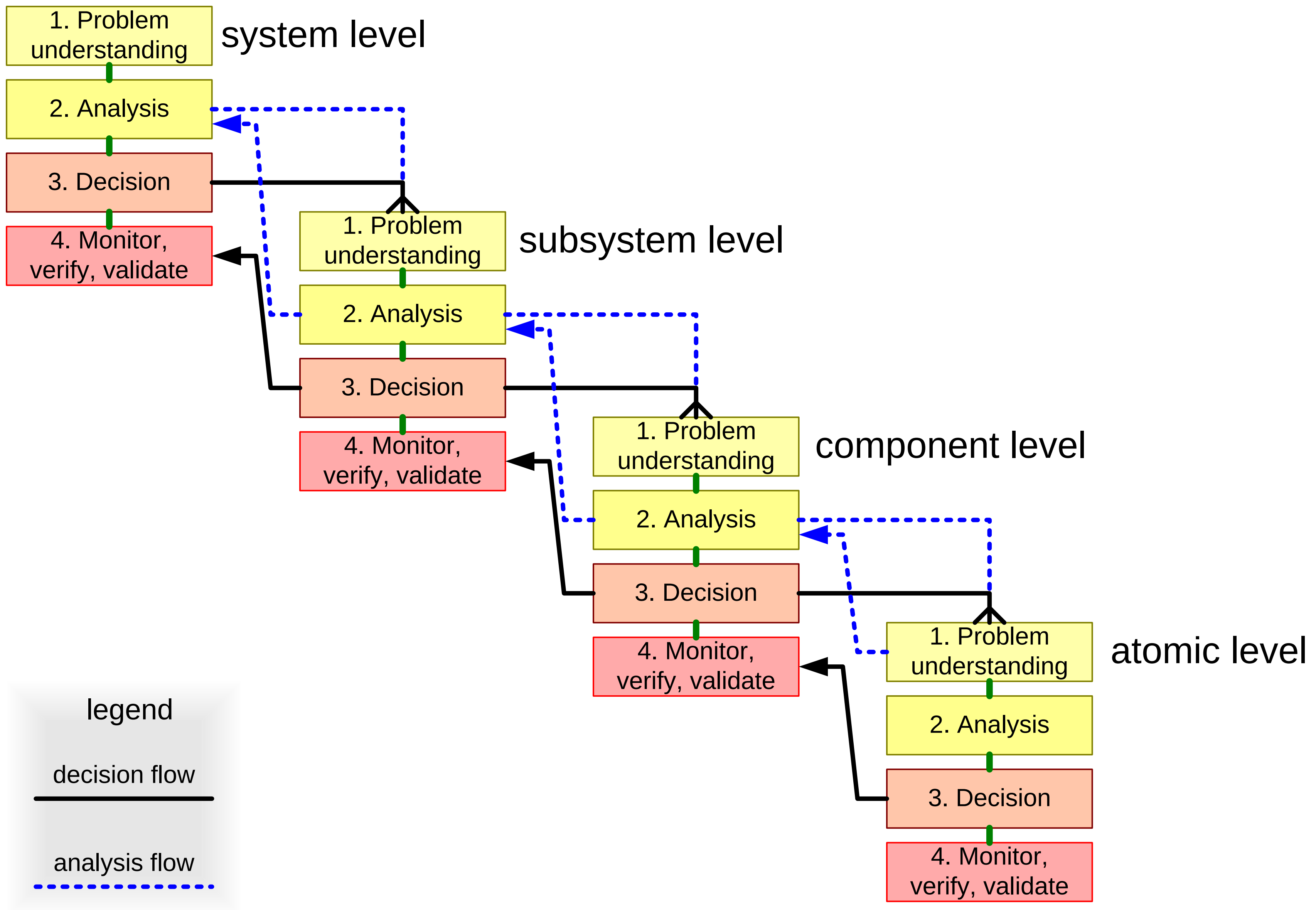
throughput	25 p/m	highperformance sensor	200 ns
cost	7 k$	high-speed moves	12 m/s
safety		additional collision detector	

high cost and performance

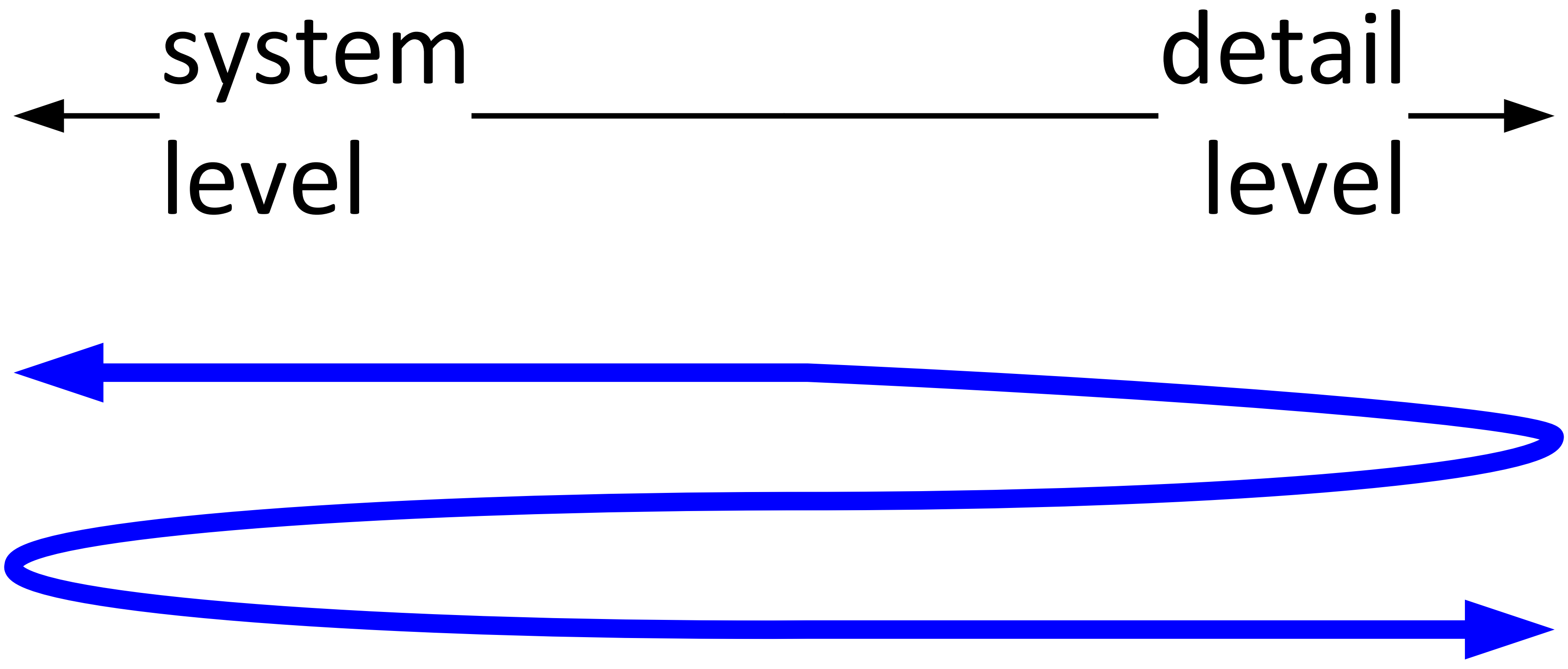
Assessment of propositions

criterions	*criterion weight*	*low cost and performance 1*	*low cost and performance 2*	*high cost and performance*
throughput	5	2	2	3
cost	5	3	3	2
safety	5	5	5	5
future proof	2	2	3	3
effort	4	5	4	4
dev. time	5	5	4	4
risk	4	4	3	3
maintenance	3	2	3	3

Recursive and concurrent application of flow



Exploration by rapid iteration



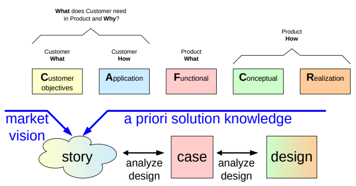
Story How To

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

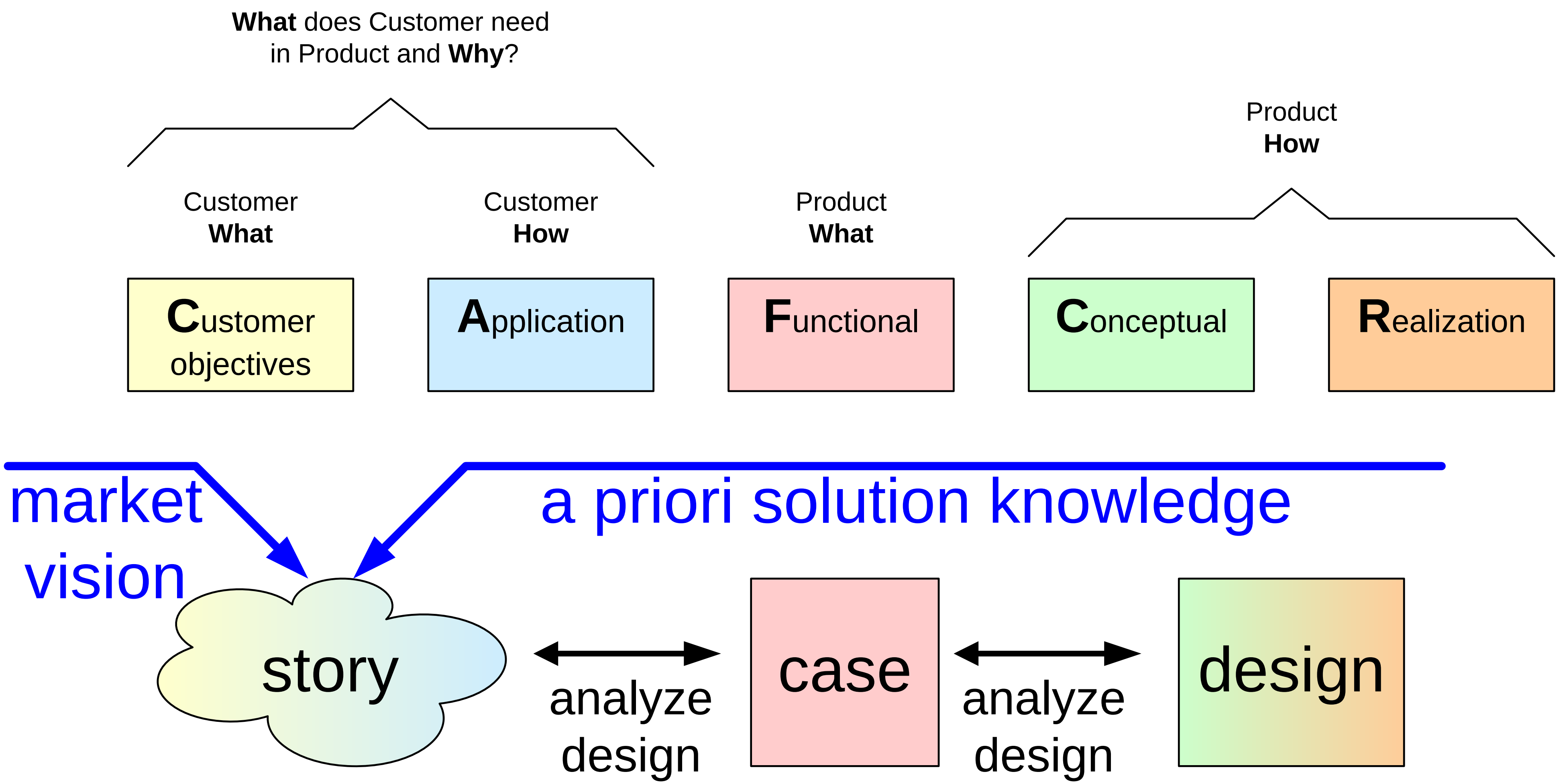
Abstract

A story is an easily accessible story or narrative to make an application live. A good story is highly specific and articulated entirely in the problem domain: the native world of the users. An important function of a story is to enable specific (*quantified, relevant, explicit*) discussions.

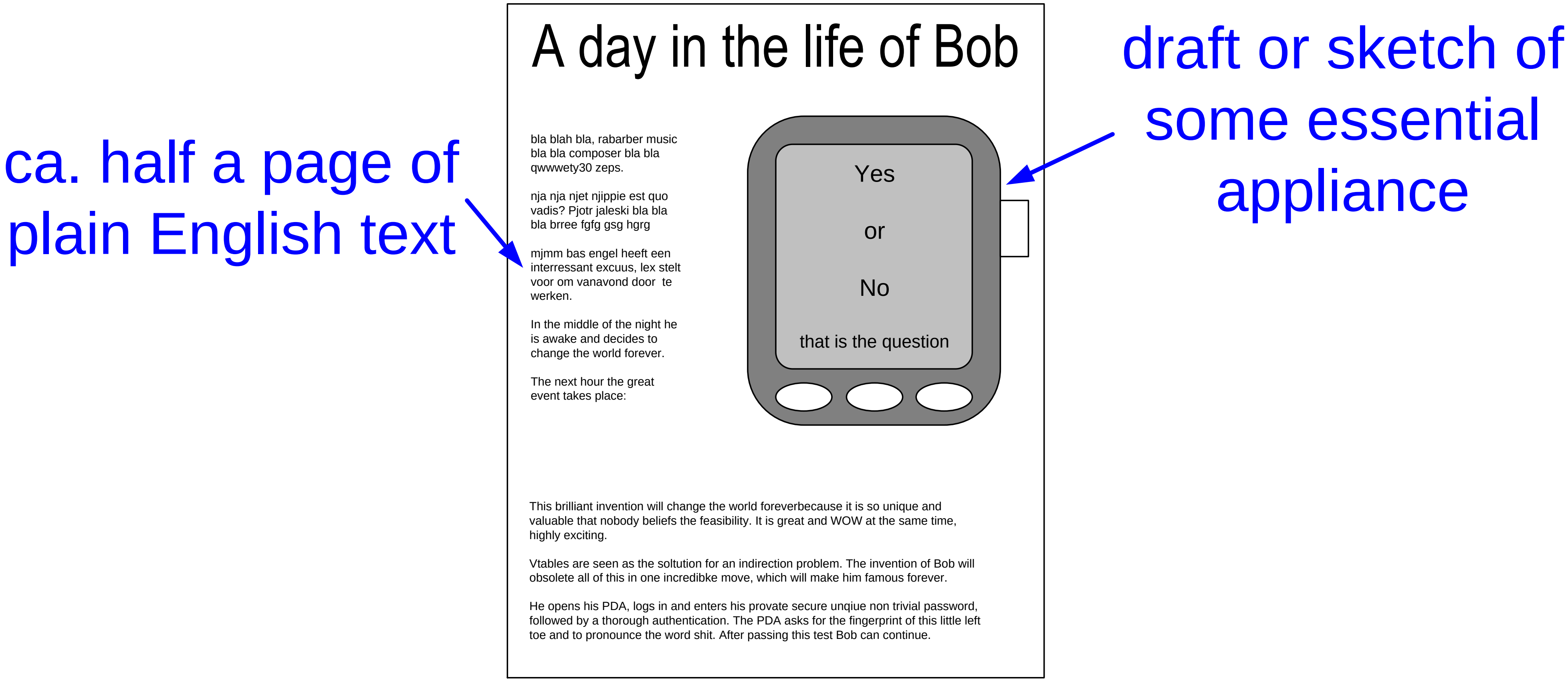
February 15, 2024
status: concept
version: 1.2



From story to design



Example story layout



Points of attention

- purpose — What do you need to know for specification and design?
- scope — "umbrella" or specific event?
- viewpoint, stakeholders — Define your stakeholder and viewpoint f.i. user, maintainer, installer
- visualization — Sketches or cartoon. Helps to share and communicate ideas
- size (max 1 A4) — Can be read or told in few minutes
- recursive decomposition, refinement

Criteria for a good story

Customer objectives / **A**pplication	• accessible, understandable "Do you see it in front of you?"
Customer objectives / **A**pplication	• valuable, appealing attractive, important "Are customers queuing up for this?"
Conceptual / **R**ealization	• critical, challenging "What is difficult in the realization?" "What do you learn w.r.t. the design?"
Application	• frequent, no exceptional niche "Does it add significantly to the bottom line?"
Application / **F**unctional	• specific names, ages, amounts, durations, titles, ...

Example of a story

Betty is a 70-year-old woman who lives in Eindhoven. Three years ago her husband passed away and since then she lives in a home for the elderly. Her 2 children, Angela and Robert, come and visit her every weekend, often with Betty's grandchildren Ashley and Christopher. As so many women of her age, Betty is reluctant to touch anything that has a technical appearance. She knows how to operate her television, but a VCR or even a DVD player is way to complex.

When Betty turned 60, she stopped working in a sewing studio. Her work in this noisy environment made her hard-of-hearing with a hearing-loss of 70dB around 2kHz. The rest of the frequency spectrum shows a loss of about 45dB. This is why she had problems understanding her grandchildren and why her children urged her to apply for hearing aids two years ago. Her technophobia (and her first hints or arthritis) inhibit her to change her hearing aids' batteries. Fortunately her children can do this every weekend.

This Wednesday Betty visits the weekly Bingo afternoon in the meetingplace of the old-folk's home. It's summer now and the tables are outside. With all those people there it's a lot of chatter and babble. Two years ago Betty would never go to the bingo: "I cannot hear a thing when everyone babbles and clatters with the coffee cups. How can I hear the winning numbers?!". Now that she has her new digital hearing instruments, even in the bingo cacophony, she can understand everyone she looks at. Her social life has improved a lot and she even won the bingo a few times.

That same night, together with her friend Janet, she attends Mozart's opera The Magic Flute. Two years earlier this would have been one big low rumbly mess, but now she even hears the sparkling high piccolos. Her other friend Carol never joins their visits to the theaters. Carol also has hearing aids, however hers only "work well" in normal conversations. "When I hear music it's as if a butcher's knife cuts through my head. It's way too sharp!". So Carol prefers to take her hearing aids out, missing most of the fun. Betty is so happy that her hearing instruments simply know where they are and adapt to their environment.

source: Roland Mathijssen
Embedded Systems Institute
Eindhoven

Value and Challenges in this story

Customer objectives

Application

Value proposition in this story:

quality of life:

- active participation in different social settings

usability for nontechnical elderly people:

- "intelligent" system is simple to use
- loading of batteries

Conceptual

Realization

Challenges in this story:

Intelligent hearing instrument

Battery life — at least 1 week

No buttons or other fancy user interface on the hearing instrument, other than a robust On/Off method

The user does not want a technical device but a solution for a problem

Instrument can be adapted to the hearing loss of the user

Directional sensitivity (to prevent the so-called cocktail party effect)

Recognition of sound environments and automatic adaptation (adaptive filtering)

source: Roland Mathijssen, Embedded Systems Institute, Eindhoven

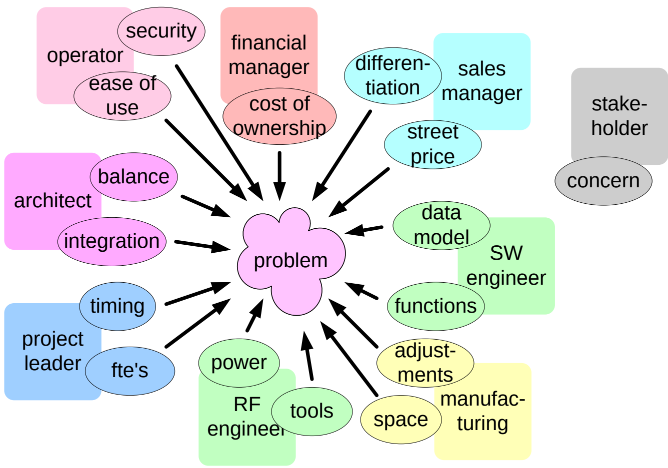
Personal multi media appliance

Create a story for a personal multi media appliance.

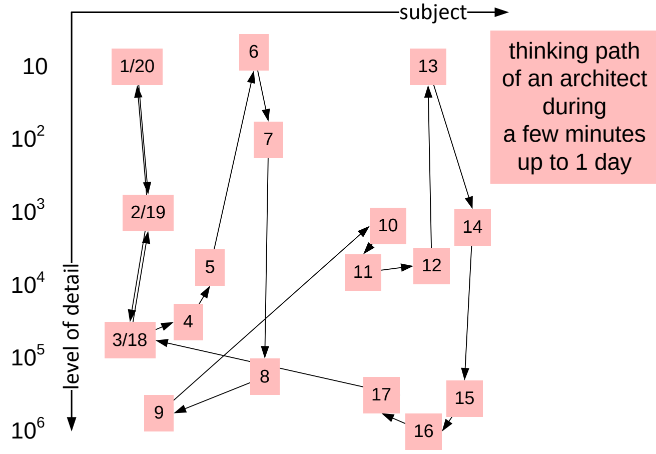
Derive a case description from the story, with functions and quantitative requirements.

Architect Way of Working

Viewpoint Hopping



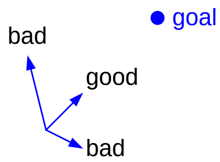
Chaotic Path



Perceptive vs Judging

open
perceptive
scanning

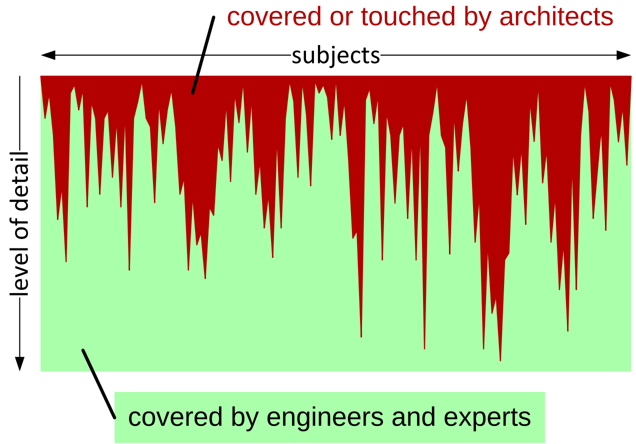
scanning
while
structuring
and judging



drunkard's walk
the world is full
of interesting
needs, technologies, ...

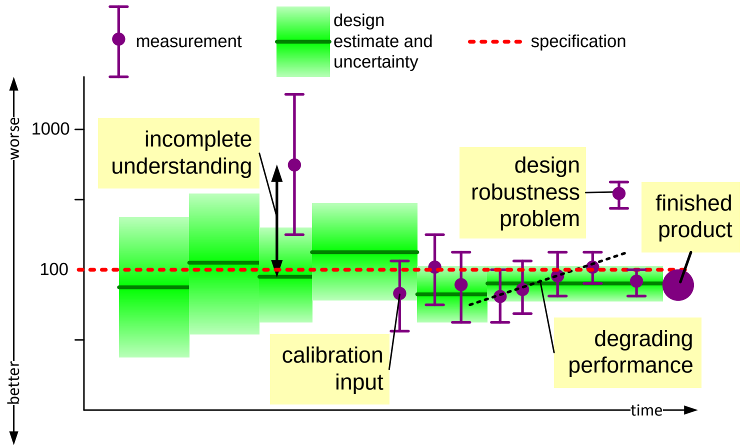
straight for the goal
ignore everything
that is not contributing
directly to the goal

Varying Depth

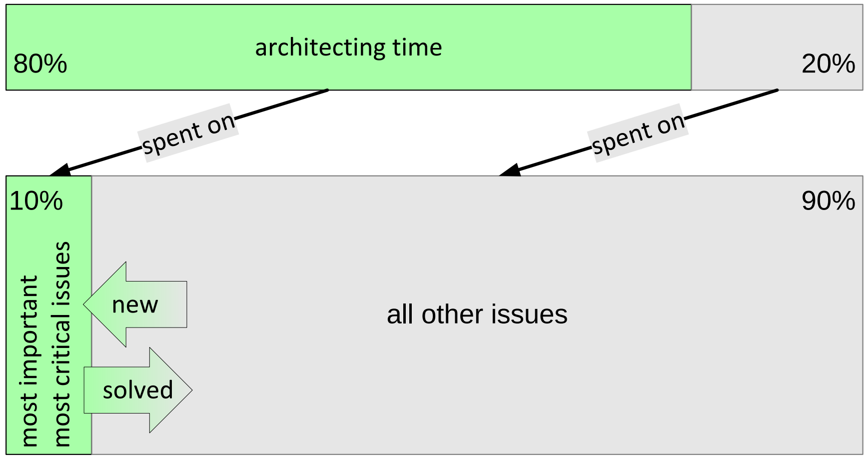


Some Architecting Means

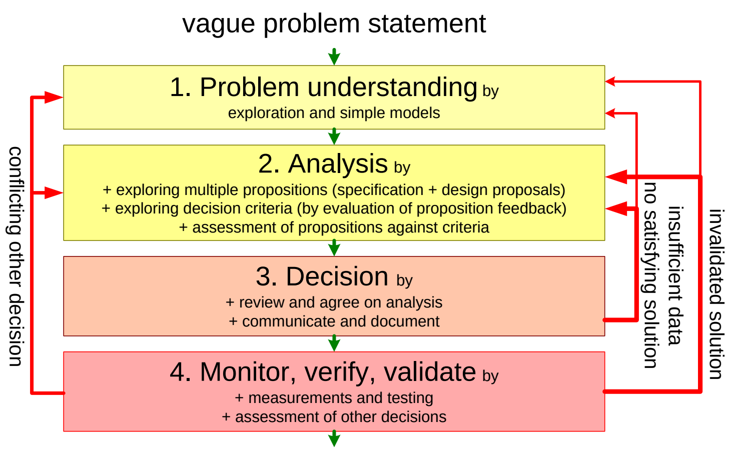
Quantification and Margins



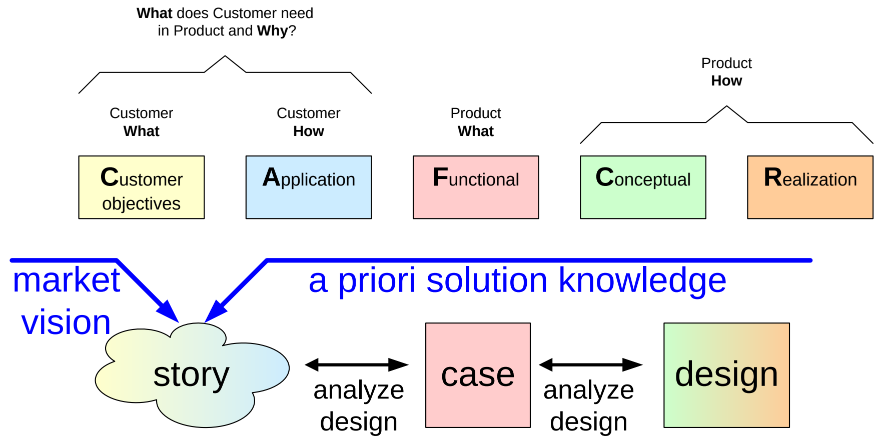
Focus on Key Issues



Phased Problem Solving



Story Telling



Module Roadmapping

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

This module addresses roadmapping.

Distribution

February 15, 2024
status: draft
version: 1.2

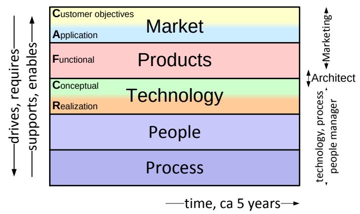
Roadmapping

by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

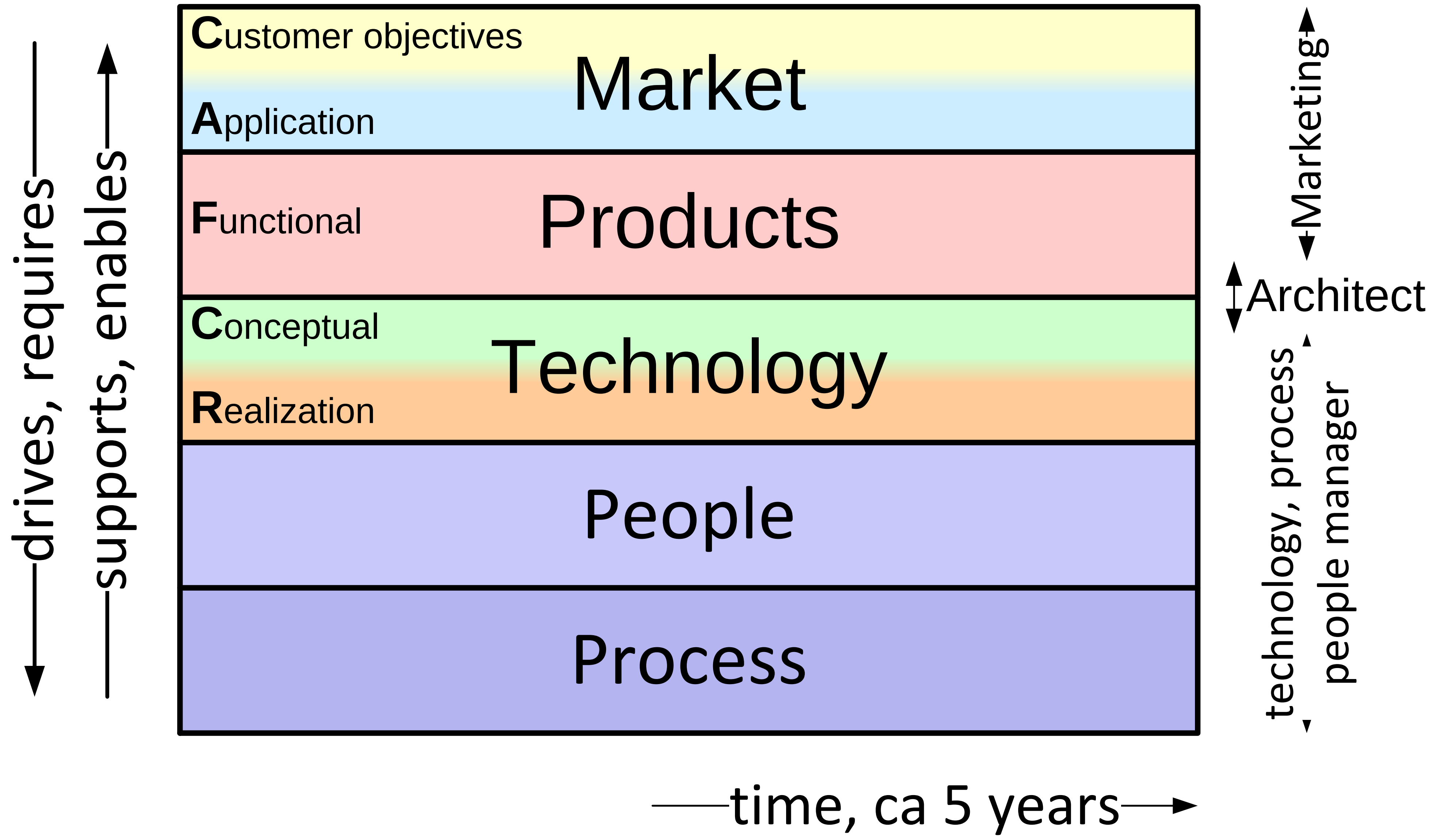
Abstract

This article describes what a roadmap is, how to create and maintain a roadmap, the involvement of the stakeholders, and criteria for the structure of a roadmap.

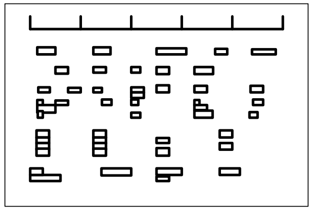
February 15, 2024
status: concept
version: 2.0



The Roadmap Integrates Five Views



Granularity of Roadmap Material

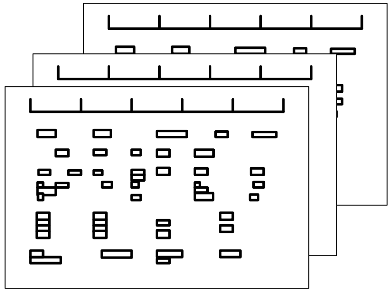


Top-level roadmap

Single page

Poster

part of many presentations

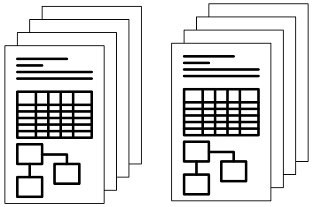


Supporting roadmaps

Single page
per view
or per driver

Poster

part of many presentations



Supporting reports

Document
per relevant
subject

Problems that Occur without Roadmapping

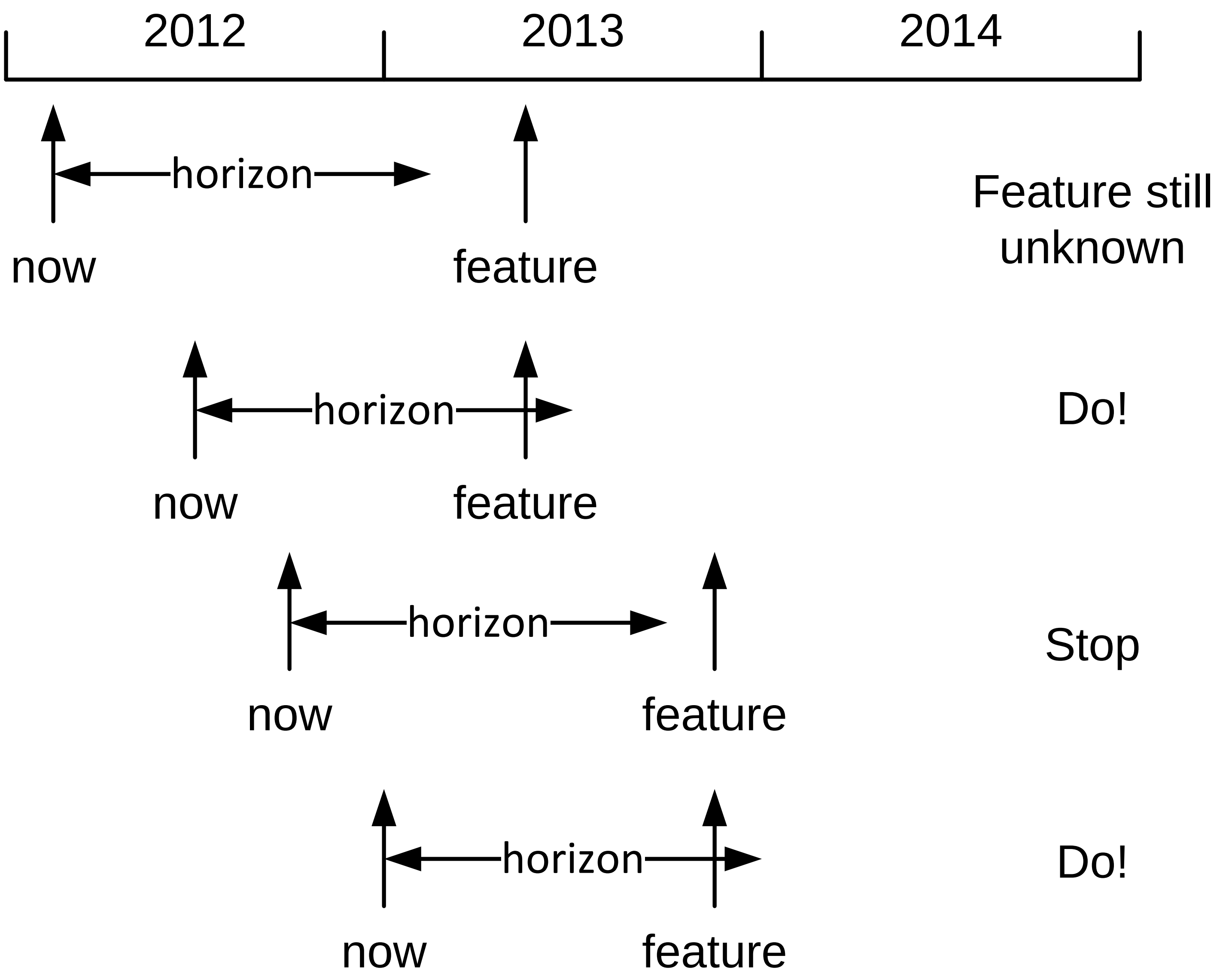
Frequent changes in product policy

Late start up of long lead activities, such as people recruitment and process change

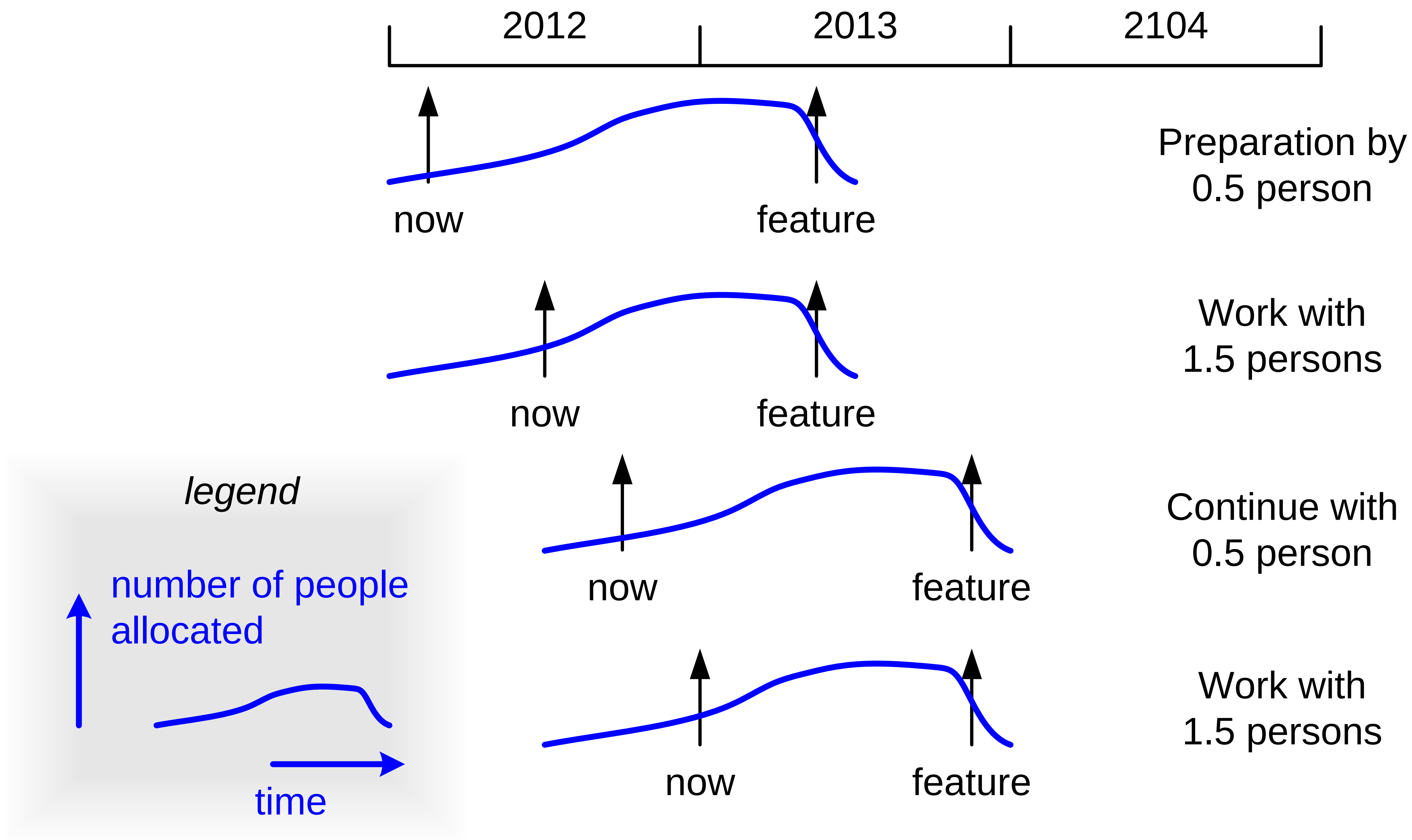
Diverging activities of teams

Missed market opportunities

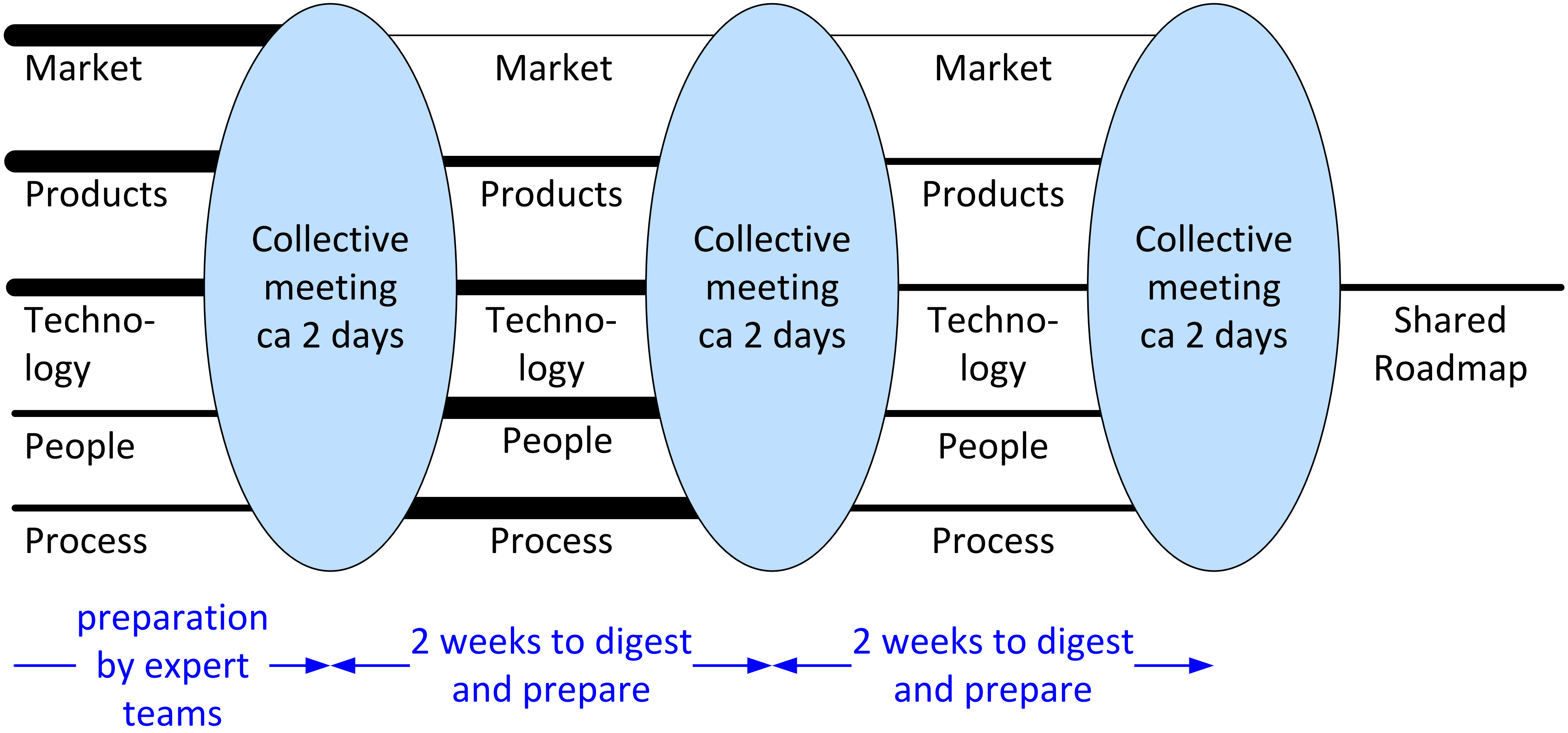
Management with a Limited Horizon



Management with a Broader Time Perspective



Creation or Update of Roadmap in Burst Mode



Typical Stakeholders of a Roadmap

business manager overall enterprise responsible

marketing manager(s)

discipline or line managers

people, process, and technology manager(s)

operational manager(s) project or program managers

architect(s)

