All About Systems Engineering; Introductory Course

by *Gerrit Muller*

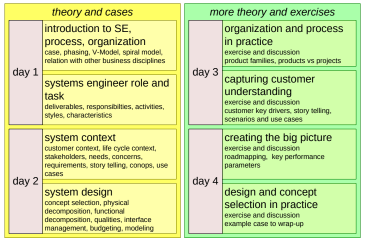
University of South-Eastern Norway-NISE

Abstract

This introductory course sketches all fundamentals of Systems Engineering. Starting at the business contexts, touching Project, Processes, and Organization. The role of the Systems Engineer is discussed, and the relation with other roles, e.g. project leader and product manager. The architecting and design tools are shown; from Stakeholder Needs to Requirements to Modeling and Analysis.

Distribution

November 18, 2023
status: preliminary draft
version: 0.2


theory and cases
day 1
introduction to SE, process, organization
case, phasing, V-Model, spiral model, relation with other business disciplines
systems engineer role and task
deliverables, responsibilties, activities, styles, characteristics
day 2
system context
customer context, life cycle context, stakeholders, needs, concerns, requirements, story telling, conops, use cases
system design
concept selection, physical decomposition, functional decomposition, qualities, interface management, budgeting, modeling
more theory and exercises
day 3
organization and process in practice
exercise and discussion
product families, products vs projects
capturing customer understanding
exercise and discussion
customer key drivers, story telling, scenarios and use cases
day 4
creating the big picture
exercise and discussion
roadmapping, key performance parameters
design and concept selection in practice
exercise and discussion
example case to wrap-up

Introduction Course Program

day 1

morning

introduction to SE, process, organization

case, phasing, V-Model, spiral model, relation with other business disciplines

afternoon

systems engineer role and task

deliverables, responsibilties, activities, styles, characteristics

day 2

morning

system context

customer context, life cycle context, stakeholders, needs, concerns, requirements, story telling, conops, use cases

afternoon

system design

concept selection, physical decomposition, functional decomposition, qualities, interface management, budgeting, modeling

Project Systems Engineering Introduction; Phasing, Process, Organization

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

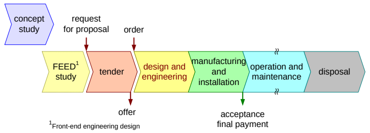
e-mail: gaudisite@gmail.com

www.gaudisite.nl

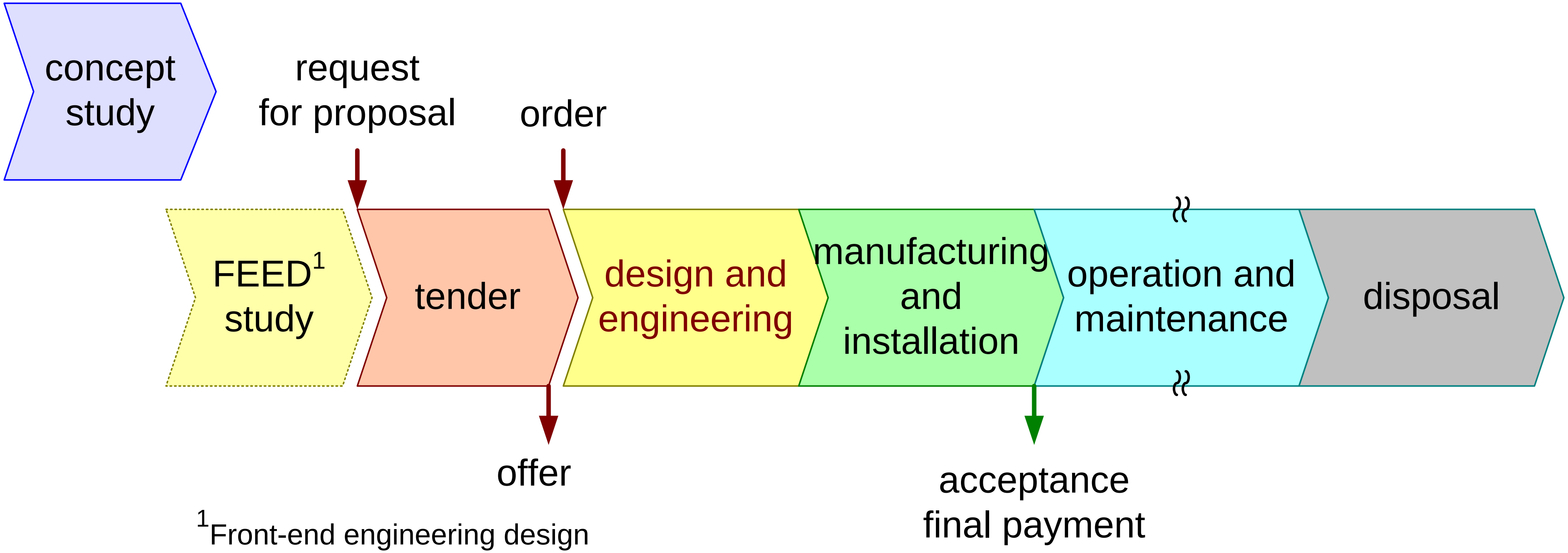
Abstract

The fundamental concepts and approach to project oriented Systems Engineering are explained. We look at project phasing, phase transition, processes, and organization.

November 18, 2023
status: preliminary draft
version: 0.2

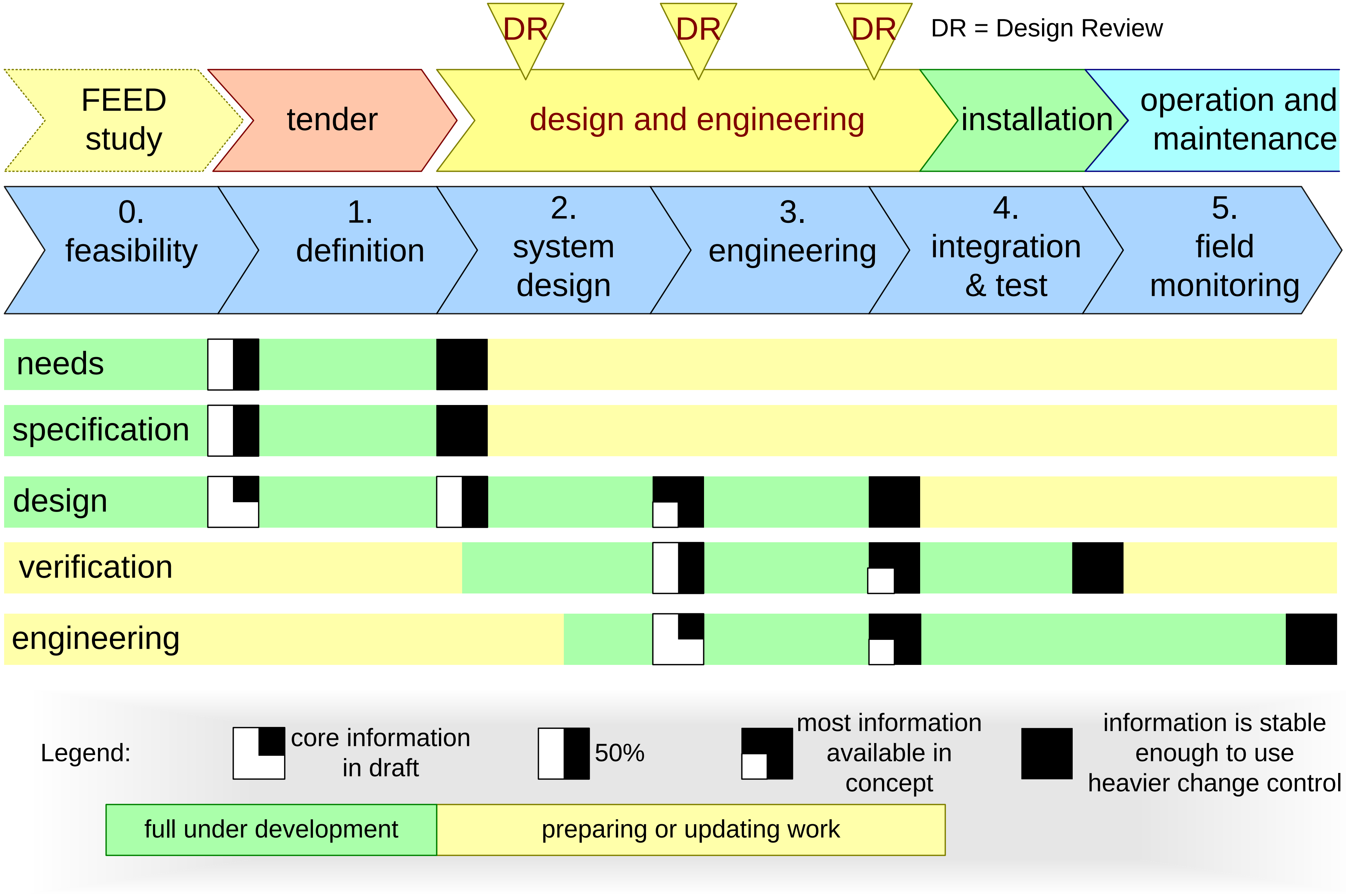


Project Life Cycle

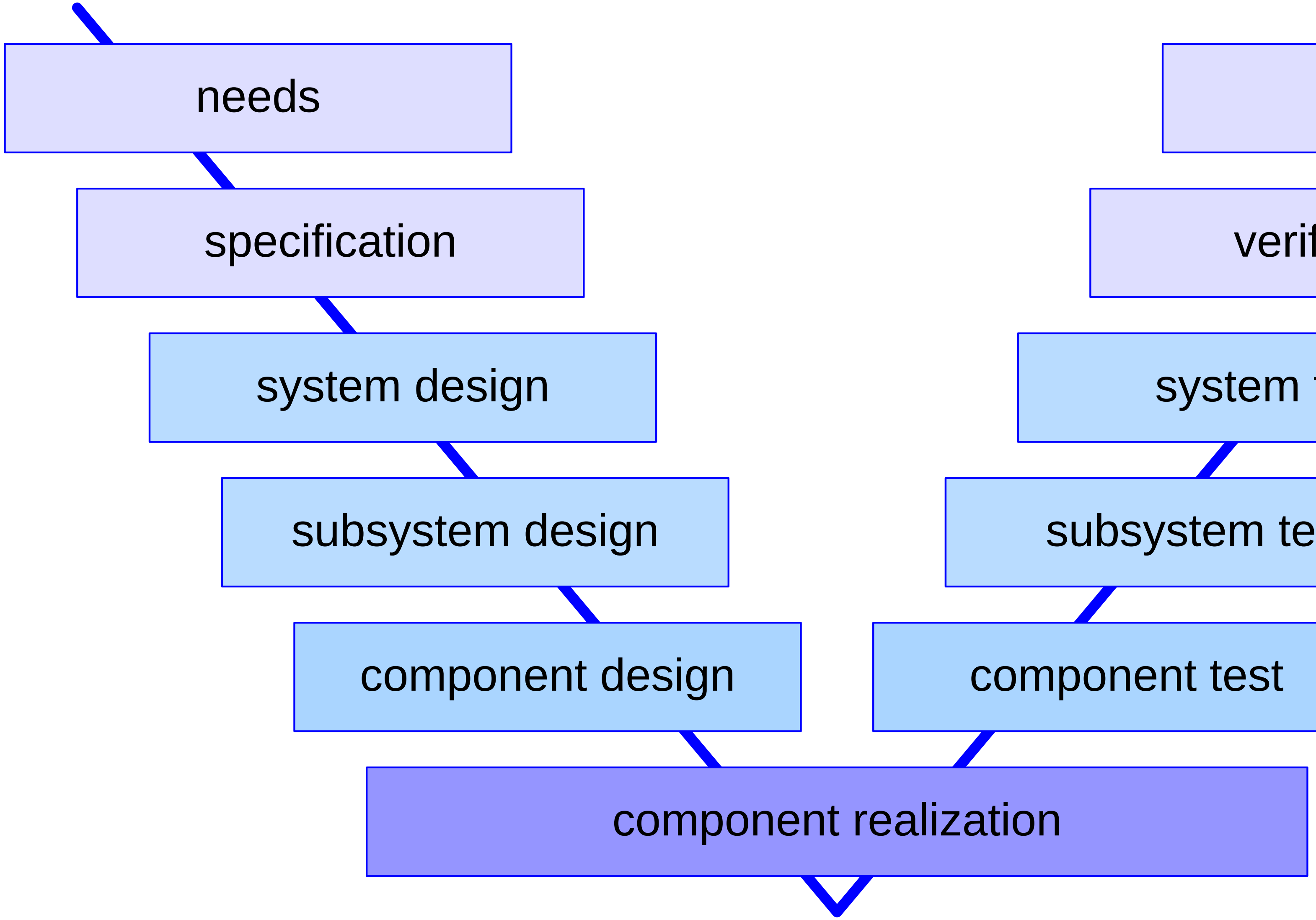


[1] Front-end engineering design

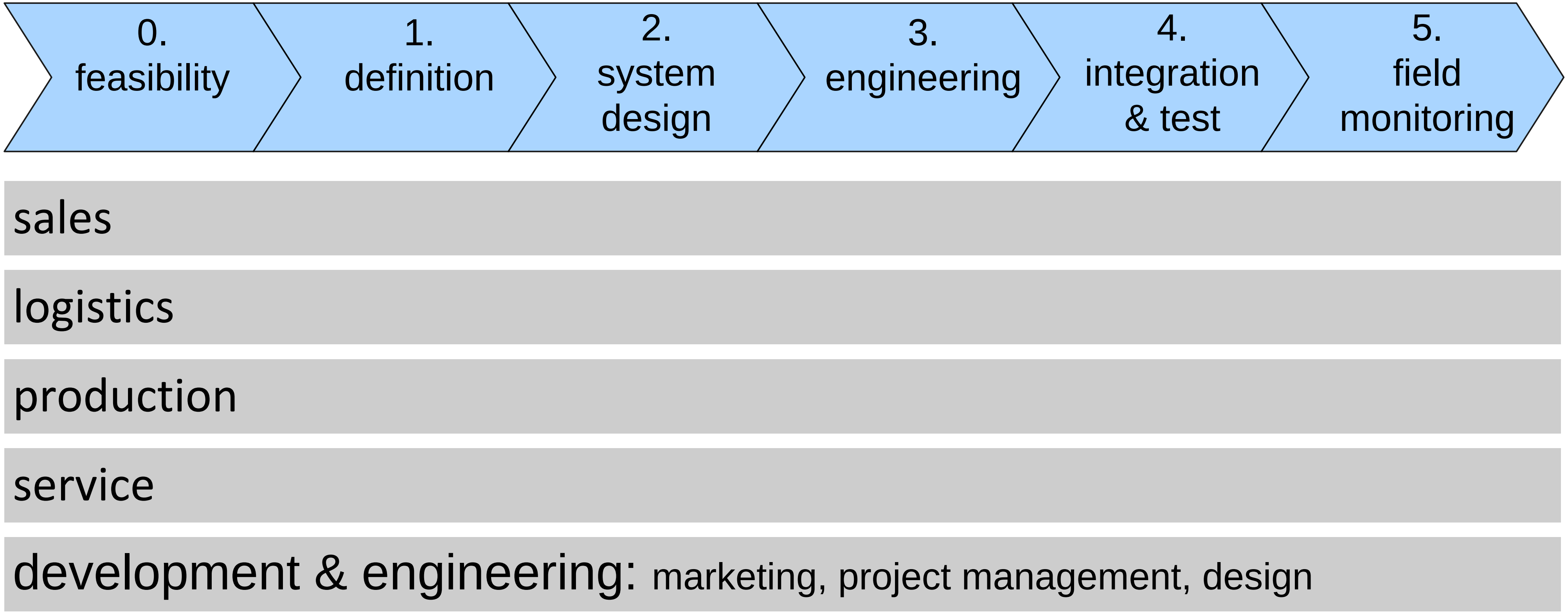
Phased Project Approach



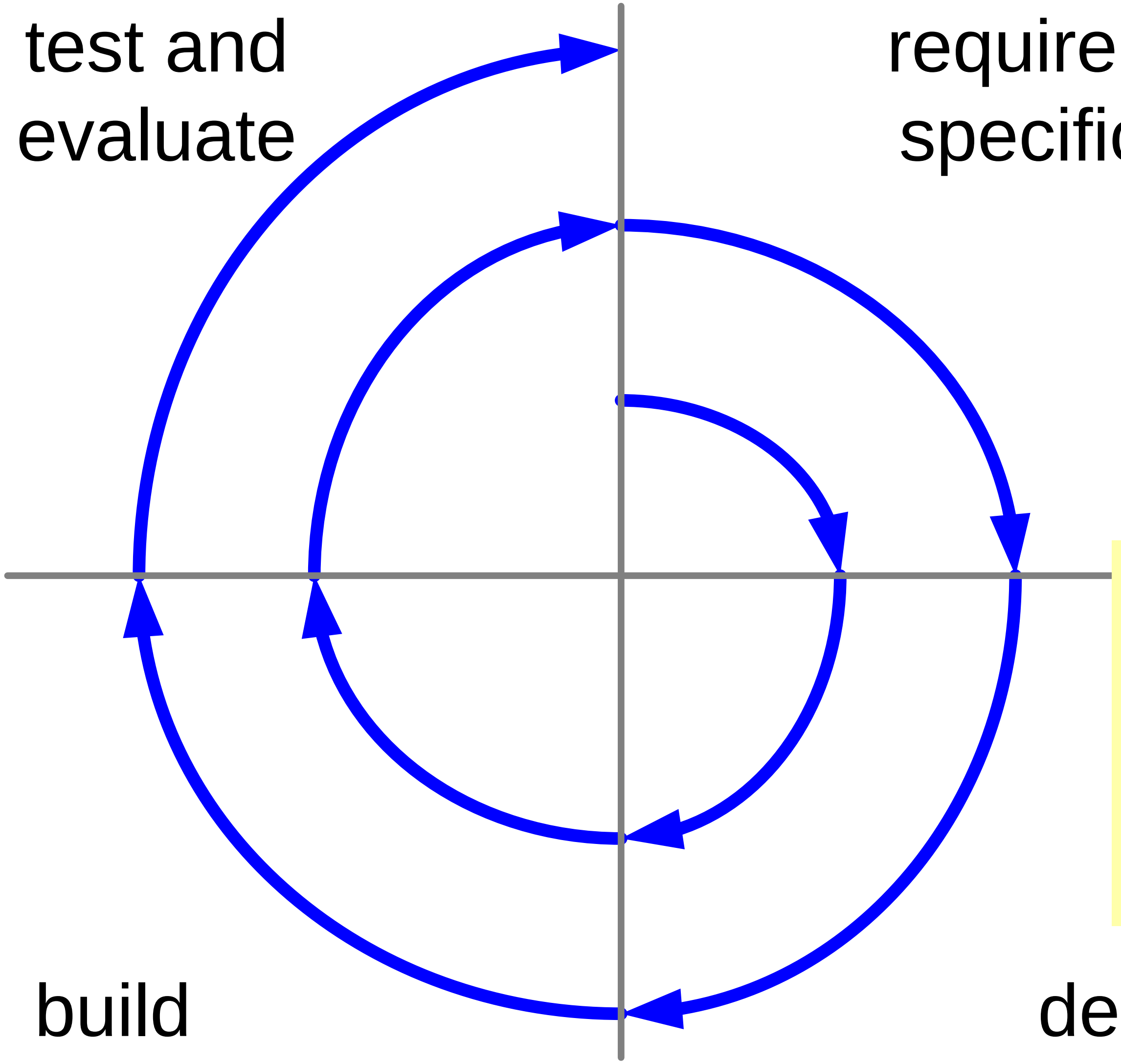
V-Model


needs
specification
system design
subsystem design
component design
component realization
component test
subsystem test
system test
verification
validation

