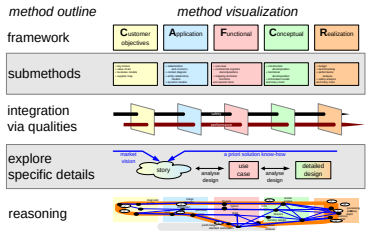
Multi-Objective Embedded Systems Design, based on CAFCR

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

Abstract

This course provides submethods for the CAFCR views. Qualities are provided to integrate the views. Story telling is introduced as a means to explore requirements. In a second iteration over the CAFCR views a thread of reasoning over the views is created.

October 11, 2020
status: draft
version: 0.2



Module CAFCR course info

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

Abstract

This module provides the information about the CAFCR course: “Multi-Objective Embedded Systems Design, based on CAFCR”.

October 11, 2020
status: draft
version: 0

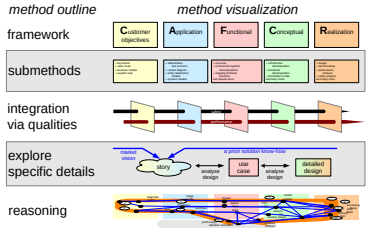
logo
TBD

Multi-Objective Embedded Systems design, based on CAFCR

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

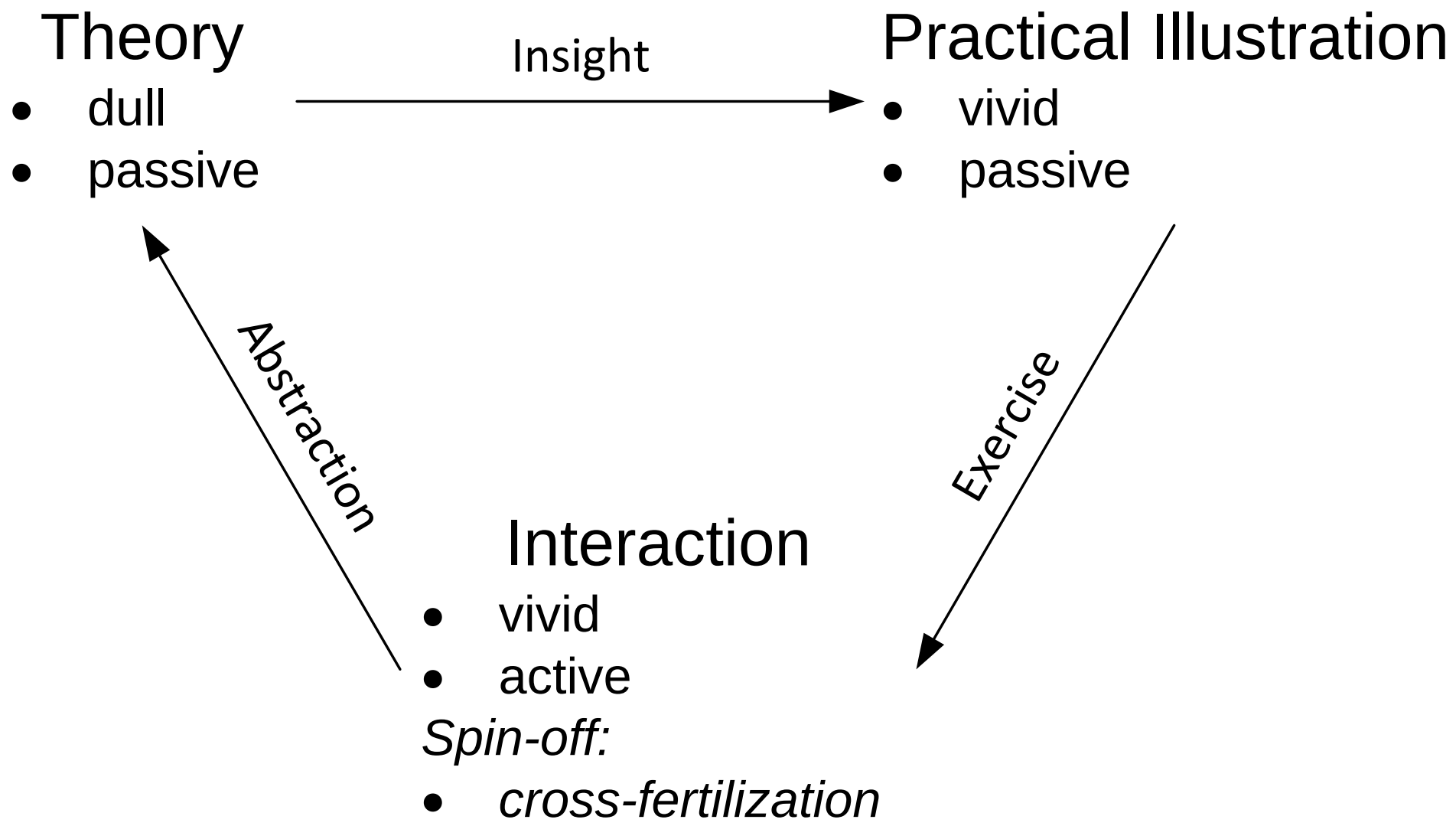
Abstract

The course Multi-Objective Embedded Systems Design, based on the CAFCR-views, is described. The program existing of 10 modules is described. The course format, iterating theory, illustration and interaction is explained. The course heavily emphasizes the practical application of the method. In every module the theory is applied on the participants products. Teams of 4 participants with the same background apply the method on their own product and report the results.

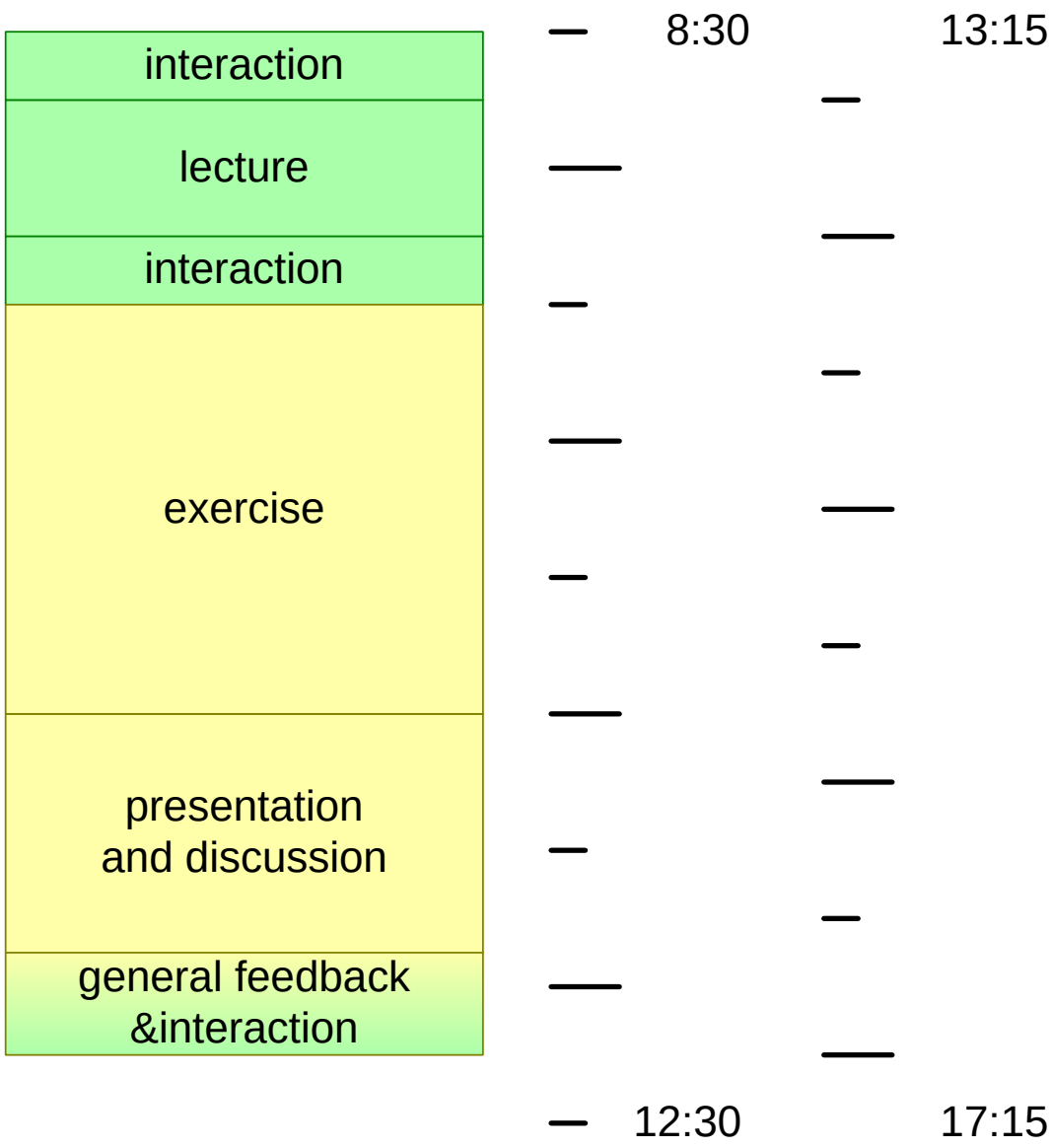


October 11, 2020
status: preliminary draft
version: 0.1

Complementing Forms



Template of One Session



Course Program

Time	Subject
Session 1	Method overview
Session 2	Functional View
Session 3	Customer Views
Session 4	Design Views
Session 5	Story telling
Session 6	Qualities
Session 7	Customer Views (2)
Session 8	Functional View (2), Cases
Session 9	Design Views (2)
Session 10	wrap up

Rules of the Broadcast Part

- 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.

Rules of the Interactive and the Practice Part

- 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.

Evaluation of the Expectations

Please write your name and expectations with a marker on one A4 page.
Describe your expectations as one-liner or in a few keywords.
These pages will be displayed on the wall of the room.
At the end of the course we will look back on these expectations, with the purpose of two-way learning.

Module Architecting Method Overview

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

Abstract

This module described the overview of the complete architecting method.

October 11, 2020
status: draft
version: 0

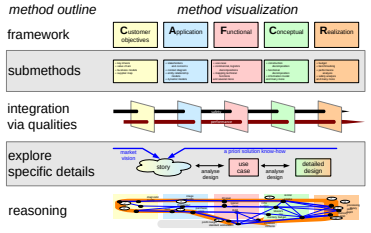
logo
TBD

Overview of CAFCR and Threads of Reasoning

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

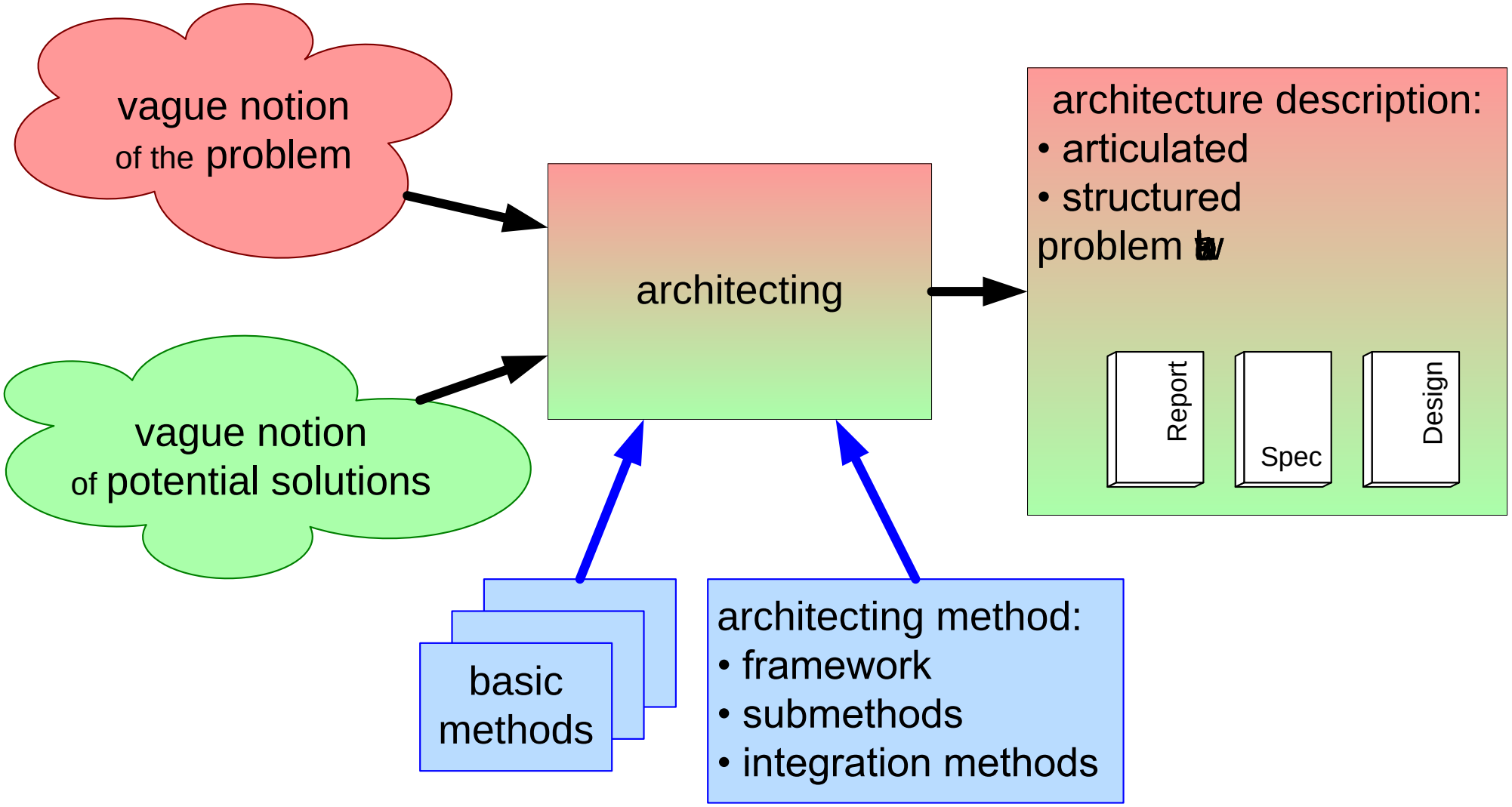
Abstract

The described architecting method uses the CAFCR model as starting point. Qualities are used as orthogonal dimension to integrate the CAFCR views. Story telling is used to add specifics. Threads of reasoning combine all the information into a coherent overview.



October 11, 2020
status: finished
version: 1.5

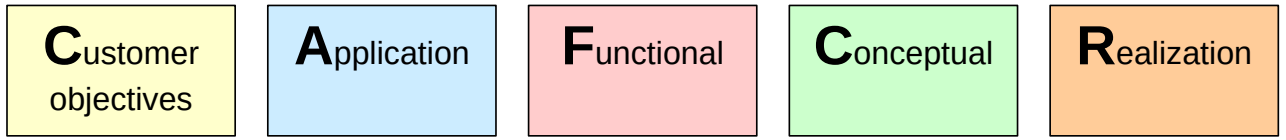
From vague notions to articulate and structured



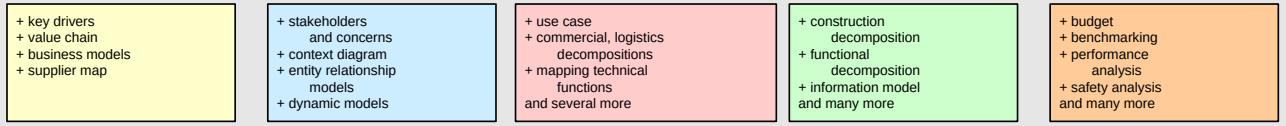
Overview of architecting method

method outline *method visualization*

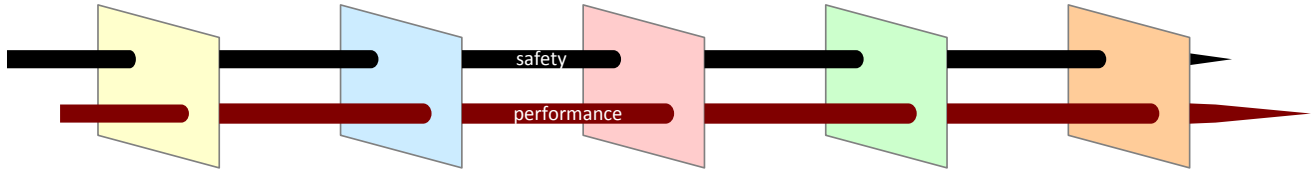
framework



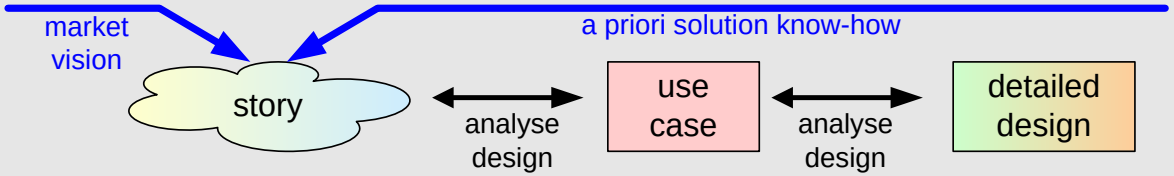
submethods



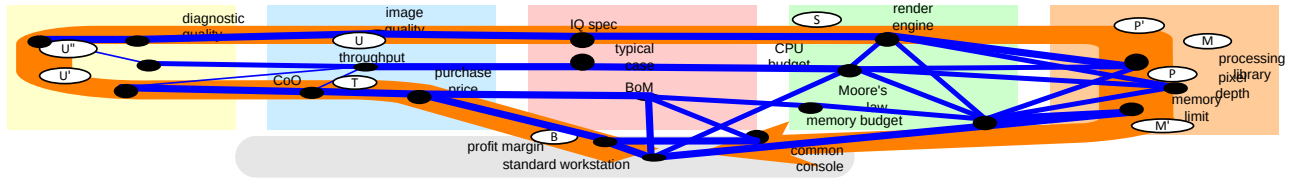
integration via qualities



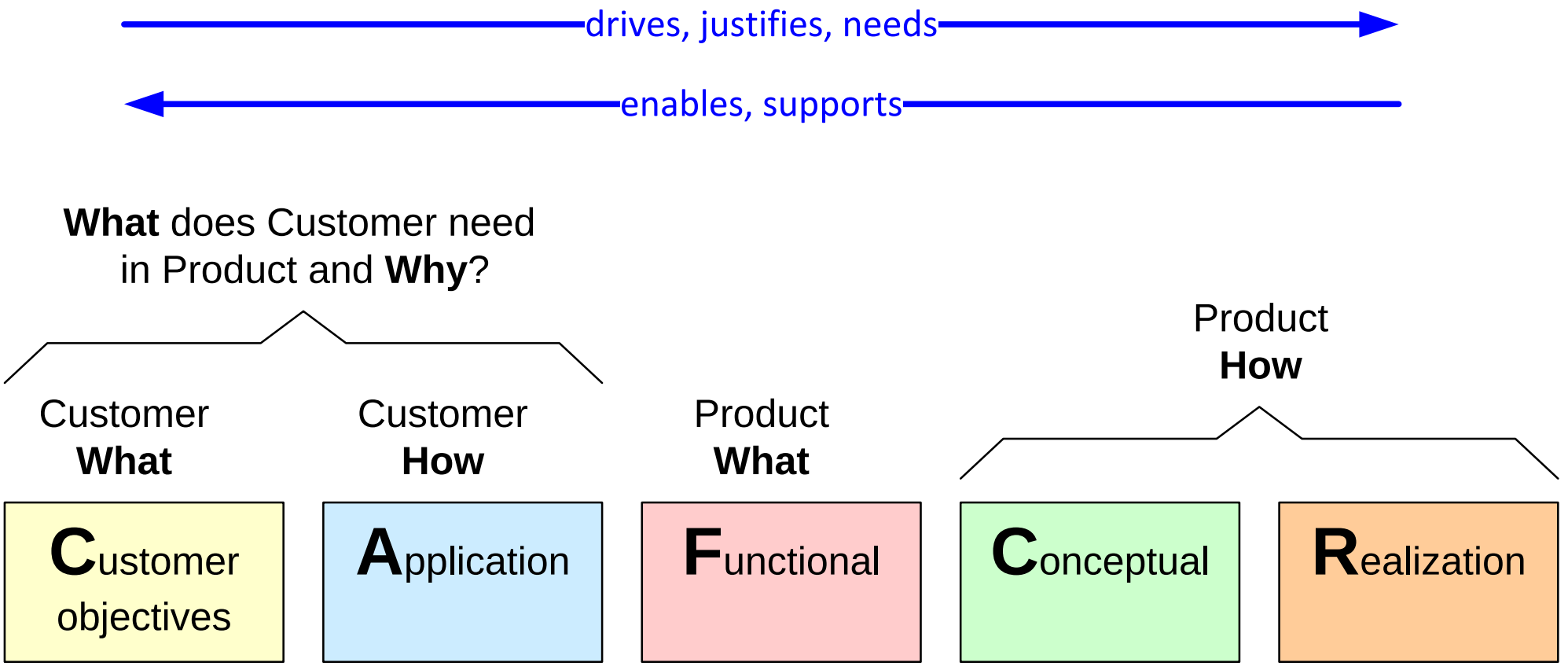
explore specific details



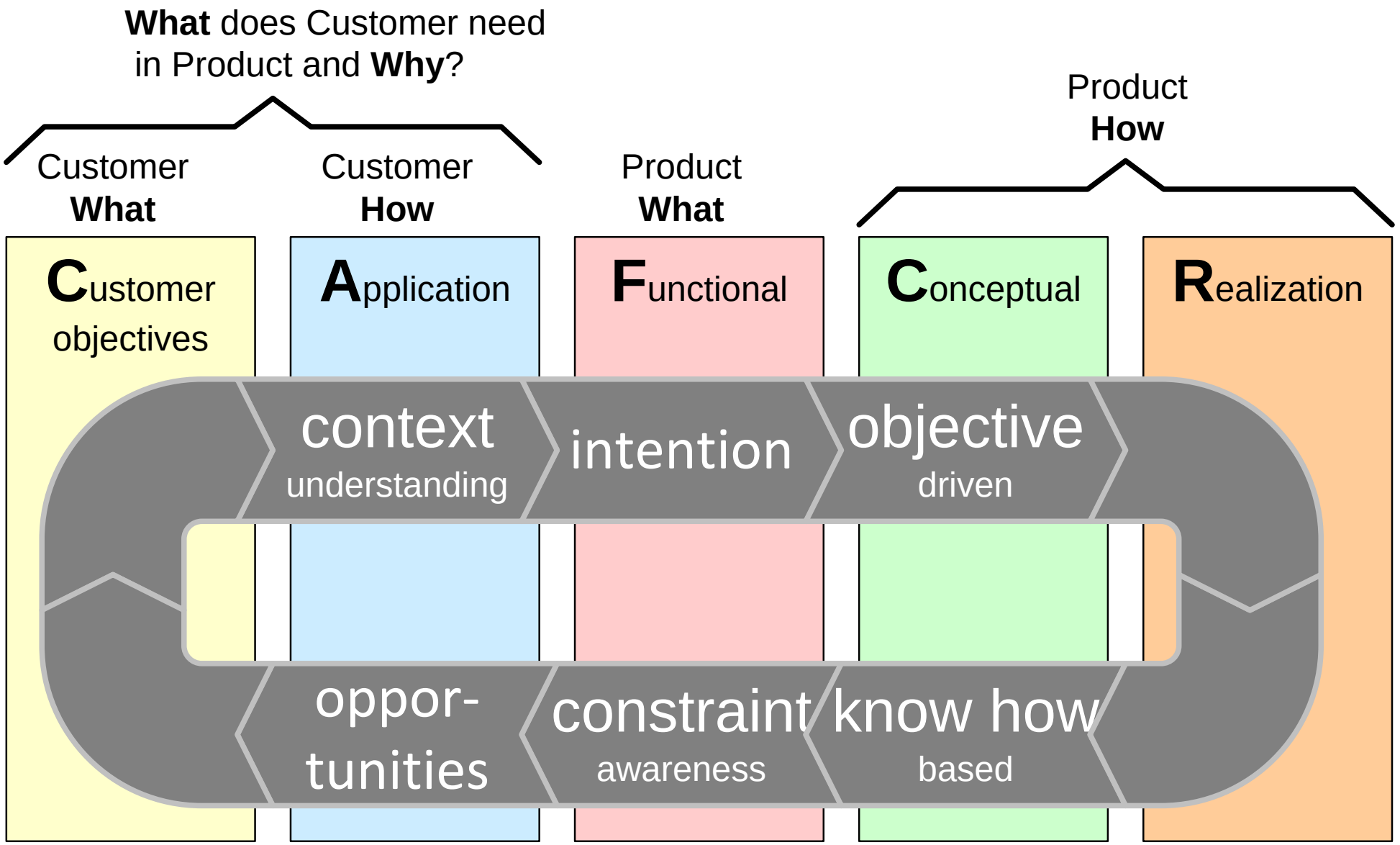
reasoning



The “CAFCR” model



Five viewpoints for an architecture



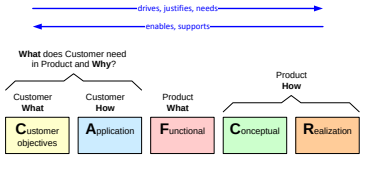
Short introduction to basic “CAFCR” model

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

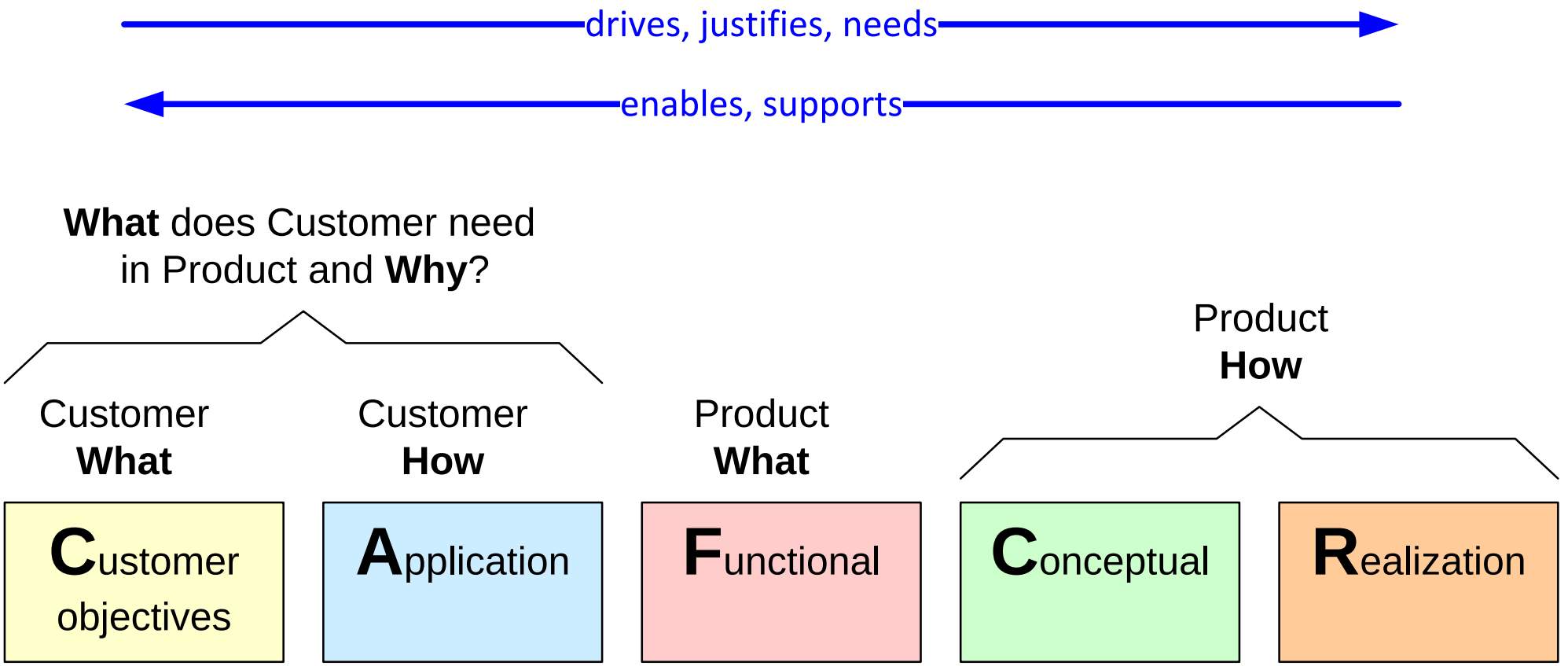
Abstract

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.

October 11, 2020
status: draft
version: 0.4

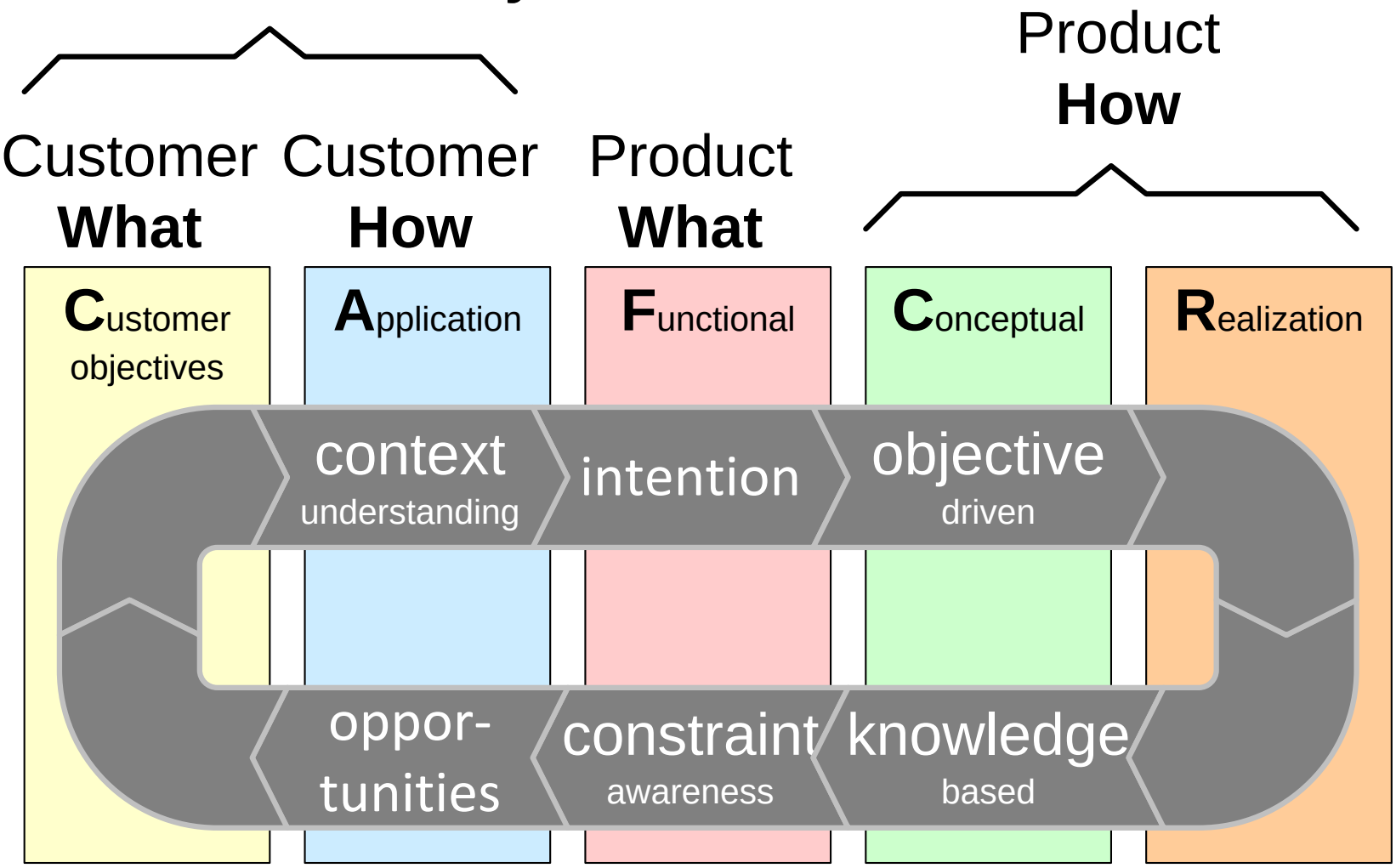


The “CAFCR” model

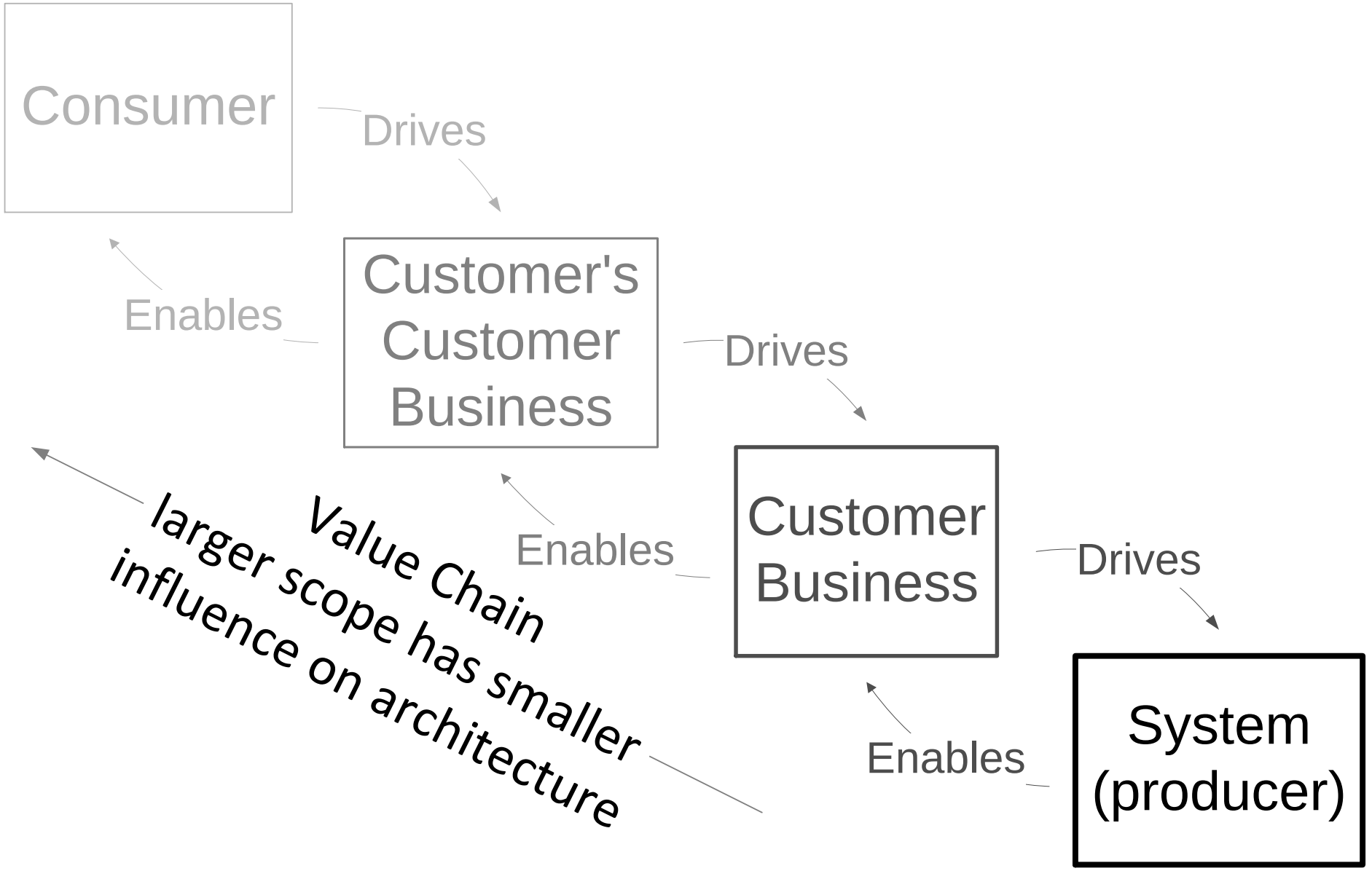


Integrating CAFCR

What does Customer need in Product and **Why**?