Target of the First Session

Shared vision on market

First iteration of possible products as an answer to the market

Share technology status, as starting point for technology roadmap

Explore people and technology status, to identify main issues

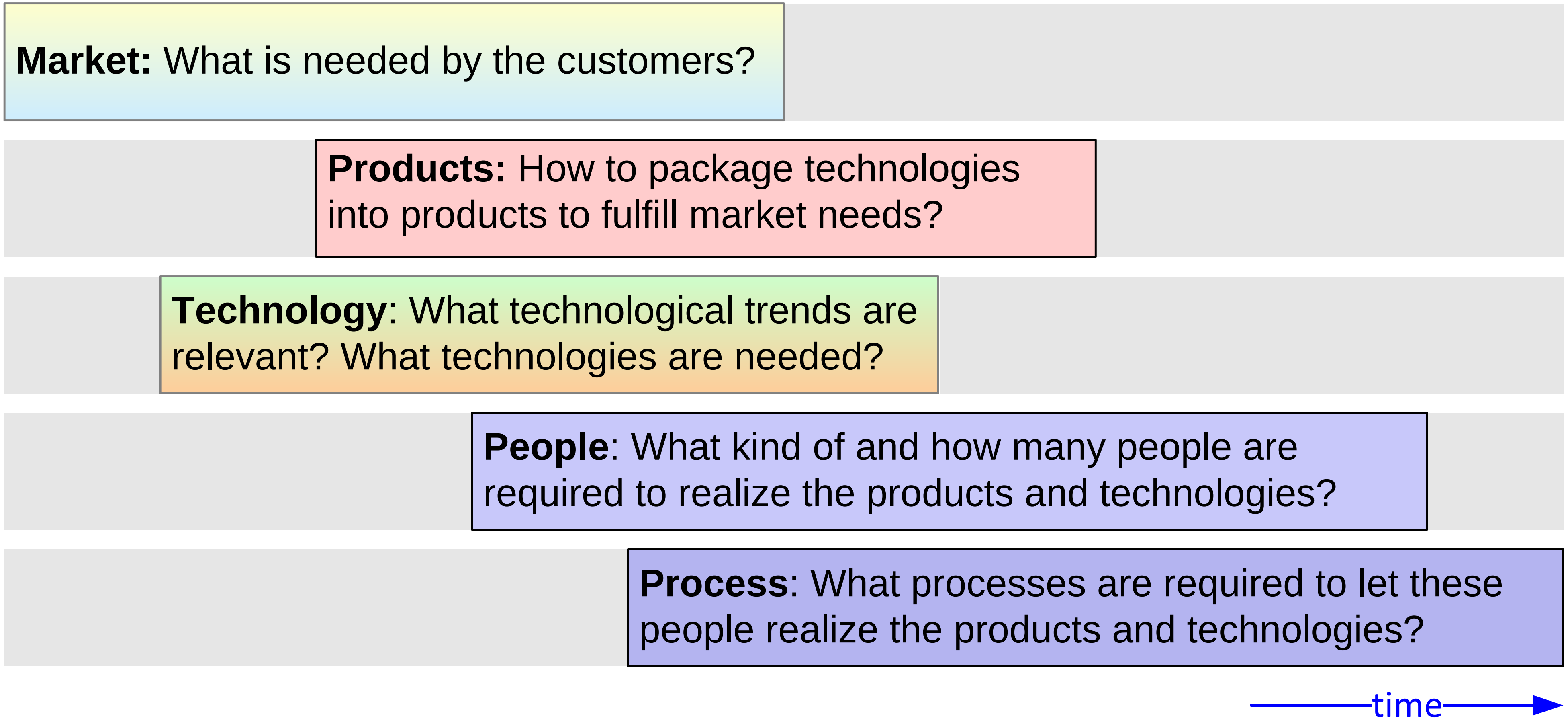
Target of the Second Session

Obtaining a shared vision on the desired technology roadmap

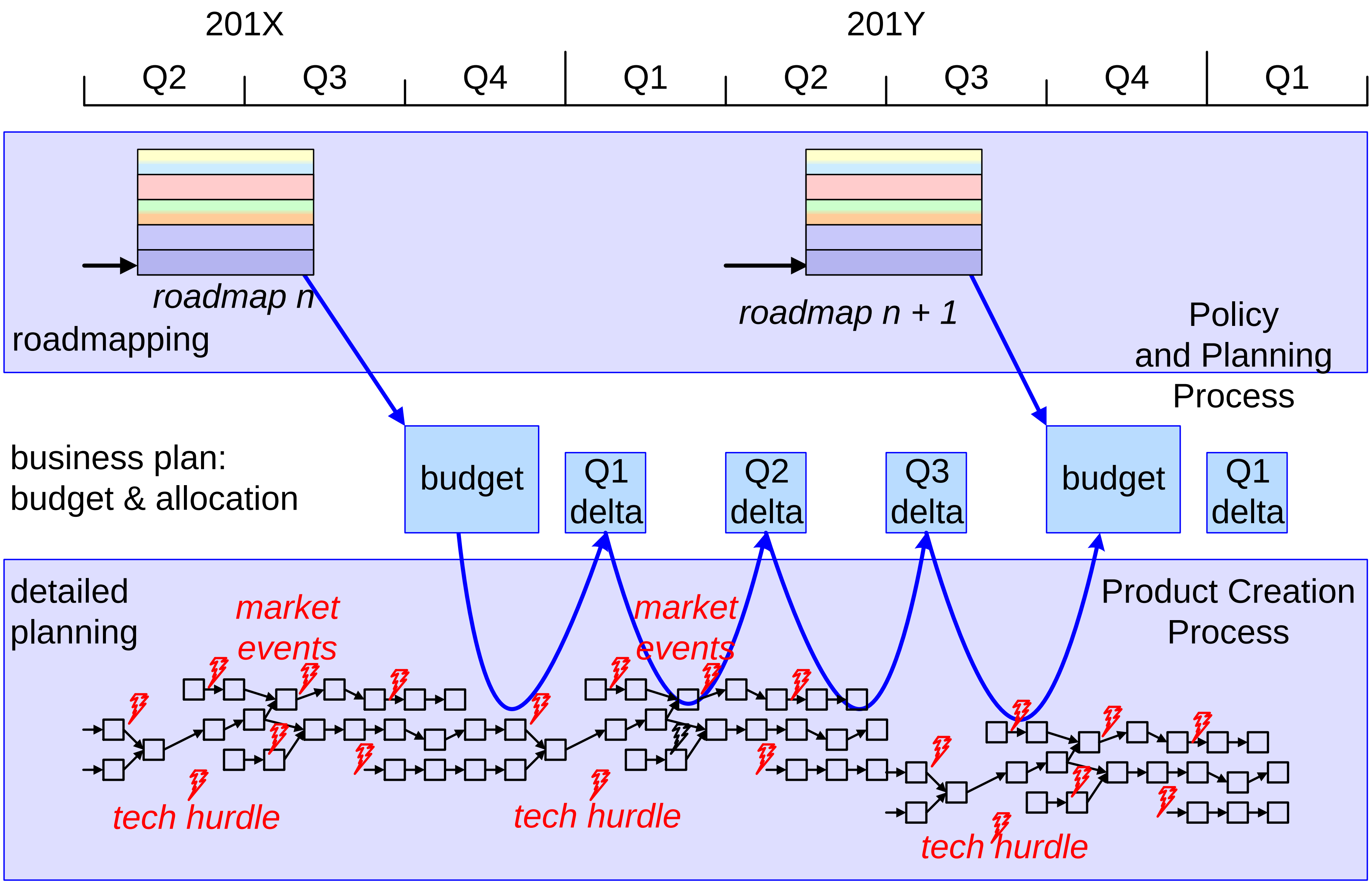
Sharing the people and process issues required for the products defined in the first iteration

Analyzing a few scenarios for products, technologies, people, and process

The Roadmap Update Visualized in Time



From Roadmap to Detailed Plans



3-Tier Approach

	horizon	*update*	*scope*	*type*
roadmap	5 years	1 year	portfolio	vision
budget	1 year	3 months	program	commitment
detailed plan	1 mnth-1yr	1 day-1 mnth	program or activity	control means

Roadmap Essentials

Selection of most important or relevant issues

Key drivers as a means to structure the roadmap

Nothing is certain; ambiguity is normal

Use facts whenever possible

Don’t panic in case of impossibilities

Requirements for a Good Roadmap

Recognizable issues for all stakeholders

Clear positioning in time; uncertainty can be visualized

The main events (enabling or constraining) must be present

Limited amount of information to maintain the overview

Sources of Facts

Market analysis reports

number of customers, market size, competition, trends

Installed base

change requests, problem reports, historical data

Manufacturing (statistical process control)

statistical process control

Suppliers (roadmaps, historical data)

roadmaps, historical data

Internal reports (technology studies, simulations)

technology studies, simulations

Causes for Overestimation

Quantization effects of small activities (the amount of time is rounded to manweeks/months/years)

Uncertainty is translated into margins at every level (module, subsystem, system)

Counting activities twice (e.g., in technology development and in product development)

Quantization effects of persons/roles (full time project leader, architect, product manager, et cetera per product)

Lack of pragmatism (technical ambition is not too bad during the roadmap process, as long as it does not pre-empt a healthy decision)

Too many bells and whistles without business or customer value

Market Product Life Cycle Consequences for Architecting

by *Gerrit Muller* University of South-Eastern Norway-NISE

e-mail: gaudisite@gmail.com

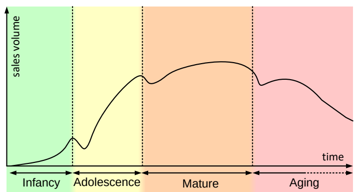
www.gaudisite.nl

Abstract

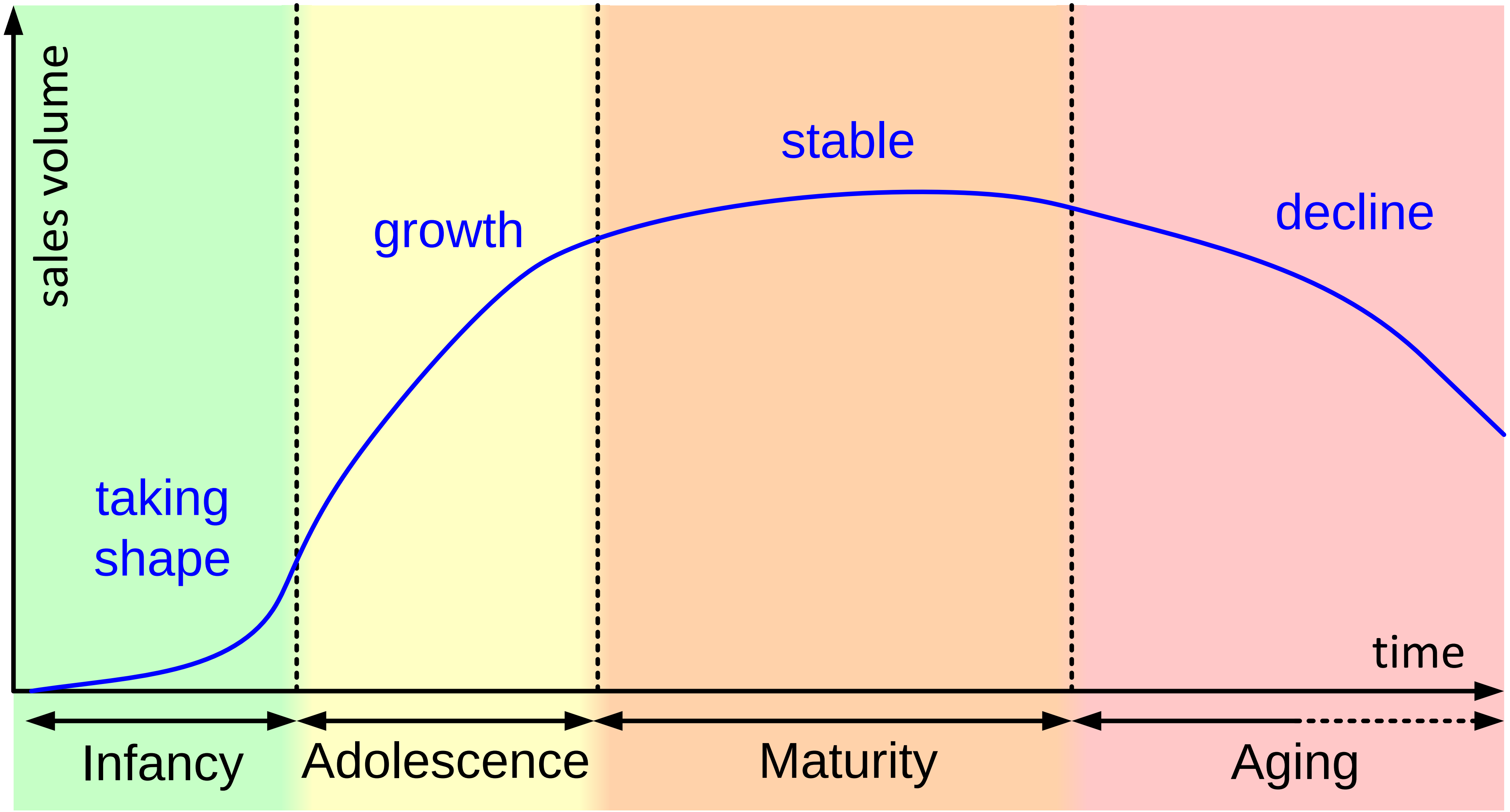
The lifecycle of a product category in the market determines many aspects of the architecting approach. The lifecycle consists typical of 4 phases: infancy, adolesence, mature and aging.

A discontinuity in market success is seen in the transition from one phase to the next phase. The explanation given is that the phases differ in characterictics and require different approaches. The right approach for one phase is sub optimal for the next phase. A set of characteristics per phase is given and the consequences for architecting are discussed.

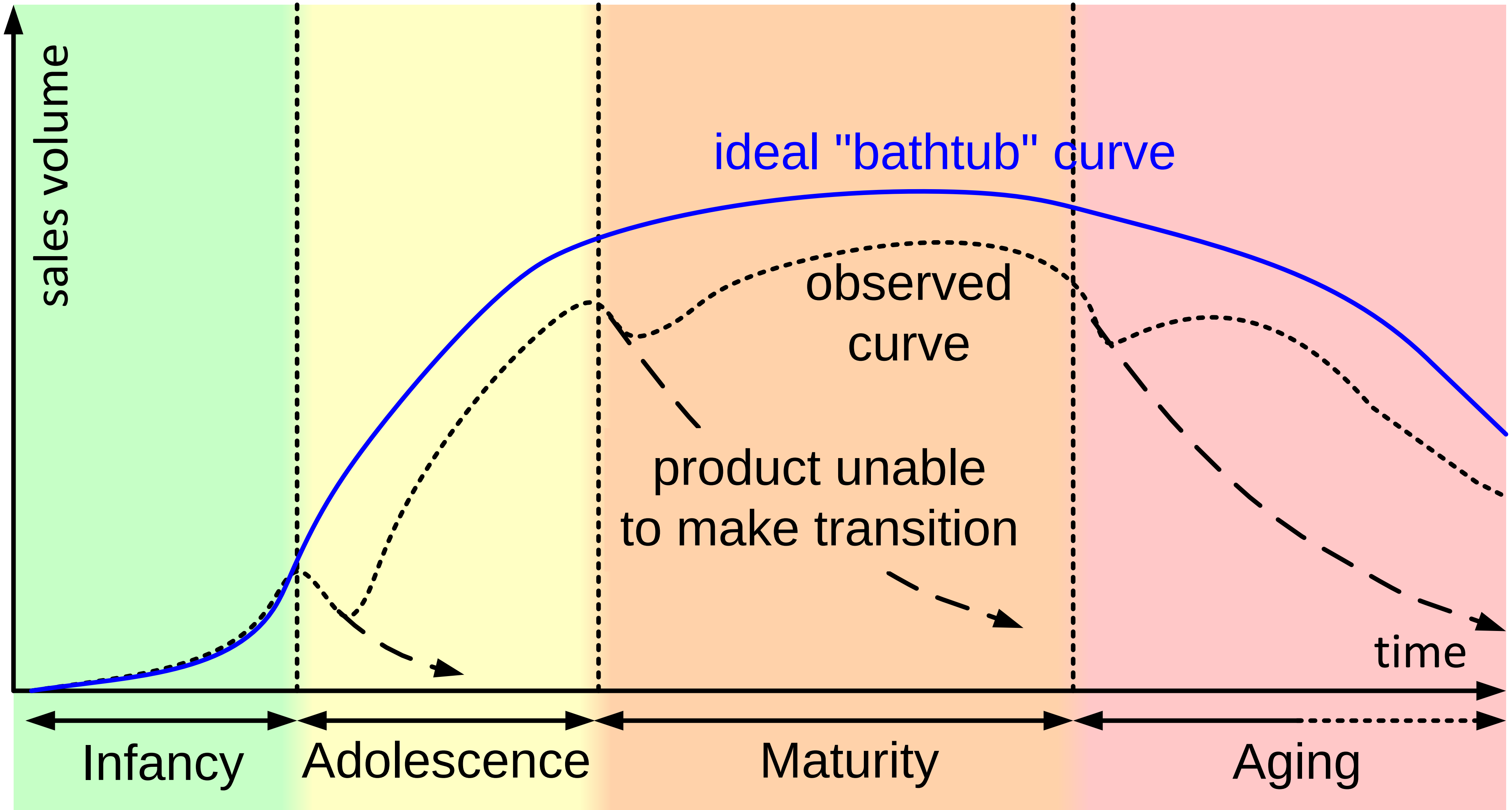
February 15, 2024
status: concept
version: 1.2



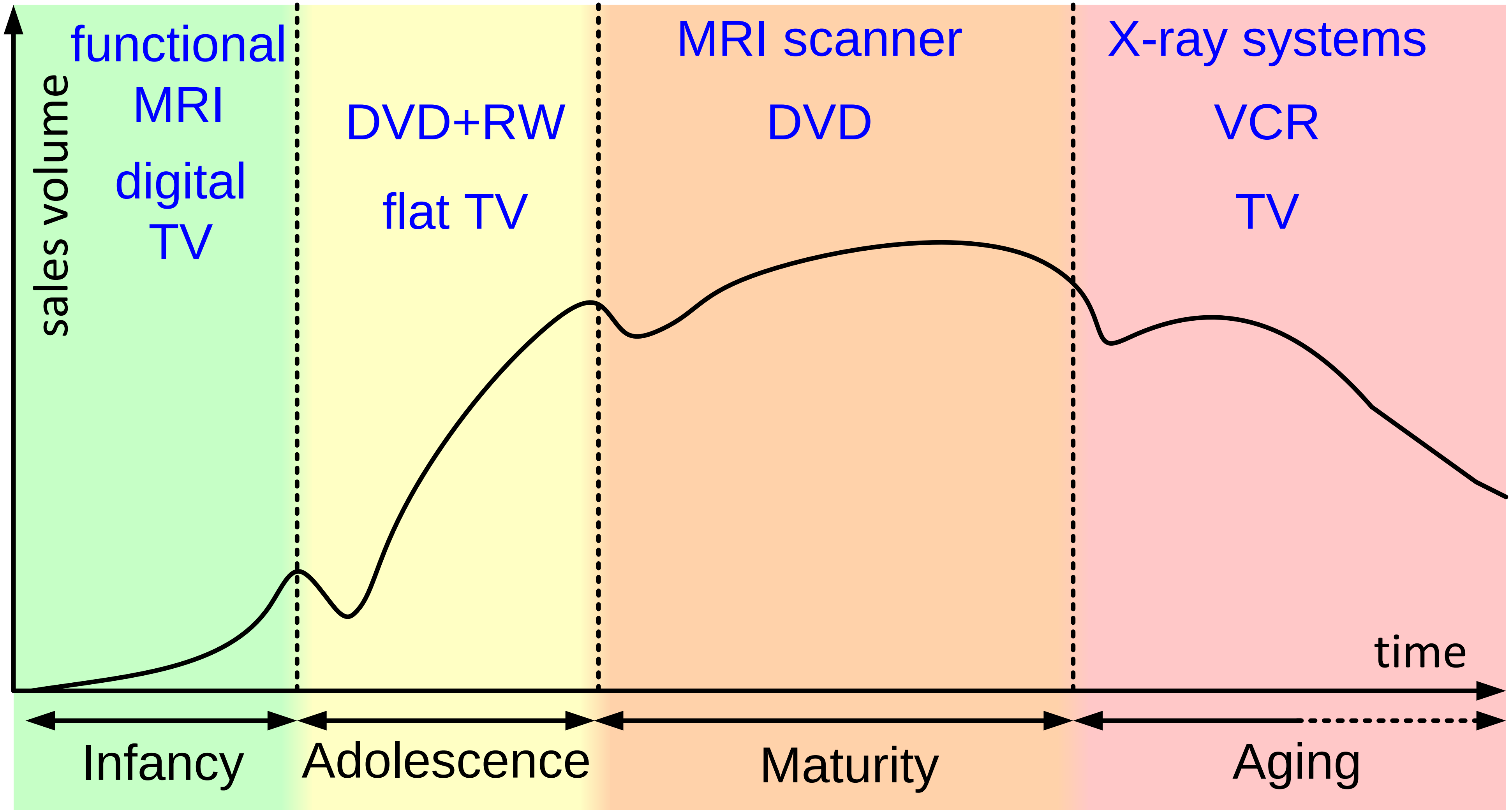
Ideal Bathtub Curve



Market Product Life Cycle Phases in Practice



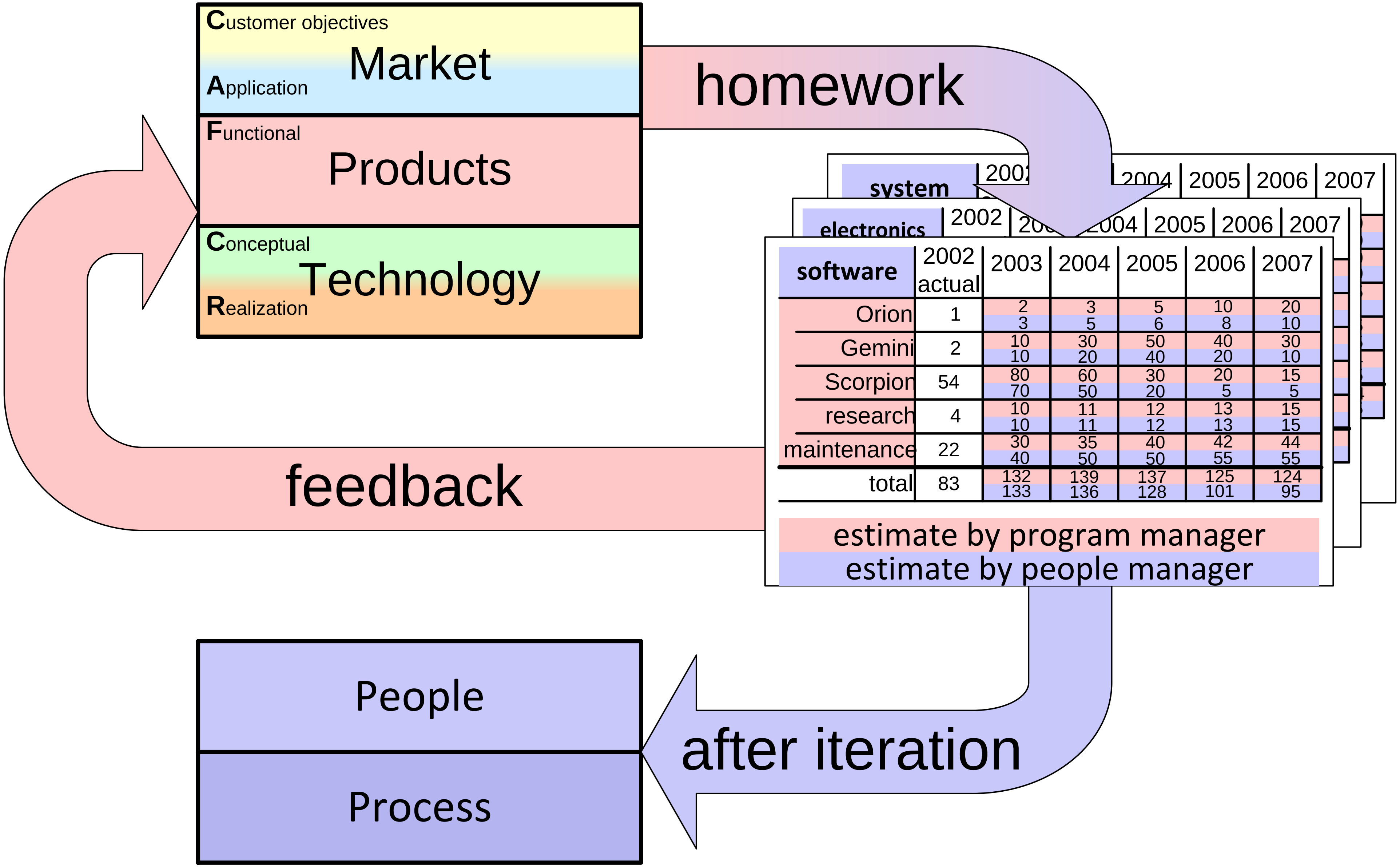
Examples of Product Classes on the Curve



Attributes per Phase

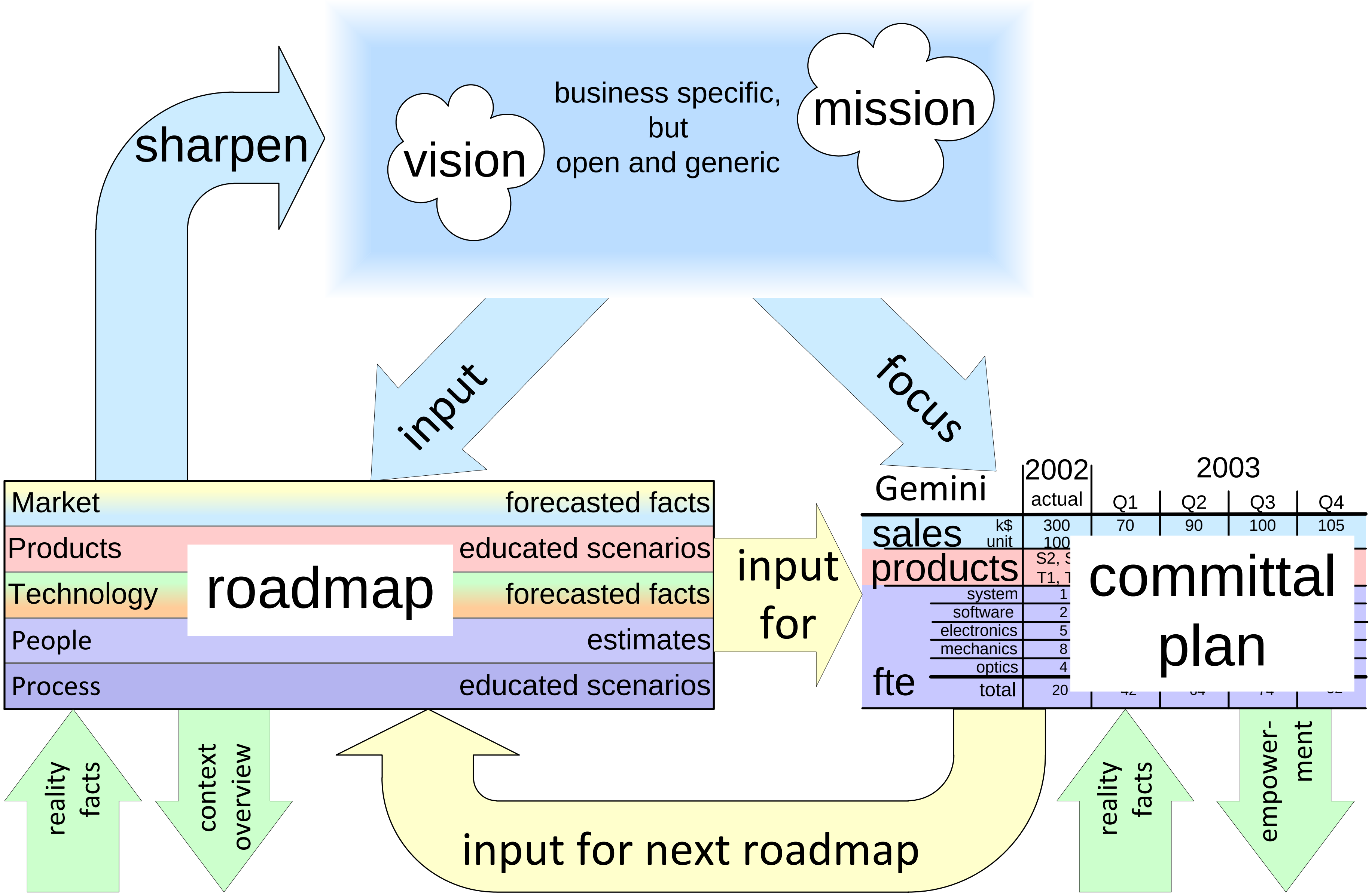
	Infancy	Adolescence	Mature	Ageing
Driving factor	Business vision		Stable business model	Harvesting of assets
Value from	Responsiveness	Features	Refinements / service	Refining existing assets
Requirements	Discovery	Select strategic	Prioritize	Low effort high value only
Dominant technical concerns	Feasibility	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

From Market, Product, Technology to People, Process



software	2002 actual	2003	2004	2005	2006	2007
Orion	1	2 3	3 5	5 6	10 8	20 10
Gemini	2	10 10	30 20	50 40	40 20	30 10
Scorpion	54	80 70	60 50	30 20	20 5	15 5
research	4	10 10	11 11	12 12	13 13	15 15
maintenance	22	30 40	35 50	40 50	42 55	44 55
total	83	132 133	139 136	137 128	125 101	124 95

Summary of strategy process



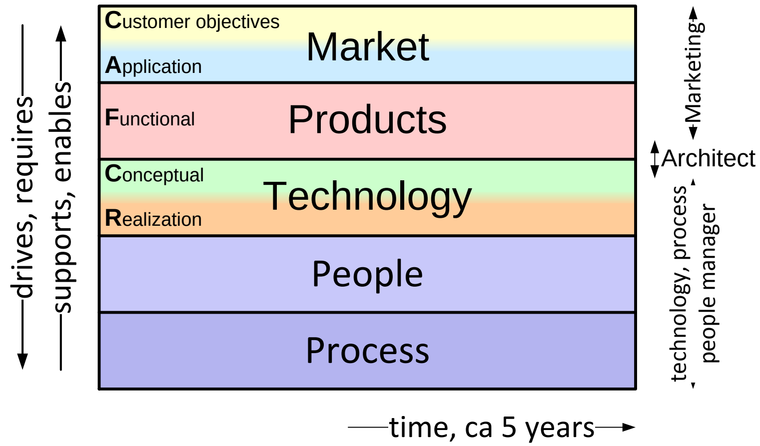
USN ESI

Exercise Roadmapping

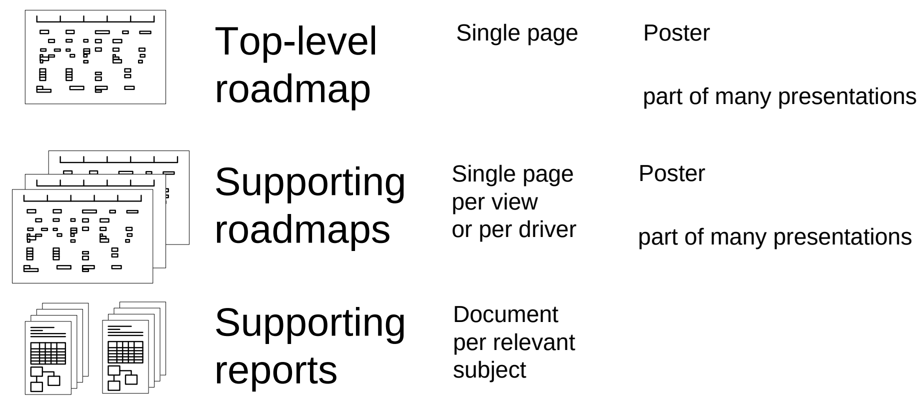
Make a roadmap on the basis of what you know at this moment, or what you perceive as the "shared expectation".
Try to fill in as many views (market, products, technology, people and process) as possible.
Present an overview by minimizing the contents to the most essential data.

Roadmap Creation

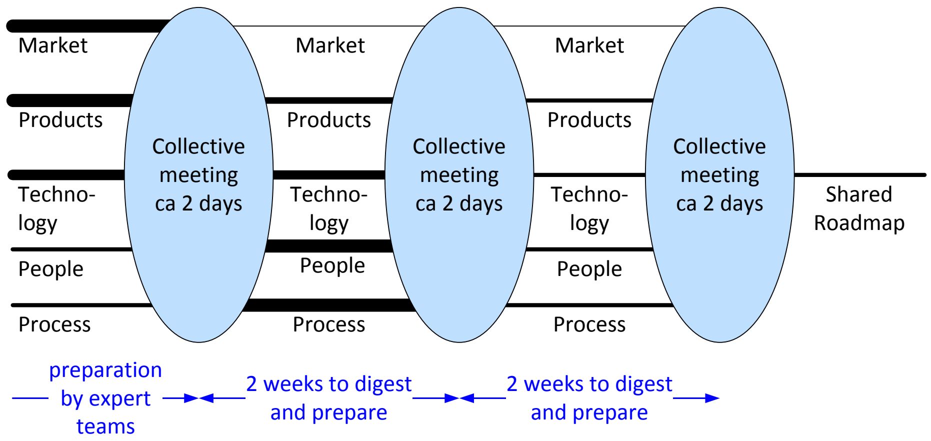
The Roadmap Integrates Five Views



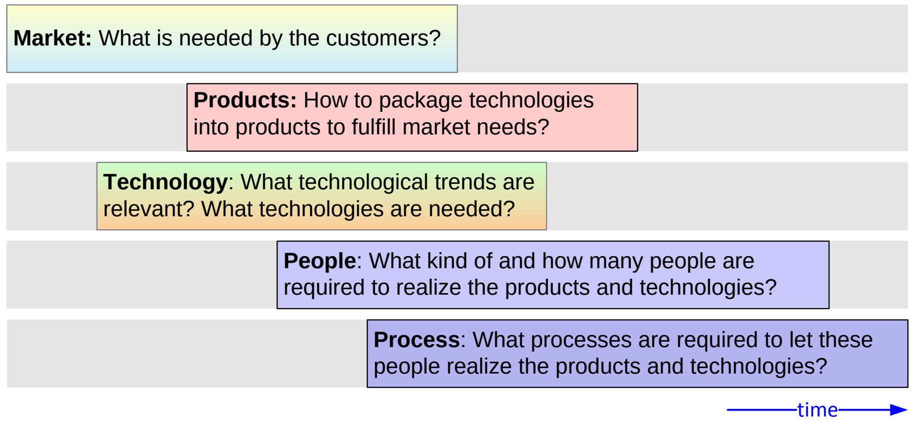
Multiple Levels



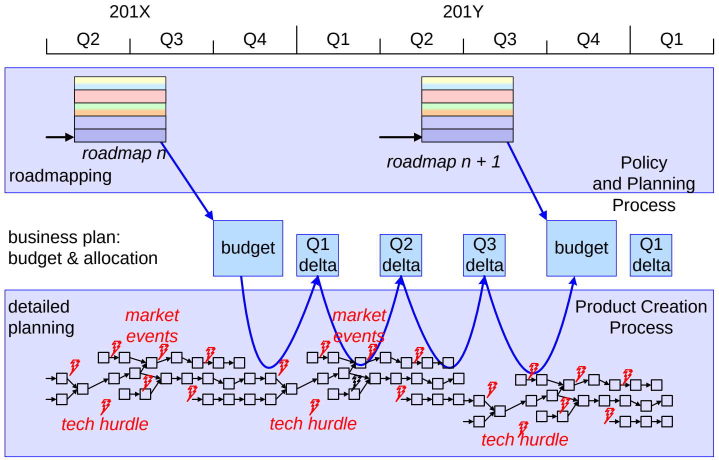
Creation in Teams



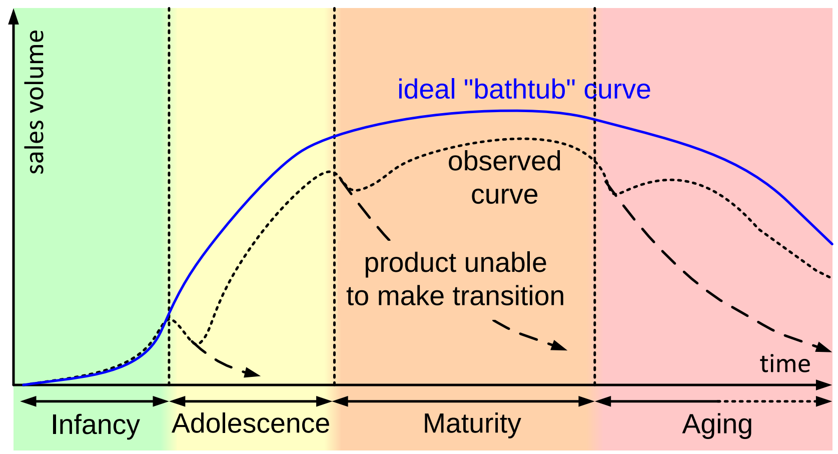
Order of Creation



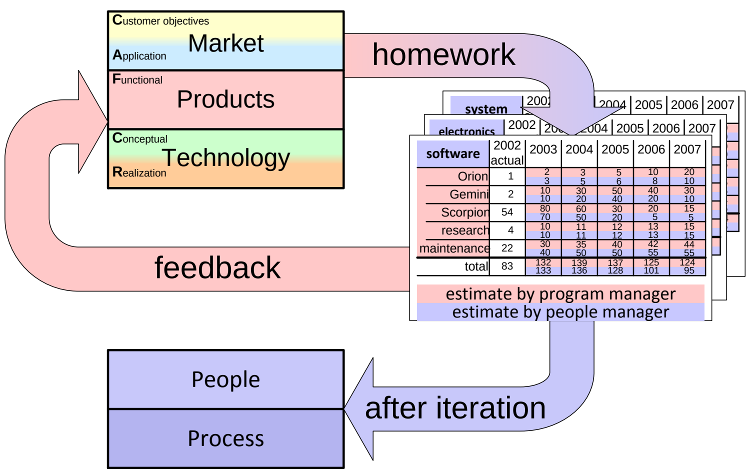
Time Horizons



Life Cycle Transitions



People and Process



intentionally left blank

Module Product Families and Generic Developments

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

This module addresses product families and generic developments.

February 15, 2024
status: preliminary draft
version: 1.3

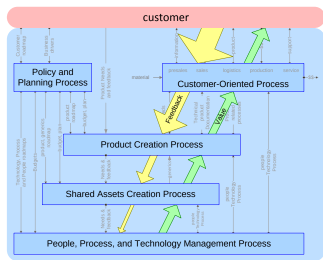
Product Families and Generic Aspects

by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

Most products fit in a larger family of products. The members of such a product family share a lot of functionality and features. It is attractive to share implementations, designs et cetera between those members to increase the efficiency of the entire company.
In practice many difficulties pop up when product developments become coupled, due to the partial developments which are shared. This article discusses the advantages and disadvantages of a family approach based on shared developments and provides some methods to increase the chance on success.

February 15, 2024
status: concept
version: 2.3



Typical Examples of Generic Developments

Platform

Common components

Standard design

Framework

Family architecture

Generic aspects, functions, or features

Reuse

Products (in project environment)

Claimed Advantages of Generic Developments

Reduced time to market	building on shared components
Reduced cost per function	build every function only once
Improved quality Improved reliability Improved predictability	maturing realization
Easier diversity management	modularity
Increases uniformity Employees only have to understand one base system	less learning
Larger purchasing power	economy of scale
Means to consolidate knowledge	
Increase added value	not reinventing existing functionality
Enables parallel developments of multiple products	
“Free” feature propagation	product-to-product or project-to-project

Experiences with reuse, from counterproductive to effective

bad	good
longer time to market	reduced time to market
high investments	reduced investment
lots of maintenance	reduced (shared) maintenance cost
poor quality	improved quality
poor reliability	improved reliability
diversity is opposed	easier diversity management
lot of know how required	understanding of one base system
predictable too late	improved predictability
dependability	larger purchasing power
knowledge dilution	means to consolidate knowledge
lack of market focus	increase added value
interference	enables parallel developments
but integration required	free feature propagation

Successful examples of reuse

homogeneous domain	cath lab MRI television waferstepper
hardware dominated	car airplane shaver television
limited scope	audio codec compression library streaming library

Limits of successful reuse

struggle with integration/convergence with other domains

TV: digital networks and media
cath lab: US imaging, MRI

poor/slow response on paradigm shifts

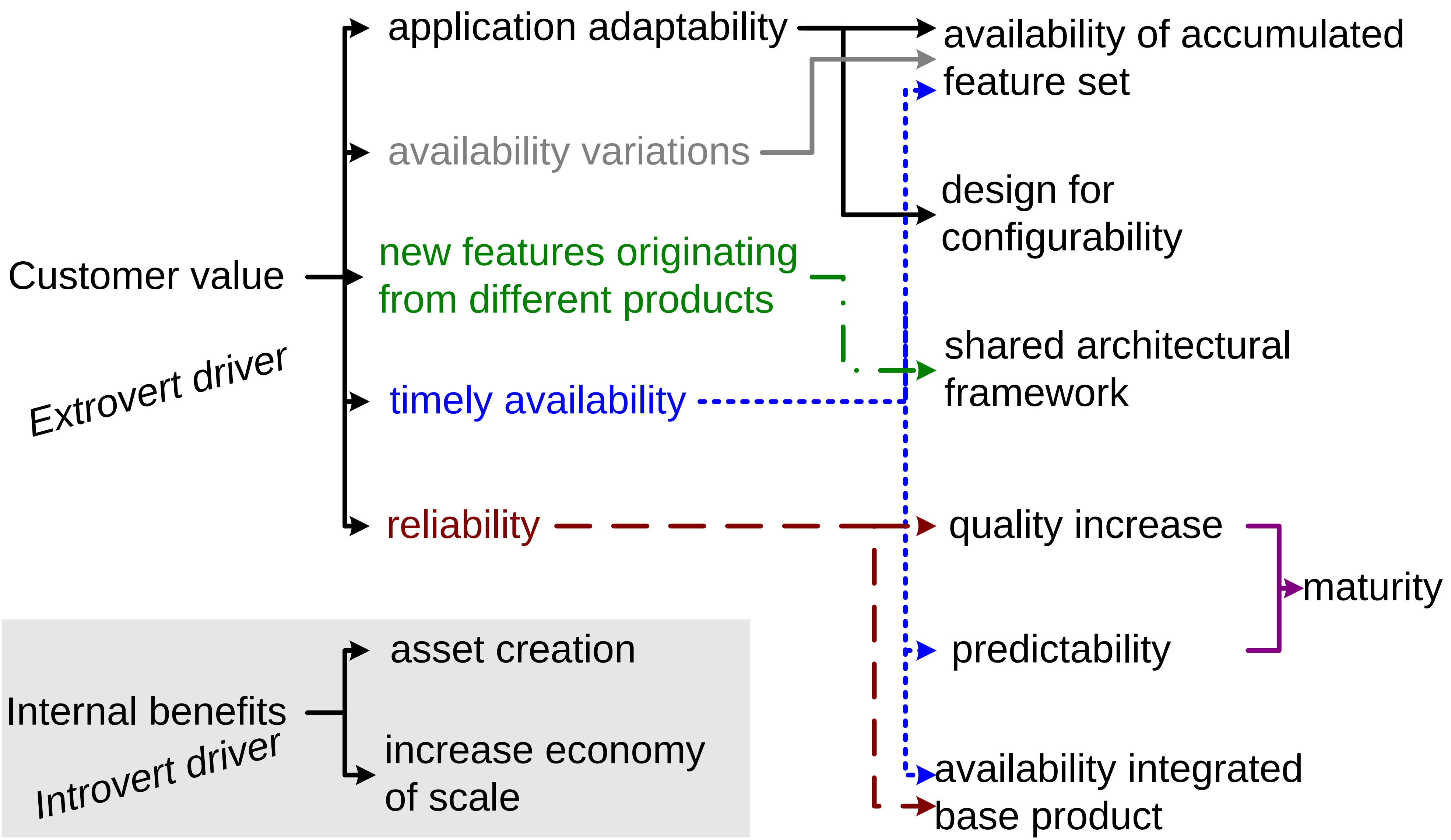
TV: LCD screens
cath lab: image based acquisition control

software maintenance, configurations, integration, release

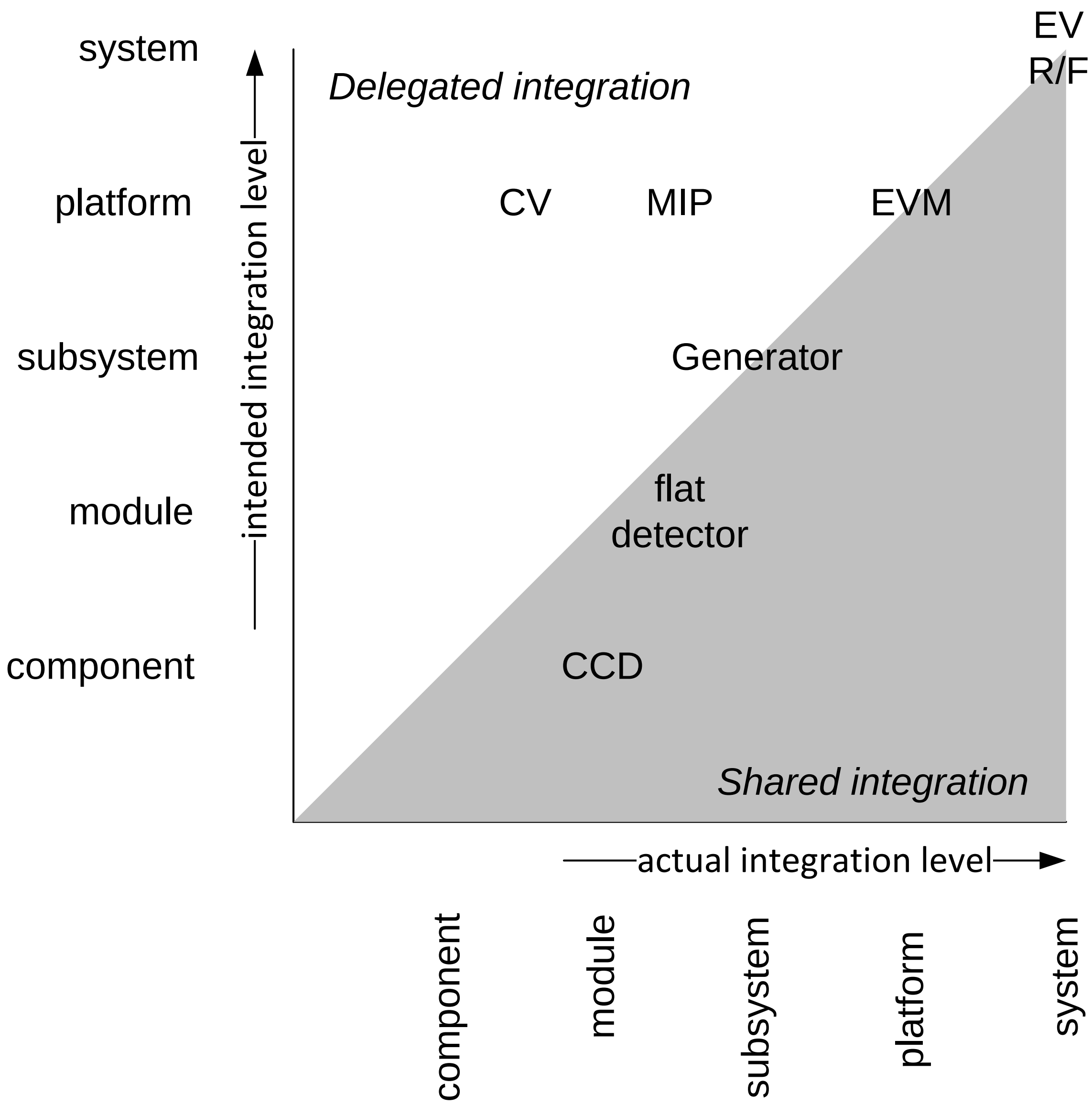
MRI: integration and test
wafersteppers: number of configurations

how to innovate?