All Business Functions Participate

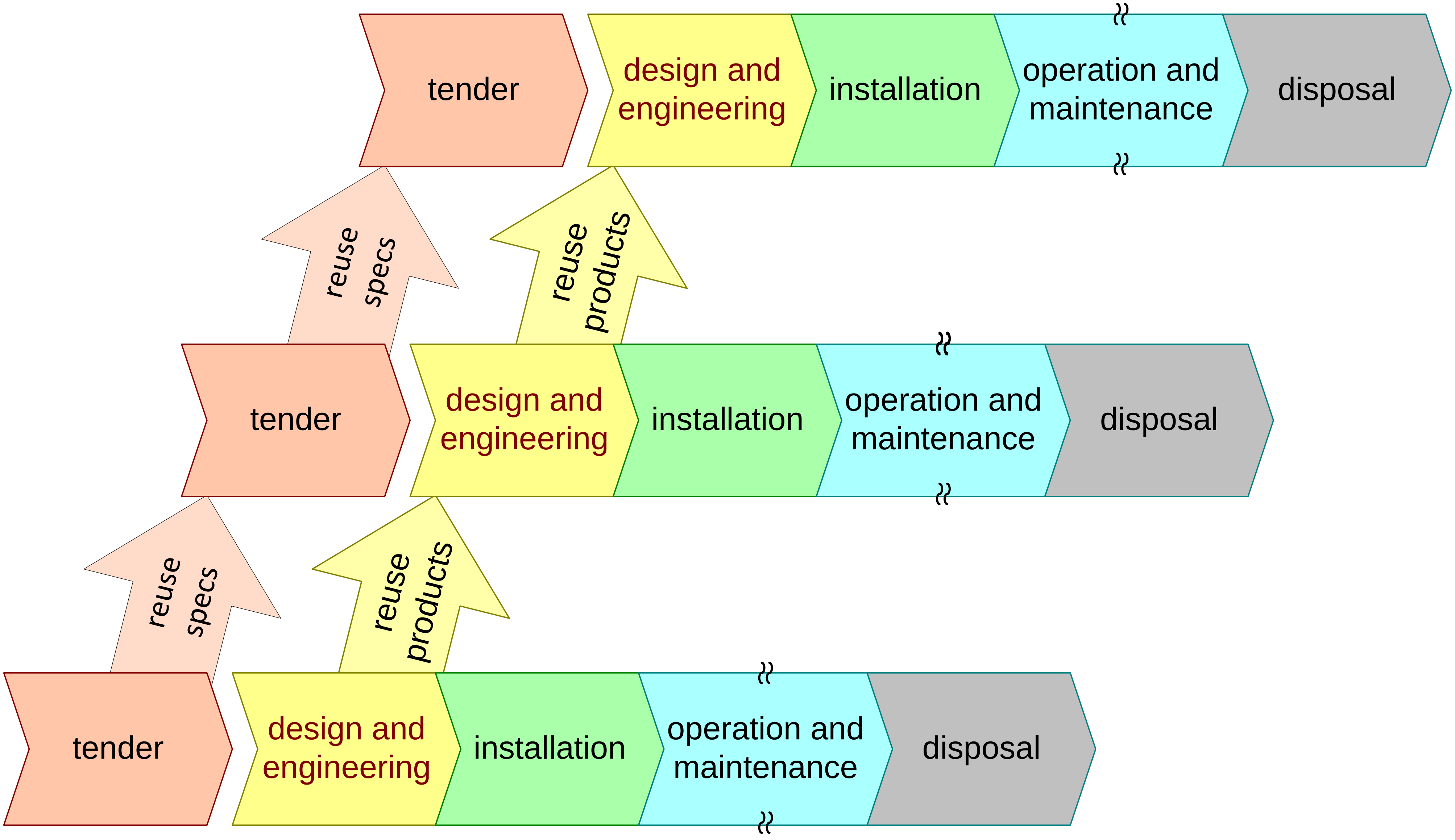


Evolutionary PCP model

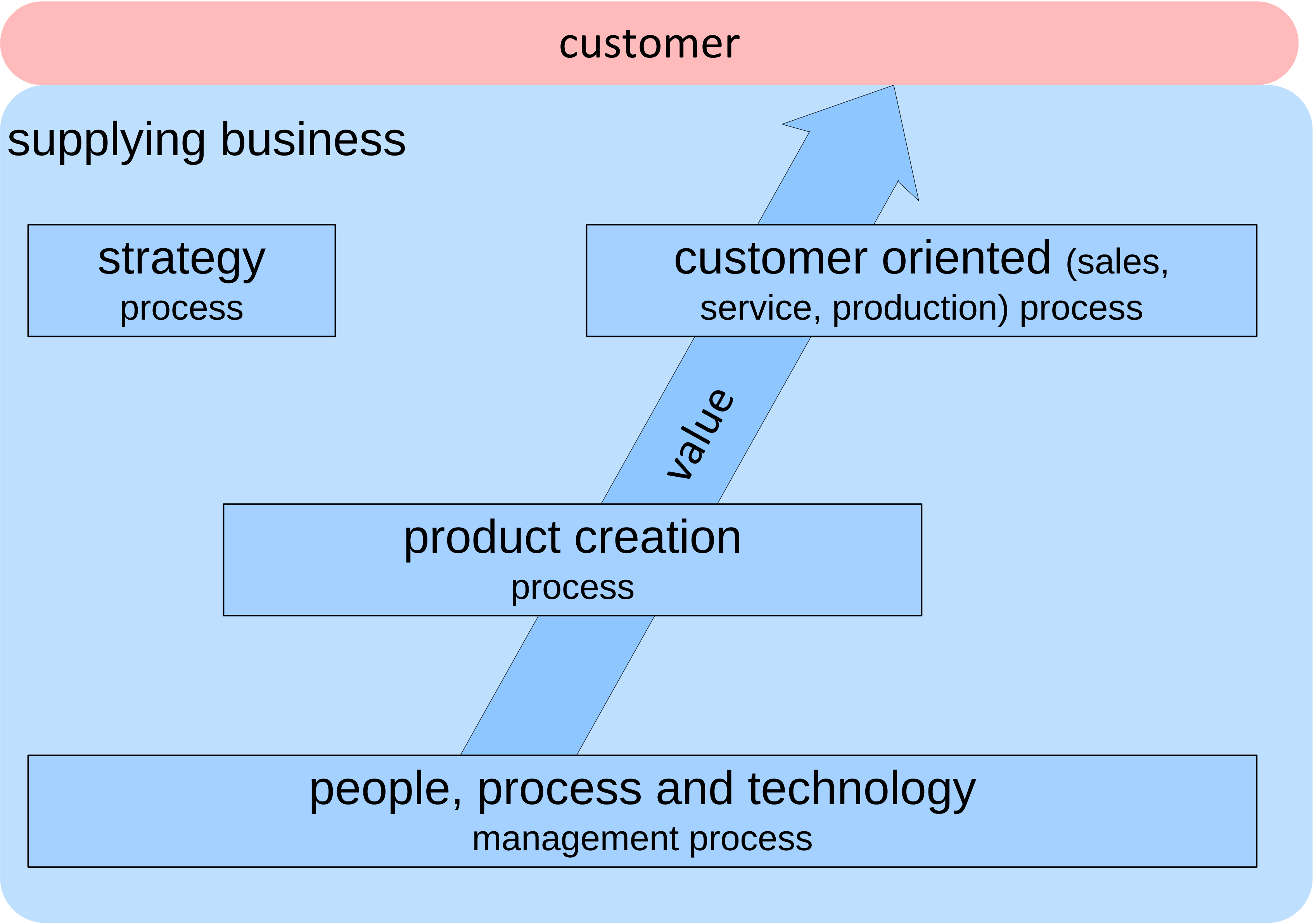


2% of budget (EVO)
2 weeks (XP)
up to 2 months
per cyclus

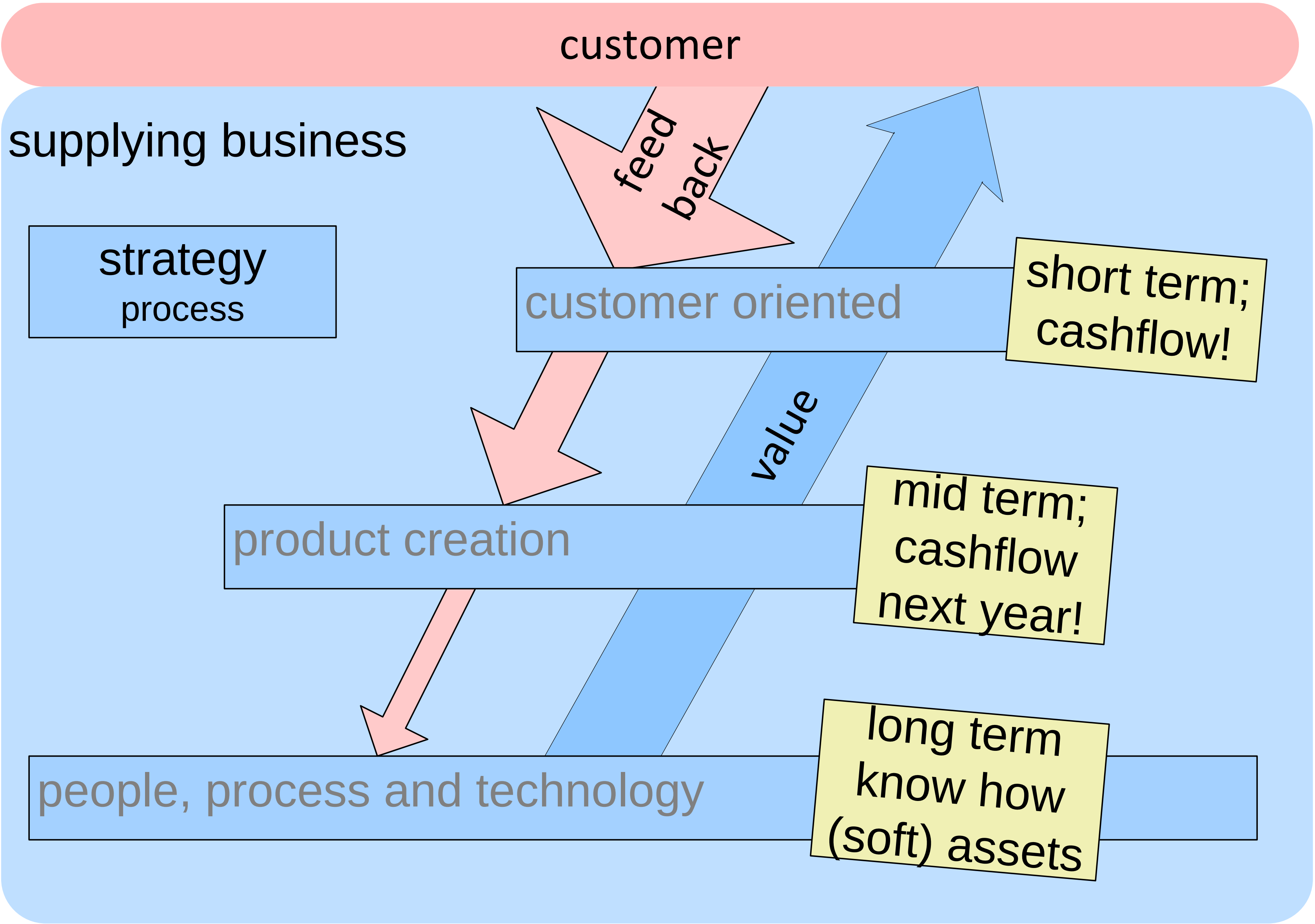
Reuse and Products



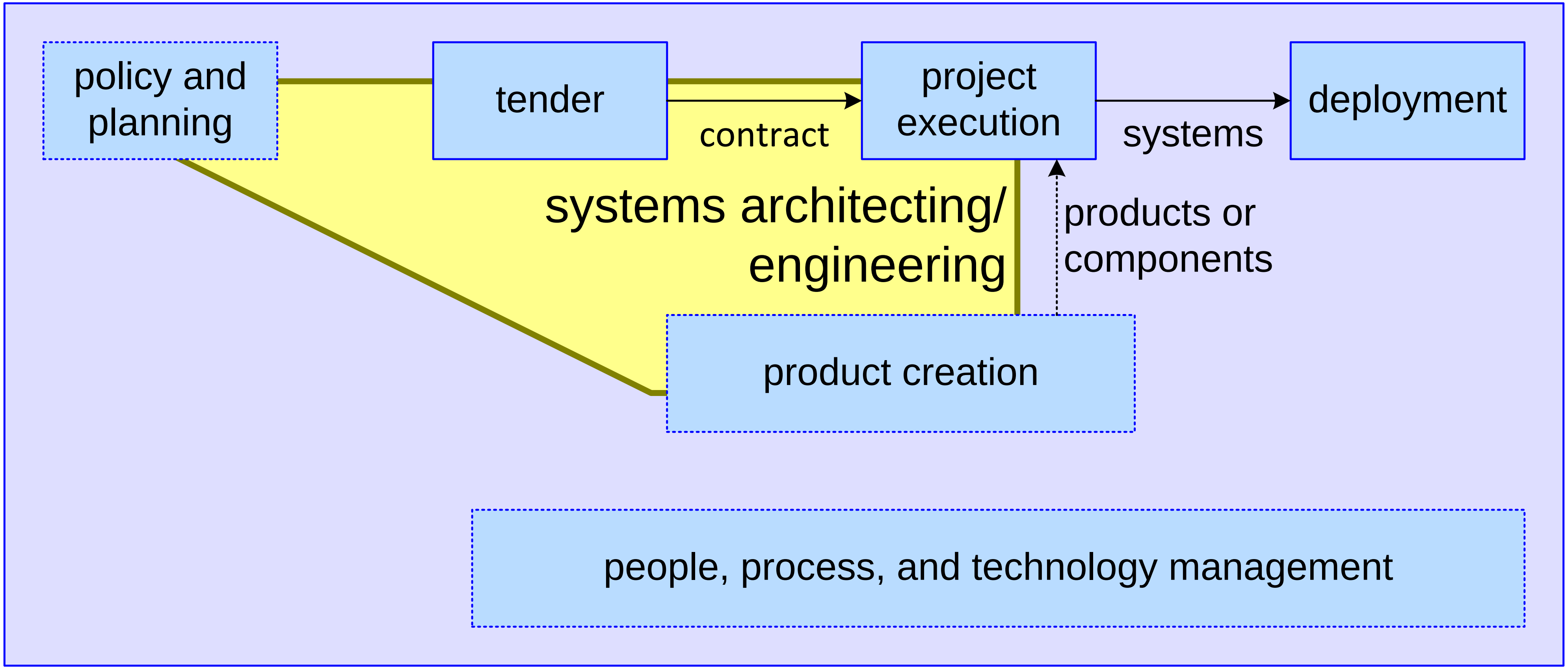
Simplified Process View



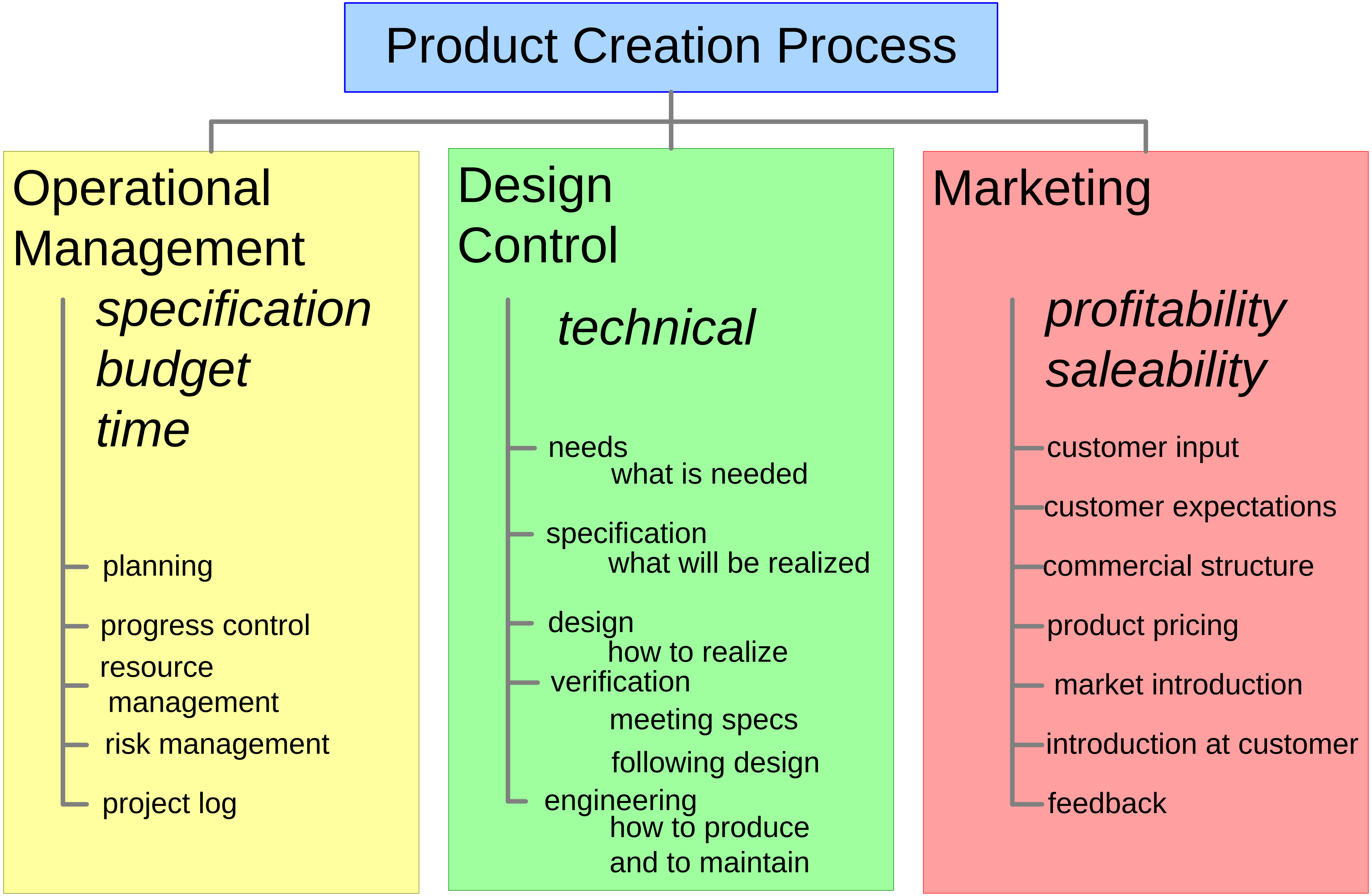
Simplified Process; Money and Feedback



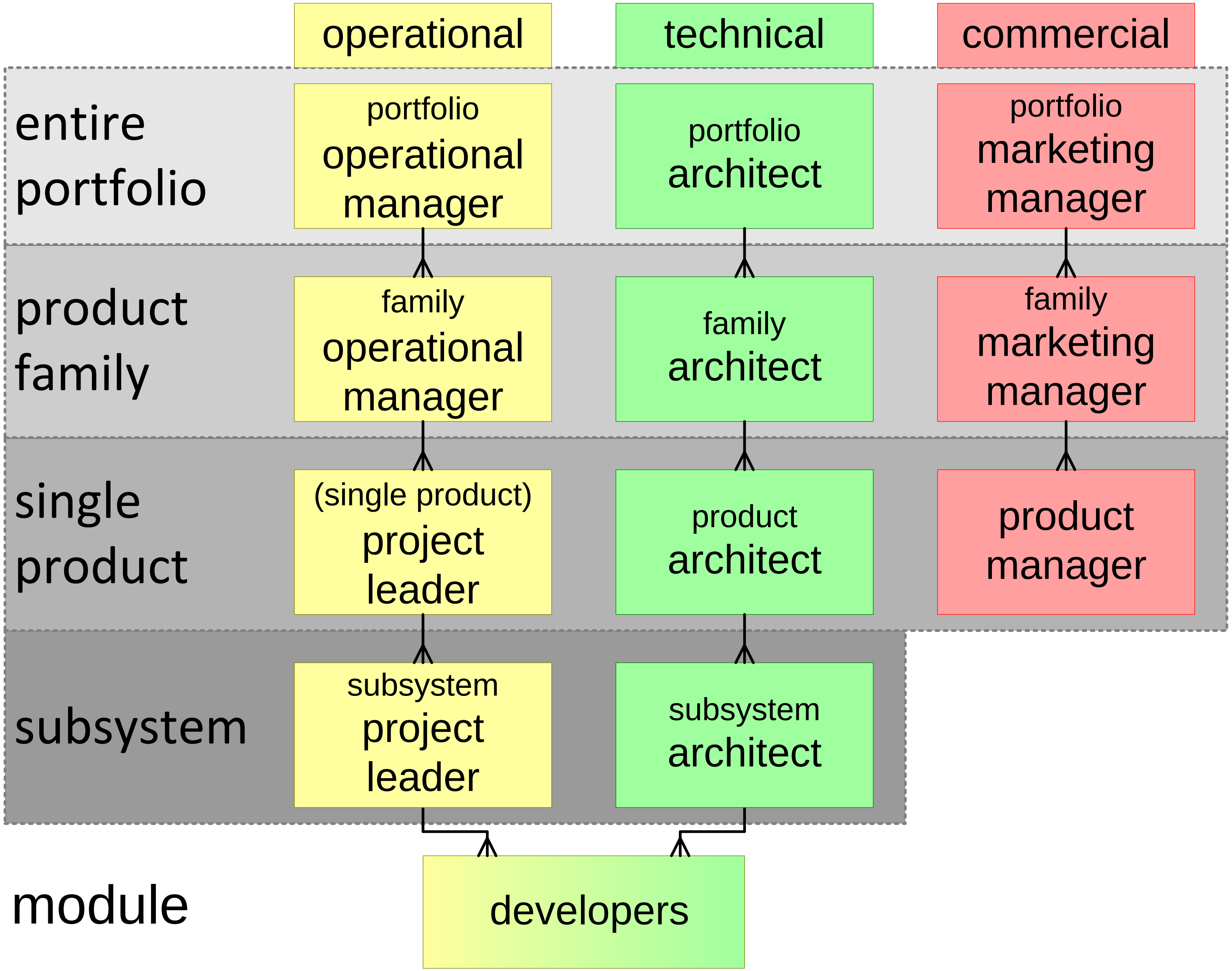
Simplified process diagram for project business



Decomposition of the Product Creation Process

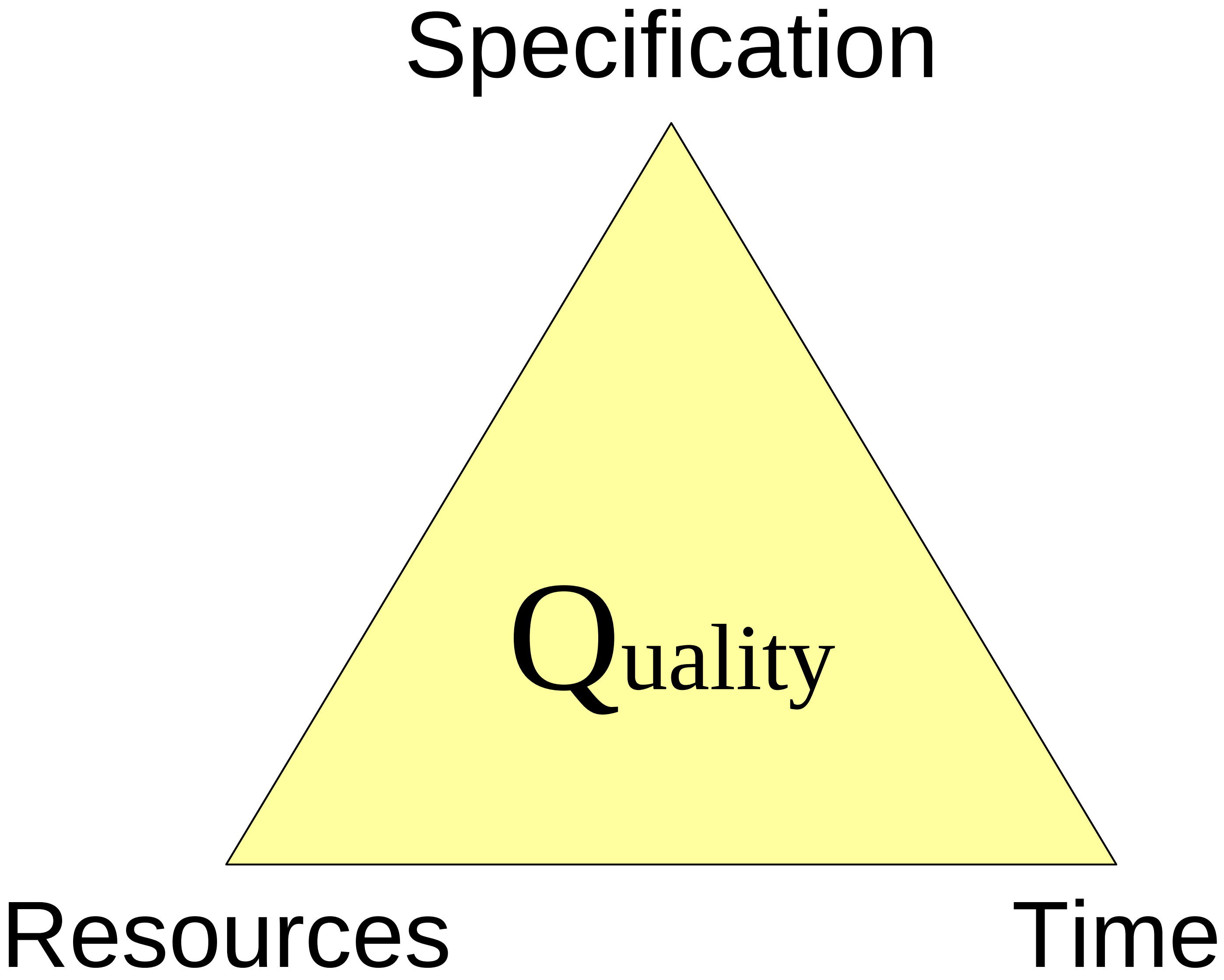


Operational Organization of the PCP

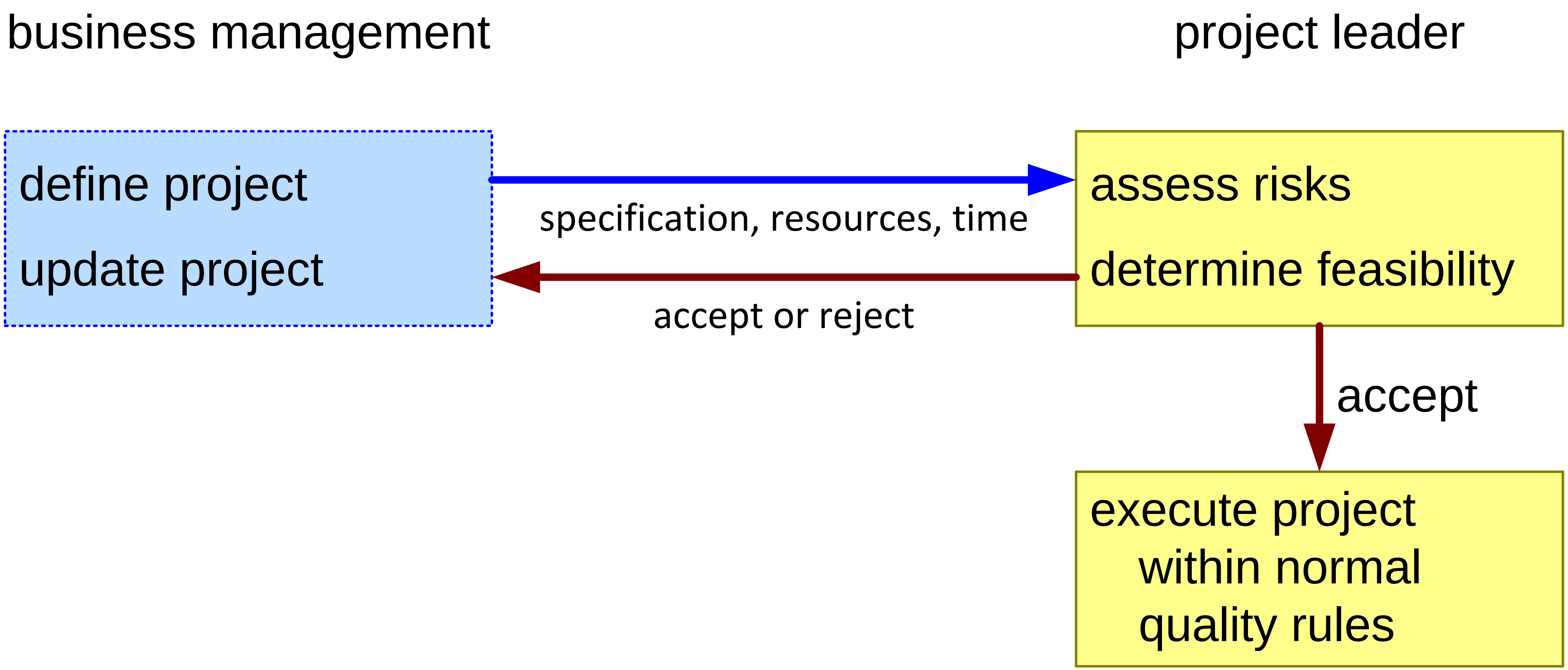


	operational	technical	commercial
entire portfolio	portfolio operational manager	portfolio architect	portfolio marketing manager
product family	family operational manager	family architect	family marketing manager
single product	(single product) project leader	product architect	product manager
subsystem	subsystem project leader	subsystem architect	
module	developers	developers	

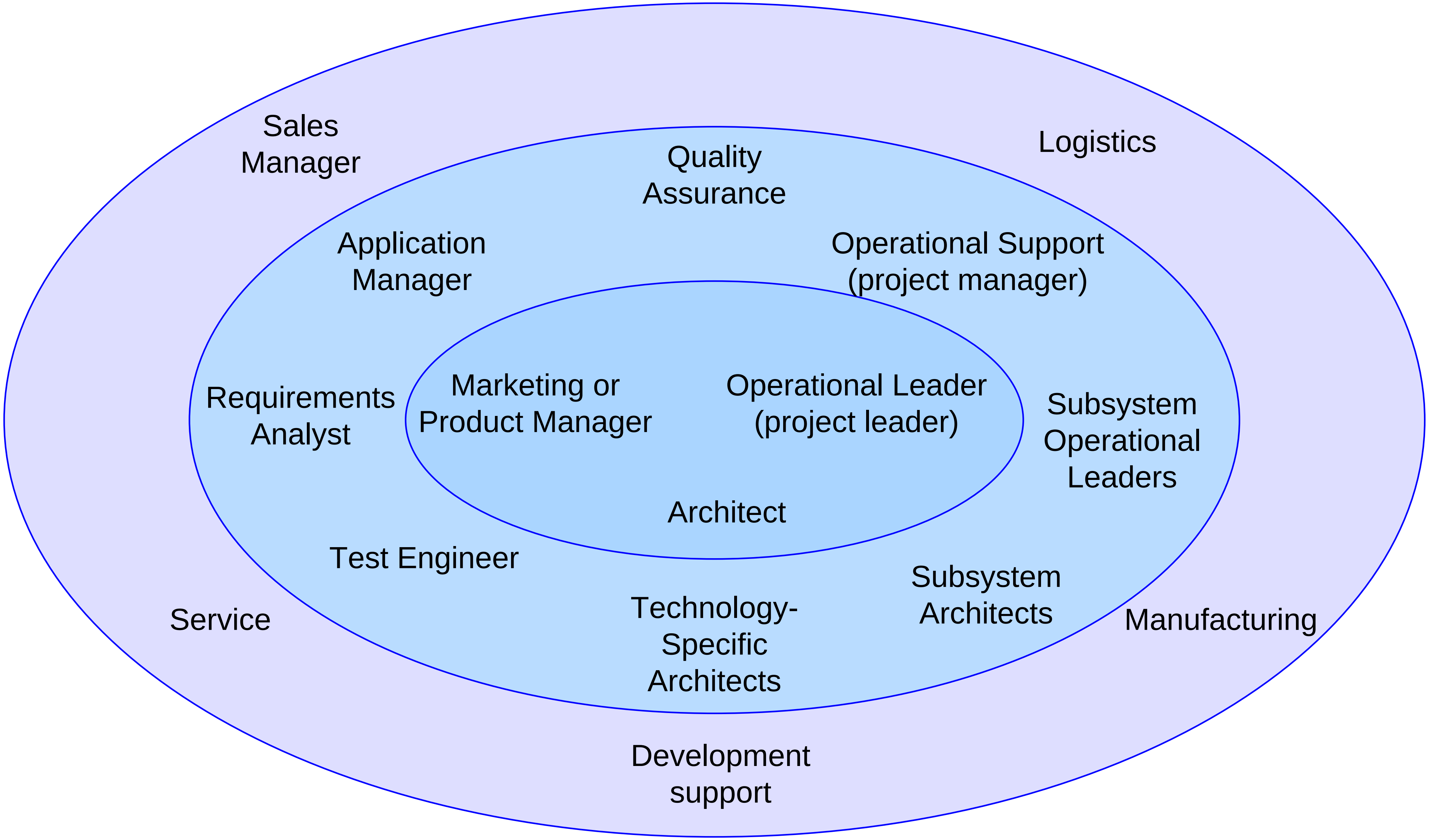
Prime Responsibilities of the Operational Leader



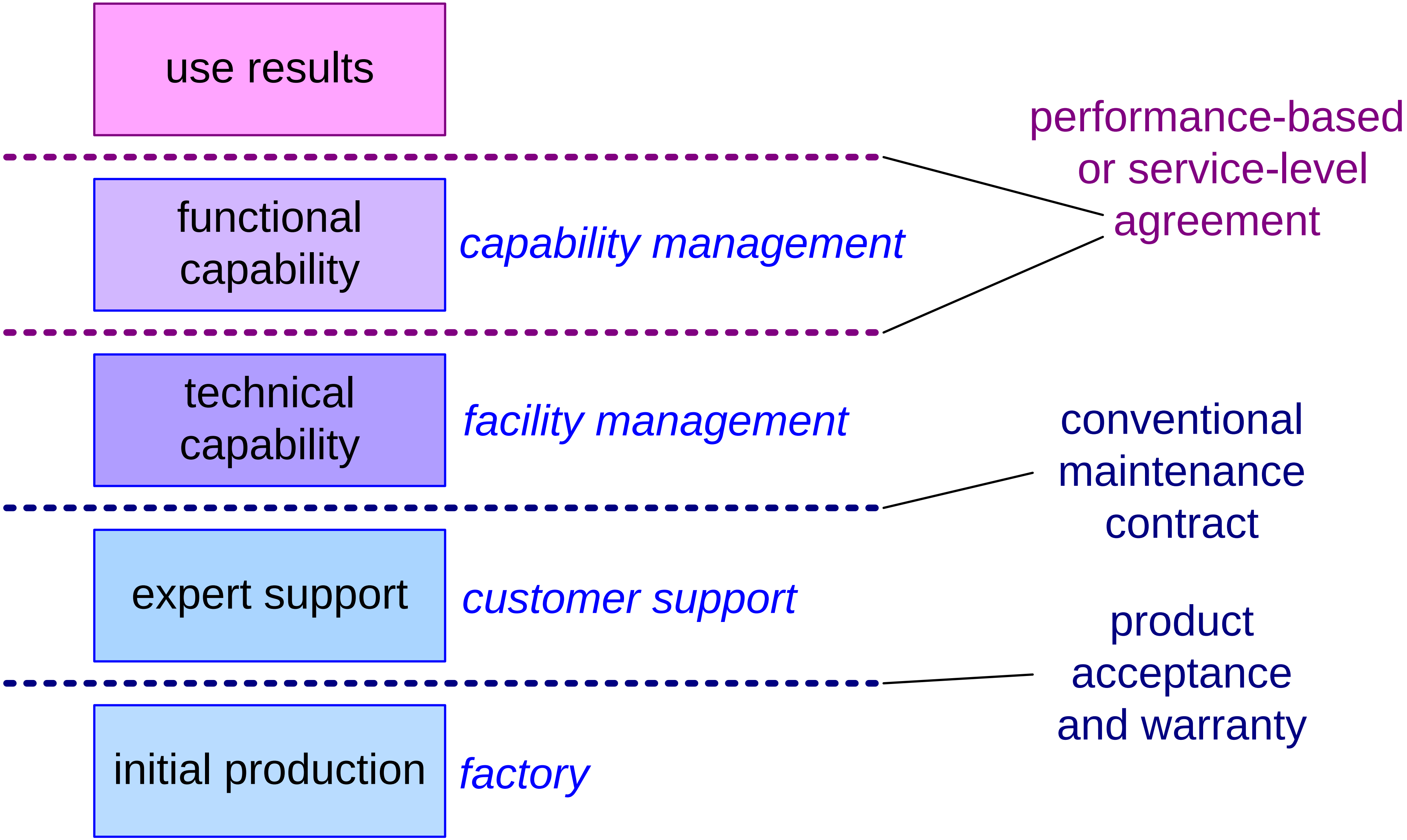
The Rules of the Operational Game



Operational Teams


Sales Manager
Logistics
Quality Assurance
Application Manager
Operational Support (project manager)
Marketing or Product Manager
Operational Leader (project leader)
Requirements Analyst
Subsystem Operational Leaders
Architect
Test Engineer
Subsystem Architects
Service
Technology-Specific Architects
Manufacturing
Development support

What Service Level to Deliver?