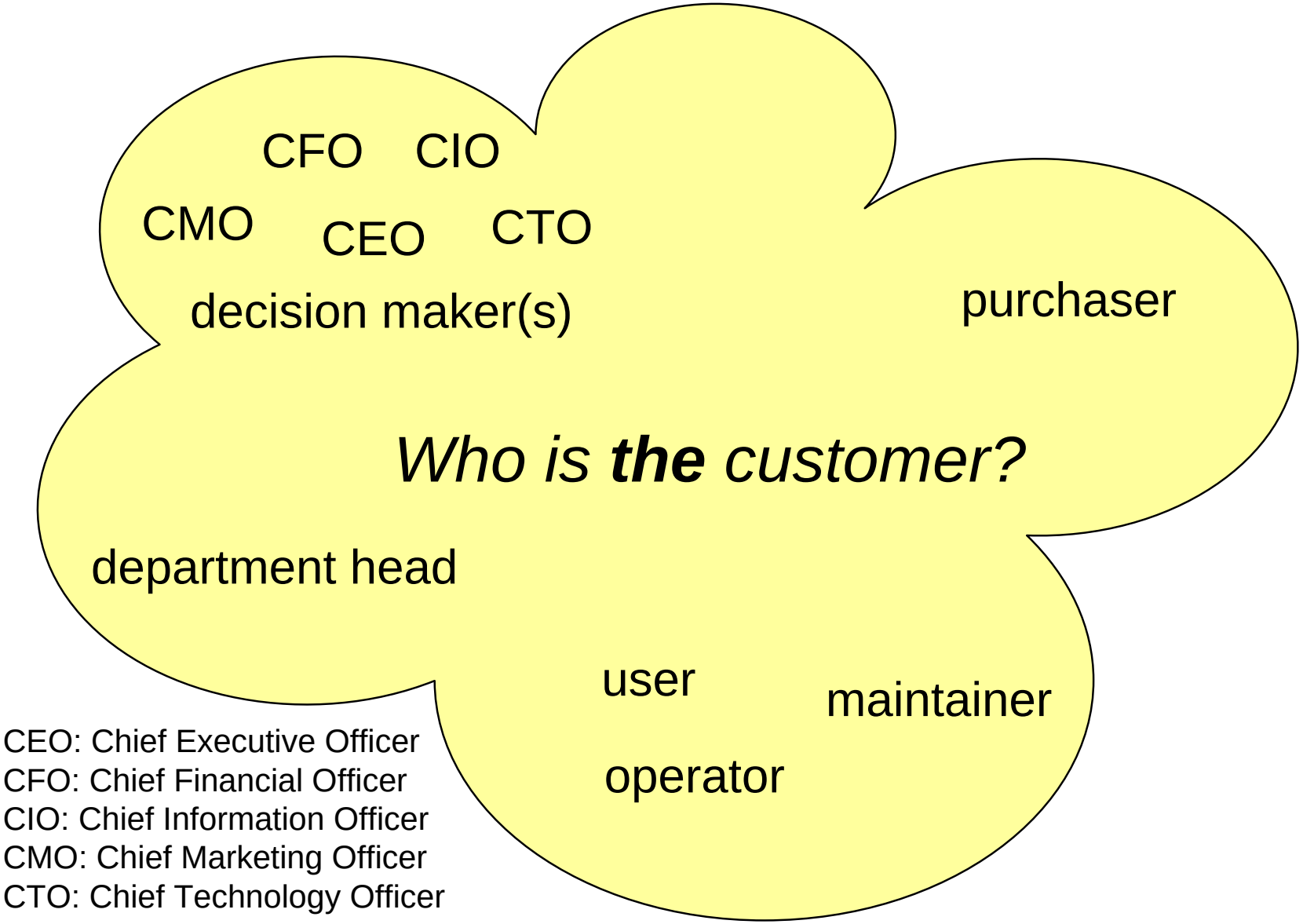
CAFCR can be applied recursively



Market segmentation

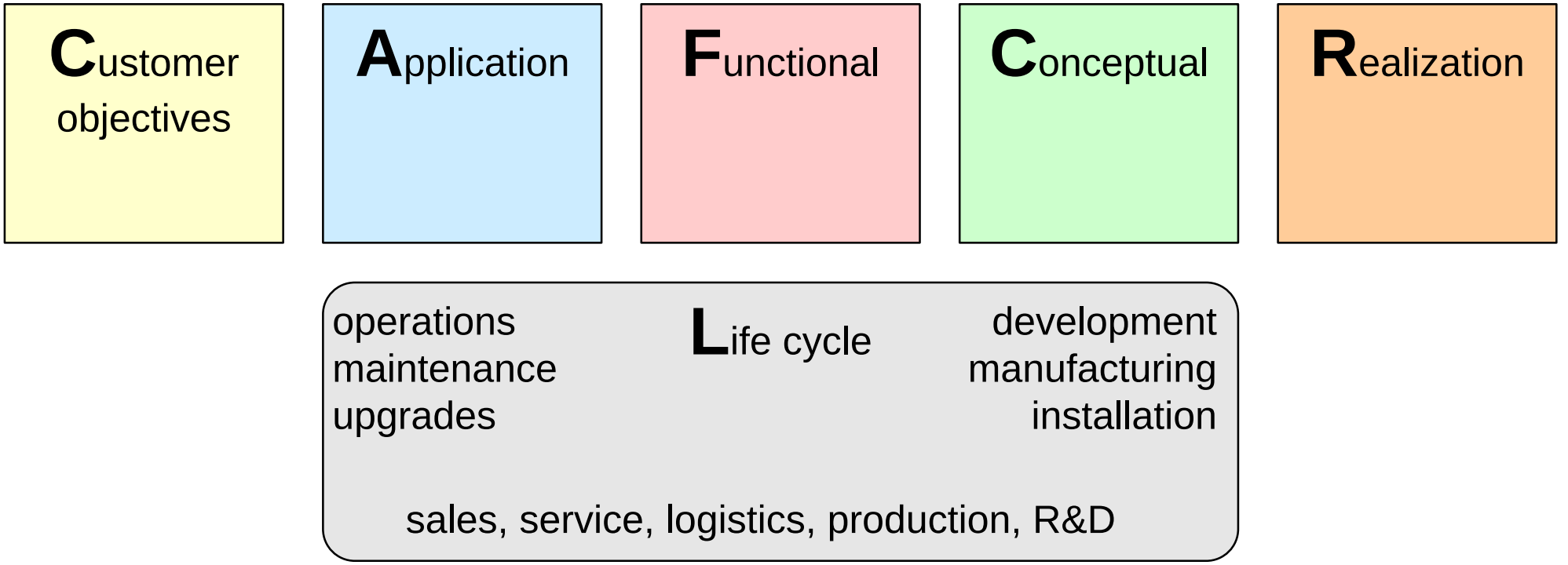
segmentation axis	examples
geographical	USA, UK, Germany, Japan, China
business model	profit, non profit
economics	high end versus cost constrained
consumers	youth, elderly
outlet	retailer, provider, OEM, consumer direct

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



Exercise Architecting Method Overview

- make a bottom-up analysis of your product:
  1. realization
  2. conceptual
  3. functional
  4. application
  5. customer objectives
  6. qualities
- use time boxes of 15 minutes per view
- show the most dominant decomposition of that view, as diagram or as a list

Module Functional View

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

Abstract

This module addresses the Functional View.

October 11, 2020
status: draft
version: 0

logo
TBD

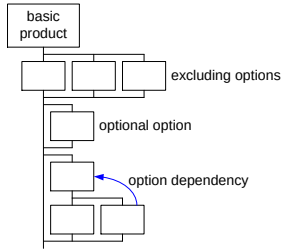
The functional view

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

Abstract

The purpose of the functional view is described. A number of methods or models is given to use in this view: (use) case descriptions, commercial decomposition function and feature specifications performance models and specifications, information models. The role of standards is discussed.

October 11, 2020
status: preliminary draft
version: 1.0



Example personal video recorder use case contents

typical use case(s)	worst case, exceptional, or change use case(s)
interaction flow (functional aspects) select movie via directory start movie be able to pause or stop be able to skip forward or backward set recording quality	**functional** multiple inputs at the same time extreme long movie directory behaviour in case of extreme many short movies
performance and other qualities (non-functional aspects) response times for start / stop response times for directory browsing end-of-movie behaviour relation recording quality and storage	**non-functional** response time with multiple inputs image quality with multiple inputs insufficient free space response time with many directory entries replay quality while HQ recording

Recommendations for working with use cases

+ combine related functions in one use case

- do not make a separate use case for every function

+ include non-functional requirements in the use cases

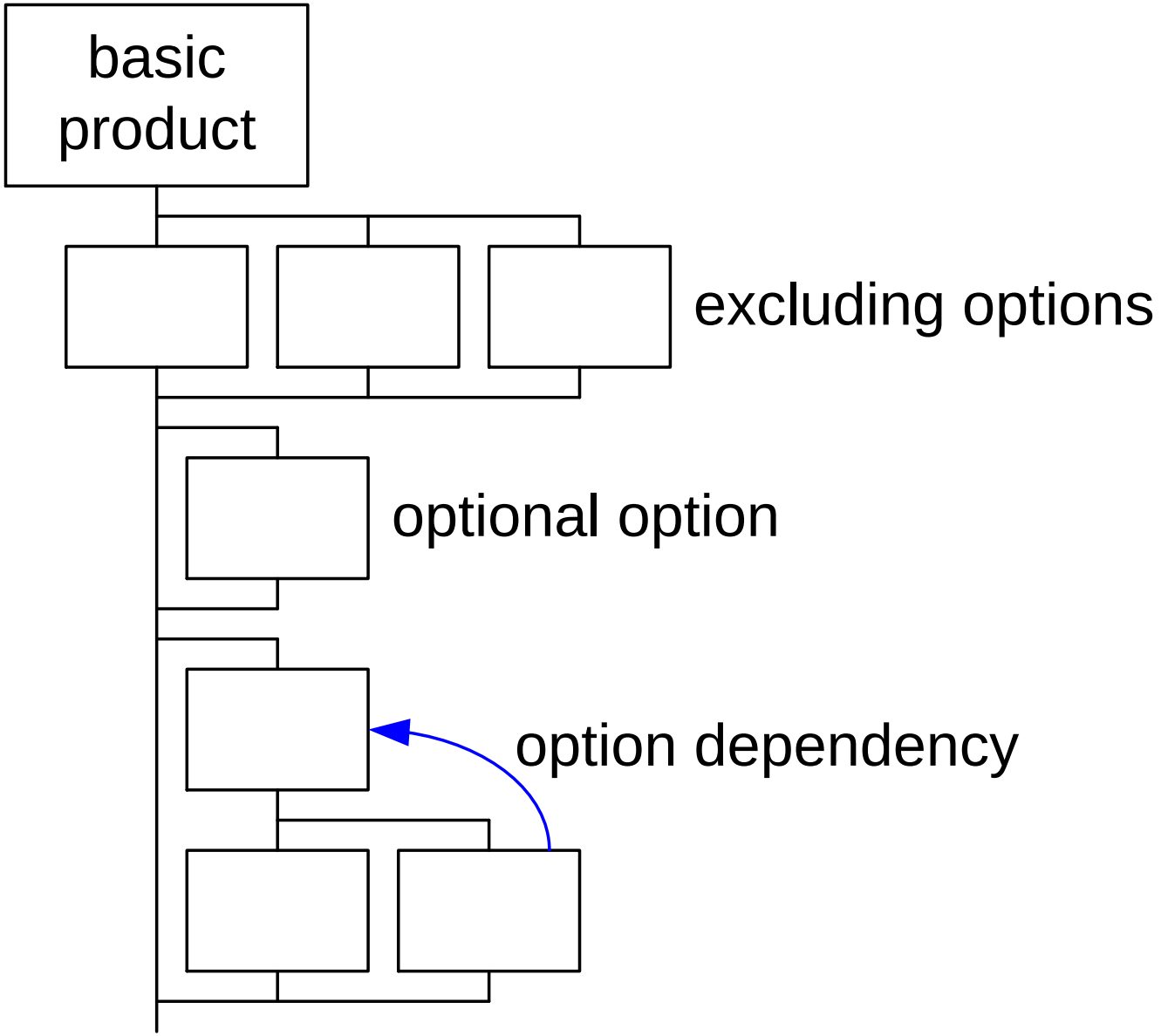
+ minimise the amount of required *worst case* and *exceptional use cases*

- excessive amounts of use cases propagate to excessive implementation efforts

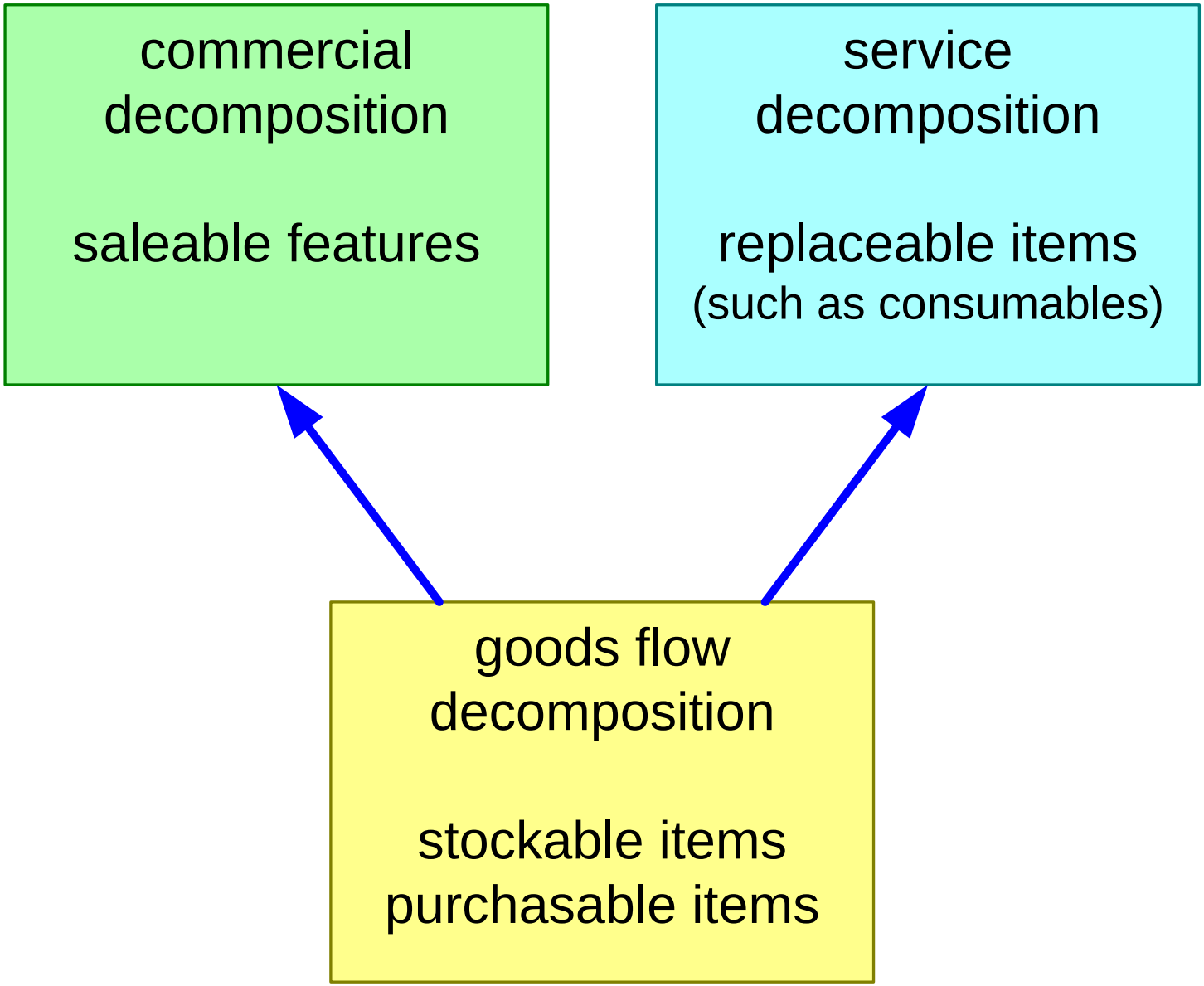
+ reduce the amount of these use cases in steps

- a few well chosen *worst case* use cases simplifies the design

Commercial Decomposition



Logistic decompositions for a product



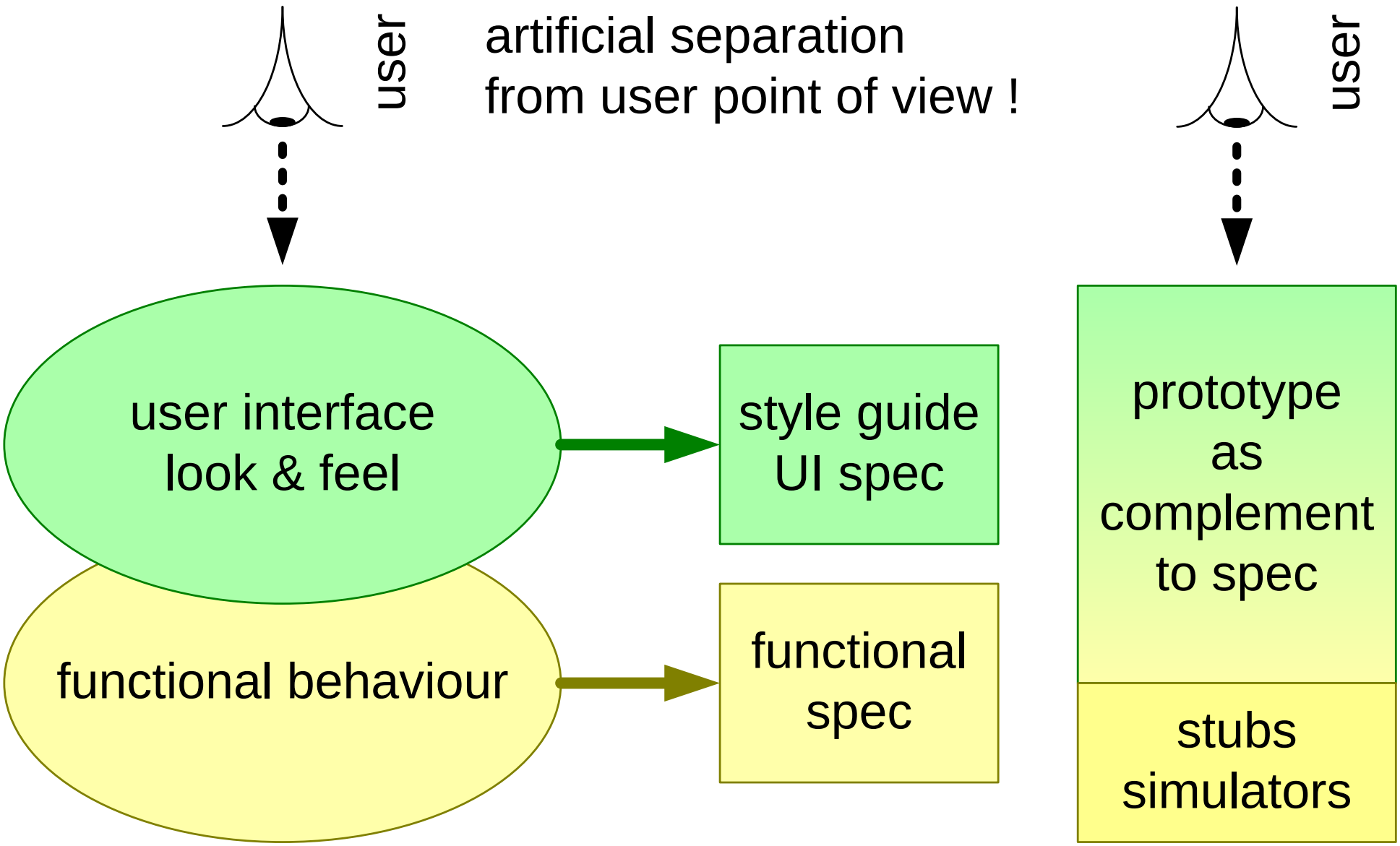
Mapping technical functions on products

technical functions \ *products*	home cinema system	flat screen cinema TV	bedroom TV
HD display	+	+	-
SD->HD up conversion	+	+	-
HD->SD down conversion	+	+	o
HD storage	o	-	-
SD storage	o	-	o
HD IQ improvement	+	+	-
SD IQ improvement	+	+	+
HD digital input	+	+	o
SD digital input	+	+	o
SD analog input	o	+	+
6 HQ channel audio	+	o	-
2 channel audio	-	+	+

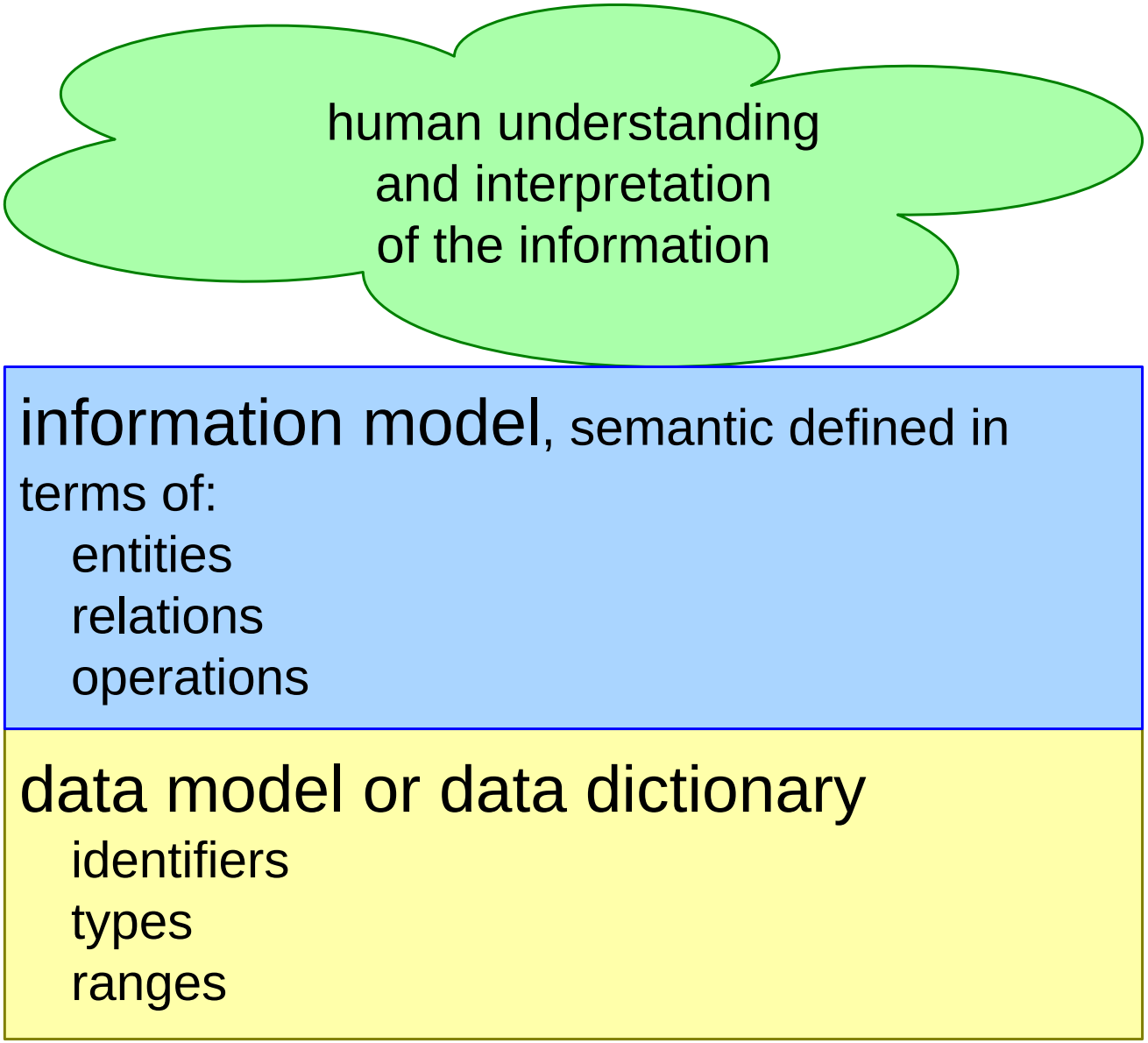
legend

+	present
o	optional
-	absent

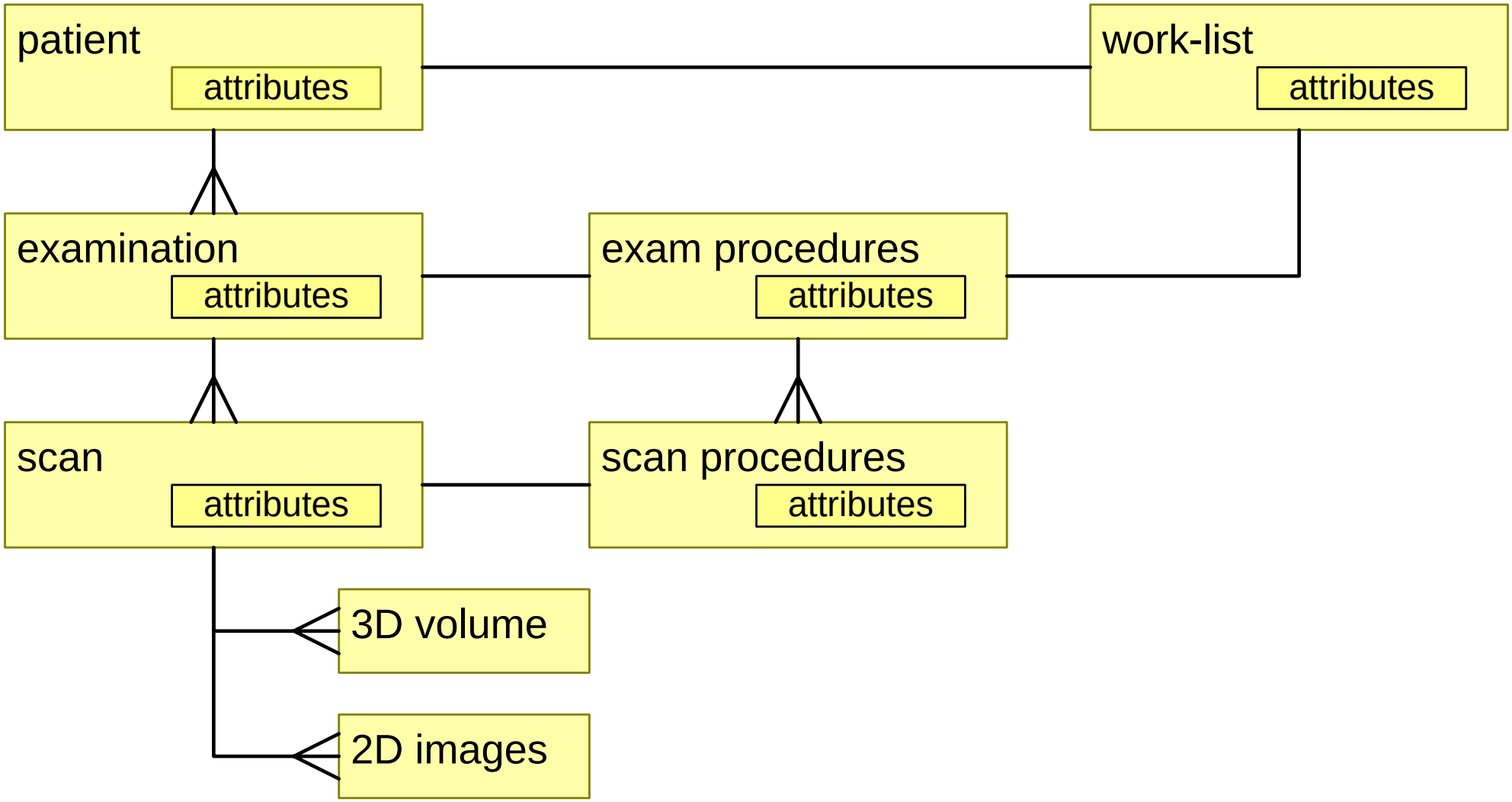
Relation between user interface and functional specification



Layering of information definitions



Example partial internal information model

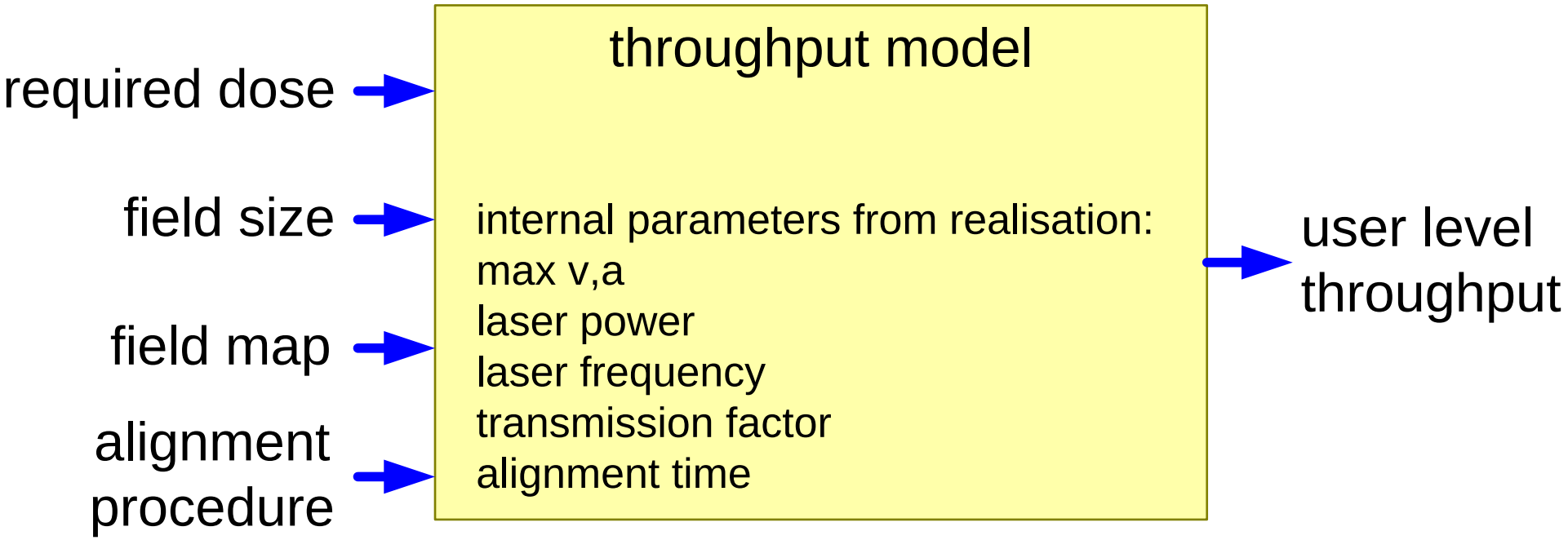


Small part of a datamodel

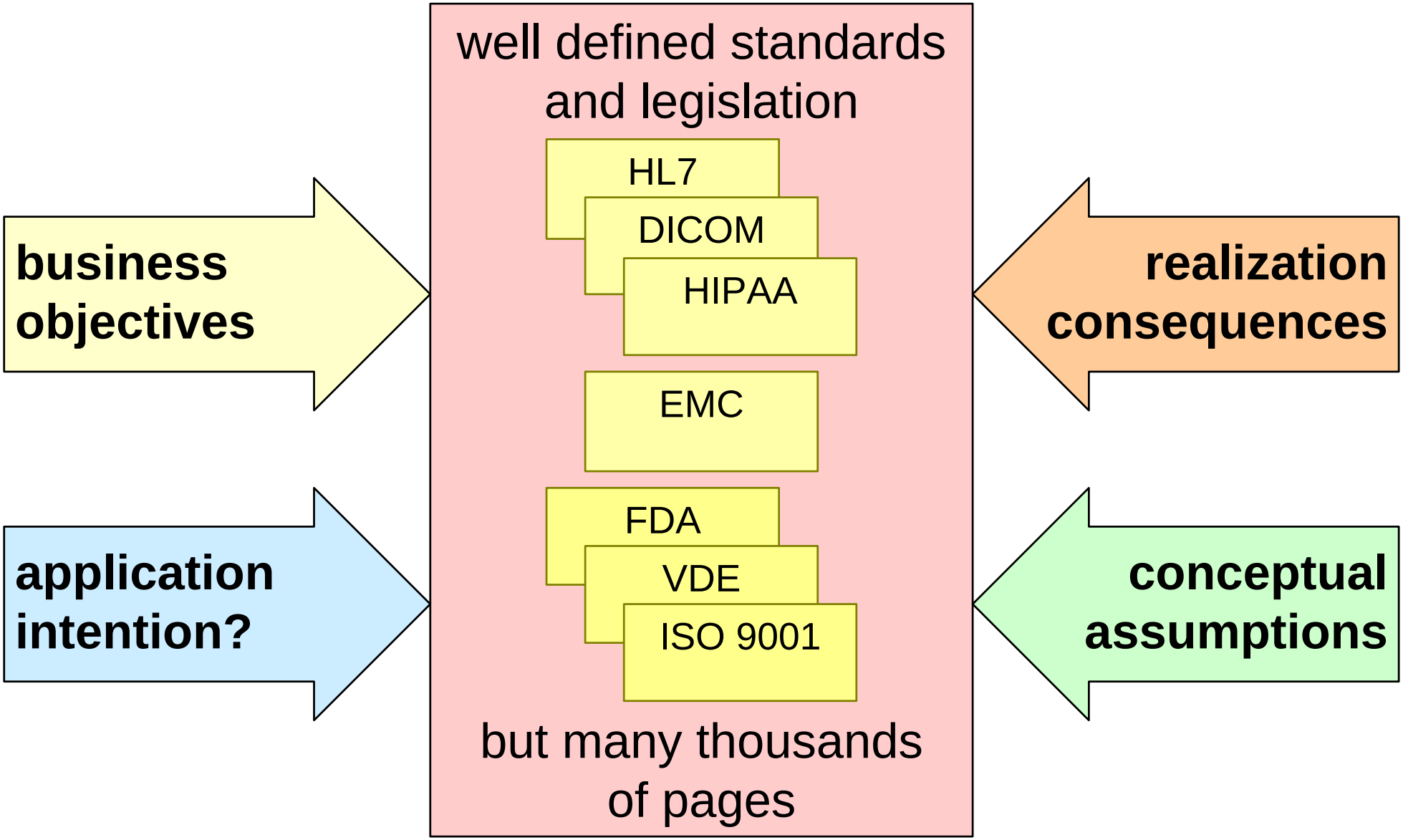
12 bit Image:
- nx: 16 bit unsigned integer
- ny: 16 bit unsigned integer
- pixels[nx][ny]: 16 bit unsigned integers [0..4095]

16 bit Image:
- nx: 16 bit unsigned integer
- ny: 16 bit unsigned integer
- pixels[nx][ny]: 16 bit unsigned integers

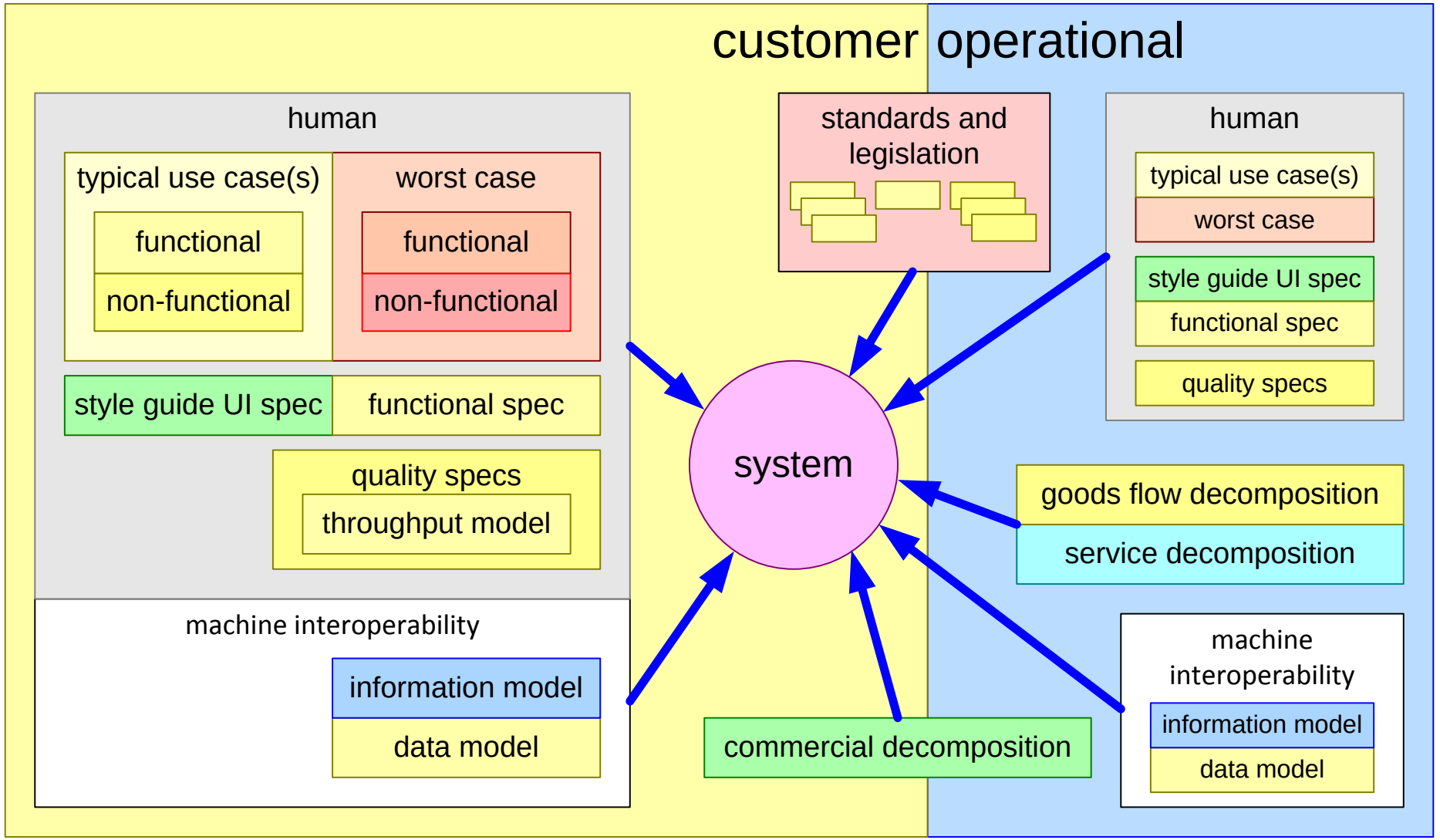
Example of performance modelling



The role of standards



Functional view summary



Functional view = What: externally observable

Exercise Functional View

- Make an overview of functions, performance figures, interfaces and optional features
- identify "most important" (related to CA-views)
- identify "most challenging" (related to CR-views)
- explain why "most important" or "most challenging"
- present in 5 minutes

Goals:

- create awareness of the breadth of the specifiation
- share the spec with the team
- create a "living" image of the Functional view

Exercise Functional View, second iteration

- Define a typical case, both functions and quantitative
- Create a single page product specification
- Define a worst case, suitable for design exploration and verification

Module Customer Side

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

Abstract

This module addresses The Customer Objectives and Application Views:

October 11, 2020
status: draft
version: 0

logo
TBD

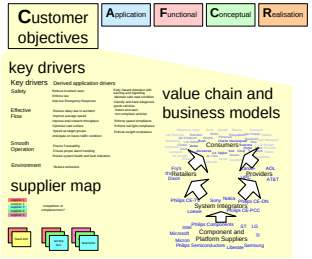
The customer objectives view

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

Abstract

The purpose of the customer objectives view is described. A number of methods or models is given to use in this view: customer key drivers to understand the essentials, value chains and business models to understand the position of the customer and a supplier map to understand the supply side of the customer.