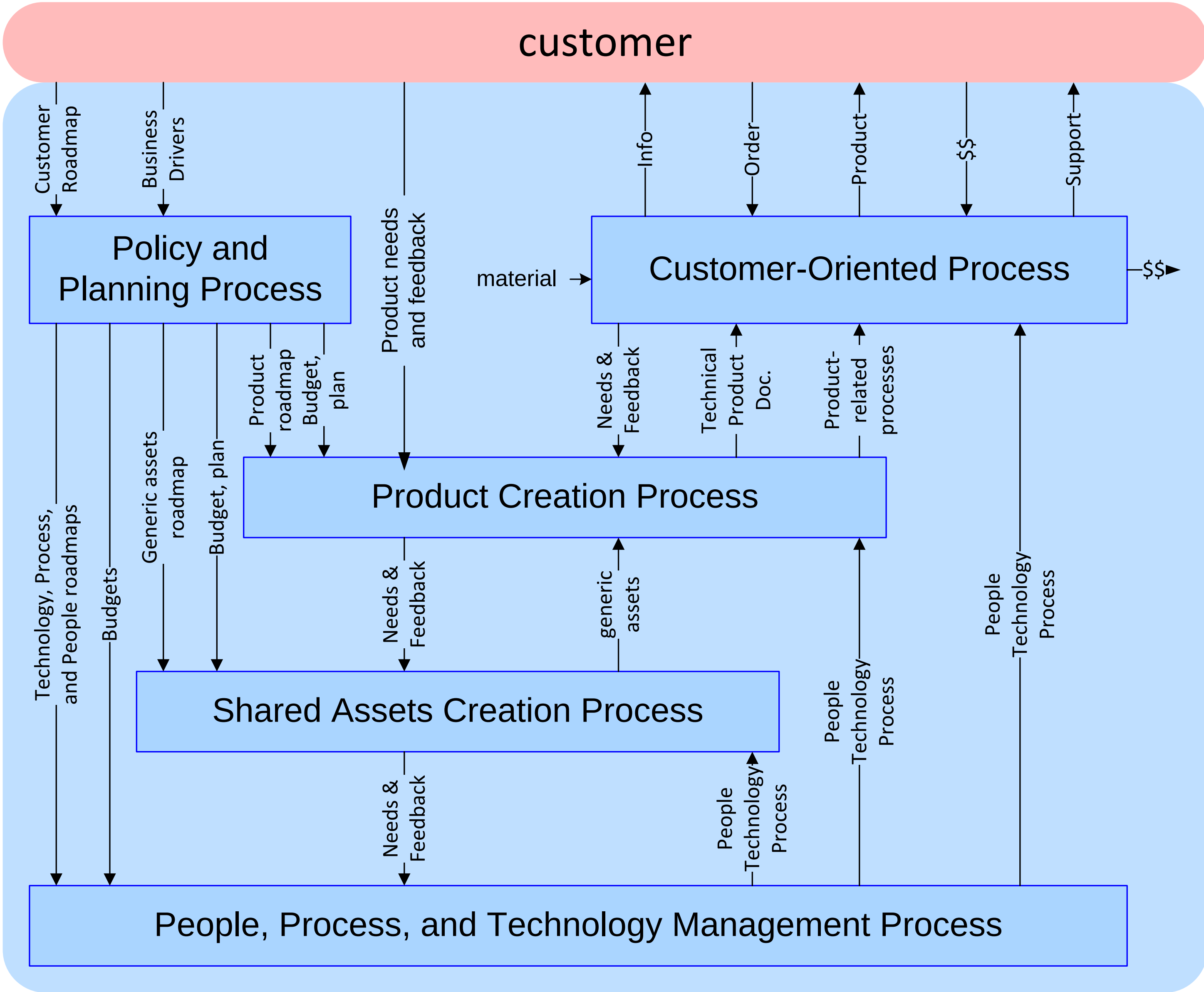
Drivers for Generic Developments



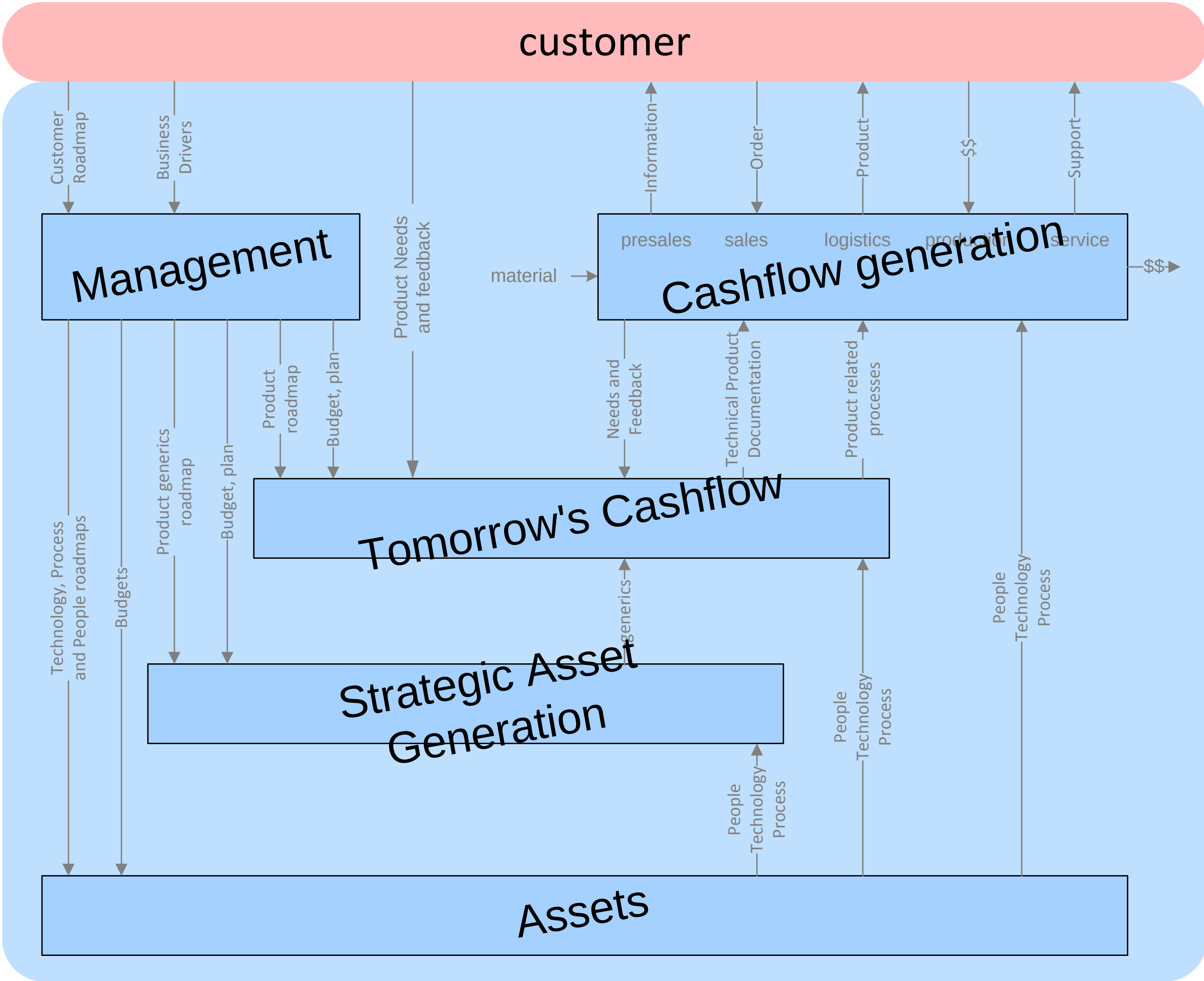
Granularity of generic developments shown in 2 dimensions



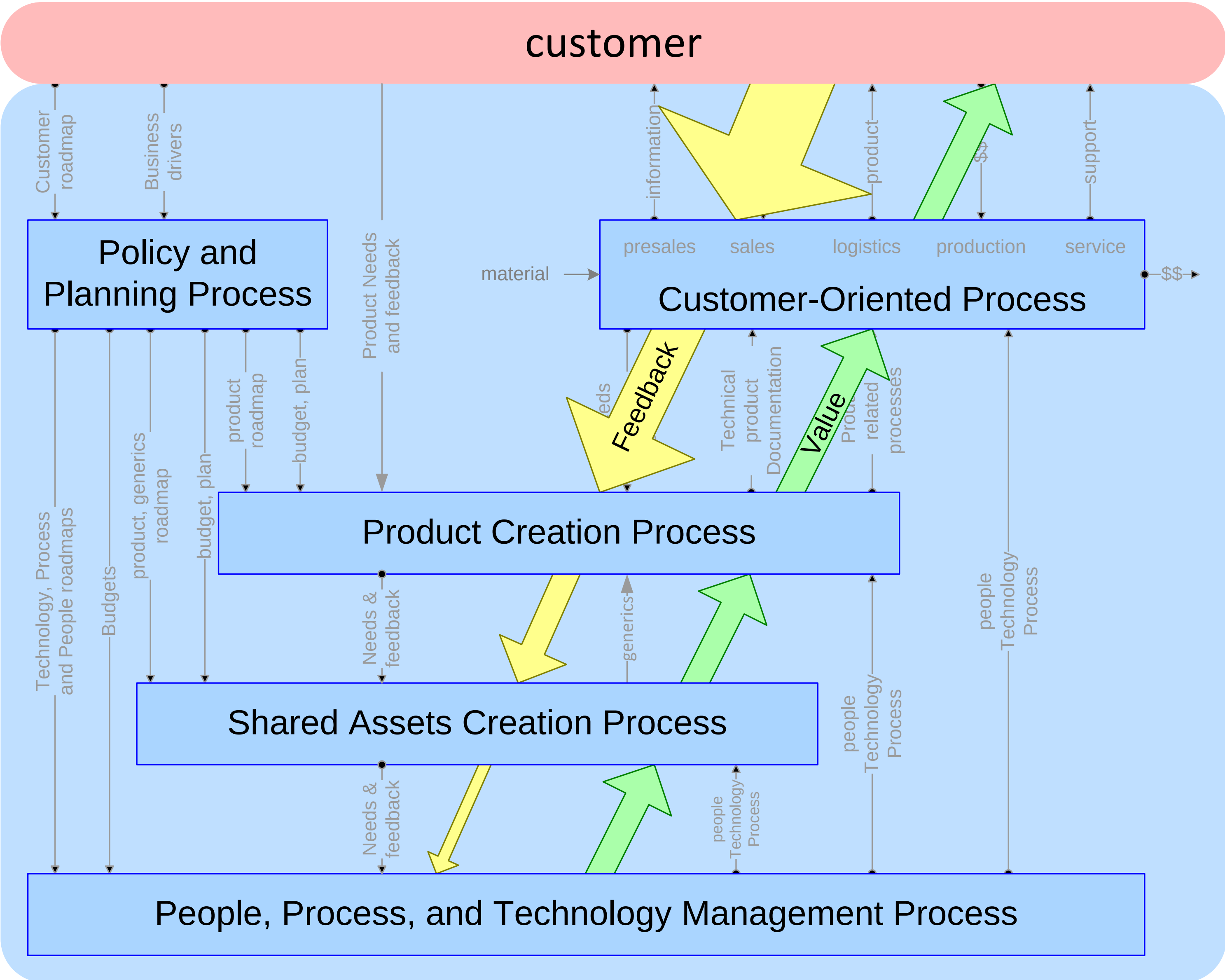
Modified Process Decomposition



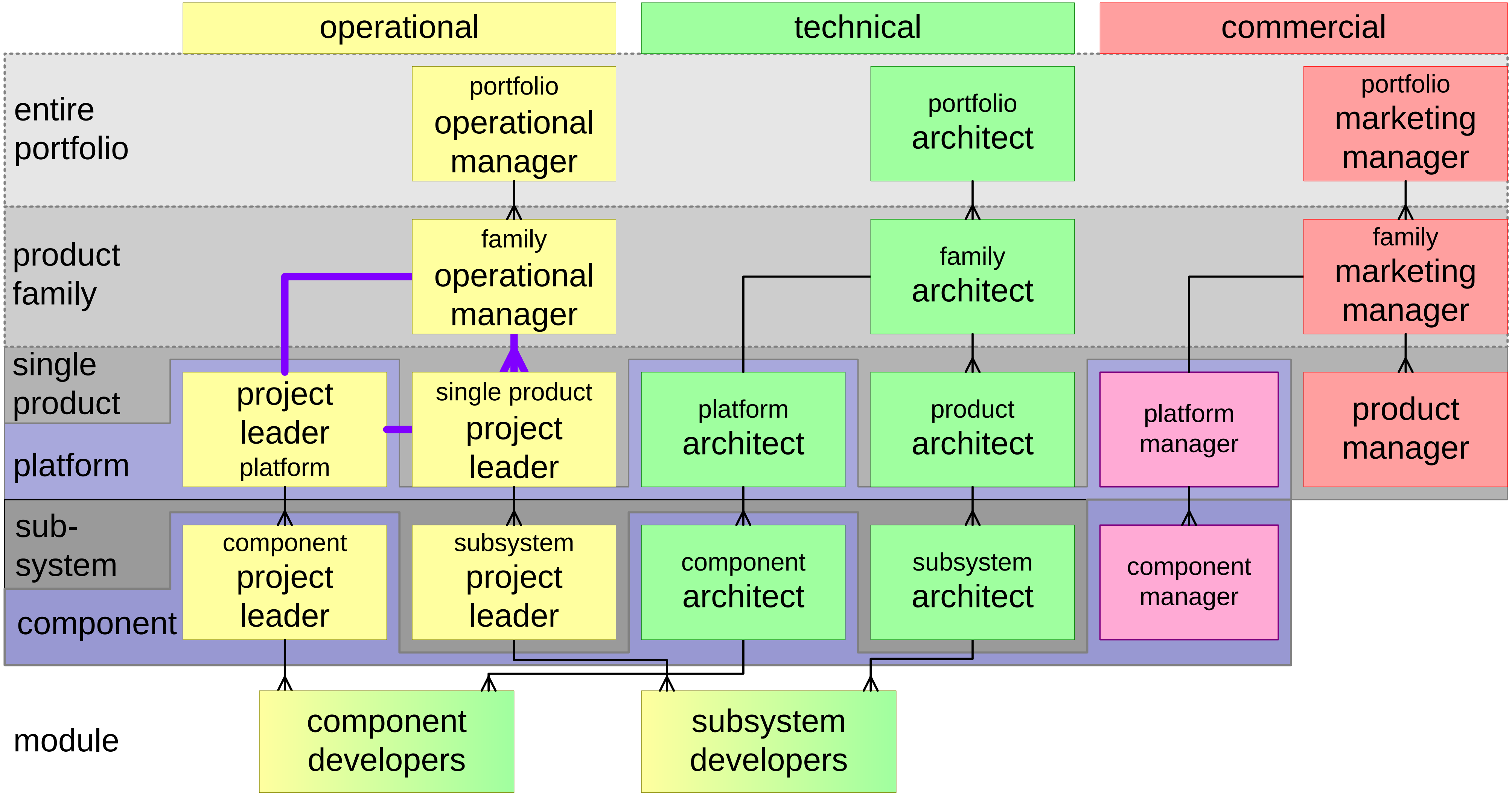
Financial Viewpoint on Process Decomposition



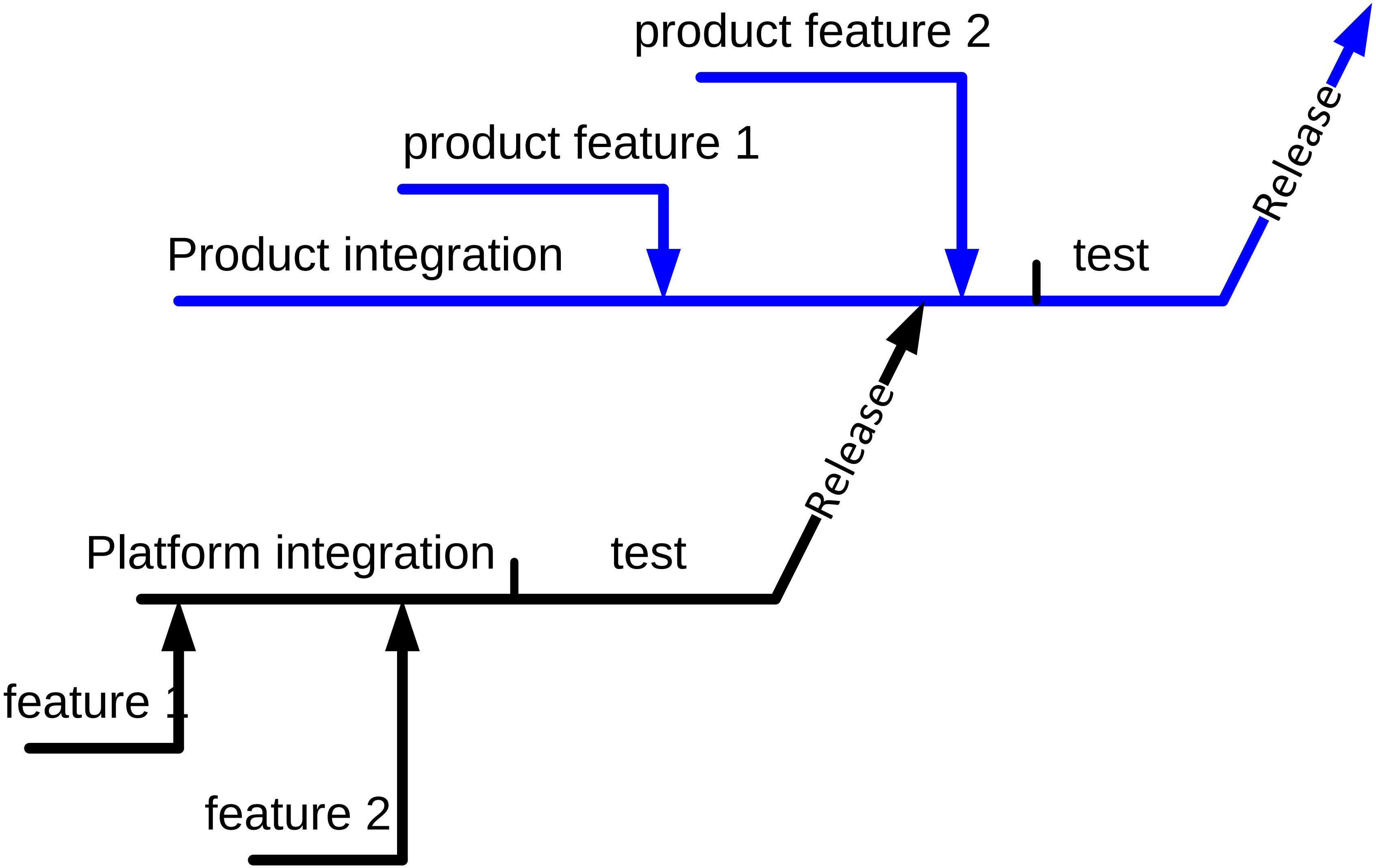
Value and Feedback Flow



Modified Operational Organization PCP



Propagation Delay Platform Feature to Market



Sources of Failure in Generic Developments

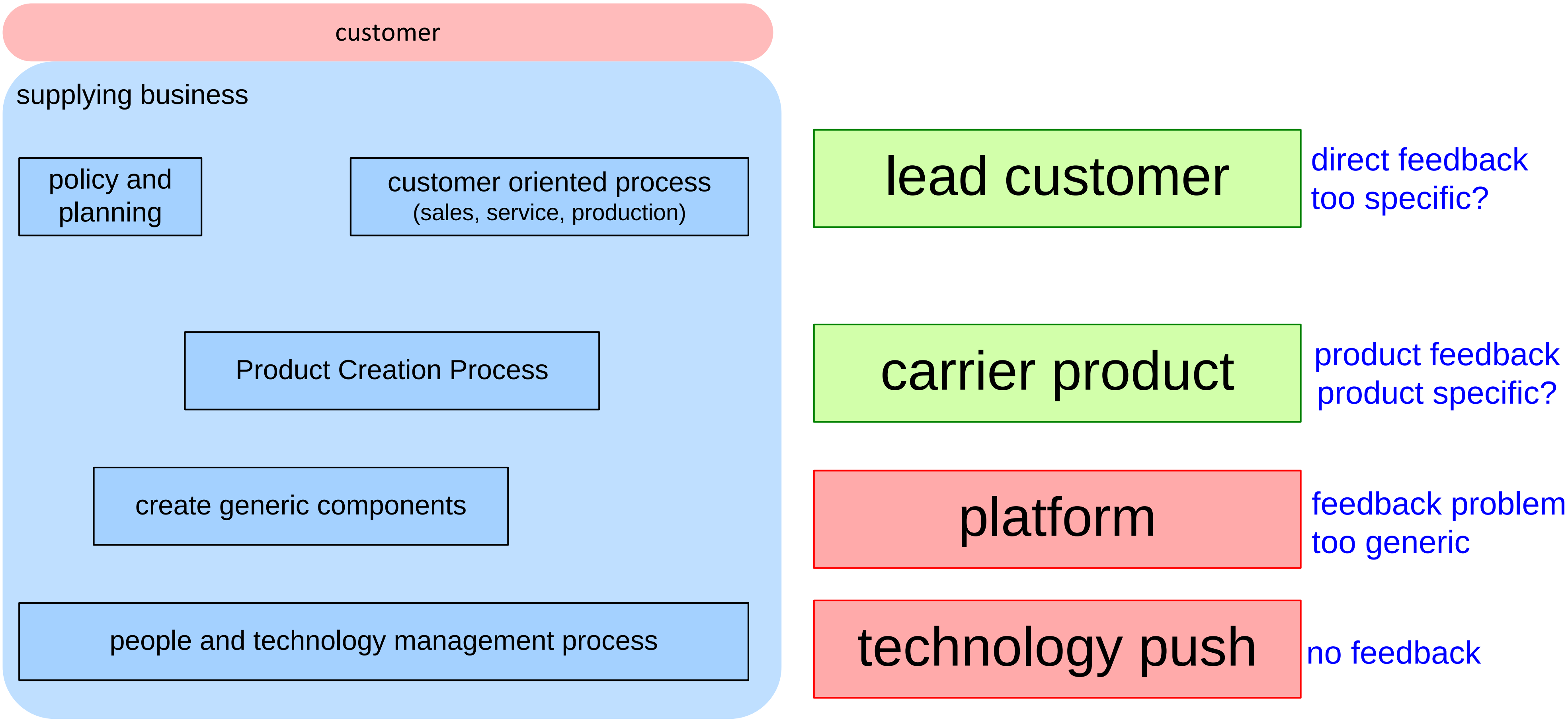
Technical

- Too generic
- Innovation stops
  (stable interfaces)
- Vulnerability

Process/People/Organization

- Forced cooperation
- Time platform feature to market
- Unrealistic expectations
- Distance platform developer to customer
- No marketing ownership
- Bureaucratic process (no flexibility)
- New employees, knowledge dilution
- Underestimation of platform support
- Overstretching of product scope
- Nonmanagement, organizational scope increase
- Underestimation of integration
- Component/platform determines business policy
- Subcritical investment

Models for Generic Development



Exercise Generic Developments

What are the top 3 benefits for your product family or generic development?

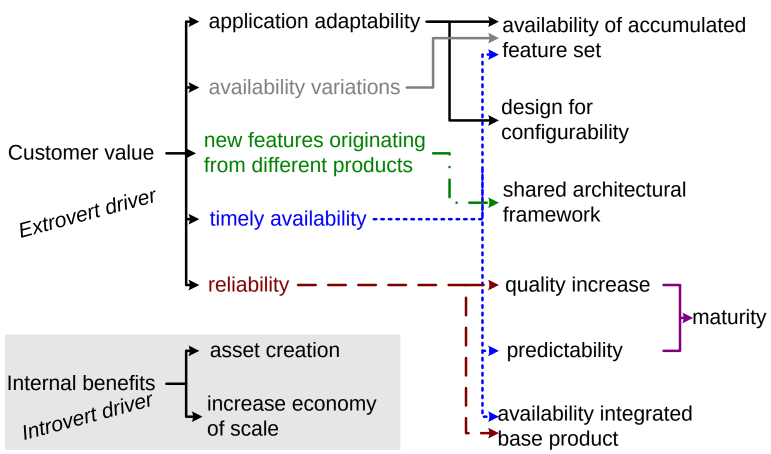
What are the top 3 disadvantages?

Harvesting Synergy

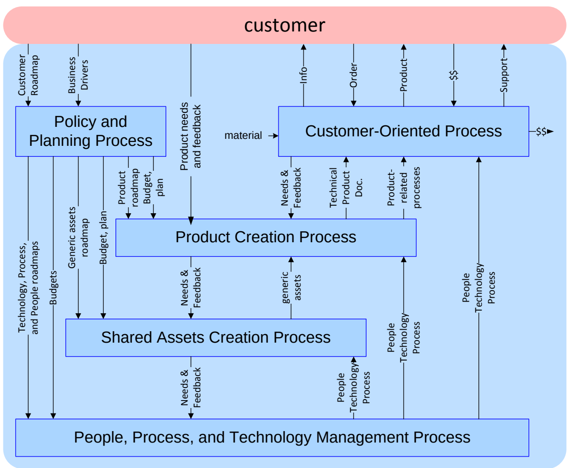
Contradicting Experiences

bad	good
longer time to market	reduced time to market
high investments	reduced investment
lots of maintenance	reduced (shared) maintenance cost
poor quality	improved quality
poor reliability	improved reliability
diversity is opposed	easier diversity management
lot of know how required	understanding of one base system
predictable too late	improved predictability
dependability	larger purchasing power
knowledge dilution	means to consolidate knowledge
lack of market focus	increase added value
interference	enables parallel developments
but integration required	free feature propagation

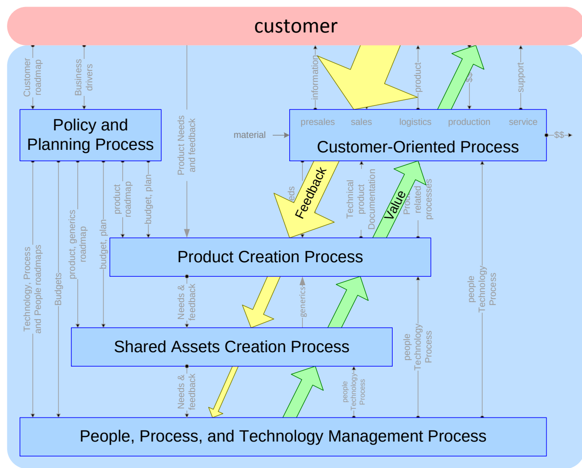
Drivers



Shared Asset Creation Process

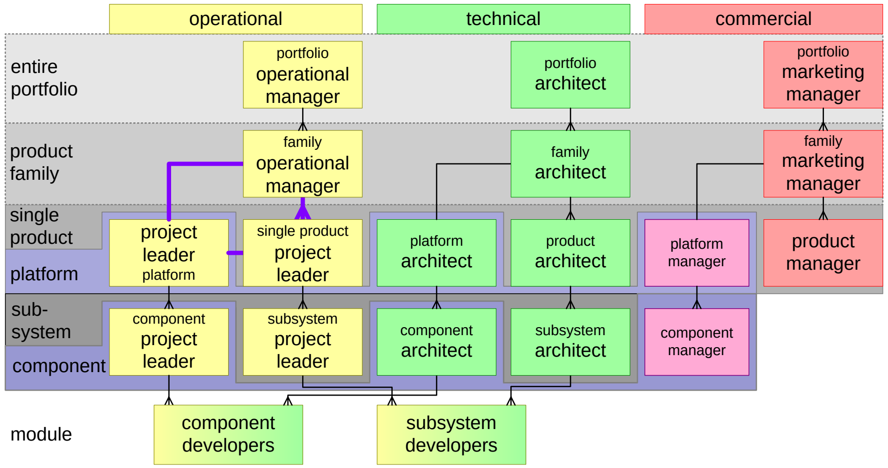


Longer Chains

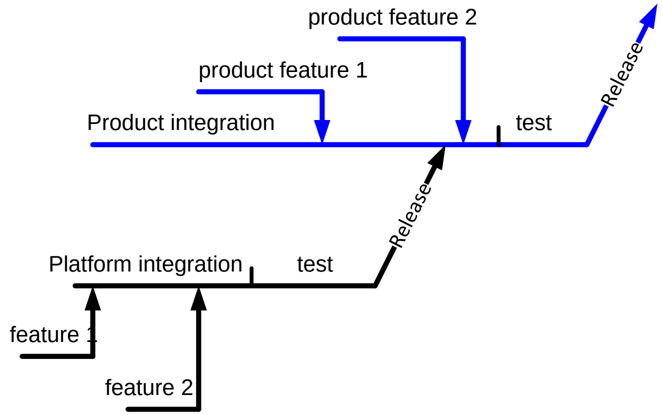


Some Architecting Means

Organizational Complexity



Delay to Market



Pitfalls

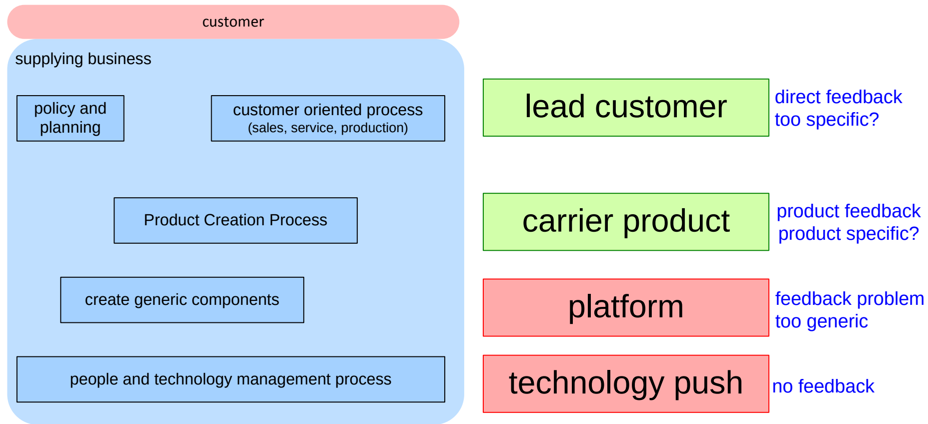
Technical

- Too generic
- Innovation stops (stable interfaces)
- Vulnerability

Process/People/Organization

- Forced cooperation
- Time platform feature to market
- Unrealistic expectations
- Distance platform developer to customer
- No marketing ownership
- Bureaucratic process (no flexibility)
- New employees, knowledge dilution
- Underestimation of platform support
- Overstretching of product scope
- Nonmanagement, organizational scope increase
- Underestimation of integration
- Component/platform determines business policy
- Subcritical investment

Successful and Failing Models



Module Supporting Processes

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

This module addresses supporting processes, for instance documentation, templates, and reviewing.

February 15, 2024
status: draft
version: 1.4

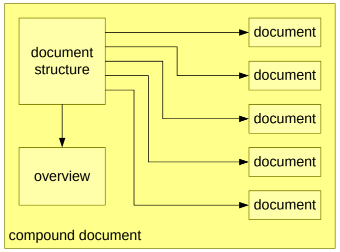
Granularity of Documentation

by *Gerrit Muller* USN-SE
e-mail: `gaudisite@gmail.com`
`www.gaudisite.nl`

Abstract

The design of documentation is discussed, with emphasis on the requirements, the need for decomposition, the measures needed to maintain overview and criteria for granularity.

February 15, 2024
status: concept
version: 1.2



Requirements for the Entire Documentation Structure

Accessibility for the readers

Low threshold for the readers

Low threshold for the authors

Completeness

Consistency

Maintainability

Scalability

Evolvability

Process to ensure the quality of the information

Requirements from Reader Point of View

Convenient

easy

fast

viewing

printing

searching

Requirements per Document

High cohesion (within the unit)

Low coupling (outside of the unit)

Accessibility for the readers

Low threshold for the reader

Low threshold for the author

Manageable steps to create, review, and change

Clear responsibilities

Clear position and relation with the context

Well-defined status of the information

Timely availability

Accessibility Requirements

Ease of reading, “juiciness”

High signal-to-noise ratio: information should not be hidden in a sea of words.

Understandability

Reachability in different ways, e.g., by hierarchical or full search

Reachability in a limited number of steps

Responsibility Requirements

single author

limited amount of reviewers

Scalability Requirements

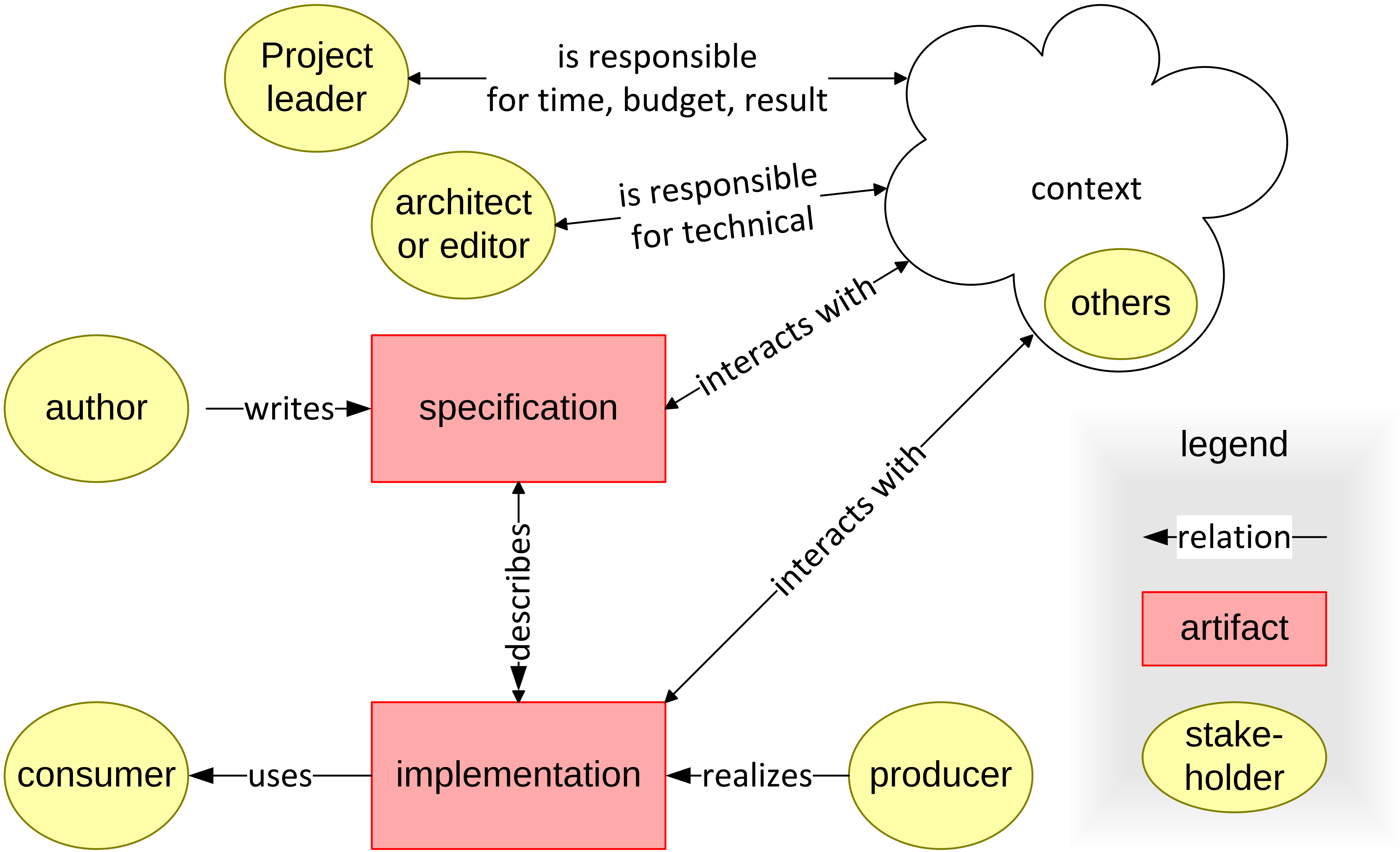
well defined documentation structure

overview specifications at higher aggregation levels

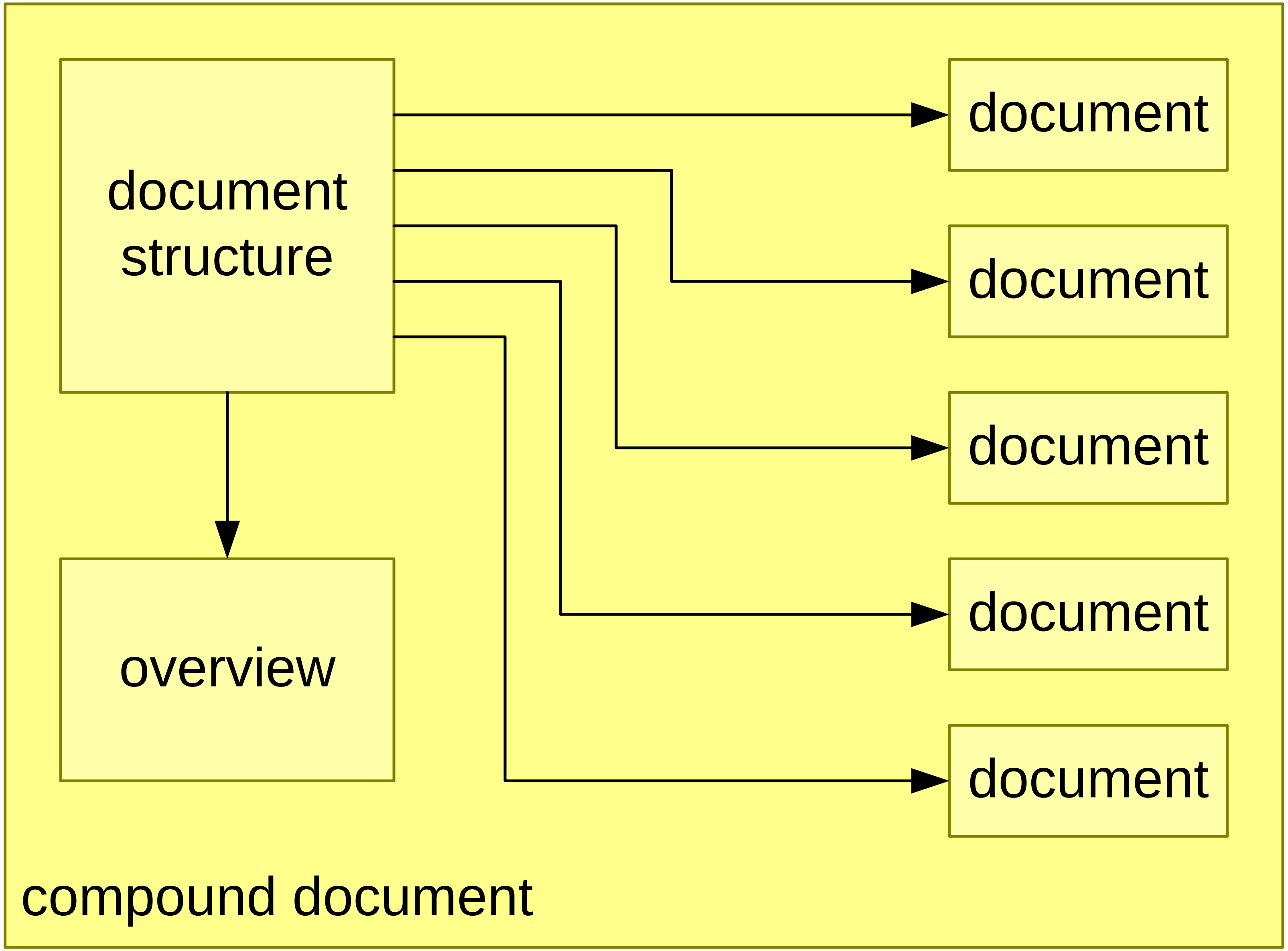
recursive application of structure and overview

delegation of review process

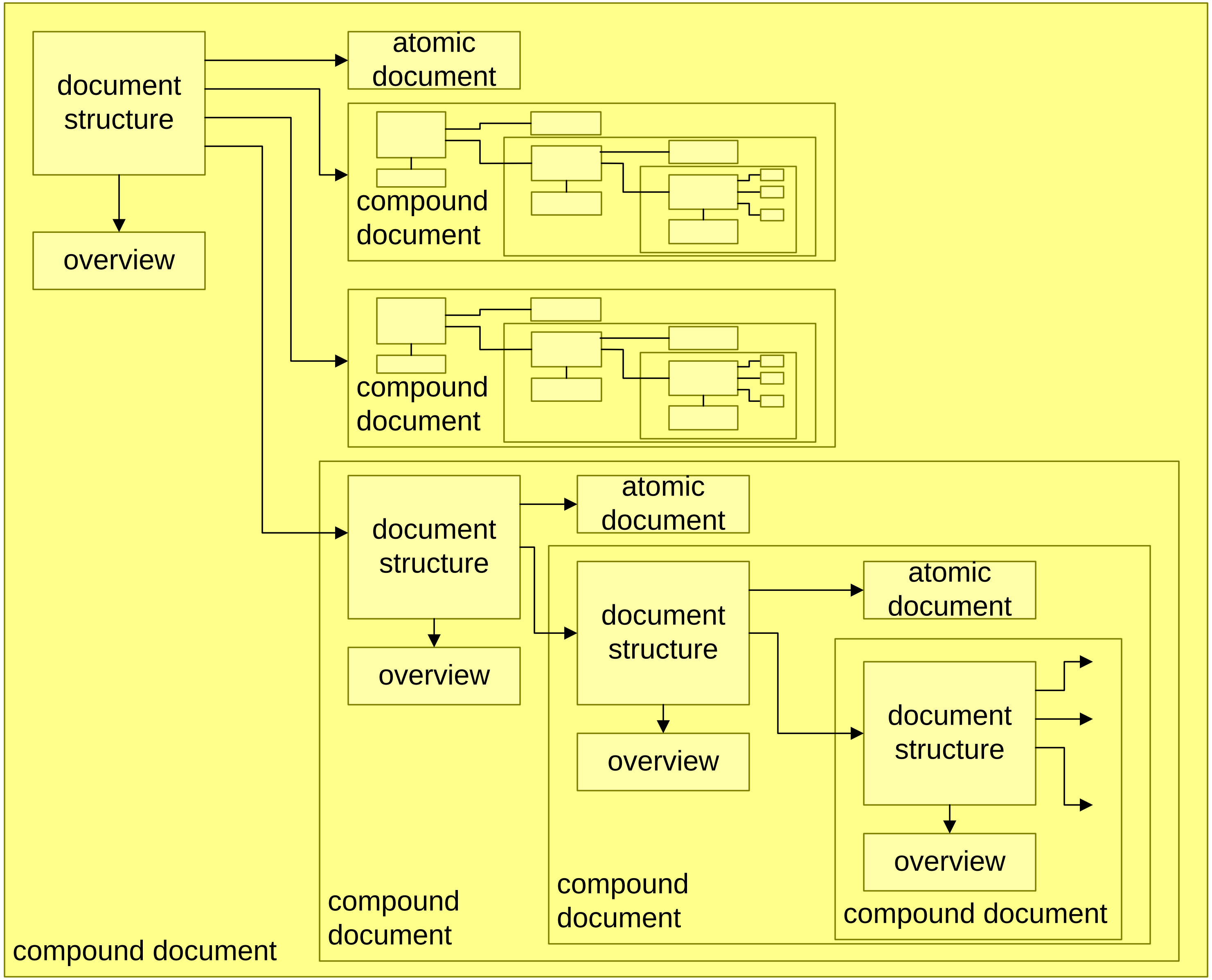
The Stakeholders of a Single Document



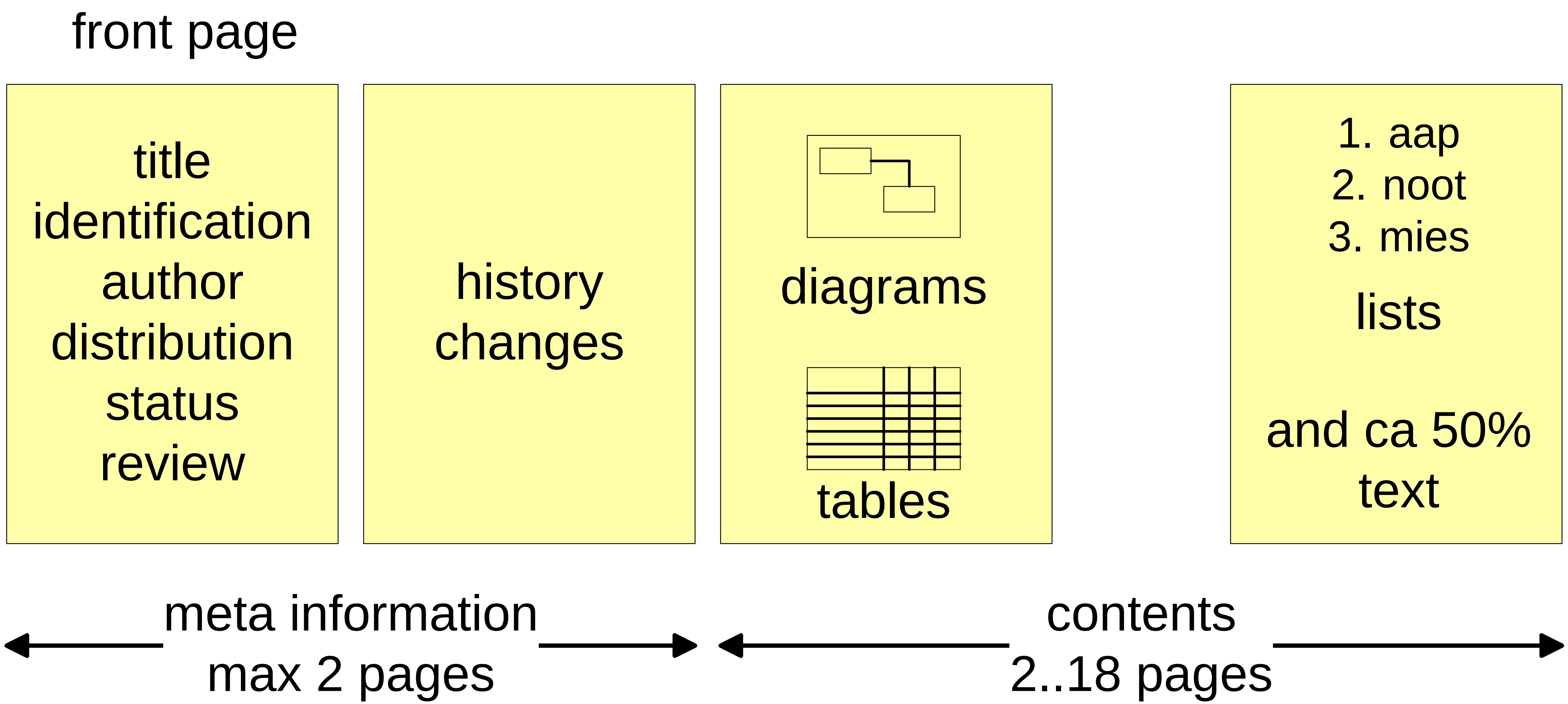
Decomposition of Large Documents



Documentation Tree by Recursive Decomposition



Payload: the Ratio between Content and Overhead



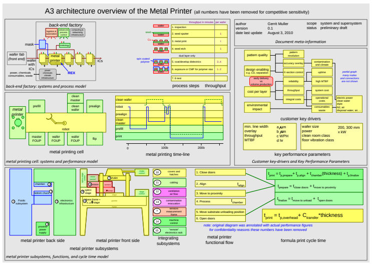
LEAN and A3 Approach to Supporting Processes

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

LEAN product development is in the process and means area pragmatic. Low tech tools, such as paper, pen and magnets, with very direct interaction are used. For communication the use of single A3-size documents is promoted, because this is a manageable amount of information.

February 15, 2024
status: draft
version: 0.1



Characteristics of LEAN

A holistic, systems approach to product development
including people, processes, and technology.

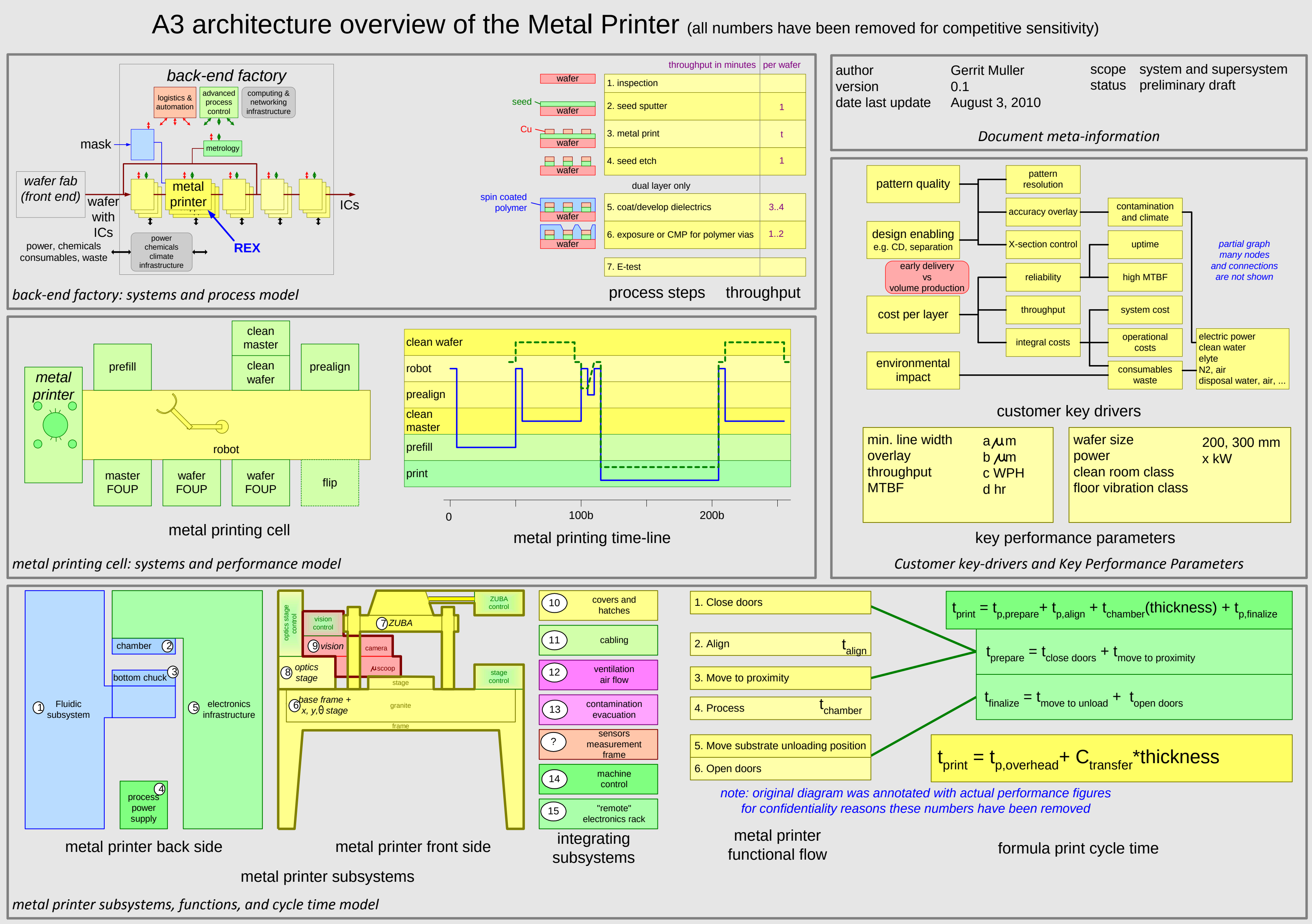
Multi-disciplinary from the early start, with a drive to be fact based.