Systems Engineering Management Plan (SEMP)

How the project will perform the systems engineering process:

- main events and activities
- roles and responsibilities
- work products
- procedures and standards

Bridge between project management and engineering (NASA 2016)

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.

Distribution

November 18, 2023
status: preliminary draft
version: 1.0

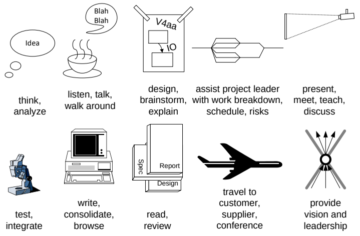
The Role and Task of the System Architect

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

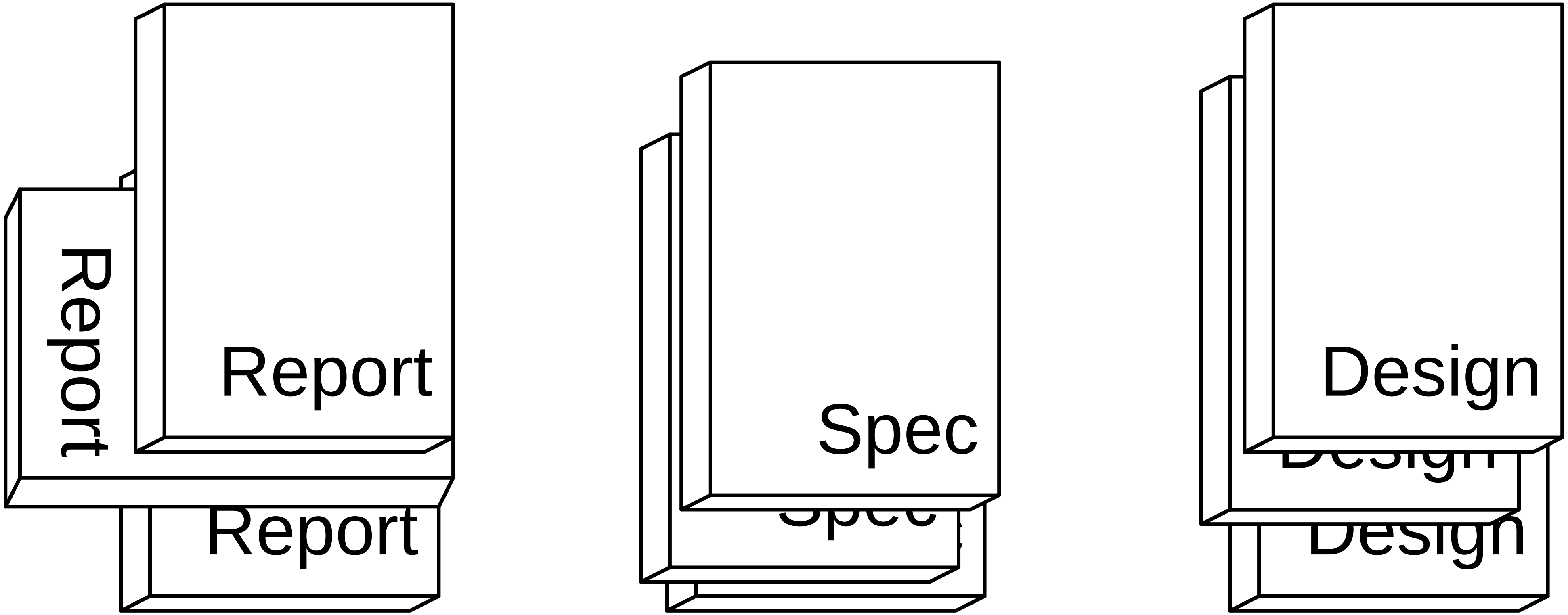
Abstract

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.

November 18, 2023
status: concept
version: 2.0


Idea
Blah
Blah
V4aa
think, analyze
listen, talk, walk around
design, brainstorm, explain
assist project leader with work breakdown, schedule, risks
present, meet, teach, discuss
Spec
Report
Design
test, integrate
write, consolidate, browse
read, review
travel to customer, supplier, conference
provide vision and leadership

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)*

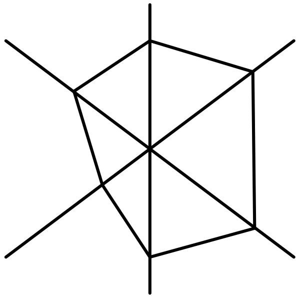
Verification Specification *(how the system will be verified)*

Verification Report *(the result of the verification)*

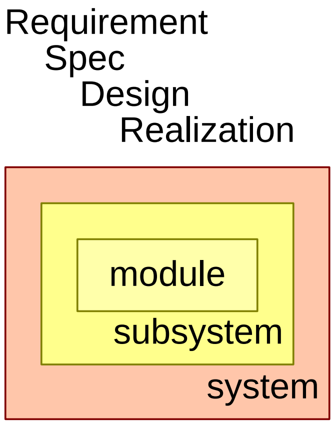
Feasibility Report *(the results of a feasibility study)*

Roadmap

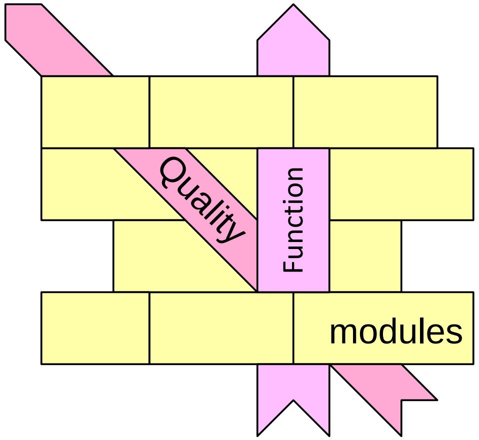
Responsibilities of the System Architect



Balance

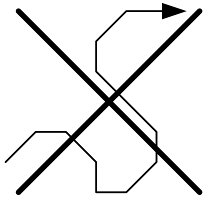


Consistency



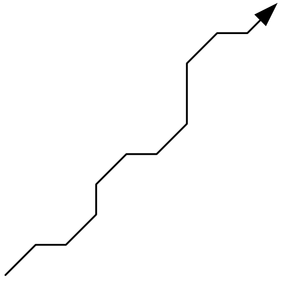
Decomposition
Integration

Overview

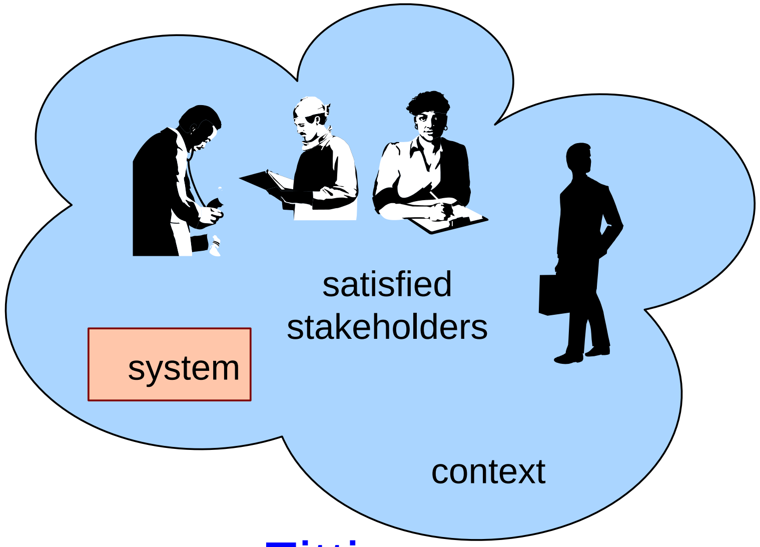


KISS

Elegance
Simple



Integrity

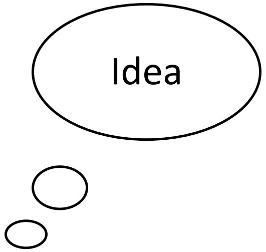


Fitting

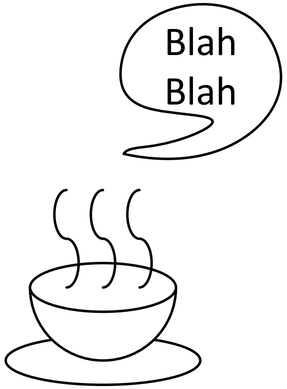
Examples of Secondary Responsibilities

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

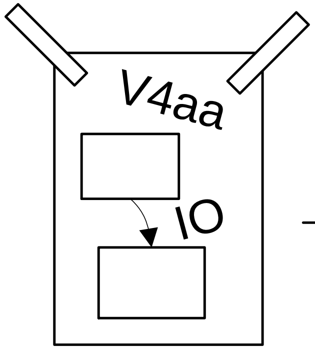
What does the System Architect do?



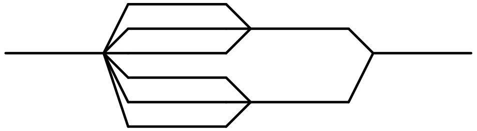
think, analyze



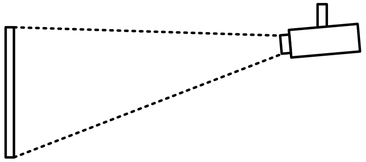
listen, talk, walk around



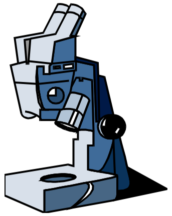
design, brainstorm, explain



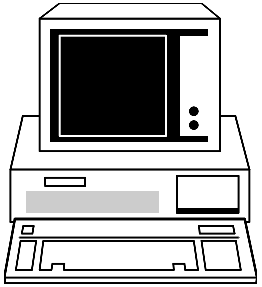
assist project leader with work breakdown, schedule, risks



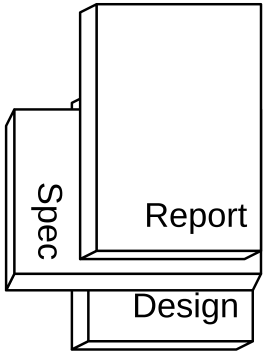
present, meet, teach, discuss



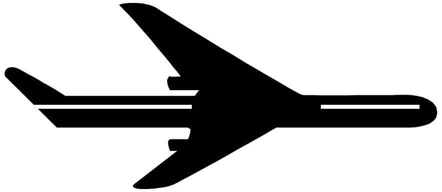
test, integrate



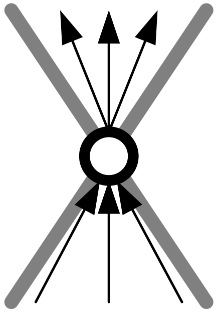
write, consolidate, browse



read, review

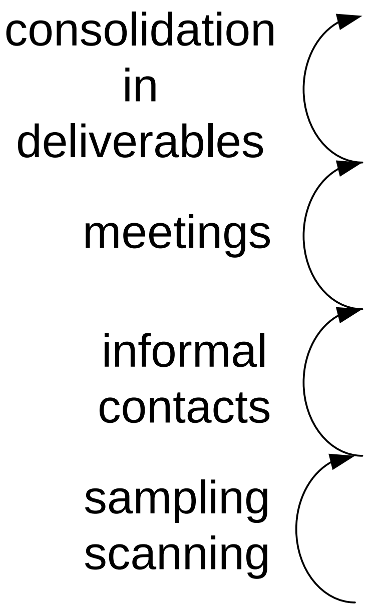


travel to customer, supplier, conference



provide vision and leadership

From Detail to Overview



consolidation in deliverables

meetings

informal contacts

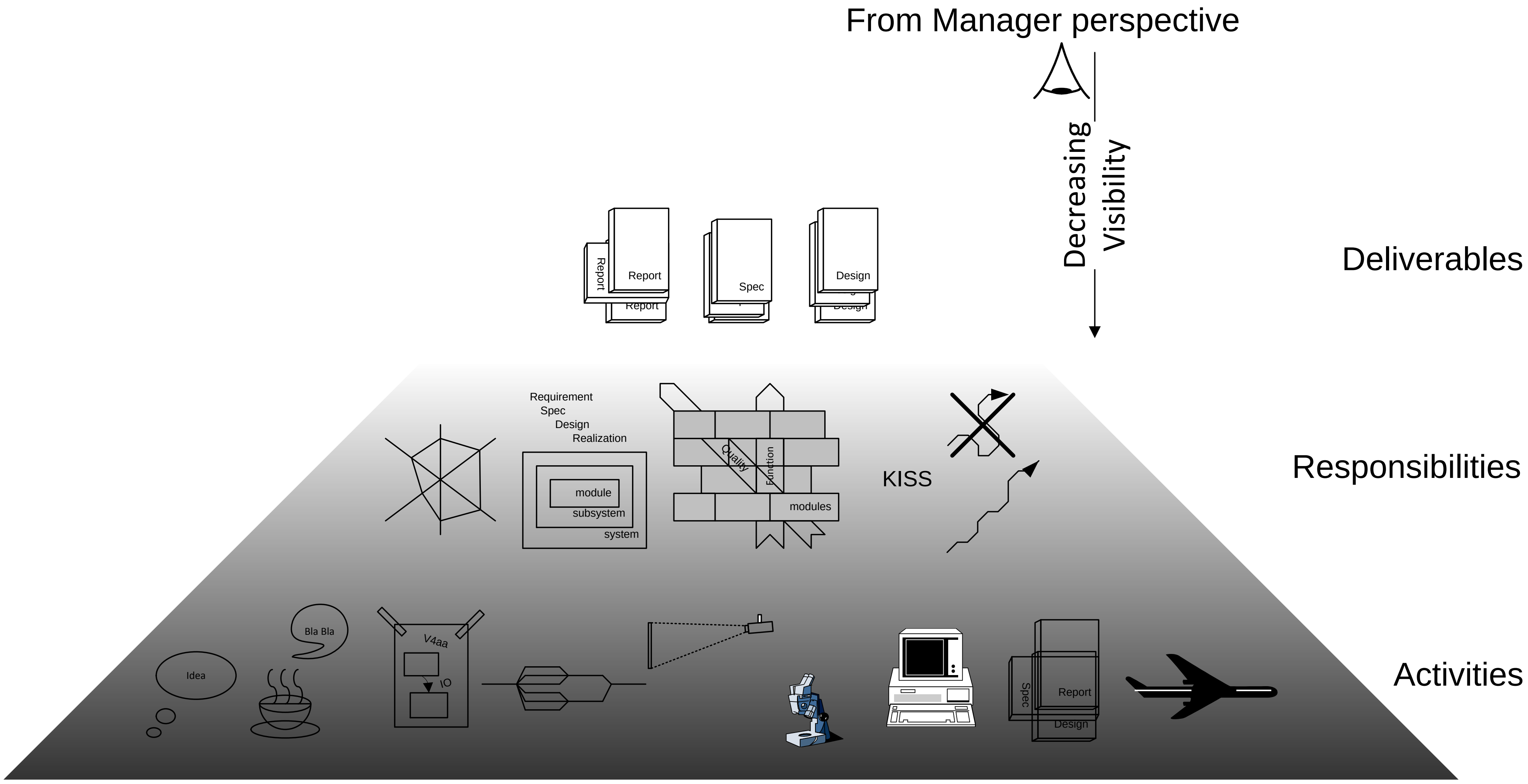
sampling scanning

	Quantity per year (order-of-magnitude)	architect time per item
driving views	10	100 h
shared issues	10^2	1 h
touched details	10^4	0.5 - 10 min
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

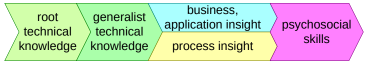
by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

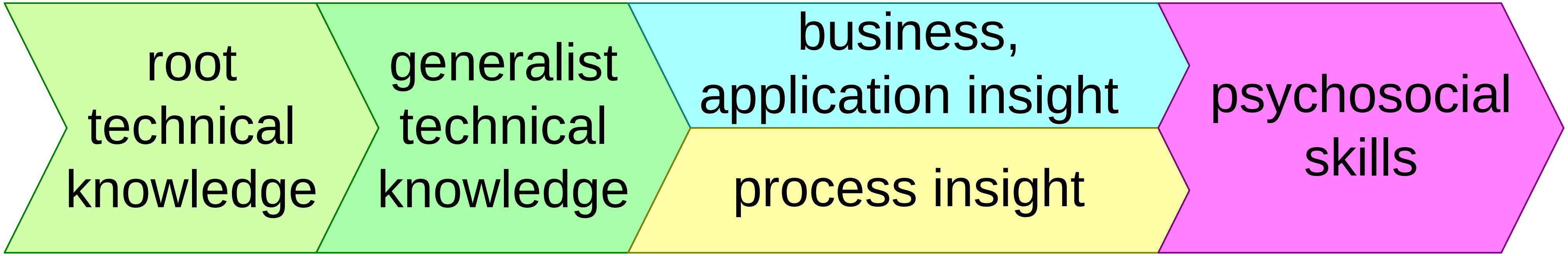
The typical phases of a system architect development are described, beginning at the fundamental technology knowledge, with a later broadening in technology and in business aspects. Finally the subtlety of individual human beings is taken into account.

Distribution

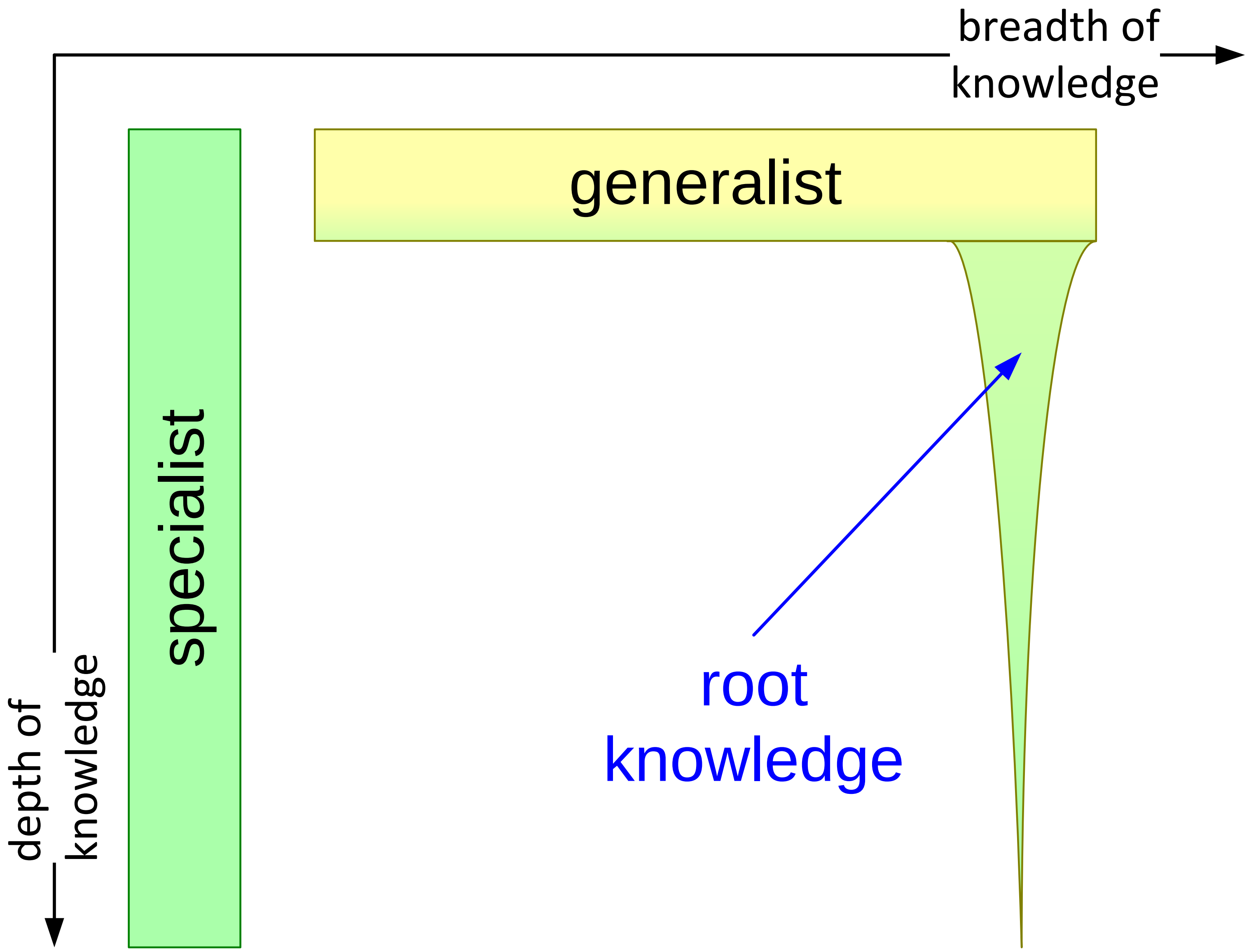
November 18, 2023
status: concept
version: 1.1



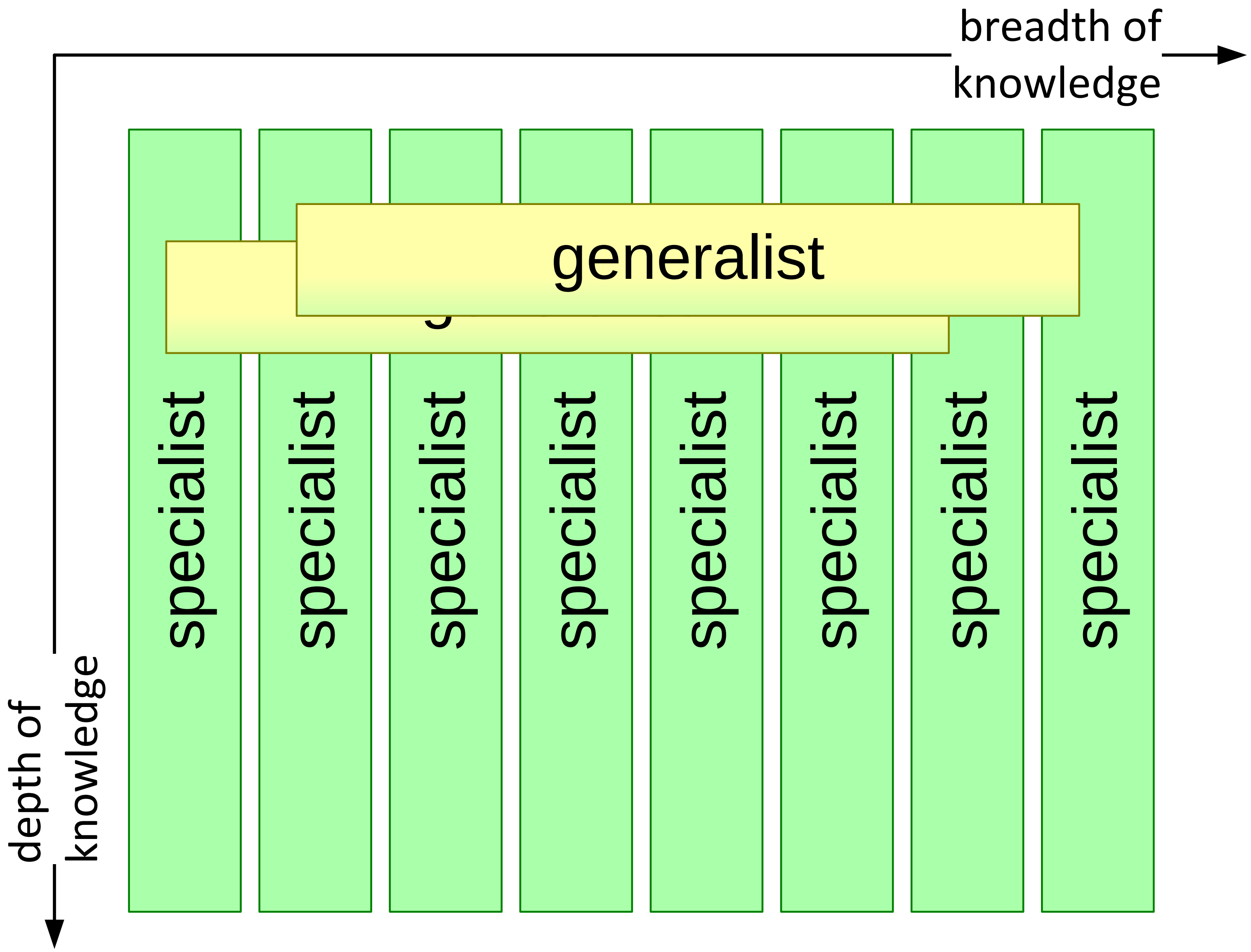
Typical Growth of a System Architect



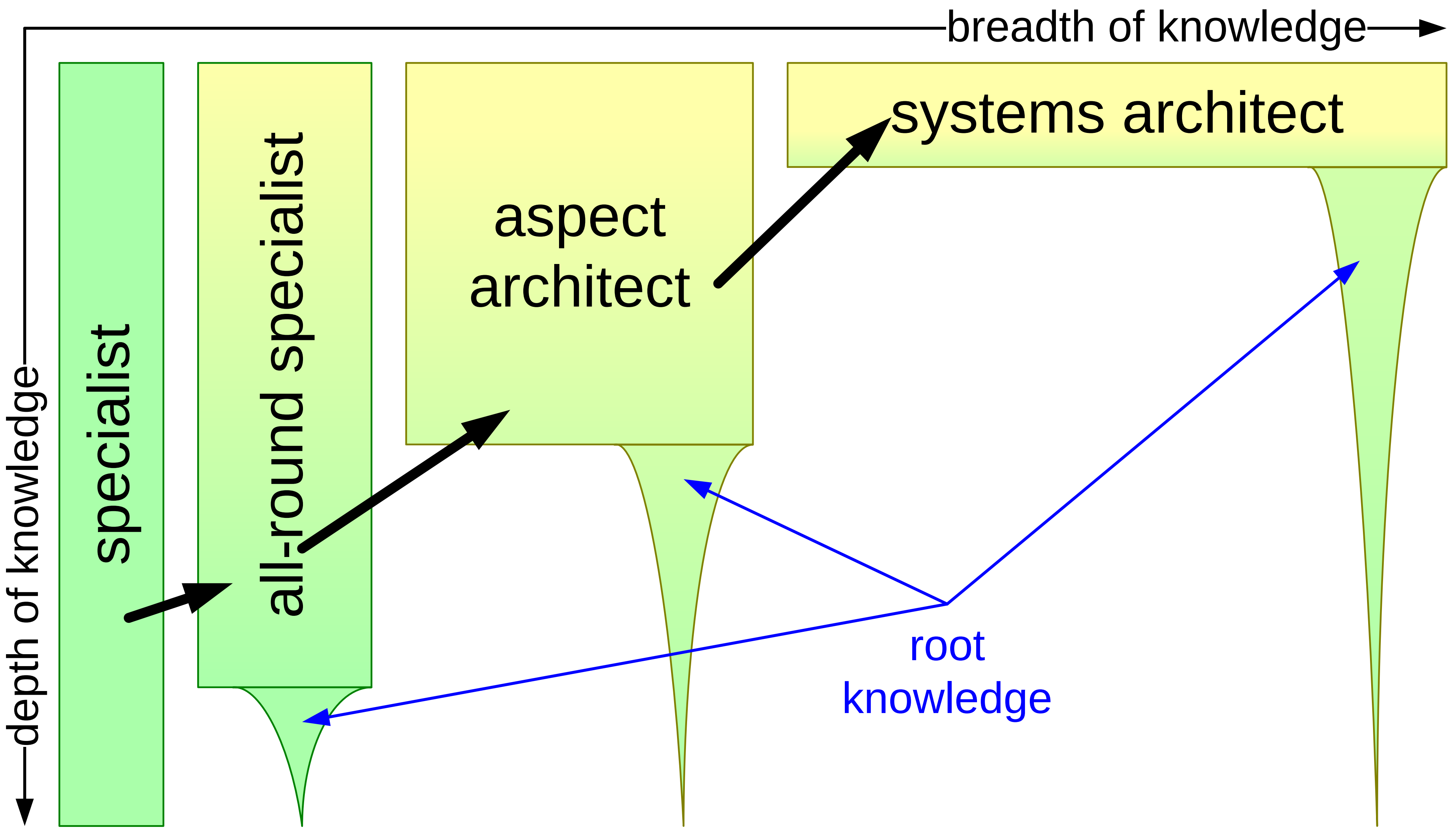
Generalist versus Specialist



Generalists and Specialists are Complementary



Spectrum from Specialist to System Architect

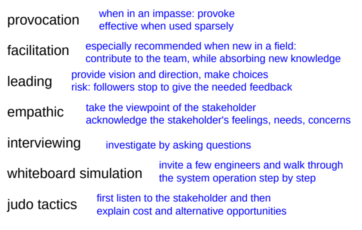


Architecting Interaction Styles

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.