October 11, 2020
status: preliminary draft
version: 0.3



Customer objectives overview

Customer objectives | **A**pplication | **F**unctional | **C**onceptual | **R**ealisation

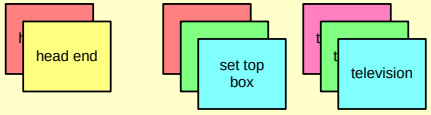
key drivers

Key drivers	Derived application drivers	
Safety	Reduce Accident rates	Early hazard detection with warning and signalling
	Enforce law	Maintain safe road condition
	Improve Emergency Response	Classify and track dangerous goods vehicles
Effective Flow	Reduce delay due to accident	Detect and warn non compliant vehicles
	Improve average speed	
	Improve total network throughput	Enforce speed compliance
	Optimise road surface	Enforce red light compliance
	Speed up target groups	Enforce weight compliance
	Anticipate on future traffic condition	
Smooth Operation	Ensure Traceability	
	Ensure proper alarm handling	
	Ensure system health and fault indication	
Environment	Reduce emissions	

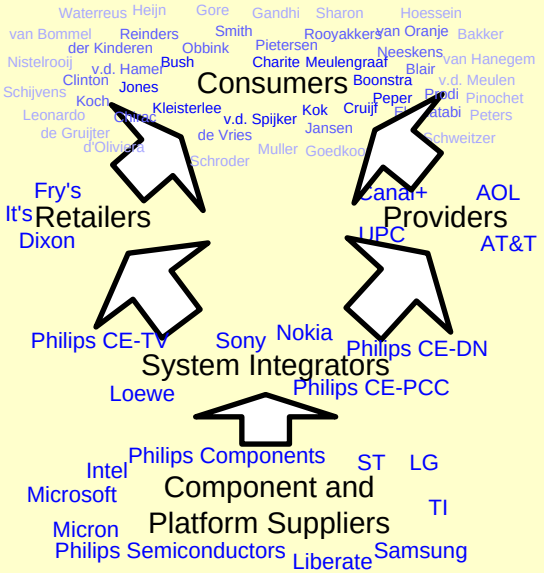
supplier map

supplier 1
supplier 2
supplier 3
supplier 4
supplier 5

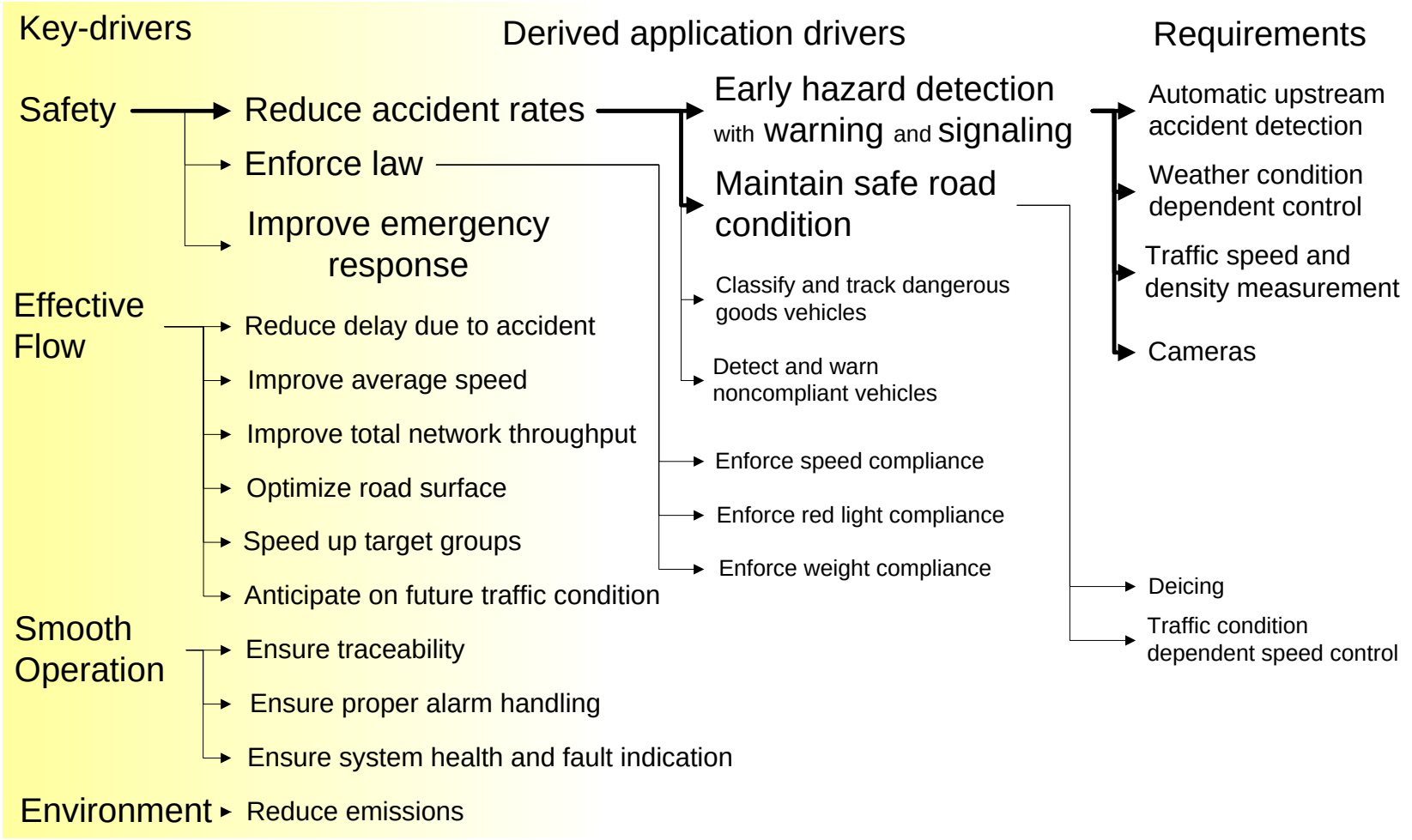
competitors or complementors?



value chain and business models



Example motorway management key drivers



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

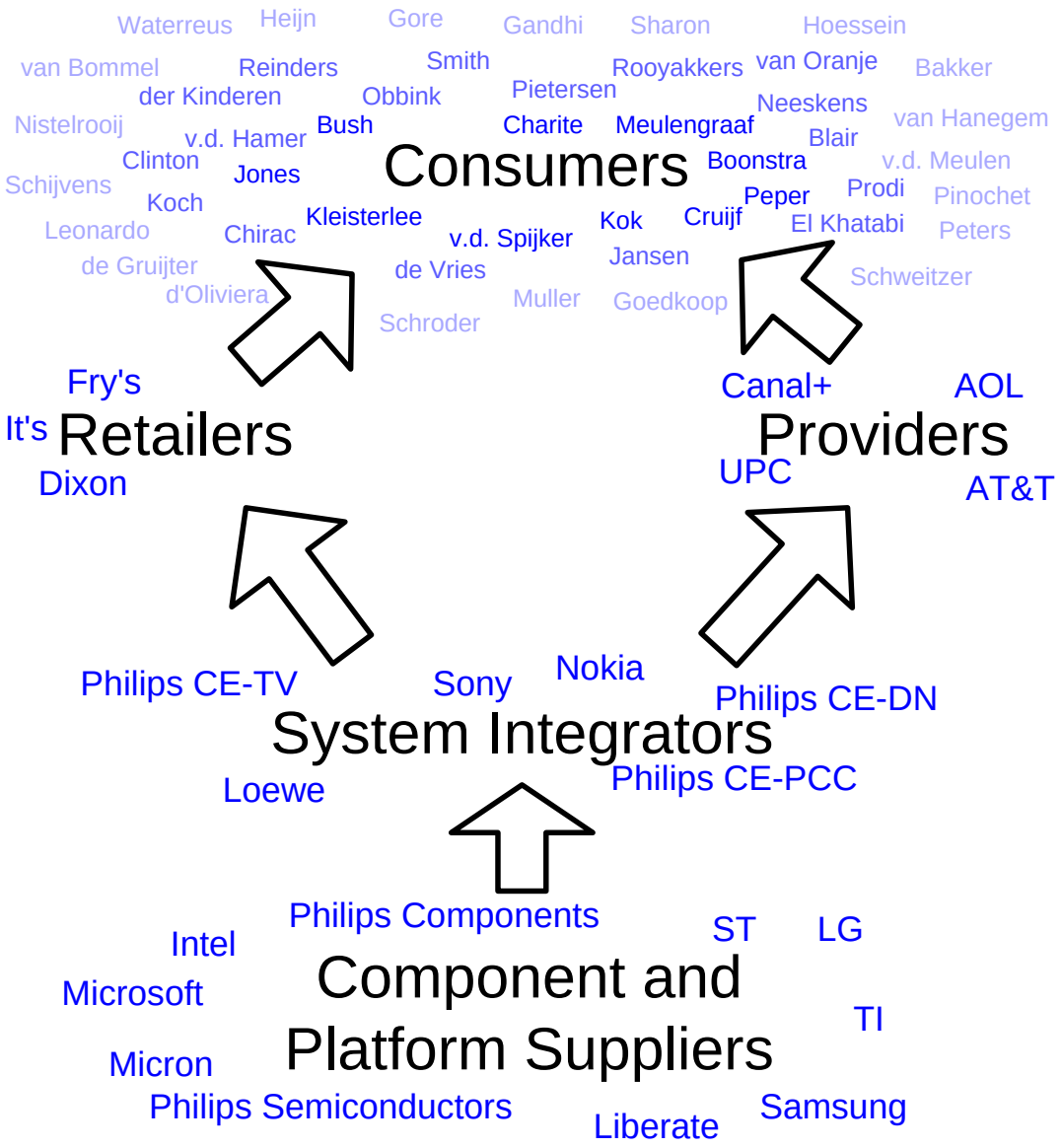
Submethod to Link Key Drivers to Requirements

• 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

Key Driver Recommendations

• 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

Example value chain

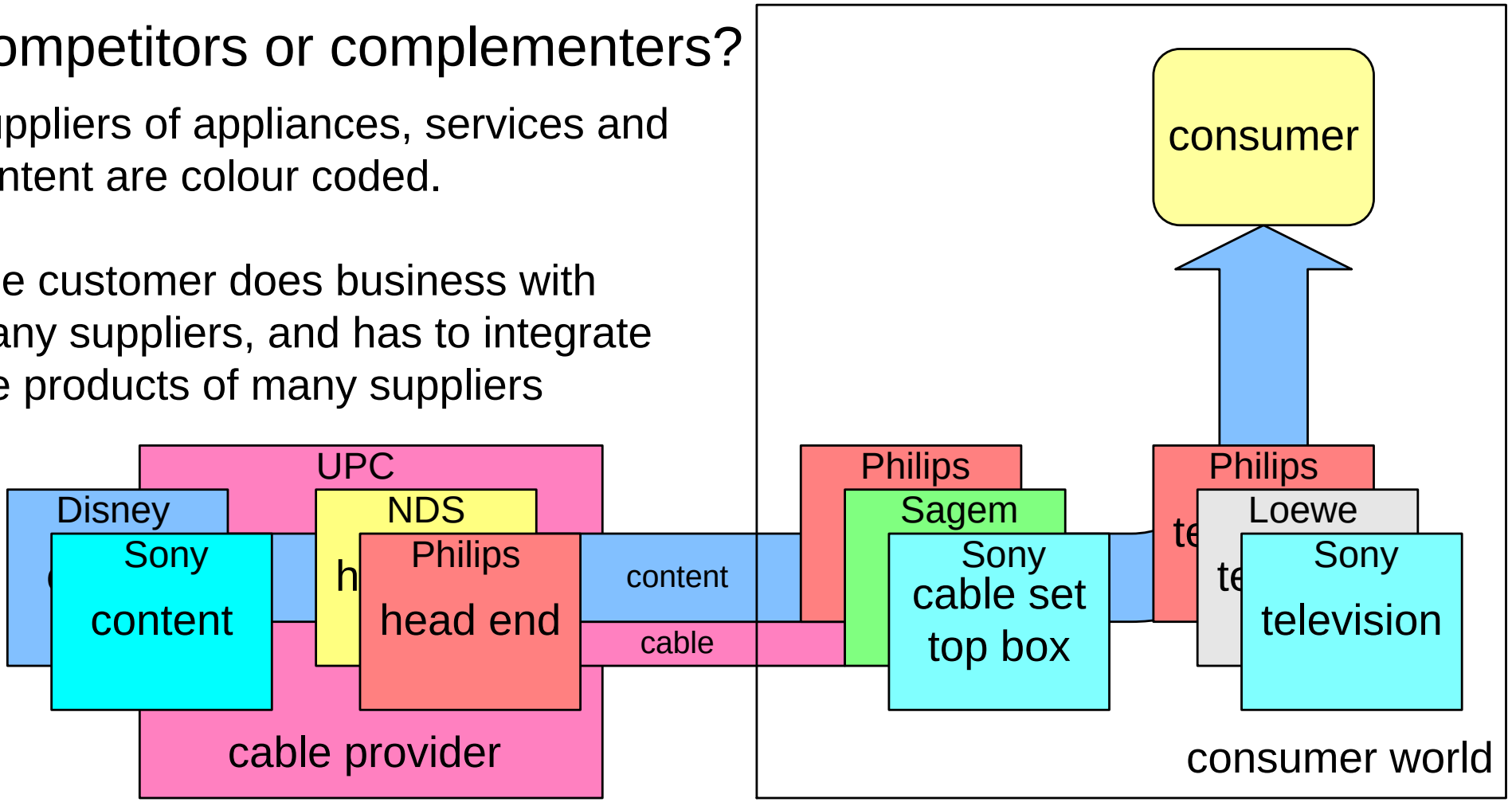


Example of simple supplier map

competitors or complementers?

Suppliers of appliances, services and content are colour coded.

The customer does business with many suppliers, and has to integrate the products of many suppliers

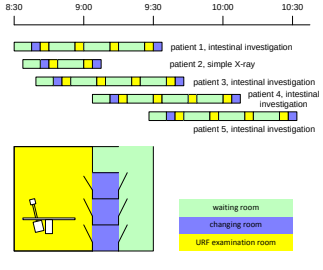


The application view

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

Abstract

The purpose of the application view is described. A number of methods or models is given to use in this view: stakeholder and concerns, context diagram, static entity relationship models and dynamic flow models.

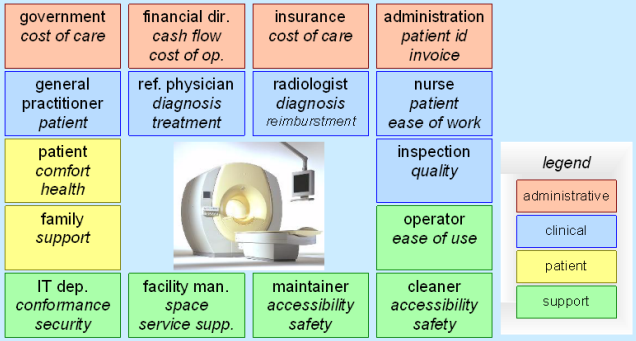


October 11, 2020
status: preliminary draft
version: 0.2

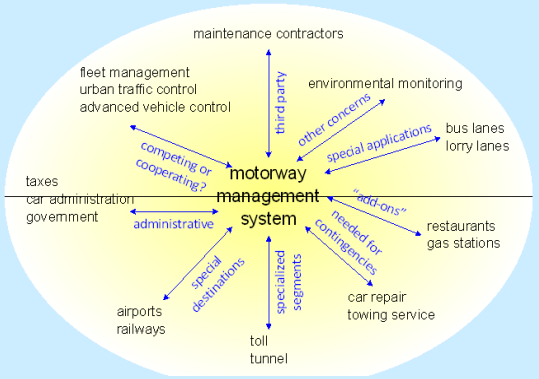
Application view overview

Customer objectives | **Application** | Functional | Conceptual | Realisation

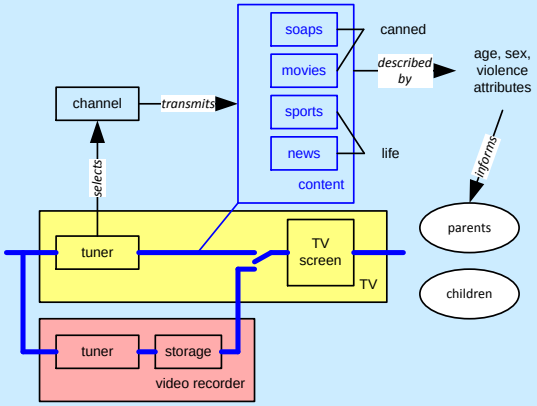
stakeholders and concerns



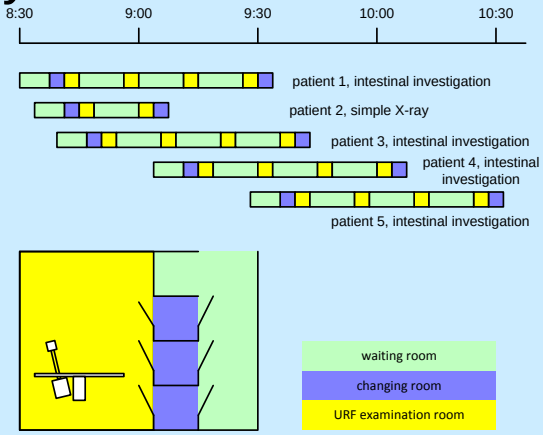
context diagrams



entity relationship models



dynamic models



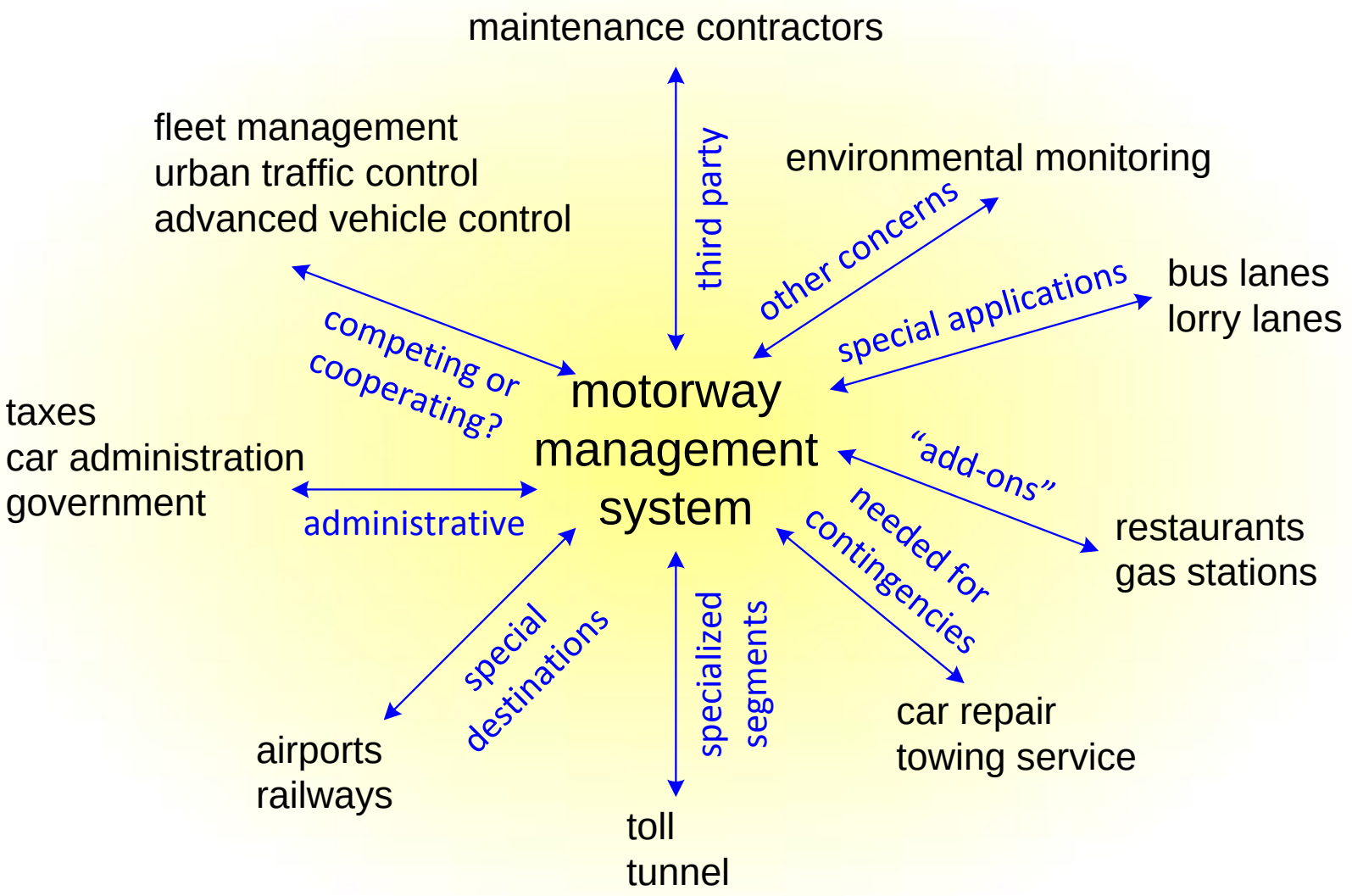
Stakeholders and concerns MRI scanner

government *cost of care*	financial dir. *cash flow cost of op.*	insurance *cost of care*	administration *patient id invoice*
general practitioner *patient*	ref. physician *diagnosis treatment*	radiologist *diagnosis reimburstment*	nurse *patient ease of work*
patient *comfort health*			inspection *quality*
family *support*			operator *ease of use*
IT dep. *conformance security*	facility man. *space service supp.*	maintainer *accessibility safety*	cleaner *accessibility safety*

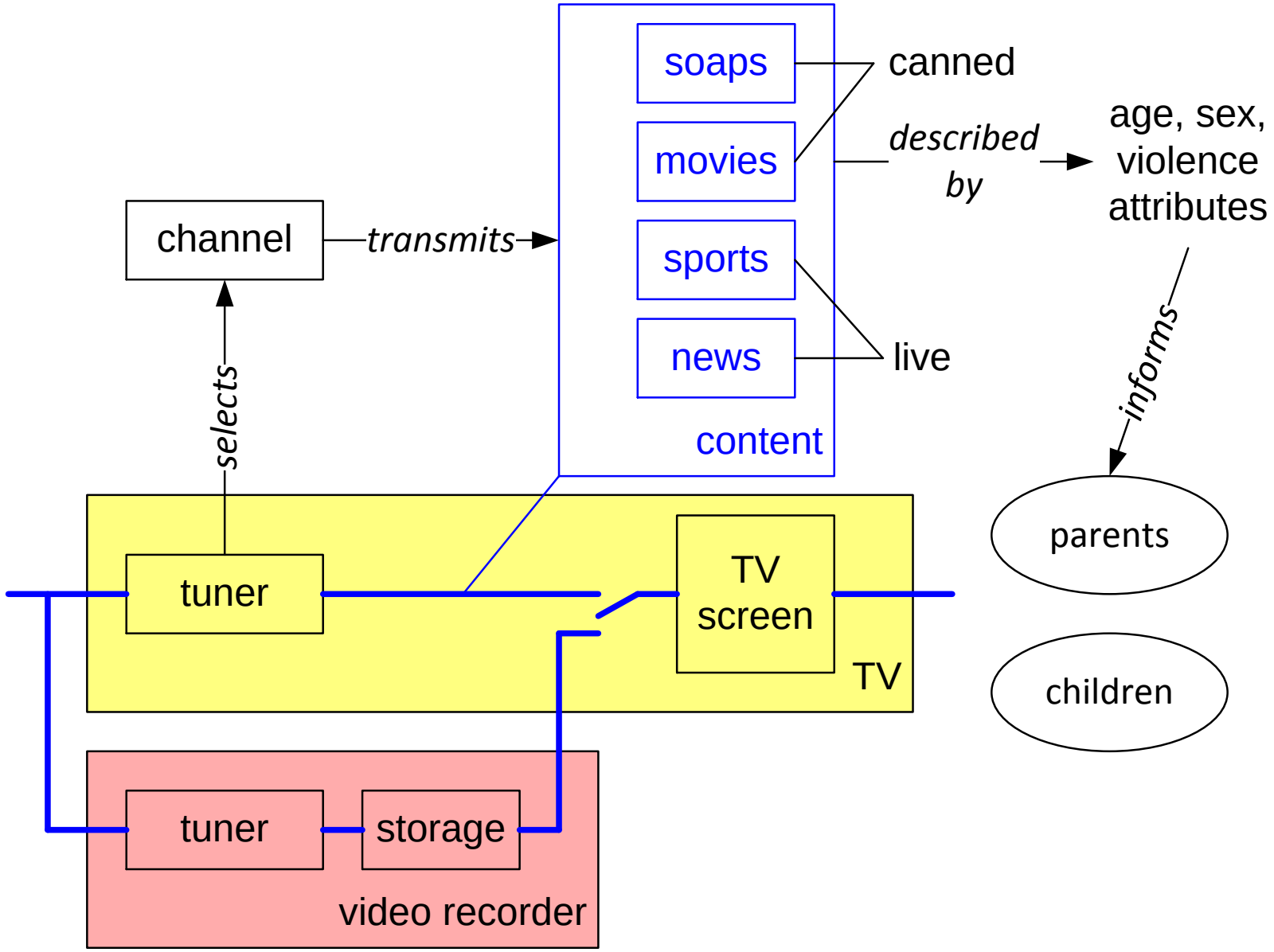
legend

- administrative
- clinical
- patient
- support

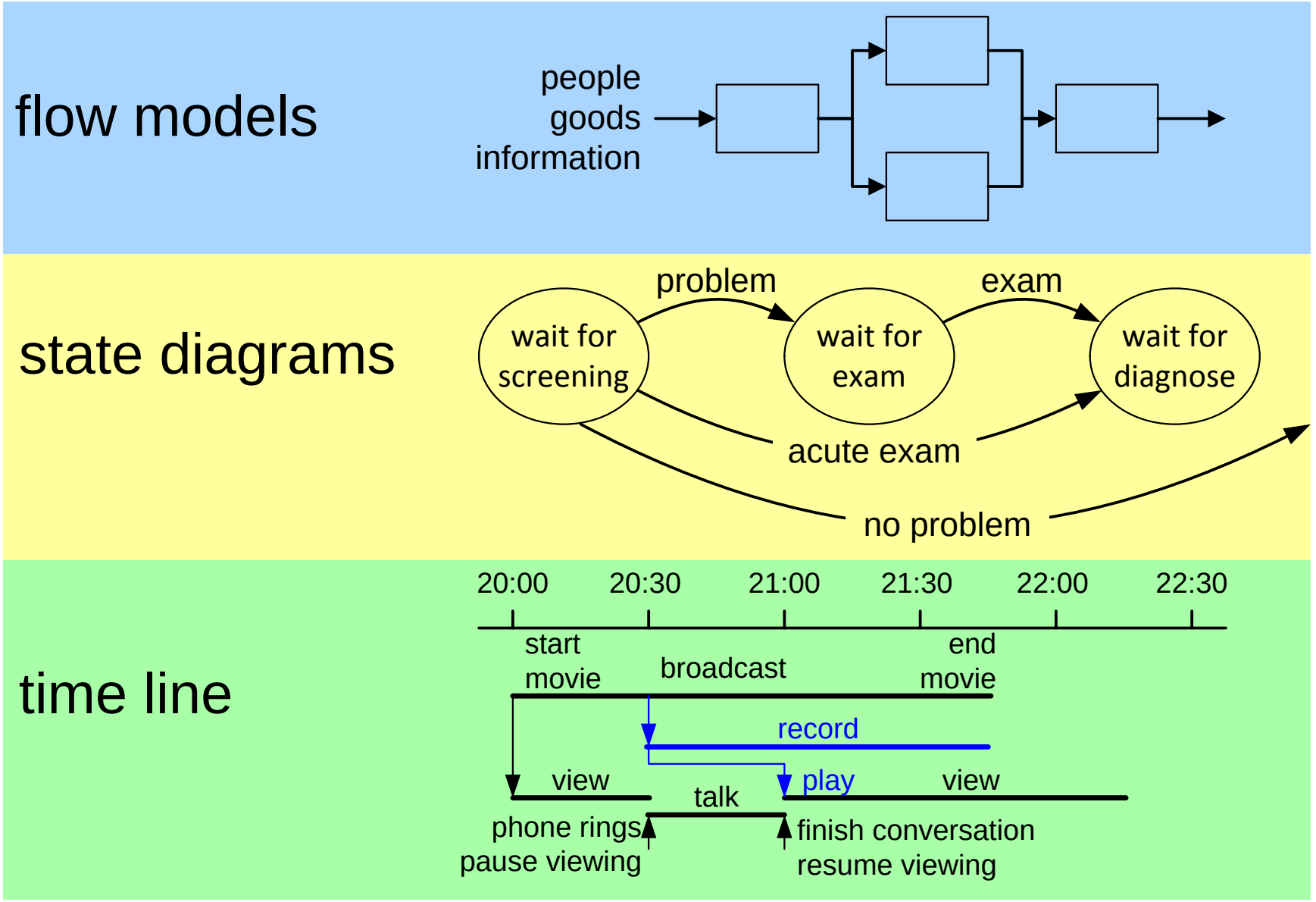
Context of motorway management system



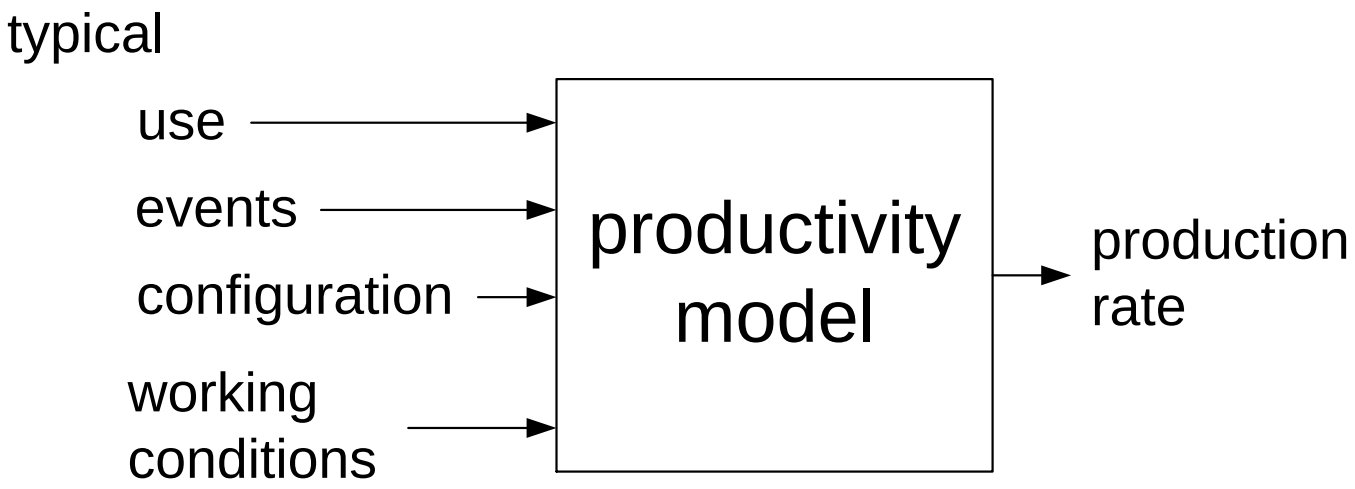
Example of simple TV application model



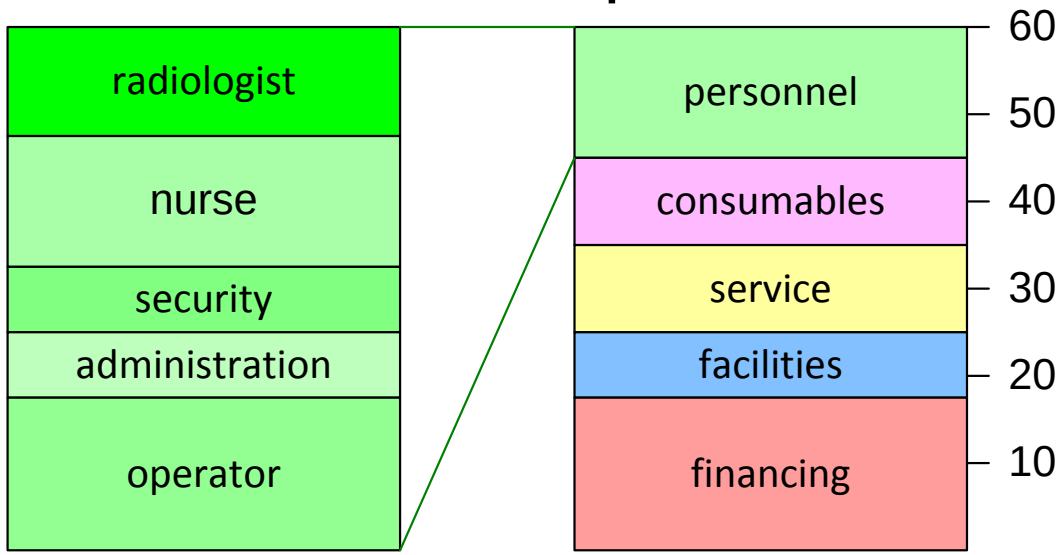
Examples of dynamic models



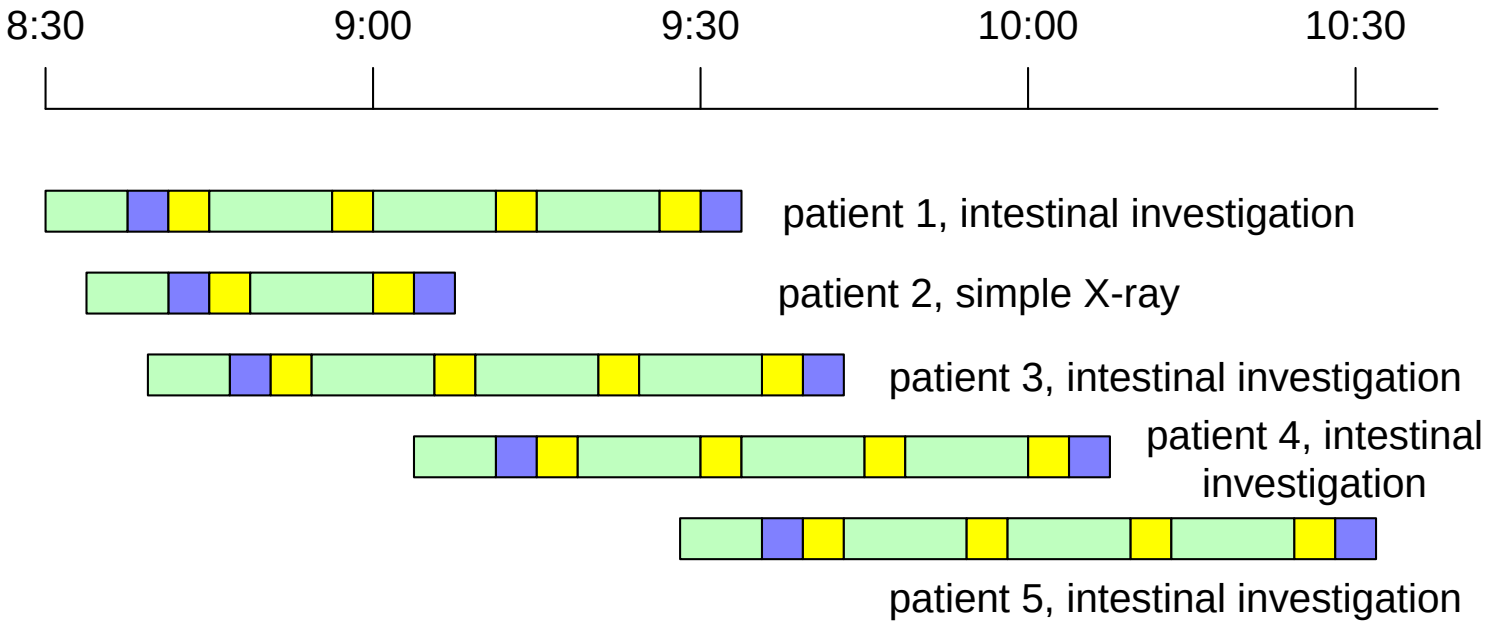
Productivity and Cost models

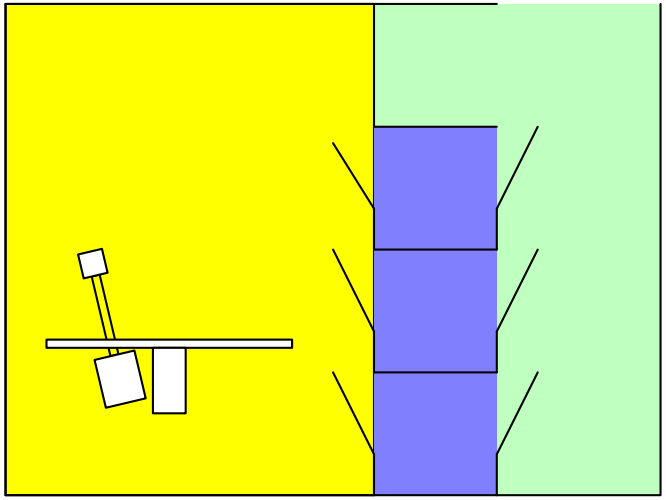


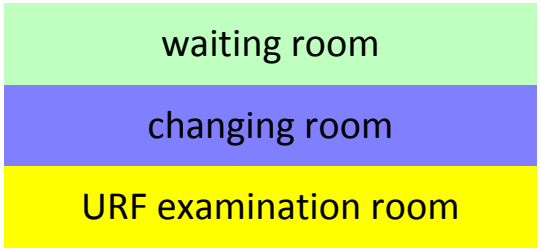
Cost Of Ownership model



Dynamics of an URF examination room







Exercise Customer Side

- Determine stakeholders, key drivers and context of the product.
- Translate these drivers into application drivers and link them to the requirements.

Exercise Customer Side, second iteration

- Create a (max) 8 sheet presentation describing the customer objectives and application.

Module Design Side

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

Abstract

This module addresses the Conceptual and Realization Views.

October 11, 2020
status: draft
version: 0

logo
TBD

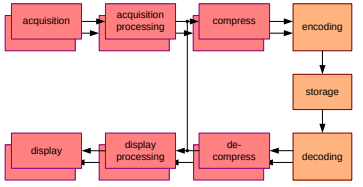
The conceptual view

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

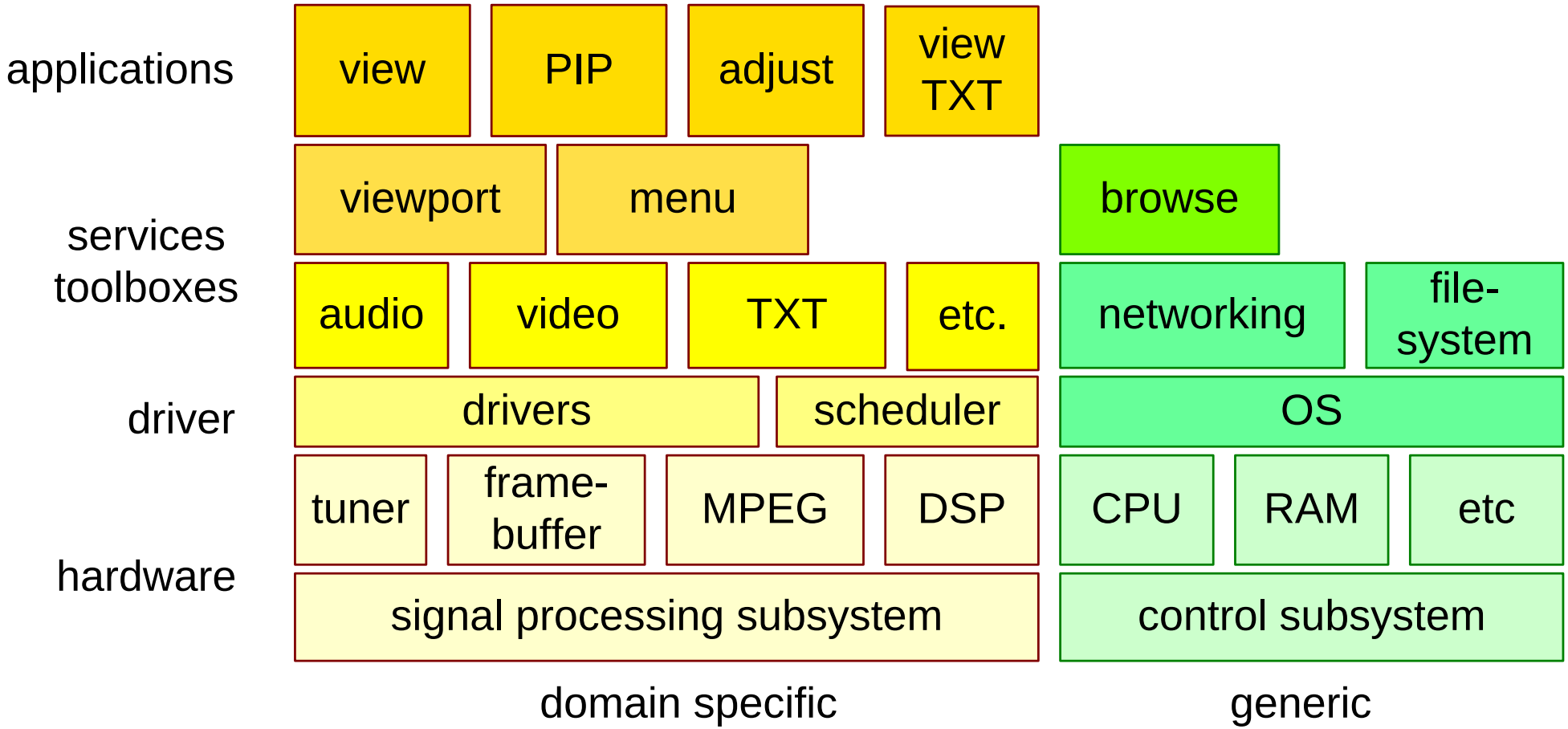
Abstract

The purpose of the conceptual view is described. A number of methods or models is given to use in this view: construction decomposition, functional decomposition, class or object decomposition, other decompositions (power, resources, recycling, maintenance, project management, cost, ...), and related models (performance, behavior, cost, ...); allocation, dependency structure; identify the infrastructure (factoring out shareable implementations), classify the technology in *core*, *key* and *base* technology; integrating concepts (start up, shutdown, safety, exception handling, persistency, resource management,...).