Customer understanding as the the starting point.

Continuous improvement and learning as cultural value.

Small distance between engineers and real systems, including manufacturing, sales and service and the system of interest.

Example of A3 Architecture Overview

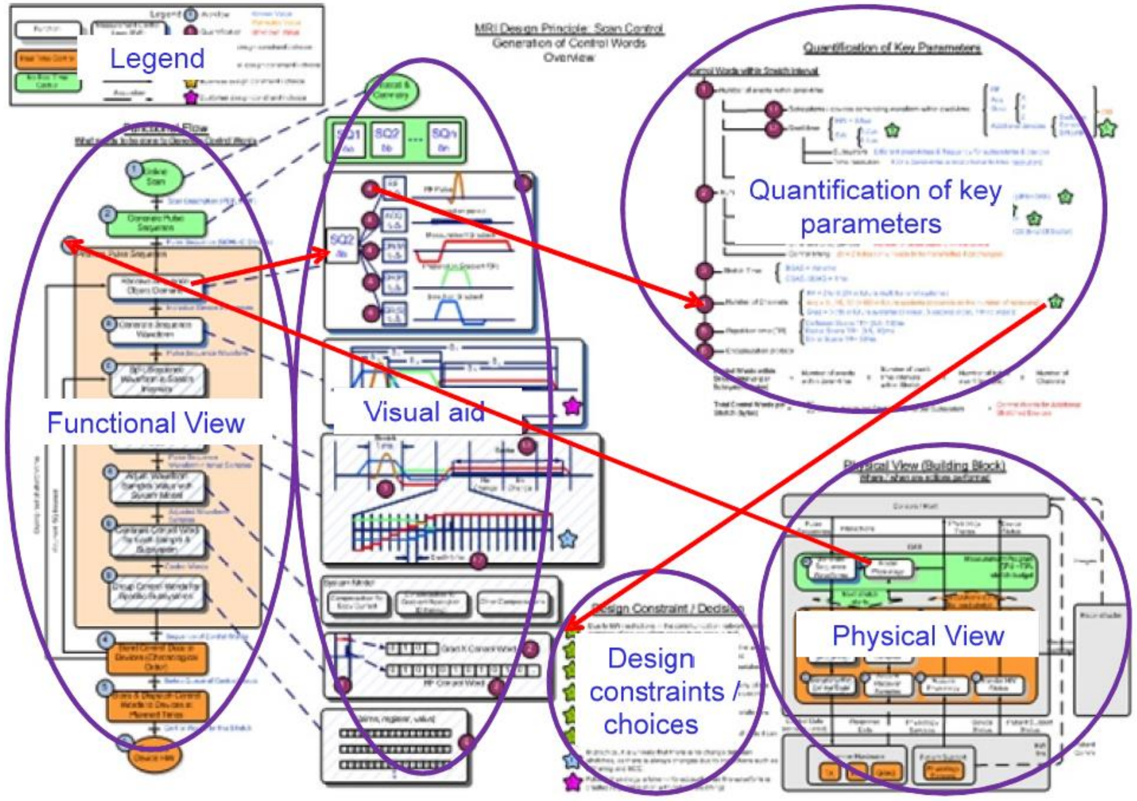


multiple related views

quantifications

one topic
per A3

capture
"hot" topics



source: PhD thesis Daniel Borches http://doc.utwente.nl/75284/

digestable
(size limitation)

practical
close to stakeholder experience

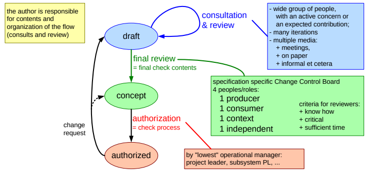
Light Weight Review Process

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

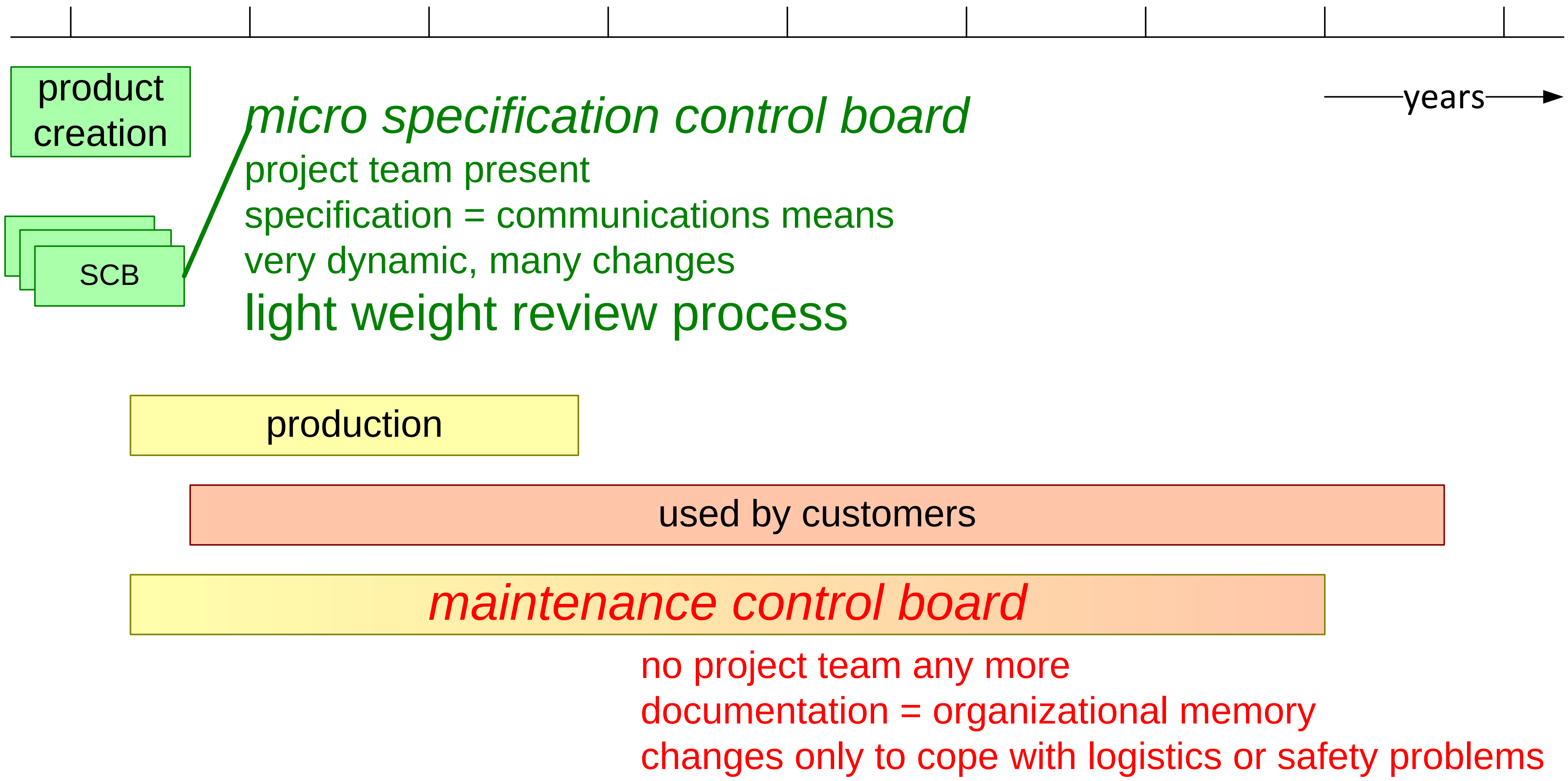
Abstract

A light weight review process is described that can be used for documents made during product creation. This review process is focused on improving the contents of specifications as early as possible. The process is light weight to increase the likelihood that it is performed *de facto* instead of *pro forma*.

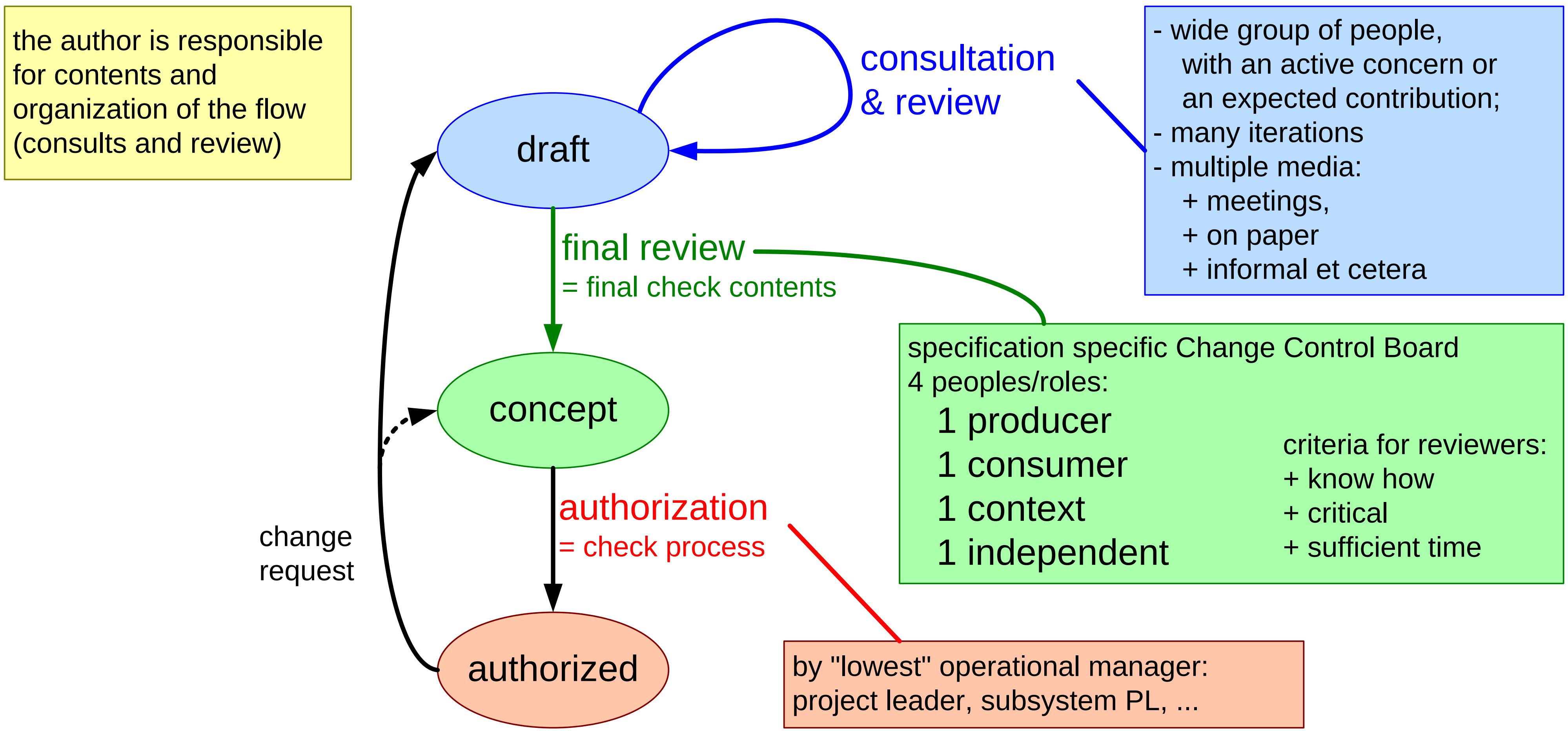
February 15, 2024
status: preliminary draft
version: 0



Product Life Cycle and Change Management



Light Weight Specification Review Process

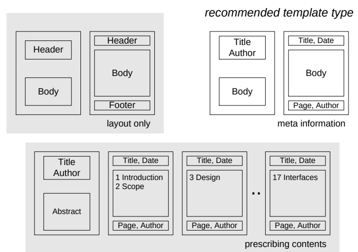


Template How To

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

The introduction of a new process (way of working) is quite often implemented by supplying ready-to-go tools and templates. This implementation mainly serves the purpose of a smooth introduction of the new process.
Unfortunately the benefits of templates are often cancelled by unforeseen side-effects, such as unintended application, inflexibility, and so on. This intermezzo gives hints to avoid the **Template Trap**, so that templates can be used more effectively to support introduction of new processes.



February 15, 2024
status: draft
version: 1.6

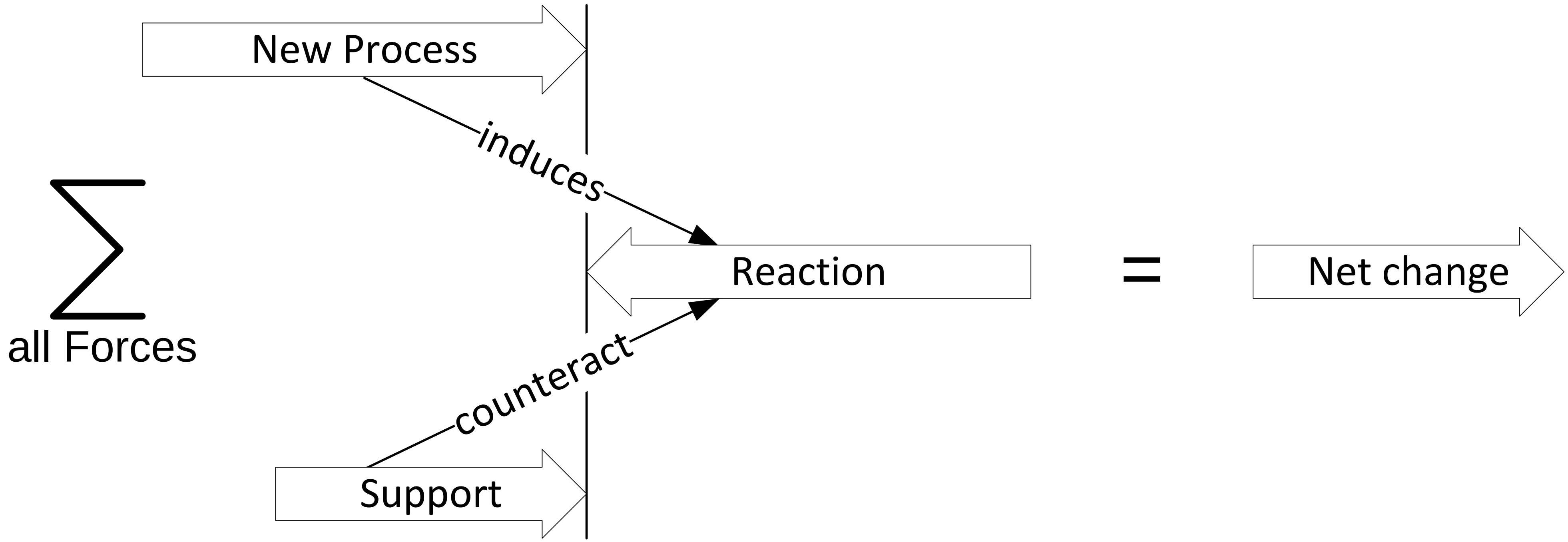
Rationale for Templates

- Low threshold to apply a (new) process (1)
- Low effort to apply a (new) process (2)
- No need to know low level implementation details (3)
- Means to consolidate and reuse experiences (4)

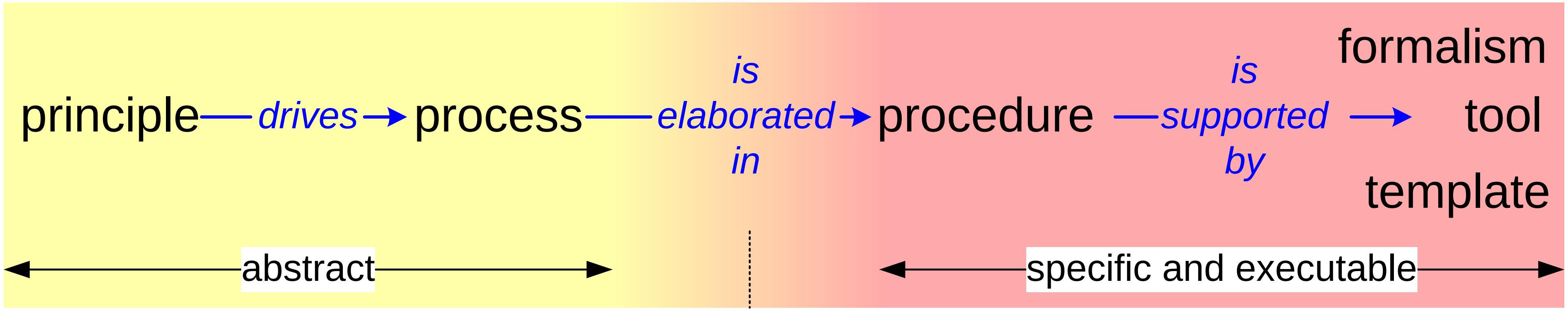
Bogus Arguments for Templates

- Obtain a uniform look (5)
- Force the application of a (new) process (6)
- Control the way a new process is applied (7)

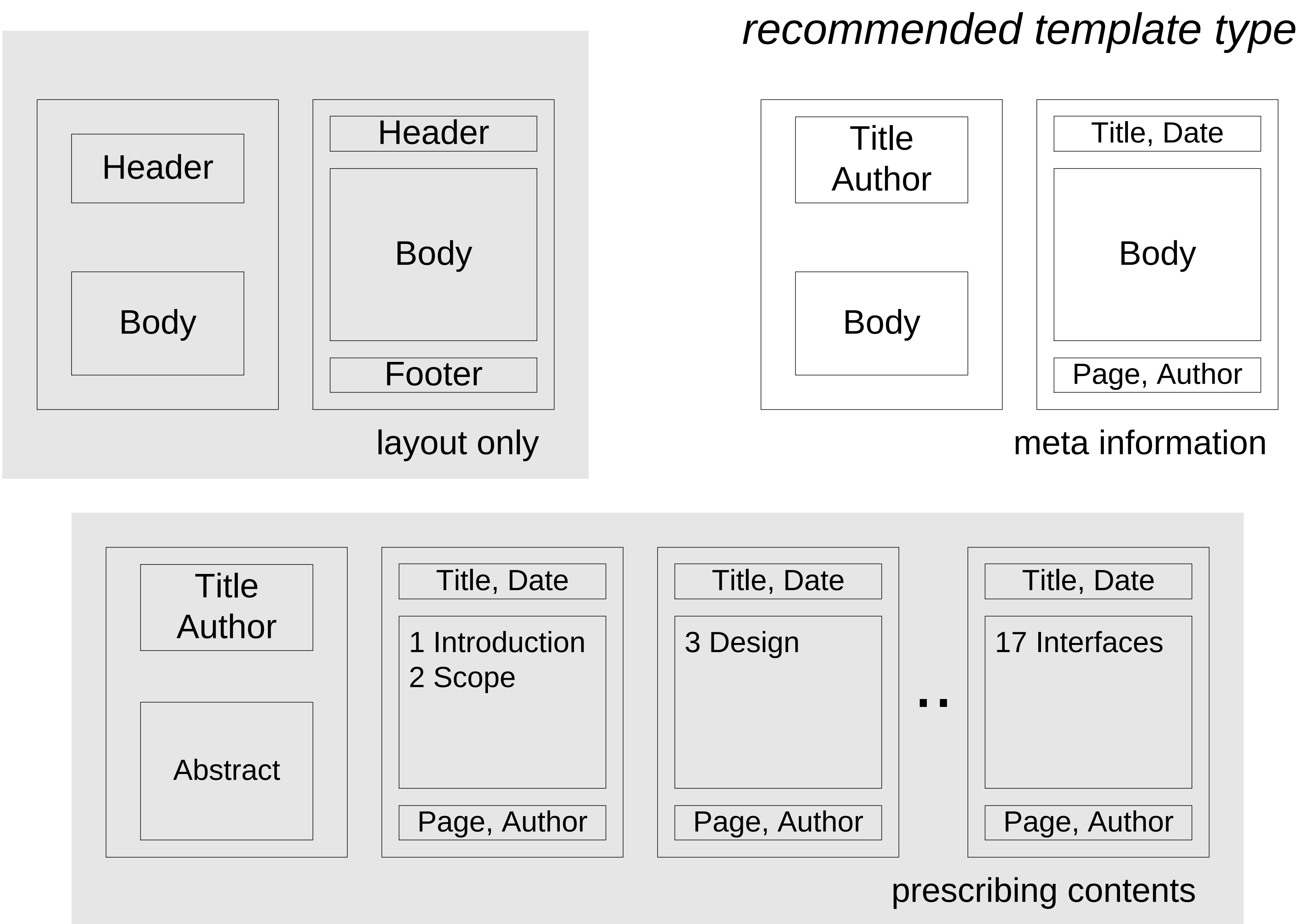
Forces of Change: Action = - Reaction



Template as Support for Process



Types of Templates



Recommendation

template type	context knowhow	value
layout only	no	low
meta information	process	high
prescribing content	process and domain	constraining

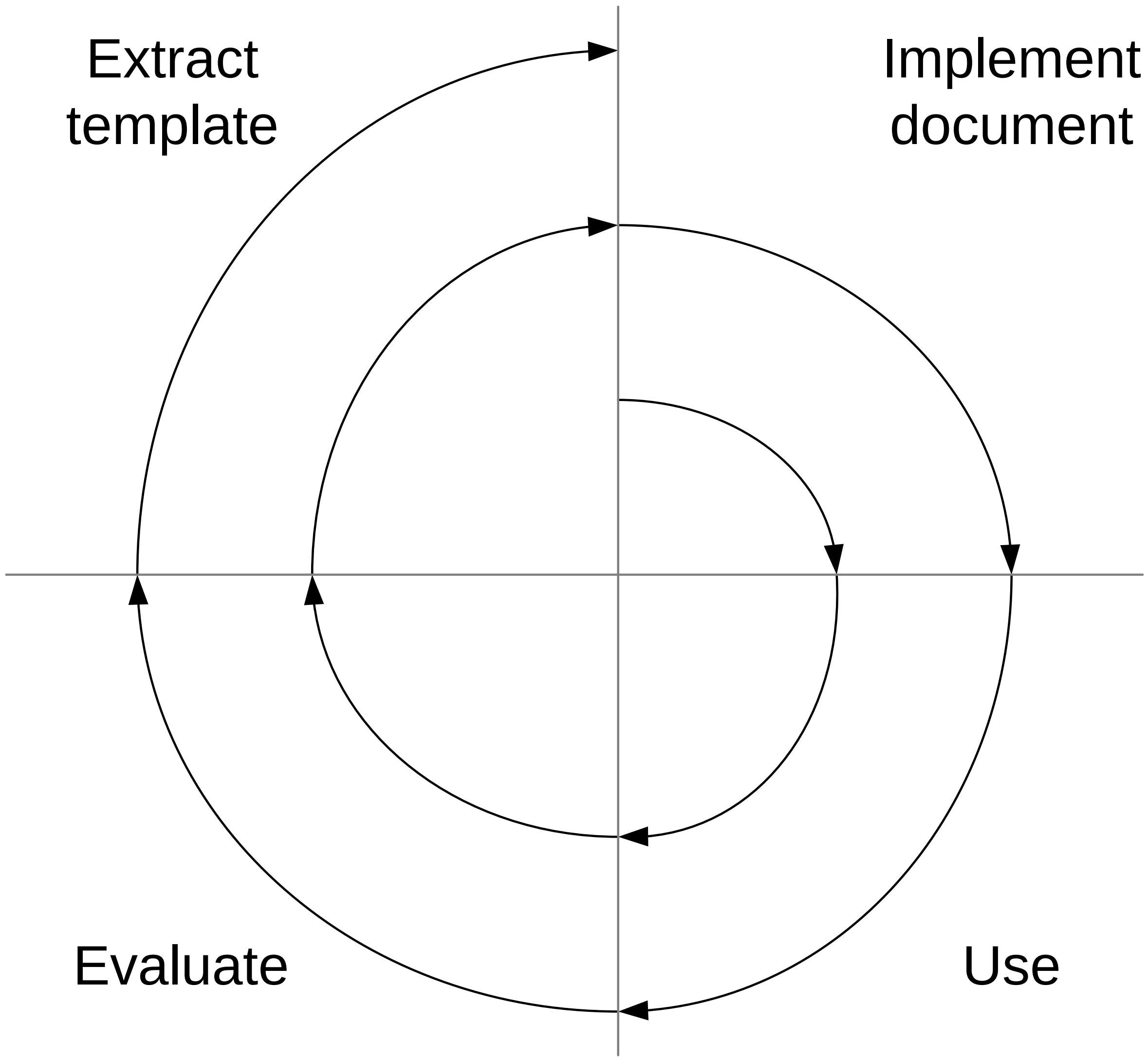
- Use templates for meta-information.
- Use checklists for structure and contents.

Template Development

Templates are an optimization of the Copy Paste Modify pattern:

- Look for a similar problem
- Copy its implementation
- Modify the copy to fulfil the new requirements

Spiral model: Use before Re-use



Example Guidelines Meta Information(1)

Mandatory per page:

- Author
- Title
- Status
- Version
- Date of last update
- Unique Identification
- Business Unit
- Page number

Example Guidelines Meta Information(2)

Mandatory per document:

- Distribution (Notification) list
- Reviewers and commentators
- Document scope (Product family, Product, Subsystem, Module as far as applicable)
- Change history

Example Guidelines Meta Information(3)

Recommended Practice:

- Short statement on frontpage stating what is expected from the addressed recipients, for example:
  - Please send comments before february 29, this document will be reviewed on that date
  - This document is authorized, changes are only applied via a change request
- See Granularity of Documentation [**?**] for guidelines for modularization and contents

Template Pitfalls

- Author follows template instead of considering the purpose of the document.
- Template is too complex.
- There is an unmanageable number of variants.
- Mandatory use of templates results in:
  - no innovation of templates (= no learning)
  - no common sense in deployment
  - strong dependency on templates

Recommendation:

- Enforce the procedure *(what)*
- Provide the template *(how)* as supporting means.

Summary

- Templates support (new) processes
- Use templates for layout and meta information support
- Do not use templates for documents structure or contents
- Stimulate evolution of templates, keep them alive
- Keep templates simple
- Standardize on **what** (process or procedure), not on **how** (tool and template)
- Provide (mandatory) guidelines and recommended practices
- Provide templates as a supportive choice, don't force people to use templates

System Integration How-To

by *Gerrit Muller* USN-SE

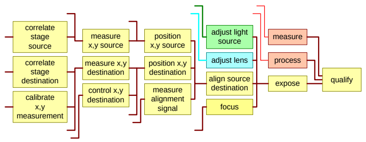
e-mail: gaudisite@gmail.com

www.gaudisite.nl

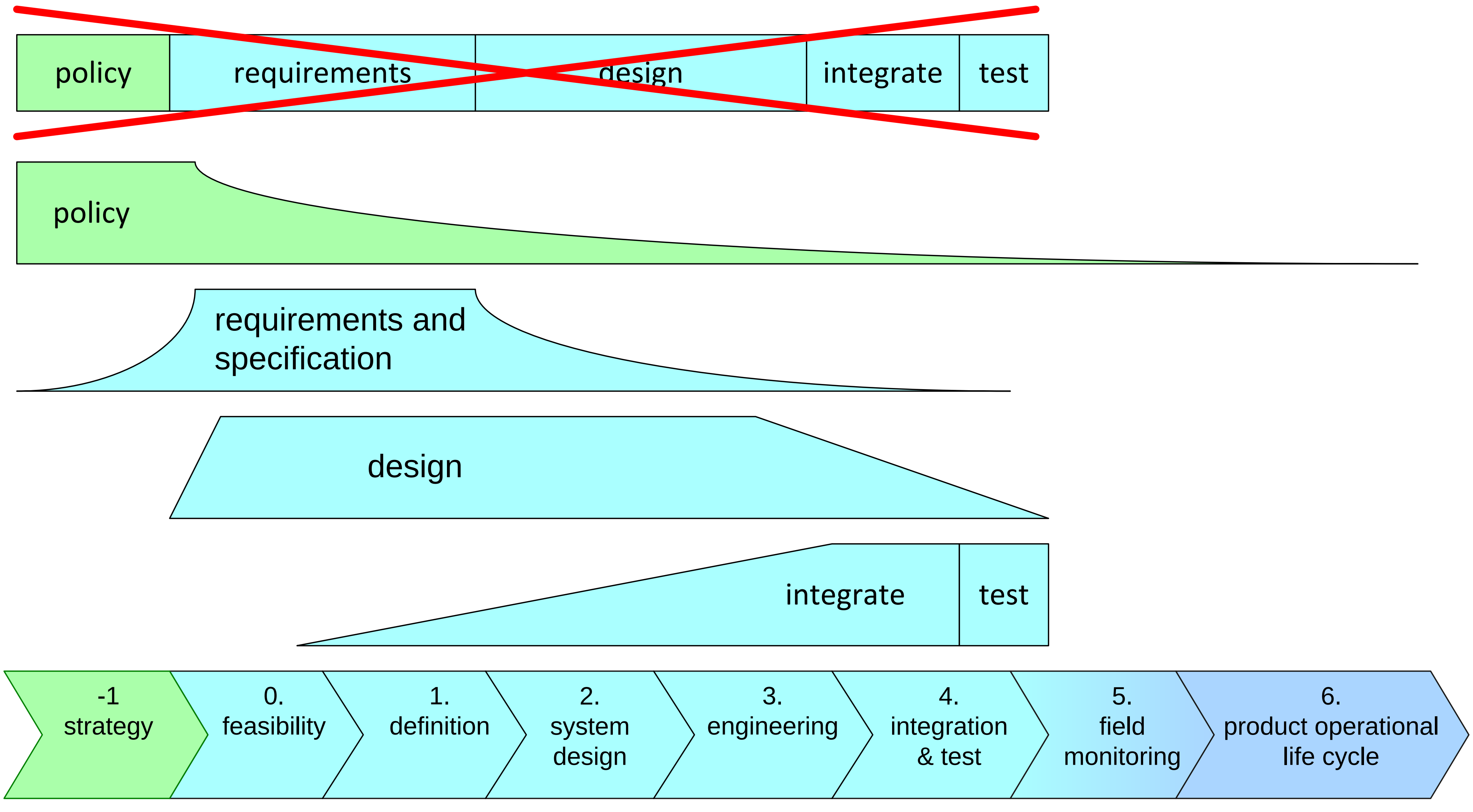
Abstract

In this document we will discuss the full integration flow. We will discuss the goal of integration, the relation between integration and testing, what is integration and how to integrate, an approach to integration, scheduling and dealing with disruptive events, roles and responsibilities, configuration management aspects, and typical order of integration problems occurring in real life.

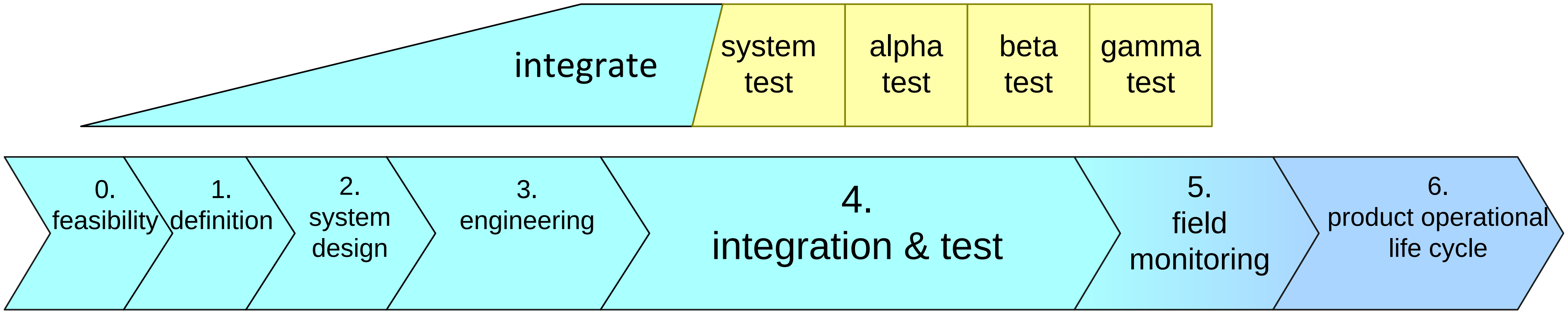
February 15, 2024
status: concept
version: 0.2



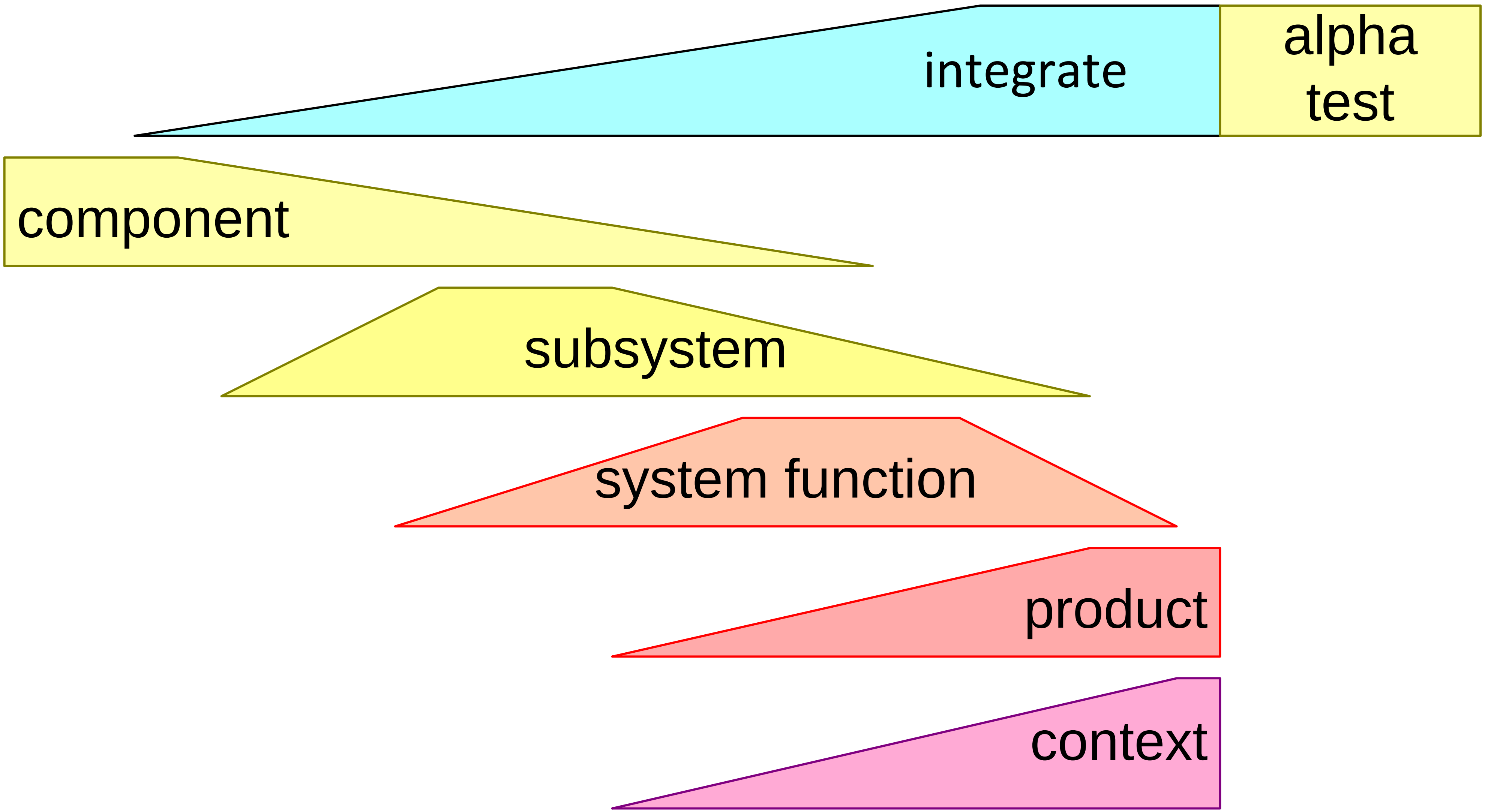
Typical Concurrent Product Creation Process



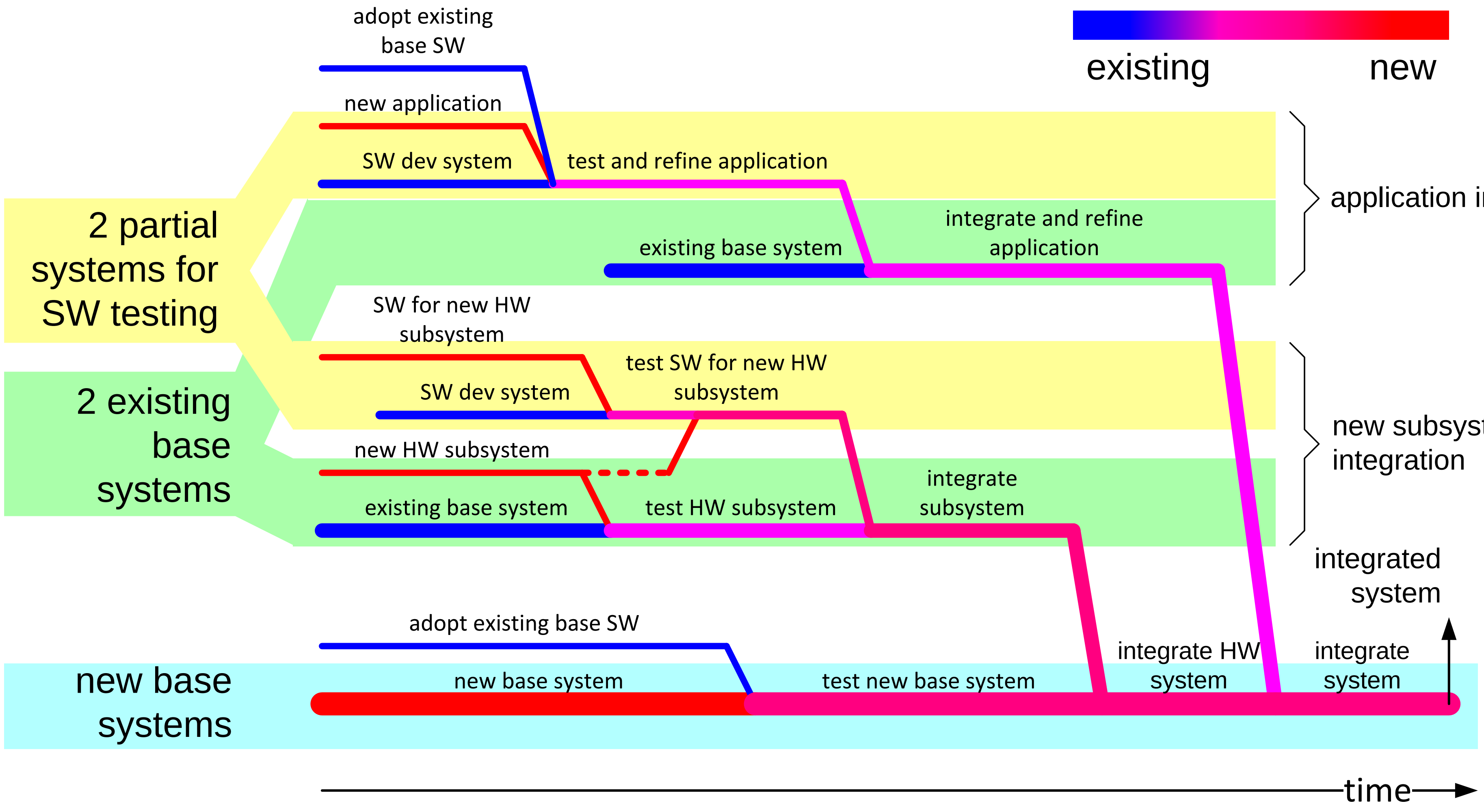
Zooming in on Integration and Tests



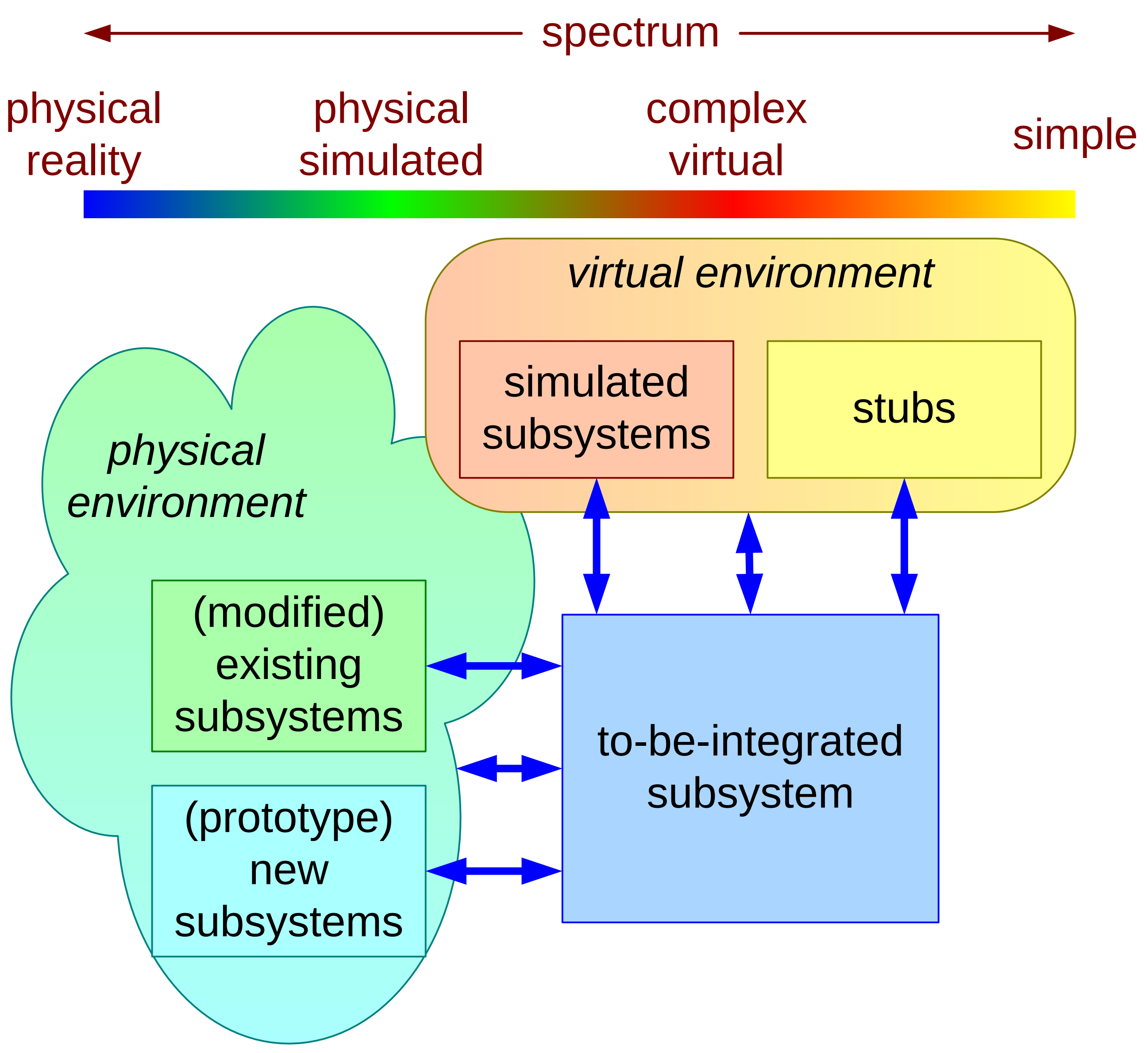
Integration Takes Place in a Bottom-up Fashion



Transition from Previous System to New System



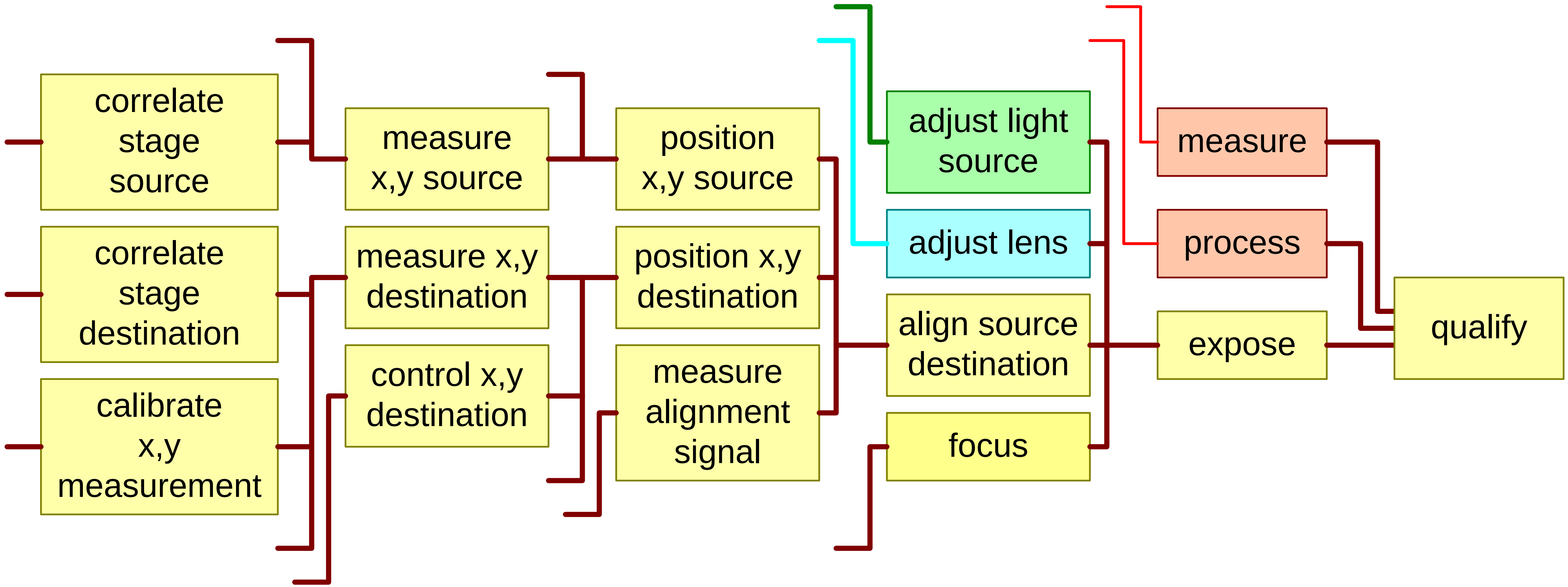
Alternatives to Integrate a Subsystem Early in the Project



Stepwise Integration Approach

1	Determine most critical system performance parameters.
2	Identify subsystems and functions involved in these parameters.
3	Work towards integration configurations along these chains of subsystems and functions.
4	Show system performance parameter as early as possible; start with showing "typical" system performance.
5	Show "worst-case" and "boundary" system performance.
6	Rework manual integration tests in steps into automated regression tests.
7	Monitor regression results with human-driven analysis.
8	Integrate the chains: show system performance of different parameters simultaneously on the same system.