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

November 18, 2023
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:

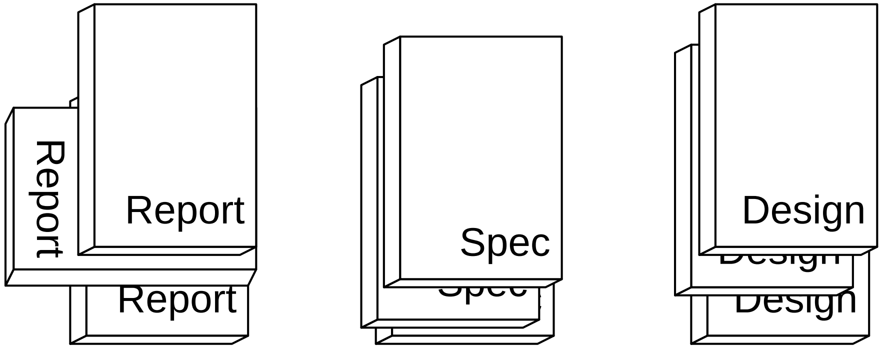
- 1 operational leader (project leader)
- 1 system architect
- 1 marketing manager
- 1 observer (optional)

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

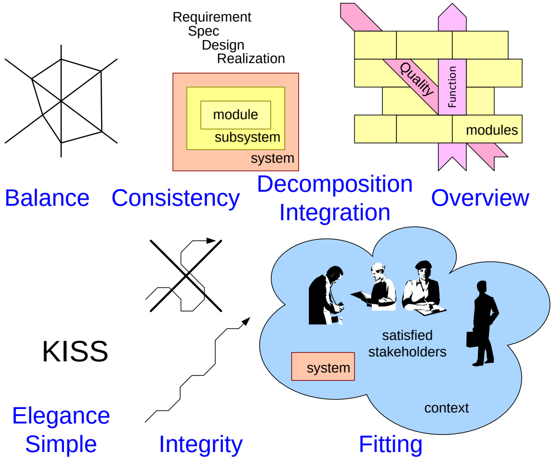
Present (max. 2 flips) the result and the process (the relation and interaction of the three roles).

Role and Task of a System Architect

Deliverables



Responsibilities



Balance

Consistency

Decomposition
Integration

Overview

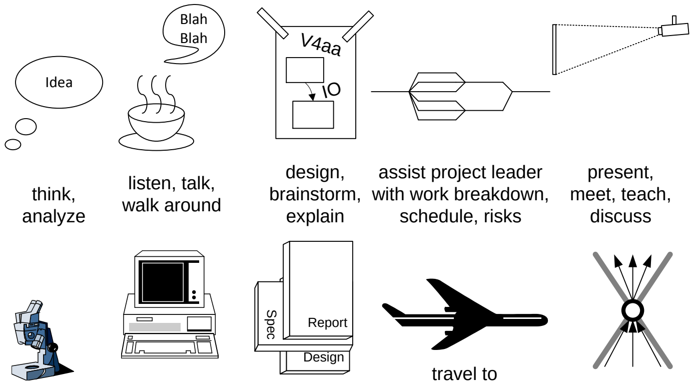
KISS

Elegance
Simple

Integrity

Fitting

Daily Activities



think,
analyze

listen, talk,
walk around

design,
brainstorm,
explain

assist project leader
with work breakdown,
schedule, risks

present,
meet, teach,
discuss

test,
integrate

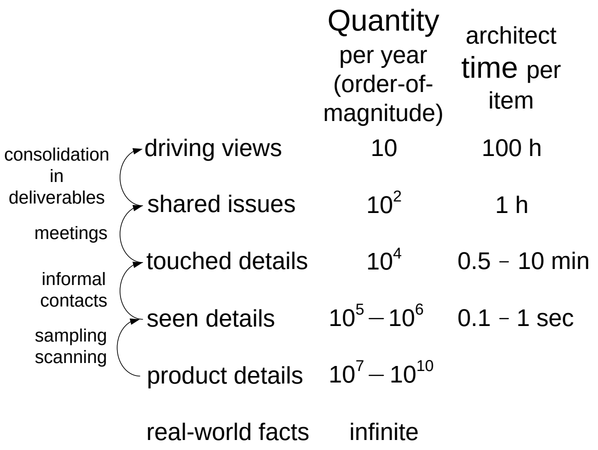
write,
consolidate,
browse

read,
review

travel to
customer,
supplier,
conference

provide
vision and
leadership

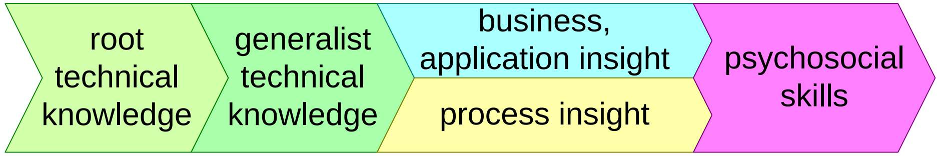
From detail to overview



	Quantity per year (order-of-magnitude)	architect time per item
driving views	10	100 h
shared issues	10^2	1 h
touched details	10^4	0.5 - 10 min
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



root technical knowledge

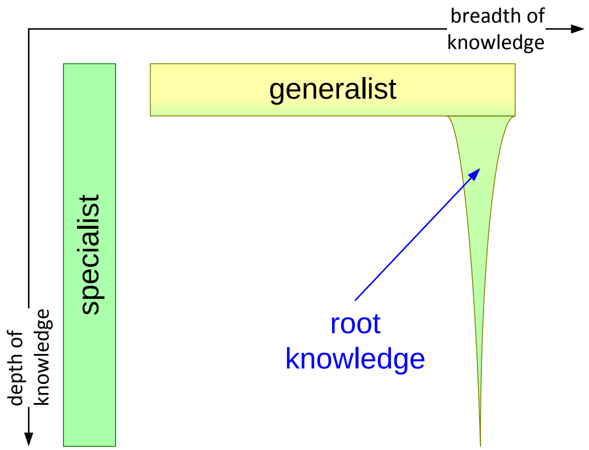
generalist technical knowledge

business, application insight

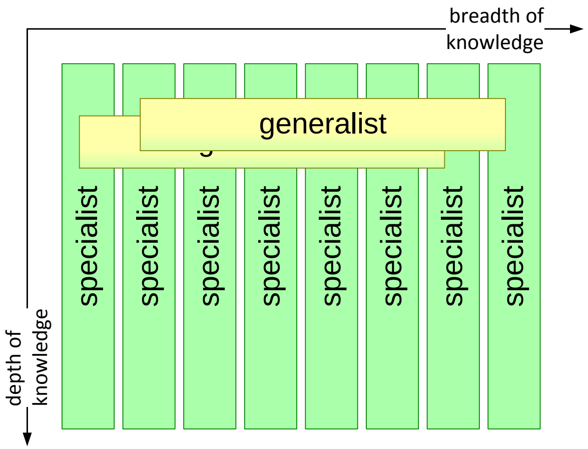
process insight

psychosocial skills

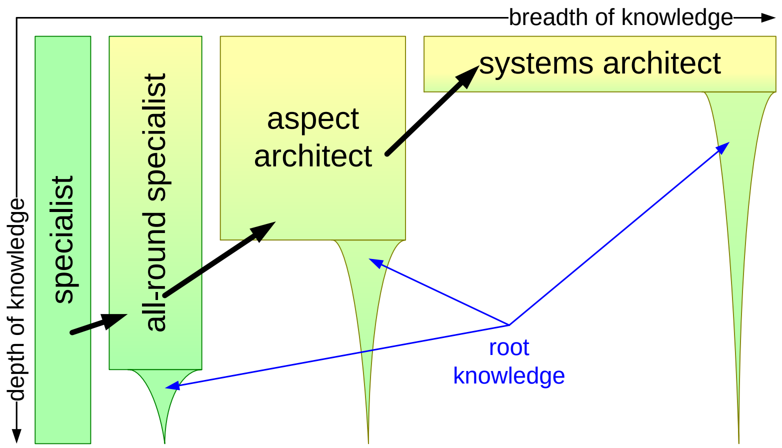
Generalist vs Specialist



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?

Distribution

November 18, 2023
status: concept
version: 1.4

Fundamentals of Requirements Engineering

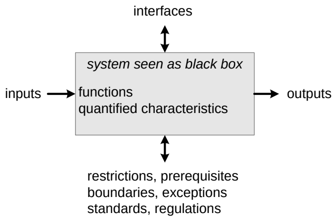
by *Gerrit Muller* USN-SE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

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.

Distribution

November 18, 2023
status: concept
version: 0.1



Definition of “Requirement”

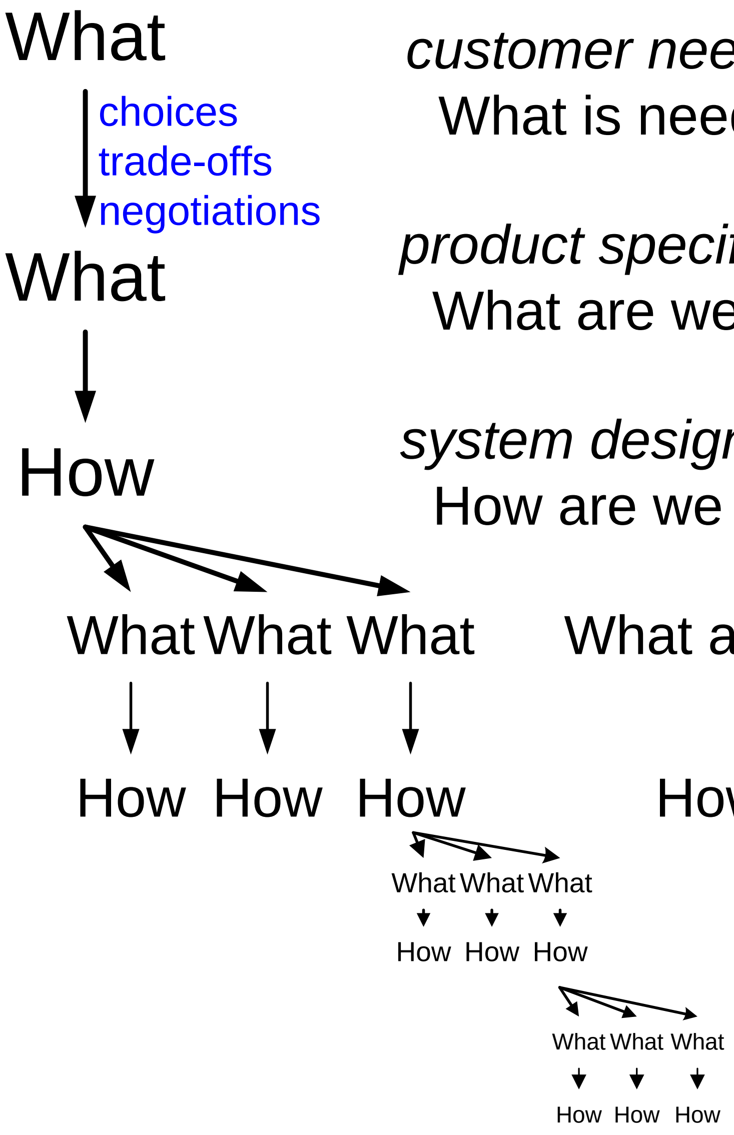
Requirements describing the needs of the customer: ***Customer Needs***

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?

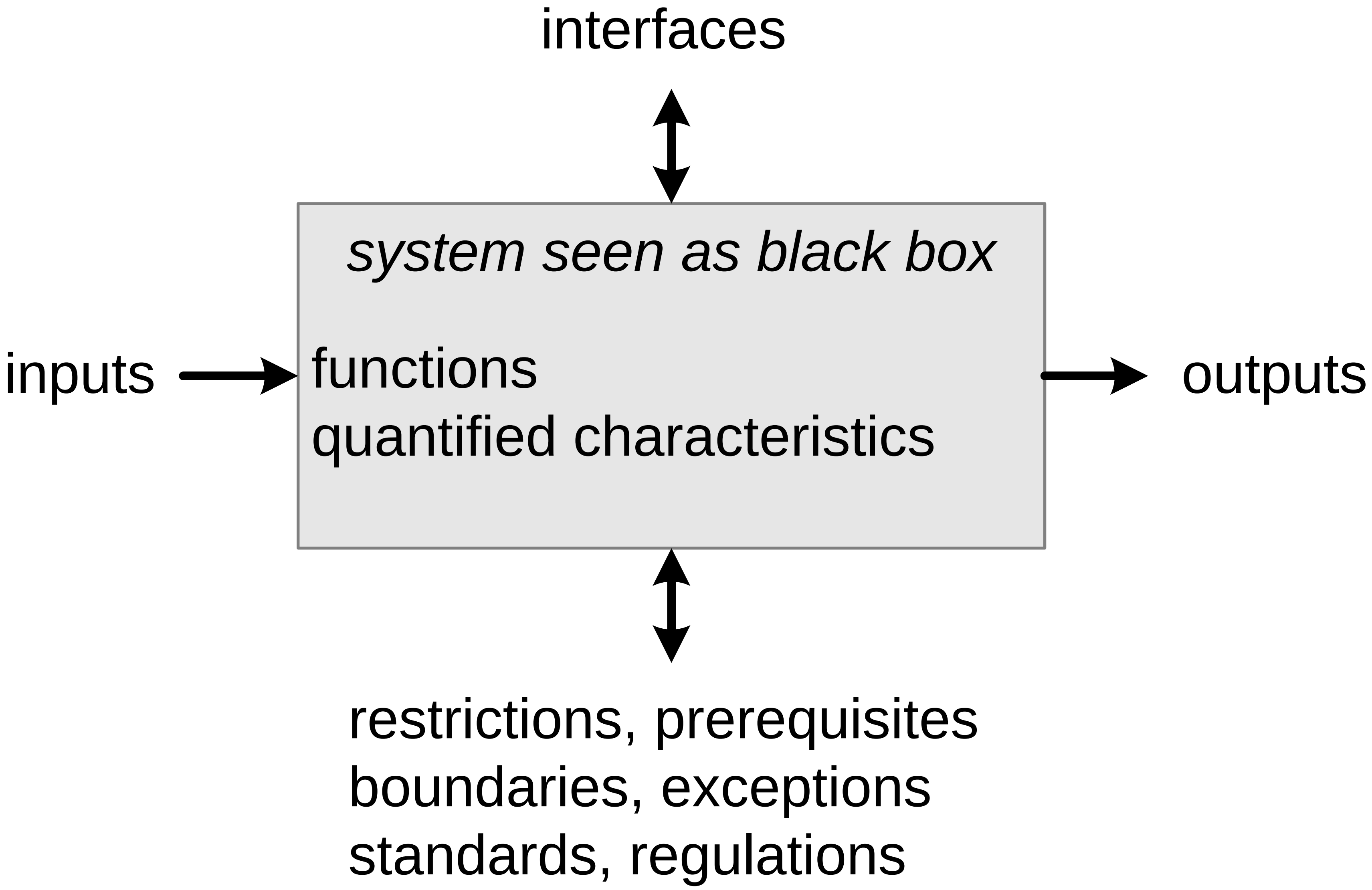
system design:
How are we going to realize the product?

What are the subsystems we will realize?

How will the subsystems be realized?

up to "atomic" components

System as a Black Box



interfaces

system seen as black box

inputs

functions

quantified characteristics

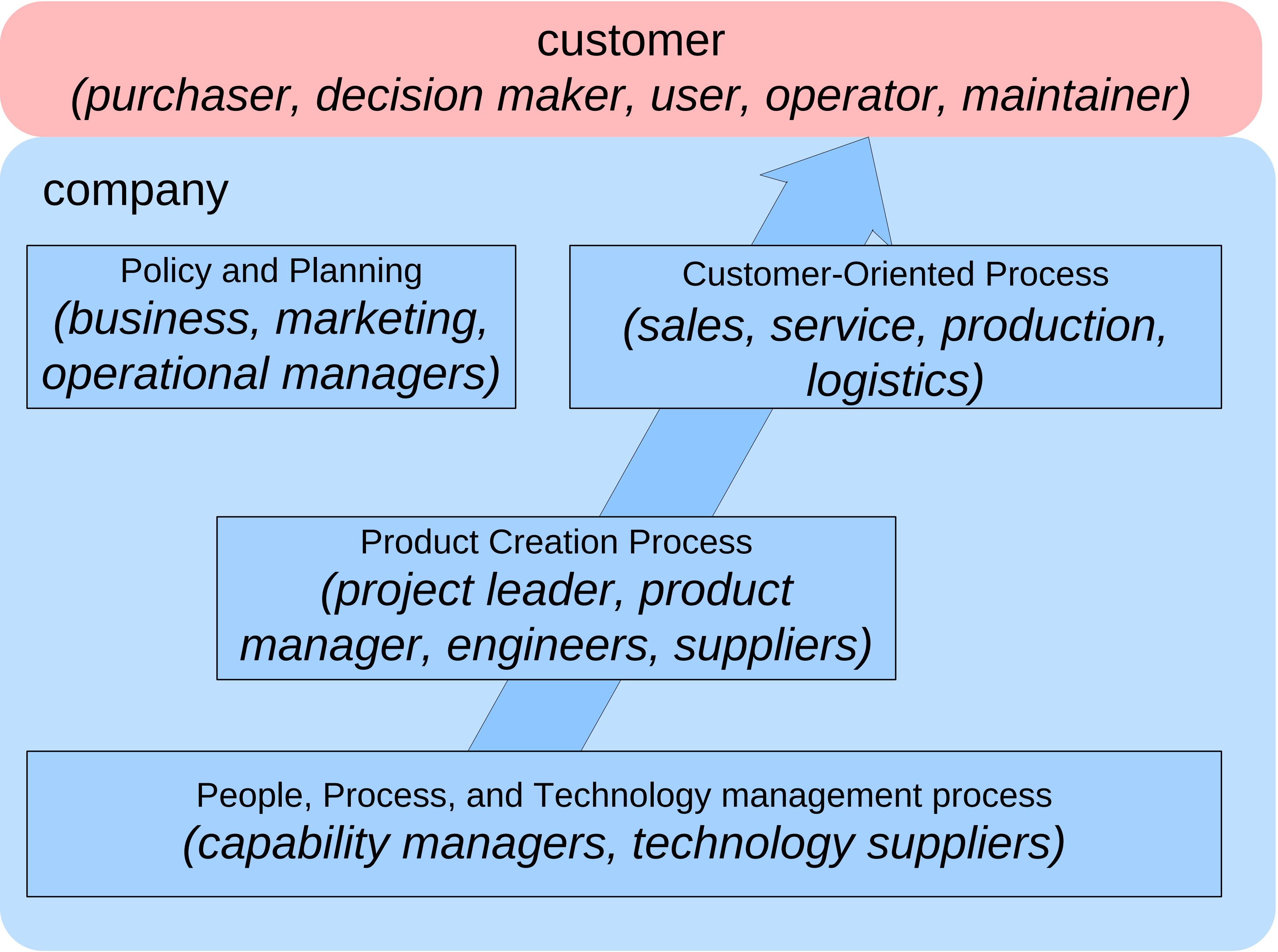
outputs

restrictions, prerequisites

boundaries, exceptions

standards, regulations

Stakeholders w.r.t. Requirements


customer
(purchaser, decision maker, user, operator, maintainer)
company
Policy and Planning
(business, marketing, operational managers)
Customer-Oriented Process
(sales, service, production, logistics)
Product Creation Process
(project leader, product manager, engineers, suppliers)
People, Process, and Technology management process
(capability managers, technology suppliers)

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

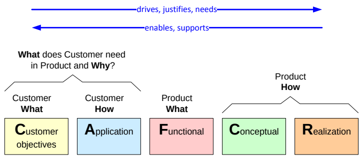
www.gaudisite.nl

Abstract

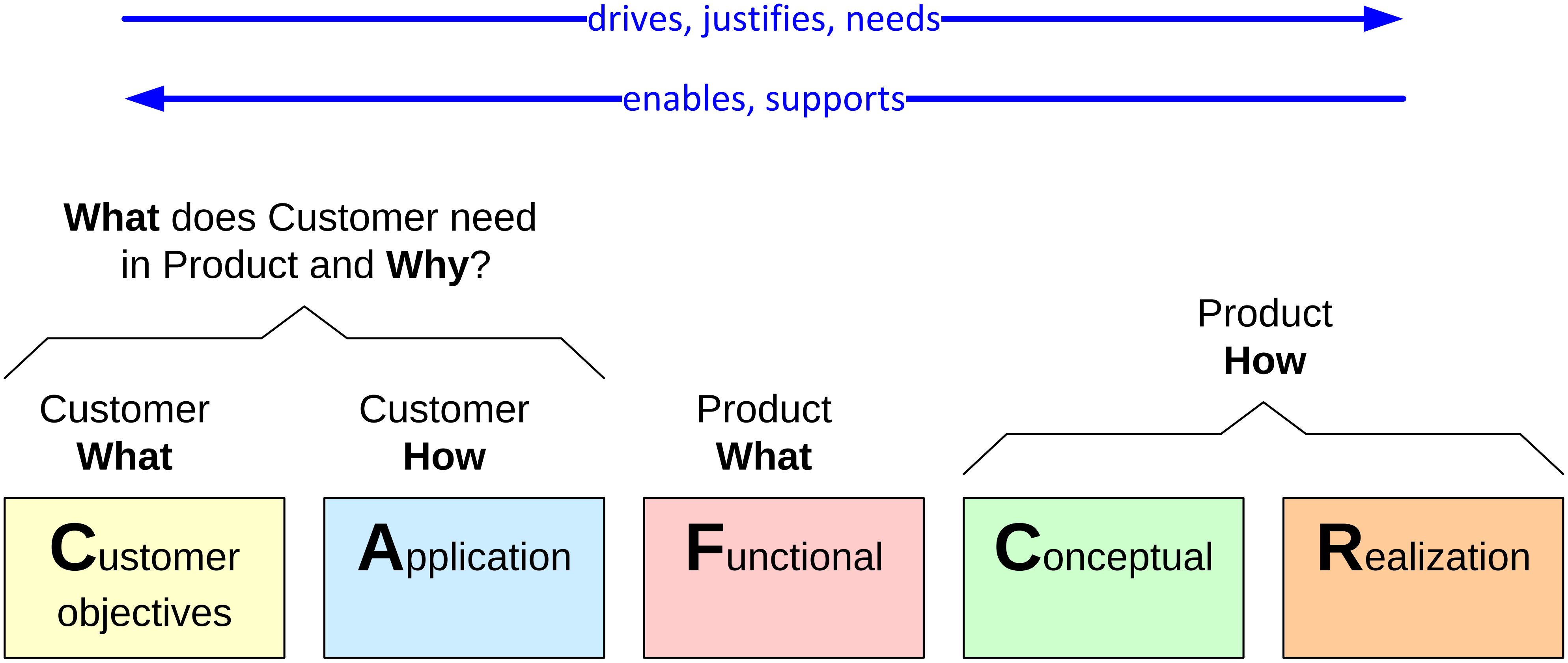
The basic “CAFCR” reference model is described, which is used to describe a system in relation to its context. The main stakeholder in the context is the customer. The question “Who is the customer?” is addressed.