October 11, 2020
status: preliminary draft
version: 0.7



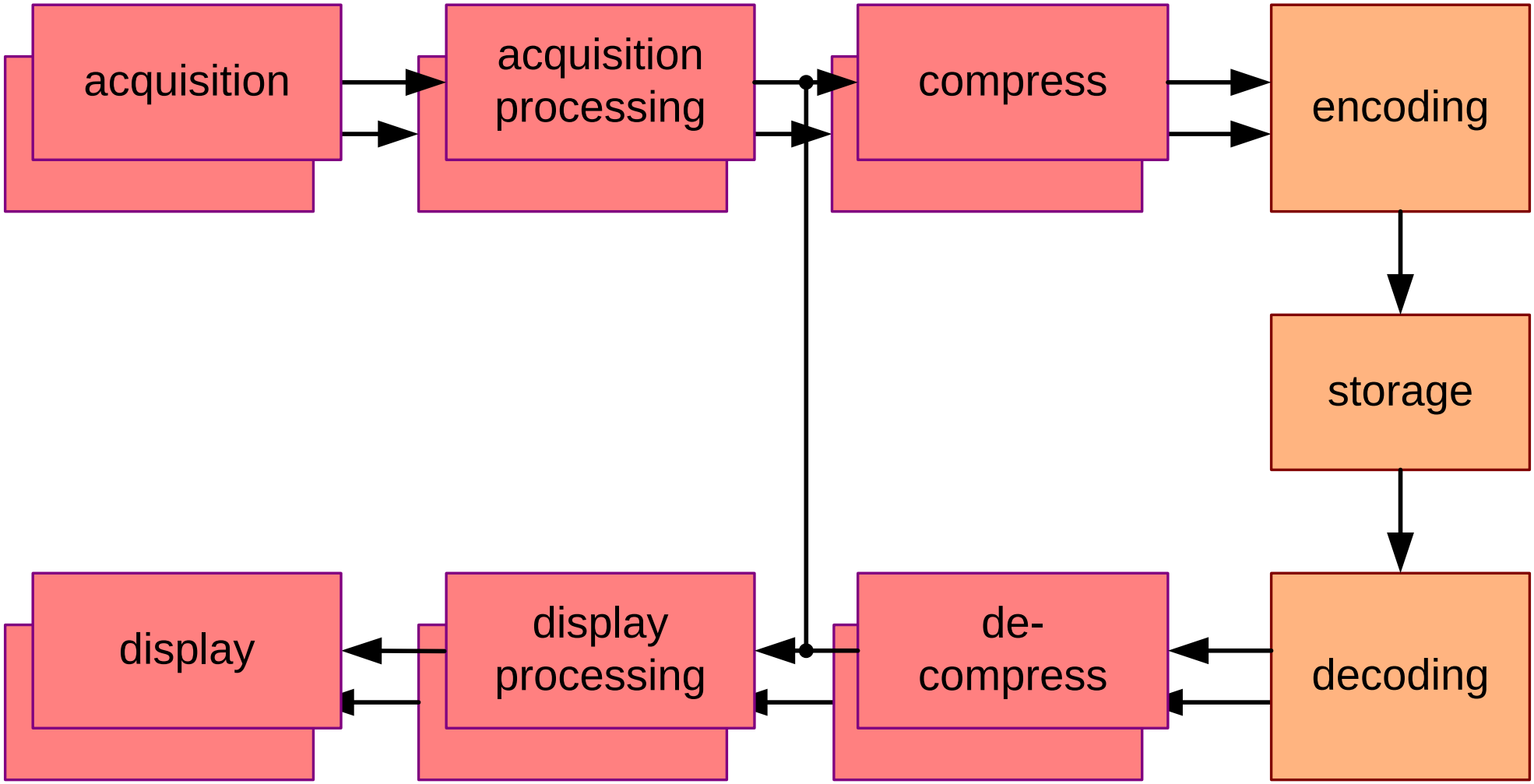
Example construction decomposition simple TV



Characterization of the construction decomposition

management of design	SW example	HW example
unit of creation storage update	file	PCB IP cells IP core
unit of aggregation for organisation test release	package module	box IP core IC

Example functional decomposition camera type device



Characterization of the functional decomposition

How;
what is the flow of internal activities
to realise external functionality?

some keywords:

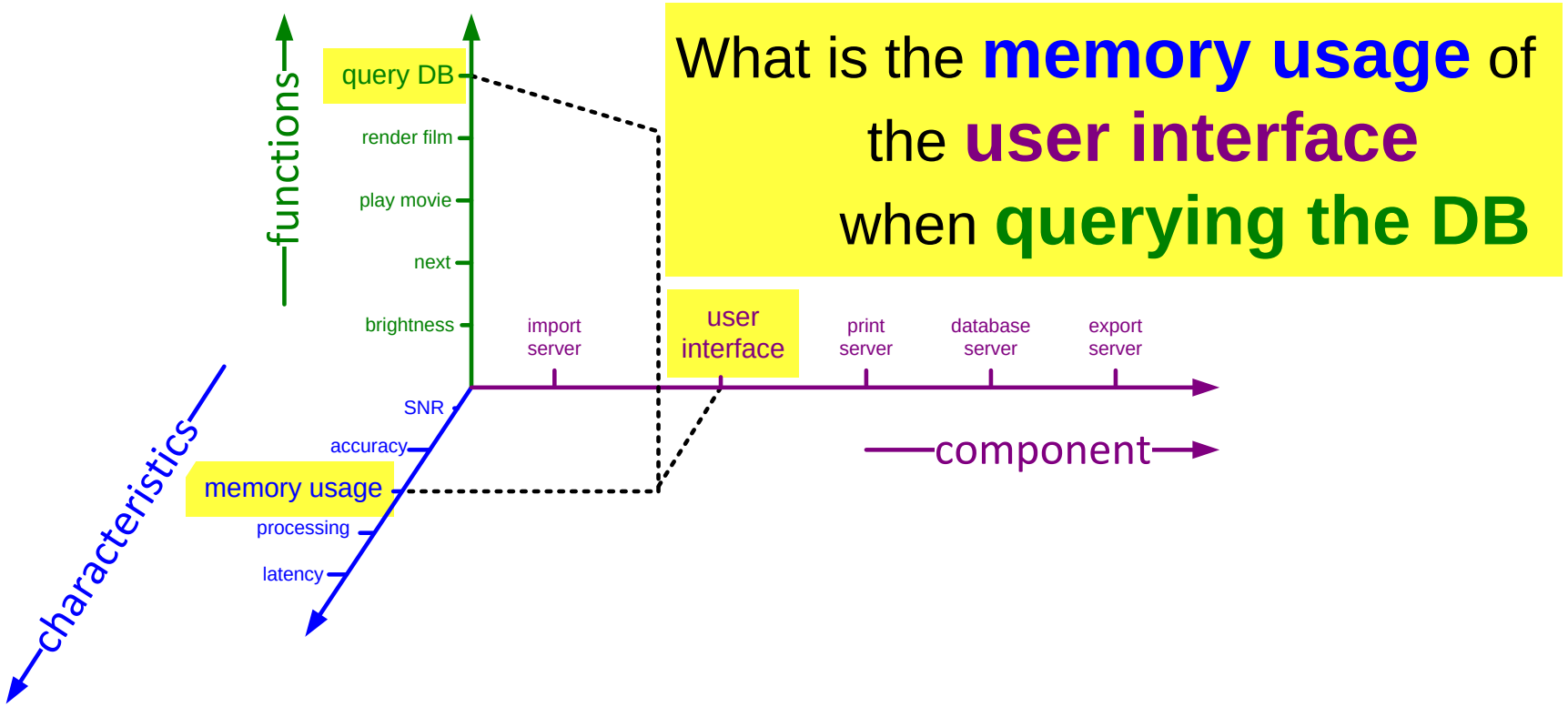
activities
transformation
input output

data flow
control flow

multiple functional decompositions are possible and valuable!

Question generator for multiple decompositions

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



Selection factors to improve the question generator

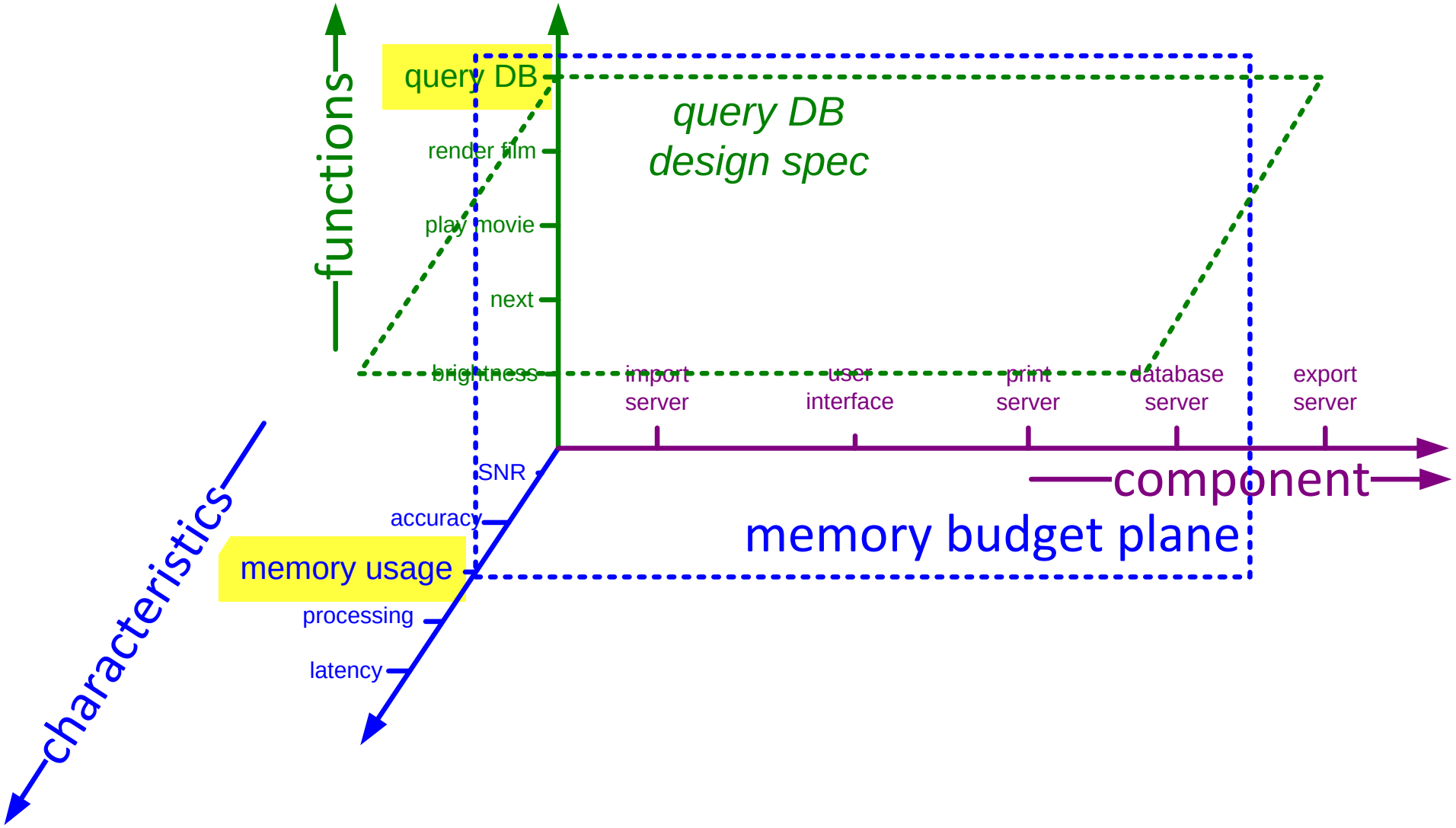
Critical for system performance

Risk planning wise

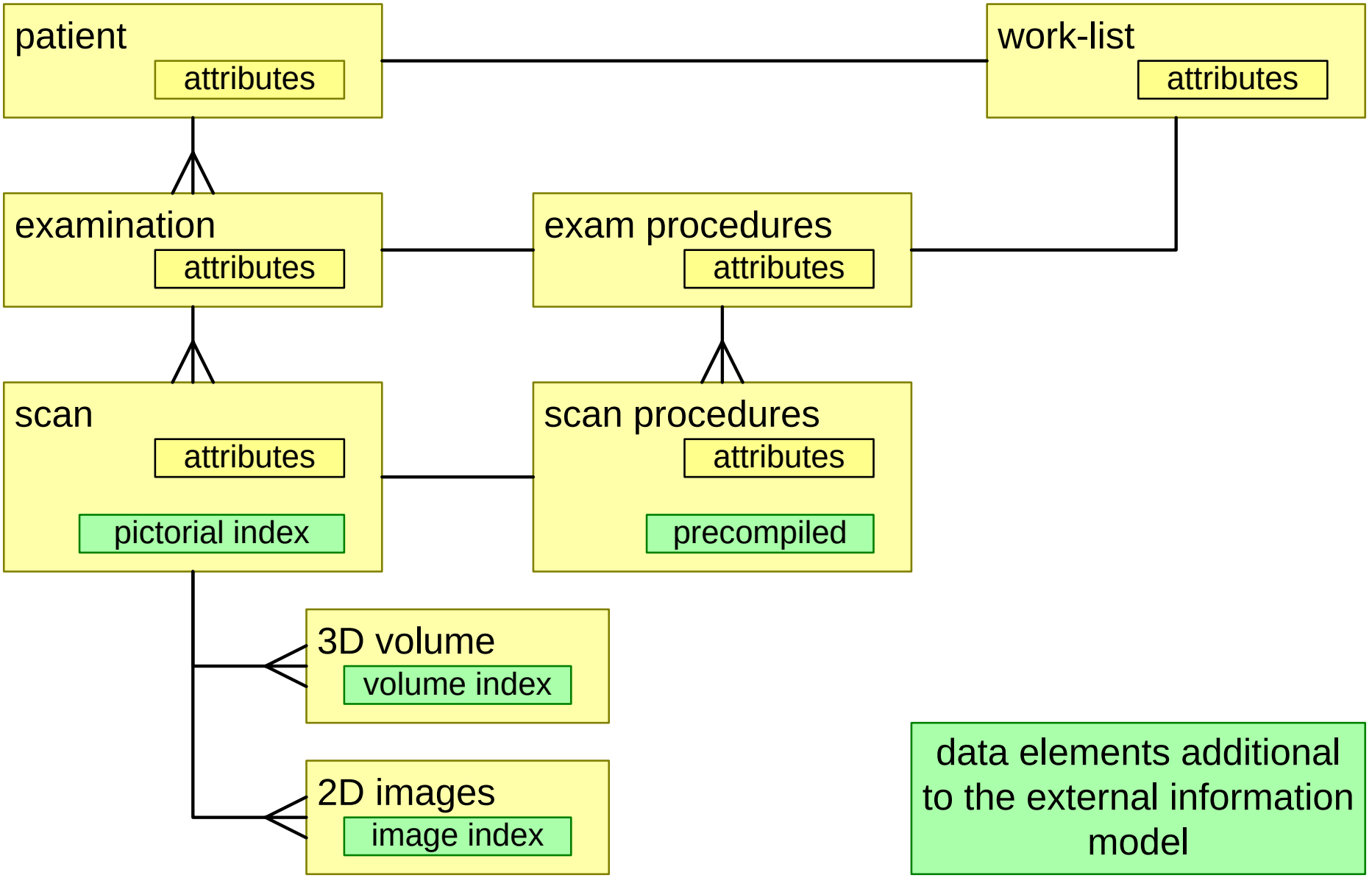
Least robust part of the design

Suspect part of the design
- experience based
- person based

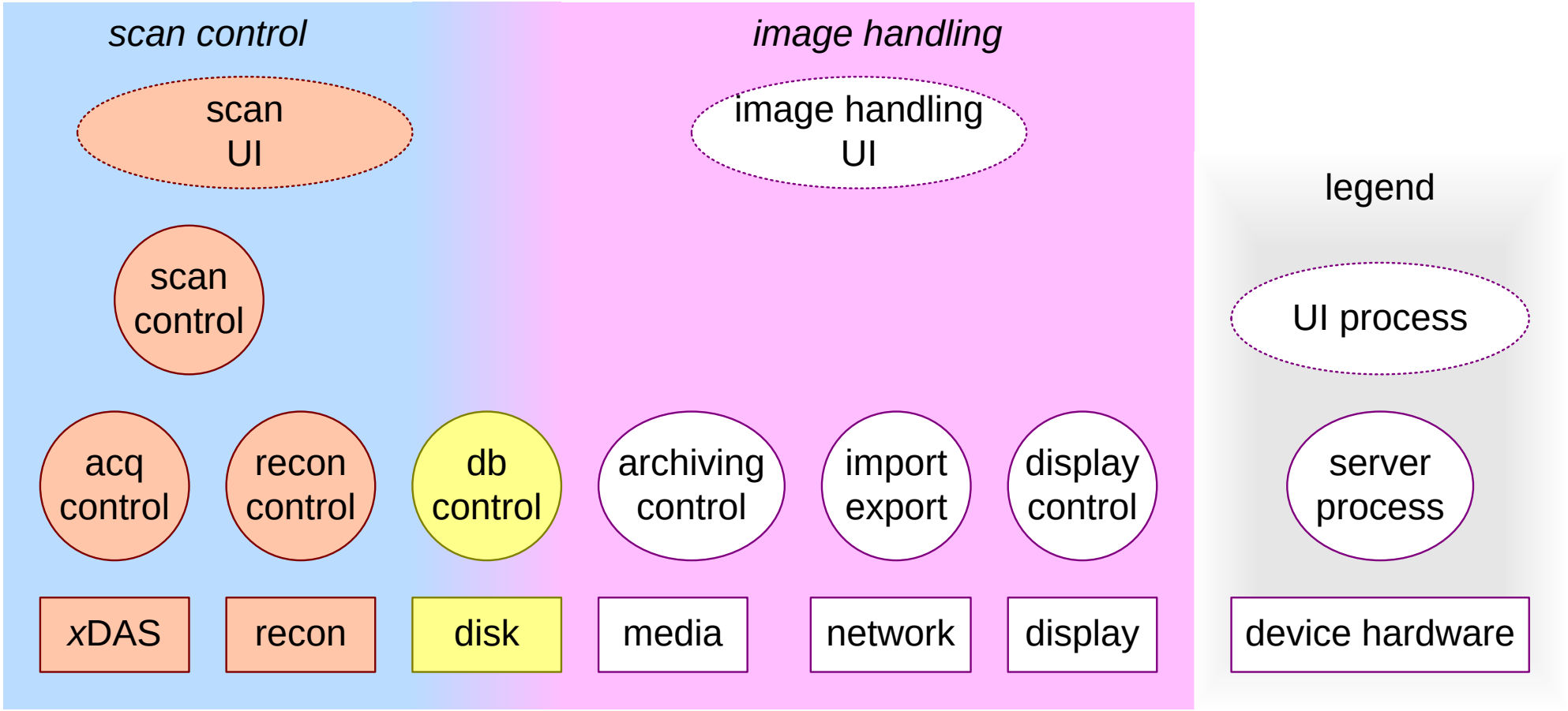
Addressing planes or lines



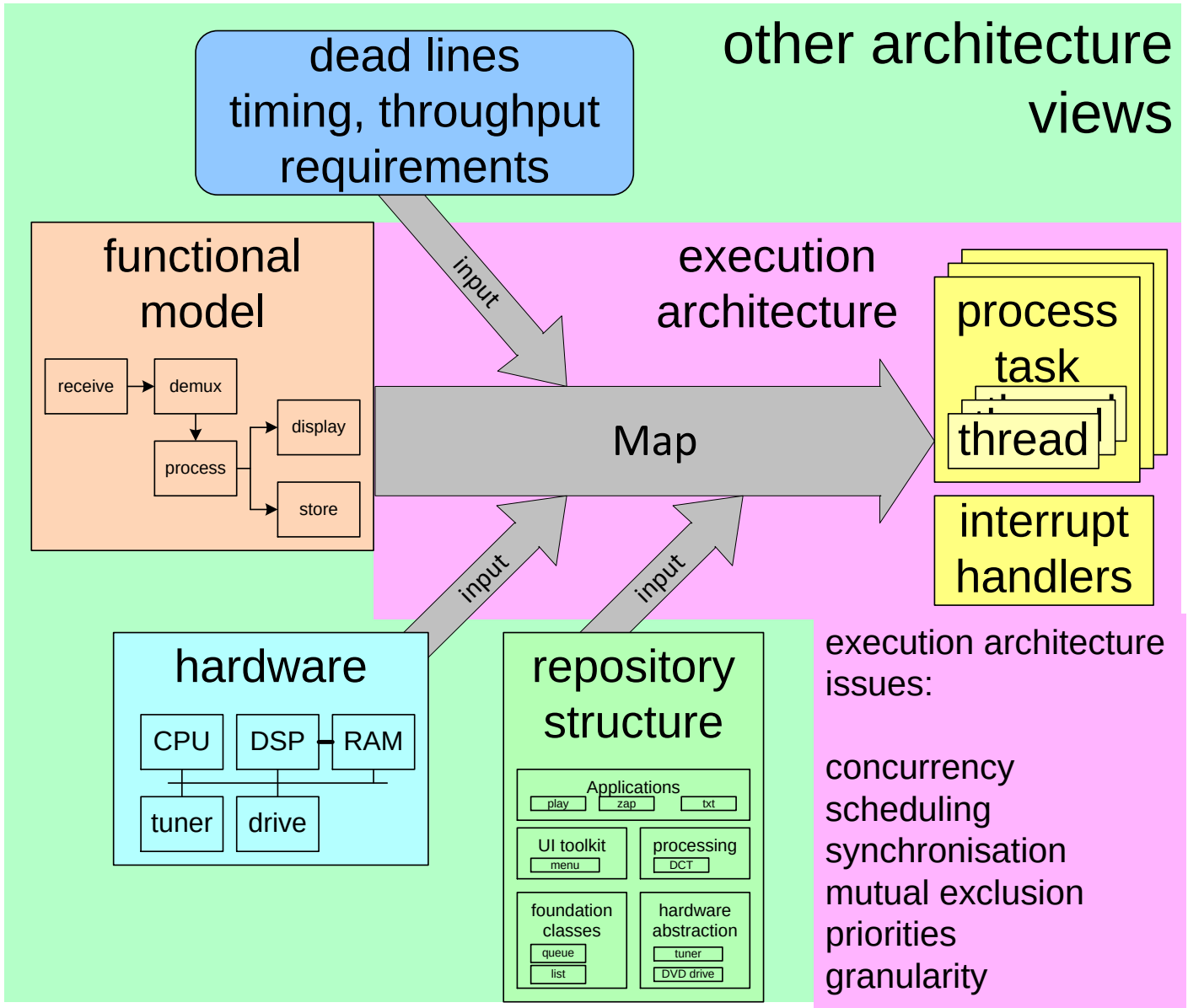
Example partial internal information model



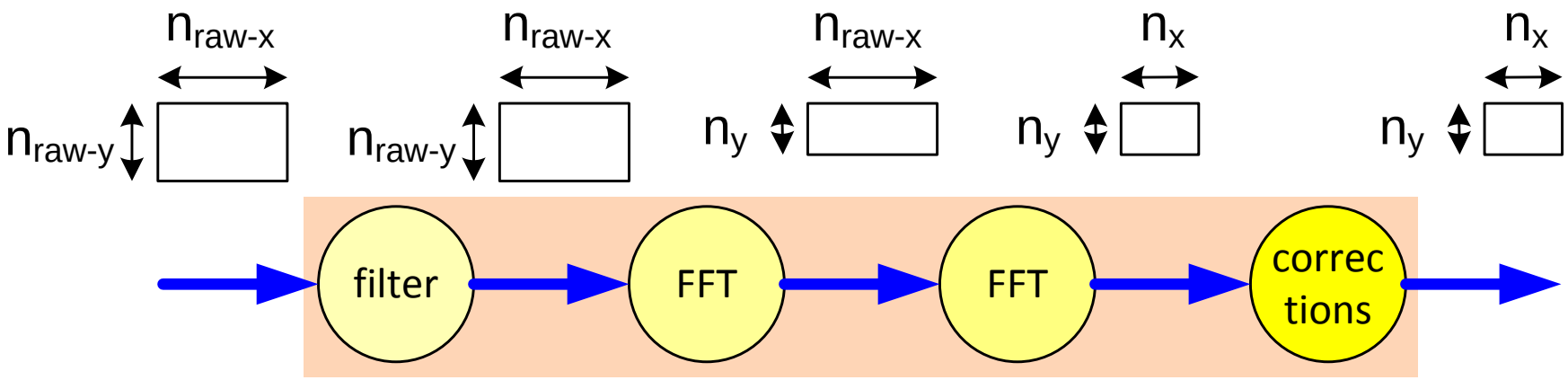
Example process decomposition



Execution architecture



Performance Model



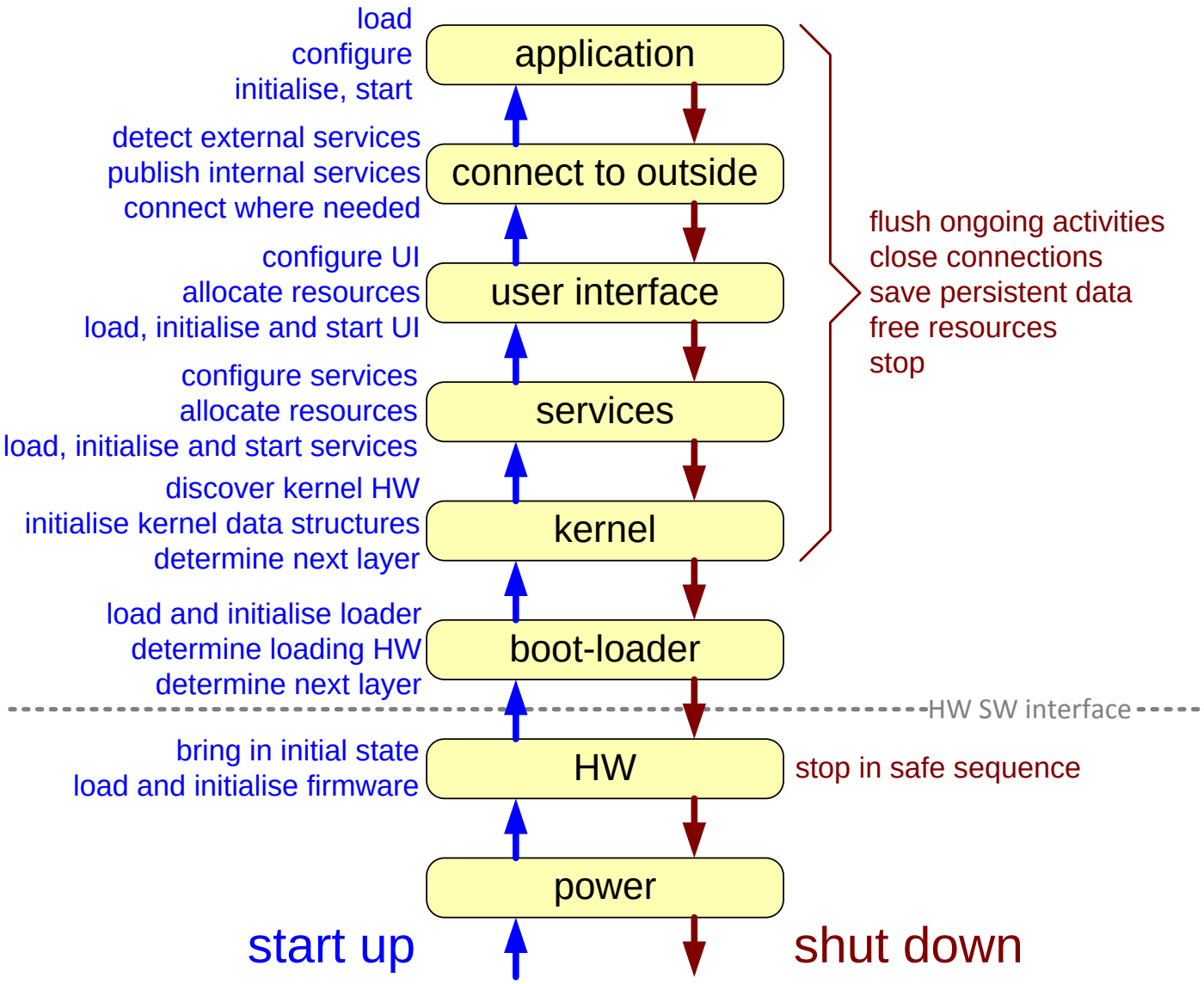
$$t_{recon} = t_{filter}(n_{raw\text{-}x}, n_{raw\text{-}y}) + n_{raw\text{-}x} * (t_{fft}(n_{raw\text{-}y}) + t_{col\text{-}overhead}) + n_y * (t_{fft}(n_{raw\text{-}x}) + t_{row\text{-}overhead}) + t_{corrections}(n_x, n_y) + t_{control\text{-}overhead}$$

$$t_{fft}(n) = c_{fft} * n * \log(n)$$

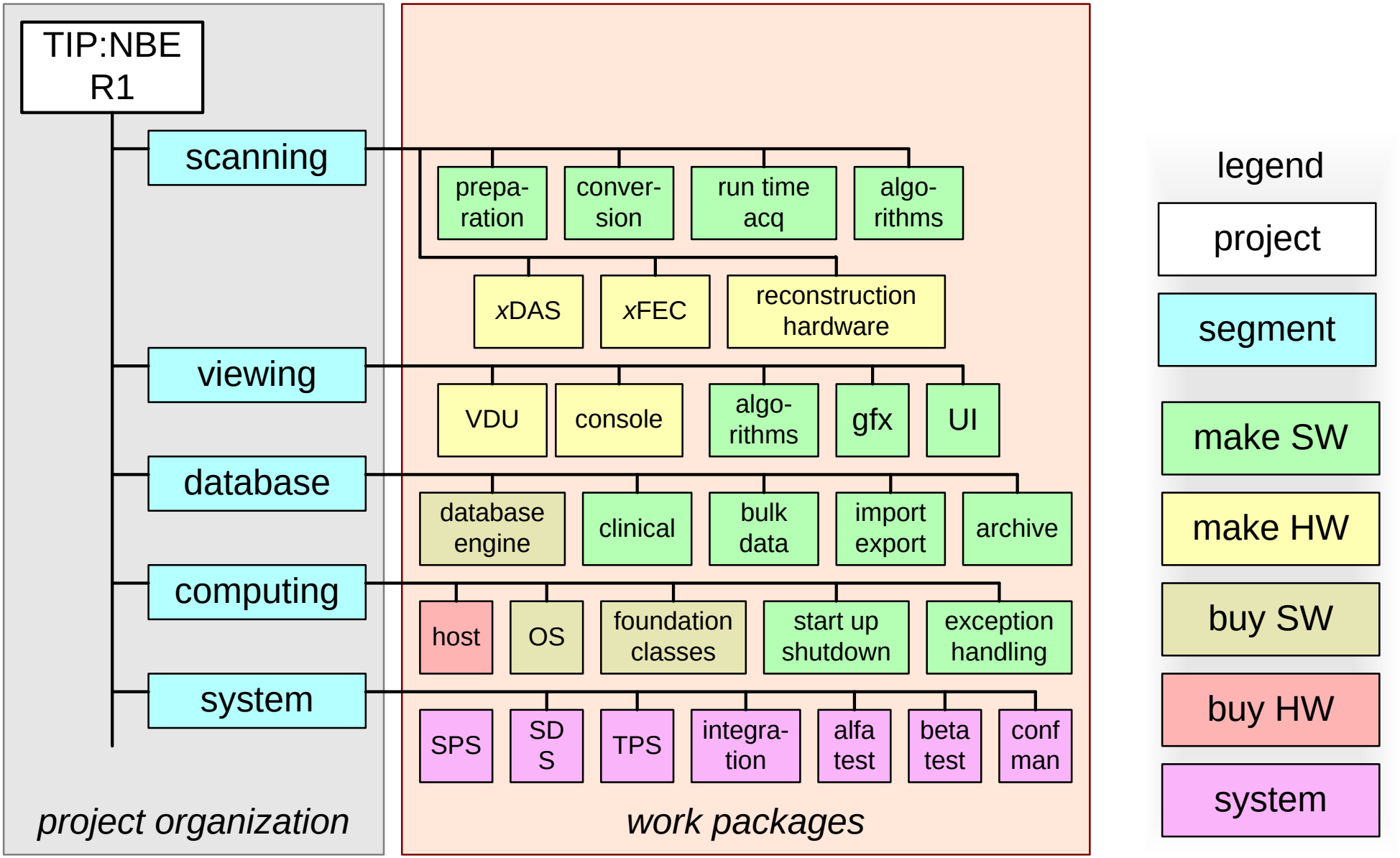
Safety, Reliability and Security concepts

- containment (limit failure consequences to well defined scope)
- graceful degradation (system parts not affected by failure continue operation)
- dead man switch (human activity required for operation)
- interlock (operation only if hardware conditions are fulfilled)
- detection and tracing of failures
- black box (log) for post mortem analysis
- redundancy

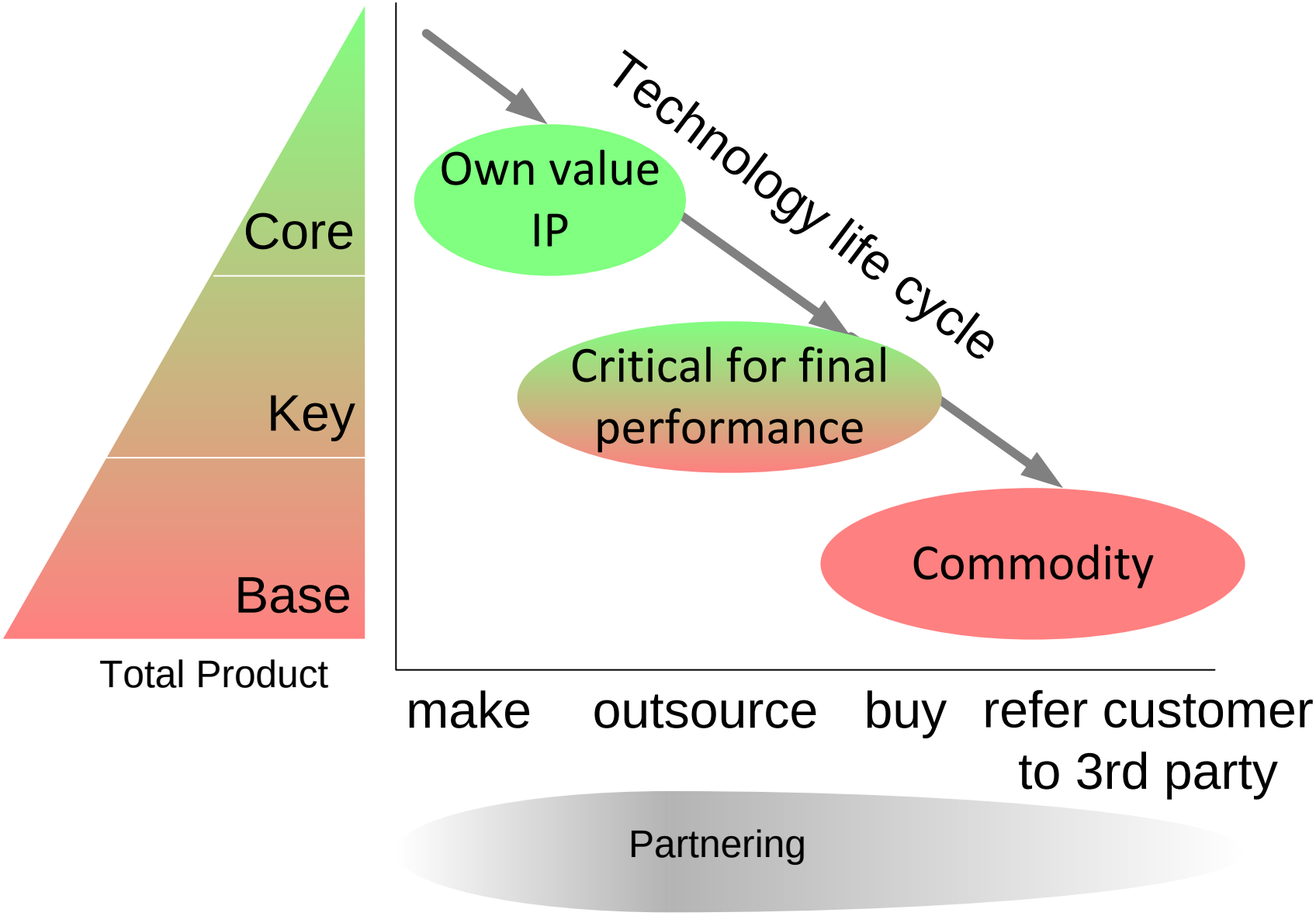
Simplified start up sequence



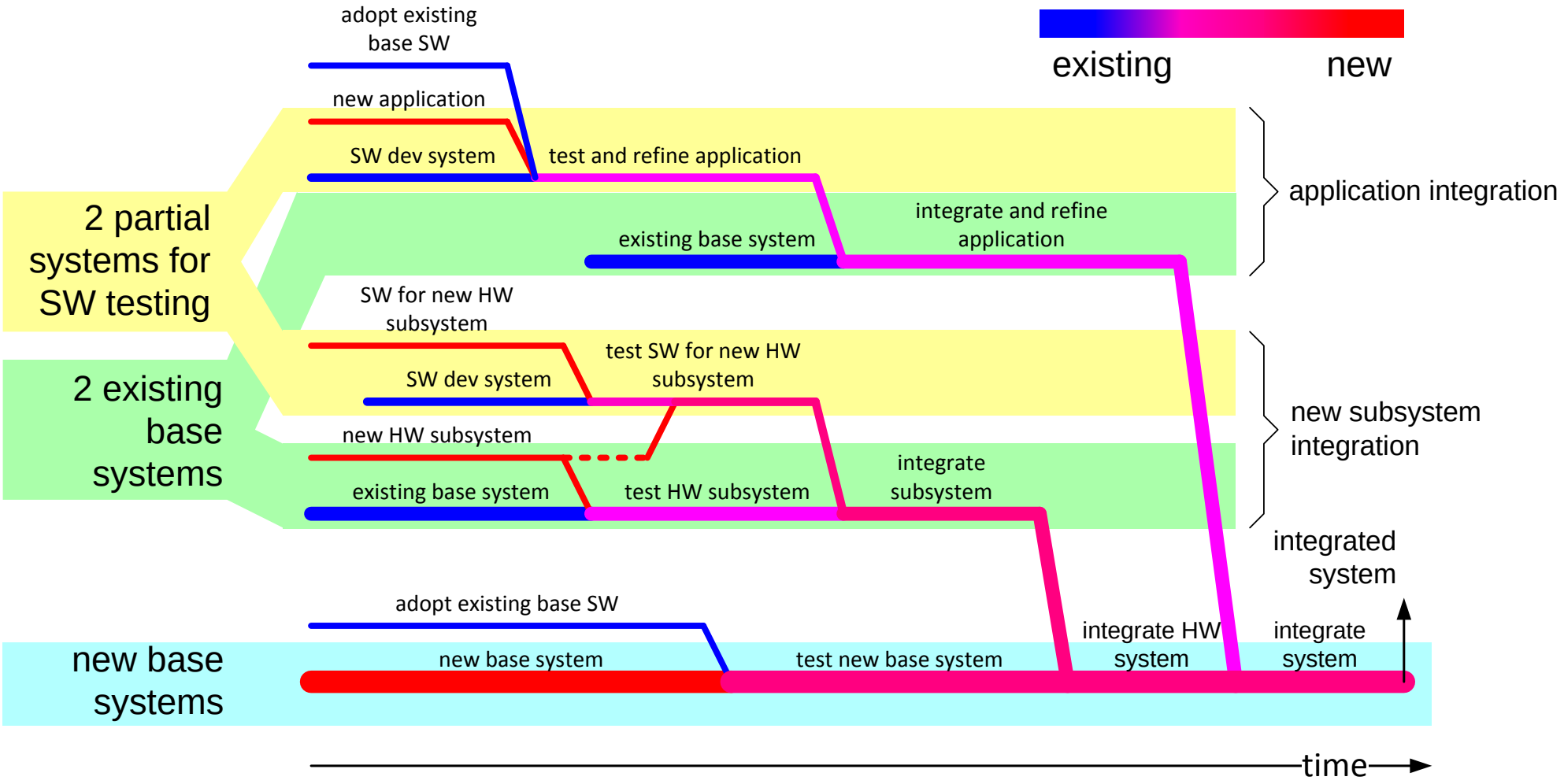
Example work breakdown



Core, Key or Base technology



Example integration plan



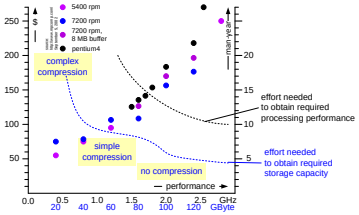
The realization view

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

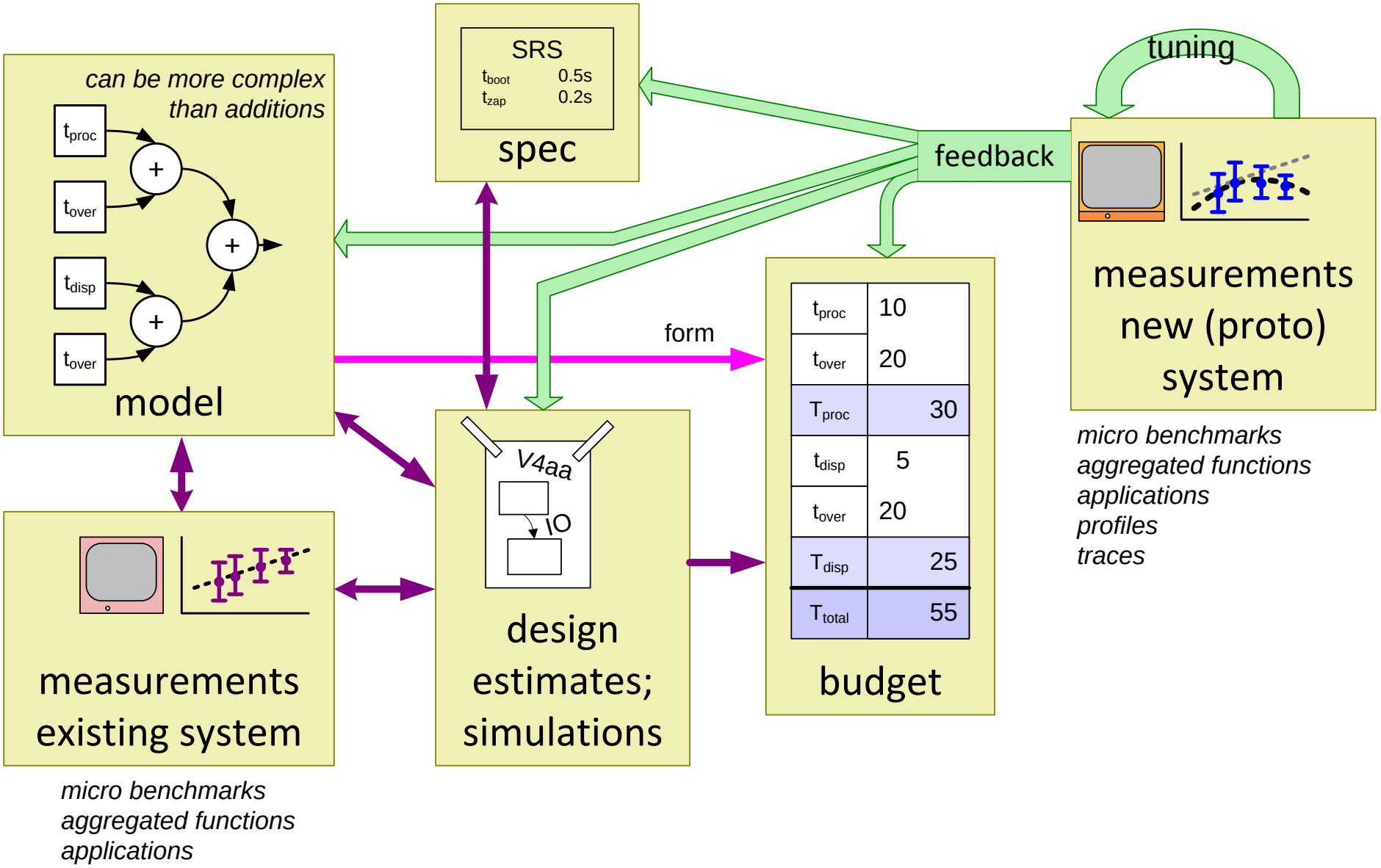
Abstract

The realization view looks at the actual technologies used and the actual implementation. Methods used here are logarithmic views, micro-benchmarks and budgets.
Analysis methods with respect to safety, reliability and security provide a link back to the functional and conceptual views.