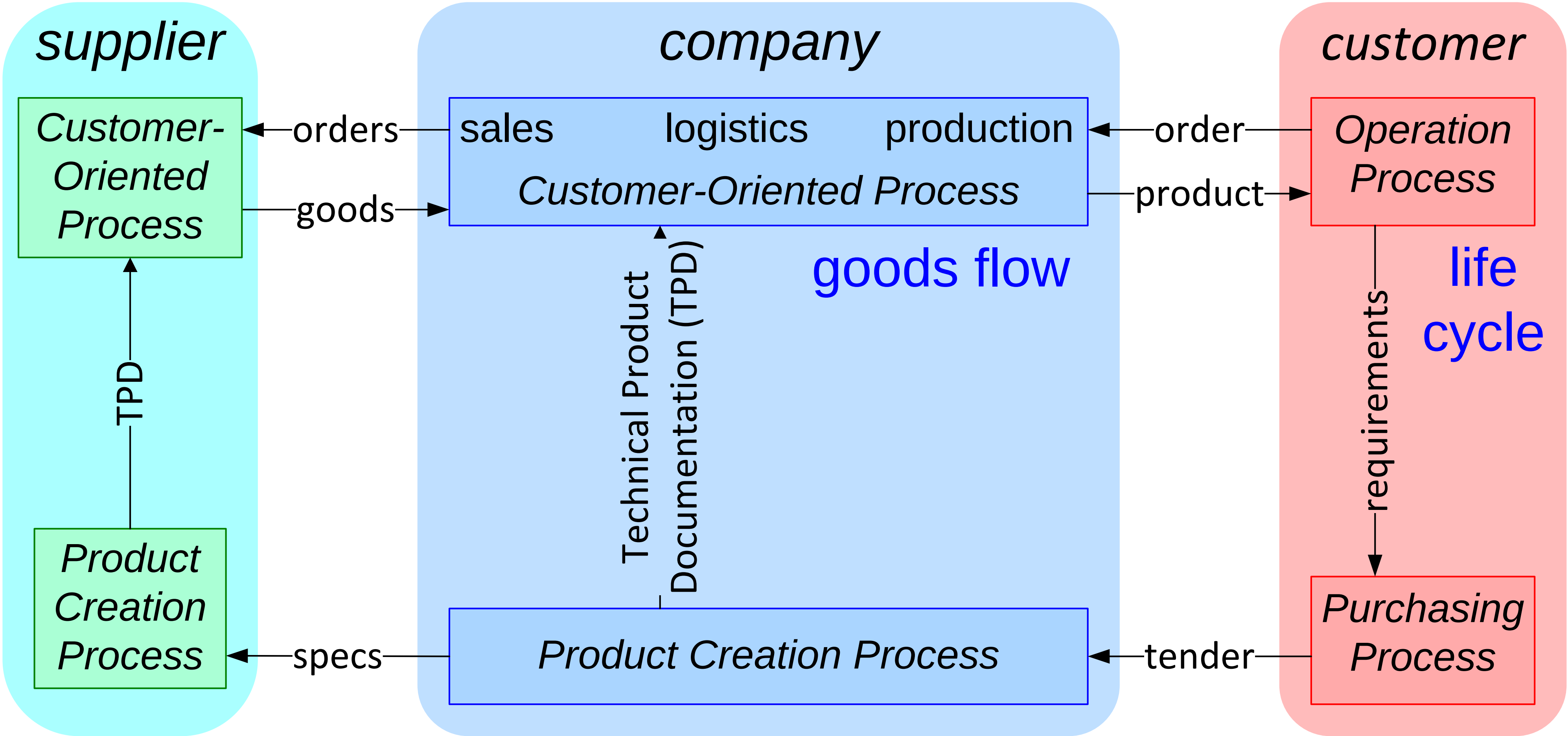
Order of Functions Required for the IQ of a Waferstepper



Roles and Responsibilities During the Integration Process

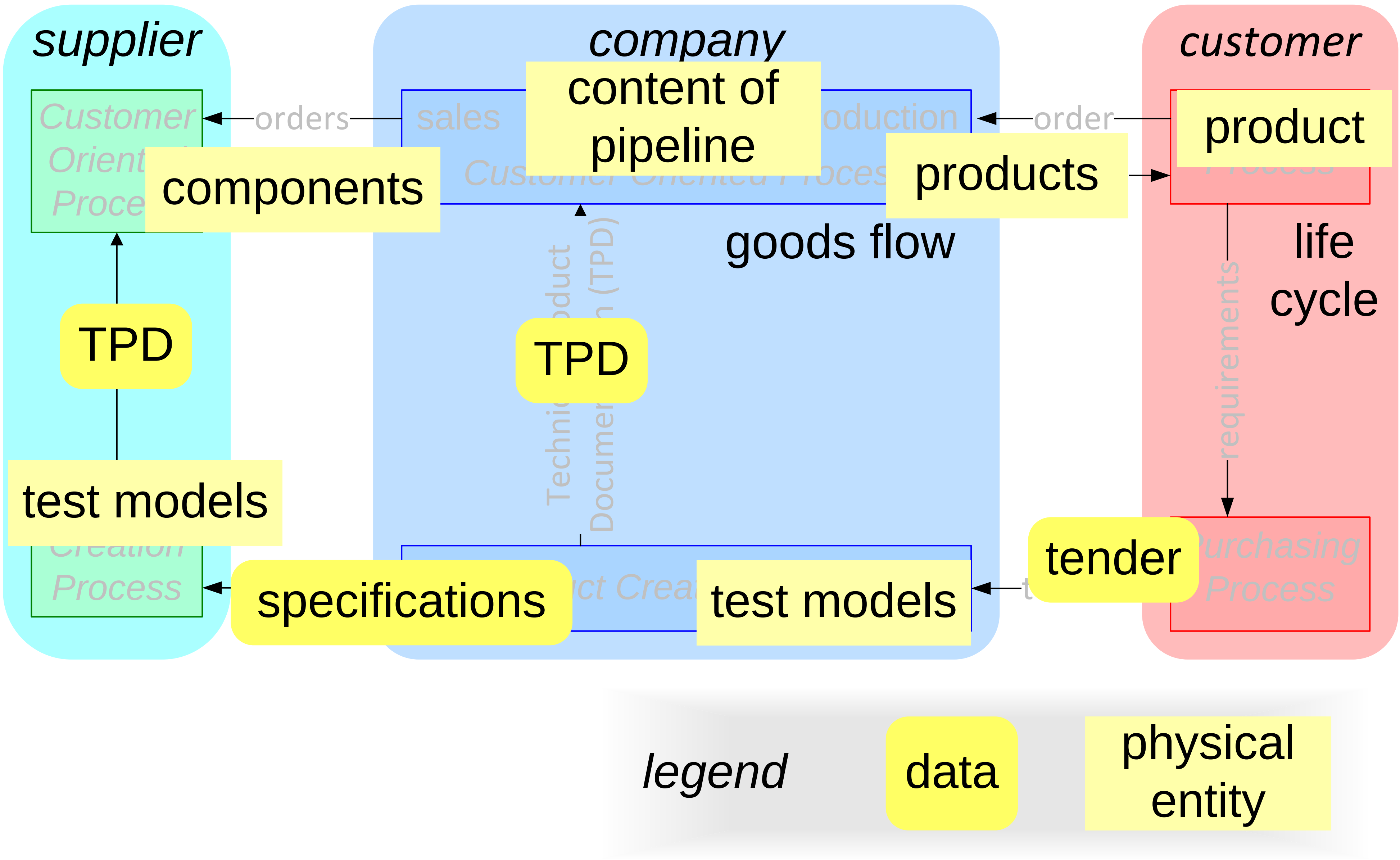
project leader organization resources schedule budget	*systems architect/ engineer/integrator* system requirements design inputs test specification schedule rationale troubleshooting participate in test	*system tester* test troubleshooting report
logistics and administrative support configuration orders administration	*engineers* design component test troubleshooting participate in test	*machine owner* maintain test model support test

Simplified Process Diagram



version: 0.2
February 15, 2024
SINTprocessDecomposition

Configuration Management Entities



Typical Order of Integration Problems

1. The (sub)system does not build.
2. The (sub)system does not function.
3. Interface errors.
4. The (sub)system is too slow.
5. Problems with the main performance parameter, such as image quality.
6. The (sub)system is not reliable.

Exercise Documentation

Make a design for the documentation structure of the case, take into account a.o.:

- target audience per documentation module
- lifecycle
- author
- size (budget)

Present (max 1 flip) the proposed documentation structure and the rationale.

Documentation

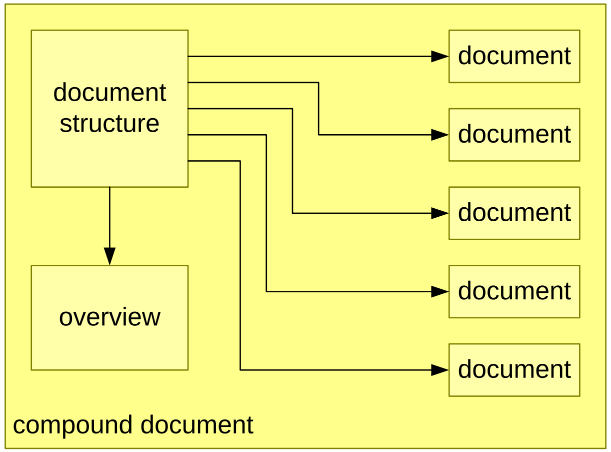
Requirements Entire Documentation

- Accessibility for the readers
- Low threshold for the readers
- Low threshold for the authors
- Completeness
- Consistency
- Maintainability
- Scalability
- Evolvability
- Process to ensure the quality of the information

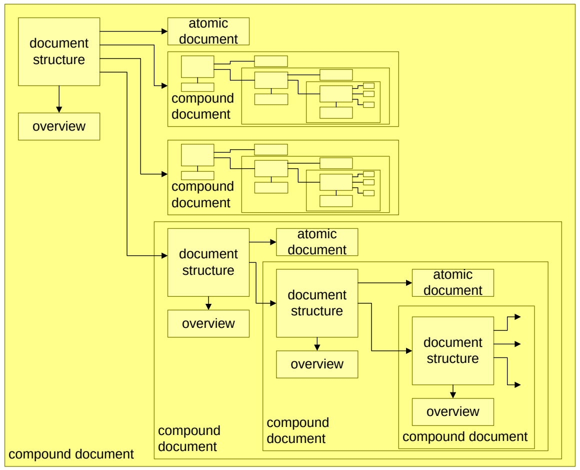
Requirements per Document

- High cohesion (within the unit)
- Low coupling (outside of the unit)
- Accessibility for the readers
- Low threshold for the reader
- Low threshold for the author
- Manageable steps to create, review, and change
- Clear responsibilities
- Clear position and relation with the context
- Well-defined status of the information
- Timely availability

Decompose Large Documents



Recursive Decomposition



Documentation

Maximize Payload

front page

title identification author distribution status review	history changes	diagrams tables	1. aap 2. noot 3. mies lists and ca 50% text

meta information
max 2 pages

contents
2..18 pages

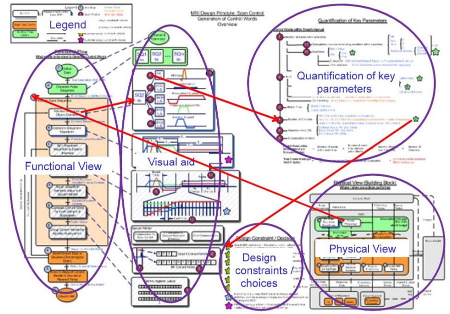
A3s

multiple related views

quantifications

one topic
per A3

capture
"hot" topics



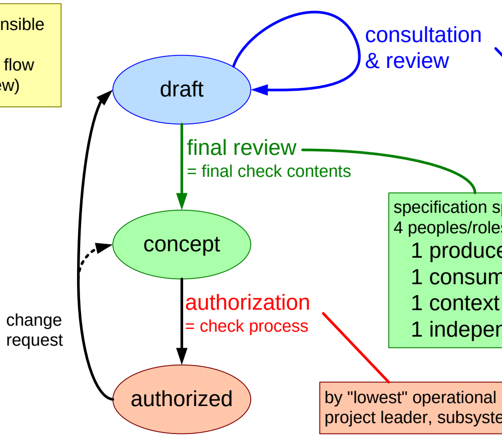
source: PhD thesis Daniel Borches http://doc.utwente.nl/75284/

digestable
(size limitation)

practical
close to stakeholder experience

Light Weight Review

the author is responsible for contents and organization of the flow (consults and review)



- wide group of people, with an active concern or an expected contribution;
- many iterations
- multiple media:
  + meetings,
  + on paper
  + informal et cetera

specification specific Change Control Board
4 peoples/roles:
1 producer
1 consumer
1 context
1 independent

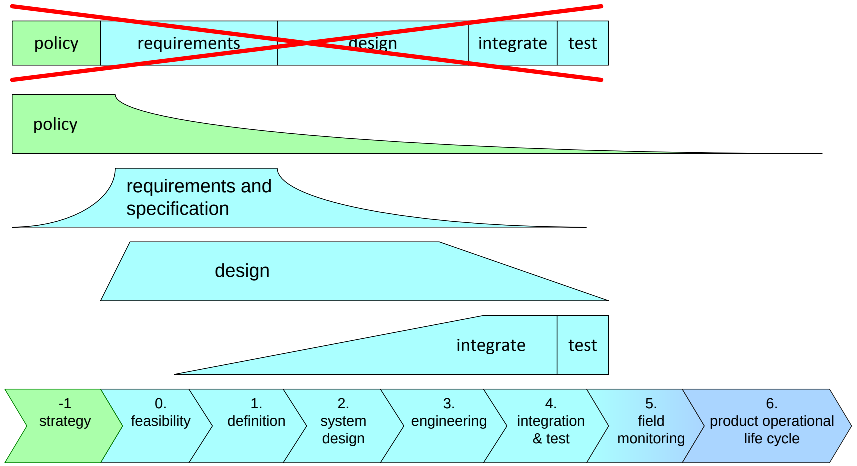
criteria for reviewers:
+ know how
+ critical
+ sufficient time

by "lowest" operational manager:
project leader, subsystem PL, ...

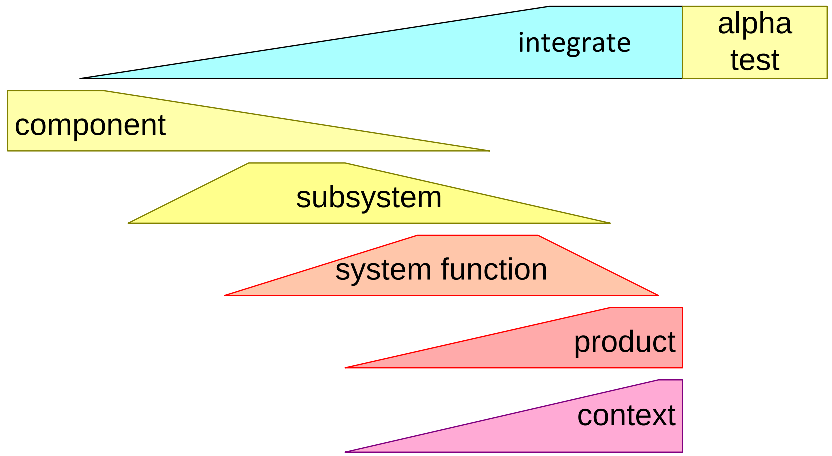
intentionally left blank

Systems Integration

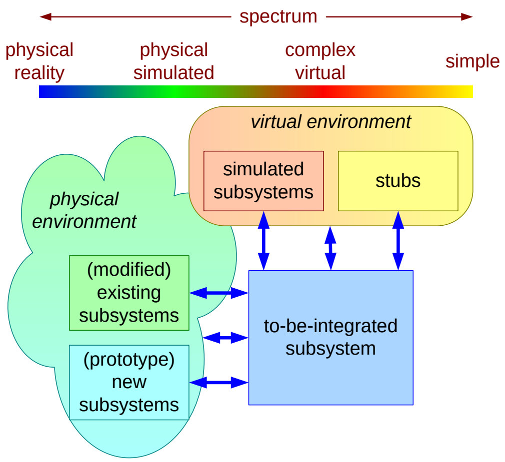
Integration Starts at Feasibility



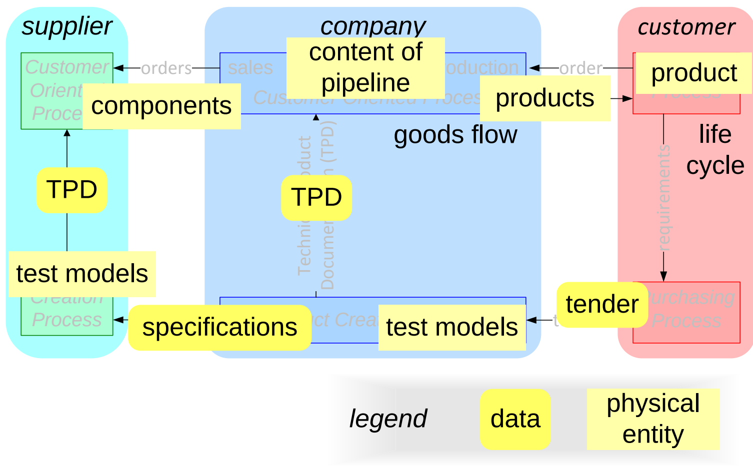
Bottom-up



Alternatives for Early Integration



Propagation of Configuration Issues



Module Role of Software in Complex Systems

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

This module addresses the role of software in complex systems

February 15, 2024
status: preliminary draft
version: 1.0

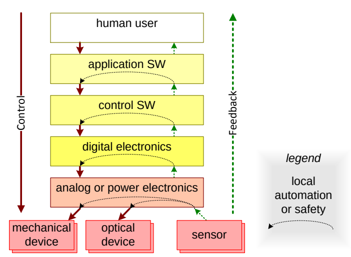
The Role of Software in Systems

by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

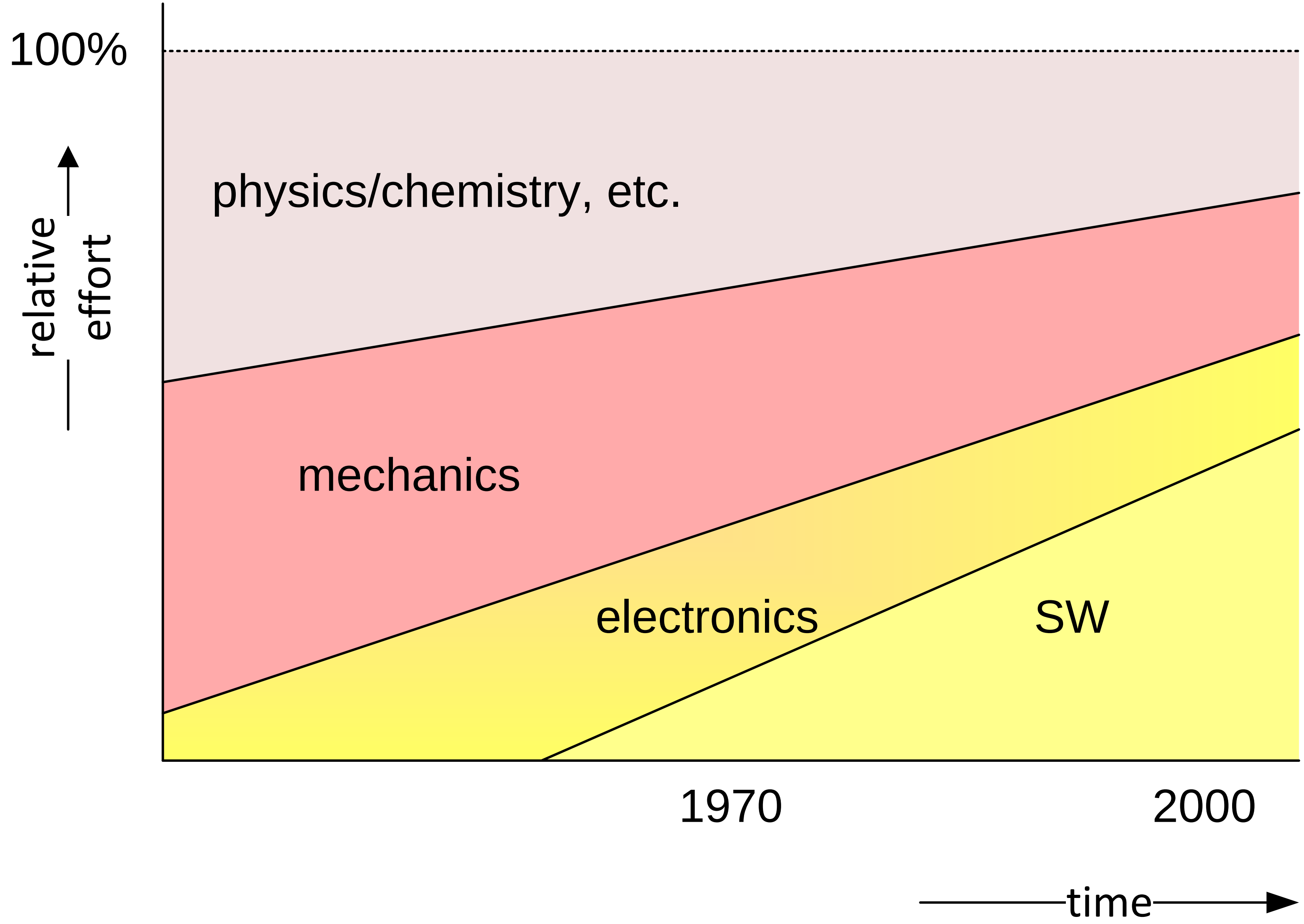
Abstract

Software is a dominating factor in the development of complex systems. It plays a crucial role in the performance of the final product at the one hand, while it contributes significant to the development cost and elapsed time of development. This paper discusses the role of software in the broader system context. An improved understanding of the role of software enables the system architect, and the other stakeholders of the product creation process, to integrate the software development better. In this way hardware-software tradeoffs can be made, balancing performance, costs and risks.

February 15, 2024
status: concept
version: 1.3



Relative Contribution of SW



Mismatch between Role and Discipline

role of software

integration technology

captures *application* functionality

defines lot of *system* behavior

determines how much of potential *system* performance is achieved

acts as director

mismatch!

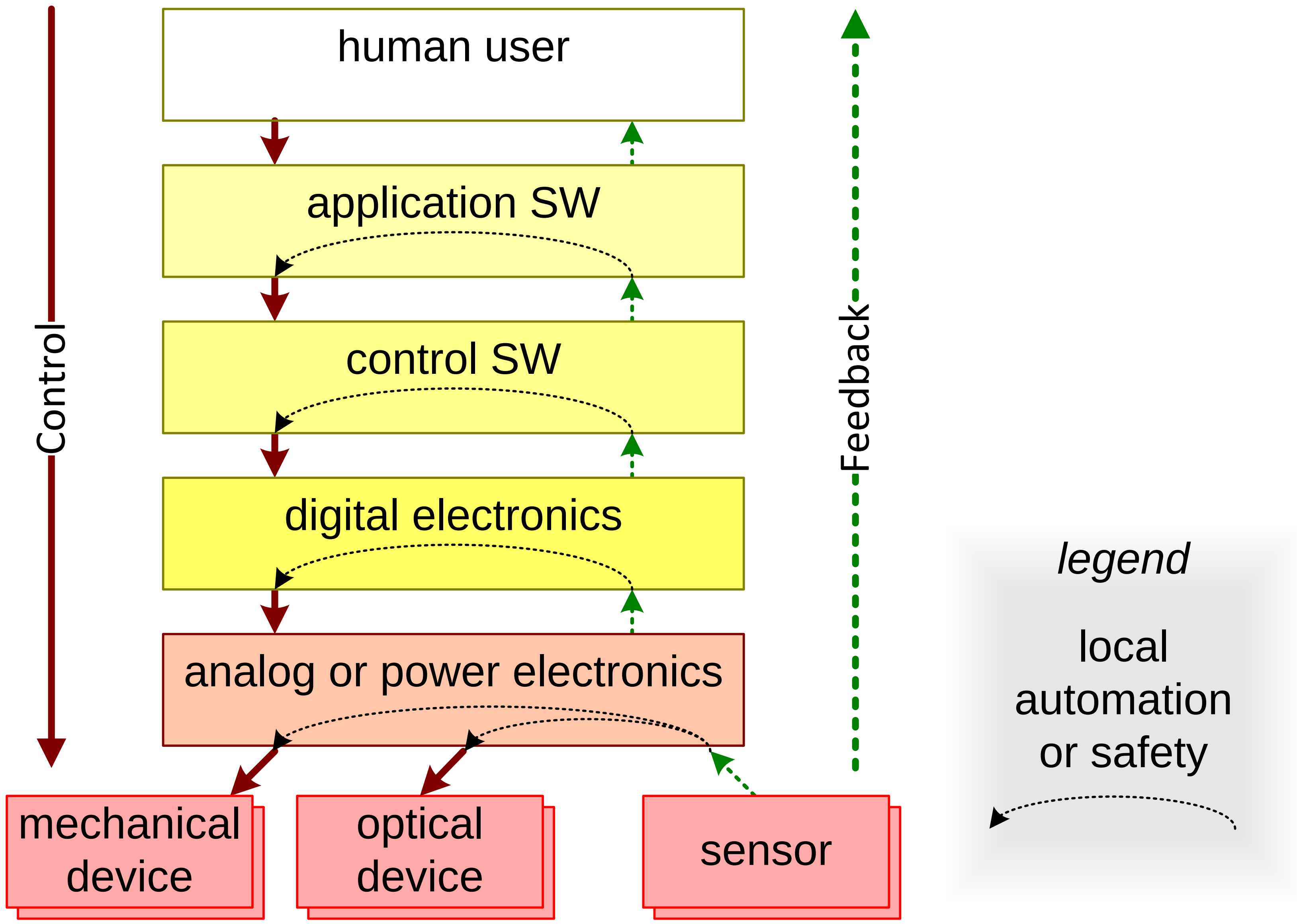
focus of software discipline

software technologies, such as:

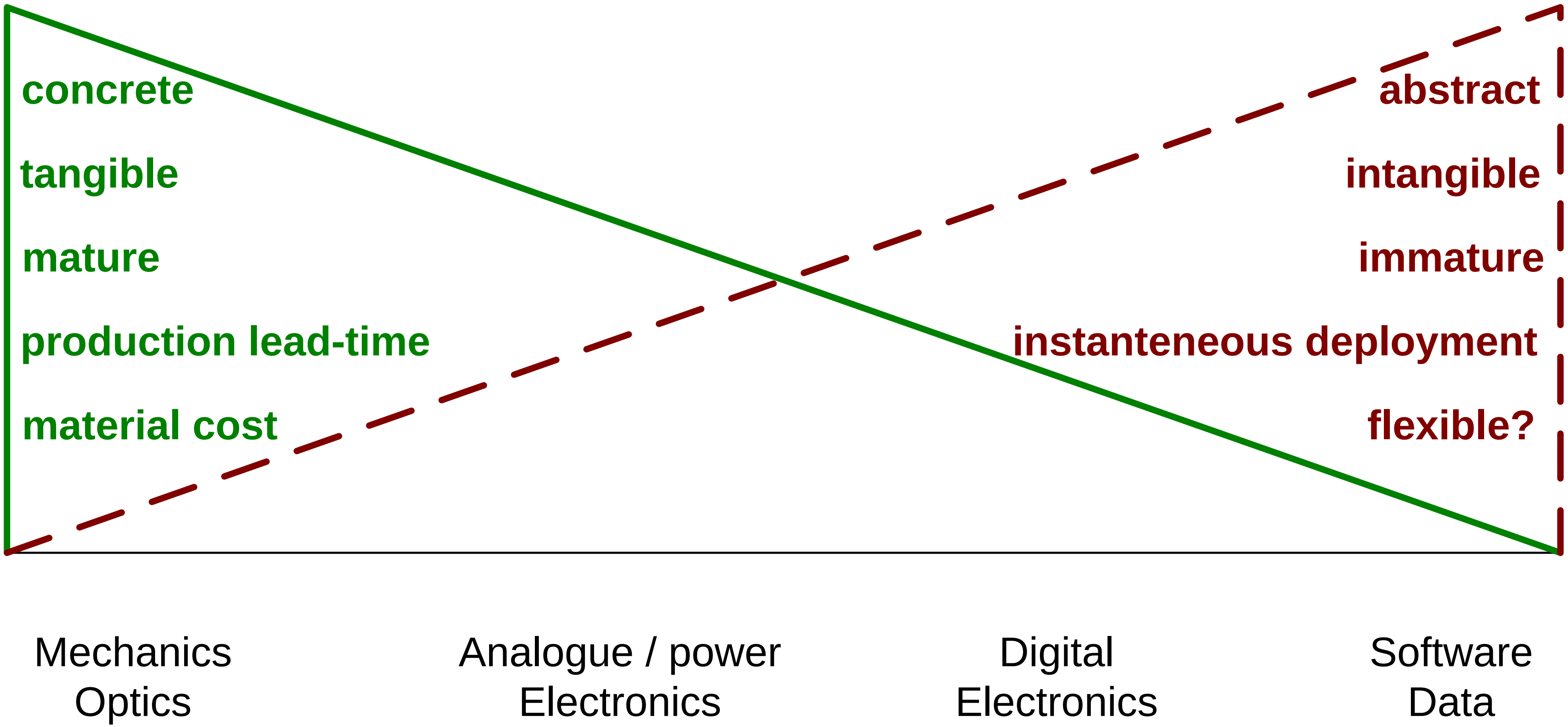
- programming languages
- data bases
- operating systems
- component technologies

engineering practices

Control Hierarchy along Technology axis



Characterization of disciplines



Quality Attributes annotated with SW relation

usable
- usability
- attractiveness
- responsiveness
- image quality
- wearability
- storability
- transportability

dependable
- safety
- security
- reliability
- robustness
- integrity
- availability

effective
- throughput or productivity

interoperable
- connectivity
- 3rd party extendible

liable
- liability
- testability
- traceability
- standards compliance

efficient
- resource utilization
- cost of ownership

consistent
- reproducibility
- predictability

serviceable
- serviceability
- configurability
- installability

future proof
- evolvability
- portability
- upgradability
- extendibility
- maintainability

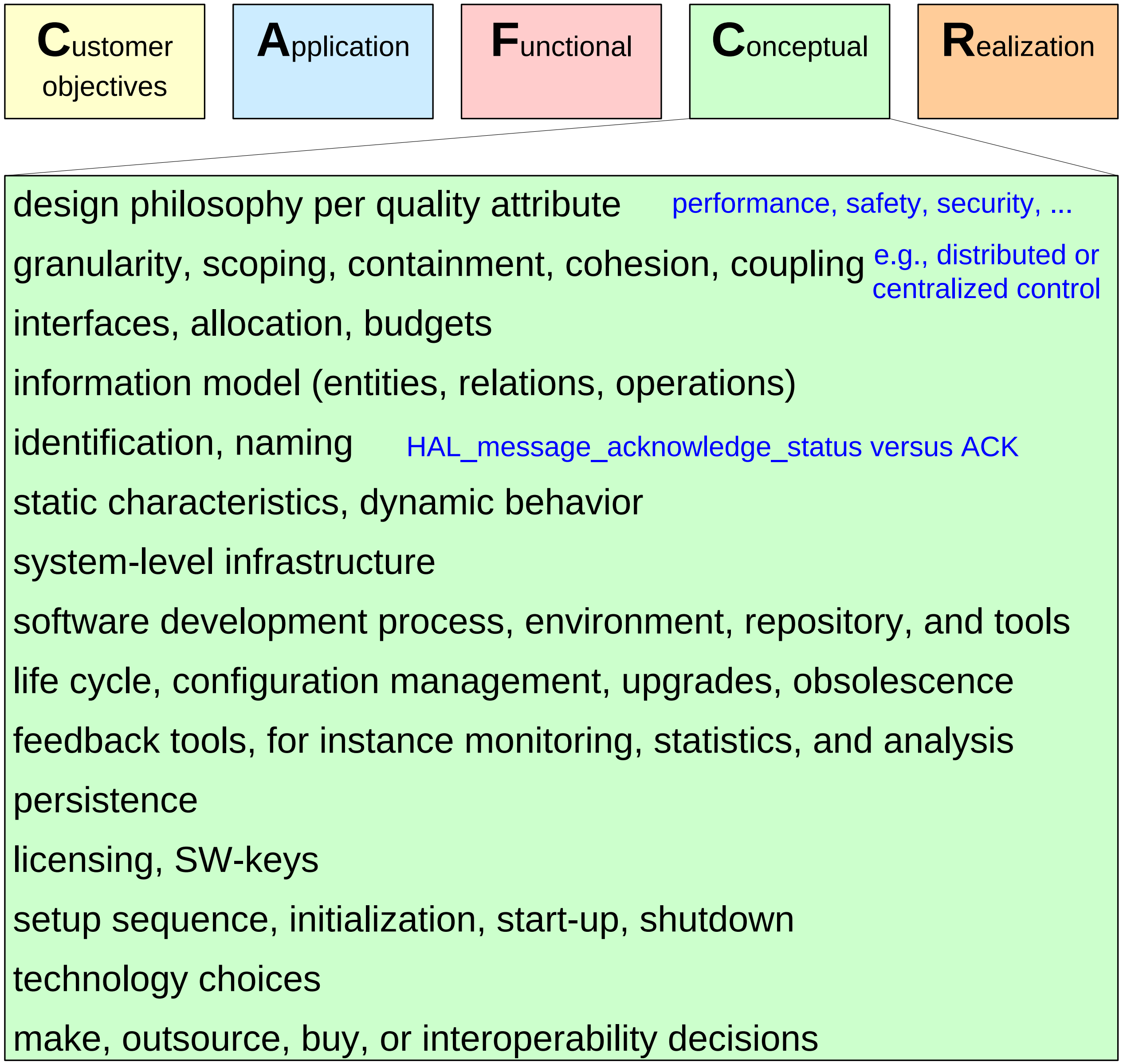
logistics friendly
- manufacturability
- logistics flexibility
- lead-time

ecological
- ecological footprint
- contamination
- noise
- disposability

down-to-earth attributes
- cost price
- power consumption
- consumption rate (water, air, chemicals, etc.)
- size, weight
- accuracy

legend
- weak SW relation
- strong SW relation

Design Aspects related to SW



SW Mechanisms

Customer objectives	Application	Functional	Conceptual	Realization

- error handling, exception handling, logging
- processes, tasks, threads
- configuration management; packages, components, files, objects, modules, interfaces
- automated testing: special methods, harness, suites
- signaling, messaging, callback scheduling, notification, active data, watchdogs, timeouts
- locking, semaphores, transactions, checkpoints, deadlock detection, rollback
- identification, naming, data model, registry, configuration database, inheritance, scoping
- resource management, allocation, fragmentation prevention, garbage collection
- persistence, caching, versioning, prefetching, lazy evaluation
- licensing, SW-keys
- bootstrap, discovery, negotiation, introspection
- call graphs, message tracing, object tracing, etc.
- distribution, allocation, transparency; component, client/server, multitier model

Exercise Role of Software in a complex product

Describe the SW in a complex product, from different viewpoints for instance:

- Give an indication of the size/complexity
- Outline the SW architecture
- Identify the top 3 critical characteristics
- Identify potential improvements
- Process
- Development environment

Software

Role of Software

role of software

integration technology
captures *application* functionality
defines lot of *system* behavior
determines how much of potential *system* performance is achieved
acts as director

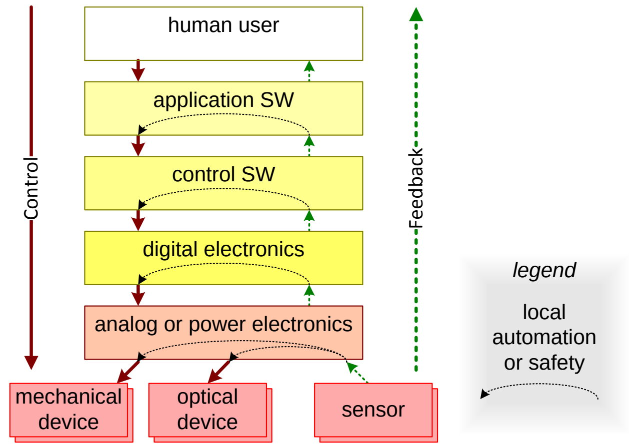
mismatch!

focus of software discipline

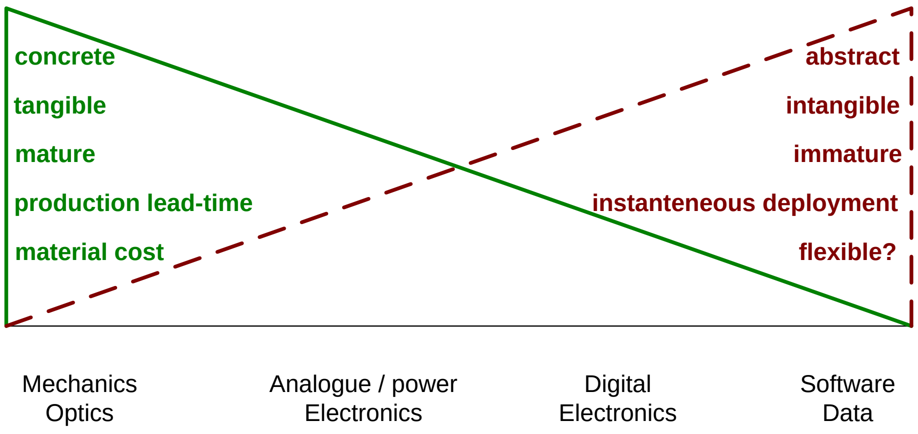
software technologies, such as:
- programming languages
- data bases
- operating systems
- component technologies

engineering practices

Control Hierarchy



Discipline Characteristics



intentionally left blank

Module Management Presentation

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

This module addresses the presentation of architectural issues to higher management teams.

February 15, 2024
status: draft
version: 1.1

Simplistic Financial Computations for System Architects.

by *Gerrit Muller* USN-SE

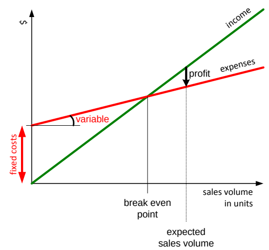
e-mail: gaudisite@gmail.com

www.gaudisite.nl

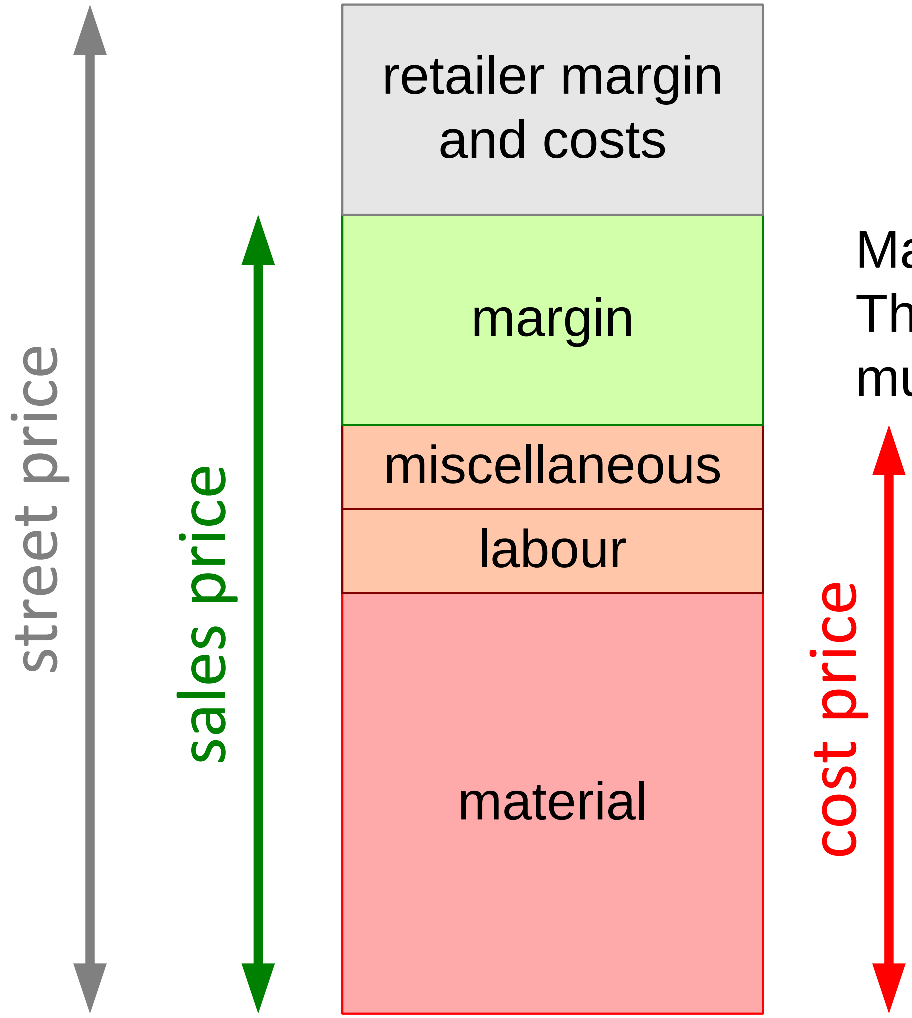
Abstract

This document explains how simple financial estimates can be made by system architects. These simplistic estimates are useful for an architect to perform sanity checks on proposals and to obtain understanding of the financial impact of proposals. Note that architects will never have full fledged financial controller know how and skills. These estimates are zero order models, but real business decisions will have to be founded on more substantial financial proposals.

February 15, 2024
status: preliminary draft
version: 1.3



Product Margin = Sales Price - Cost



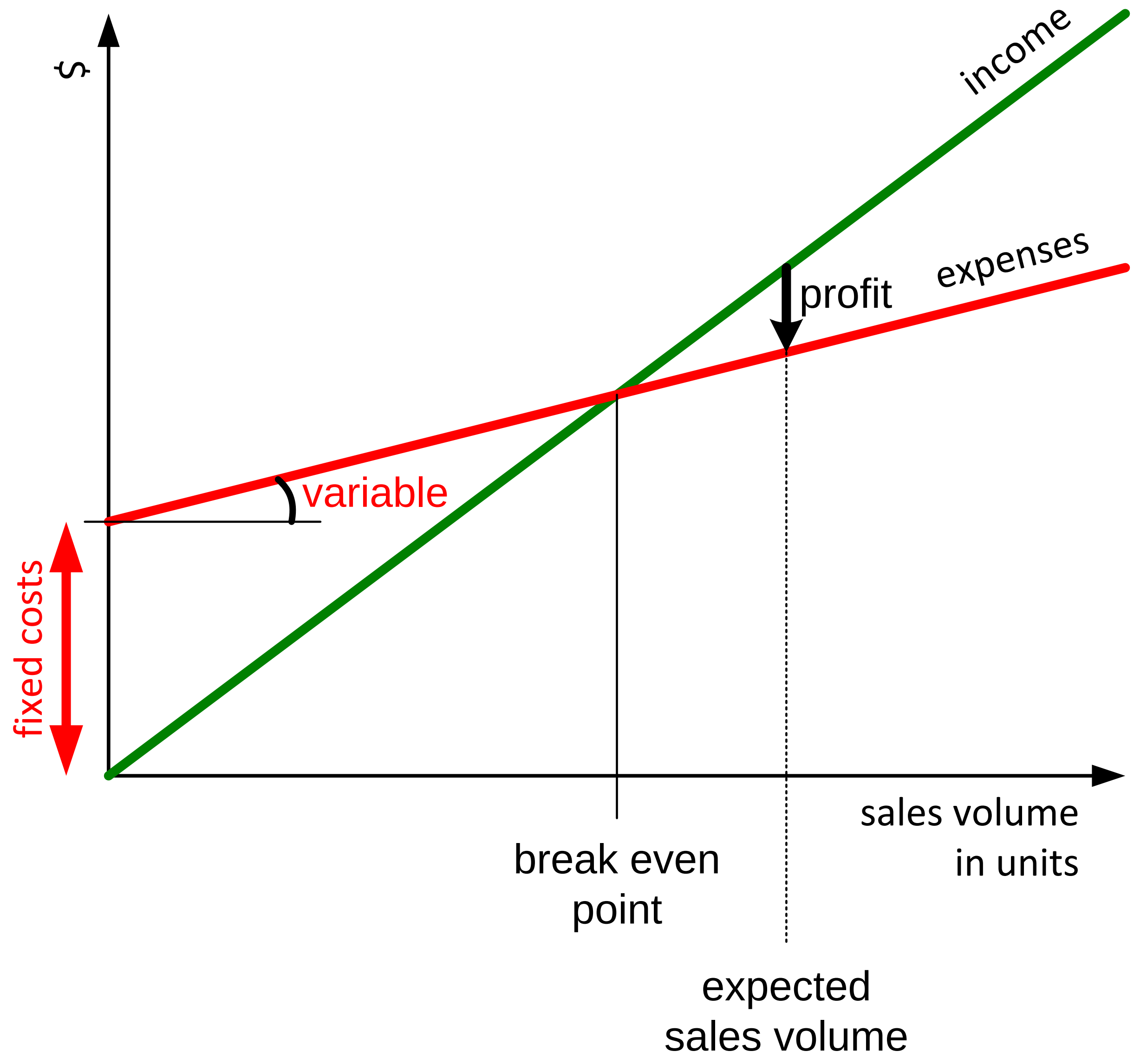
Margin per product.
The margin over the sales volume,
must cover the fixed costs, and generate profit

transportation, insurance,
royalties per product, ...