Distribution

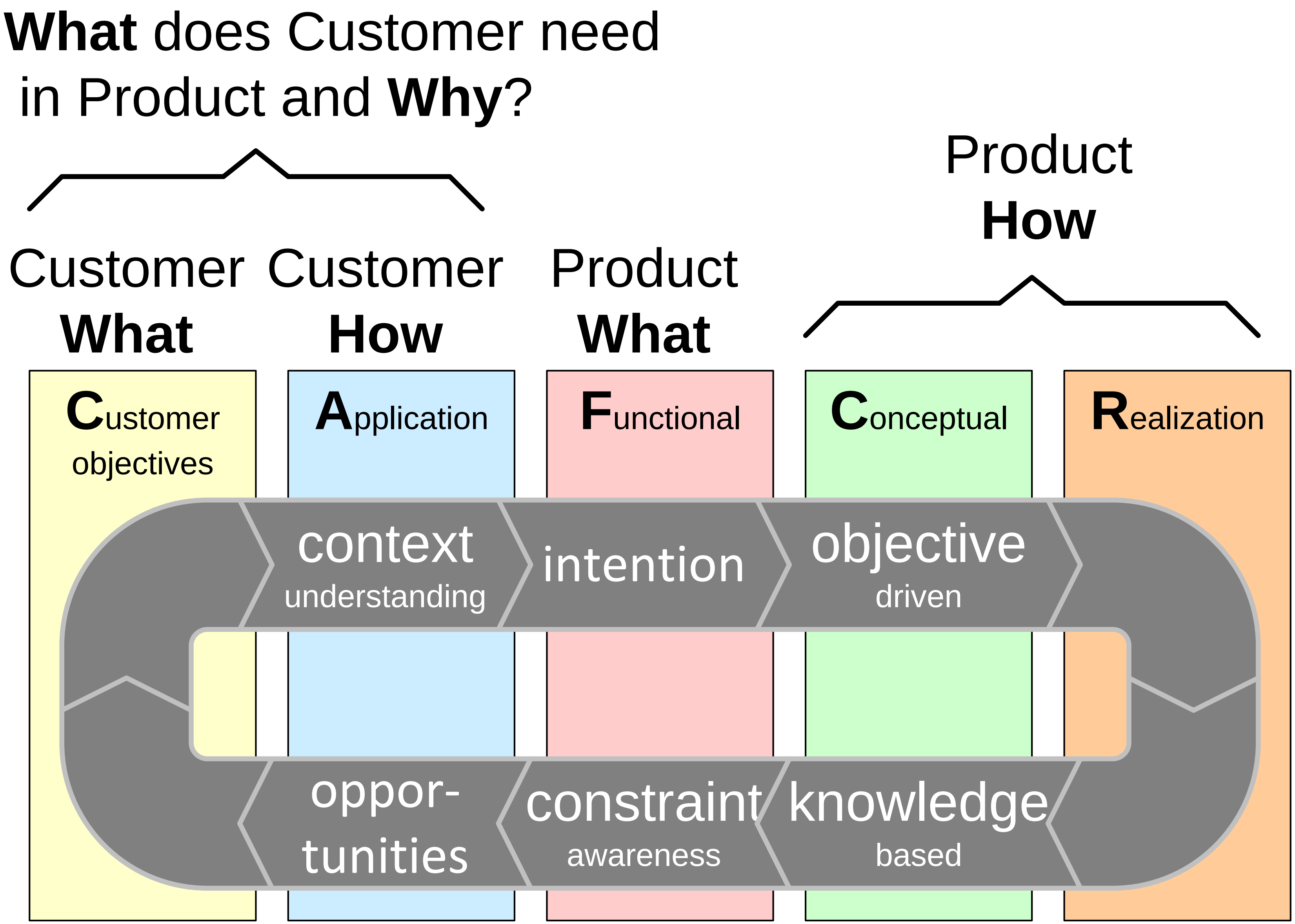
November 18, 2023
status: draft
version: 0.4



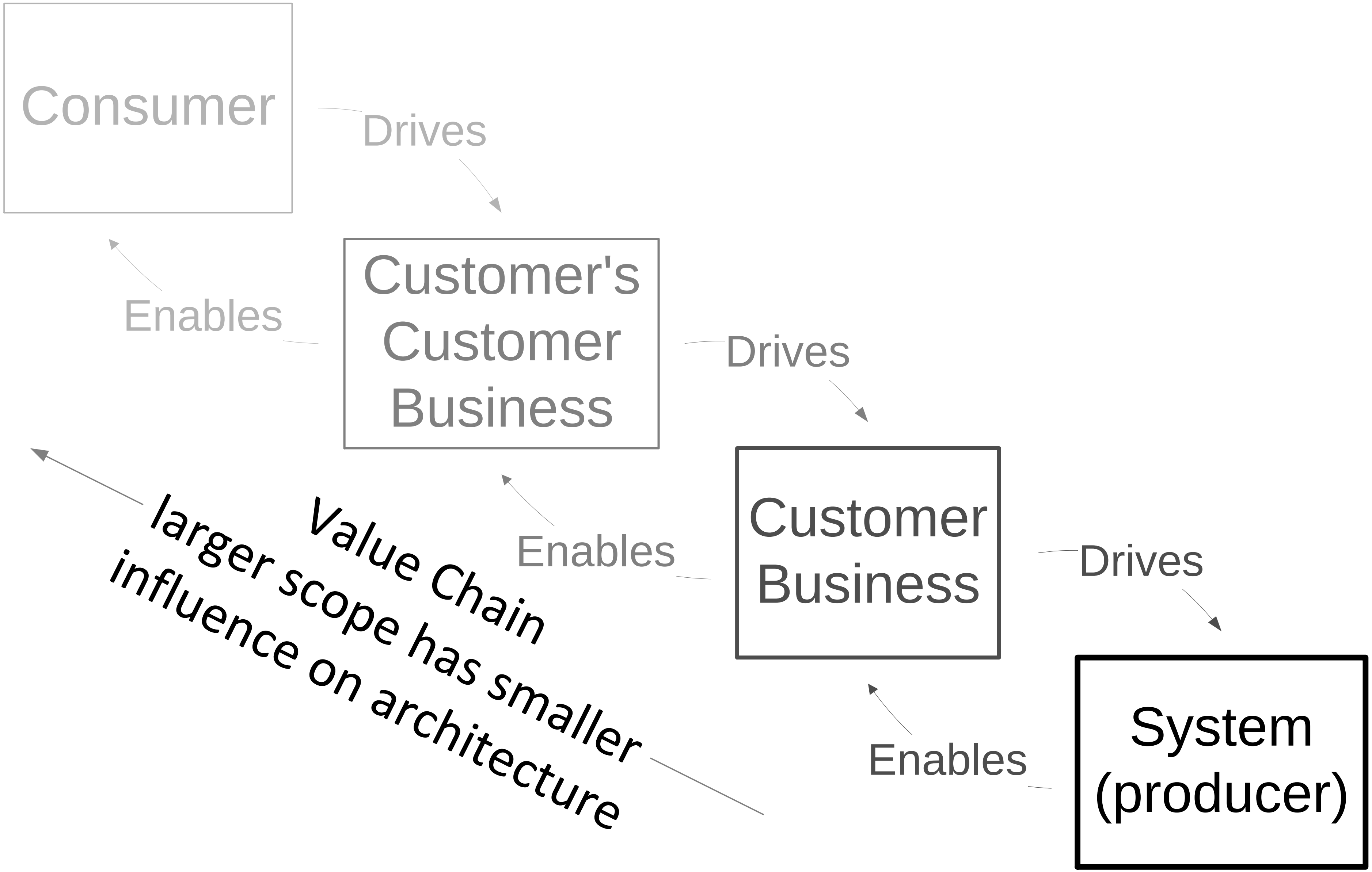
The “CAFCR” model



Integrating CAFCR



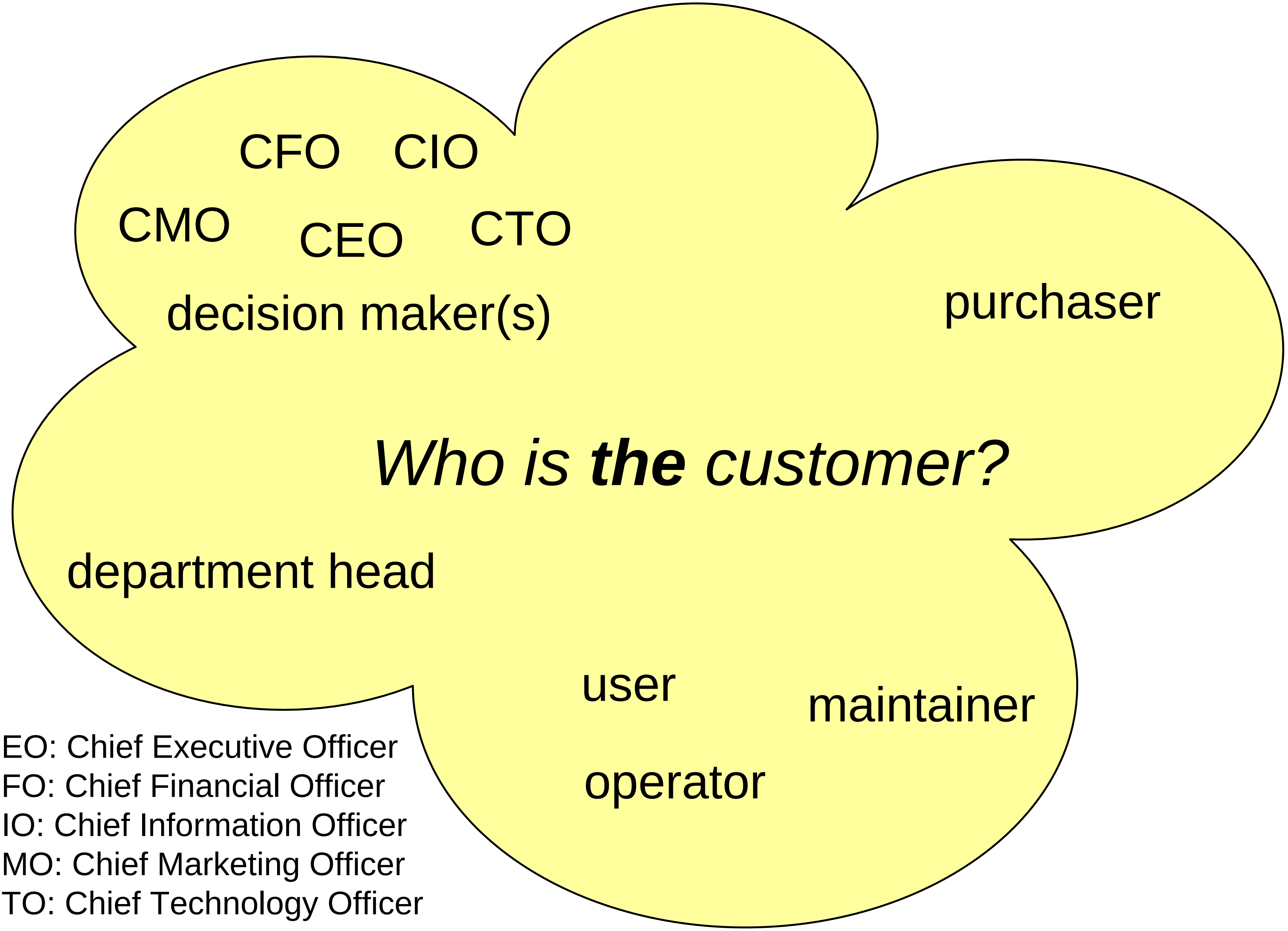
CAFCR can be applied recursively



Market segmentation

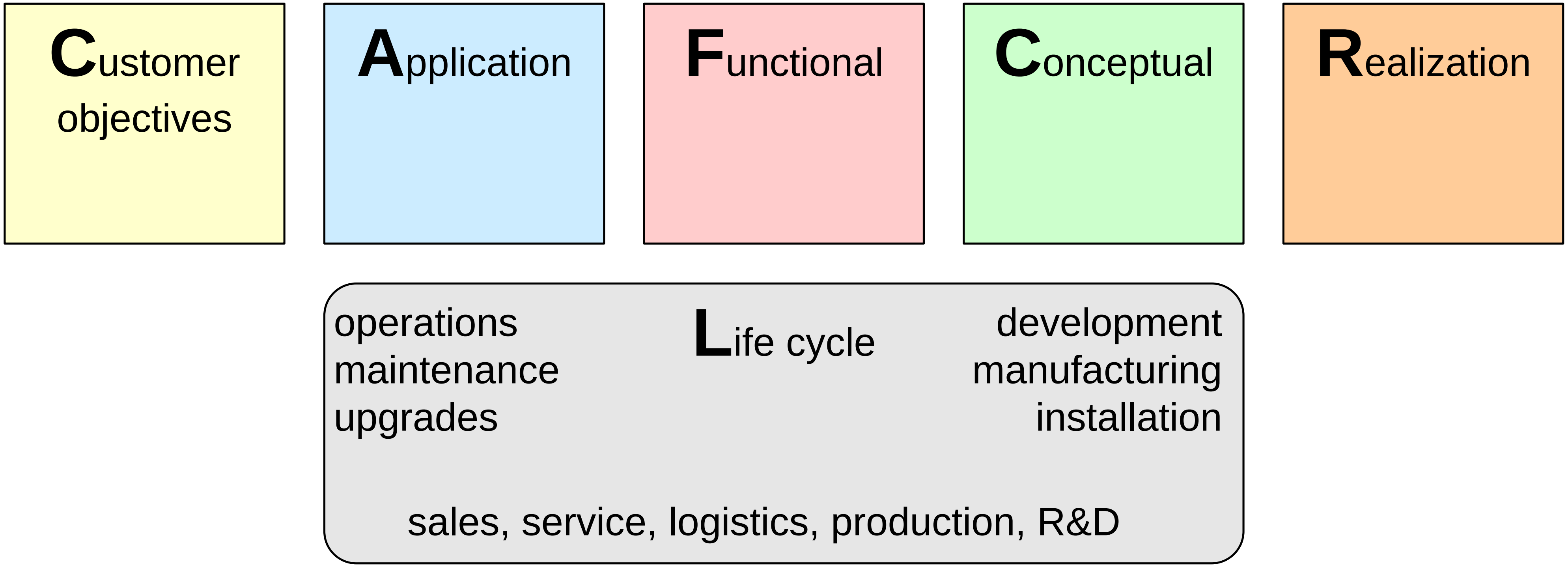
segmentation axis	examples
geographical	USA, UK, Germany, Japan, China
business model	profit, non profit
economics	high end versus cost constrained
consumers	youth, elderly
outlet	retailer, provider, OEM, consumer direct

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

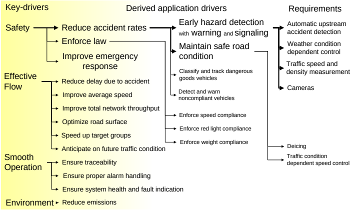


Key Drivers How To

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.

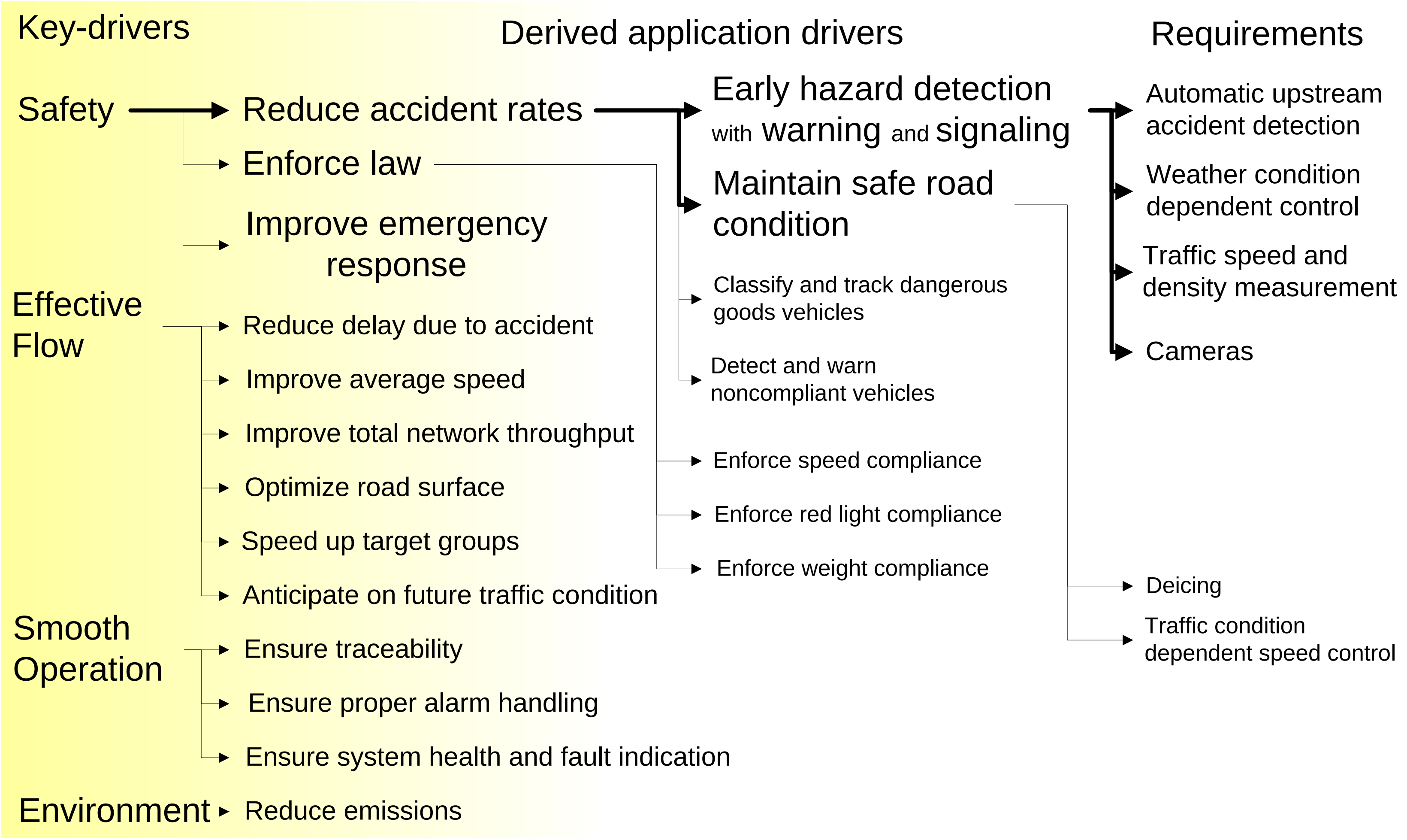

Key-drivers
Derived application drivers
Requirements
Safety
Reduce accident rates
Enforce law
Improve emergency response
Effective Flow
Reduce delay due to accident
Improve average speed
Improve total network throughput
Optimize road surface
Speed up target groups
Anticipate on future traffic condition
Smooth Operation
Ensure traceability
Ensure proper alarm handling
Ensure system health and fault indication
Environment
Reduce emissions
Early hazard detection with warning and signaling
Maintain safe road condition
Classify and track dangerous goods vehicles
Detect and warn noncompliant vehicles
Enforce speed compliance
Enforce red light compliance
Enforce weight compliance
Automatic upstream accident detection
Weather condition dependent control
Traffic speed and density measurement
Cameras
Deicing
Traffic condition dependent speed control

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

Distribution

November 18, 2023
status: draft
version: 0.2

Example Motorway Management Analysis



Key-drivers

Safety

Effective Flow

Smooth Operation

Environment

Derived application drivers

Reduce accident rates

Enforce law

Improve emergency response

Reduce delay due to accident

Improve average speed

Improve total network throughput

Optimize road surface

Speed up target groups

Anticipate on future traffic condition

Ensure traceability

Ensure proper alarm handling

Ensure system health and fault indication

Reduce emissions

Early hazard detection with warning and signaling

Maintain safe road condition

Classify and track dangerous goods vehicles

Detect and warn noncompliant vehicles

Enforce speed compliance

Enforce red light compliance

Enforce weight compliance

Requirements

Automatic upstream accident detection

Weather condition dependent control

Traffic speed and density measurement

Cameras

Deicing

Traffic condition dependent speed control

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

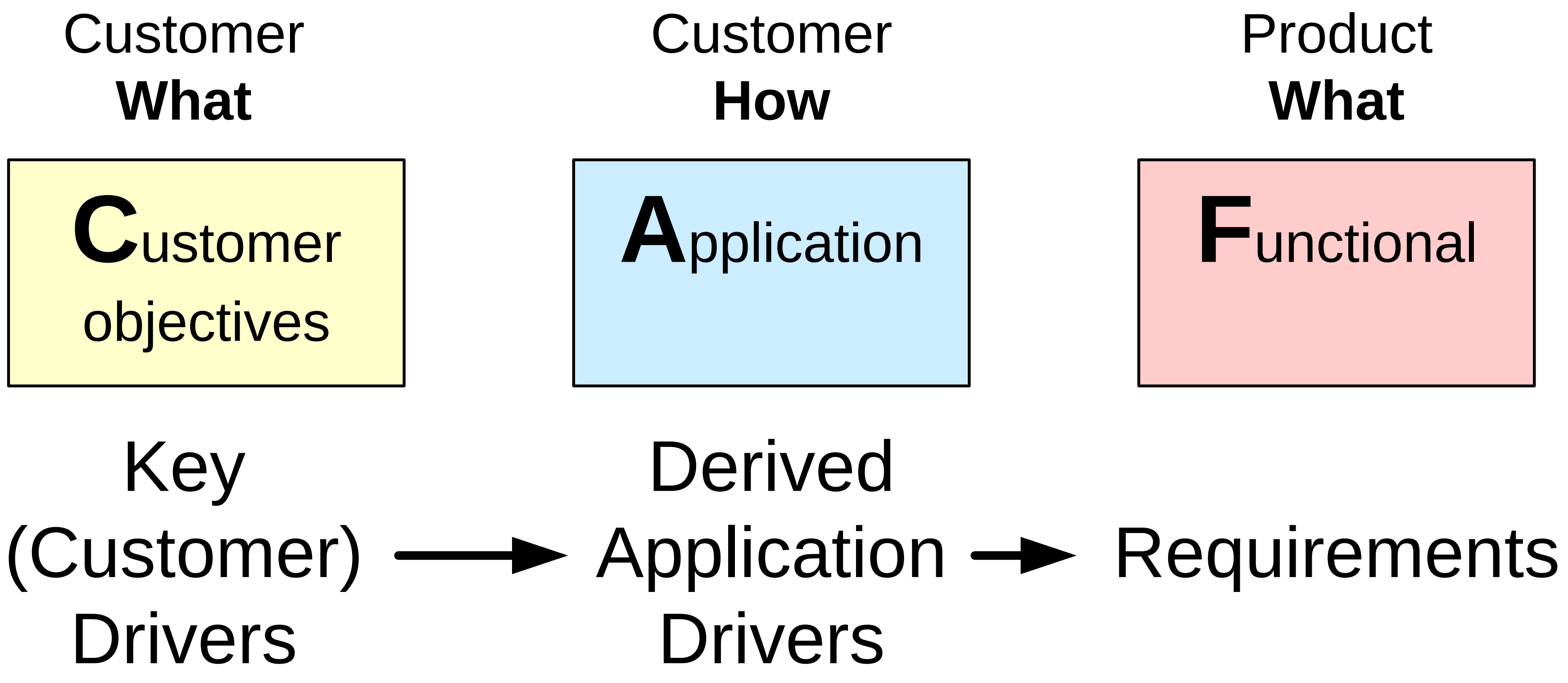
Method to create Key Driver Graph

• 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



goal

means
may be skipped or
articulated by several
intermediate steps

functions
interfaces
performance figures

Requirements Elicitation and Selection

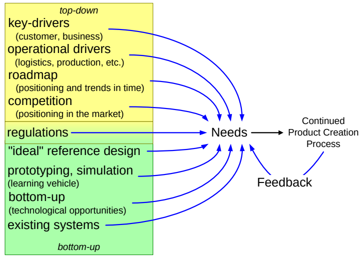
by *Gerrit Muller* University of South-Eastern Norway-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

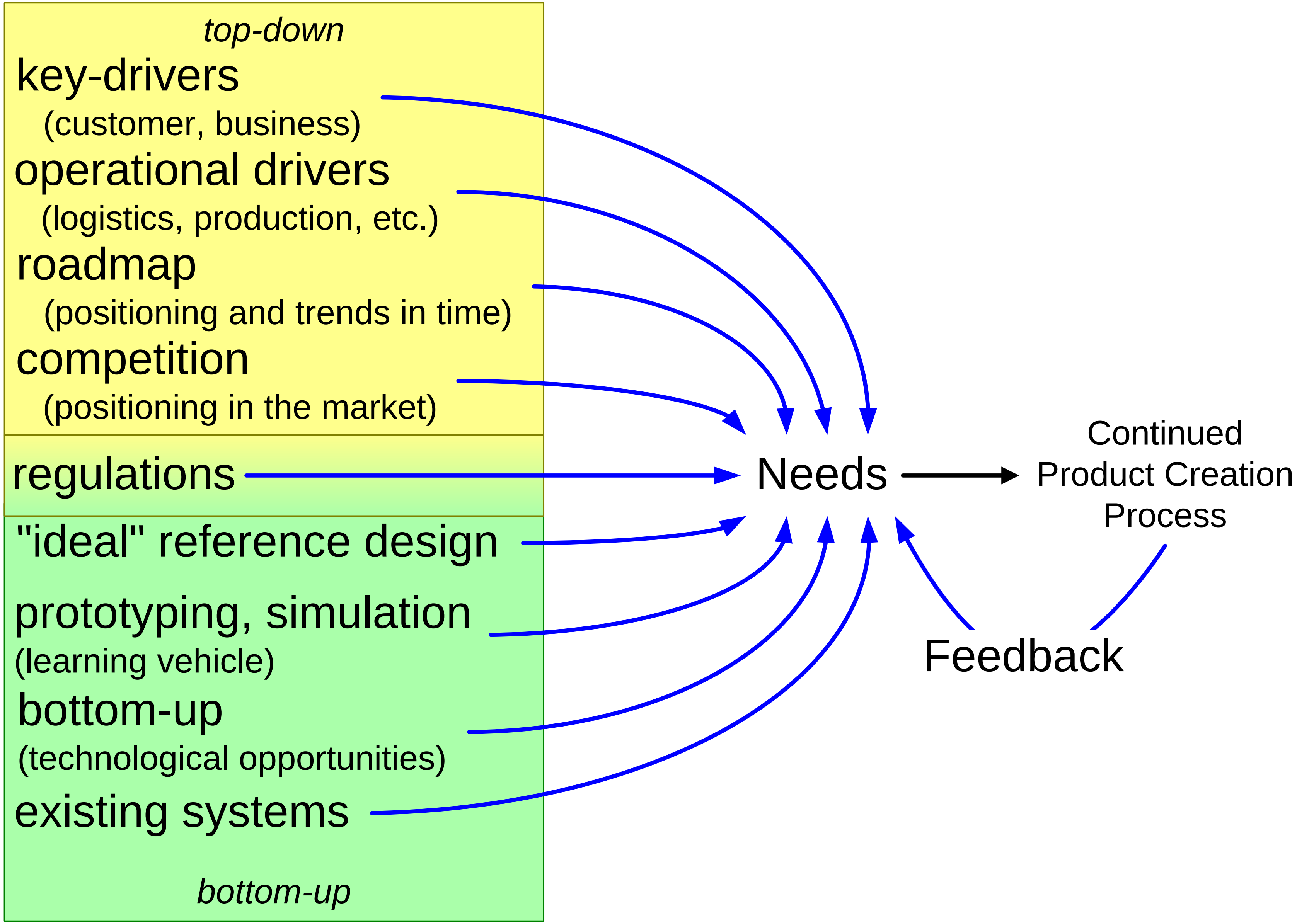
An elicitation method for needs is described using many different viewpoints. A selection process with a coarse and a fine selection is described to reduce the specification to an acceptable and feasible subset.