October 11, 2020
status: preliminary draft
version: 0.1



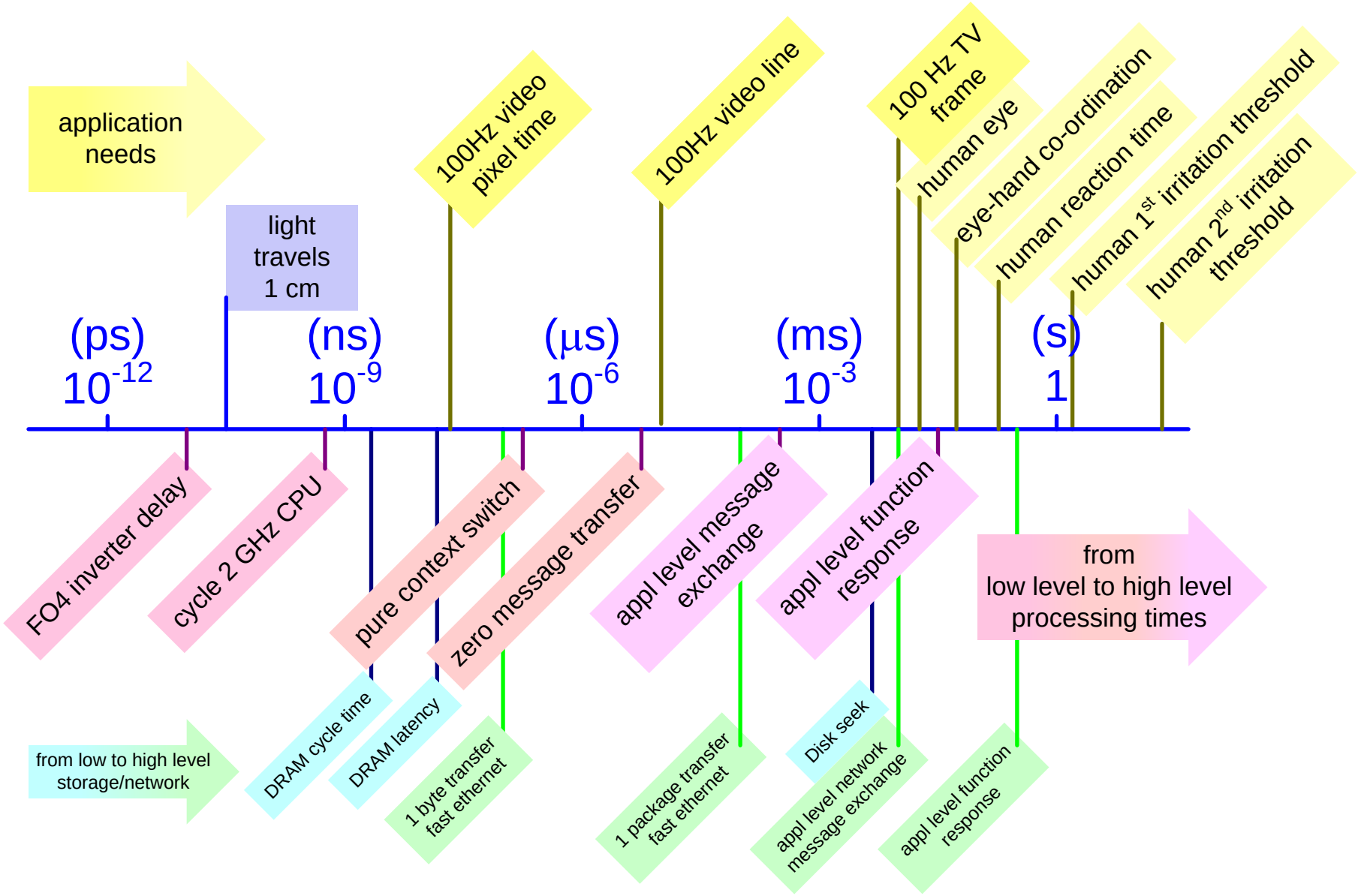
Budget based design flow



Example of a memory budget

memory budget in Mbytes	code	obj data	bulk data	total
shared code	11.0			11.0
User Interface process	0.3	3.0	12.0	15.3
database server	0.3	3.2	3.0	6.5
print server	0.3	1.2	9.0	10.5
optical storage server	0.3	2.0	1.0	3.3
communication server	0.3	2.0	4.0	6.3
UNIX commands	0.3	0.2	0	0.5
compute server	0.3	0.5	6.0	6.8
system monitor	0.3	0.5	0	0.8
application SW total	13.4	12.6	35.0	61.0
UNIX Solaris 2.x				10.0
file cache				3.0
total				74.0

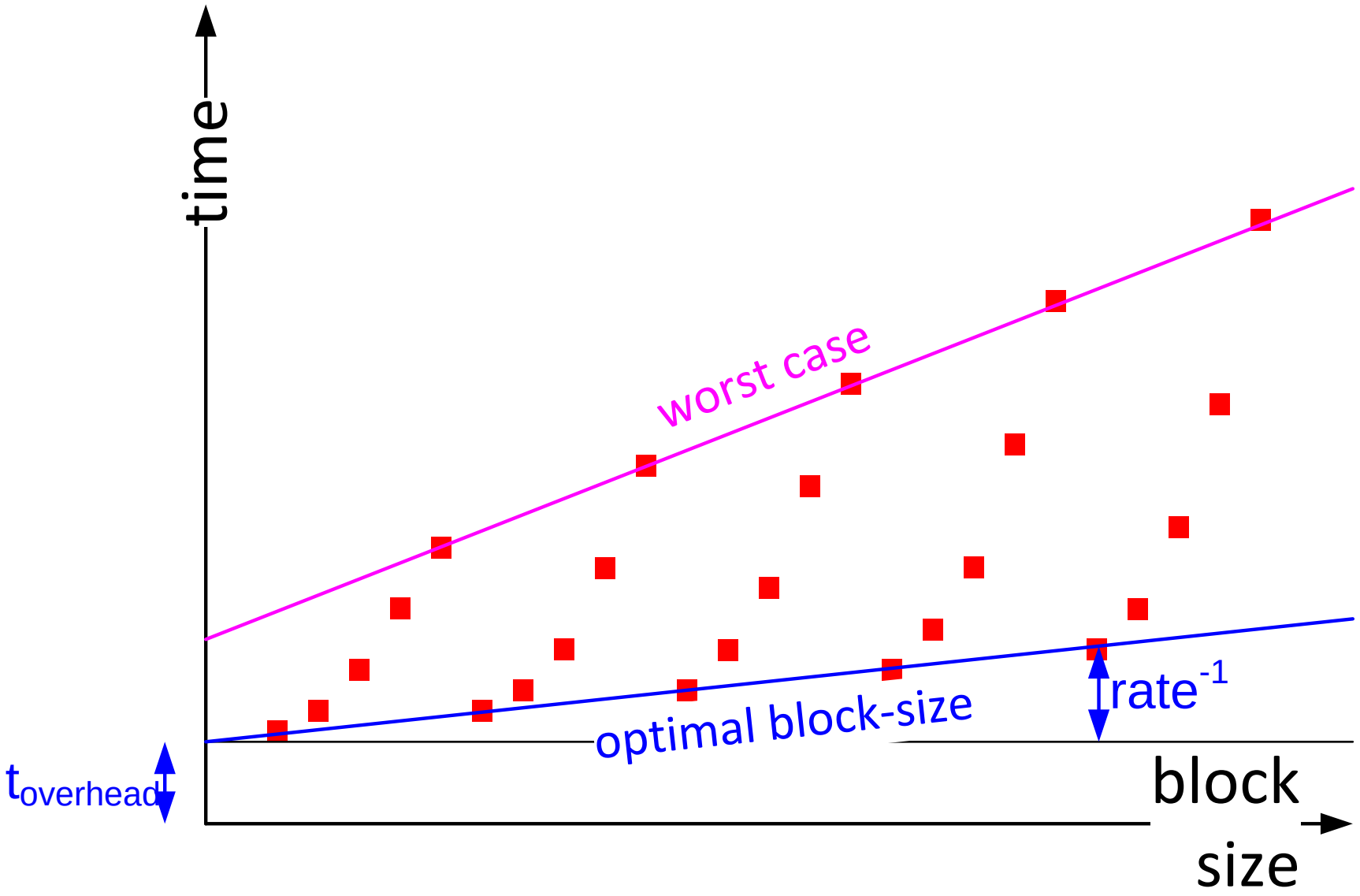
Actual timing on logarithmic scale



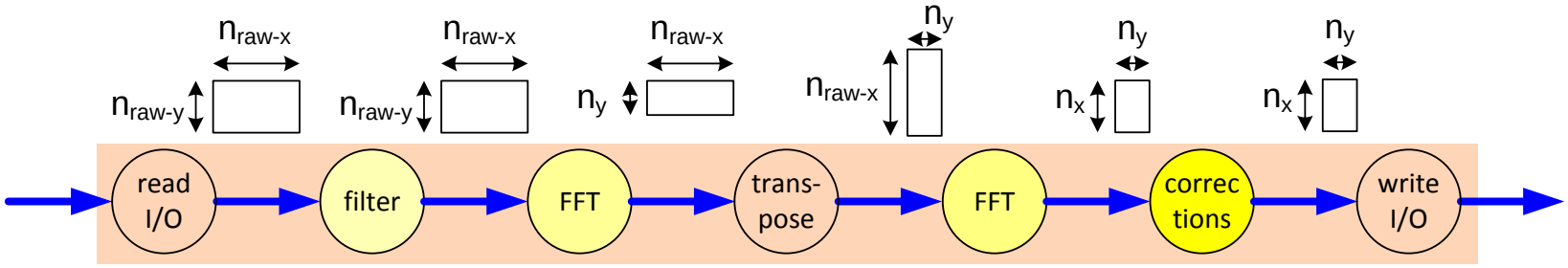
Typical micro benchmarks for timing aspects

	infrequent operations, often time-intensive	*often repeated operations*
database	start session finish session	perform transaction query
network, I/O	open connection close connection	transfer data
high level construction	component creation component destruction	method invocation same scope other context
low level construction	object creation object destruction	method invocation
basic programming	memory allocation memory free	function call loop overhead basic operations (add, mul, load, store)
OS	task, thread creation	task switch interrupt response
HW	power up, power down boot	cache flush low level data transfer

The transfer time as function of blocksize

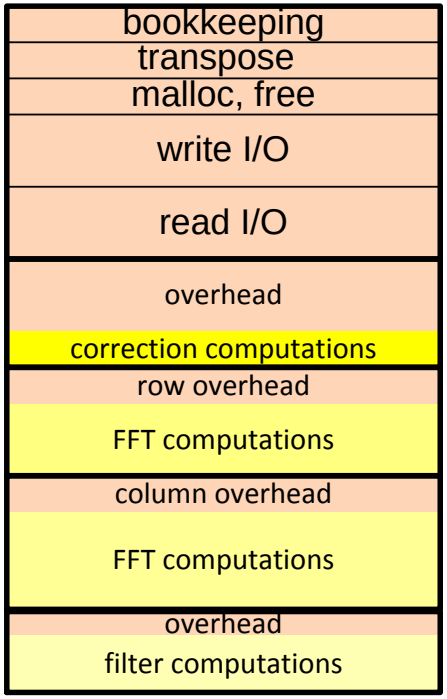


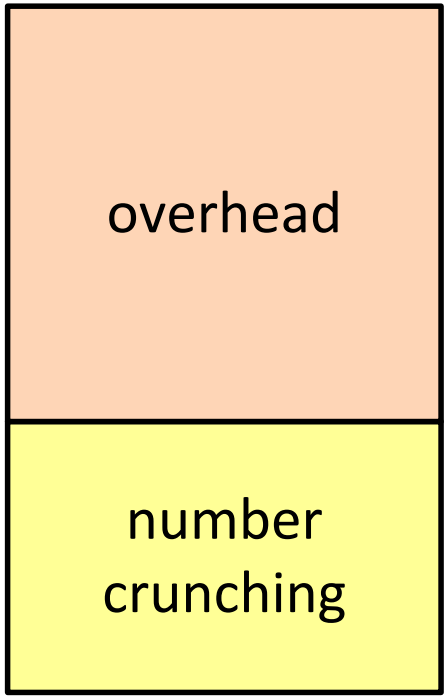
Performance evaluation



$$t_{recon} = t_{filter}(n_{raw-x}, n_{raw-y}) + n_{raw-x} * (t_{fft}(n_{raw-y}) + t_{col-overhead}) + n_y * (t_{fft}(n_{raw-x}) + t_{row-overhead}) + t_{corrections}(n_x, n_y) + t_{read\ I/O} + t_{transpose} + t_{write\ I/O} + t_{control-overhead}$$

$$t_{fft}(n) = c_{fft} * n * \log(n)$$





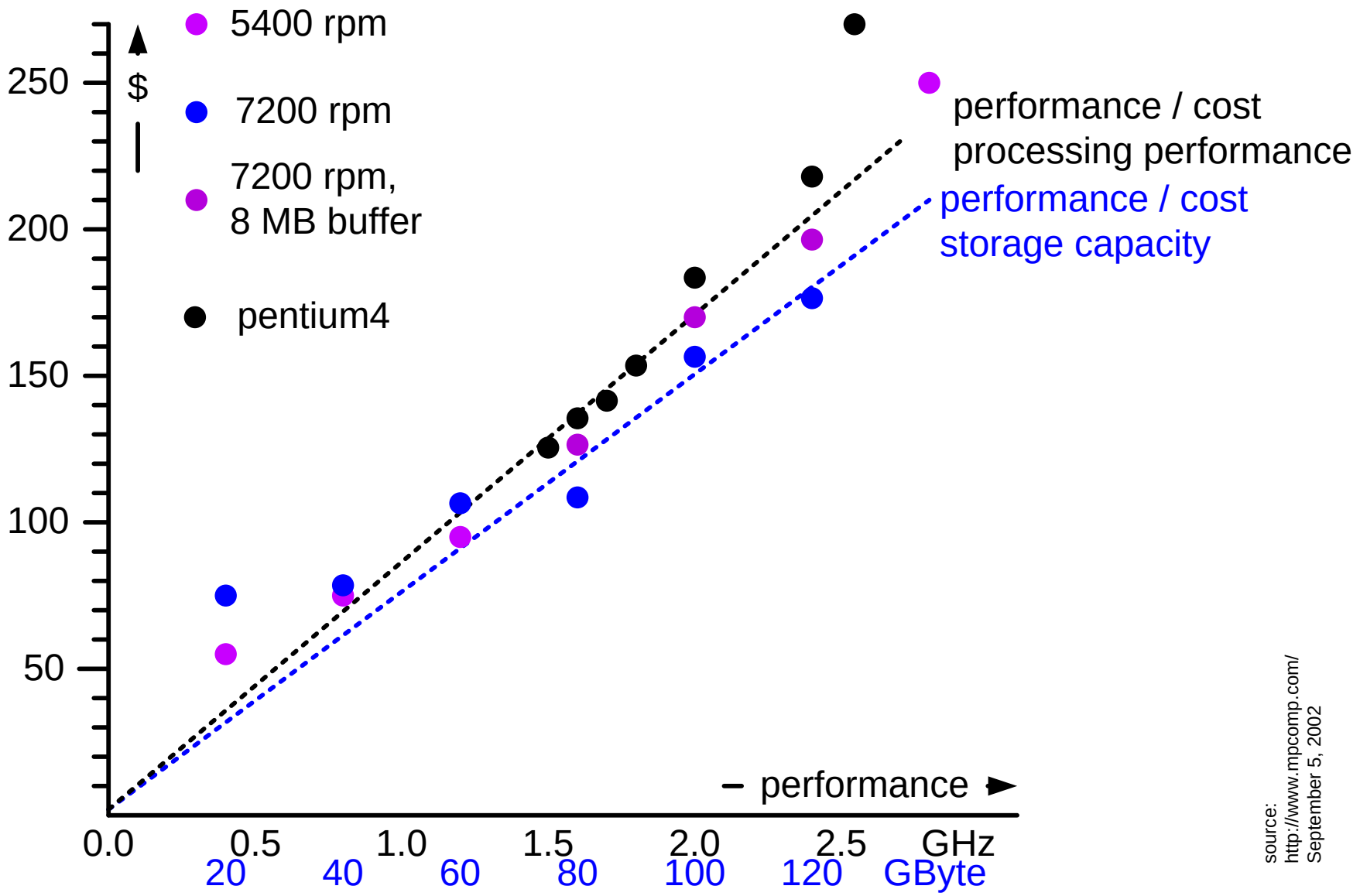
focus on overhead reduction

is more important

than faster algorithms

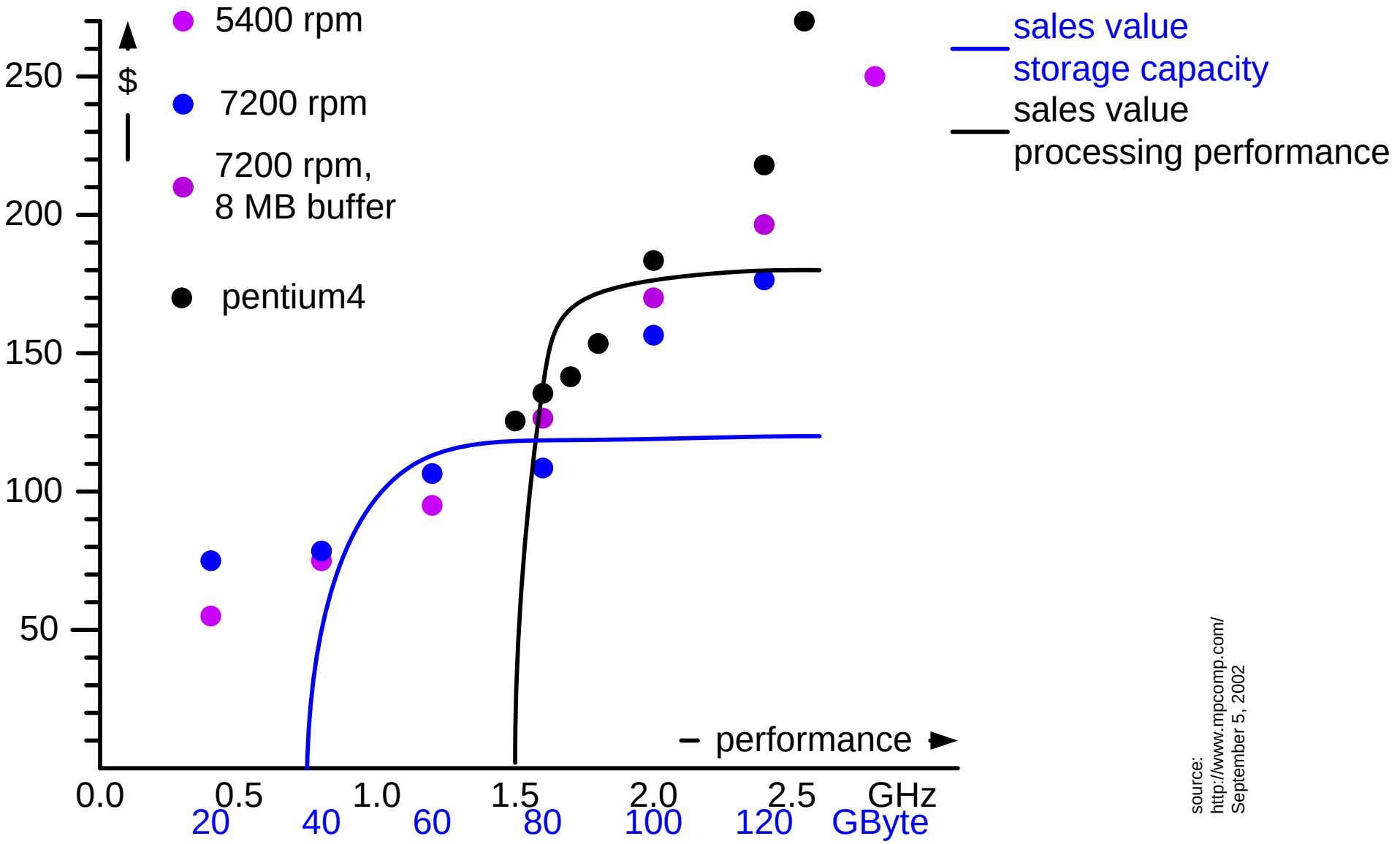
this is not an excuse for sloppy algorithms

Performance Cost, input data



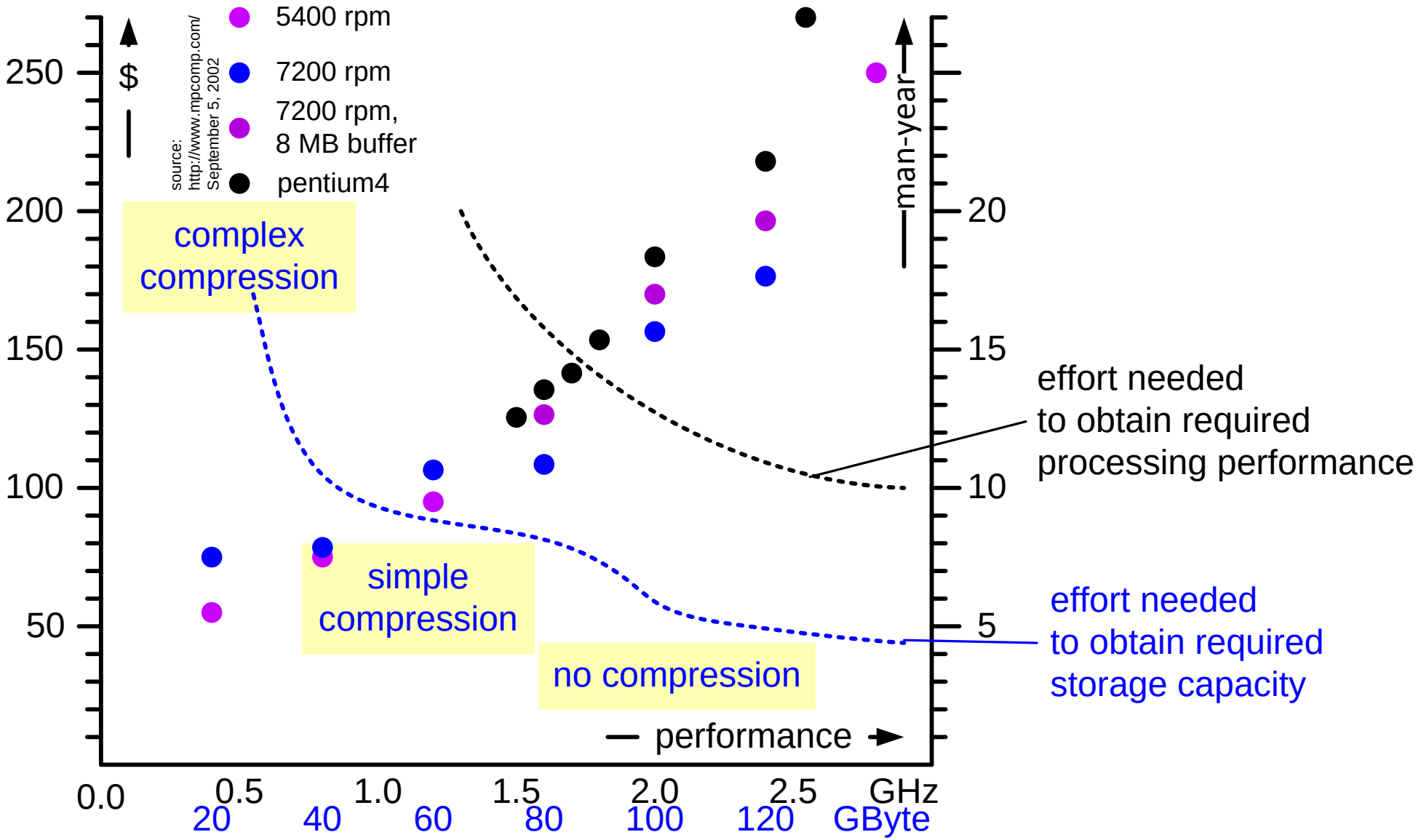
source:
http://www.mpcomp.com/
September 5, 2002

Performance Cost, choice based on sales value

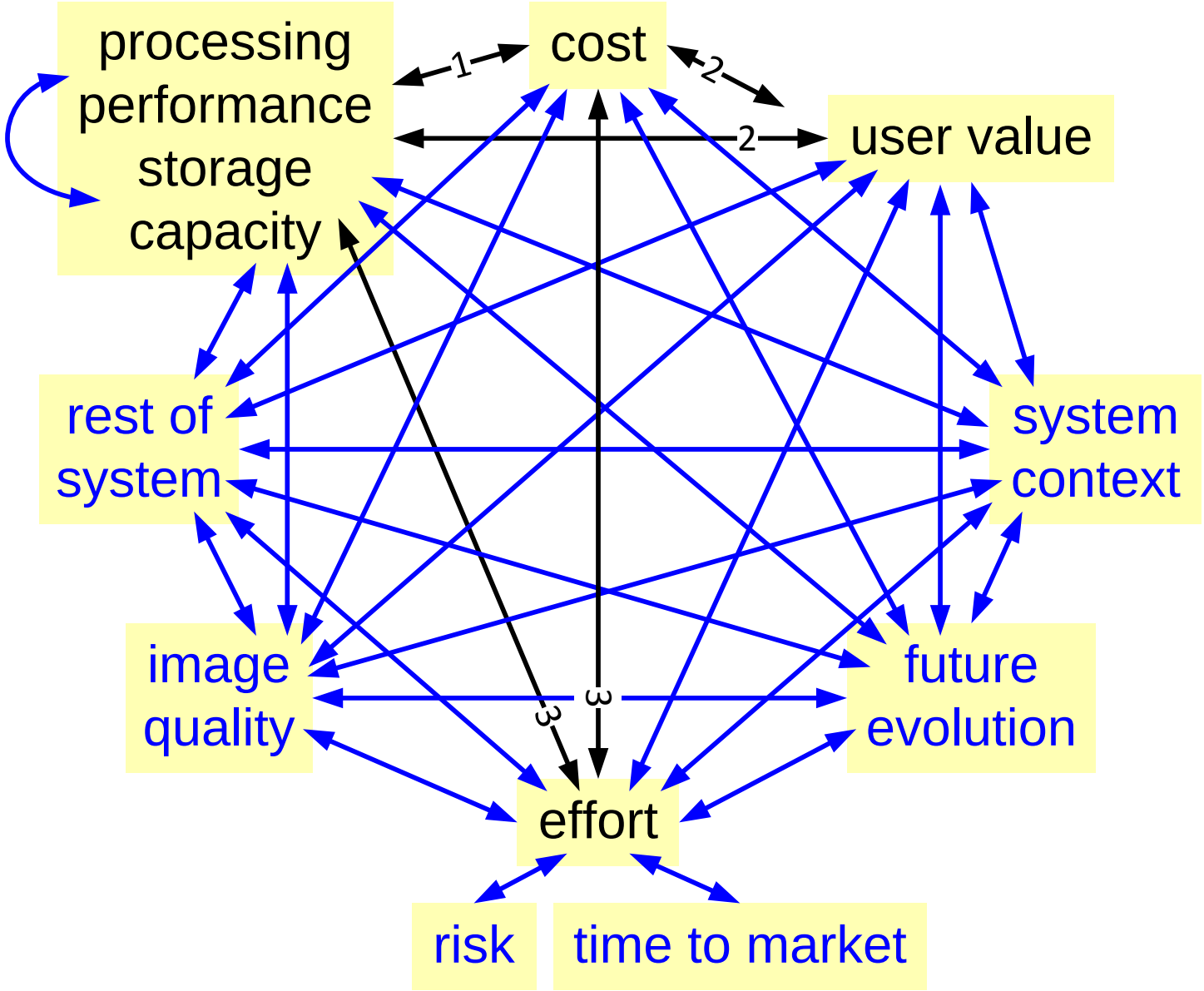


source:
http://www.mpcomp.com/
September 5, 2002

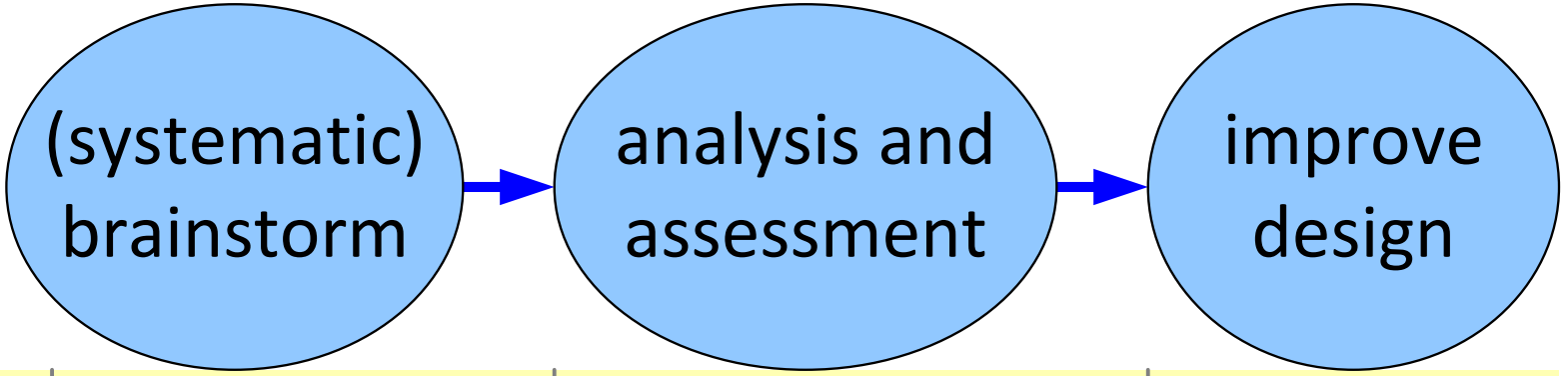
Performance Cost, effort consequences



But many many other considerations



Safety, Reliability and Security analysis methods



safety hazard analysis	potential hazards	probability severity	measures
reliability FMEA	failure modes	effects	measures
security	vulnerability risks	consequences	measures

Exercise Design Side

Make a first design:

- decomposition in functions
- decomposition in building blocks
- budgets for most important quality requirements

Exercise Design Side, second iteration

- Make a design:
  - that covers the most critical design aspects
  - that fulfills the most important and valuable customer needs
- Make a presentation of the design of maximal 8 sheets.

Module Qualities

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

Abstract

This module addresses the Qualities.

October 11, 2020
status: draft
version: 0

logo
TBD

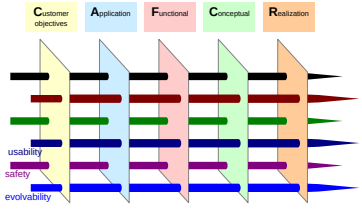
Qualities as Integrating Needles

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

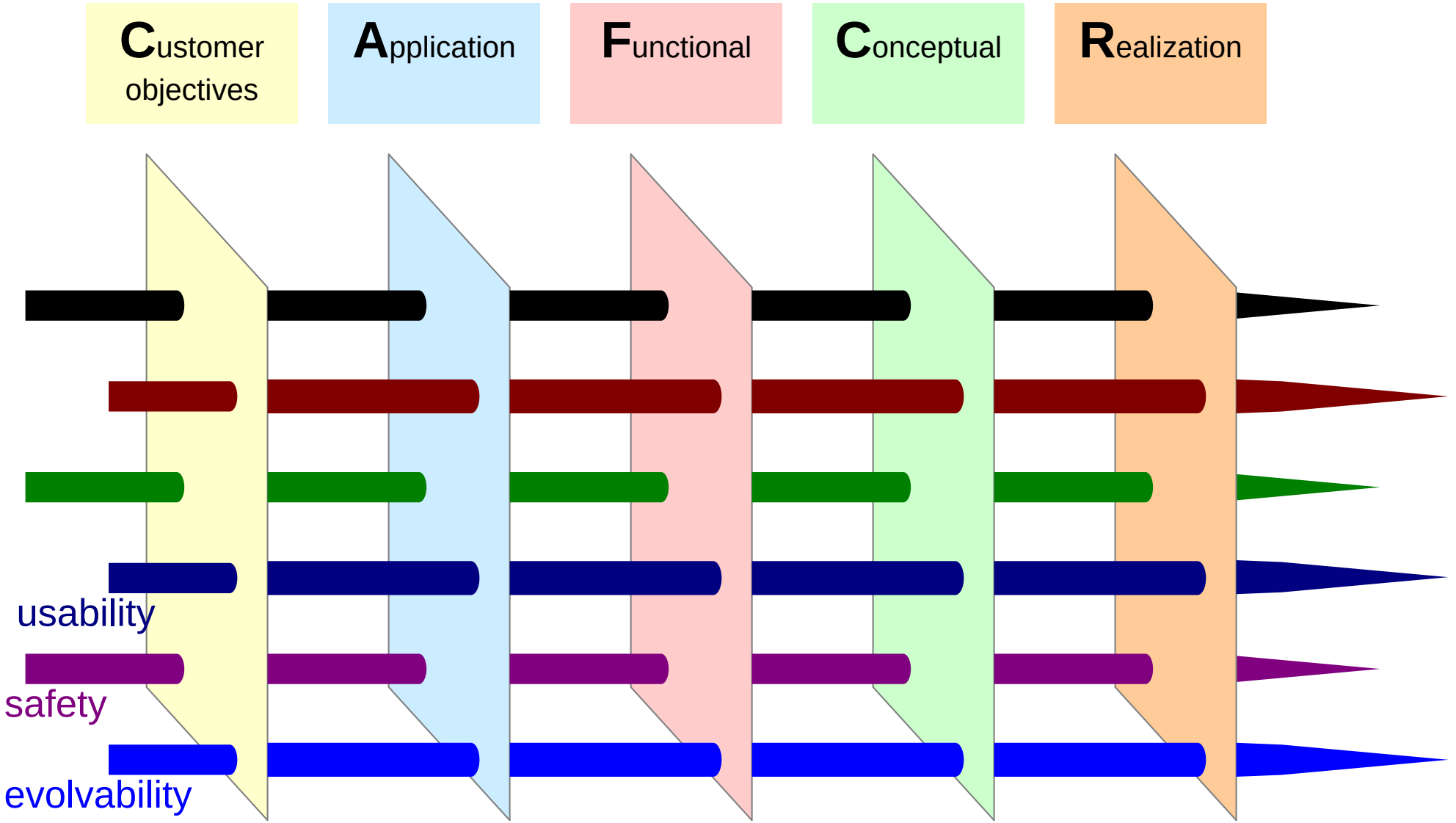
Abstract

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.