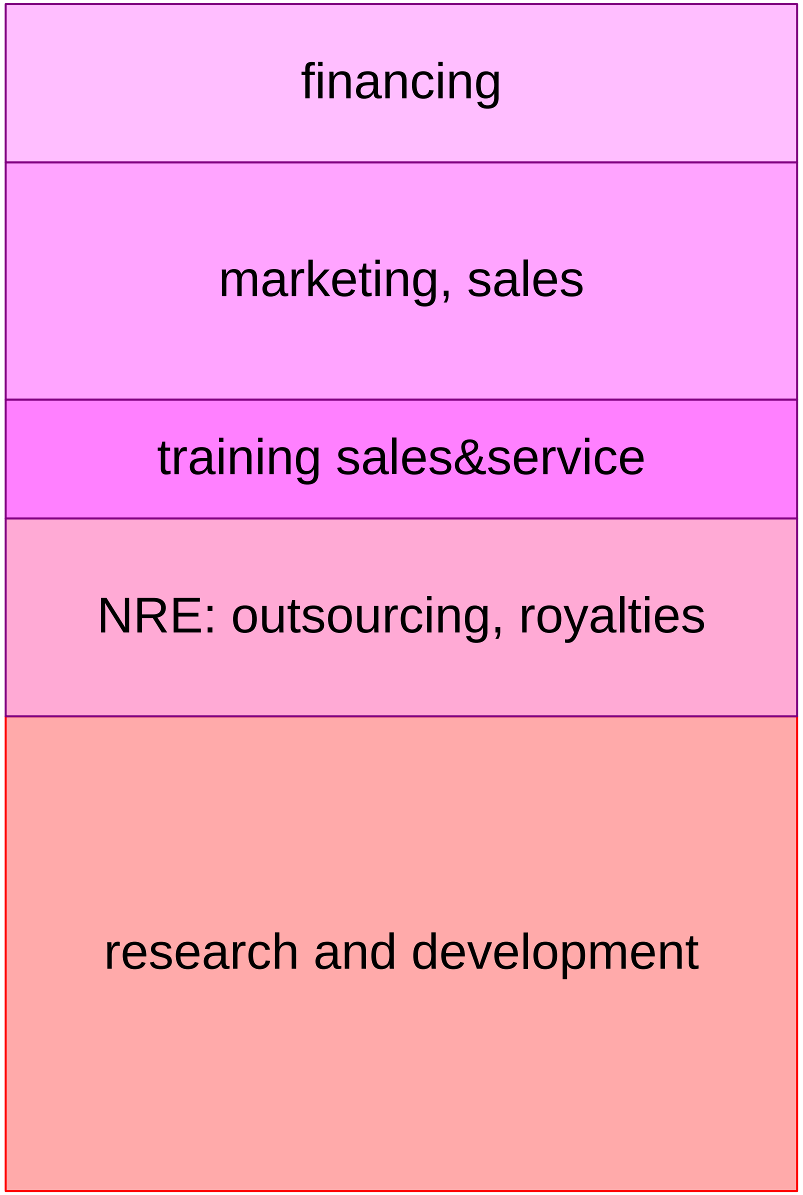
Cost per product,
excluding fixed costs

purchase price of components may cover
development cost of supplier

Profit as function of sales volume



Investments, more than R&D



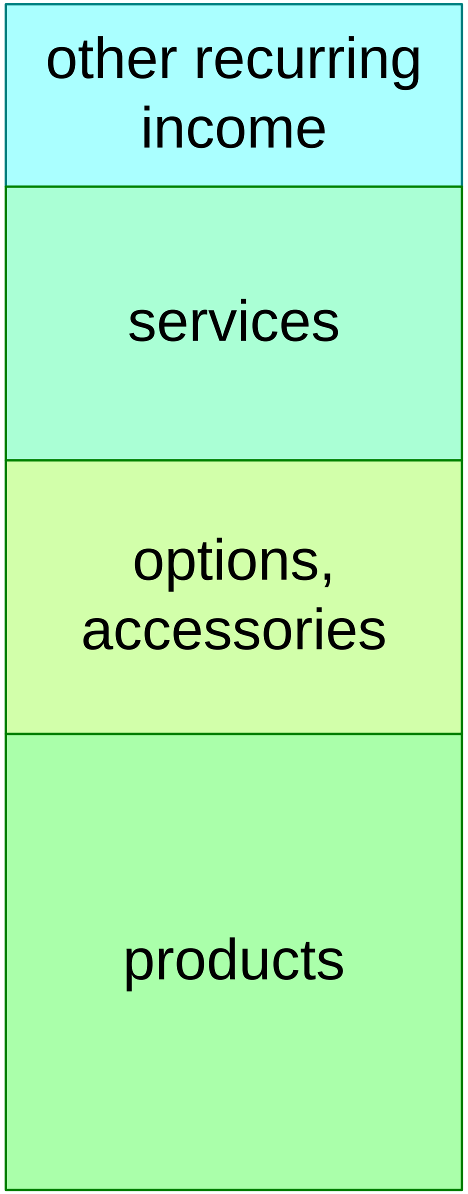
business dependent:
pharmaceutics industry
sales cost >> R&D cost

strategic choice:
NRE or per product

including:
- staff, training, tools, housing
- materials, prototypes
- overhead
- certification

often a standard staffing rate is used
that covers most costs above:
R&D investment = Effort * rate

Income, more than product sales only



license fees
pay per movie

content, portal
updates
maintenance

$$\sum_{services} income_{service}$$

$$\sum_{options} \text{sales price}_{option} * volume_{option}$$

$$\text{sales price}_{product} * volume_{product}$$

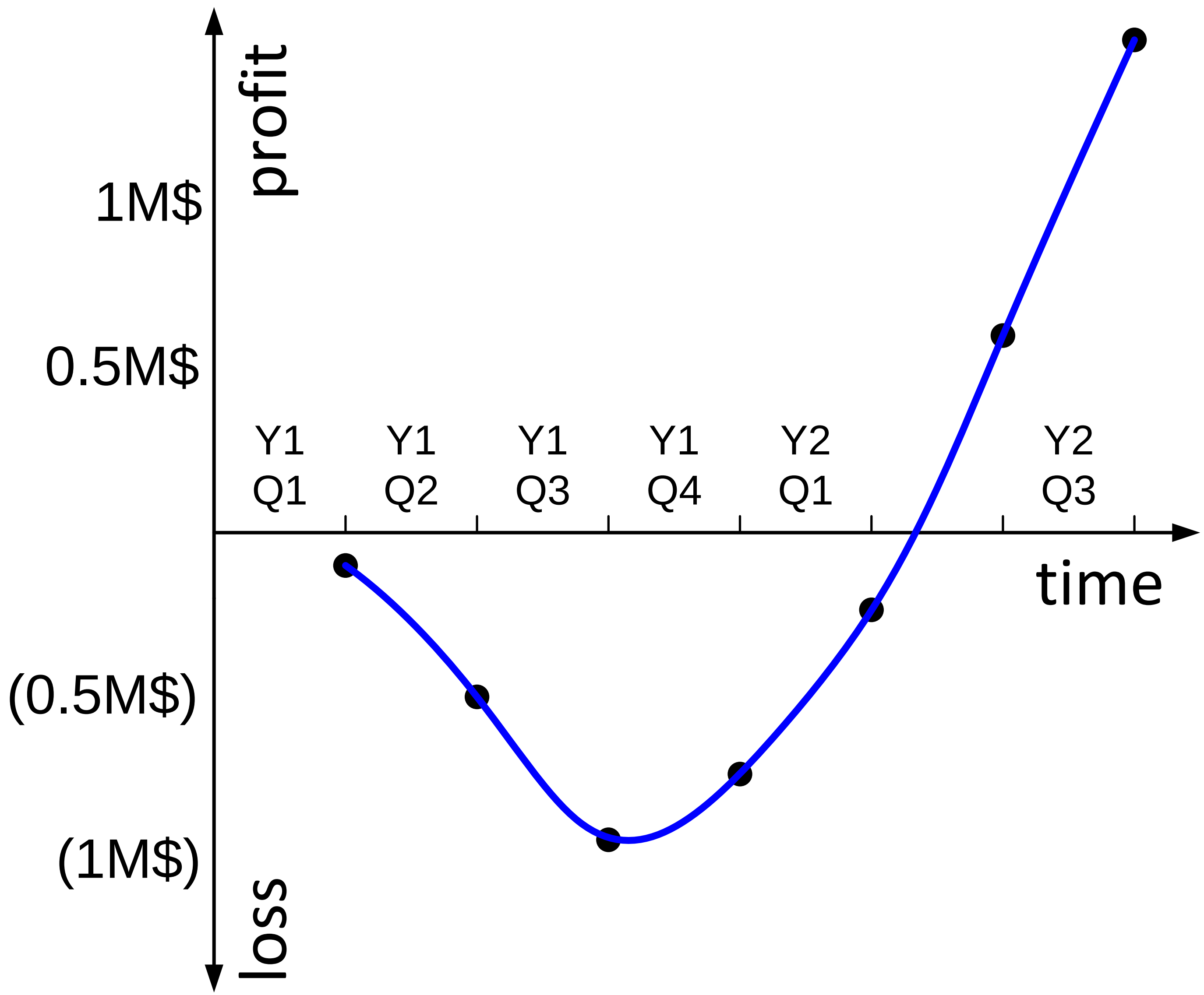
The Time Dimension

	Y1 Q1	Y1 Q2	Y1 Q3	Y1 Q4	Y2 Q1	Y2 Q2	Y2 Q3
investments	100k$	400k$	500k$	100k$	100k$	60k$	20k$
sales volume (units)	-	-	2	10	20	30	30
material & labour costs	-	-	40k$	200k$	400k$	600k$	600k$
income	-	-	100k$	500k$	1000k$	1500k$	1500k$
quarter profit (loss)	(100k$)	(400k$)	(440k$)	200k$	500k$	840k$	880k$
cumulative profit	(100k$)	(500k$)	(940k$)	(740k$)	(240k$)	600k$	1480k$

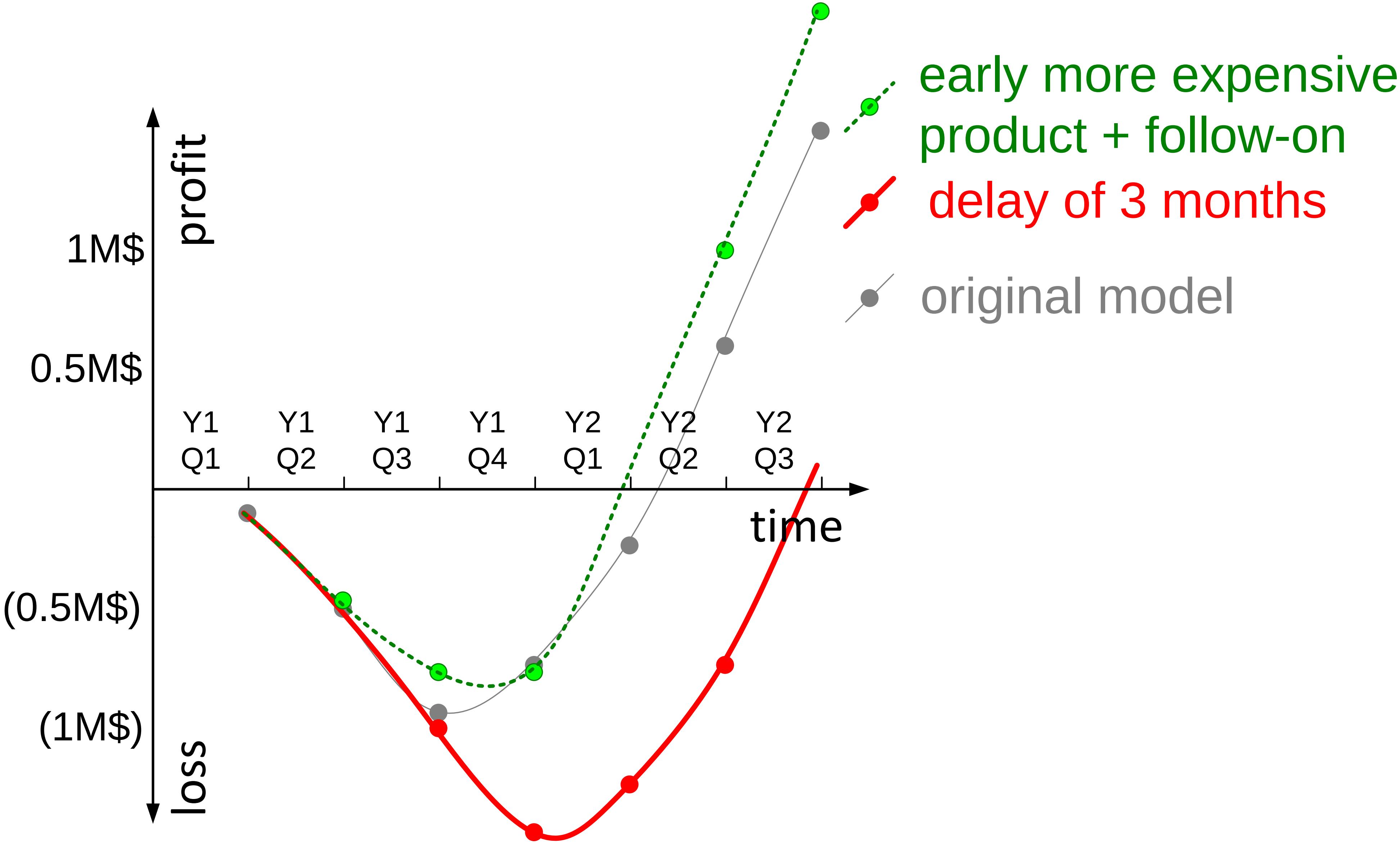
cost price / unit = 20k$
sales price / unit = 50k$

variable cost = sales volume * cost price / unit
income = sales volume * sales price / unit
quarter profit = income - (investments + variable costs)

The "Hockey" Stick

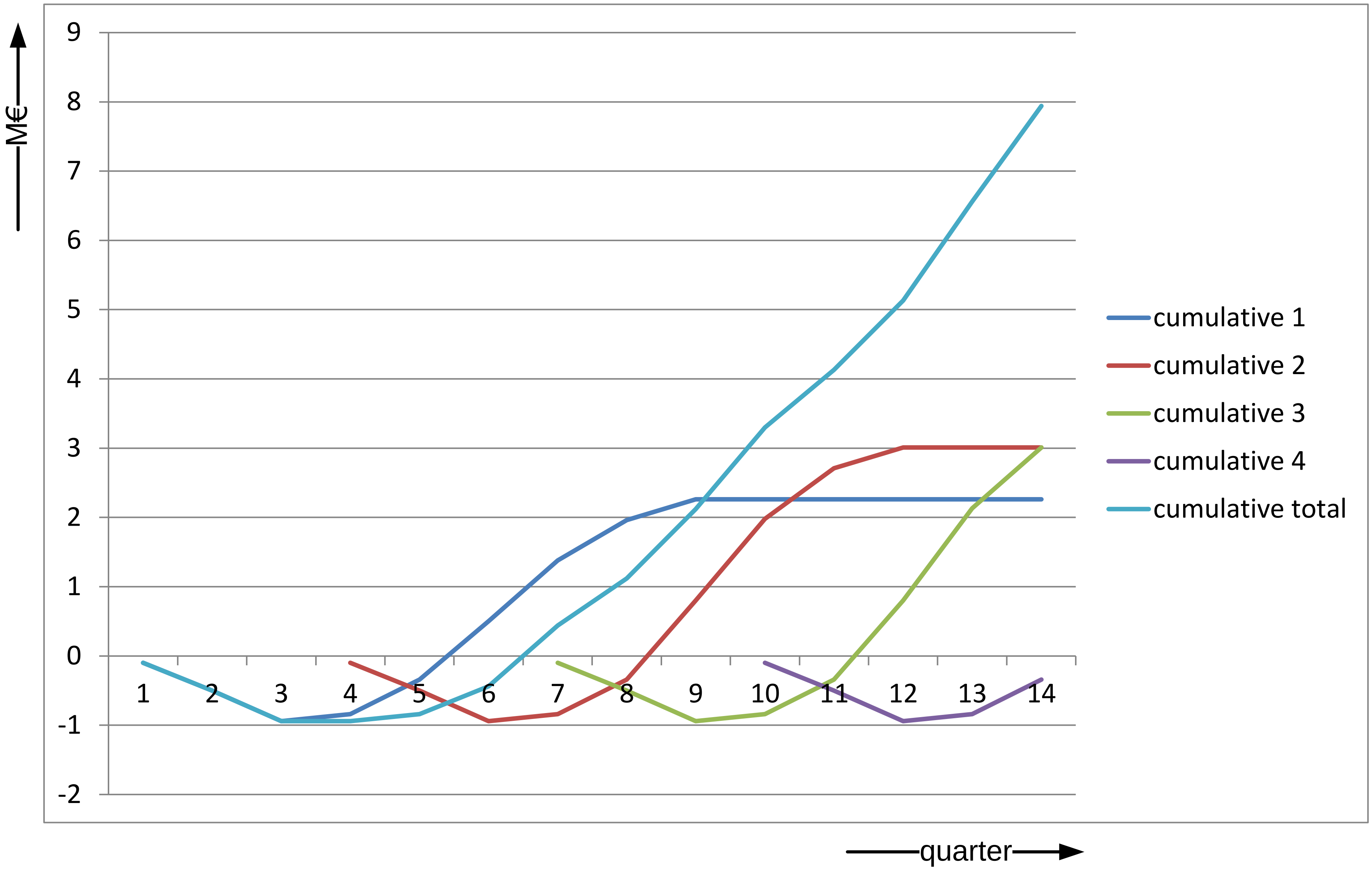


What if ...?



early more expensive product + follow-on
delay of 3 months
original model
profit
loss
time
1M$
0.5M$
(0.5M$)
(1M$)
Y1 Q1
Y1 Q2
Y1 Q3
Y1 Q4
Y2 Q1
Y2 Q2
Y2 Q3

Stacking Multiple Developments



Fashionable financial yardsticks

Return On Investments (ROI)

Net Present Value

Return On Net Assets (RONA) leasing reduces assets, improves RONA

turnover / fte outsourcing reduces headcount, improves this ratio

market ranking (share, growth) "only numbers 1, 2 and 3 will be profitable"

R&D investment / sales in high tech segments 10% or more

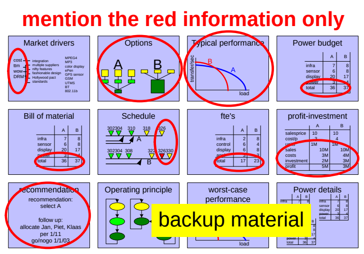
cash-flow fast growing companies combine profits with negative cash-flow, risk of bankruptcy

How to present architecture issues to higher management

by *Gerrit Muller* USN-SE
e-mail: `gaudisite@gmail.com`
`www.gaudisite.nl`

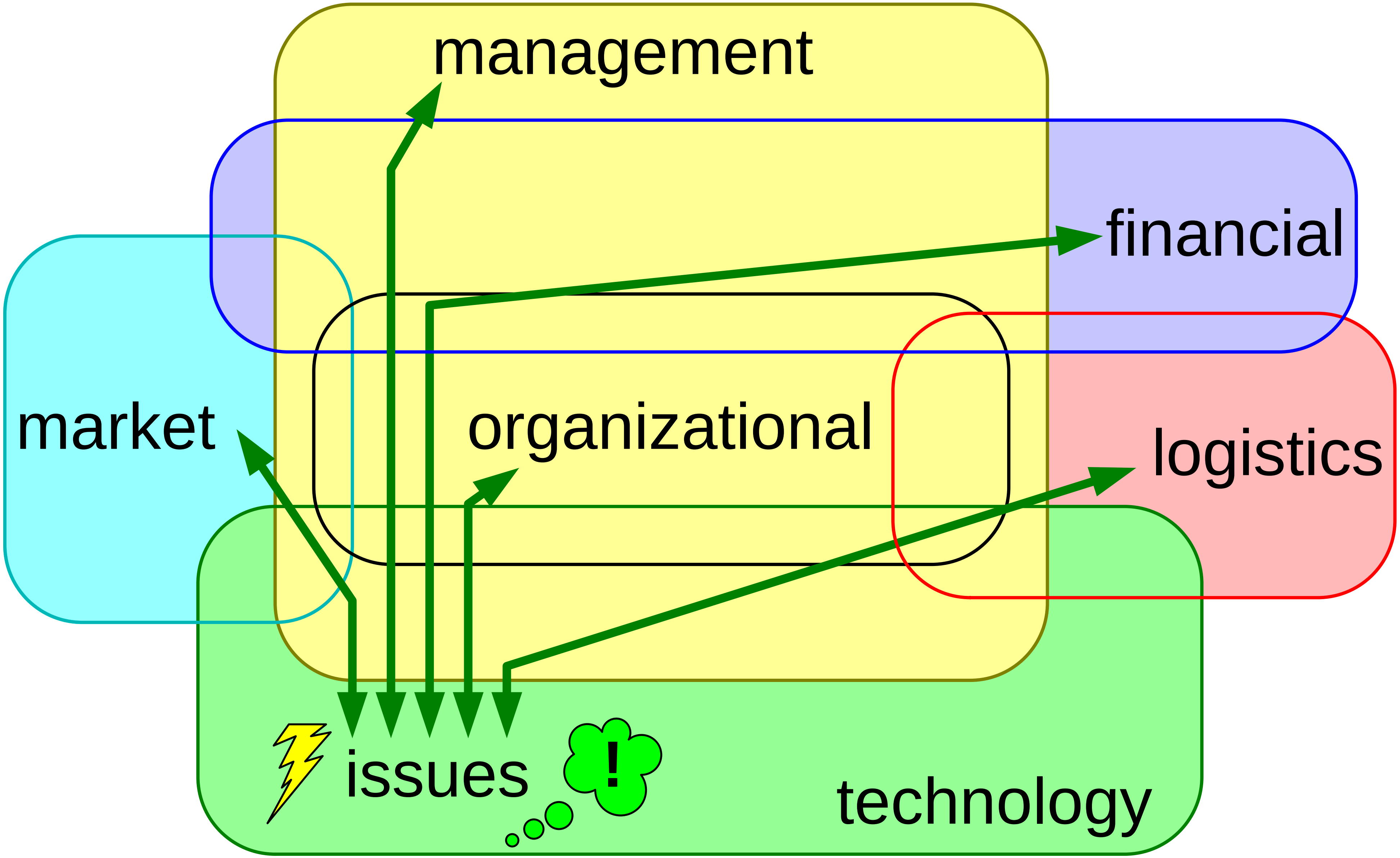
Abstract

Architects struggle with their visibility at higher management echelons. The introvert nature of architects is a severe handicap. Participation of architects in management teams is important for balanced technical sound decisions and strategy. Improved managerial communication skills of architects are required. This article describes how to give a more effective presentation to higher management teams. Subjects discussed are the preparation, content and form, do and don't advise.



February 15, 2024
status: concept
version: 0.1

Architectural issues related to managerial viewpoints



Characteristics of managers in higher management teams

common characteristics

+ action-oriented

+ solution rather than problem

+ impatient, busy

+ want facts not beliefs

+ operate in a political context

+ bottom-line oriented:
profit, return on investment,
market share, etc.

highly variable characteristics

? technology knowledge
from extensive to shallow

? style from power play to
inspirational leadership

How to prepare

Always prepare with small team!

content ⟷ mutual interaction ⟷ understand audience

content — 70% of effort

+ gather facts
+ perform analysis
+ identify goal and message
+ make presentation
+ polish presentation form

understand audience — 30% of effort

+ gather audience background
+ analysis audience interests
+ identify expected responses
+ simulate audience, exercise presentation

Recommended content

+ clear problem statement (what, why)

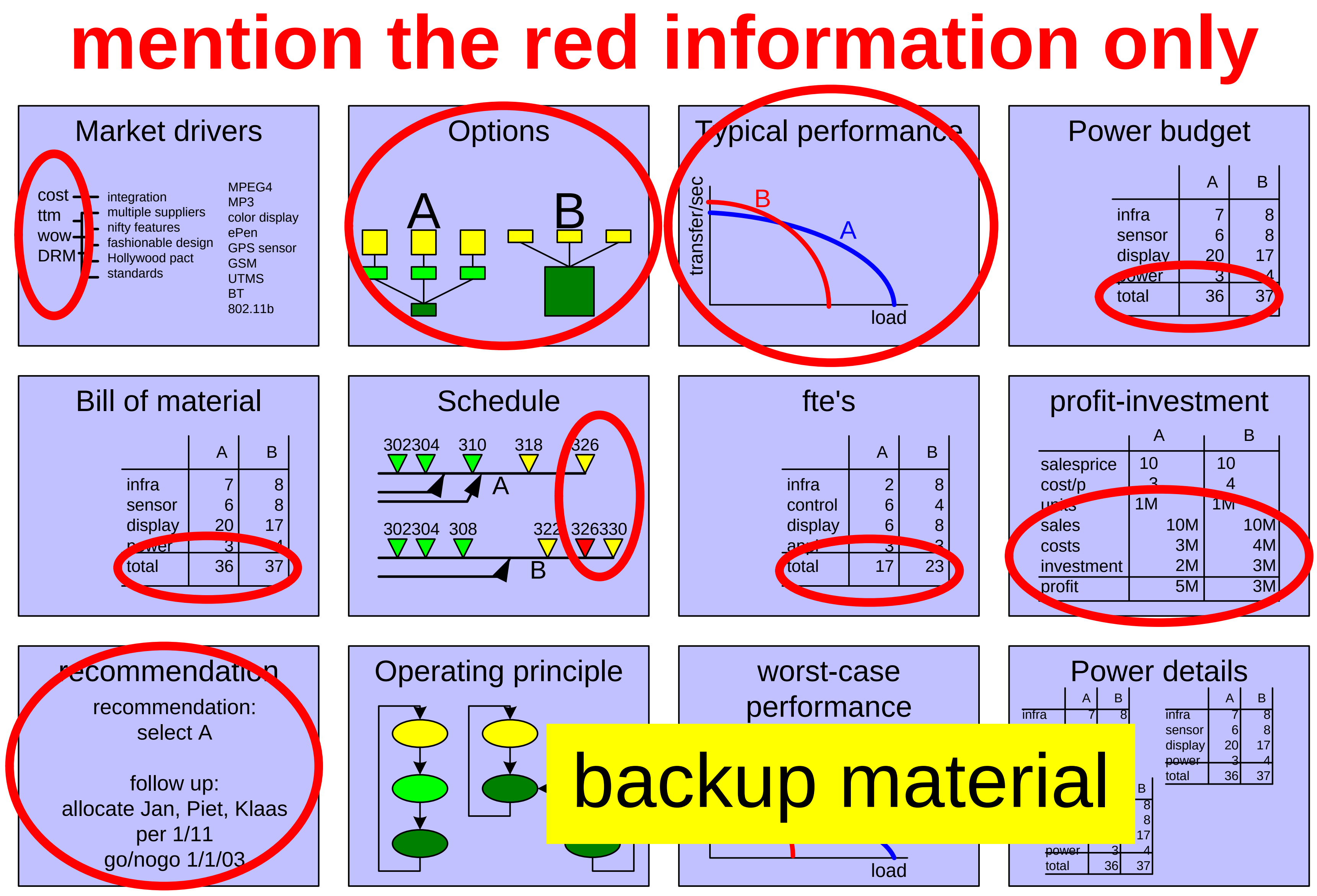
+ solution exploration (how)

+ options, recommendations

+ expected actions or decisions

supported by
facts and figures

Mentioned info, shown info and backup info



Form is important

poor form can easily distract from purpose and content

presentation material

+ professional
+ moderate use of color and animations
+ readable
+ use demos and show artifacts

presenter's appearance

+ well dressed
+ self confident but open

but stay yourself, stay authentic

Don't force your opinion, understand the audience

do not	*do*
- preach beliefs	+ quantify, show figures and facts
- underestimate technology knowledge of managers	+ create faith in your knowledge
- tell them what they did wrong	+ focus on objectives
- oversell	+ manage expectations

How to cope with managerial dominance

do not

- let one of the managers hijack the meeting
- build up tensions by withholding facts or solutions
- be lost or panic at unexpected inputs or alternatives

do

+ maintain the lead
+ be to the point and direct
+ acknowledge input, indicate consequences (facts based)

Exercise presentation to higher management

+ Bring a clear architecture message to

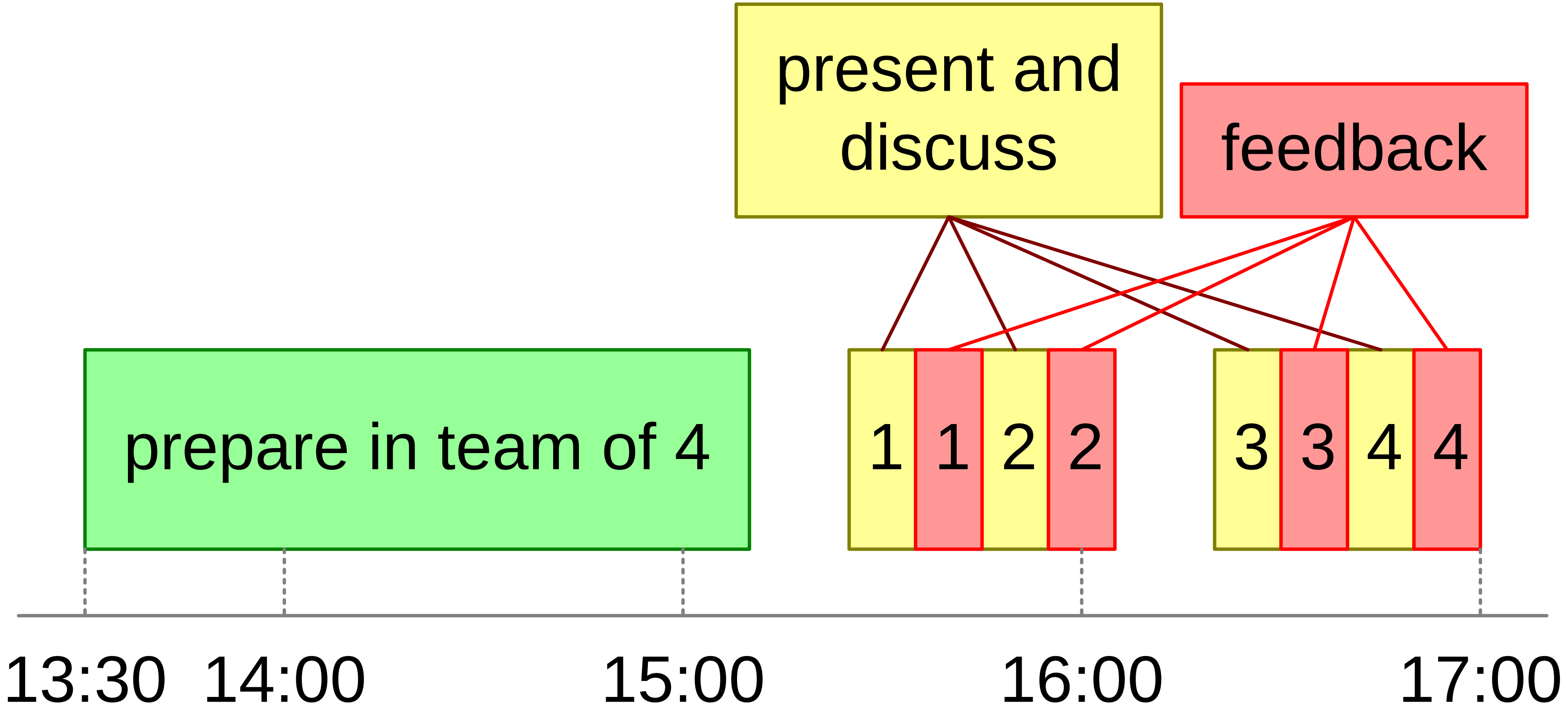
+ a Management team at least 2 hierarchical levels higher

+ with 10 minutes for presentation including discussion
(no limitation on number of slides)

* architecture message =
technology options in relation with market/product

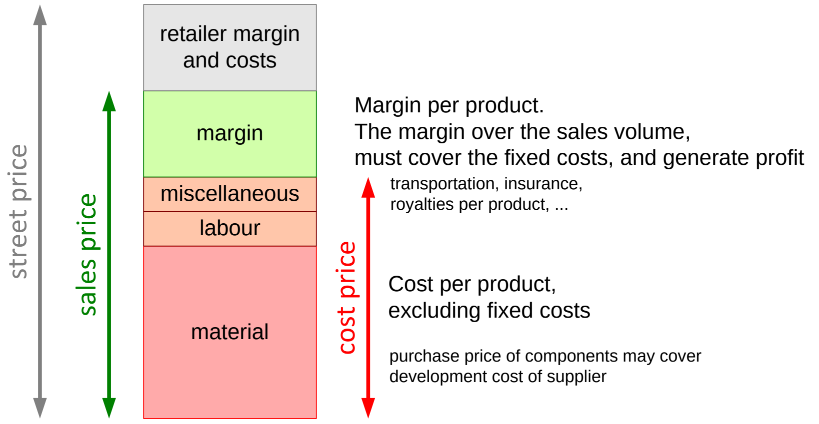
* address the concerns of the management stakeholders:
translation required from technology issues into
business consequences (months, fte's, turnover, profit, investments)

Exercise schedule

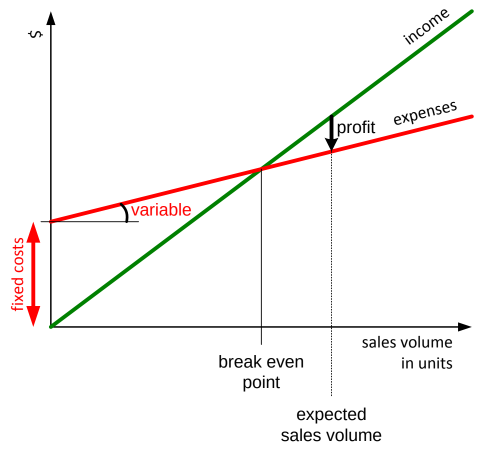


Simplistic Financial Computations

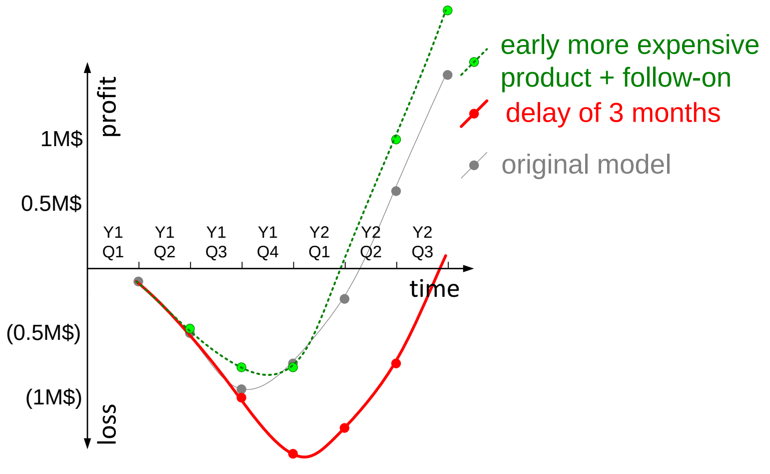
Product Margin = Sales Price - Cost



Profit as function of sales volume



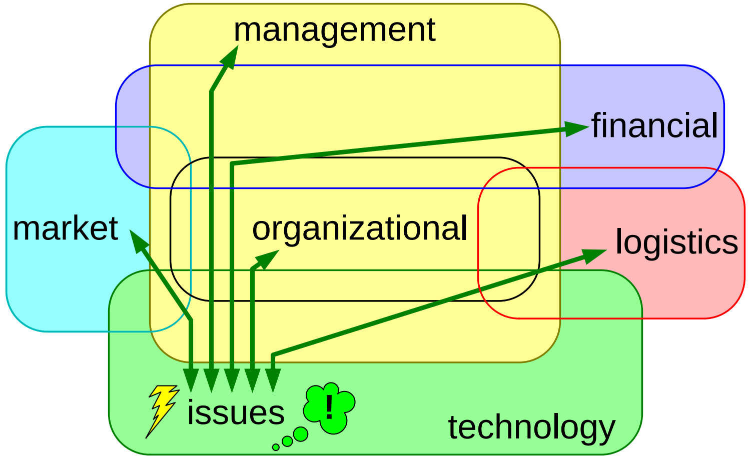
Hockey stick and scenarios



intentionally left blank

Presentation to Management

Managerial Viewpoints



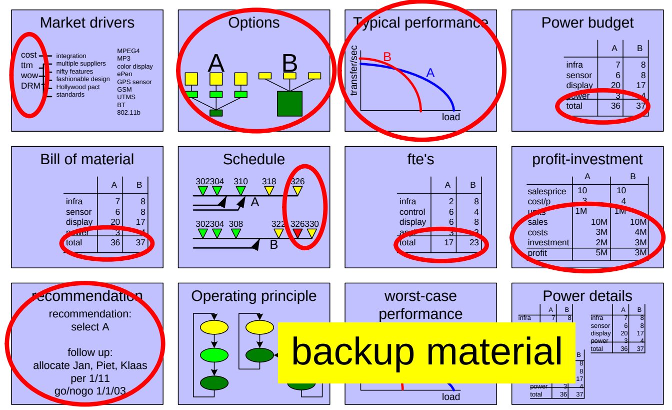
Prepare Content, Understand Audience

Always prepare with small team!

content ⟷ mutual interaction ⟷	understand audience
70% of effort	30% of effort
+ gather facts	+ gather audience background
+ perform analysis	+ analysis audience interests
+ identify goal and message	+ identify expected responses
+ make presentation	+ simulate audience, exercise presentation
+ polish presentation form	

Show underlying info

mention the red information only



Form, do and do not

poor form can easily distract from purpose and content

presentation material	presenter's appearance
+ professional	+ well dressed
+ moderate use of color and animations	+ self confident but open
+ readable	
+ use demos and show artifacts	but stay yourself, stay authentic

Module Human Side

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

The module Human Side addresses the psycho-social aspects of systems architecting.

February 15, 2024
status: preliminary draft
version: 1.6

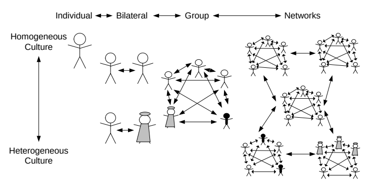
The Human Side of Systems Architecting

by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

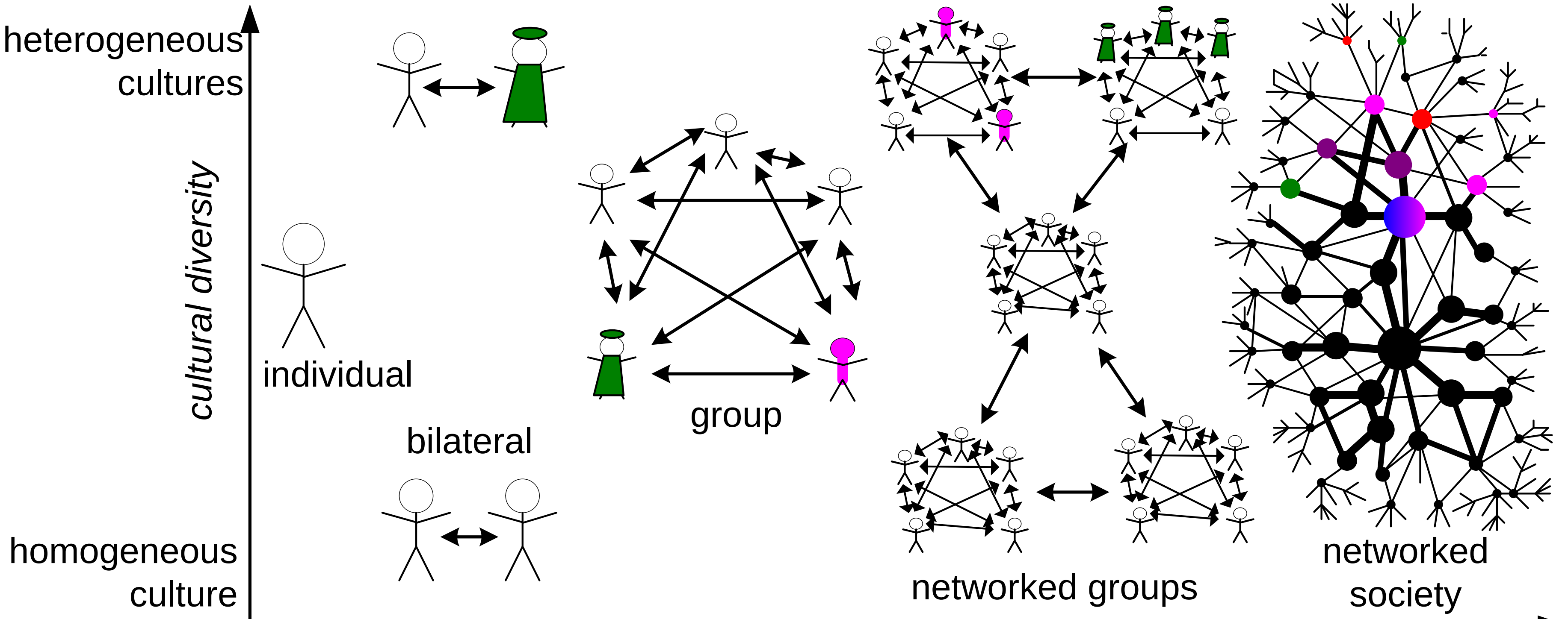
Abstract

Systems architects interact quite often with many humans, and create products that must satisfy human needs. Insight in human aspects is crucial. However, human aspects span a very broad field, the human sciences, that differs quite significantly from the technical background of most architects.

February 15, 2024
status: draft
version: 1.0



Overview of Human Aspects



psychology

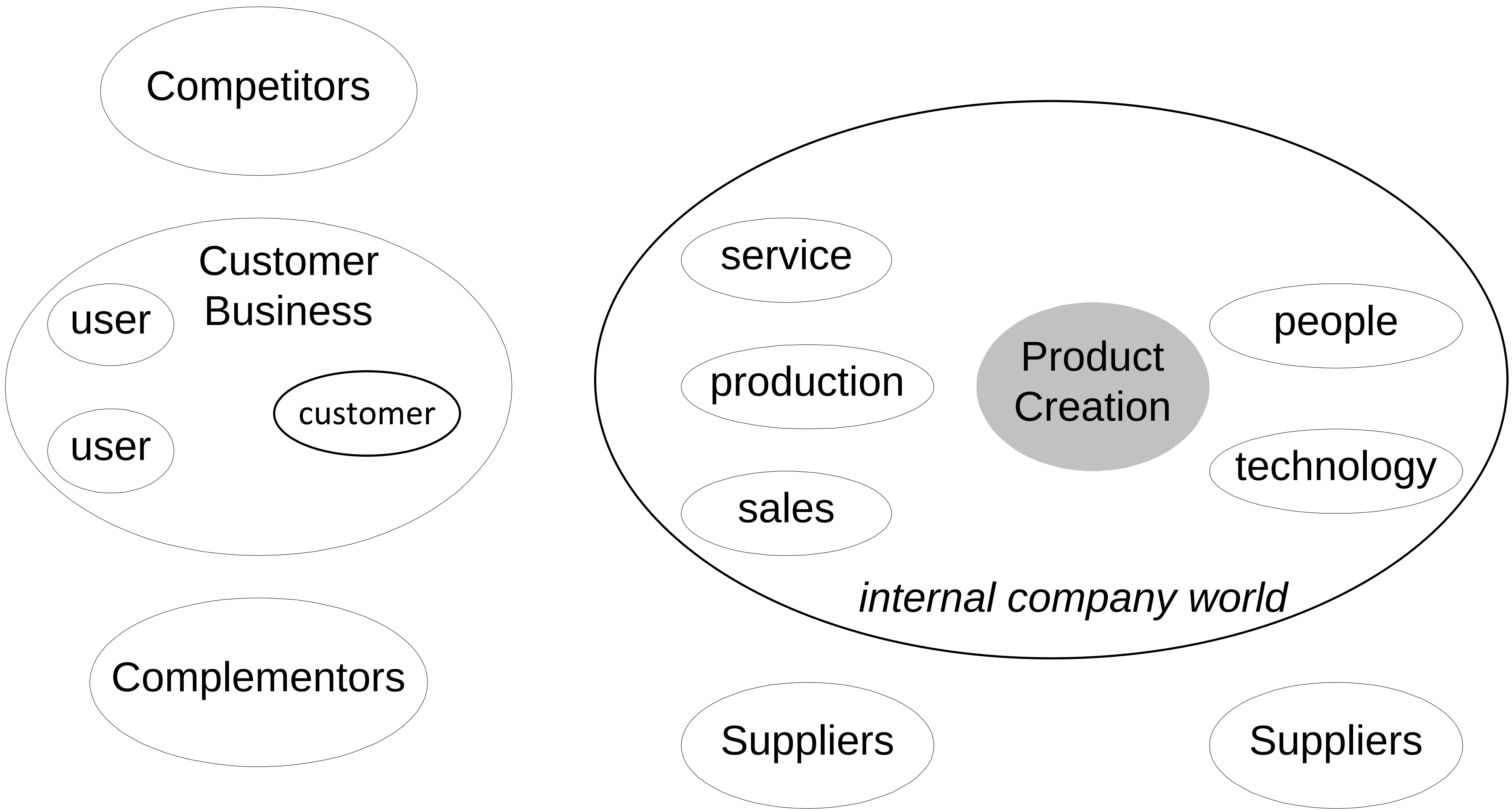
psychiatry — group dynamics — cultural anthropology

physiology — pedagogy — didactics — political science

ergonomics — sociology — criminology

medicine

Context and Stakeholders of Product Creation



Human Measure and Information Technology

A working group, consisting of

- Dieter Hammer (Technical University Eindhoven),
- Jaap van Rees (Van Rees adviesbureau),
- Jeroen van Hoven (Erasmus University Rotterdam),
- Kees van Overveld (Philips Research/TUE),
- Daan Rijsenbrij (Cap Gemini),
- Nathalie Masseus (Cap Gemini),
- and Gerrit Muller (Philips Research)

wants to increase the awareness in the ICT-architecture community of the human aspects.