Distribution

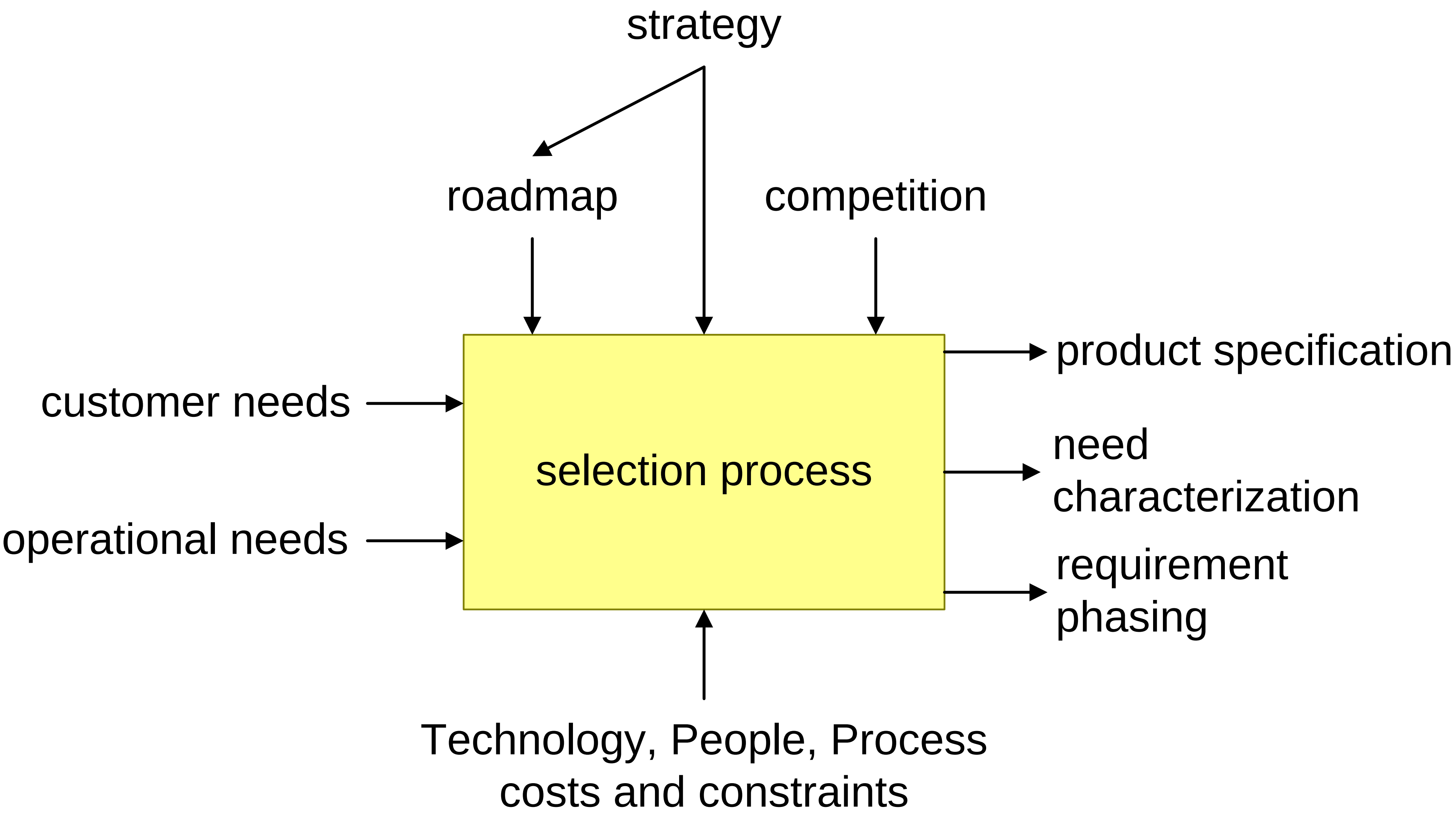
November 18, 2023
status: draft
version: 0



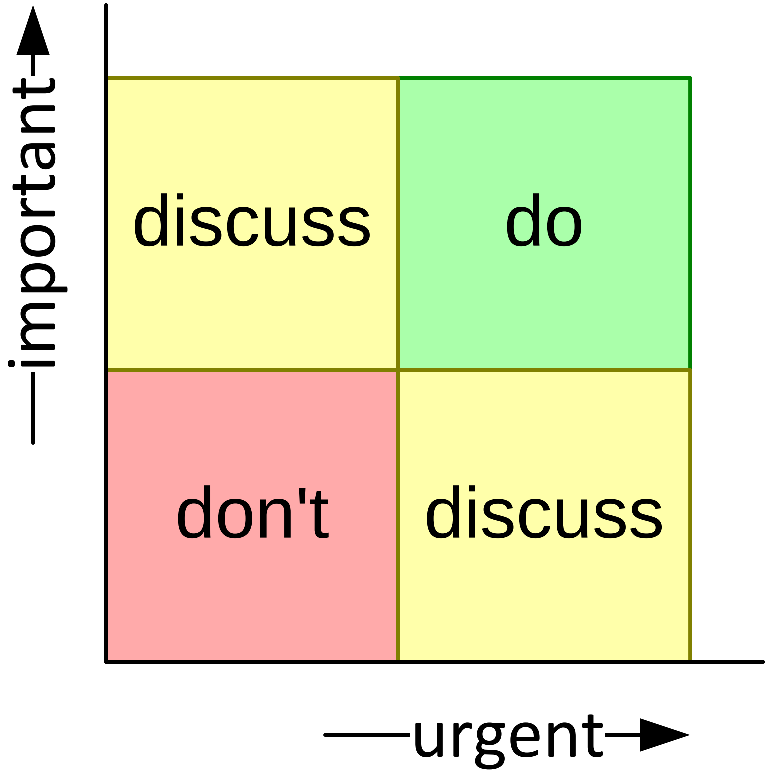
Complementary Viewpoints to Capture Requirements



Requirement Selection Process



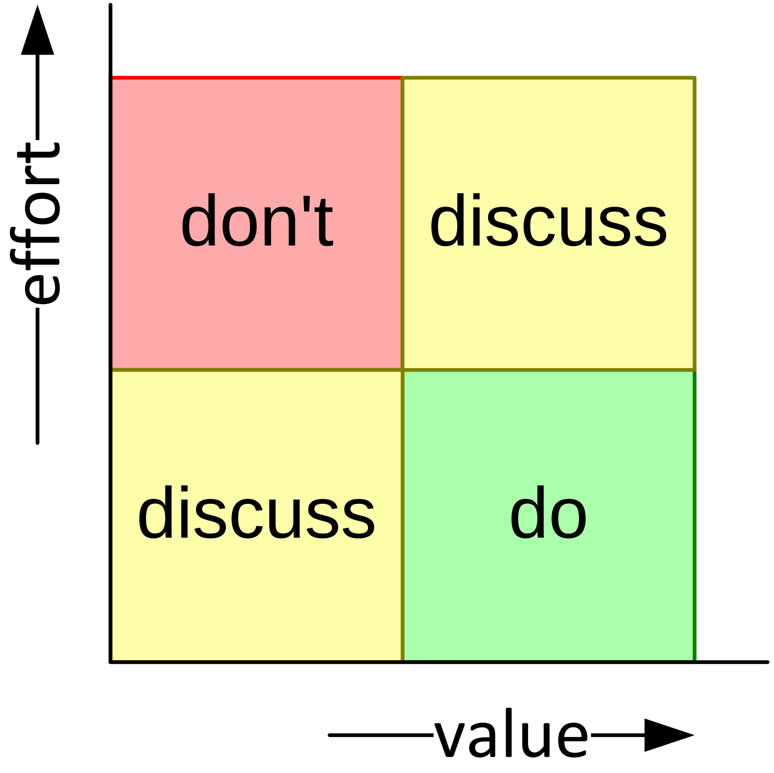
Simple Qualification Method



important

discuss	do
don't	discuss

urgent



effort

don't	discuss
discuss	do

value

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

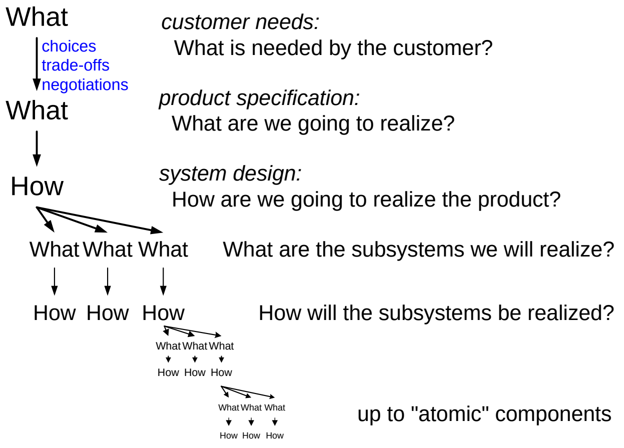
Requirements describing the needs of the customer: ***Customer Needs***

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



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

Requirements for Requirements

Specific

Unambiguous

Verifiable

Quantifiable

Measurable

Complete

Traceable

Enable Human Use

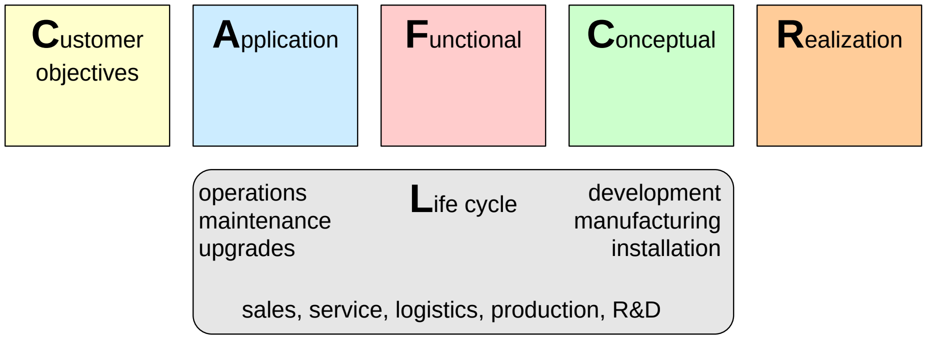
Accessible

Understandable

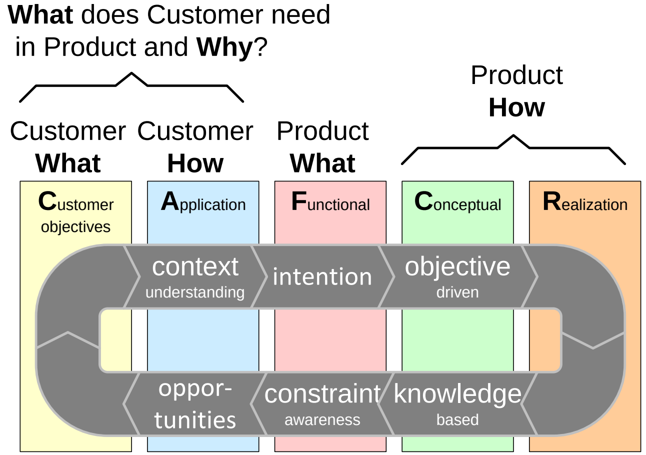
Low threshold

CAFCR, Customer Key Driver Graph

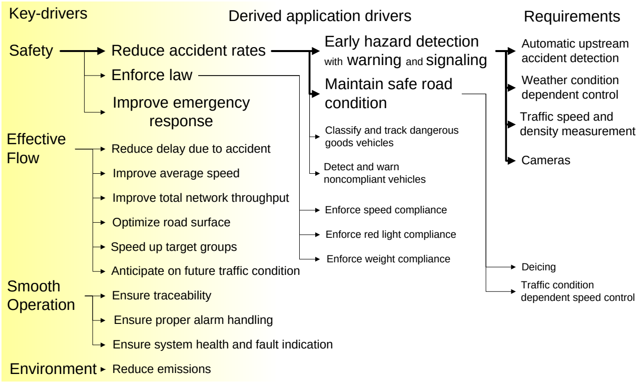
CAFCR+ Model



Iterate over Views

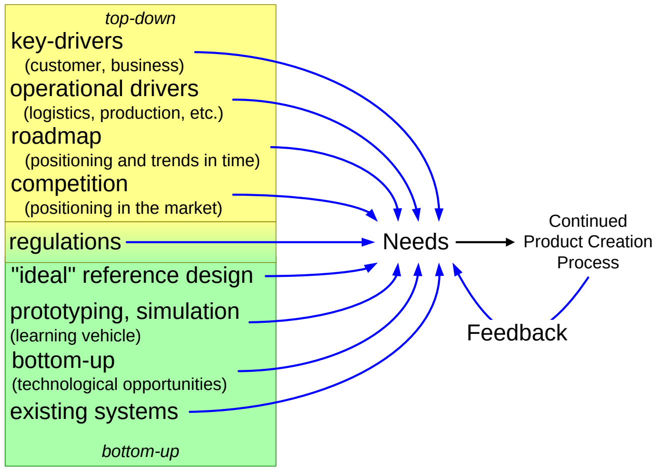


Example Key Driver Graph


Key-drivers
Derived application drivers
Requirements
Safety
Reduce accident rates
Enforce law
Improve emergency response
Effective Flow
Reduce delay due to accident
Improve average speed
Improve total network throughput
Optimize road surface
Speed up target groups
Anticipate on future traffic condition
Smooth Operation
Ensure traceability
Ensure proper alarm handling
Ensure system health and fault indication
Environment
Reduce emissions
Early hazard detection with warning and signaling
Maintain safe road condition
Classify and track dangerous goods vehicles
Detect and warn noncompliant vehicles
Enforce speed compliance
Enforce red light compliance
Enforce weight compliance
Automatic upstream accident detection
Weather condition dependent control
Traffic speed and density measurement
Cameras
Deicing
Traffic condition dependent speed control

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

Complementary Viewpoints



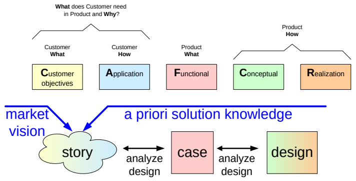
Story How To

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

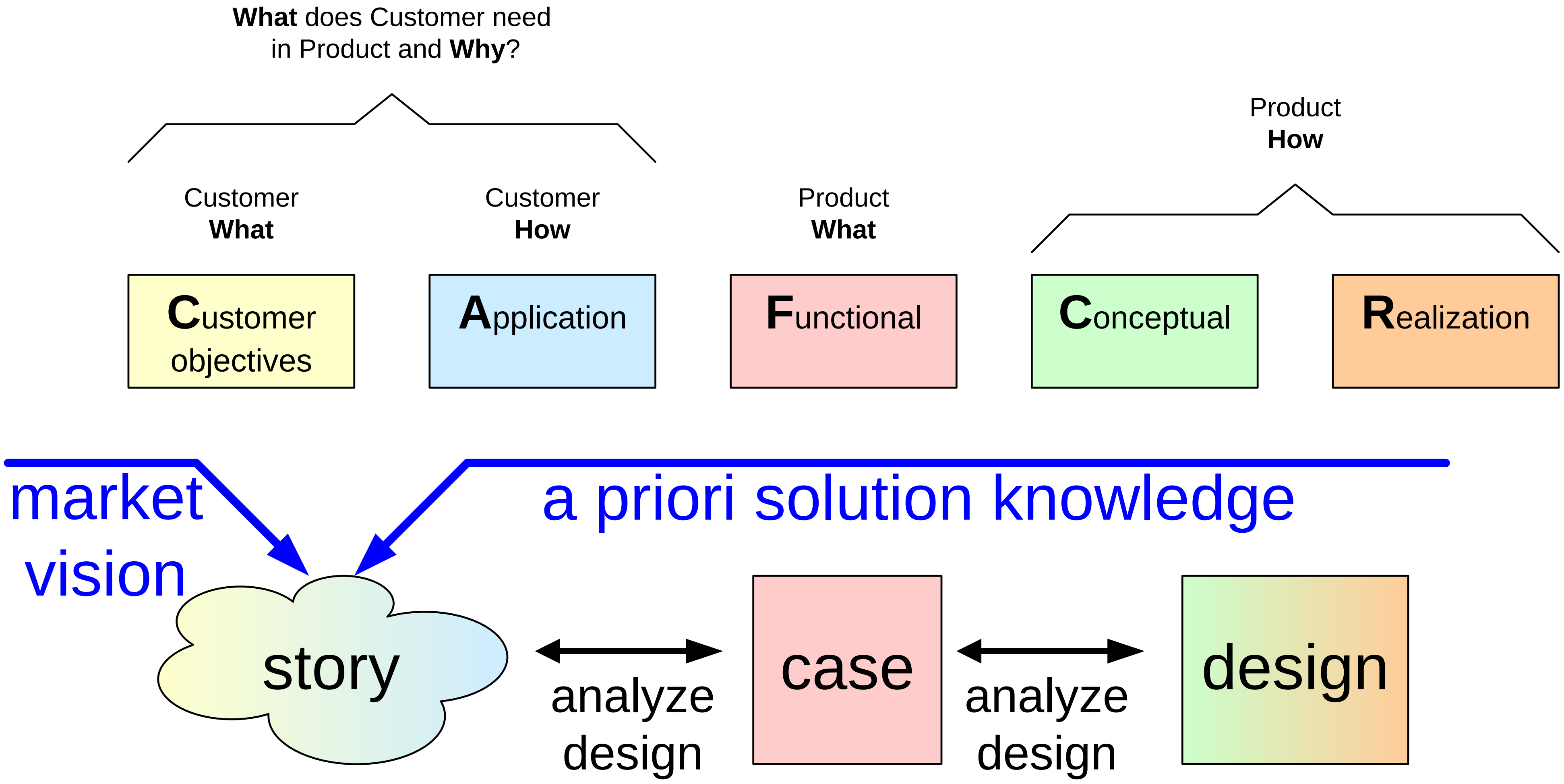
Abstract

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.

November 18, 2023
status: concept
version: 1.2

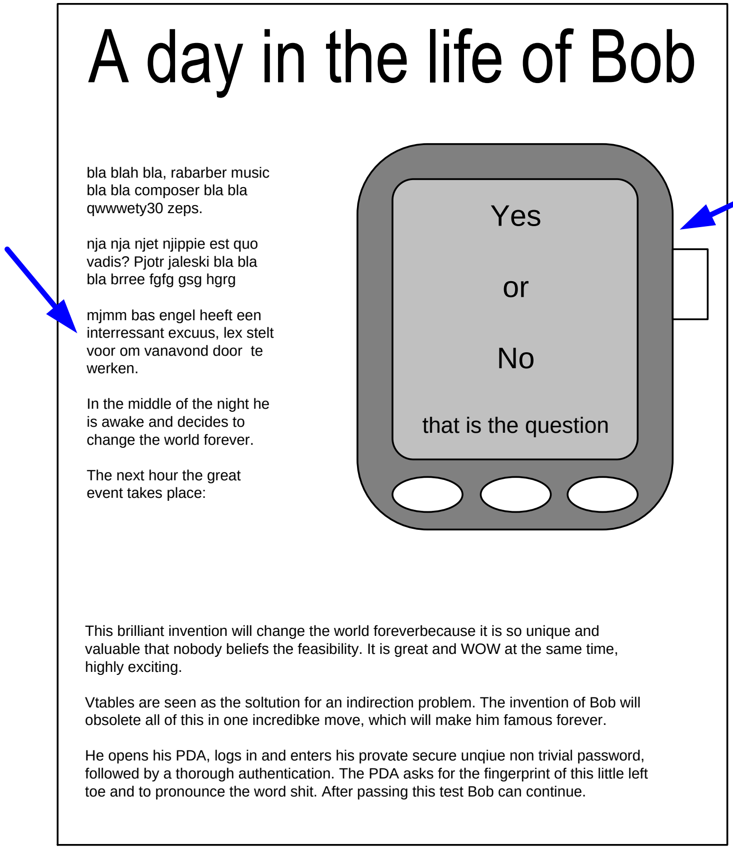


From story to design



Example story layout

ca. half a page of plain English text



A day in the life of Bob

bla blah bla, rabarber music bla bla composer bla bla qwwwety30 zeps.

nja nja njet njippie est quo vadis? Pjotr jaleski bla bla bla brree fgfg gsg hgrg

mjmm bas engel heeft een interressant excuus, lex stelt voor om vanavond door te werken.

In the middle of the night he is awake and decides to change the world forever.

The next hour the great event takes place:

This brilliant invention will change the world foreverbecause it is so unique and valuable that nobody beliefs the feasibility. It is great and WOW at the same time, highly exciting.

Vtables are seen as the soltution for an indirection problem. The invention of Bob will obsolete all of this in one incredibke move, which will make him famous forever.

He opens his PDA, logs in and enters his provate secure unqiue non trivial password, followed by a thorough authentication. The PDA asks for the fingerprint of this little left toe and to pronounce the word shit. After passing this test Bob can continue.

draft or sketch of some essential appliance

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 | Application

- accessible, understandable

"Do you see it in front of you?"

Customer objectives | Application

- valuable, appealing

attractive, important
"Are customers queuing up for this?"

Conceptual | Realization

- 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 | Functional

- 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

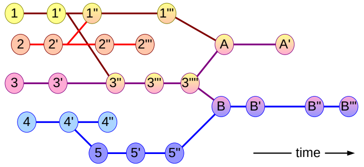
Concept Selection, Set Based Design and Late Decision Making

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

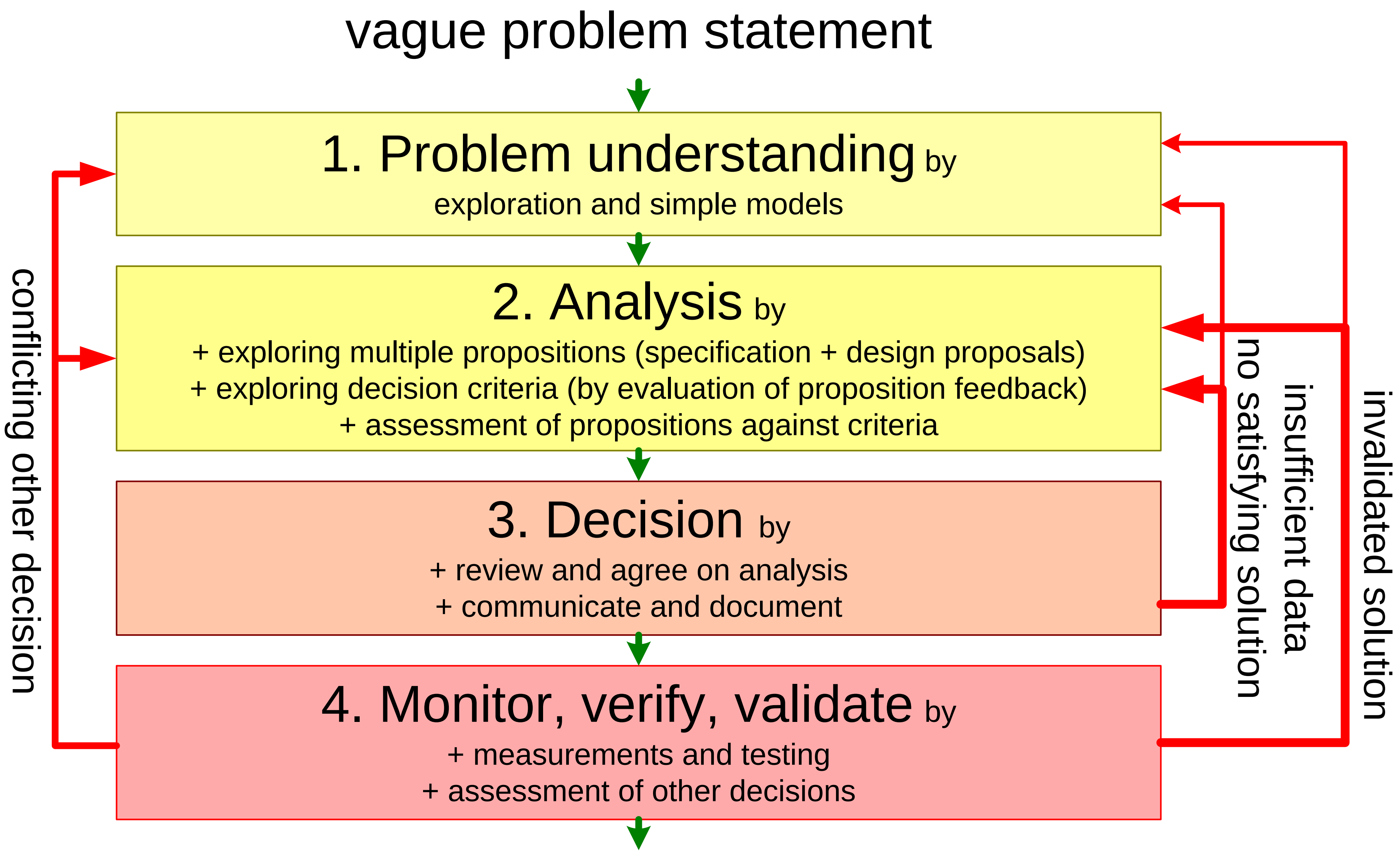
Abstract

We discuss a systems design approach where several design options are maintained concurrently. In LEAN Product Development this is called set-based design. Concentioanl systems engineering also promotes the concurrent evaluation of multiple concepts, the so-called concept selection. Finally, LEAN product development advocates to keep options open as long as feasible; the so-called late decision making.