October 11, 2020
status: finished
version: 1.3



Quality needles as generic integrating concepts



Security as example through all views

Customer objectives	**A**pplication	**F**unctional	**C**onceptual	**R**ealization
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
	desired characteristics, specifications & mechanisms			
 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
			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

Exercise Qualities

- Determine most important qualities.
- Annotate 3 most important qualities in every CAFCR view

Module Scenarios, Story Telling and Use Cases

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

Abstract

This module addresses Scenarios, Story Telling and Use Cases. Scenarios are used to cope with multiple alternatives for specification or design. Story telling is a means to explore customer needs and as a means for communication. Use Cases are used to analyze the design for specific circumstances.

October 11, 2020
status: preliminary draft
version: 0.1

logo
TBD

Content Scenarios, Story Telling, Use Cases

goal of this module

Be able to apply story telling technique.

Be able to use scenario analysis.

Be able to use use-cases for design.

content of this module

Format and criteria for stories

Elements of scenarios

Role of scenarios in decision making

Quantified use cases

exercise

Create a story and translate story via use cases in design

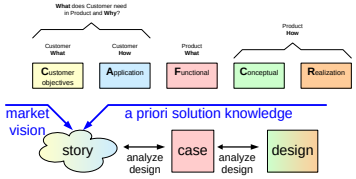
Story How To

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

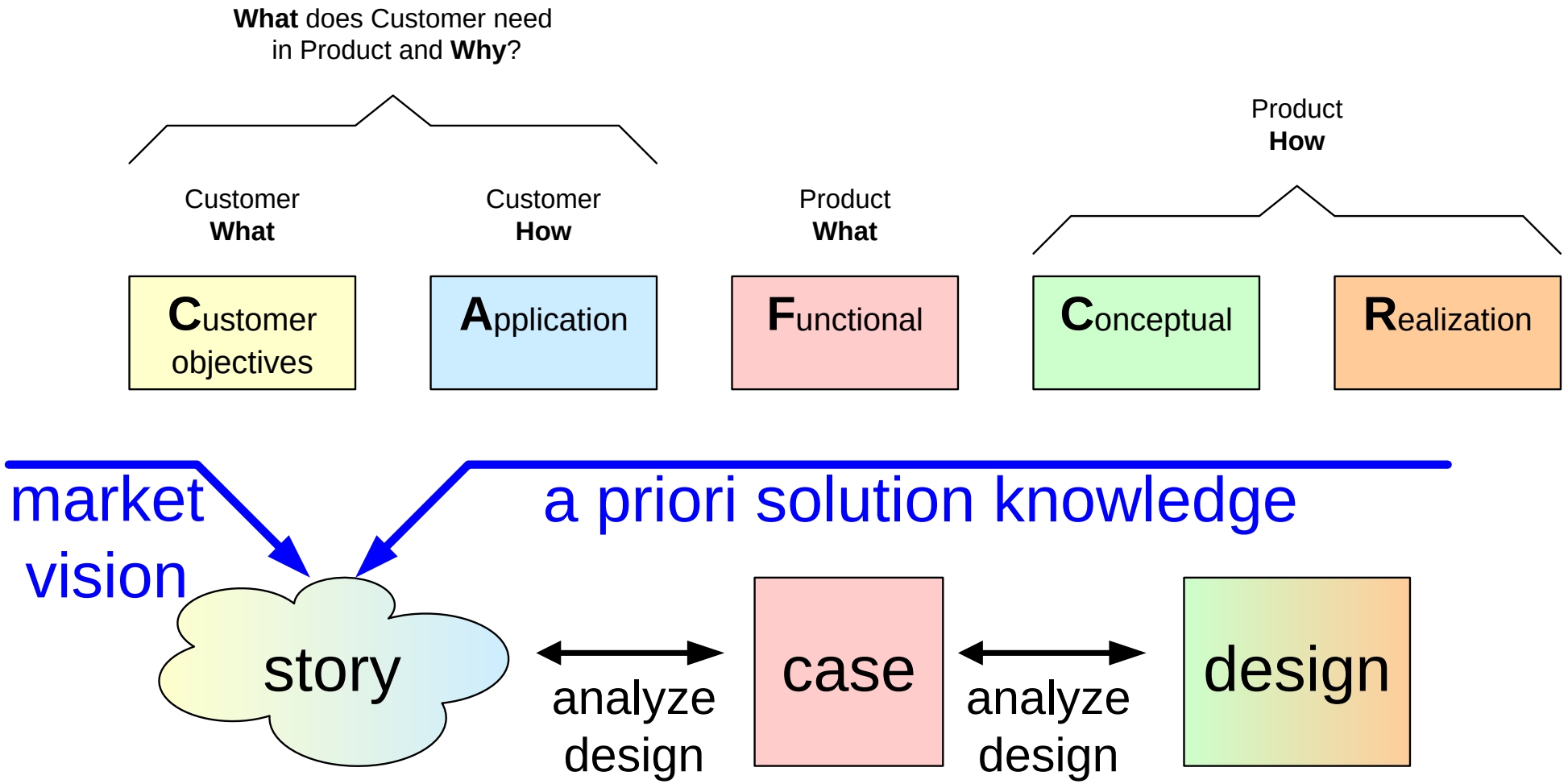
Abstract

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.

October 11, 2020
status: concept
version: 1.2

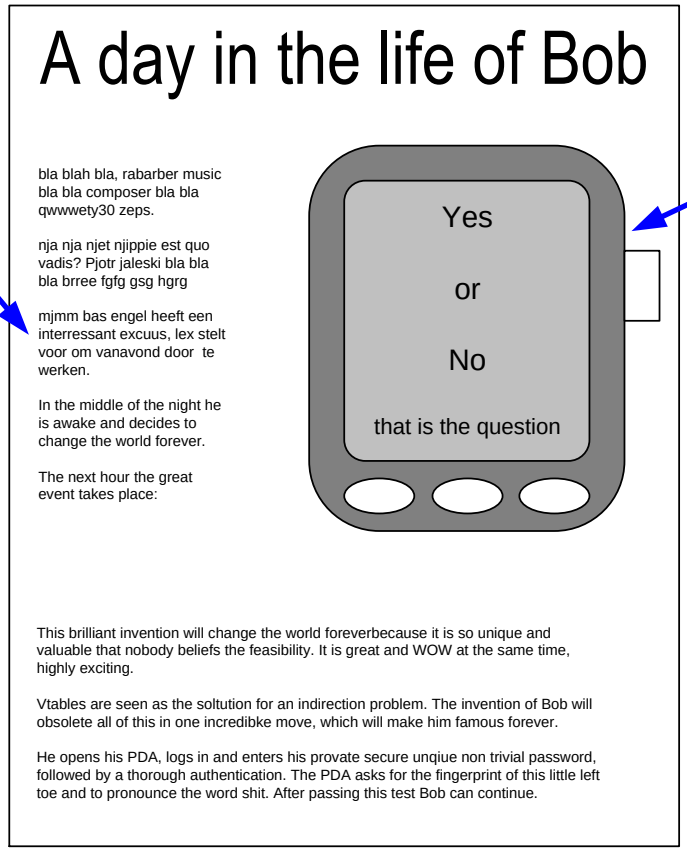


From story to design



Example story layout

ca. half a page of plain English text



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

Scenario How To

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

Abstract

Good designers keep multiple alternatives open in parallel. This improves the specification and design quality. Scenarios can be used to cope with these alternatives and as a means for communication with stakeholders.

October 11, 2020
status: planned
version: 0

logo
TBD

Scenarios

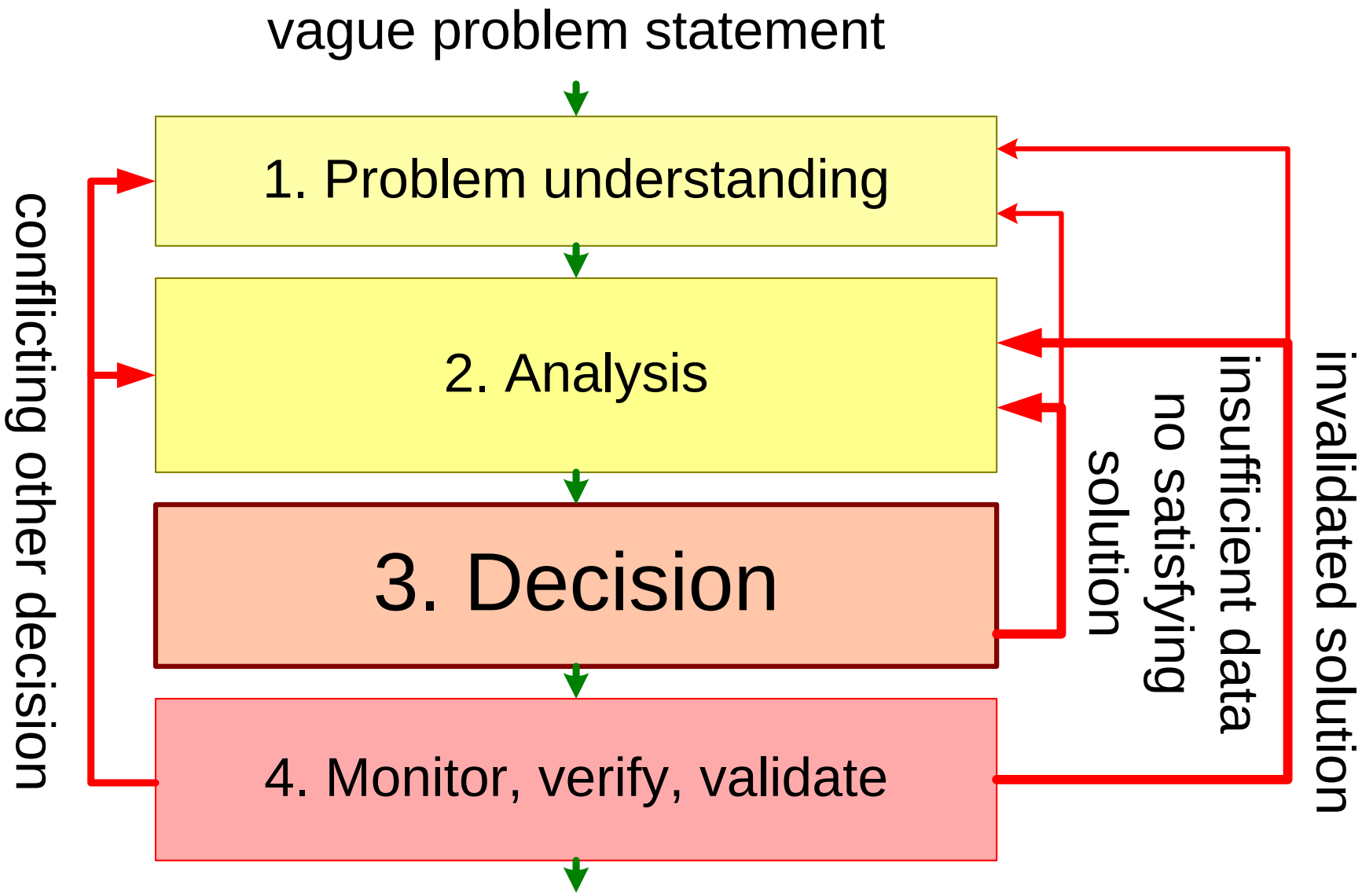
content of this presentation

Decision making

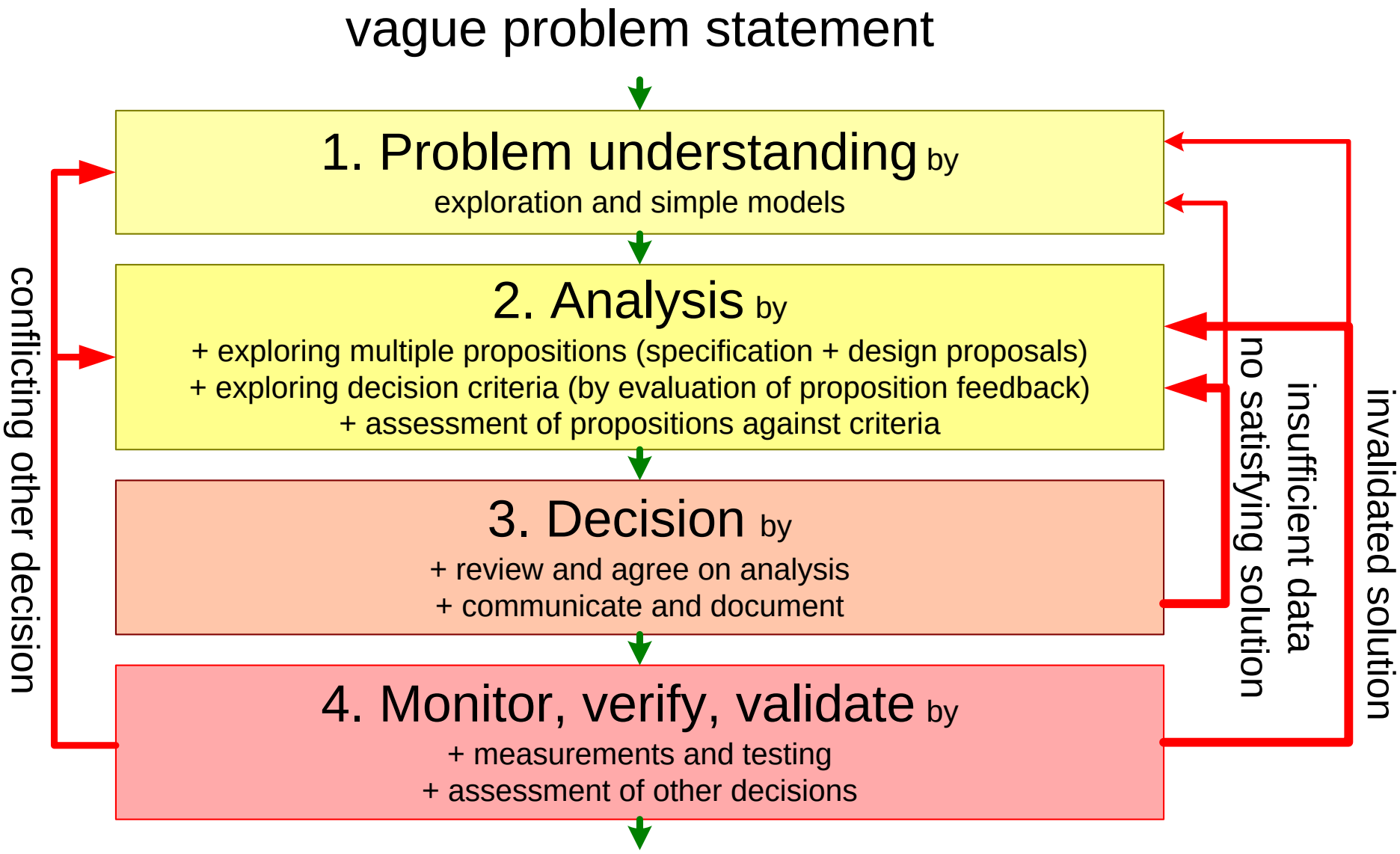
Multiple propositions

Scenarios

Decision Making Process



Flow from problem to solution



Example of 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

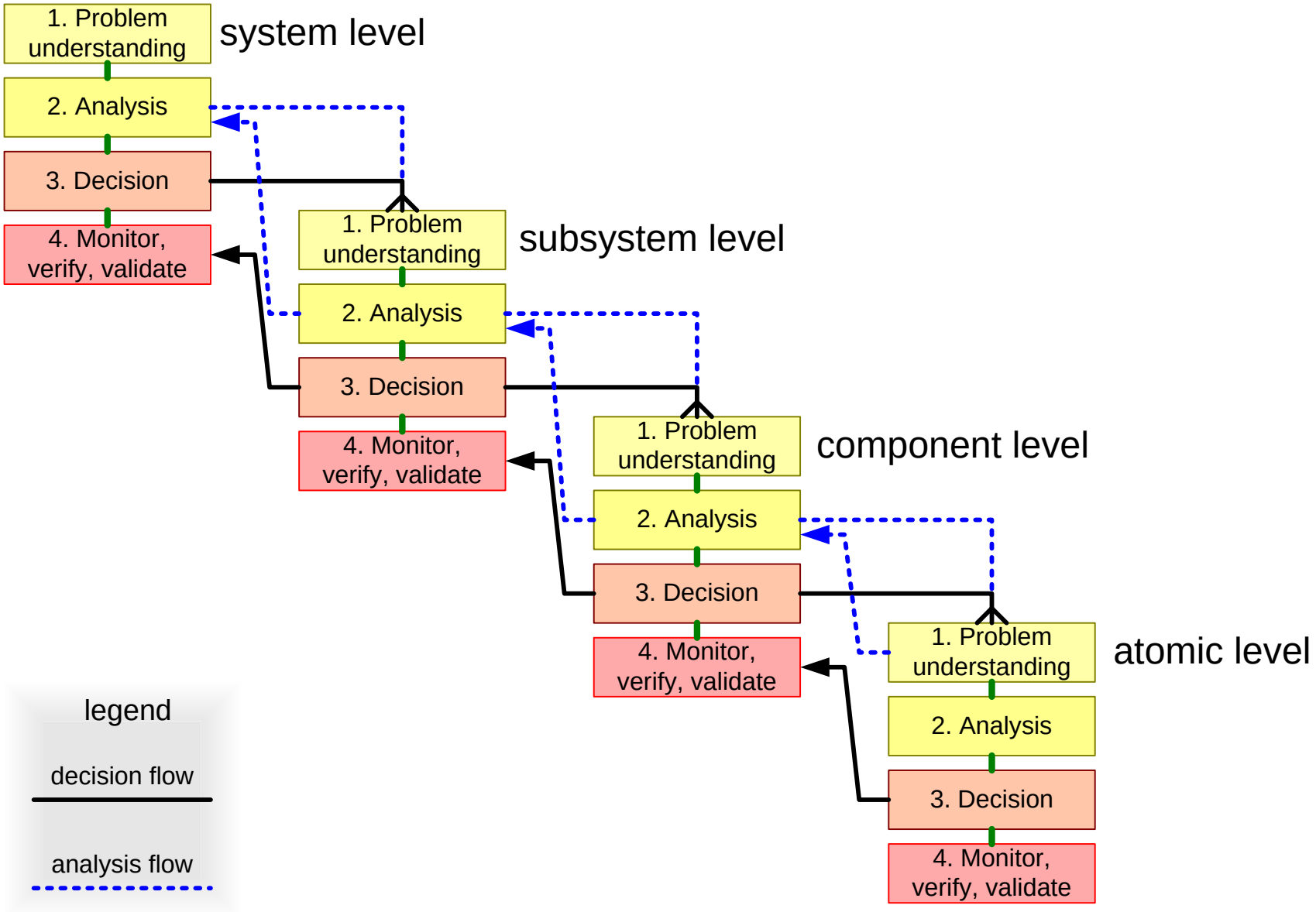
throughput	20 p/m	high-performance sensor	300 ns
cost	5 k$	high-speed moves	10 m/s
safety			

low cost and performance 2

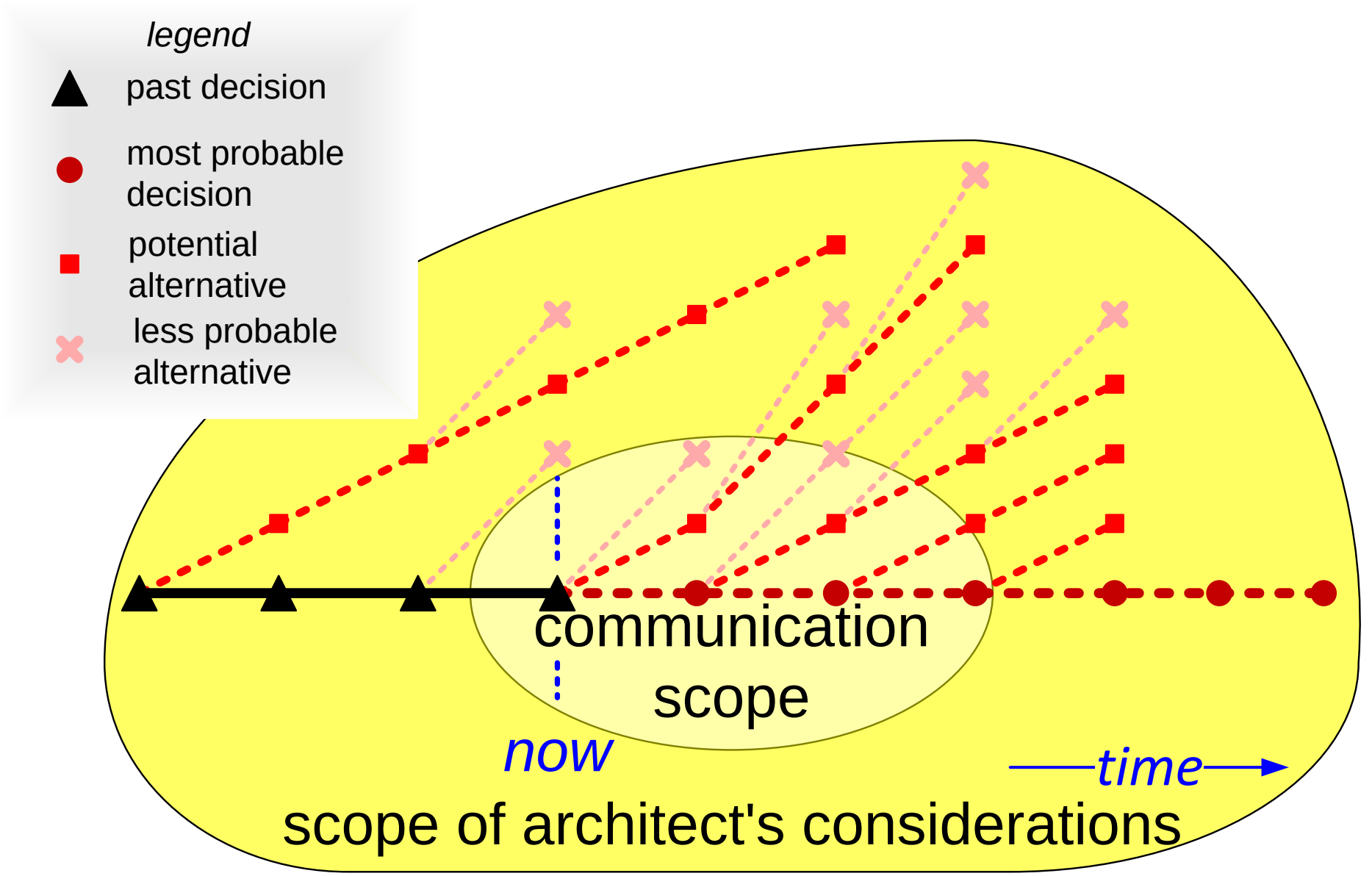
throughput	25 p/m	highperformance sensor	200 ns
cost	7 k$	high-speed moves	12 m/s
safety		additional collision detector	

high cost and performance

Recursive and concurrent application of flow



Graph of Decisions and Alternatives



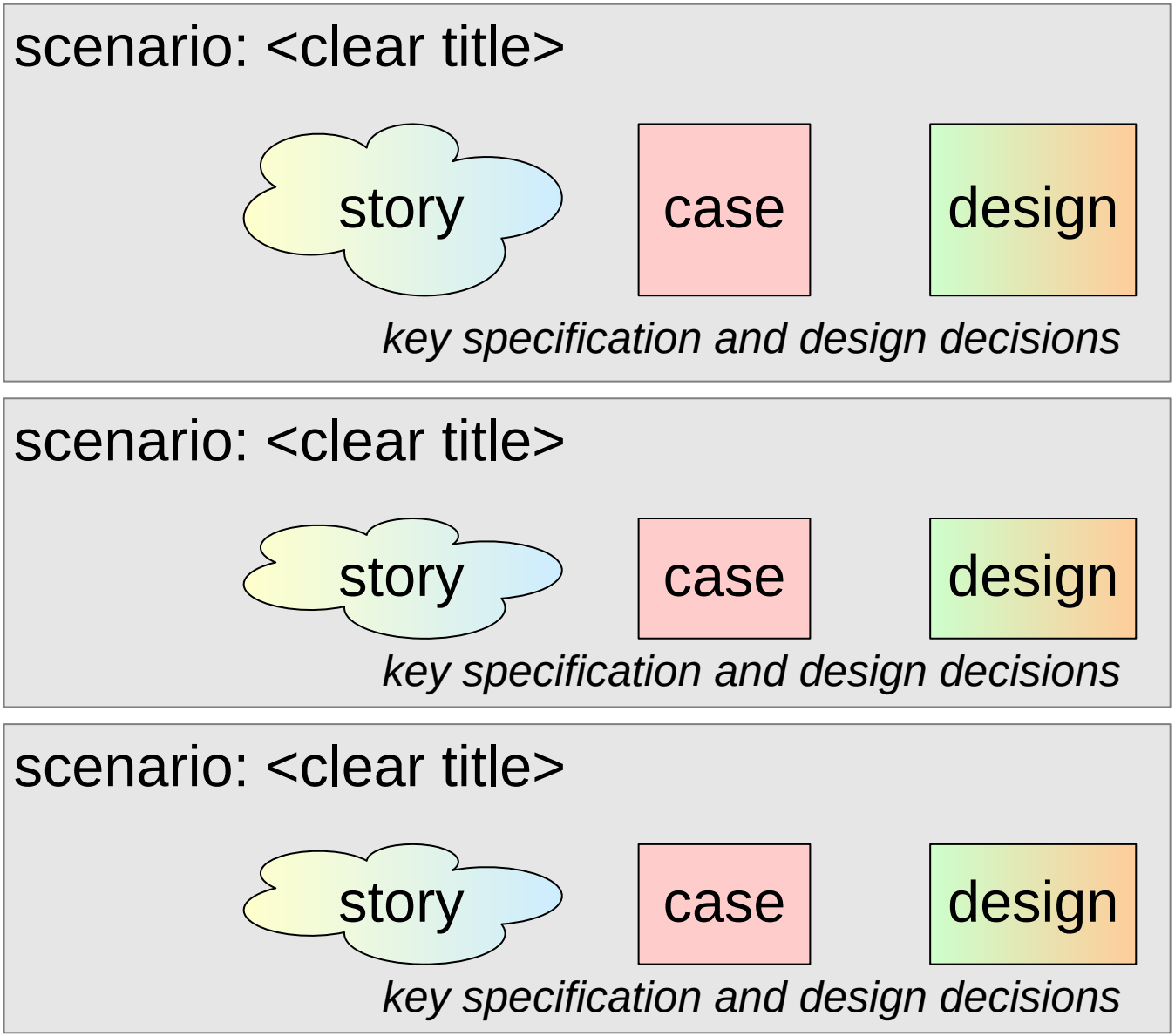
Different Types of Decisions

basic principles

requirements

architecture rules
implementation choices
f.i. technology

Elements of a Scenario



Summary of Scenarios

Summary of Scenarios

Exploration and analysis require multiple propositions.

Architects continuously work with multiple alternatives.

Scenarios have a clear title, story, use case and design.

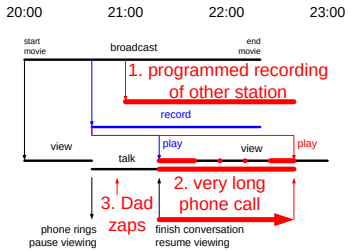
Scenarios are differentiated by key specifications and design decisions.

Use Case How To

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

Abstract

Use cases are frequently used in Software Engineering. Use cases support specification and facilitate design, analysis, verification and testing. Many designers, unfortunately, apply use cases in a rather limited way. This presentation provides recommendations for effective use cases.



October 11, 2020
status: planned
version: 0.1

Why Use Cases?

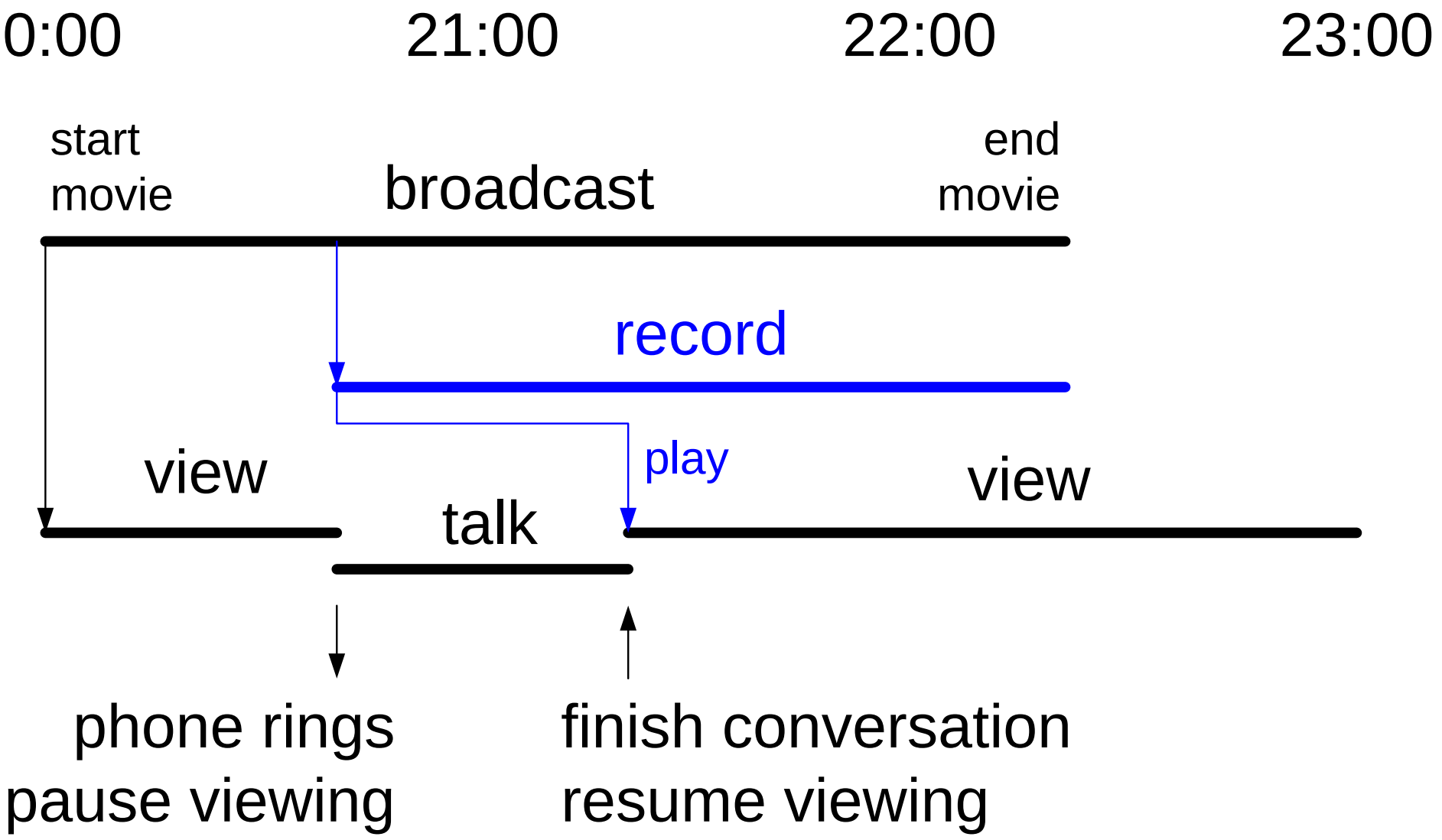
Supports or is part of specification

by providing specific data in user perspective

Facilitates analysis and design

Facilitates verification and testing

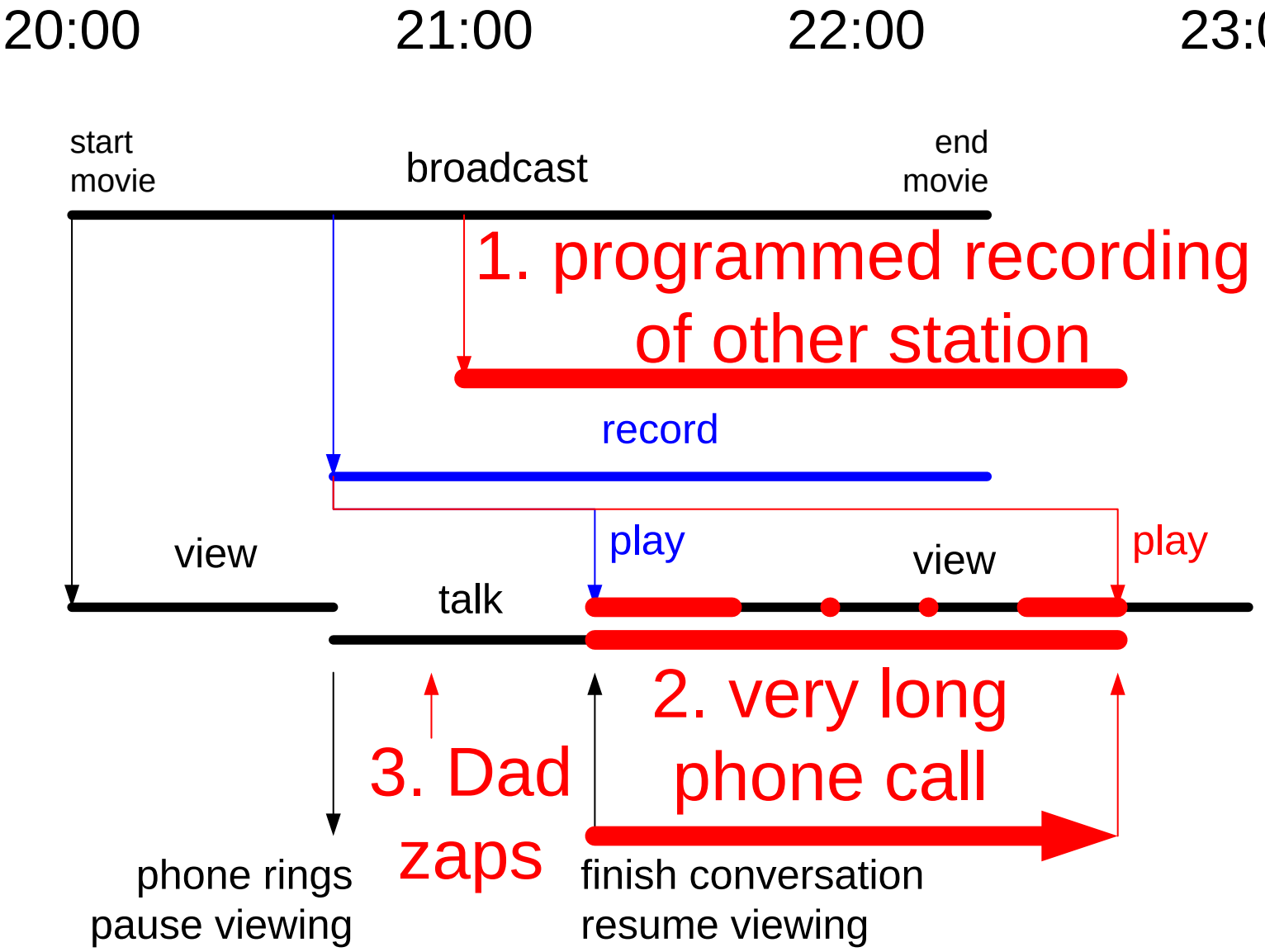
Example Time Shift recording



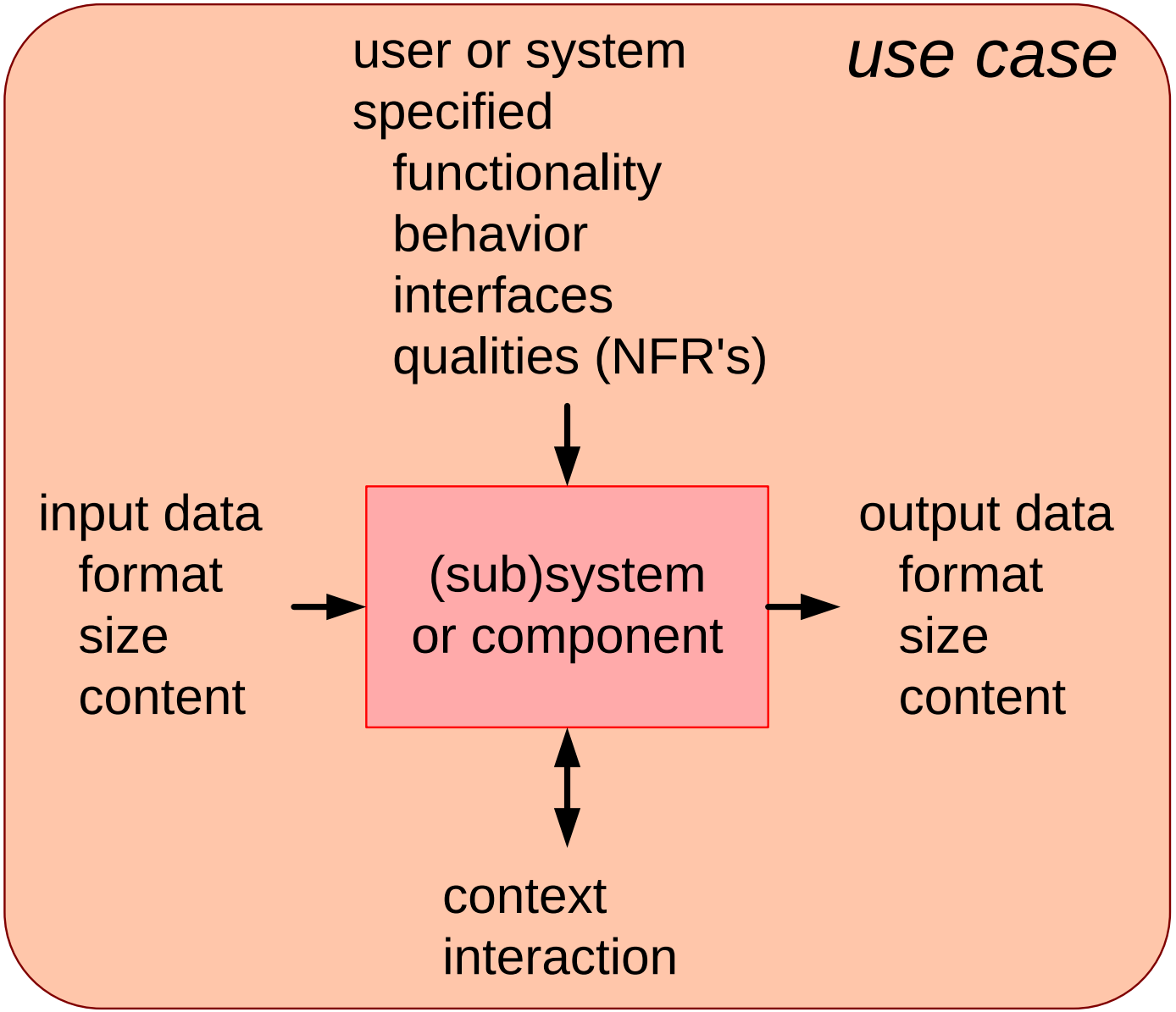
Construction limits intrude in User Experience

- number of tuners
- number of simultaneous streams (recording and playing)
- amount of available storage
- management strategy of storage space

What if?