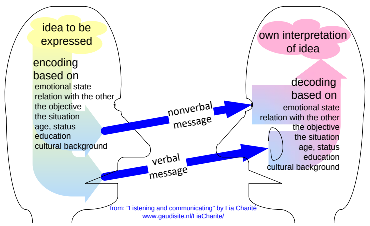
Human Side: Interpersonal Skills

by *Gerrit Muller* USN-SE
e-mail: `gaudisite@gmail.com`
`www.gaudisite.nl`

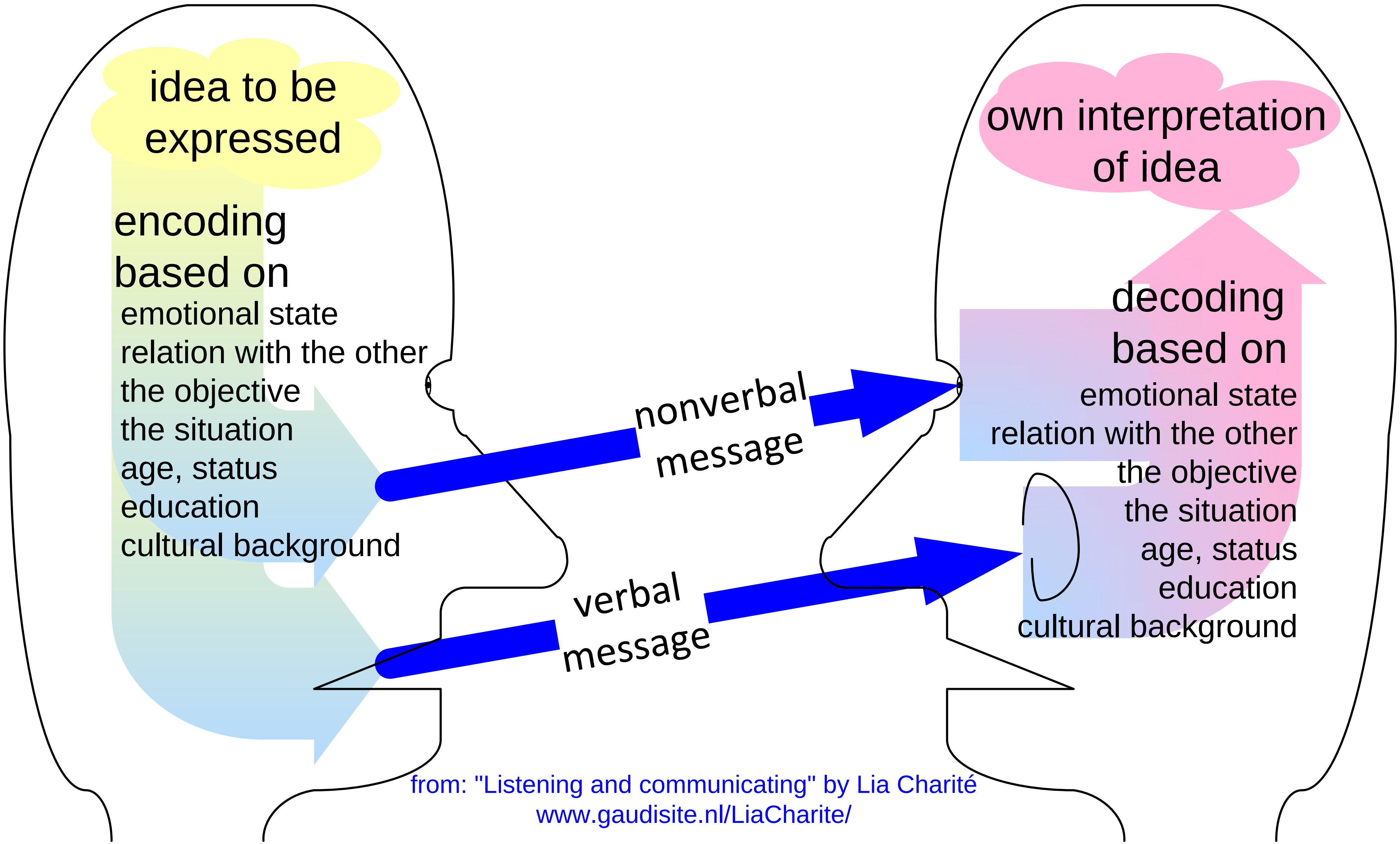
Abstract

We discuss in this paper a set of skills and techniques to cooperate effectively between two individuals. We show the wonders of communication and then we address techniques such as investigation and acknowledgement, constructive feedback, conflict management, appraisal, good practices in a conversation, searching for ideas.

February 15, 2024
status: draft
version: 0.1



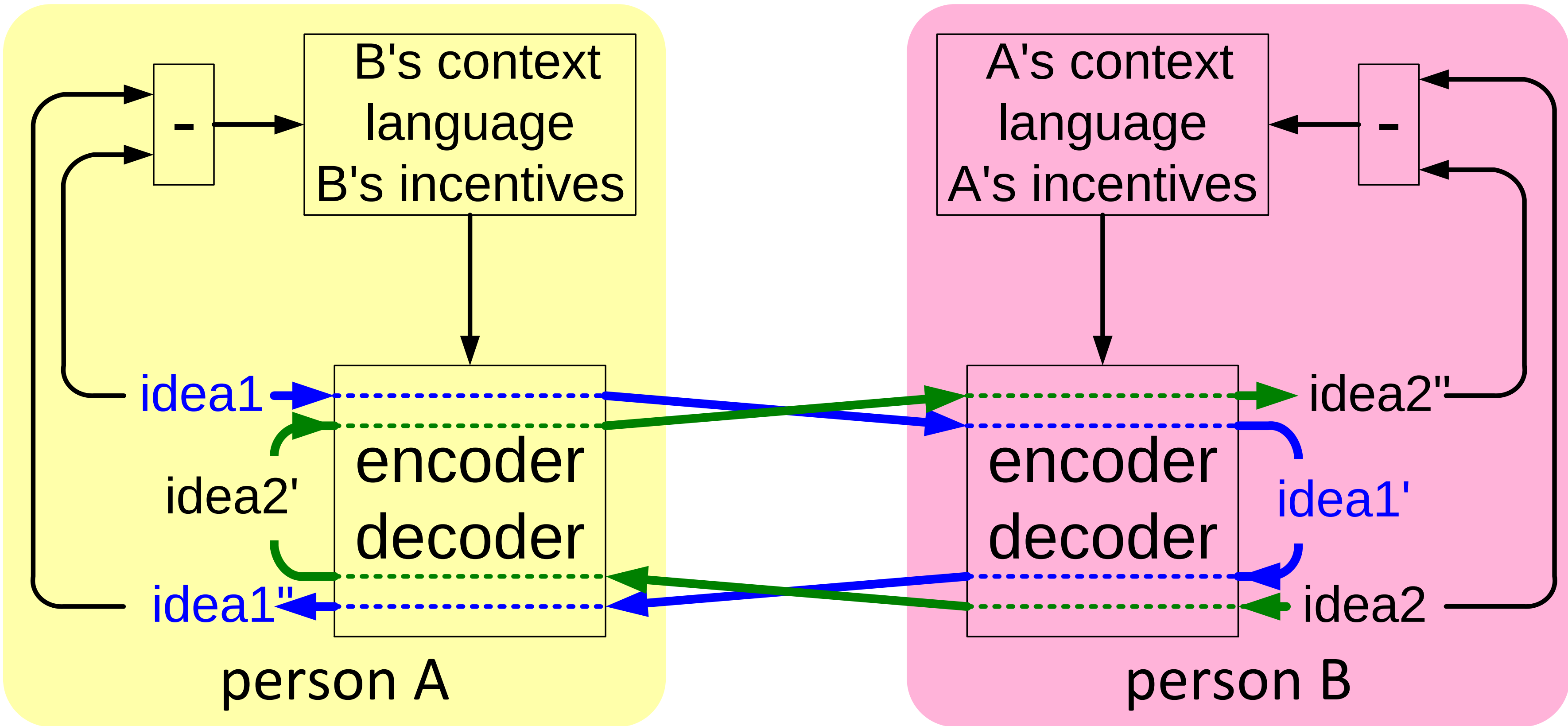
Active listening: the art of the receiver to decode the message



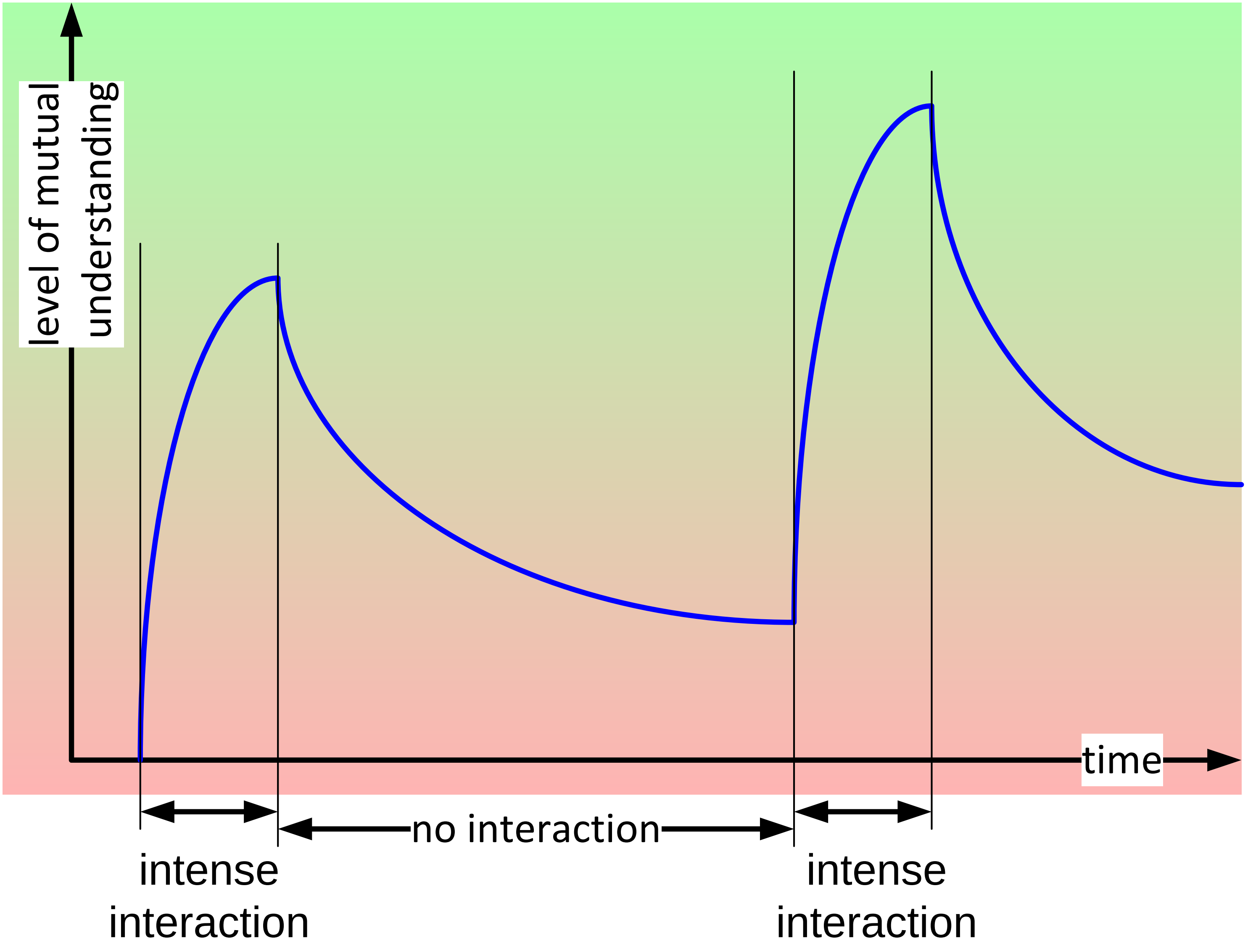
from: "Listening and communicating" by Lia Charité
www.gaudisite.nl/LiaCharite/

Intense interaction needed for mutual understanding

to calibrate:
repeat many times with different
examples, illustrations, and explanations



Mutual understanding as function of time



The material for interpersonal skills

is based on a set of techniques

from a course

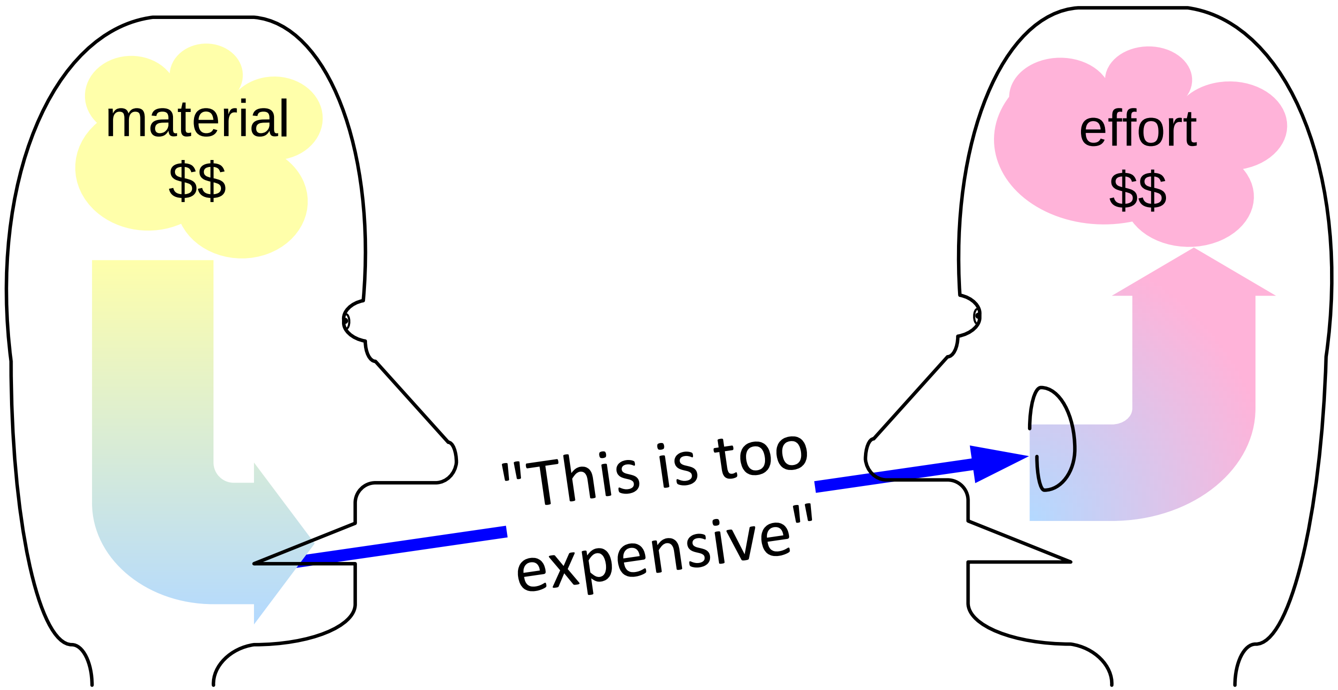
"Interpersonal Management Skills"

by

Hay Management Consultants

in 1998

Investigate and Acknowledge



investigate:
What has been said and why?

acknowledge:
Paraphrase what has been said and why?
i.e. use your own words

When a decision will be taken or an action will be started on the basis of exchanged information, opinions or suggestions
or
when the first reaction is to reject, ignore or contradict what you just heard.

Constructive Feedback

How

+ Indicate the strong points to be kept

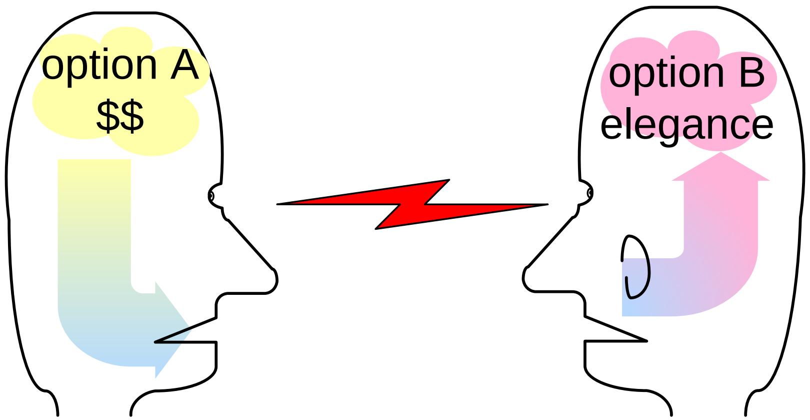
+ Indicate the points to be improved

+ Search for solutions which build upon the strong points and improve the weak points

When

You want to facilitate someone to improve his/her performance

Conflict Management



When

in case of conflict

How?

define the positions:

* indicate what is important for you and why
* investigate and acknowledge what is important for the other and why

IF

If you are willing and able to consider alternatives:

Search for alternative solutions

If you are not willing and able to consider alternatives, or no acceptable solution for both parties can be found:

Finish the conversation:

* acknowledge the right to have a different opinion
* indicate your decision and why

Appraisal

When

Someone's performance is important for you

* exceeding the expectations
* meets expectations continuously
* meets expectations, which exceed the normal
  performance level of this person

Appraise only when authentic!

How

+ Mention the performance very specific.
+ Mention the personal qualities which lead to this
  performance.
+ Describe which advantages arise for you,
  the department or the organization.

Conversation Good Practices

When you open a conversation

formulate the purpose

When you finish the conversation

summarize the agreements and the actionplan

Searching for Ideas

When asking for a suggestion	give a reaction
When supplying a suggestion	ask for a reaction
When you use or build upon ideas of others	mention the source of the ideas
When you need new or more creative ideas	remove limitations temporarily or add limitations

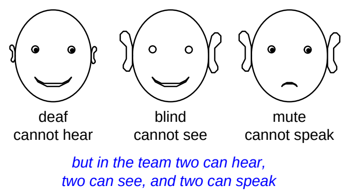
Human Side: Team Work

by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

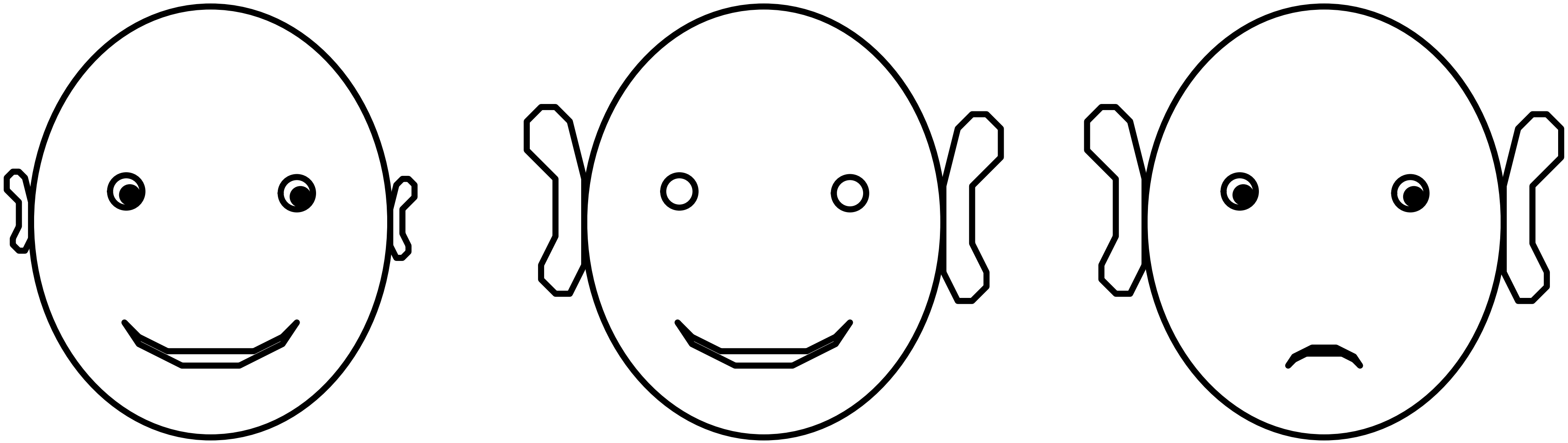
Abstract

The creation of products requires many different people to cooperate. The work is often organized in teams. The team members have complimentary skills and knowledge. In many management courses the need to design teams is emphasized. Unfortunately, often these recommendations are ignored. We re-iterate in this paper the rationale for teams and the recommendations for designing the team itself.

February 15, 2024
status: draft
version: 0.2



Teams consist of complementary people



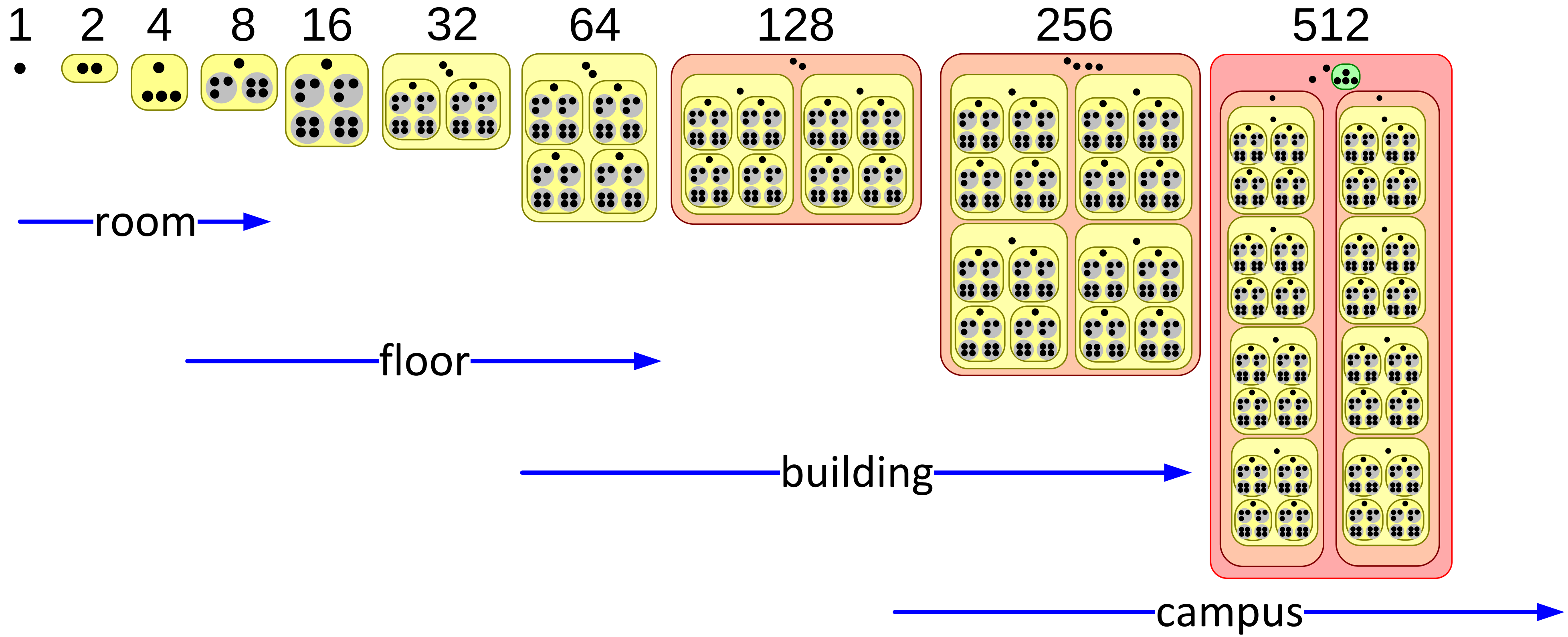
deaf
cannot hear

blind
cannot see

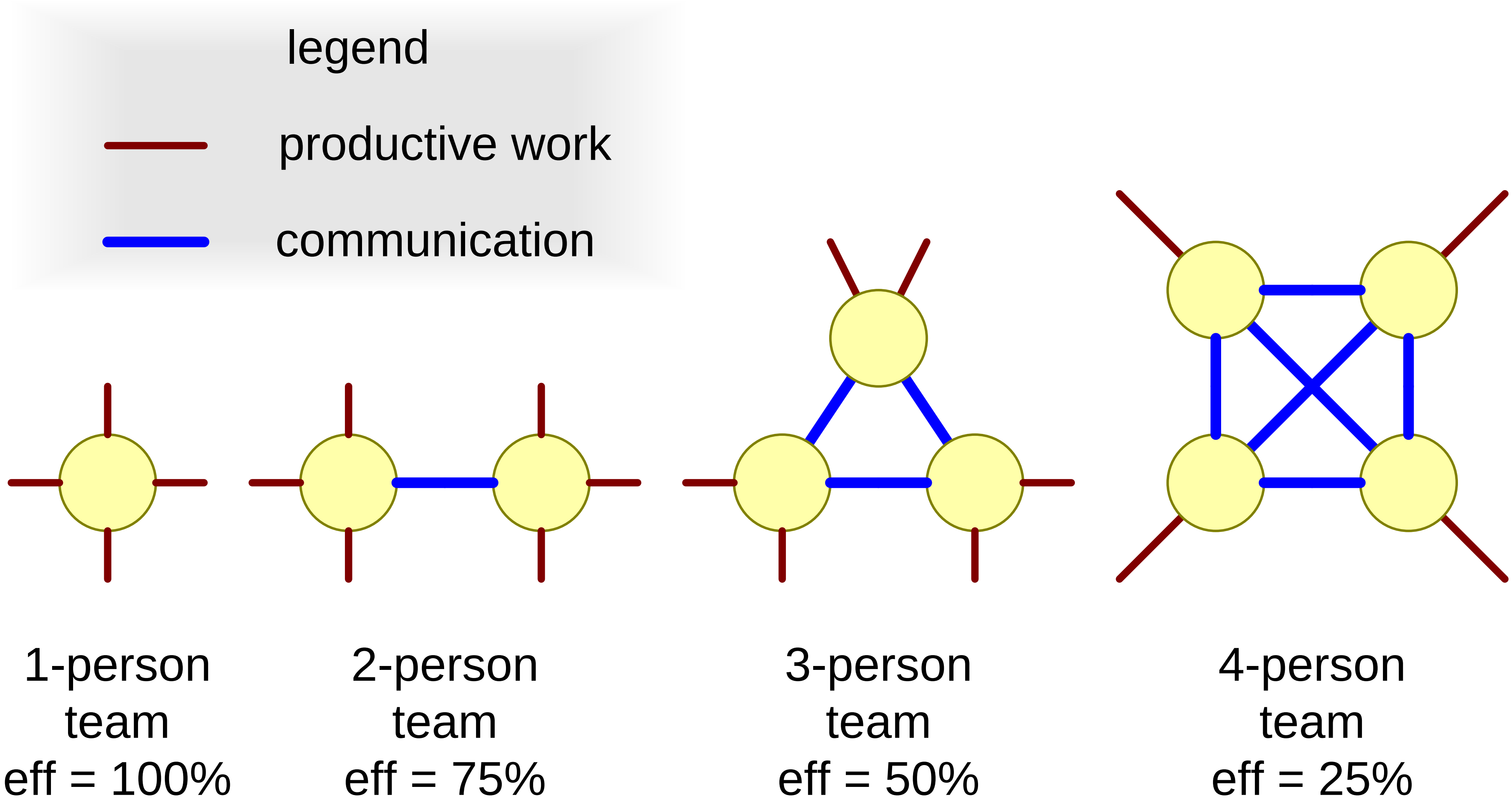
mute
cannot speak

but in the team two can hear,
two can see, and two can speak

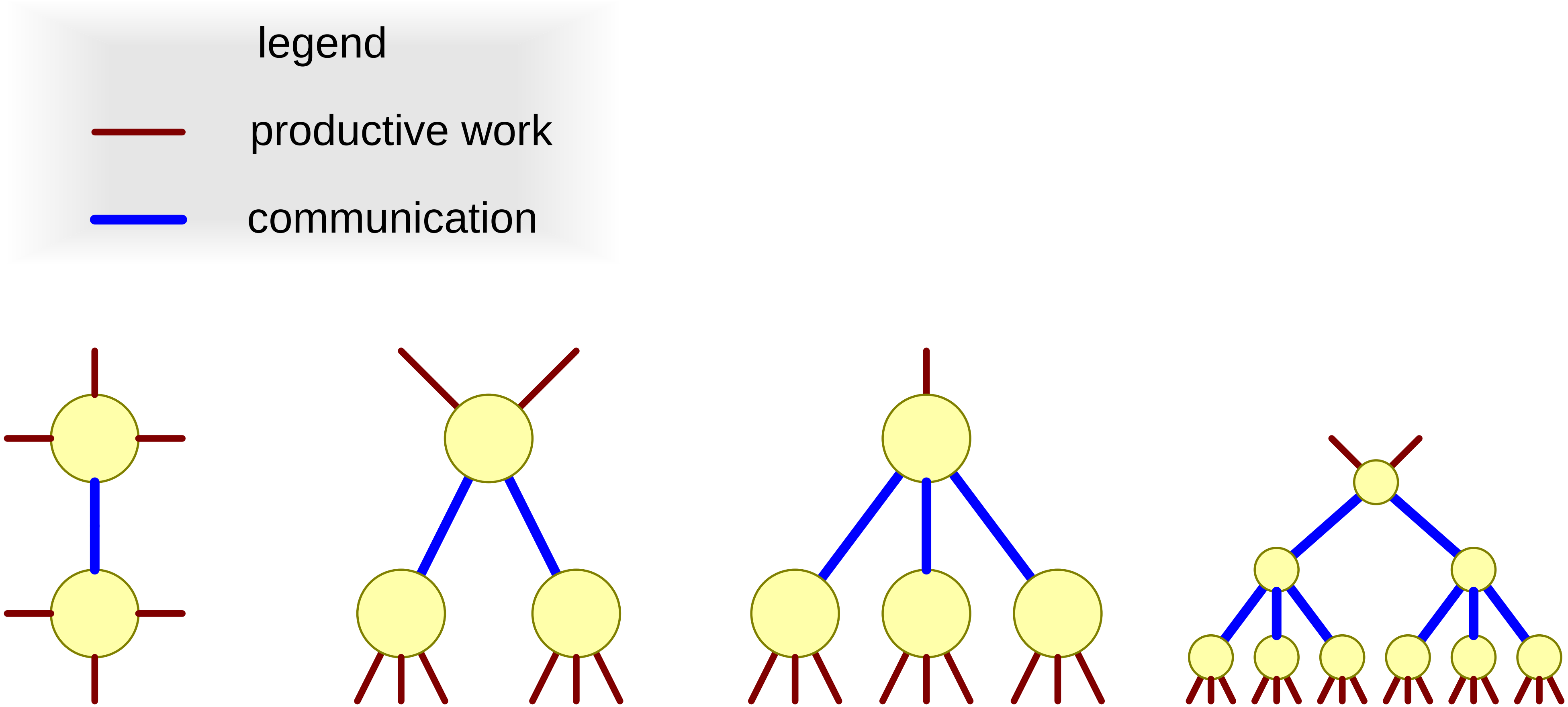
Organization size and teams



Very simplistic team model



Hierarchical simplistic team model



2-person team eff = 75%

3-person team eff = 66%

4-person team eff = 62.5%

9-person team eff ~= 56%

Many personality and role models are available

neutral facts

feeling instinctive

negative flaws

positive benefits

creative ideas

process meta

Six thinking hats by Edward de Bono

E Extraversion	⟷	Introversion **I**
S Sensing	⟷	iNtuition **N**
T Thinking	⟷	Feeling **F**
J Judging	⟷	Perceiving **P**

Myers-Briggs Type Indicators

plant
creative

team worker
co-operative,
averts friction

implementer
disciplined, conservative,
do-er

resource investigator
enthusiatic
communicator

shaper
driver, dynamic

completer finisher
conscientious, painstaking

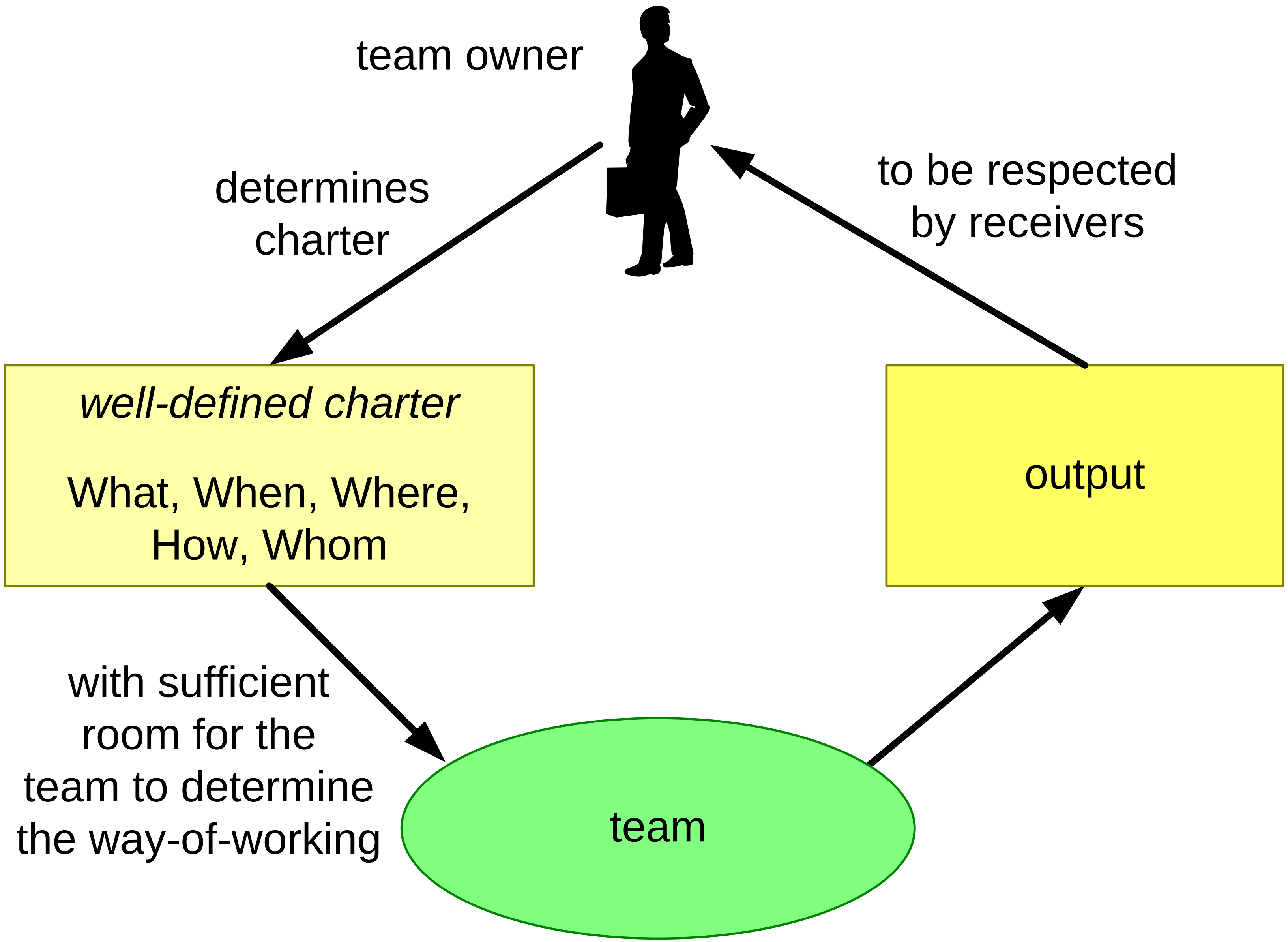
co-ordinator
mature, chairman

monitor evaluator
sober, analytical

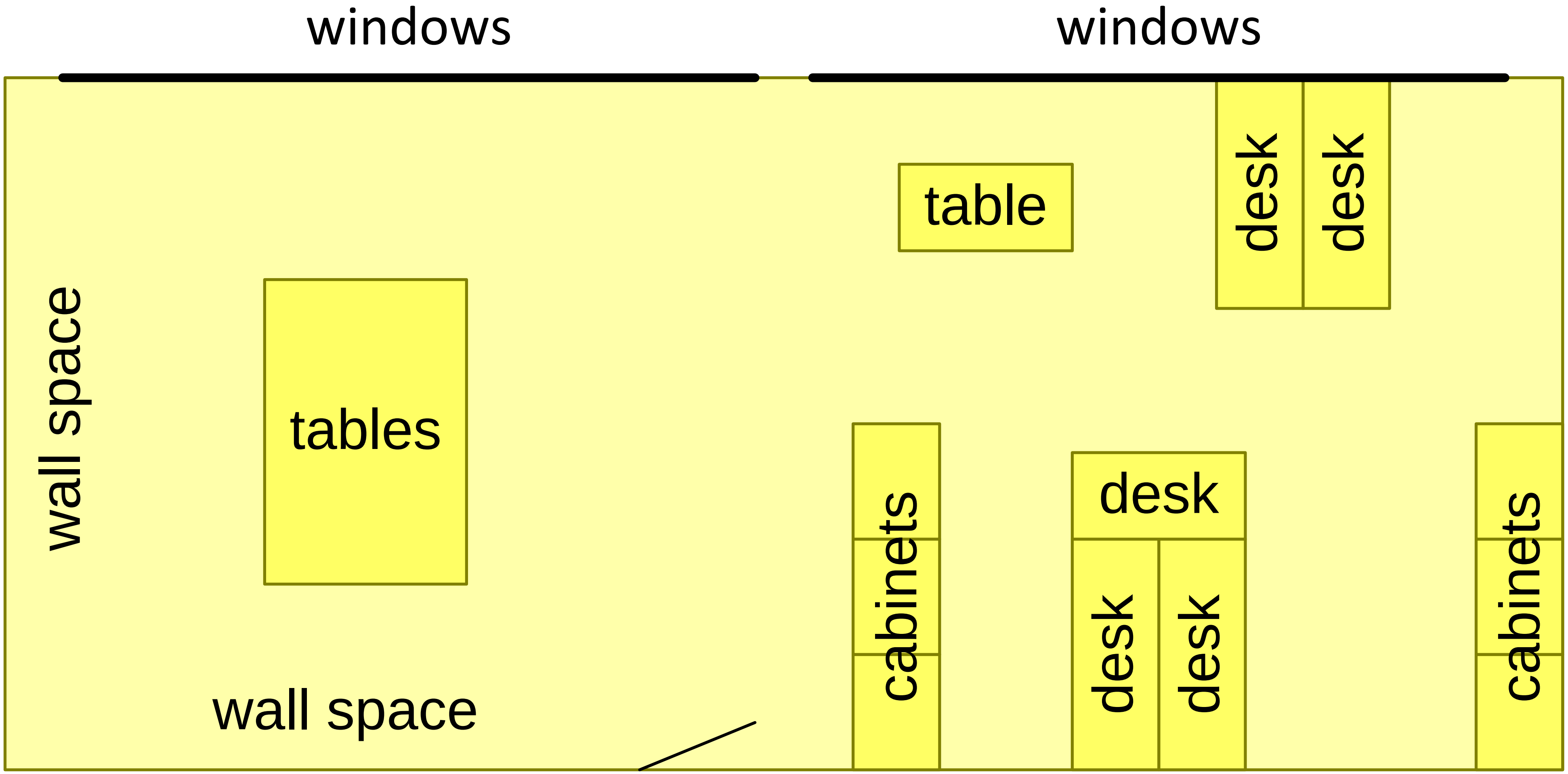
specialist
single-minded, rare skills

Belbin's team roles

Process of creating and using a team

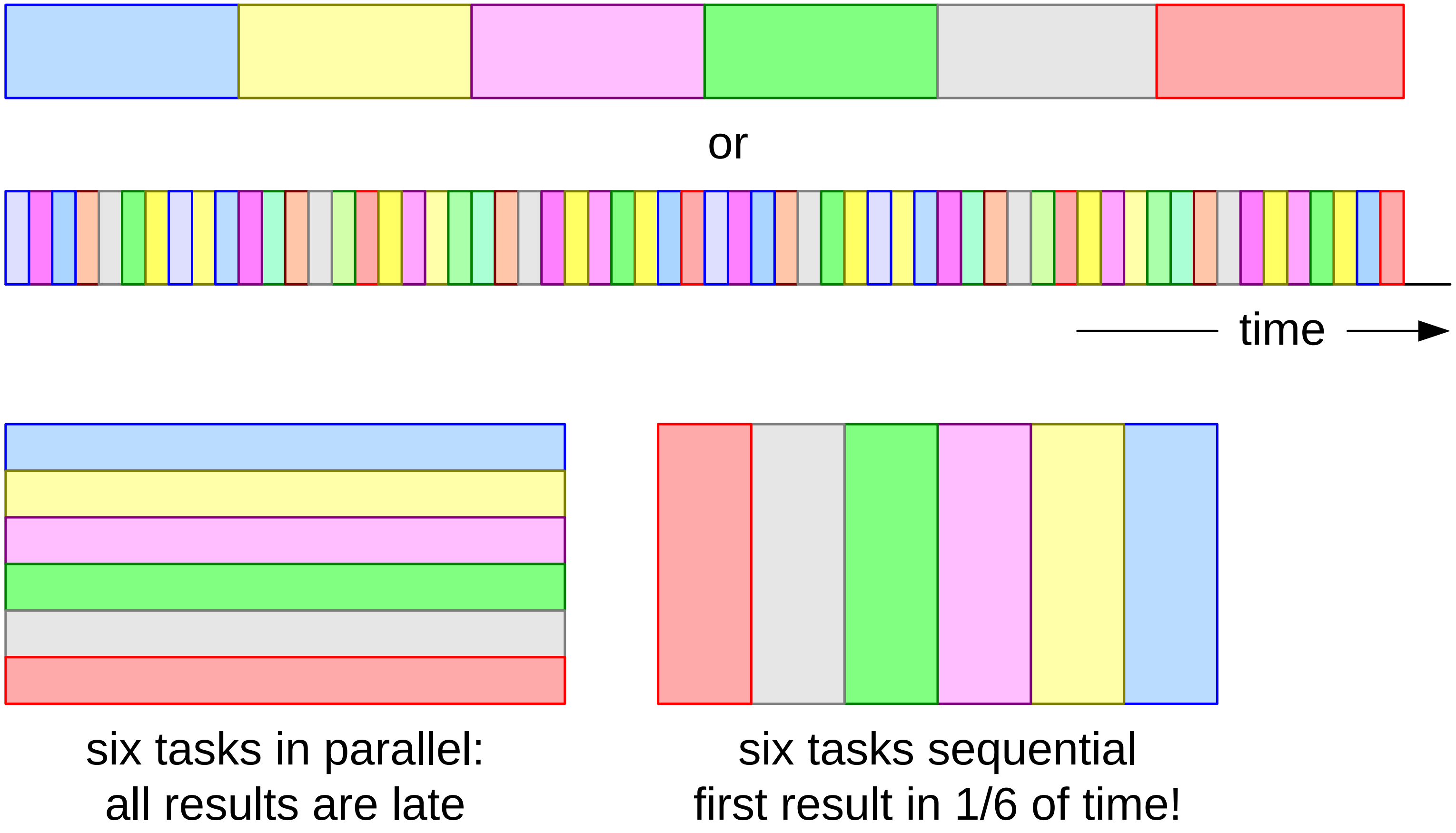


“War Room” is very effective

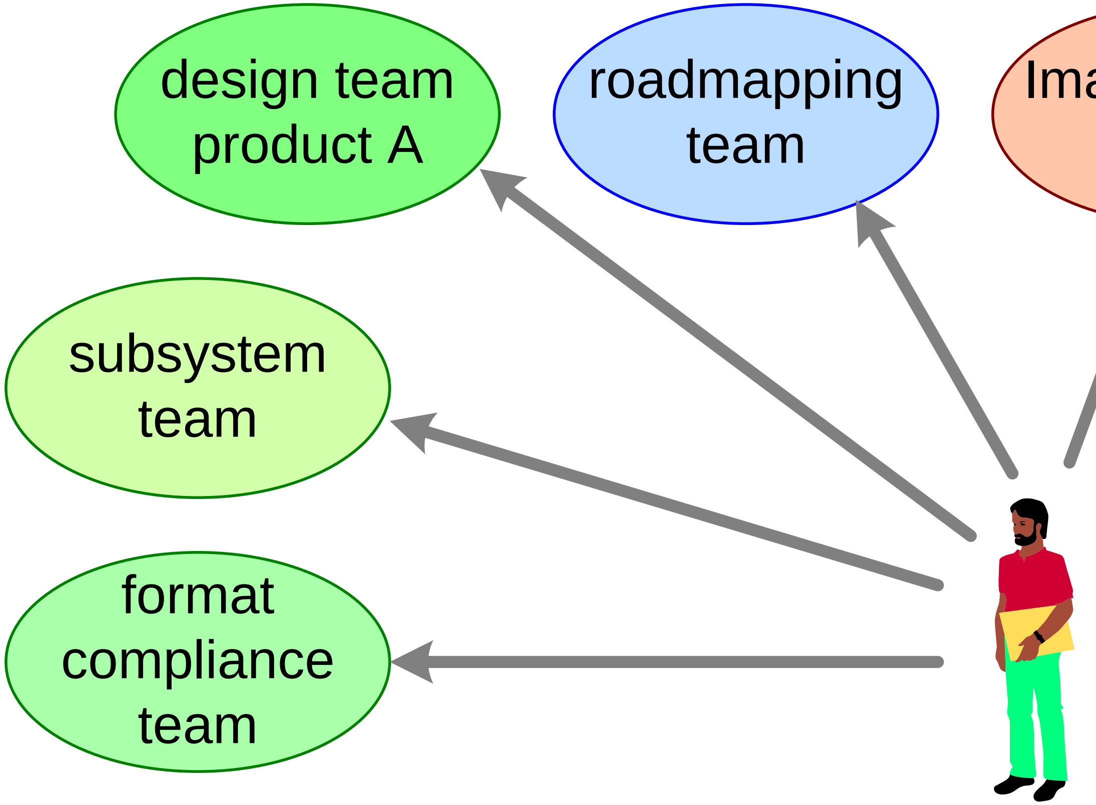


Concurrency and Fragmentation lower efficiency

How many (semi-)concurrent tasks can a person handle?
Working in burst-mode (concentrating on one task for one day, week or month) can increase efficiency.



One person will be member of multiple teams



It is quite normal to participate in many teams simultaneously.

However, a team can only function if the members are sufficiently available!

Critical Success Factors for teams

well defined charter

clear owner of the result

respect for the output of the team

freedom of way-of-working

housing and location

availability of team members

complementary roles

diversity, pluriformity

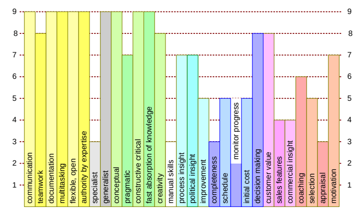
Function Profiles; The Sheep with Seven Legs

by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

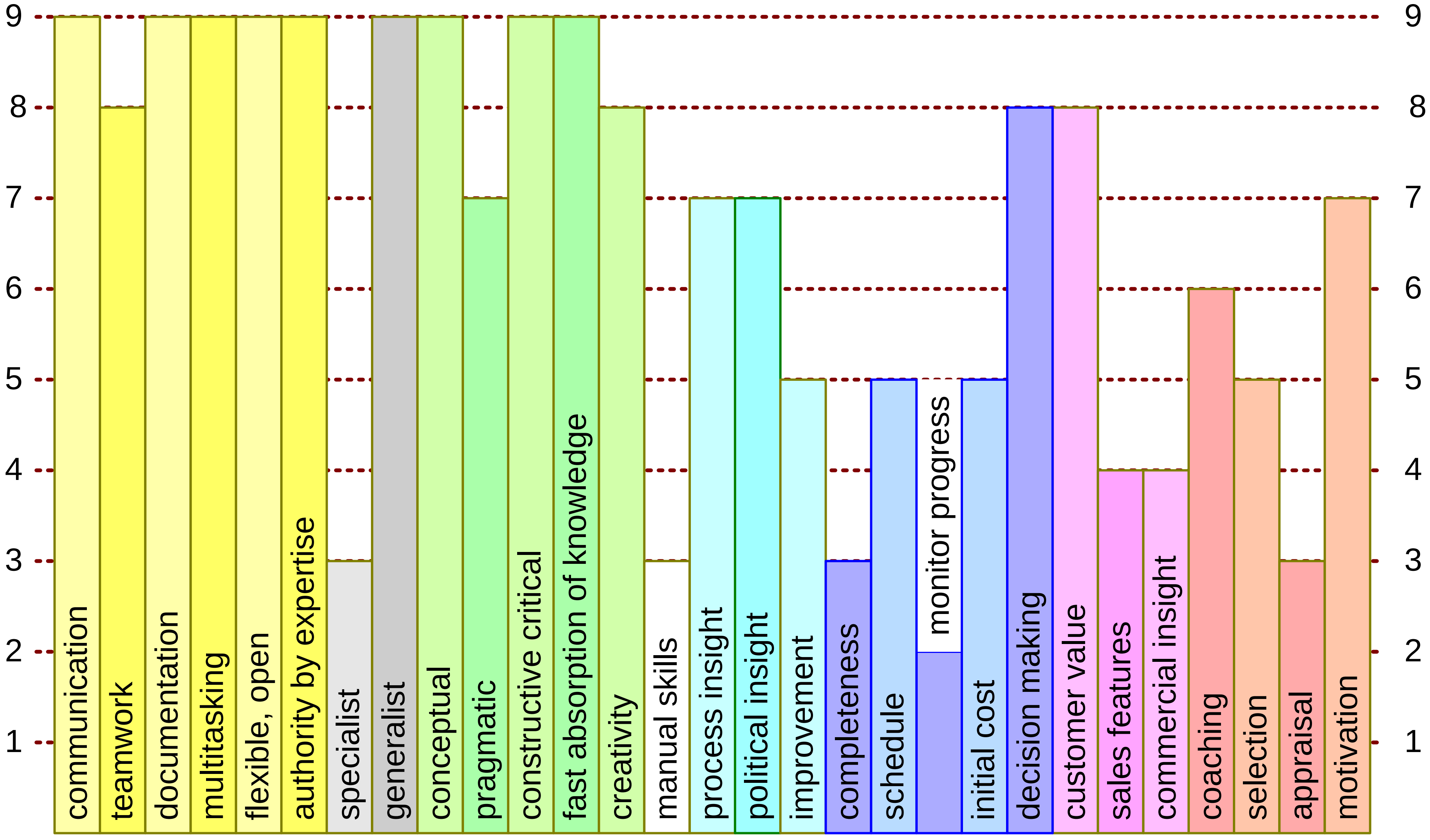
Abstract

The profile of a system architect is quantified for a large list of system architect related characteristics. For comparison the function profiles of related functions are given as well. This profile is based on personal observations and experience.