November 18, 2023
status: planned
version: 0



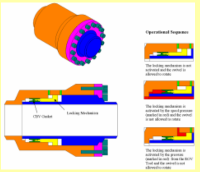
Problem Solving Approach


vague problem statement
1. Problem understanding by
exploration and simple models
2. Analysis by
+ exploring multiple propositions (specification + design proposals)
+ exploring decision criteria (by evaluation of proposition feedback)
+ assessment of propositions against criteria
3. Decision by
+ review and agree on analysis
+ communicate and document
4. Monitor, verify, validate by
+ measurements and testing
+ assessment of other decisions
conflicting other decision
no satisfying solution
insufficient data
invalidated solution

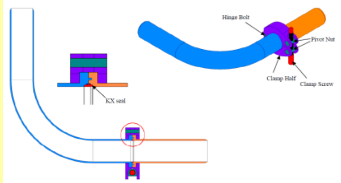
Examples of Pugh Matrix Application

Swivel concept selection

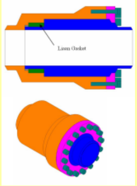
CBV swivel



clamp swivel



dynamic

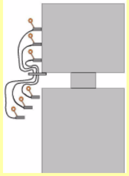


evaluation criteria	weight	CBV		clamp		dynamic	
Maturity	10						
Development level		5	50	2	20	2	50
Cost	20						
Hardware cost		4	80	2	40	5	100
Development cost		5	100	2	40	2	40
Design robustness	25						
Design life							
swivel cycles		5	125	3	75	3	75
pressure cycles		5	125	4	100	5	125
Pressure range							
internal		4	100	4	100	4	100
external		2	50	5	125	2	50
Temperature range		4	100	4	100	4	100
Installation	20						
Initial installatio/retrieval		2	40	3	60	4	80
Connection/disconnection		2	40	4	80	5	100
Operation	25						
Swivel resistance		1	25	4	100	5	125
Spool Length Short		1	25	4	100	5	125
Spool Length Long		3	75	5	125	5	125
Hub loads		2	50	4	100	5	125
Σ points			985		1165		1290

from master paper Halvard Bjørnsen, 2009

EDP-LRP connection

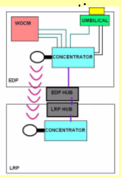
two sided



connectors in hub

connectors in hub

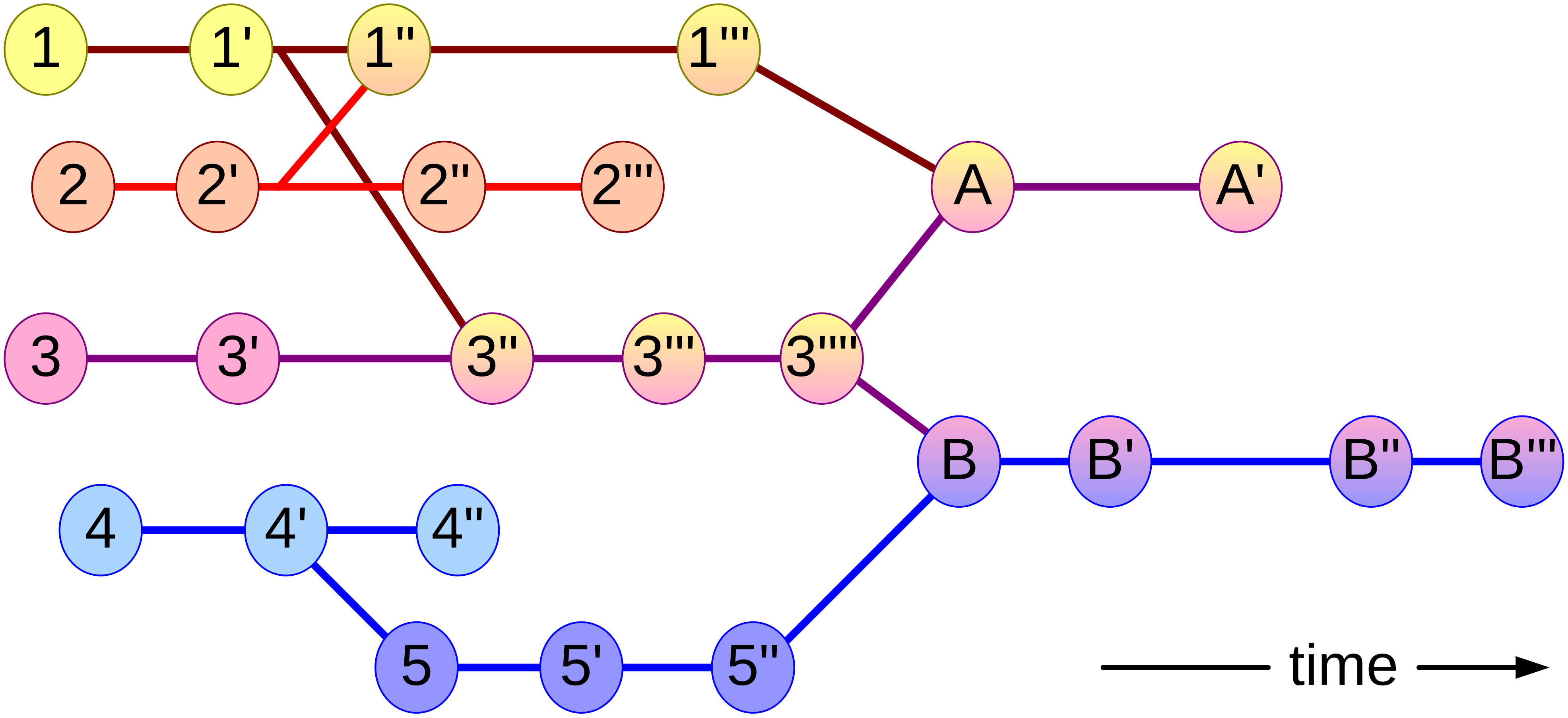
wireless



		Concepts			
Evaluation Criteria	Score	1	2	3	4
Time to connect					
Need for ROV		-	+	+	+
Design		-	+	+	+
Robustness					
Connector design		-	S	S	+
Number of parts		-	-	+	+
Handle roll-off		+	-	S	+
Influence other		+	S	-	S
Redundancy					
Design		+	-	-	S
Interchangeability		+	-	-	-
Cost					
HW cost		-	-	-	-
Manufacturing cost		S	S	-	S
Engineering cost		+	-	S	-
Service cost		-	+	+	+
Maturity		-	-	S	+
	Σ-	7	7	5	3
	ΣS	1	3	4	3
	Σ+	5	3	4	7
	Pos.	3	4	2	1

from master paper Dag Jostein Klever, 2009

Evolution of Design Options



Conclusions

Evolving multiple concepts increases insight and understanding (LEAN product development: set-based design, SE: Pugh matrix)

Articulation of criteria sharpens evaluation

The discussion about the Pugh matrix is more valuable than final bottomline summation

Delaying decisions may help to keep options (Lean Product Development: late decision making, finance: real options)

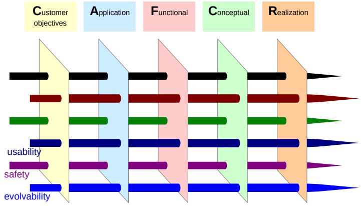
Qualities as Integrating Needles

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

Abstract

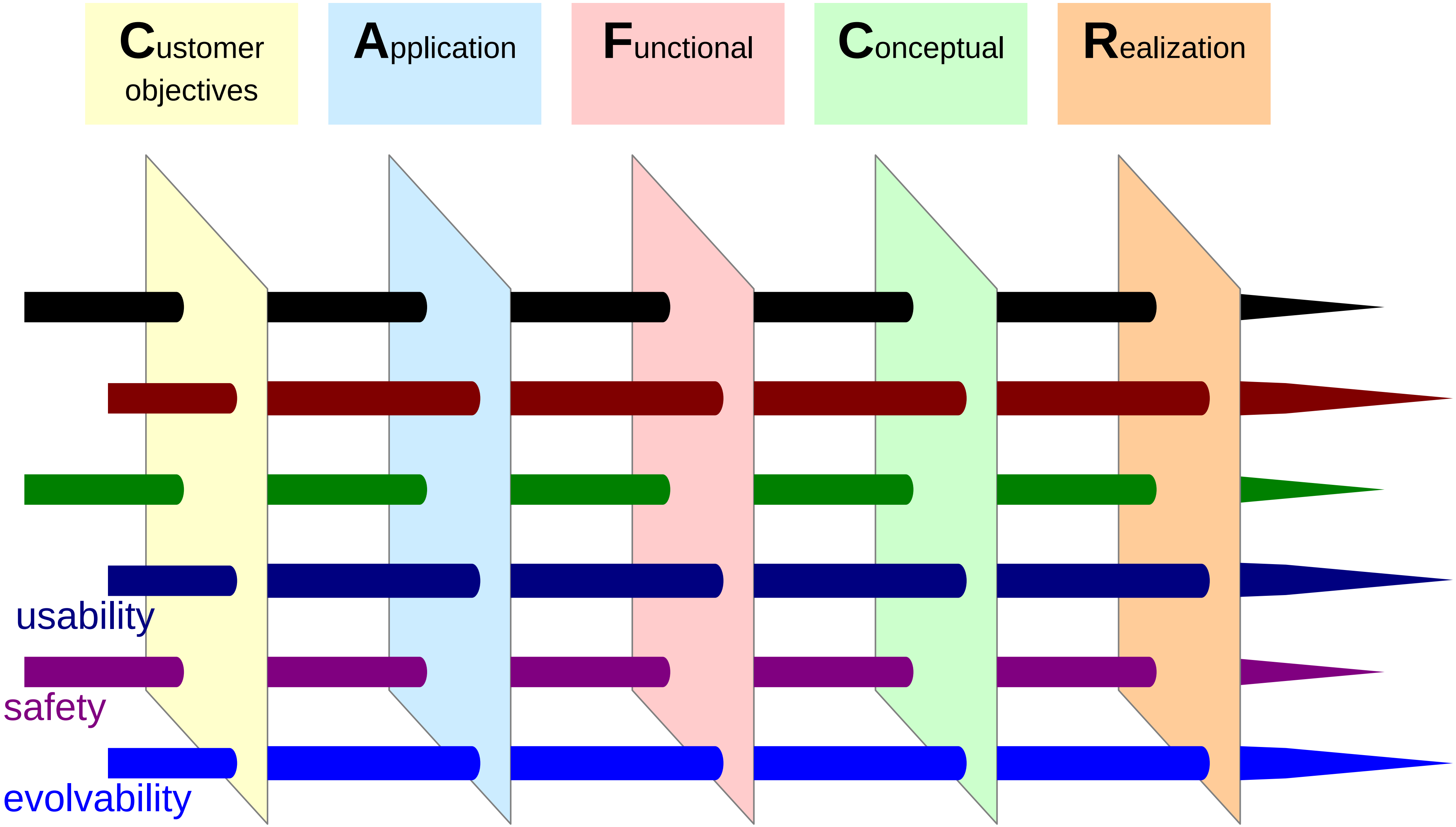
Many stakeholder concerns can be specified in terms of qualities. These qualities can be viewed from all 5 “CAFCR” viewpoints. In this way qualities can be used to relate the views to each other.

The meaning of qualities for the different views is described. A checklist of qualities is provided as a means for architecting. All qualities in the checklist are described briefly.



November 18, 2023
status: finished
version: 1.3

Quality needles as generic integrating concepts



Security as example through all views

Customer objectives	Application	Functional	Conceptual	Realization
sensitive information trusted	selection classification people information authentication badges passwords locks / walls guards administrators	functions for administration authentication intrusion detection logging quantification	cryptography firewall security zones authentication registry logging	*specific* algorithms interfaces libraries servers storage protocols
 not trusted	social contacts open passwords blackmail burglary fraud unworkable procedures	missing functionality wrong quantification	holes between concepts	bugs buffer overflow non encrypted storage poor exception handling

desired characteristics, specifications & mechanisms

threats

Quality Checklist

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
- upgradeability
- 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, et cetera)
- size, weight
- accuracy

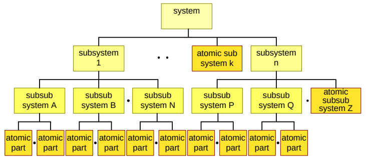
System Partitioning Fundamentals

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

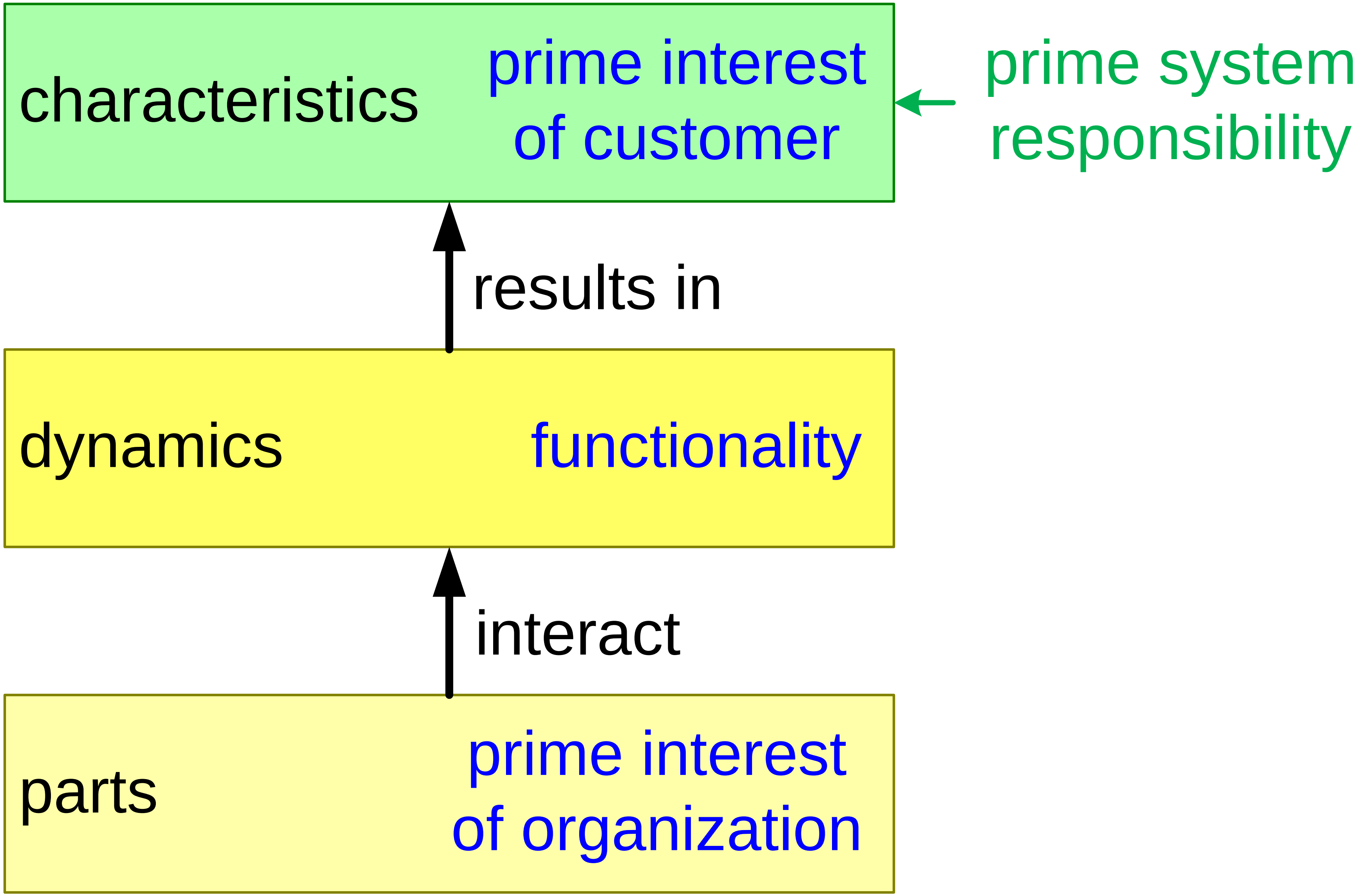
Abstract

The fundamental concepts and approach system partitioning are explained. We look at physical decomposition and functional decomposition in relation to supply chain, lifecycle support, project management, and system specification and design.

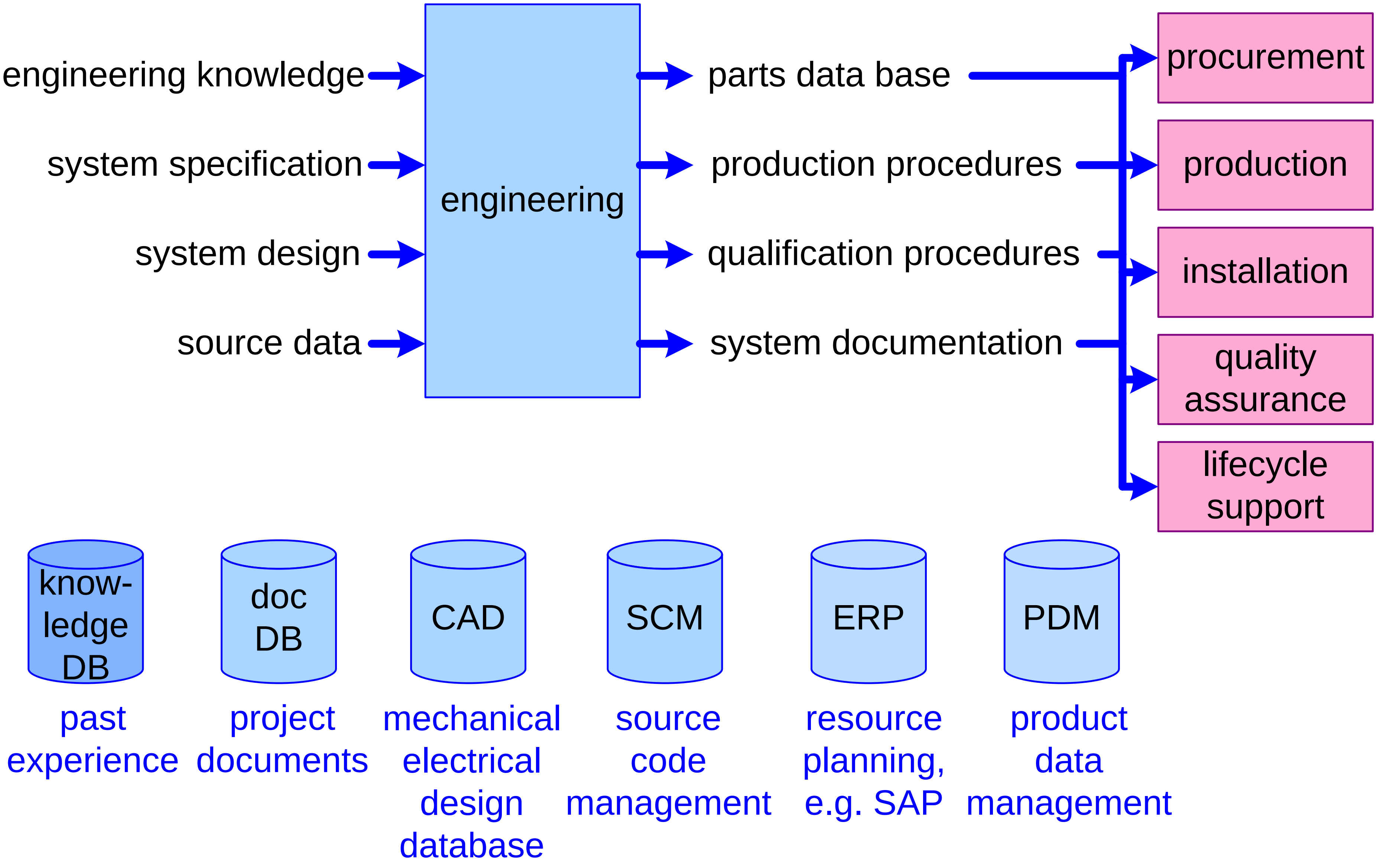
November 18, 2023
status: preliminary draft
version: 0.2



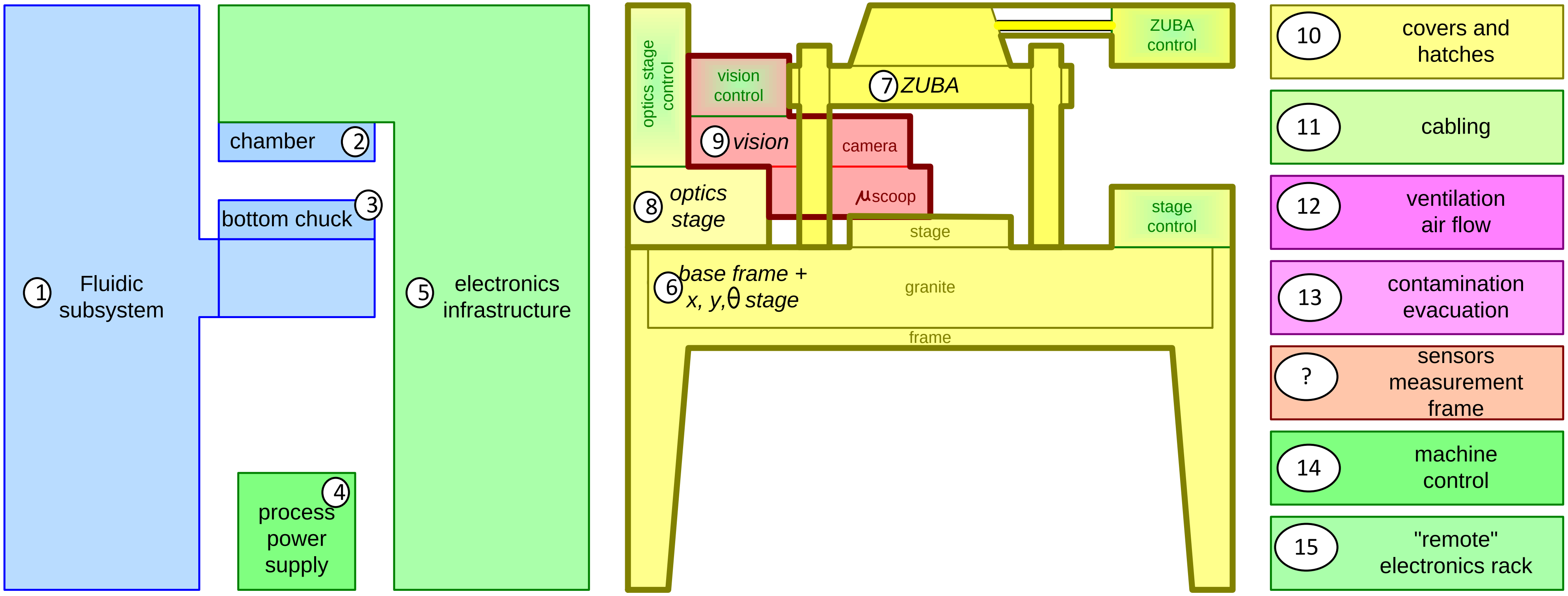
Parts, Dynamics, Characteristics


characteristics
prime interest of customer
prime system responsibility
results in
dynamics
functionality
interact
parts
prime interest of organization

Engineering



Example Physical Decomposition

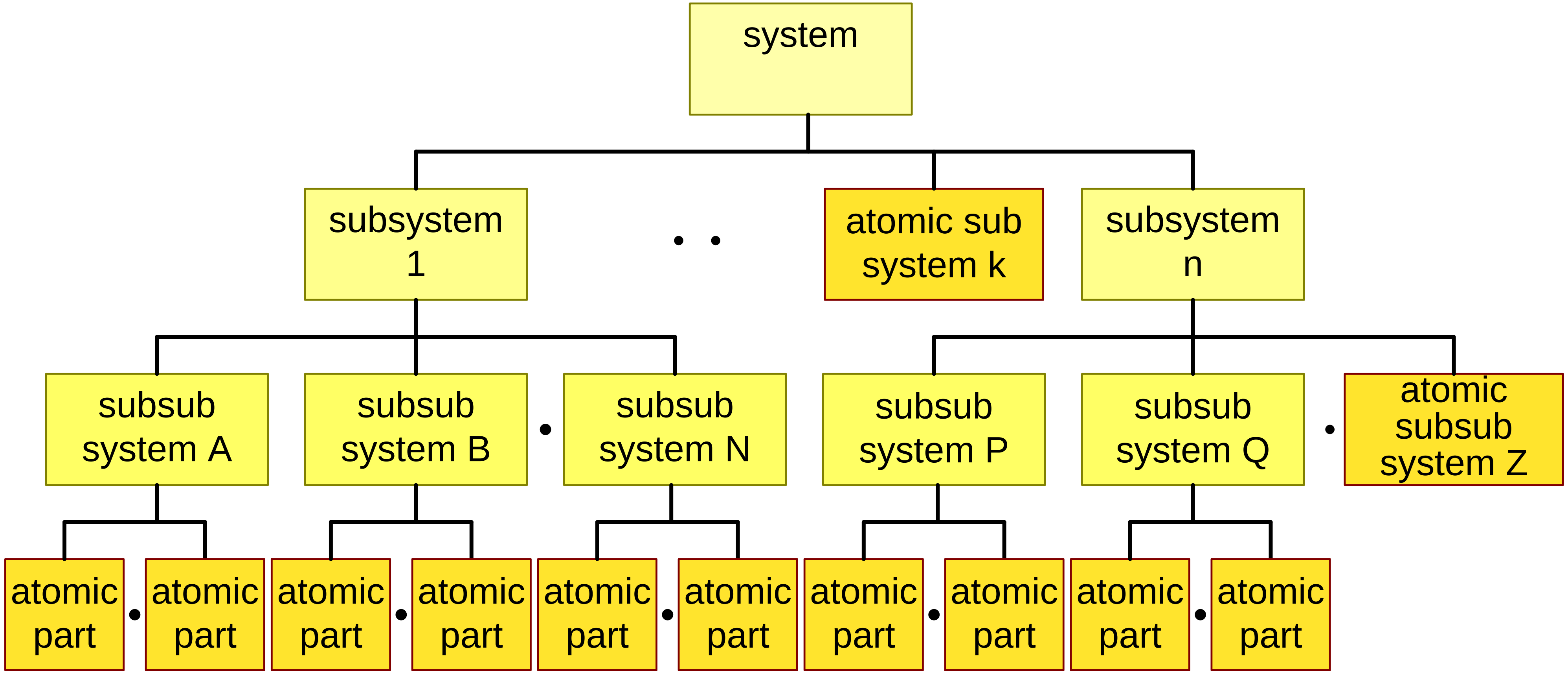


back side view

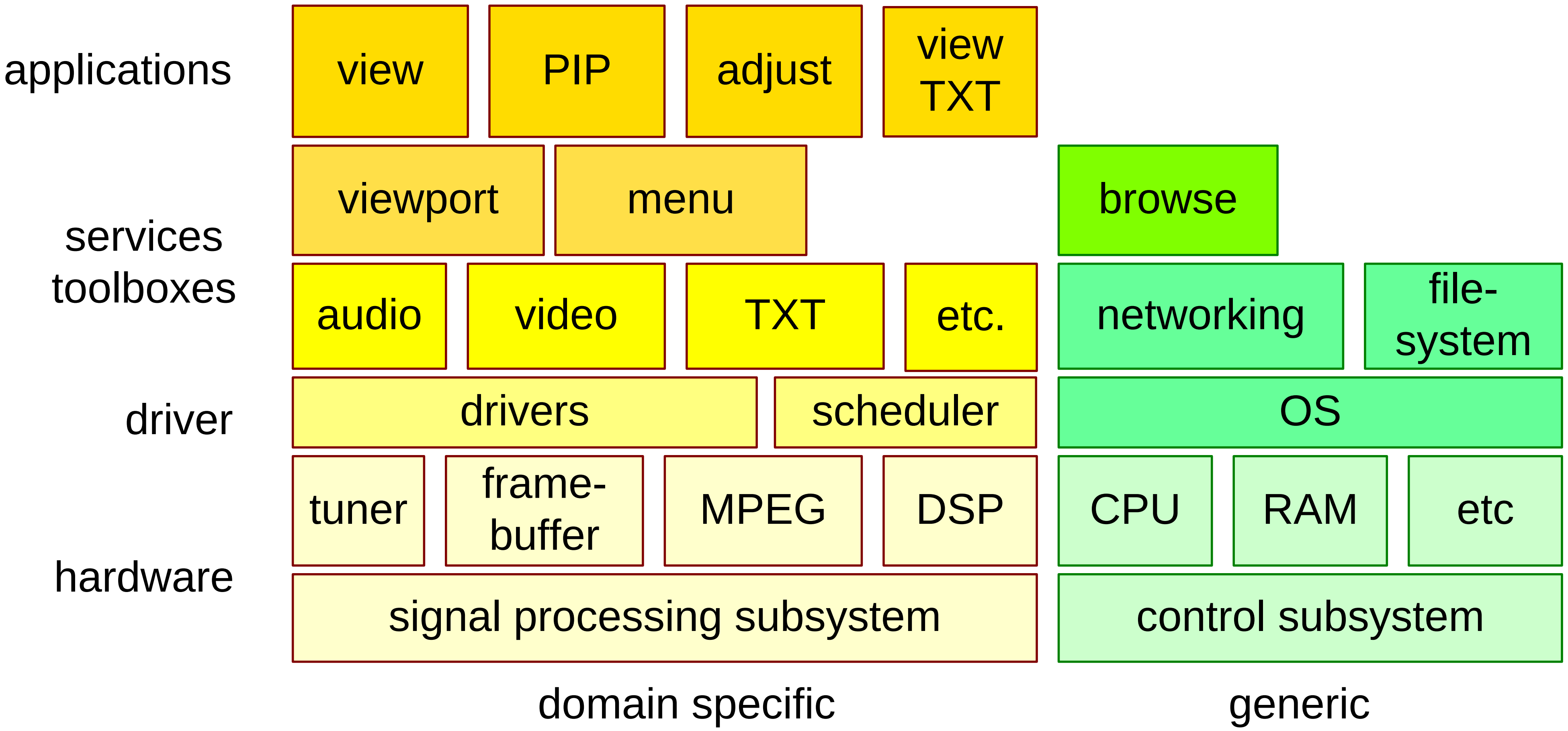
front side view

integrating

Partitioning is Applied Recursively



Software plus Hardware Decomposition



Guidelines for Partitioning

the part is cohesive

functionality and technology belongs together

the coupling with other parts is minimal

minimize interfaces

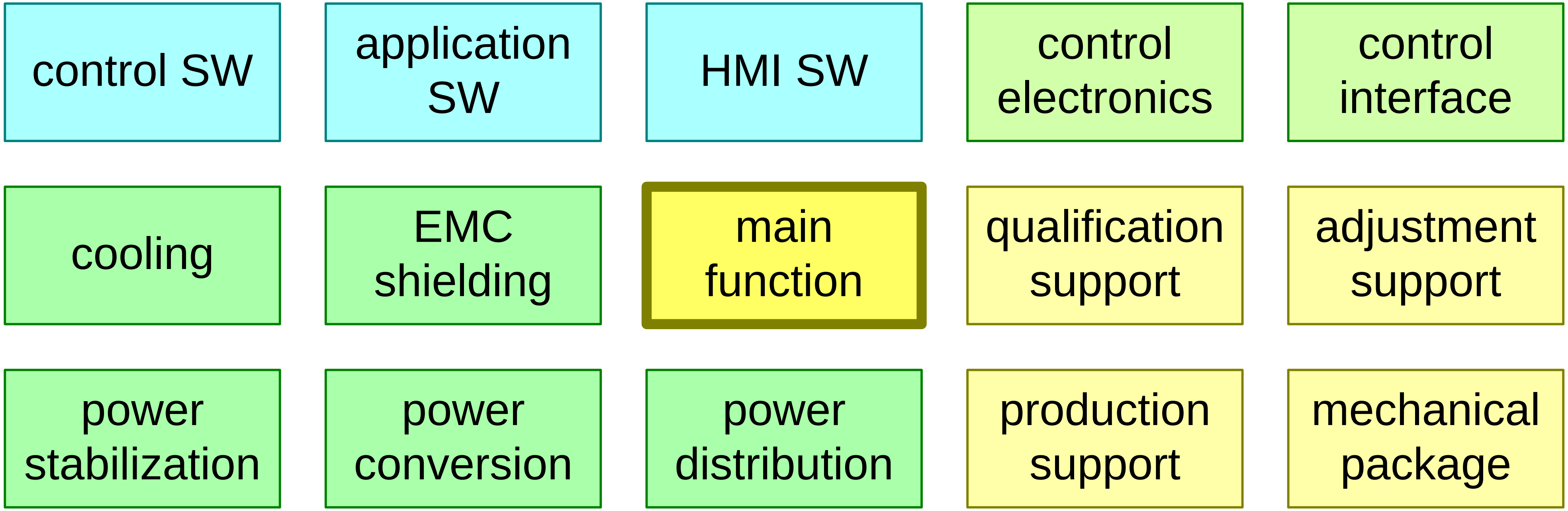
the part is selfsustained for production and qualification

can be in conflict with cost or space requirements

clear ownership of part

e.g. one department or supplier

How much self-sustained?