20:00
21:00
22:00
23:00
start movie
broadcast
end movie
1. programmed recording of other station
record
view
play
view
play
talk
2. very long phone call
3. Dad zaps
phone rings
pause viewing
finish conversation
resume viewing

Content of a Use Case

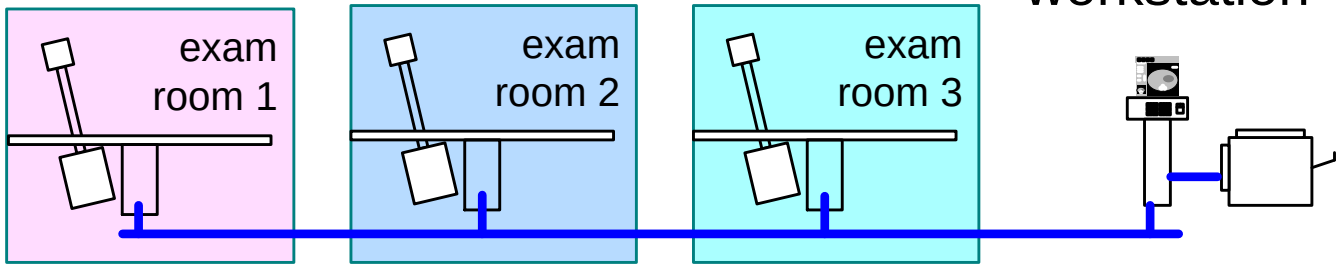


Example personal video recorder use case contents

typical use case(s)	worst case, exceptional, or change use case(s)
interaction flow (functional aspects) select movie via directory start movie be able to pause or stop be able to skip forward or backward set recording quality	functional multiple inputs at the same time extreme long movie directory behaviour in case of extreme many short movies
performance and other qualities (non-functional aspects) response times for start / stop response times for directory browsing end-of-movie behaviour relation recording quality and storage	non-functional response time with multiple inputs image quality with multiple inputs insufficient free space response time with many directory entries replay quality while HQ recording

Example of Quantification of Typical Use Case

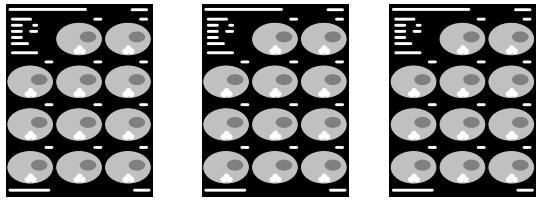
3 examination rooms connected to 1 medical imaging workstation + printer



examination room: average 4 interleaved examinations / hour

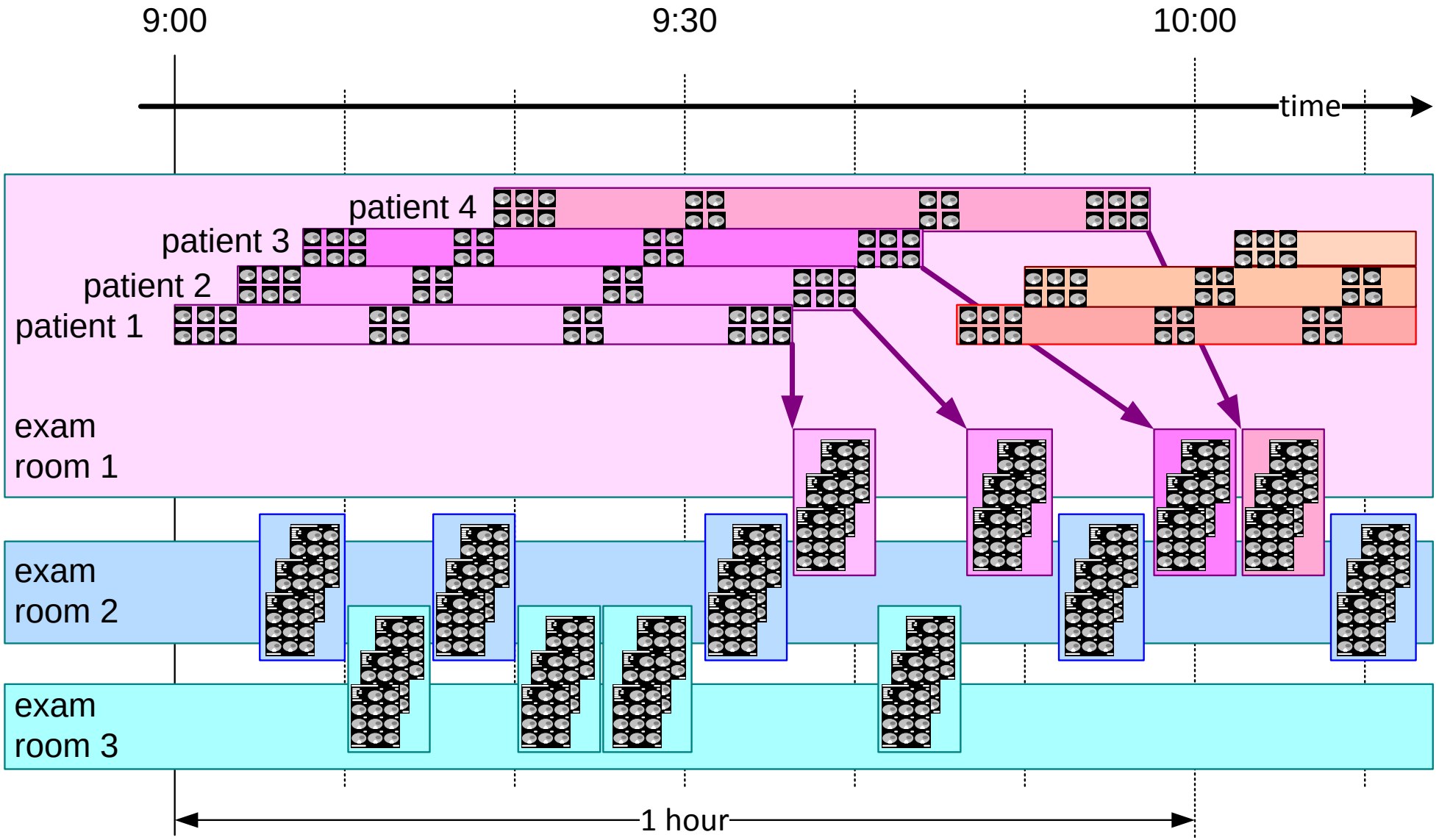
image production: 20 1024^2 8 bit images per examination

film production: 3 films of 4k*5k pixels each



high quality output
(bi-cubic interpolation)

Timing of this Use Case



Recommendations for working with use cases

+ combine related functions in one use case

- do not make a separate use case for every function

+ include non-functional requirements in the use cases

+ minimise the amount of required *worst case* and *exceptional use cases*

- excessive amounts of use cases propagate to excessive implementation efforts

+ reduce the amount of these use cases in steps

- a few well chosen *worst case* use cases simplifies the design

Exercise Scenarios, Story Telling, Use Cases

1. Create a story
   - use the criteria
2. Transform the story into a case
   - functional, as well as quantitative
3. Perform a short design exploration
   - based on the case.
4. Improve the story
   - first iteration based on feedback from case and design.

- Use time boxes to ensure that you make all the indicated steps.

Exercise Reflection

+ stories make discussions much more specific
+ implicit assumptions are identified

~ creating relevant stories is far from trivial

- too much fun

starting point for generalization: specification and design

Summary Scenarios, Story Telling, Use Cases

Conclusions

Stories help to focus early design discussions

Scenarios help to cope with multiple alternatives

Use cases address integral use: functional and quantitative

Techniques, Models, Heuristics of this module

Story telling, criterias

Scenarios

Quantified use cases

Worst case, *exceptional* and *change* use cases

Module Reasoning: Linking Business to Technology

by *Gerrit Muller* HSN-NISE
e-mail: gaudisite@gmail.com
www.gaudisite.nl

Abstract

This module addresses *Threads of Reasoning* as a means to connect business and operational needs to design and technology choices.

October 11, 2020
status: preliminary draft
version: 0.2

Module Content

goal of this module

Be able to relate *Customer* and *Operational* objectives to design and technology choices.

Be able to provide rationale for design decisions.

content of this module

Key driver method and recommendations

Threads of reasoning approach

Example in Health Care domain

exercise

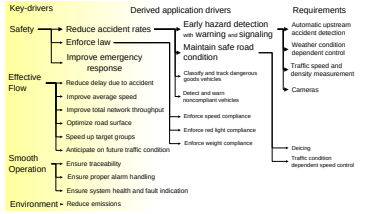
Key driver graph

Key Drivers How To

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

Abstract

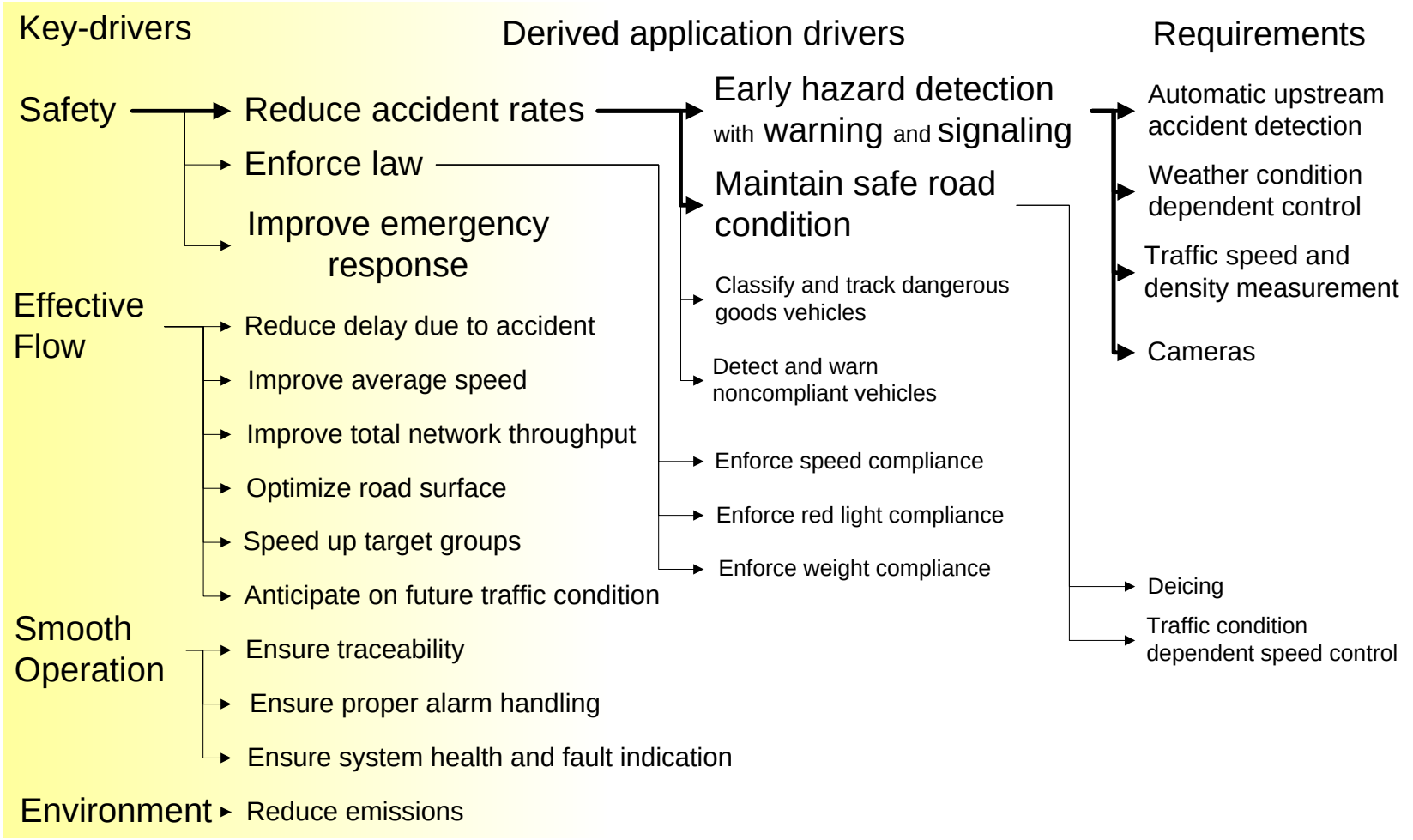
The notion of ”business key drivers” is introduced and a method is described to link these key drivers to the product specification.



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

October 11, 2020
status: draft
version: 0.2

Example Motorway Management Analysis



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

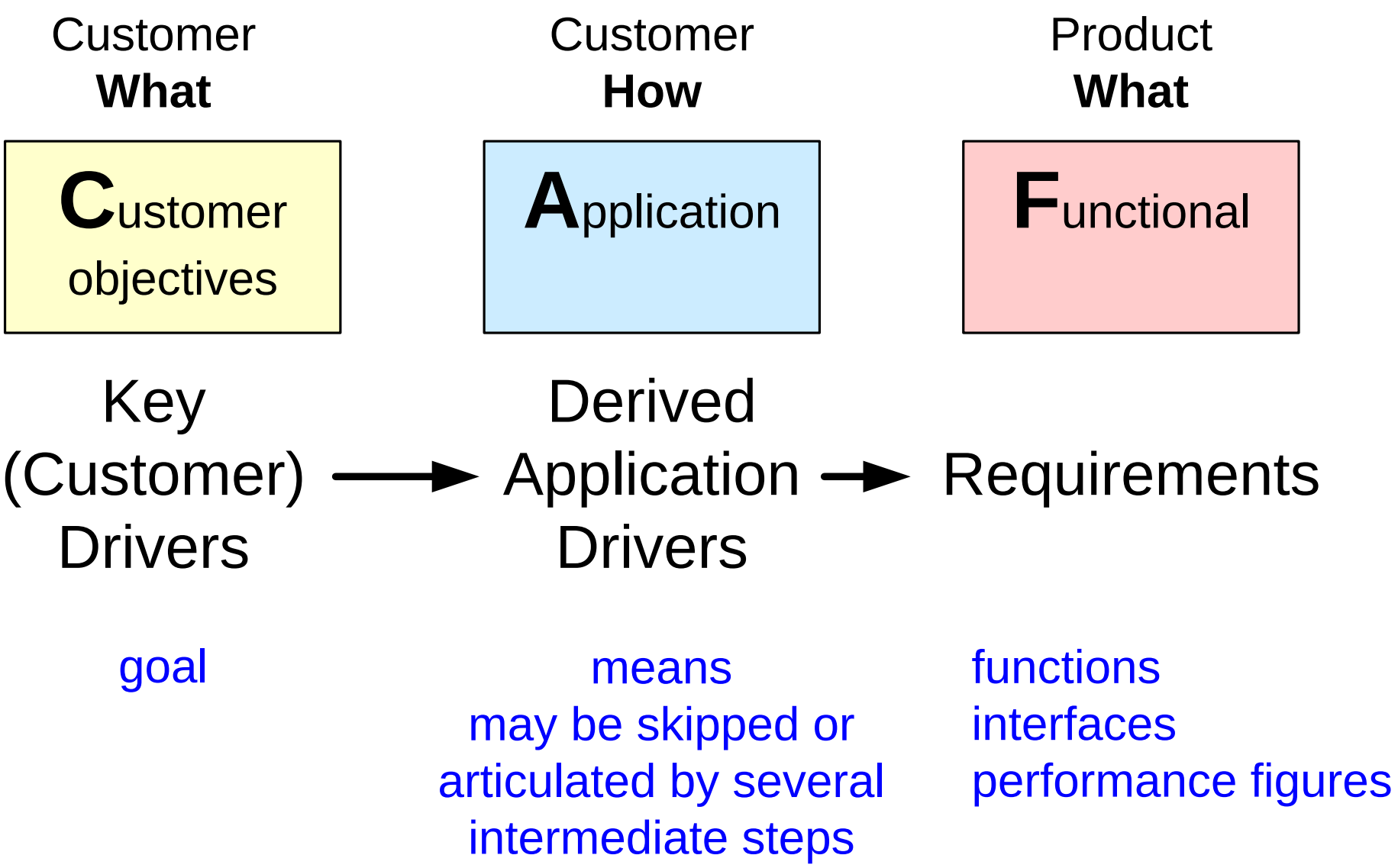
Method to create Key Driver Graph

• 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



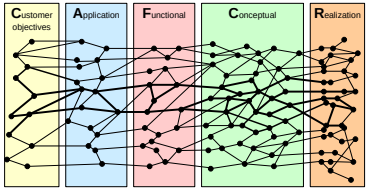
Threads of Reasoning

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

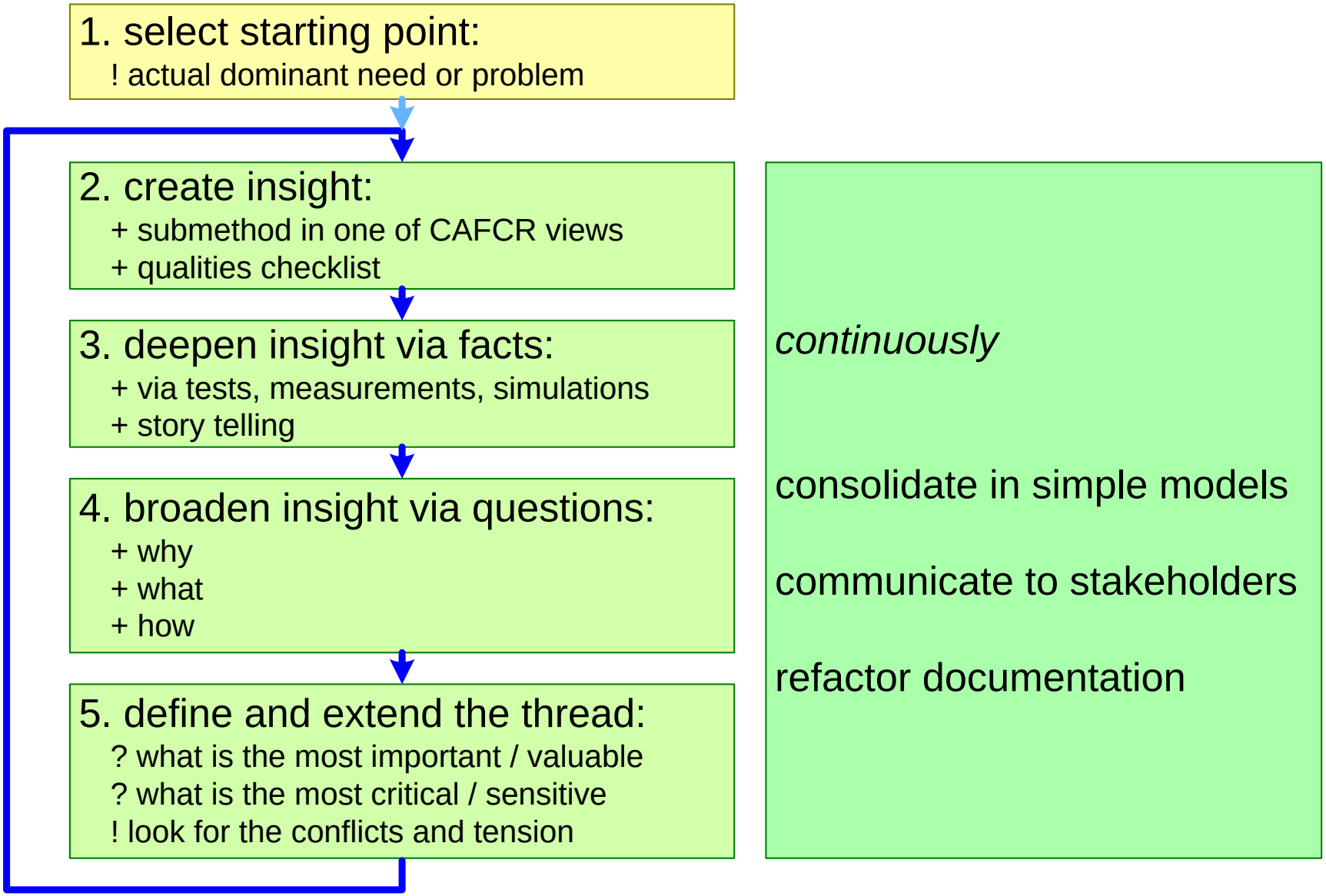
Abstract

A method of reasoning is described, which addresses cross-cutting issues. The basis is fast iteration in the problem and solution space.
A thread of reasoning is a set of highly relevant related issues, which are addressed by articulating the problem in terms of tension and analyzing it in the CAFCR framework.

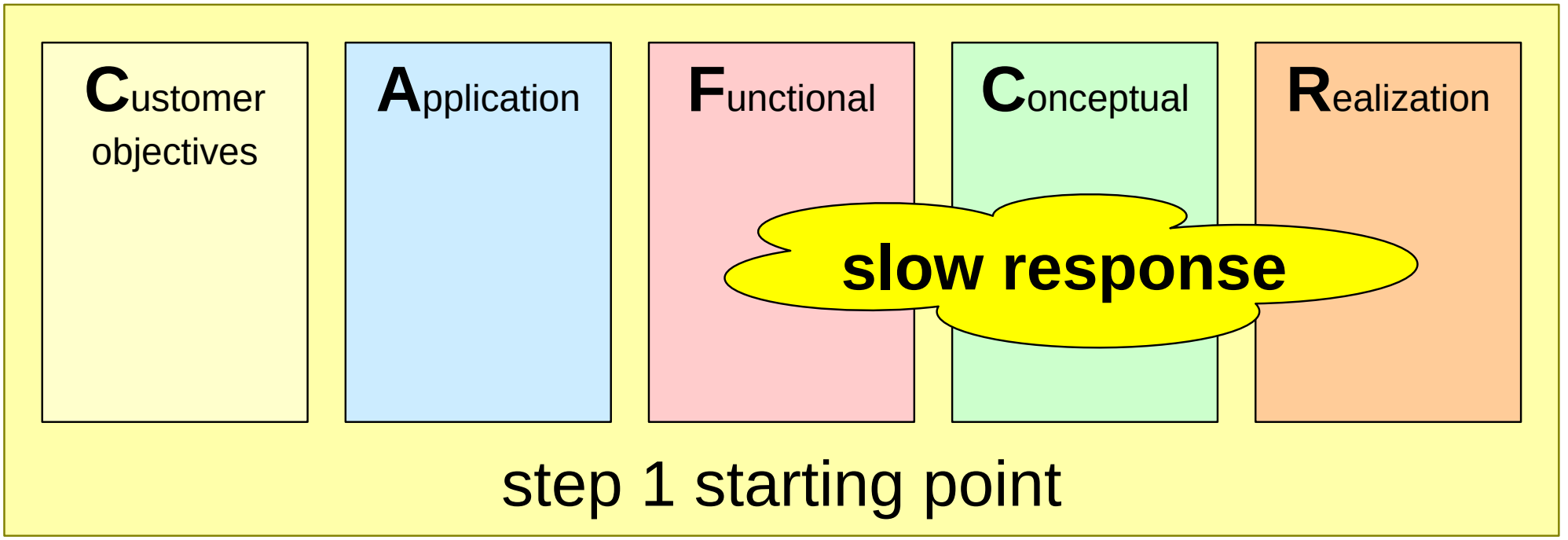
October 11, 2020
status: finished
version: 2.4