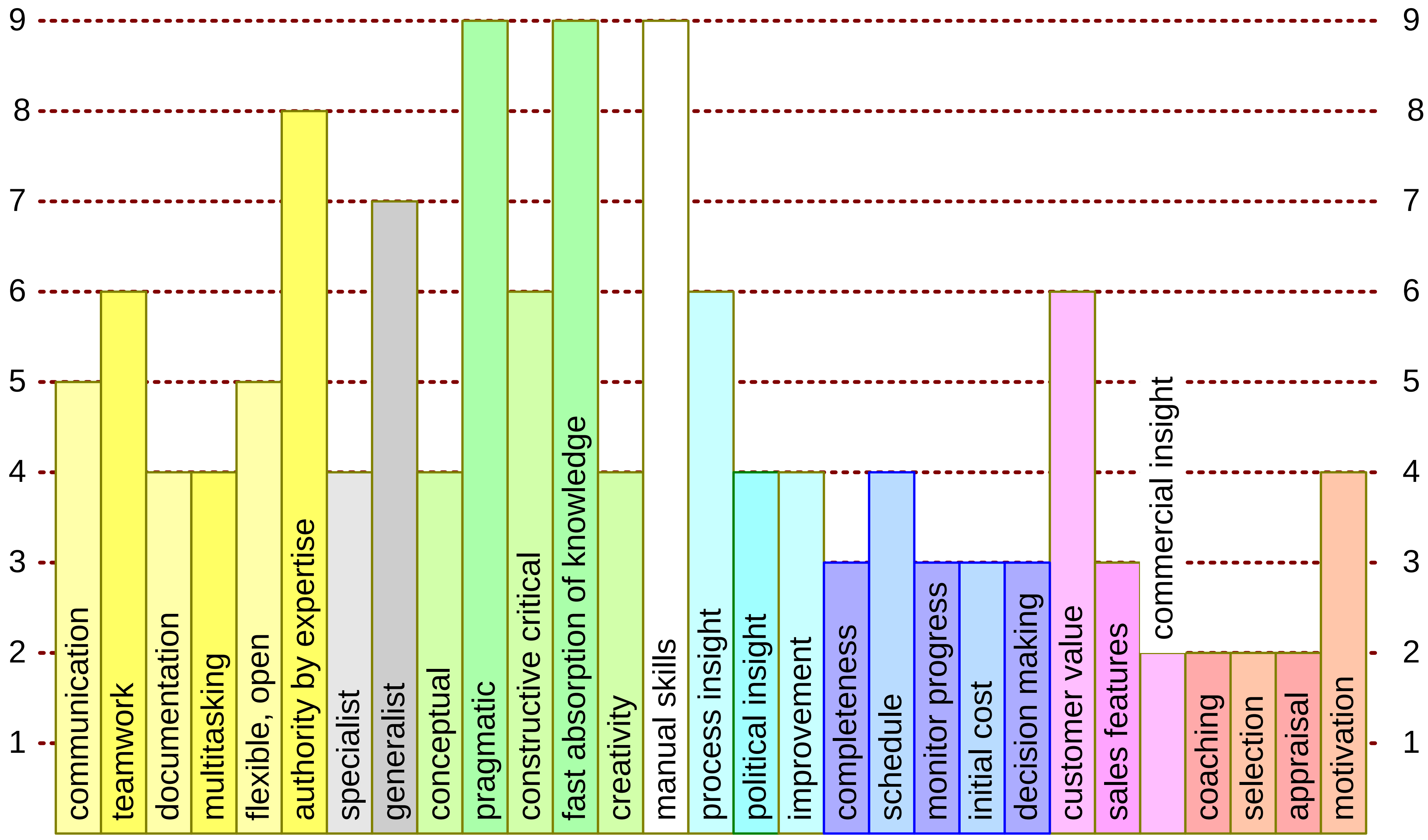
February 15, 2024
status: concept
version: 1.0



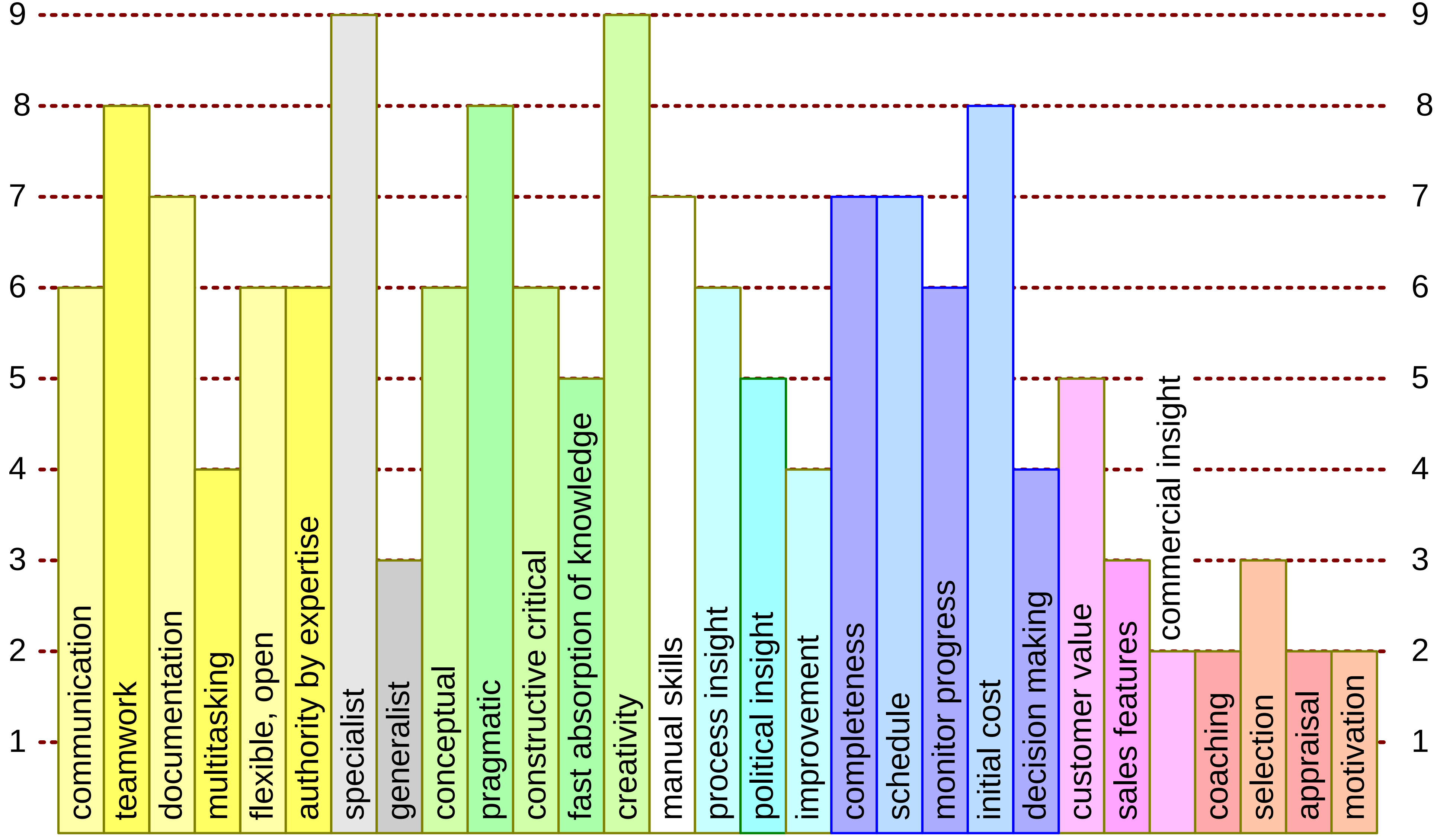
System Architect



Test Engineer

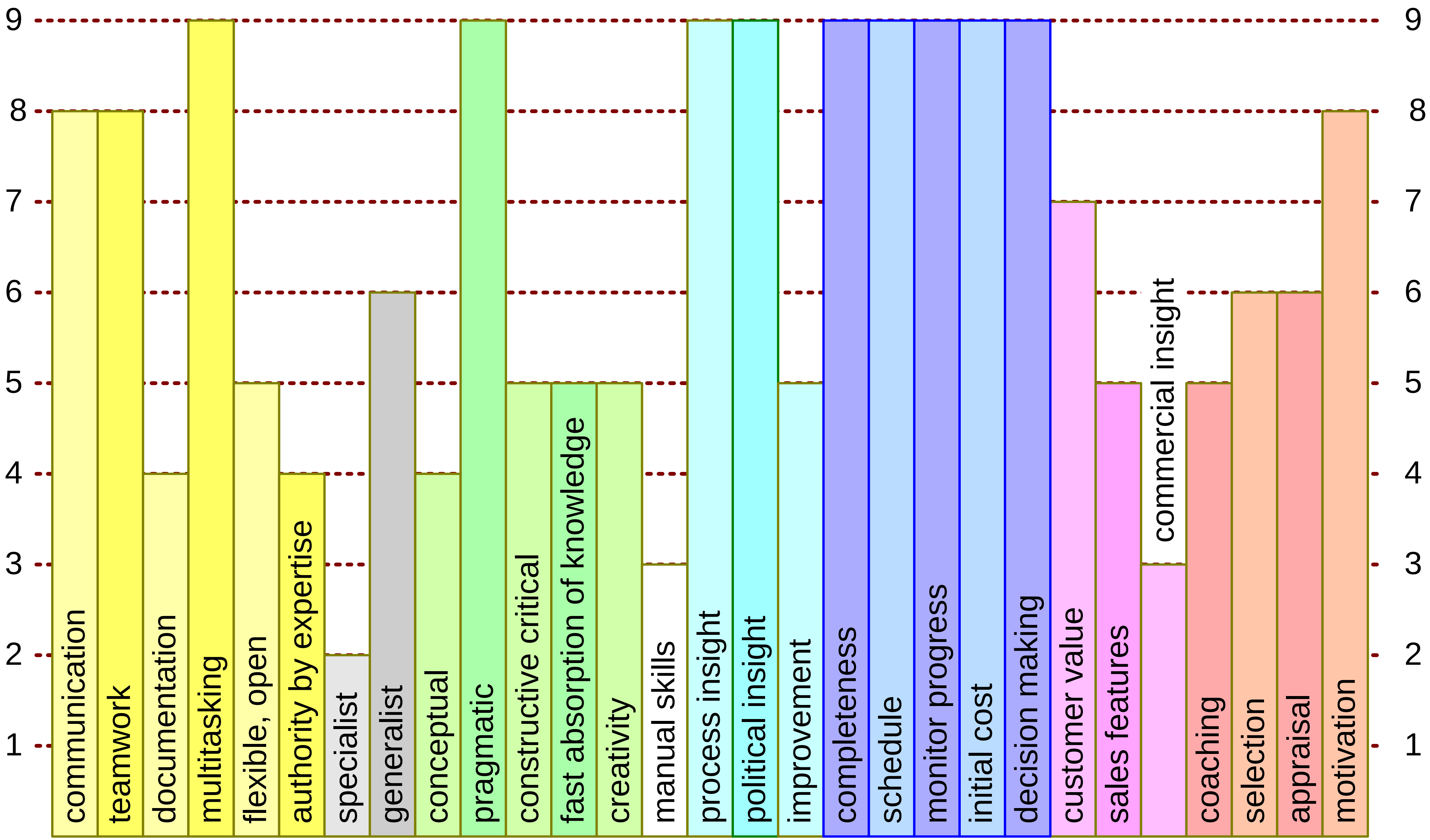


Developer

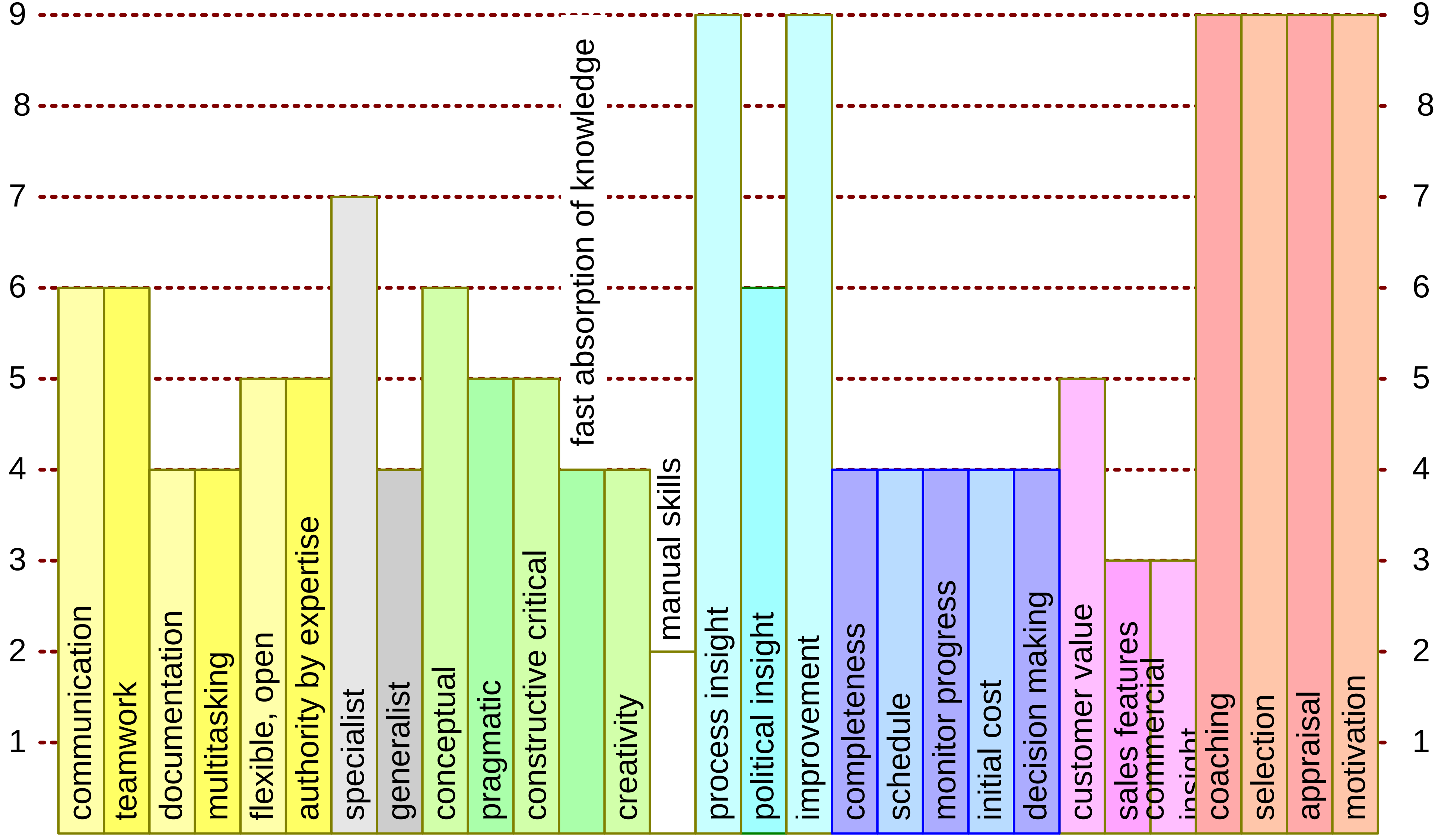


9
8
7
6
5
4
3
2
1
communication
teamwork
documentation
multitasking
flexible, open
authority by expertise
specialist
generalist
conceptual
pragmatic
constructive critical
fast absorption of knowledge
creativity
manual skills
process insight
political insight
improvement
completeness
schedule
monitor progress
initial cost
decision making
customer value
sales features
commercial insight
coaching
selection
appraisal
motivation

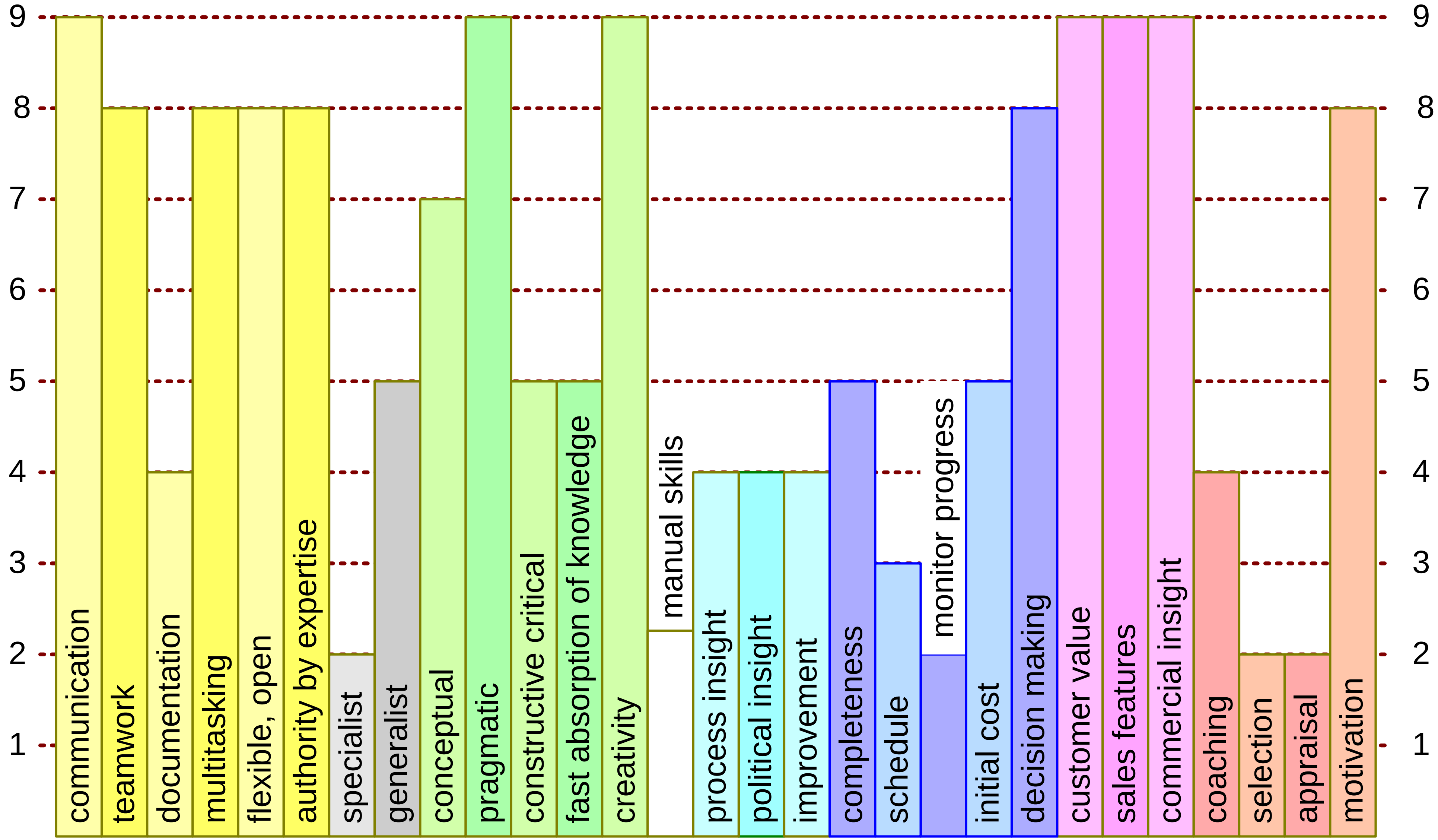
Operational Leader



Line Manager



Commercial Manager



The numbers behind the bars

	communication	teamwork	documentation	multitasking	flexible, open	authority by expertise	specialist	generalist	conceptual	pragmatic	constructive critical	fast absorption of knowledge	creativity	manual skills	process insight	politicsal insight	improvement	completeness	schedule	monitor progress	initial cost	decision making	customer value	sales features	commercial insight	coaching	selection	appraisal	motivation
systems architect	9	8	9	9	9	9	3	9	9	7	9	9	8	3	7	7	5	3	5	2	5	8	8	4	4	6	5	3	7
test engineer	5	6	4	4	5	8	4	7	4	9	6	9	4	9	6	4	4	3	4	3	3	3	6	3	2	2	2	2	4
developer	6	8	7	4	6	6	9	3	6	8	6	5	9	7	6	5	4	7	7	6	8	4	5	3	2	2	3	2	2
operational leader	8	8	4	9	5	4	2	6	4	9	5	5	5	3	9	9	5	9	9	9	9	9	7	5	3	5	6	6	8
line manager	6	6	4	4	5	5	7	4	6	5	5	4	4	2	9	6	9	4	4	4	4	4	5	3	3	9	9	9	9
commercial manager	9	8	4	8	8	8	2	5	7	9	5	5	9	2	4	4	4	5	5	2	5	8	9	9	9	4	2	2	8

Exercise Psycho-Social Side

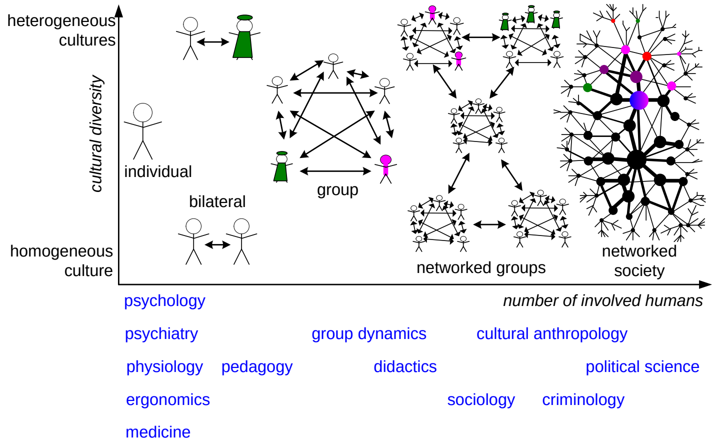
Make a (critical and honest) profile of yourself and of the operational or the line manager, who thinks he is managing you.
Select 2 characteristics which you find difficult to assess or where you expect that other people will have a totally different perception. Discuss these 2 characteristics in the group.
Present (max 1 flip) the highlights.

Characteristics

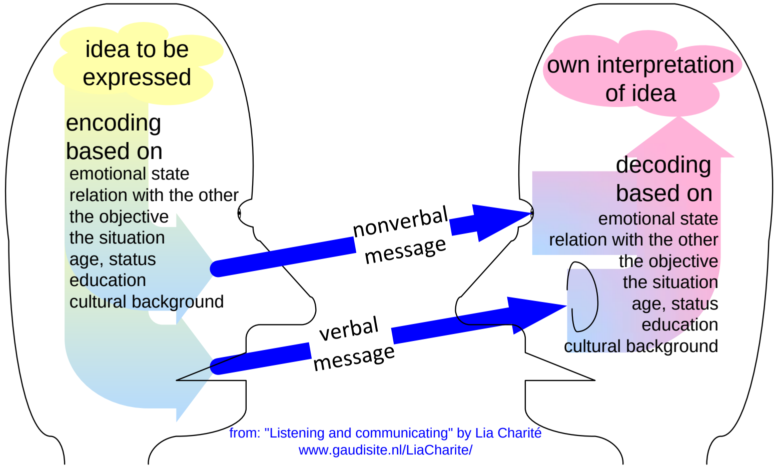
	communication	teamwork	documentation	multitasking	flexible, open	authority by expertise	specialist	generalist	conceptual	pragmatic	constructive critical	fast absorption of knowledge	creativity	manual skills	process insight	political insight	improvement	completeness	schedule	monitor progress	initial cost	decision making	customer value	sales features	commercial insight	coaching	selection	appraisal	motivation

Human Side

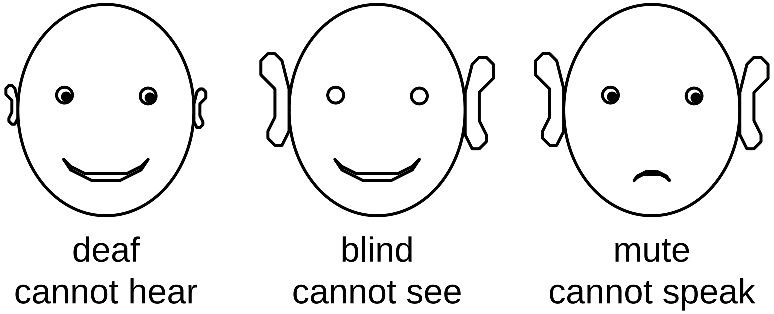
From Individual to Society



The Wonder of Bilateral Communication

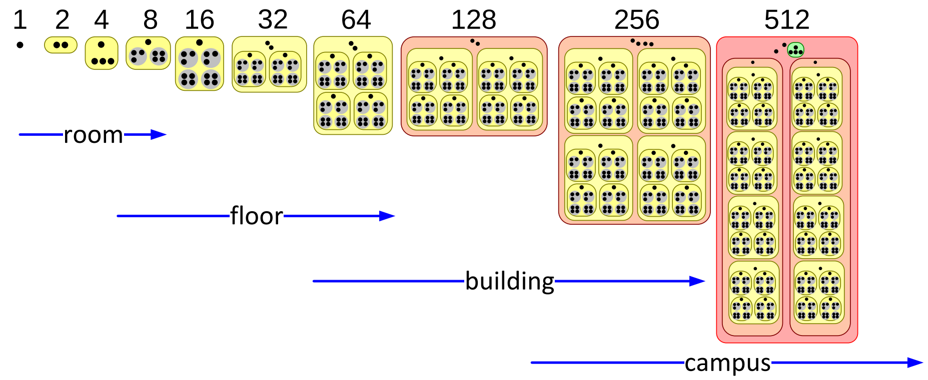


Team work



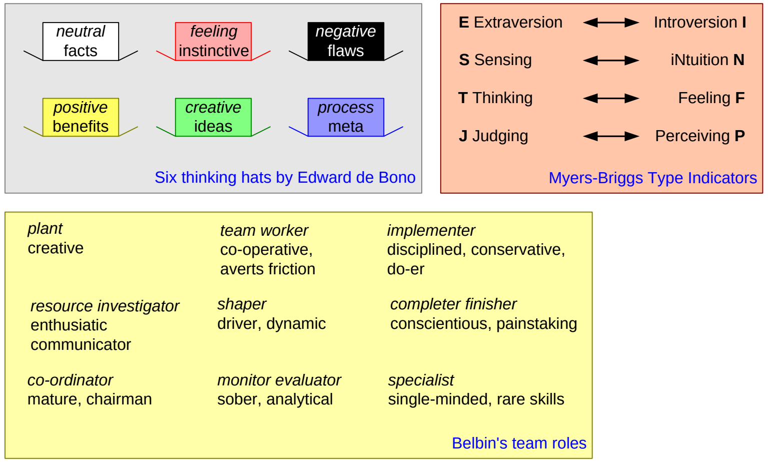
but in the team two can hear, two can see, and two can speak

Impact of Size

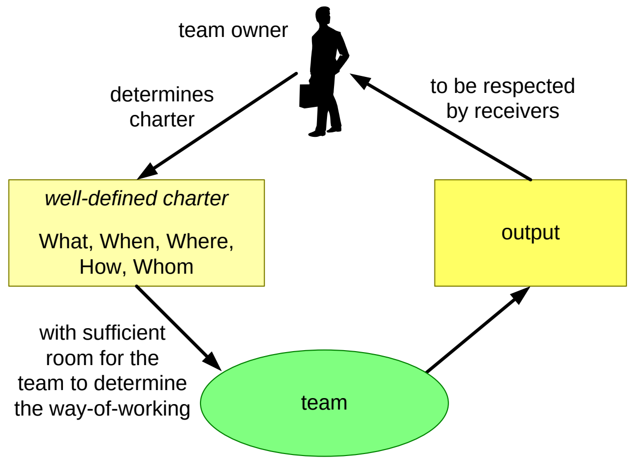


Teams

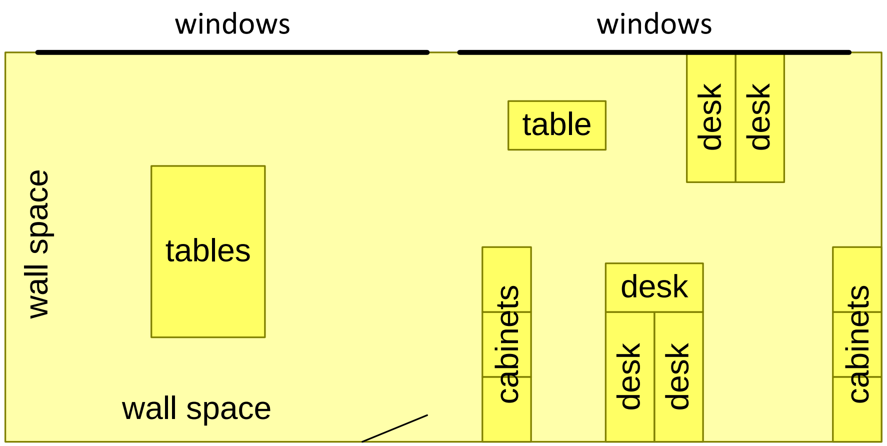
Role variations



Design the team!

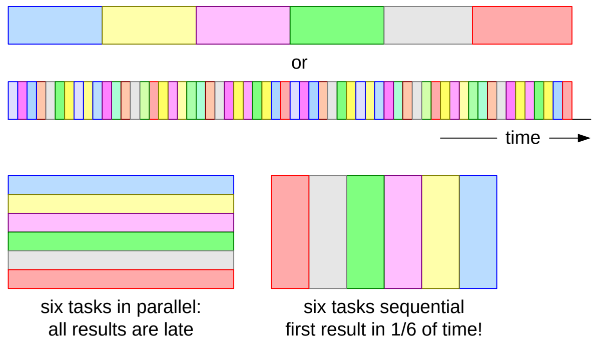


Design team environment



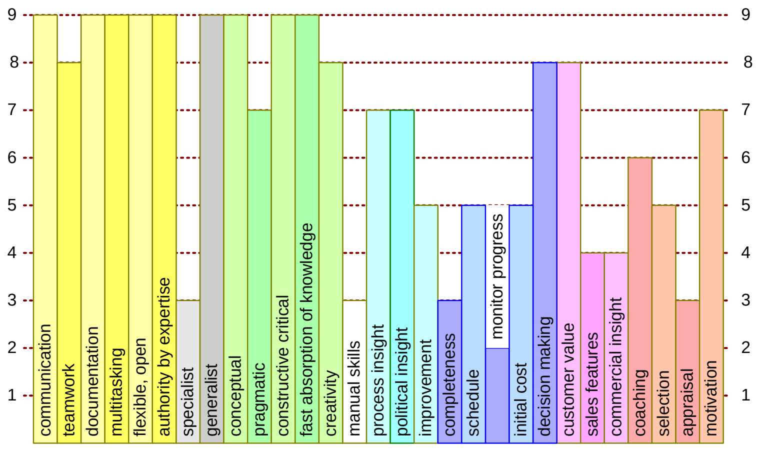
Focus, avoid fragmentation

How many (semi-)concurrent tasks can a person handle?
Working in burst-mode (concentrating on one task for one day, week or month) can increase efficiency.



Function Profiles

Sheep with 7 Legs?



intentionally left blank

intentionally left blank

intentionally left blank

Wrap Up; module 10 SARCH

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

This module addresses the Wrap Up of the course System Architecture

February 15, 2024
status: draft
version: 0.8

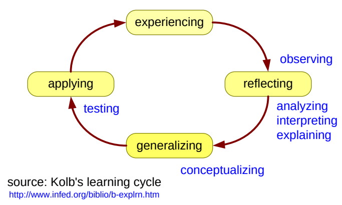
Reflection applied on Systems Architecting

by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

Reflection facilitates the learning process. We discuss a simple reflection model and provide some means for reflection.

February 15, 2024
status: preliminary draft
version: 0

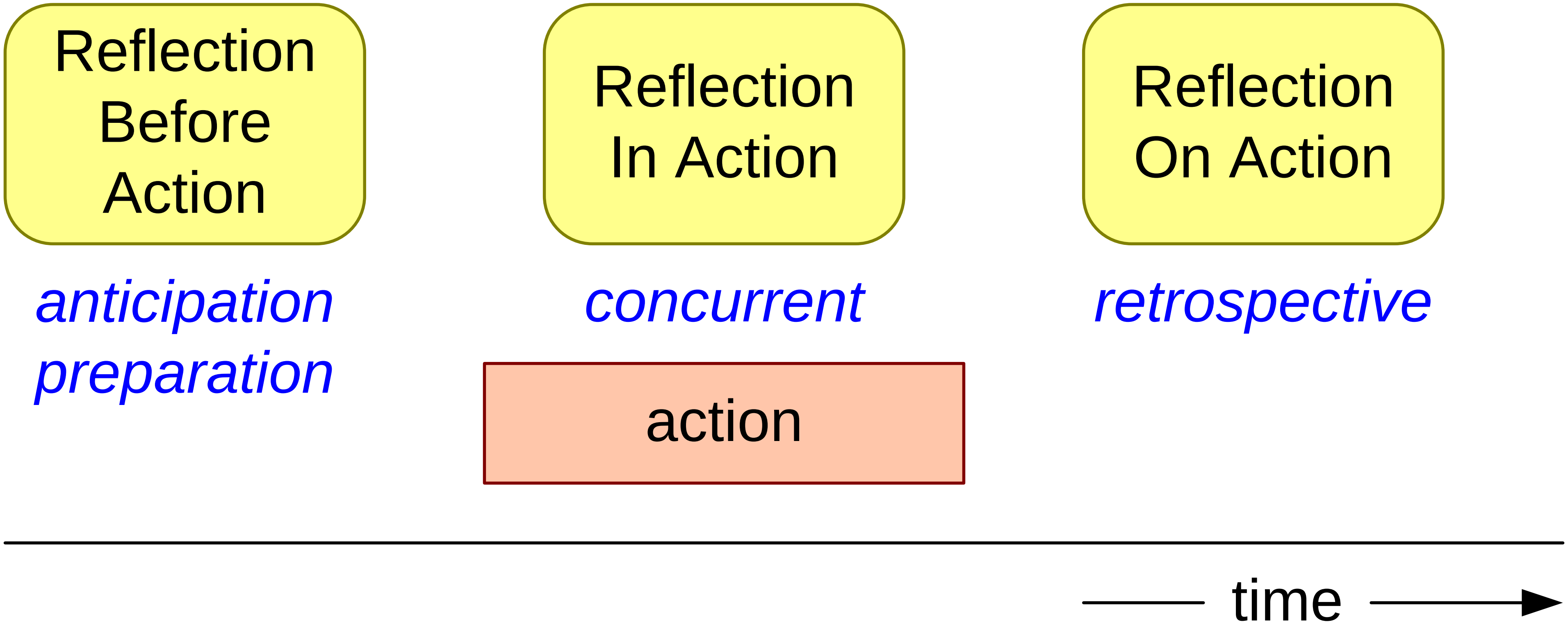


Colophon

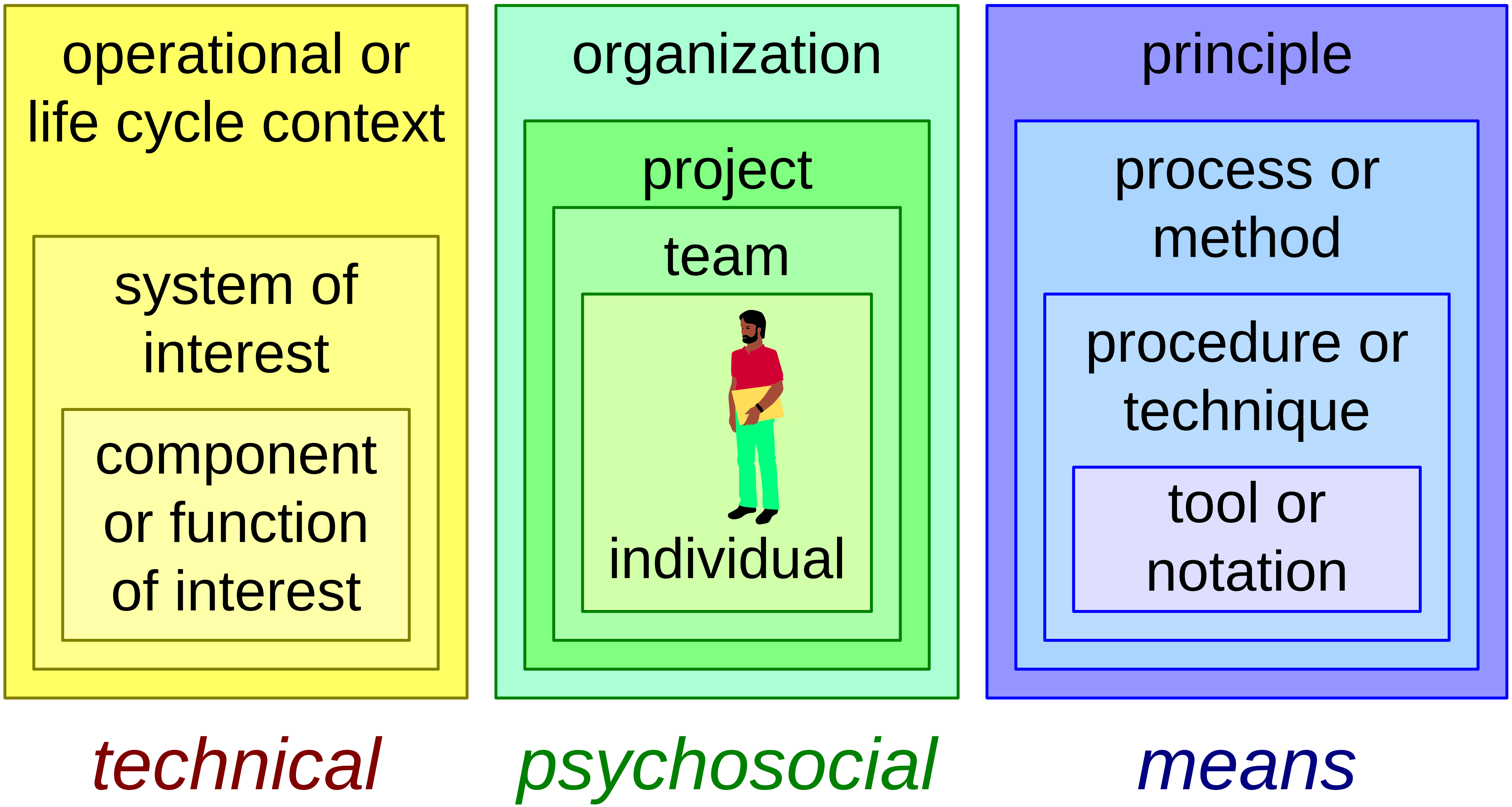
Merete Faanes from Buskerud University College created the educational flow *Reflective Practice*. Reflective Practice is a thread throughout the entire master Systems Engineering to stimulate students to relate *Education* and *Practice*.

These workshops are the result of the cooperation of Merete Faanes and Gerrit Muller

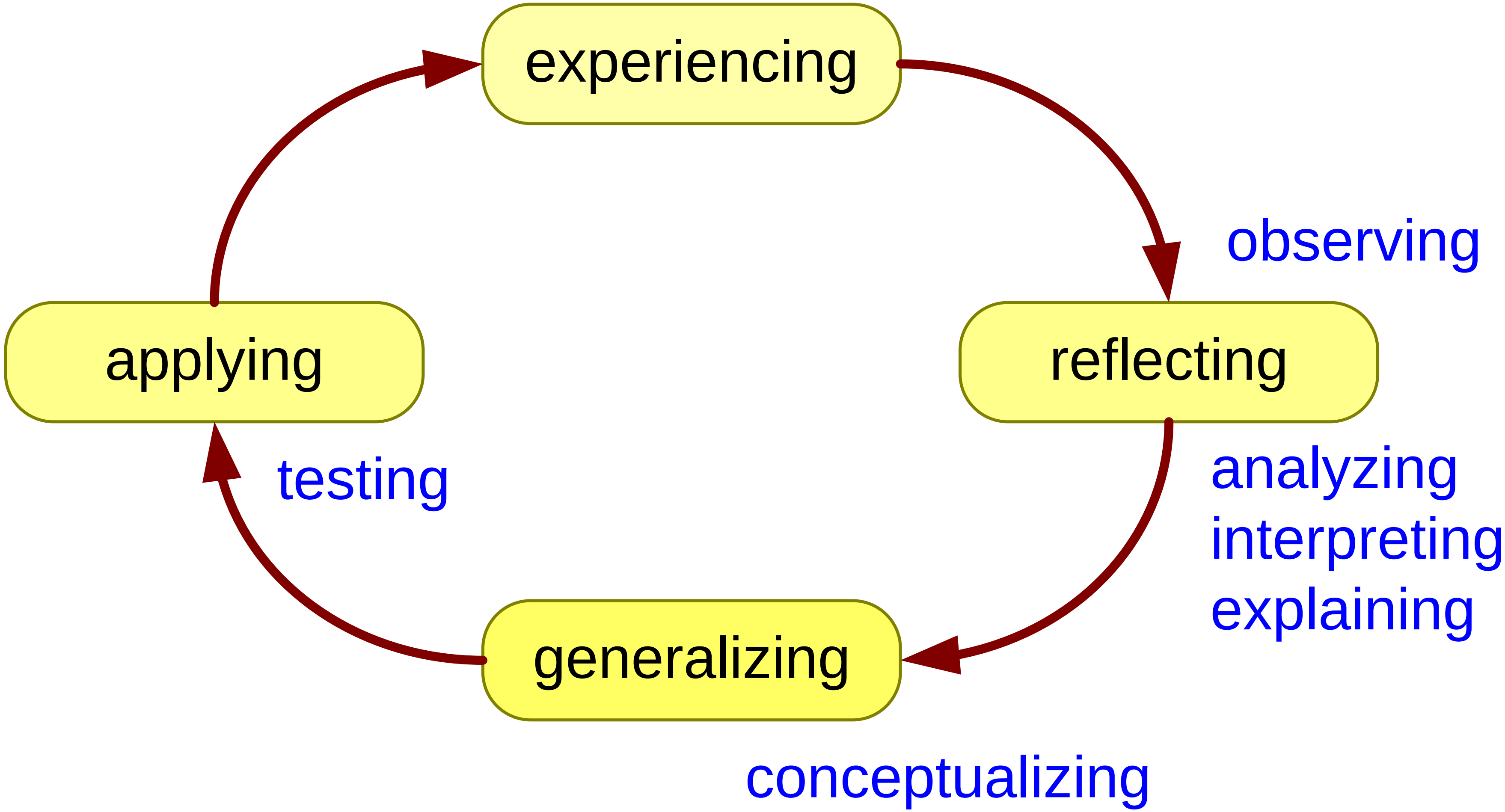
When to Reflect



Scope: What to Reflect on



Reflection Cycle



source: Kolb's learning cycle

http://www.infed.org/biblio/b-explrn.htm

Example of Reflection Questions

What stakeholders are involved?

What are their needs and concerns?

What is our goal?

How did we get in the current situation?

What is going well, what is going bad?

What approach can we take?

What do we expect to happen?

et cetera

Recommended Reflection Report Content

subject or goal

description of your experiences

analysis

lessons learned

actions as follow-up

avoid broad generic statements

illustrate with specific examples

Exercise Wrap Up

Make a personal improvement “roadmap” (a many year vision) and a personal improvement plan (feasible and visible first steps).

- Identify needed improvements, which can be influenced by yourself.
- Determine what you need to do to trigger the improvement and whom needs to be involved.
- Try to link your improvements to the rest of the business, for instance to planned products, conferences, platform releases or whatever recognizable anchor is available.

SESA Homework Assignment

- after ~3 weeks:

  a powerpoint presentation with figures, diagrams and tables of the SESA views

- after ~6 weeks:
  - a concept report with updated figures, diagrams and tables.
  - Add some explanatory text in the report.
  - Maximum size of the report 20 pages; less is better

- after ~9 weeks:
  - a complete report where the feedback on the concept report has been processed

- after 10 weeks:
  - personal reflection, plan and roadmap.

Viewpoints

- introduction, domain, company, and system-of-interest
- process and organization; how does the product/system creation process work? Diagram of the **de facto operational organization** (e.g. like the Monday morning SESA exercise). Note: no nice looking official diagrams, rather the actual situation with names. This actual situation might differ from the theory. Reflect on these differences, and the consequences.
- role and task of the system architect
- requirements management; especially a **customer key driver graph** for your system
- system architect toolkit; give examples typical tools, techniques and methods as applied on your system, and provide a **story** for your system.
- roadmapping; make a coarse **roadmap of market, product and technology** for your part of the company (in a broader context than the system only); pay special attention to the "outside" world, e.g. relevant trends.
- generic developments/product families; show and reflect on how your company tries to address similarity between systems, projects or products
- supporting processes, especially documentation
- **presentation to management**, especially high level financial figures for your system. Submit this as a separate presentation. You may use the presentation of the course itself, with updates based on the board meeting. Provide reflection on the presentation: How was the presentation in retrospect? How did the BoM respond?
- role of software in your system (so not the tools that are used in your organization)
- psycho social side

Recommendations and Guidelines

- Write a brief introduction for the assessors about your context, e.g. the domain, the company, and your system-of-interest
- Make and communicate visualizations (diagrams, figures, models, graphs) first.
- Use this assignment as opportunity to talk with other people in your organization.
- Reflect in the text on the viewpoint and its actual status; what works well, what can be improved?
- Note the maximum size of 20 pages; smaller reports get better grades :-)

Personal Plan, Roadmap,and Reflection

- in the personal plan and roadmap make sure that you relate these to your company; what does the company need and what do you want/what are your capabilities.
- the personal plan is short term oriented: what do you plan to do in the next days/weeks. Think about practical steps that allow you to learn and to earn credit.
- the personal roadmap is long term: where do you want to be in 3 to 5 years? How does this fit in your company? What steps are required?
- personal reflection max 1 A4, personal plan max 1 A4, personal roadmap max 1 A4.

Submission Instructions

Submission instructions

use for all deliverables the following conventions:

filename: SESA <your name> <subject> .<version>.<extension>

e.g. SESA John Student preassignment My Role.2.doc

where subject = {report | plan | ...}

email to: <gerrit . muller@ gmail.com>

subject: SESA <subject>

and submit in WiseFlow before the deadline.

"standard" file types preferred, e.g. pdf, jpg, doc, ppt, vsd, docx, xls, xlsx, ppt, pptx

Note: intermediate submissions are mandatory