control SW

application SW

HMI SW

control electronics

control interface

cooling

EMC shielding

main function

qualification support

adjustment support

power stabilization

power conversion

power distribution

production support

mechanical package

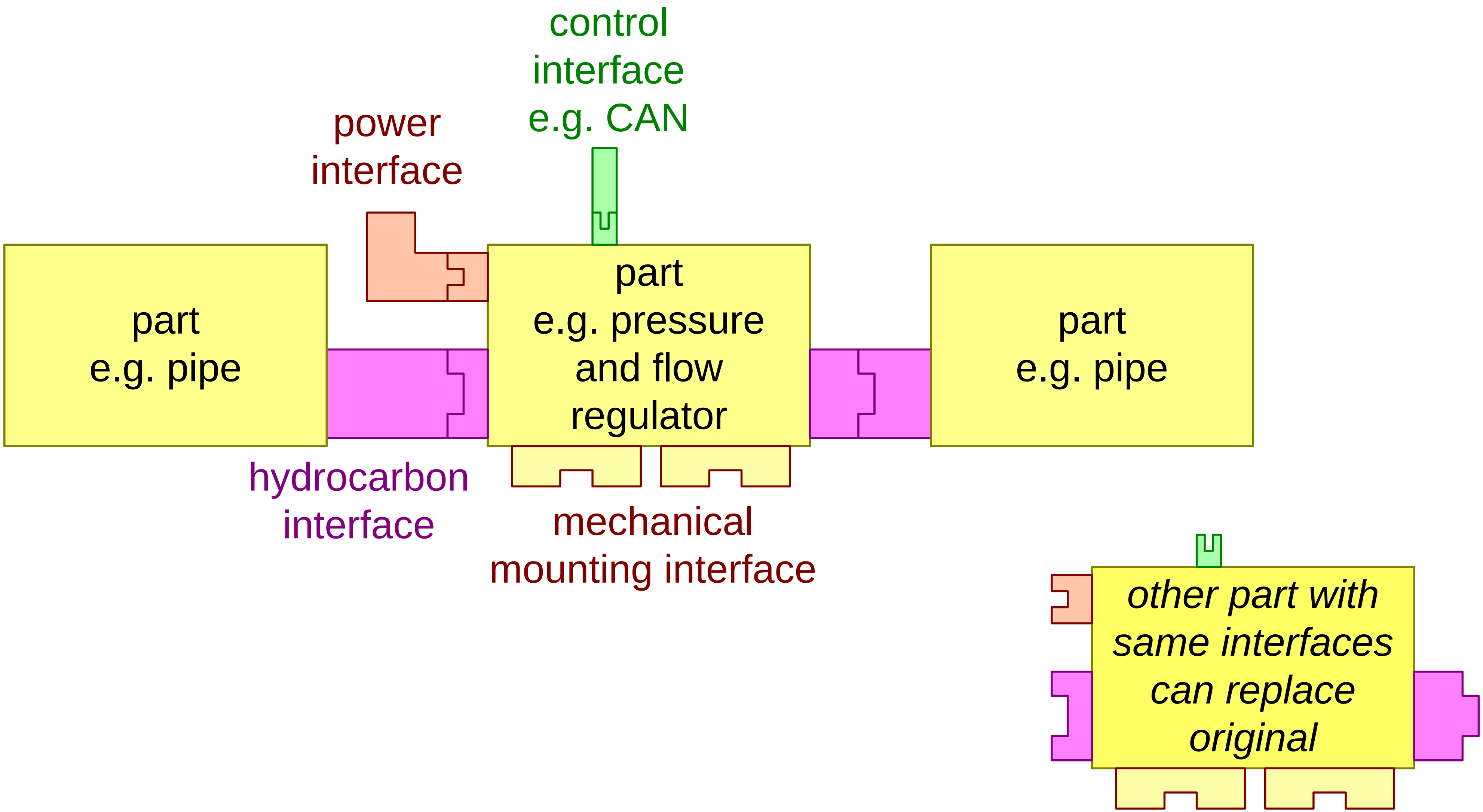
How self sustained should a part be?

trade-off:

cost/speed/space optimization

logistics/lifecycle/production flexibility clarity

Decoupling via Interfaces

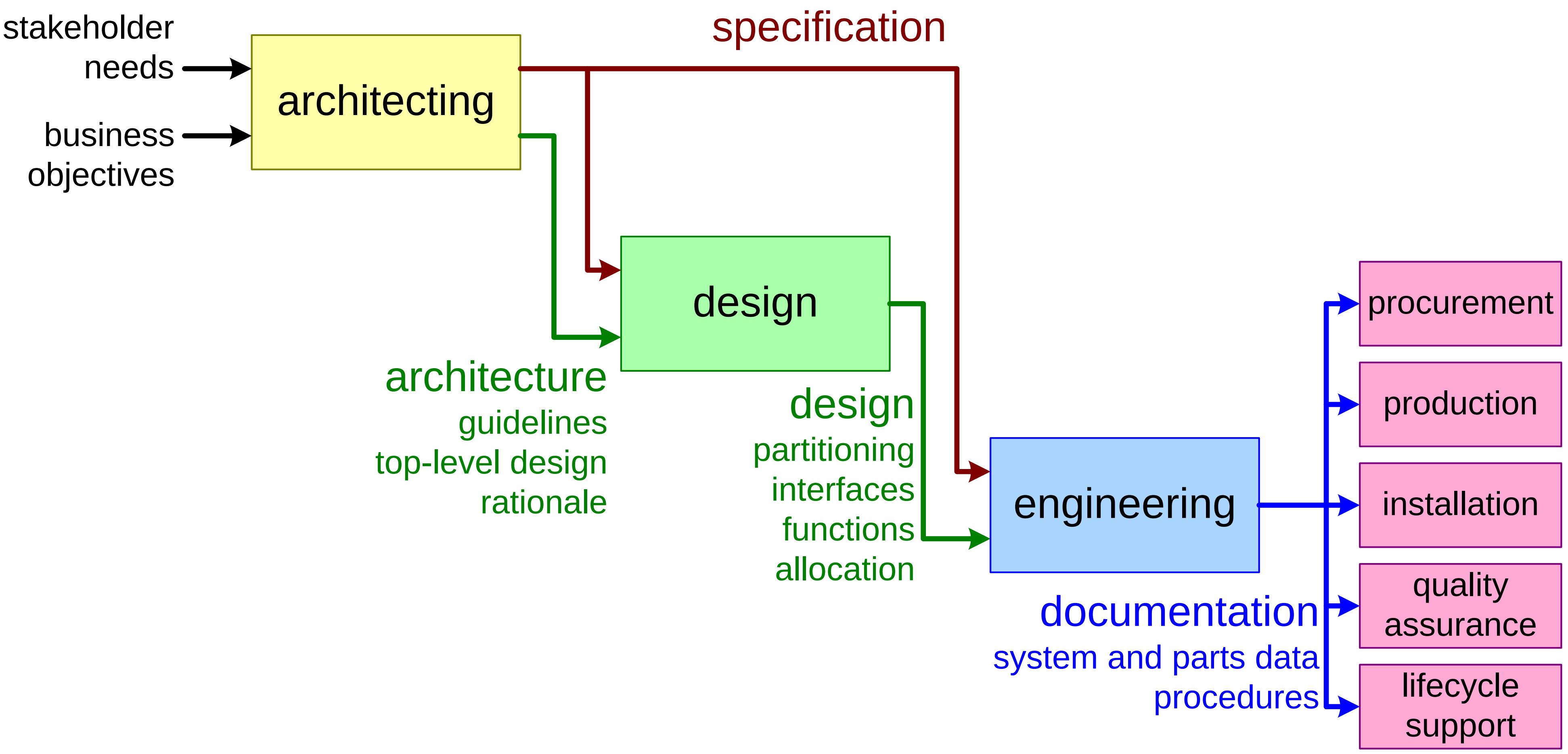


The Ideal Modularity

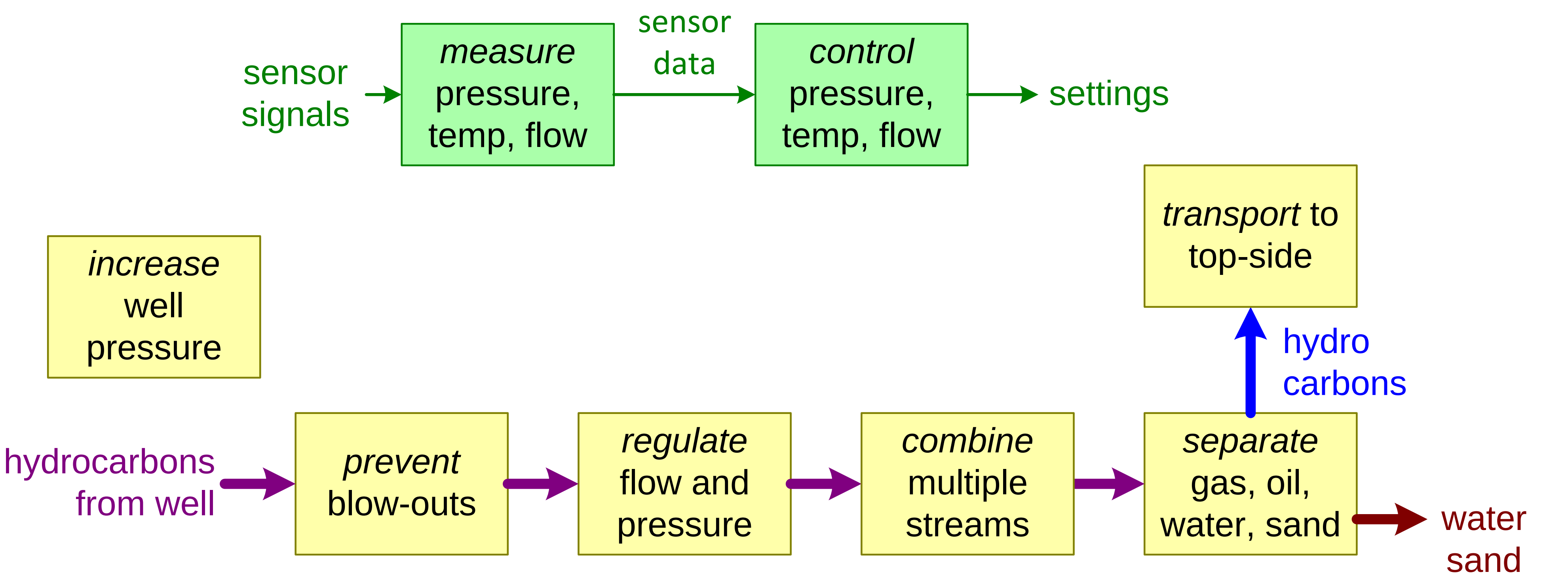
System is composed

- by using standard interfaces
- limited catalogue of variants (e.g. cost performance points)

System Creation



Simplistic Functional SubSea Example



sensor signals
measure pressure, temp, flow
sensor data
control pressure, temp, flow
settings
transport to top-side
increase well pressure
hydro carbons
hydrocarbons from well
prevent blow-outs
regulate flow and pressure
combine multiple streams
separate gas, oil, water, sand
water sand

Functional Decomposition

How does the system work and operate?

Functions describe *what* rather than *how.*

Functions are *verbs*.

Input-Process-Output paradigm.

Multiple kinds of flows:

- physical (e.g. hydrocarbons)
- information (e.g. measurements)
- control

At lower level one part ~= one function

- pump pumps, compressor compresses, controller controls

At higher level functions are complex interplay of physical parts

- e.g. regulating constant flow, pressure and temperature

Quantification

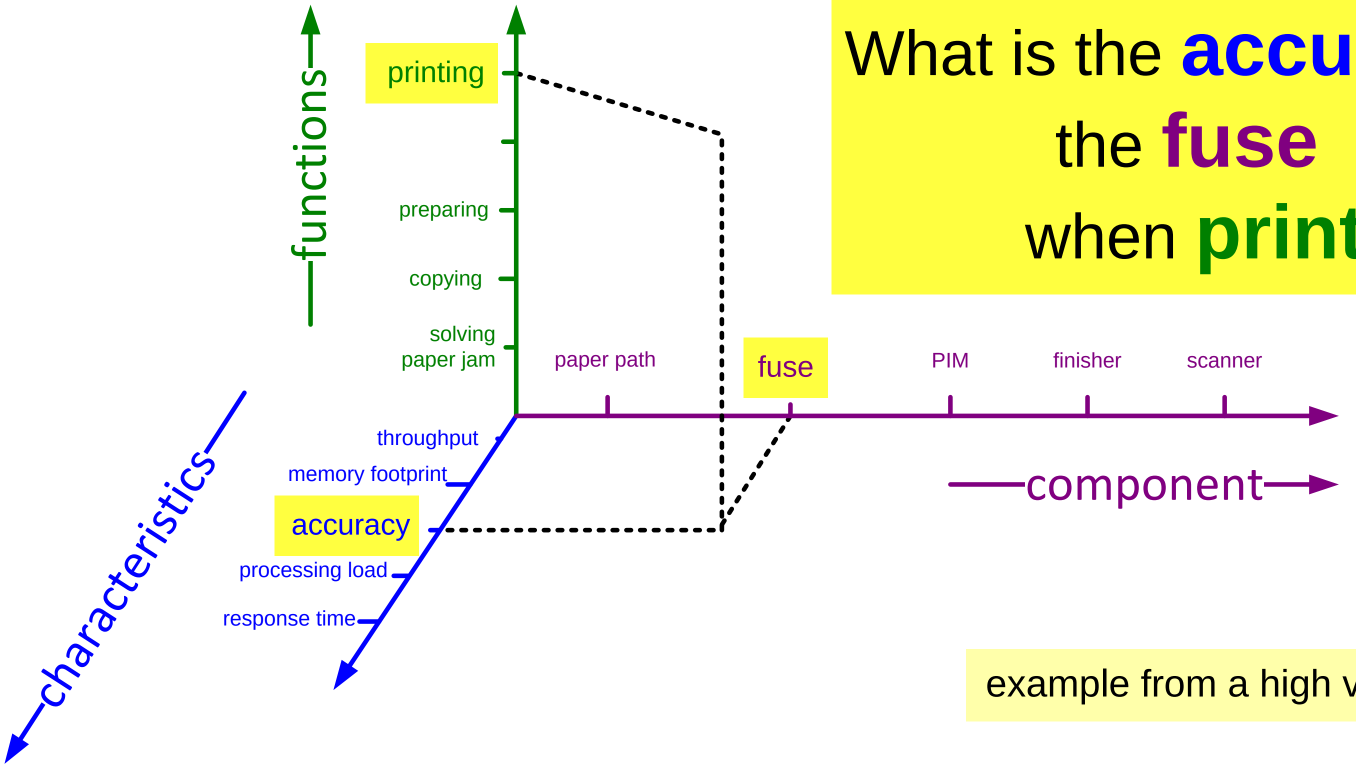
Size	2.4m * 0.7m * 1.3m
Weight	1450 Kg
Cost	30000 NoK
Reliability	MTBF 4000 hr
Throughput	3000 l/hr
Response time	0.1 s
Accuracy	+/- 0.1%

many characteristics
of a system, function or part
can be quantified

Note that quantities
*have a **unit***

Question Generator

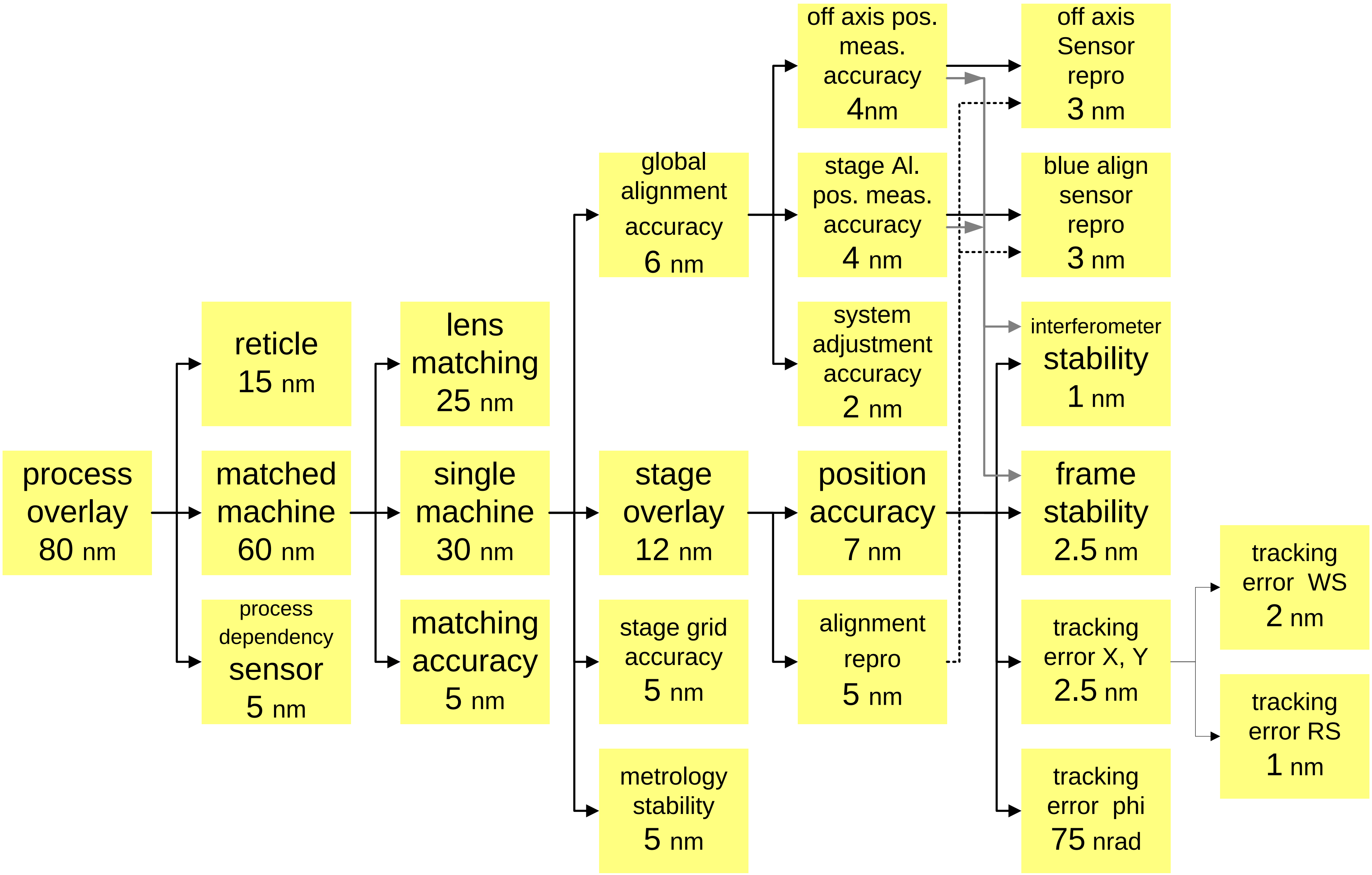
How about the **<characteristic>**
of the **<component>**
when performing **<function>**?



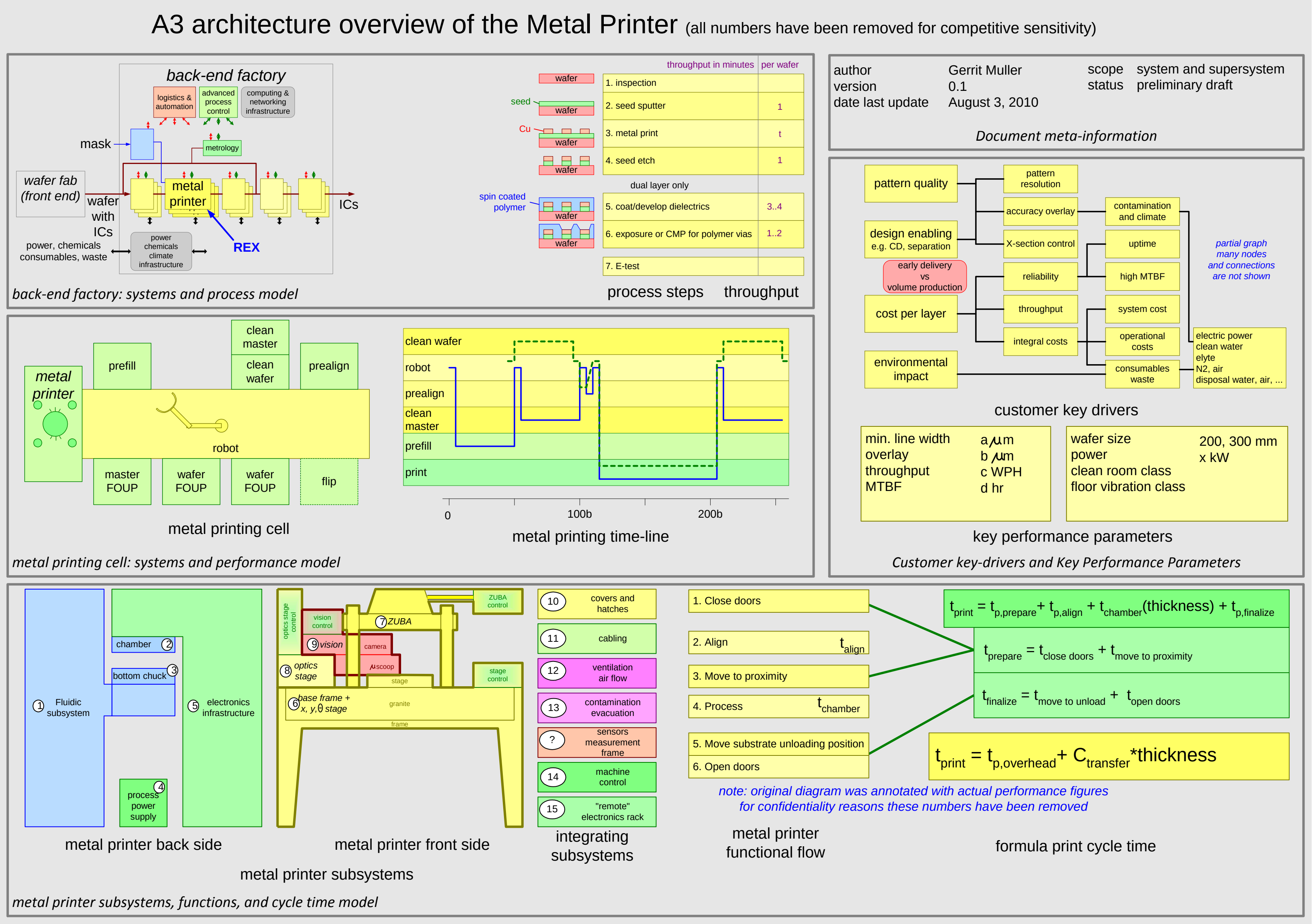
What is the **accuracy** of
the **fuse**
when **printing**

example from a high volume printer

Example Technical Budget


process overlay 80 nm
reticle 15 nm
matched machine 60 nm
process dependency sensor 5 nm
lens matching 25 nm
single machine 30 nm
matching accuracy 5 nm
global alignment accuracy 6 nm
stage overlay 12 nm
stage grid accuracy 5 nm
metrology stability 5 nm
off axis pos. meas. accuracy 4nm
stage Al. pos. meas. accuracy 4 nm
system adjustment accuracy 2 nm
position accuracy 7 nm
alignment repro 5 nm
off axis Sensor repro 3 nm
blue align sensor repro 3 nm
interferometer stability 1 nm
frame stability 2.5 nm
tracking error X, Y 2.5 nm
tracking error phi 75 nrad
tracking error WS 2 nm
tracking error RS 1 nm

Example of A3 overview



A3 architecture overview of the Metal Printer (all numbers have been removed for competitive sensitivity)

author	Gerrit Muller	scope	system and supersystem
version	0.1	status	preliminary draft
date last update	August 3, 2010		

Document meta-information

	throughput in minutes per wafer
1. inspection	
2. seed sputter	1
3. metal print	t
4. seed etch	1
dual layer only	
5. coat/develop dielectrics	3..4
6. exposure or CMP for polymer vias	1..2
7. E-test	

process steps throughput

back-end factory: systems and process model

metal printing cell

metal printing time-line

metal printing cell: systems and performance model

customer key drivers

min. line width	a μm
overlay	b μm
throughput	c WPH
MTBF	d hr

wafer size	200, 300 mm
power	x kW
clean room class	
floor vibration class	

key performance parameters

Customer key-drivers and Key Performance Parameters

metal printer back side

metal printer front side

integrating subsystems

metal printer subsystems

1. Close doors
2. Align t_{align}
3. Move to proximity
4. Process $t_{chamber}$
5. Move substrate unloading position
6. Open doors

$$t_{print} = t_{p,prepare} + t_{p,align} + t_{chamber}(thickness) + t_{p,finalize}$$

$$t_{prepare} = t_{close\ doors} + t_{move\ to\ proximity}$$

$$t_{finalize} = t_{move\ to\ unload} + t_{open\ doors}$$

$$t_{print} = t_{p,overhead} + C_{transfer}*thickness$$

note: original diagram was annotated with actual performance figures for confidentiality reasons these numbers have been removed

metal printer functional flow

formula print cycle time

metal printer subsystems, functions, and cycle time model