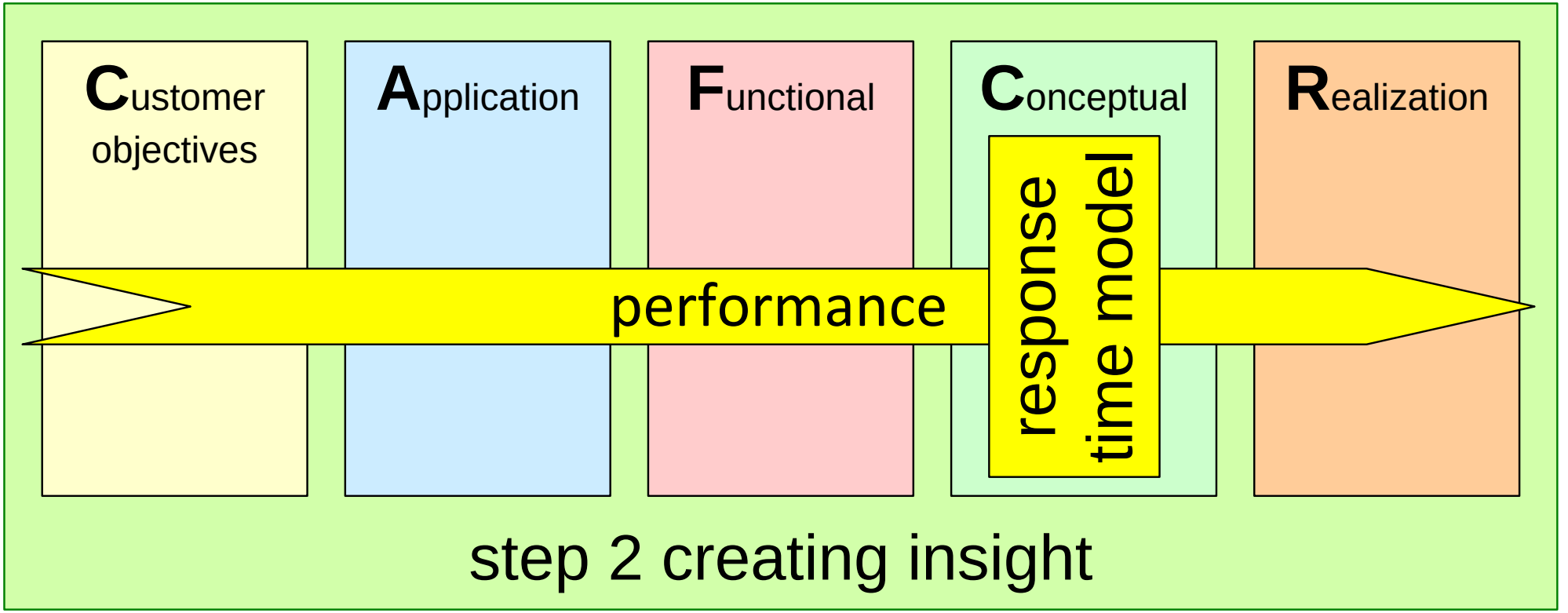
Overview of the reasoning approach



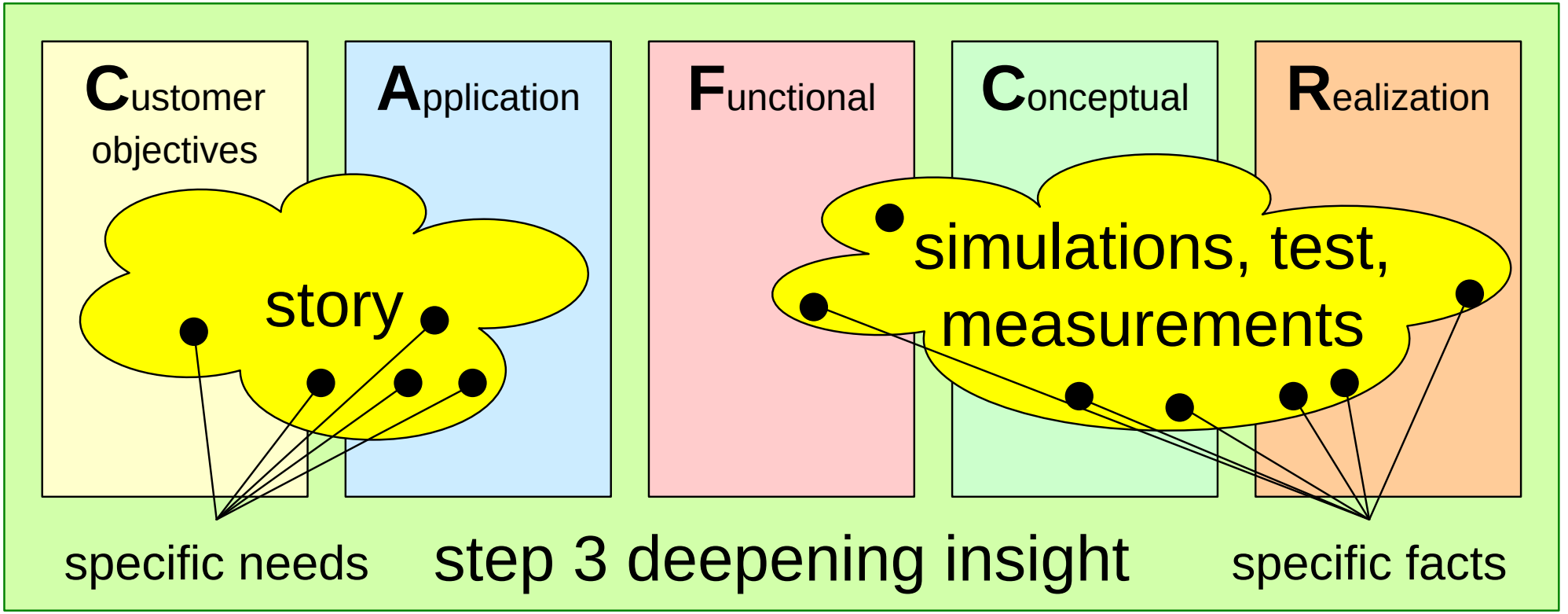
From starting point to insight



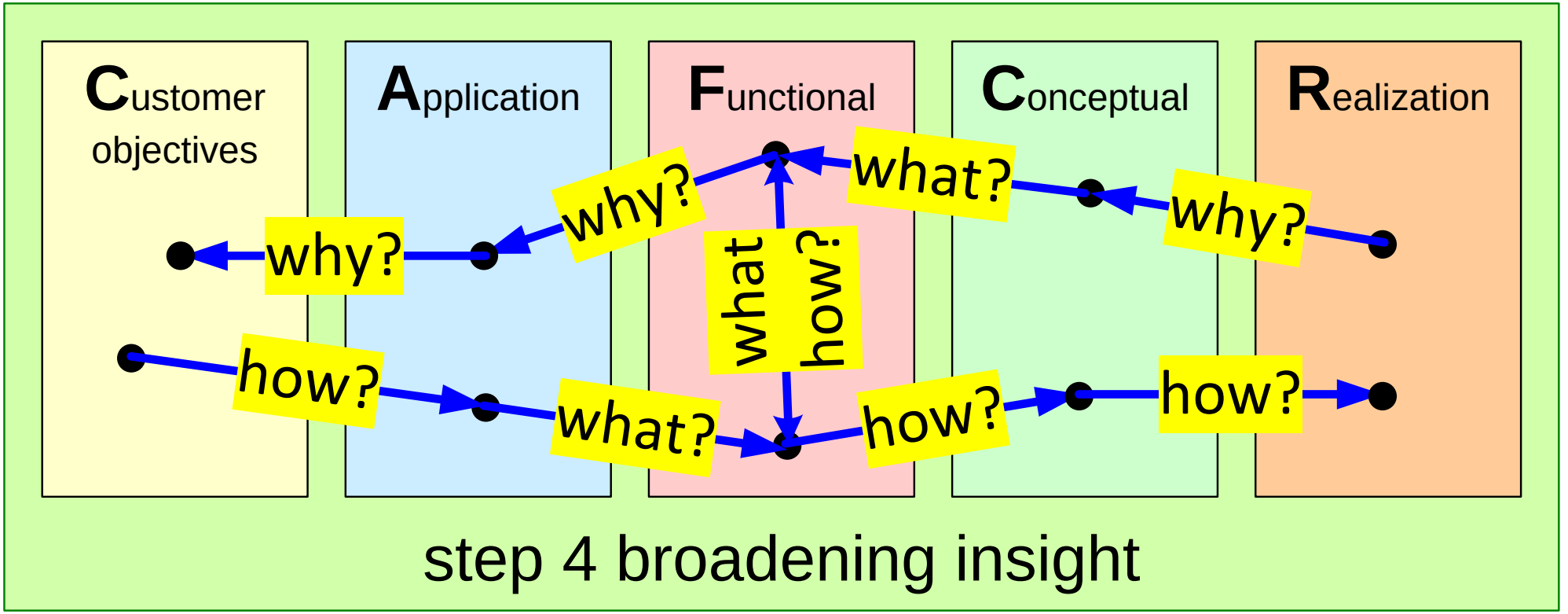
Creating Insight



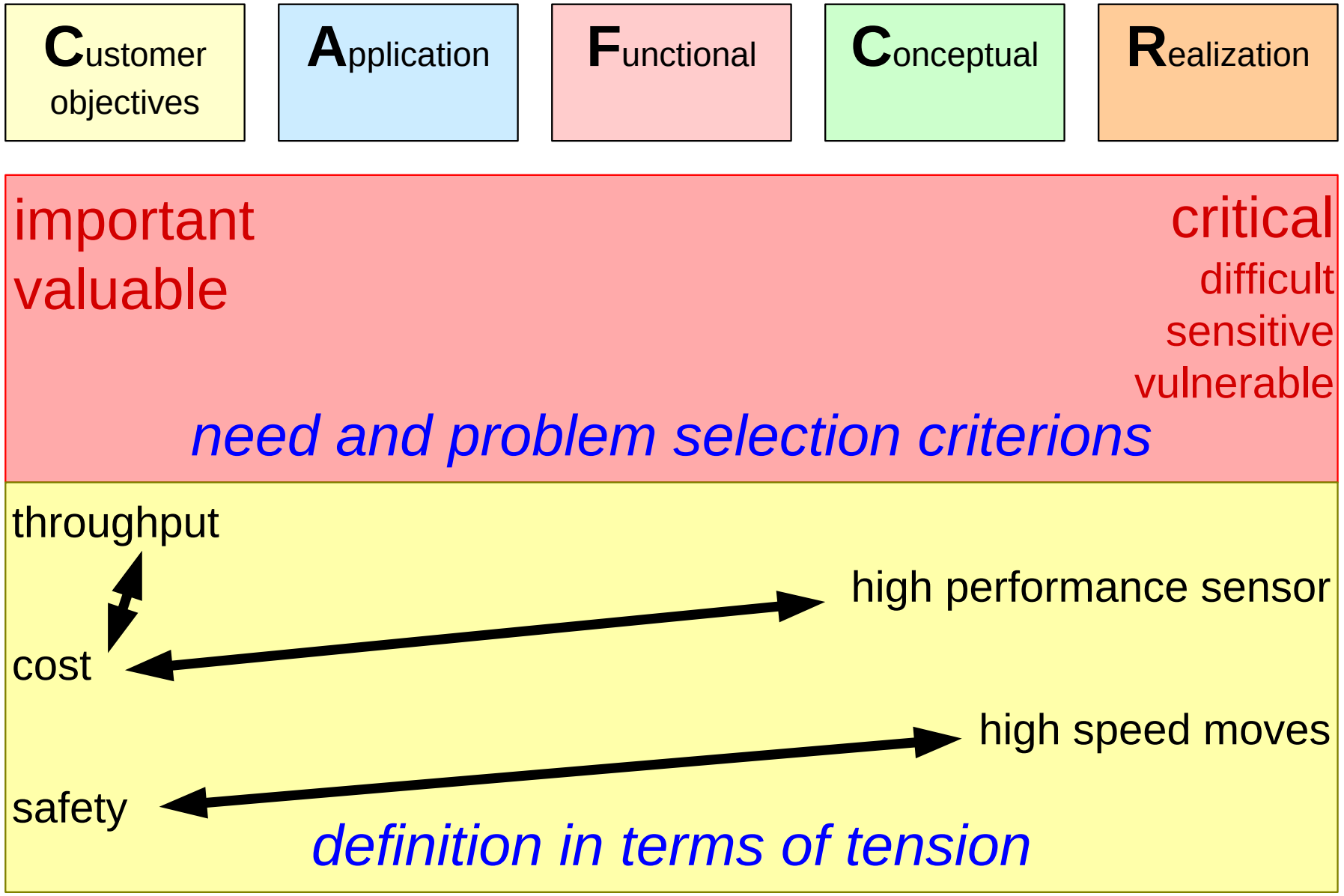
Deepening Insight



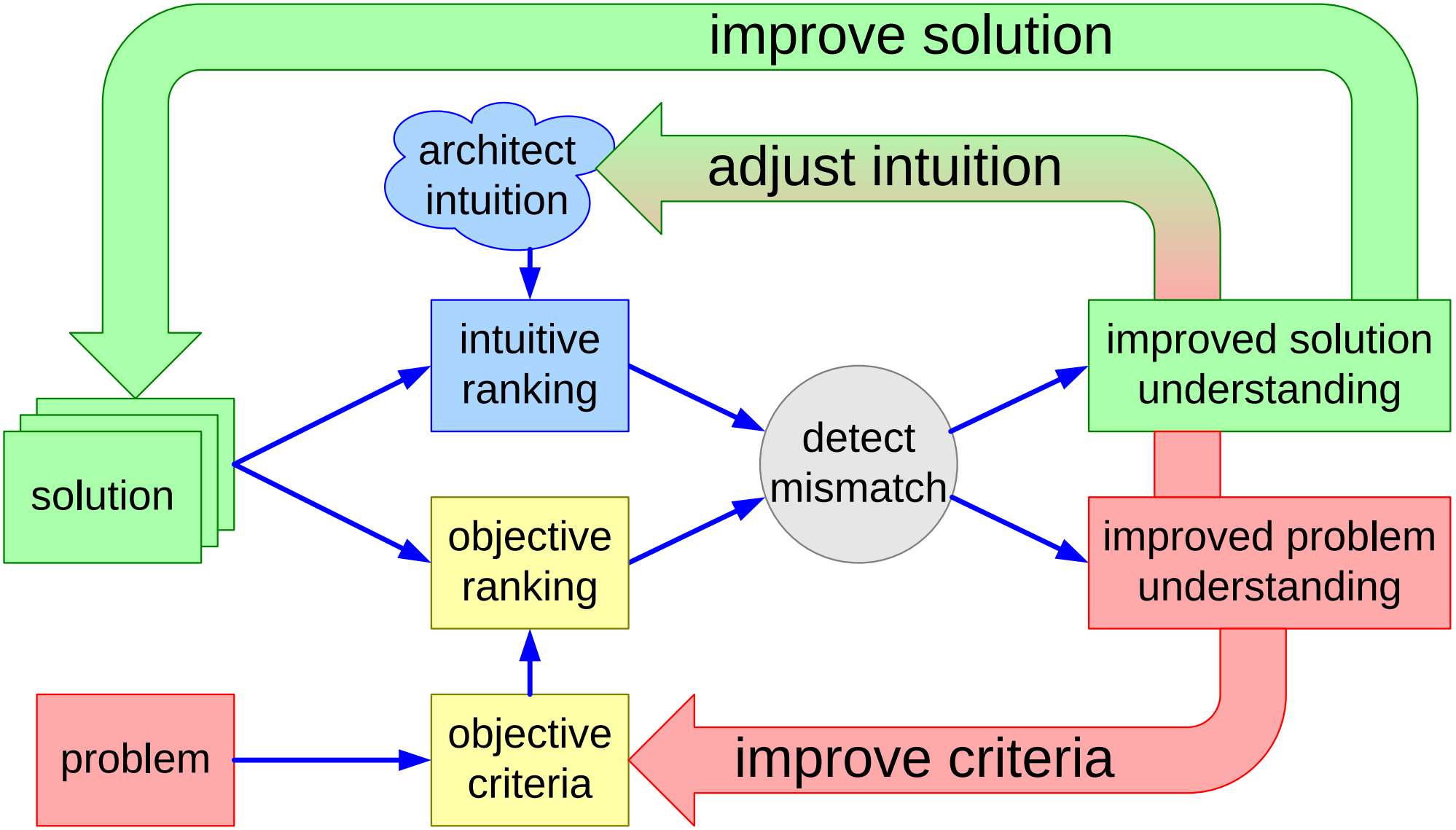
Broadening Insight



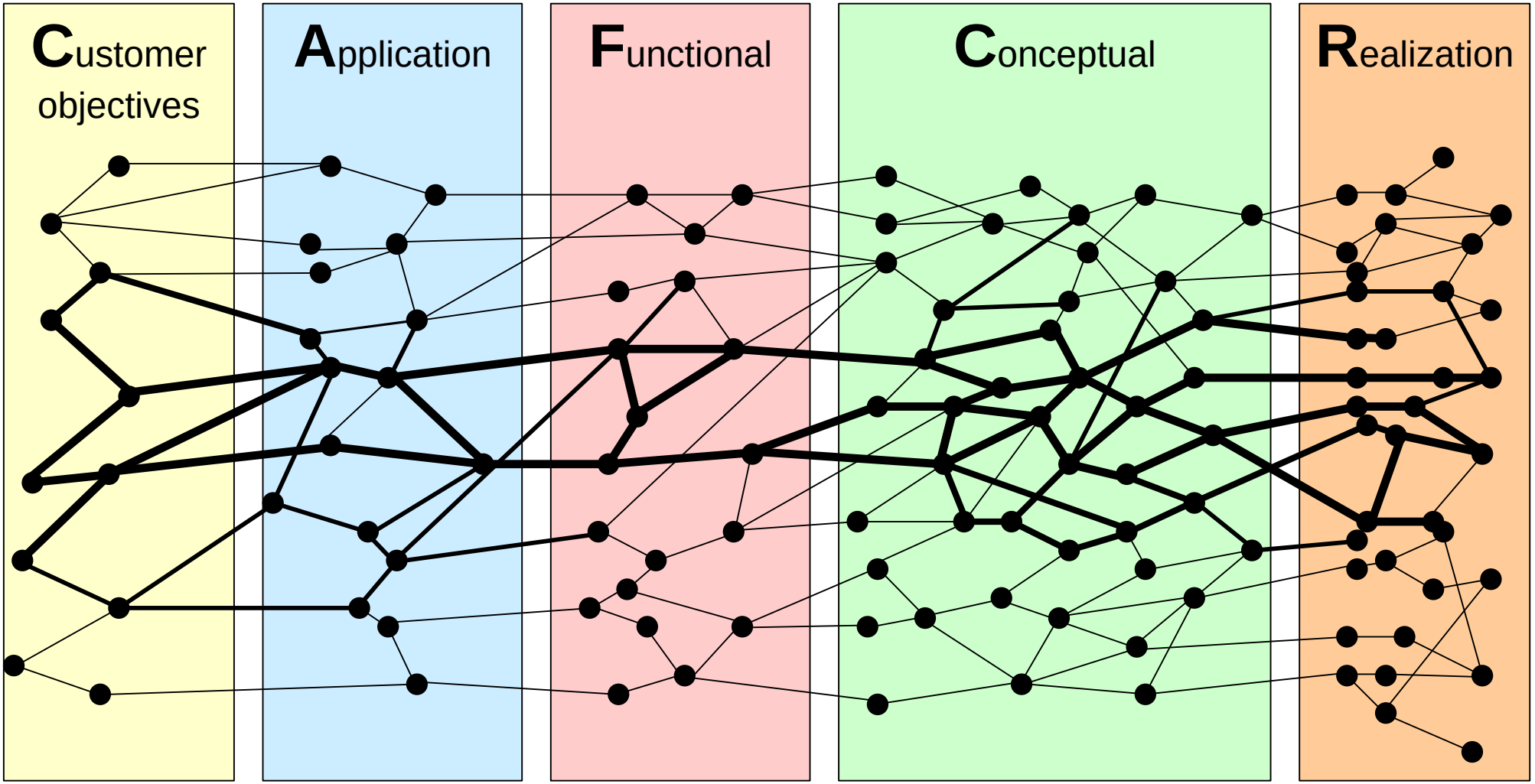
Problem identification and articulation



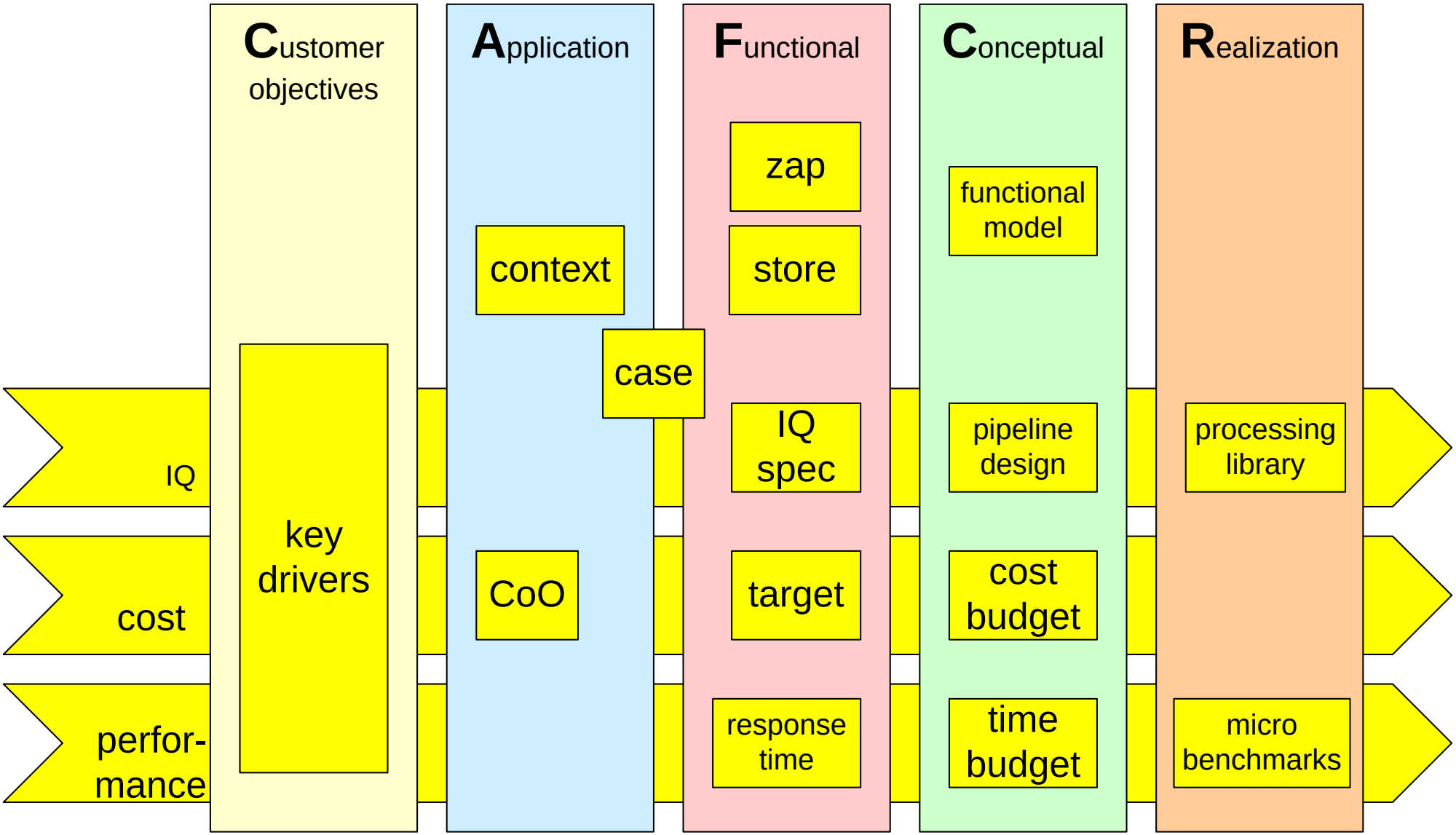
Iteration during the analysis



Thread of related issues



Documentation and communication structure



Threads of reasoning illustrated by medical imaging case

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

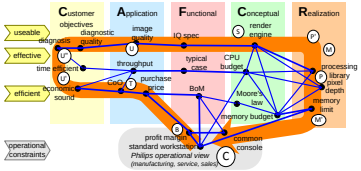
Abstract

The medical imaging workstation case is introduced. An architecting method based on the CAFCR viewpoints is explained, consisting of 4 elements:

- the CAFCR viewpoints
- qualities as integrating needles
- story telling
- threads of reasoning

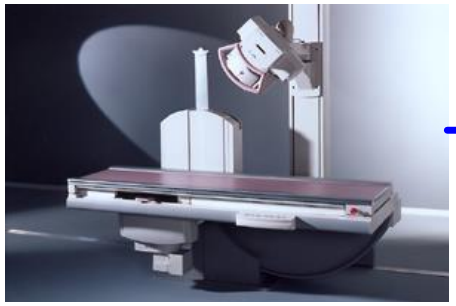
A thread of reasoning is build up in steps, based on this case. The underlying reasoning is explained.

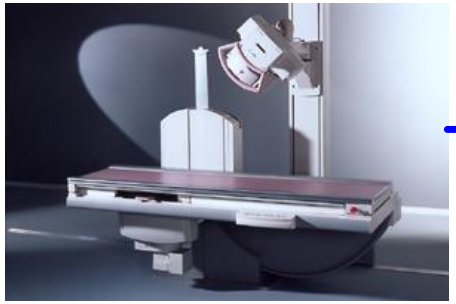
October 11, 2020
status: preliminary draft
version: 0

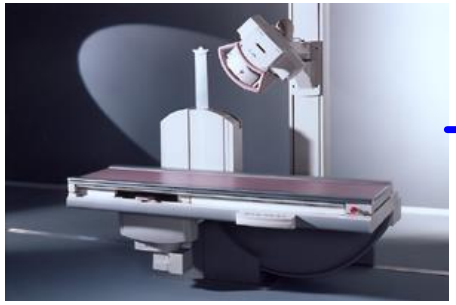


cost revisited in context of clinical needs and realization constraints; note: original threads are significantly simplified

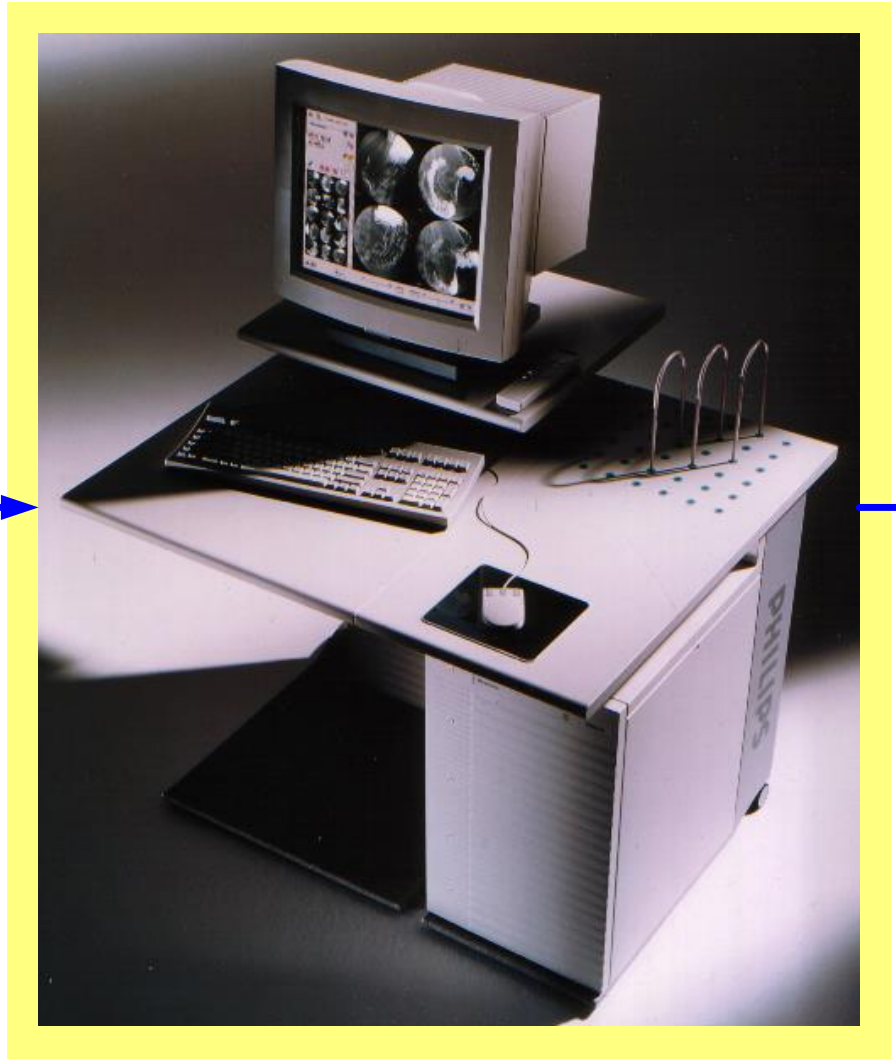
Easyvision serving three URF examination rooms







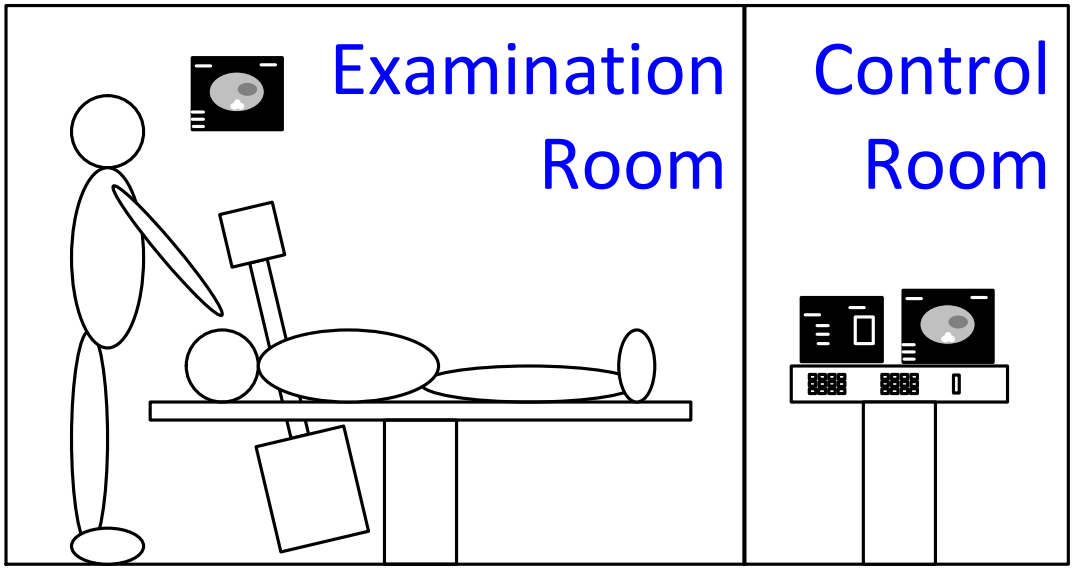
URF-systems

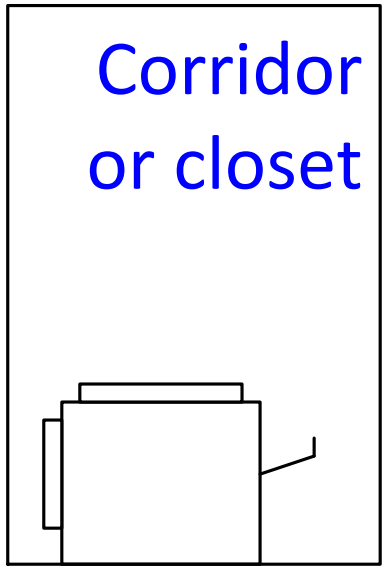


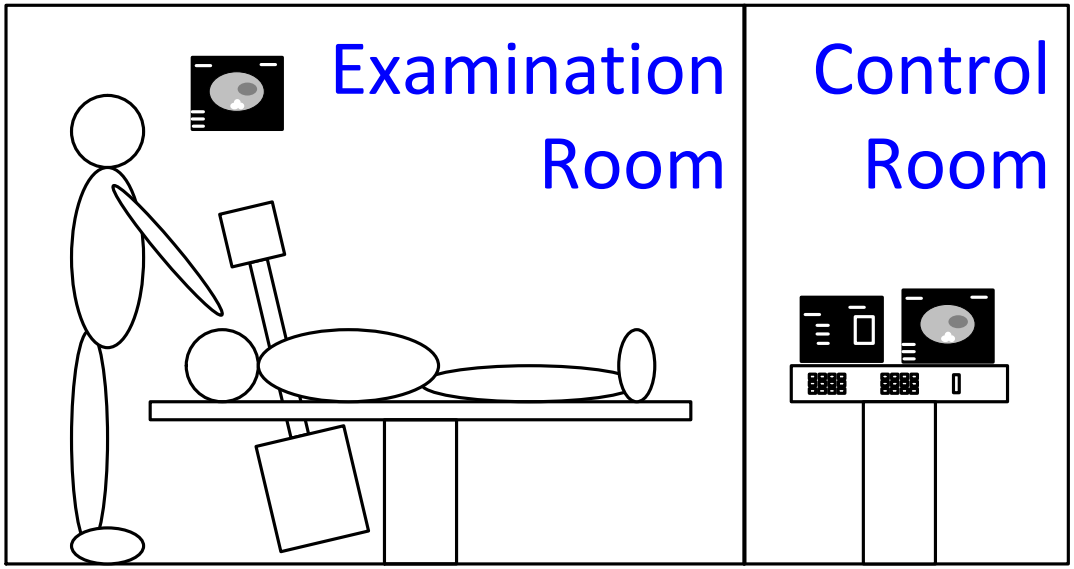
EasyVision: Medical Imaging Workstation

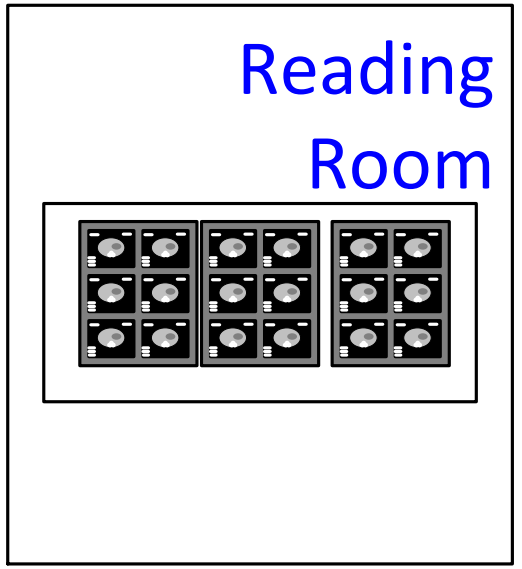
typical clinical image (intestines)

X-ray rooms from examination to reading around 1990

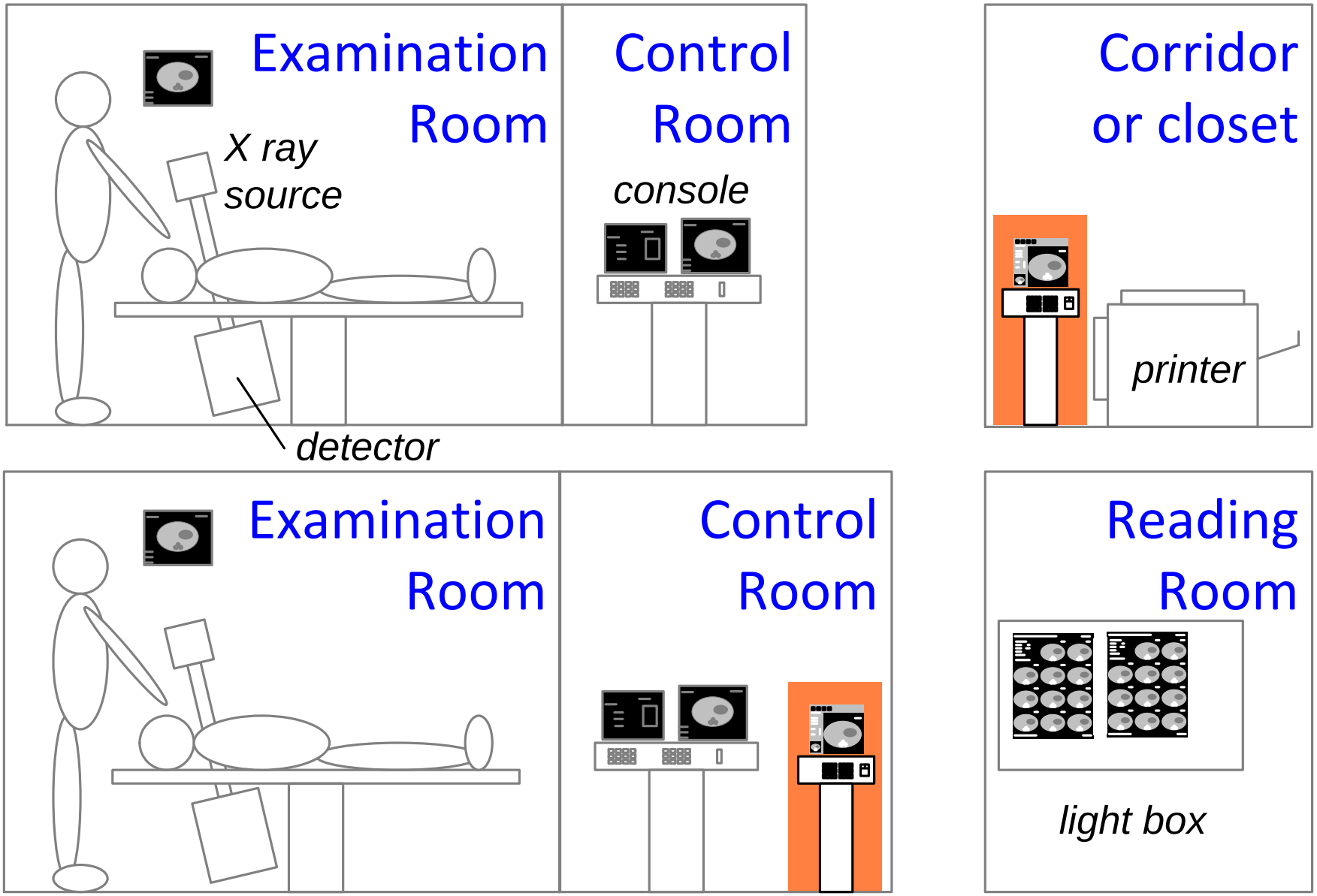




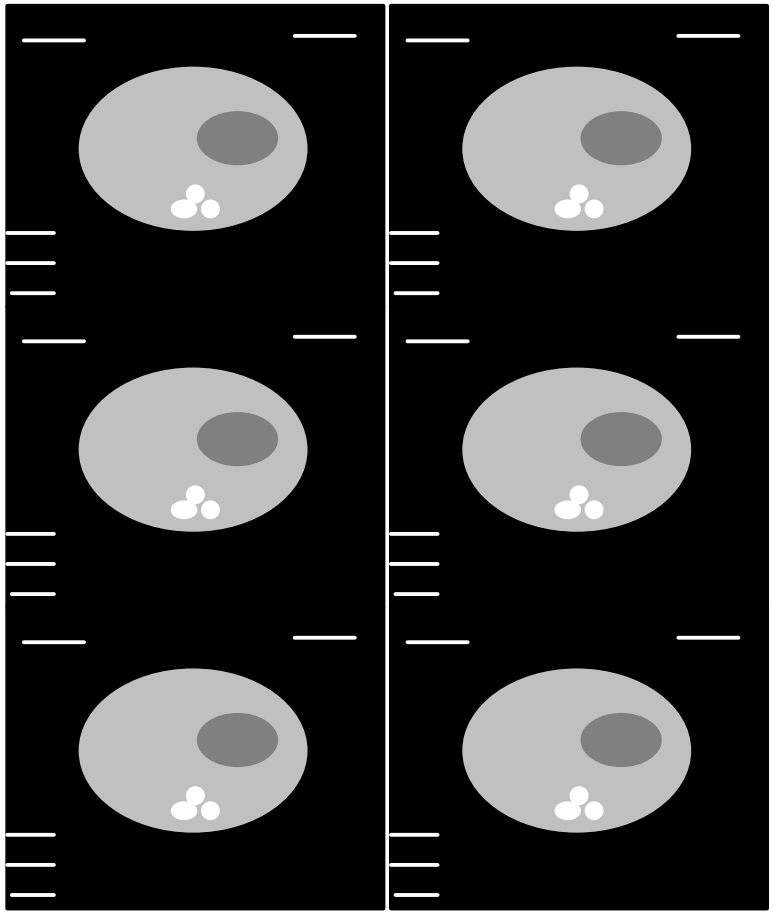




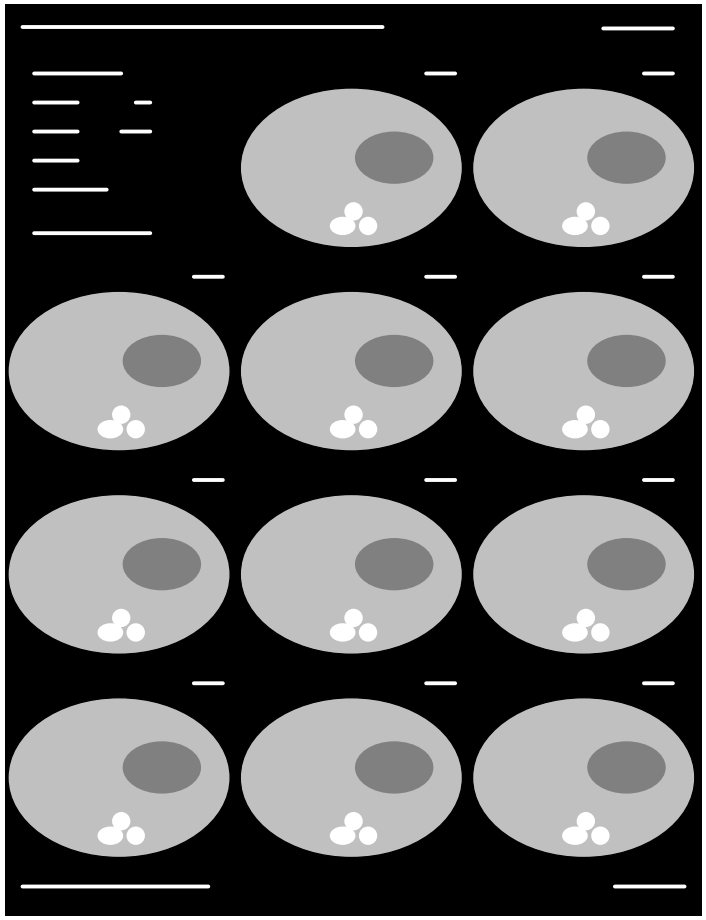
X-ray rooms with Easyvision applied as printserver



Comparison screen copy versus optimized film



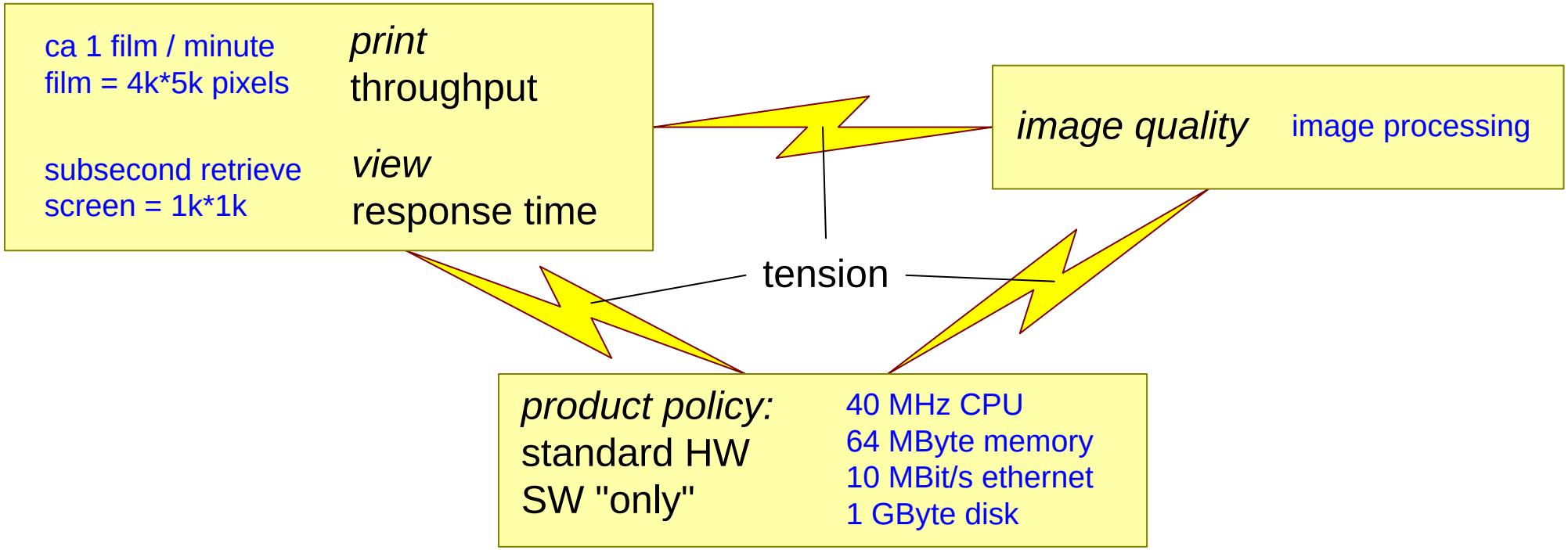
old: screen copy



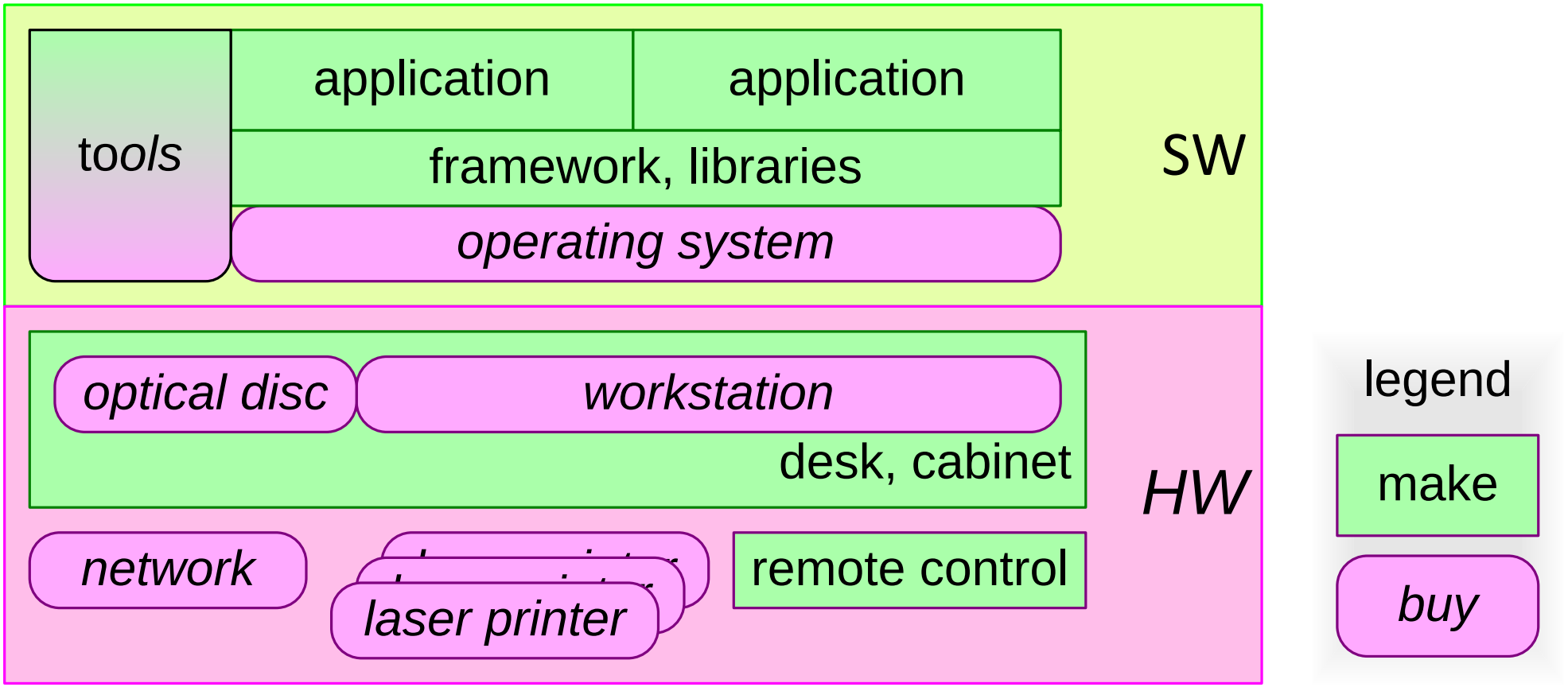
new: SW formatting

20 to 50% less film needed

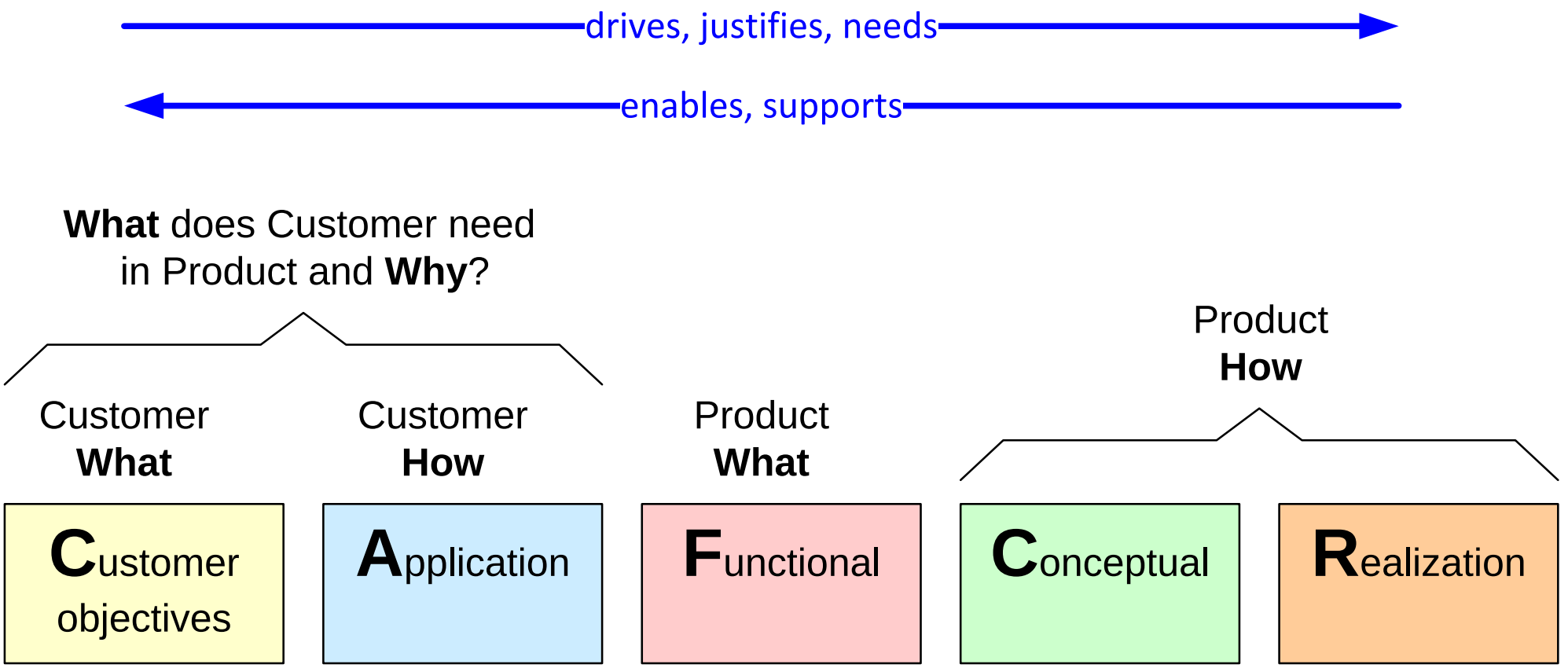
Challenges for product creation



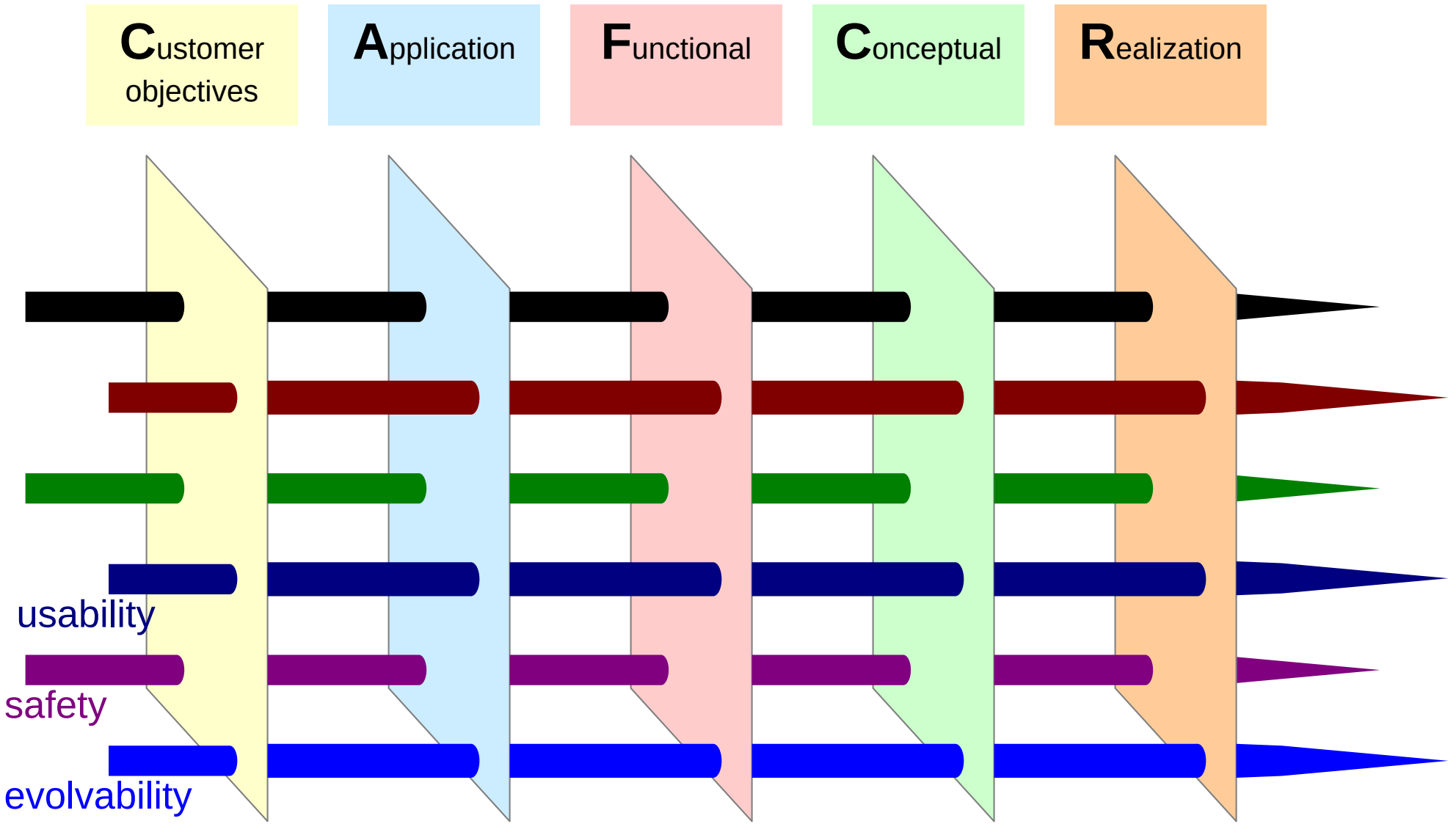
Top level decomposition



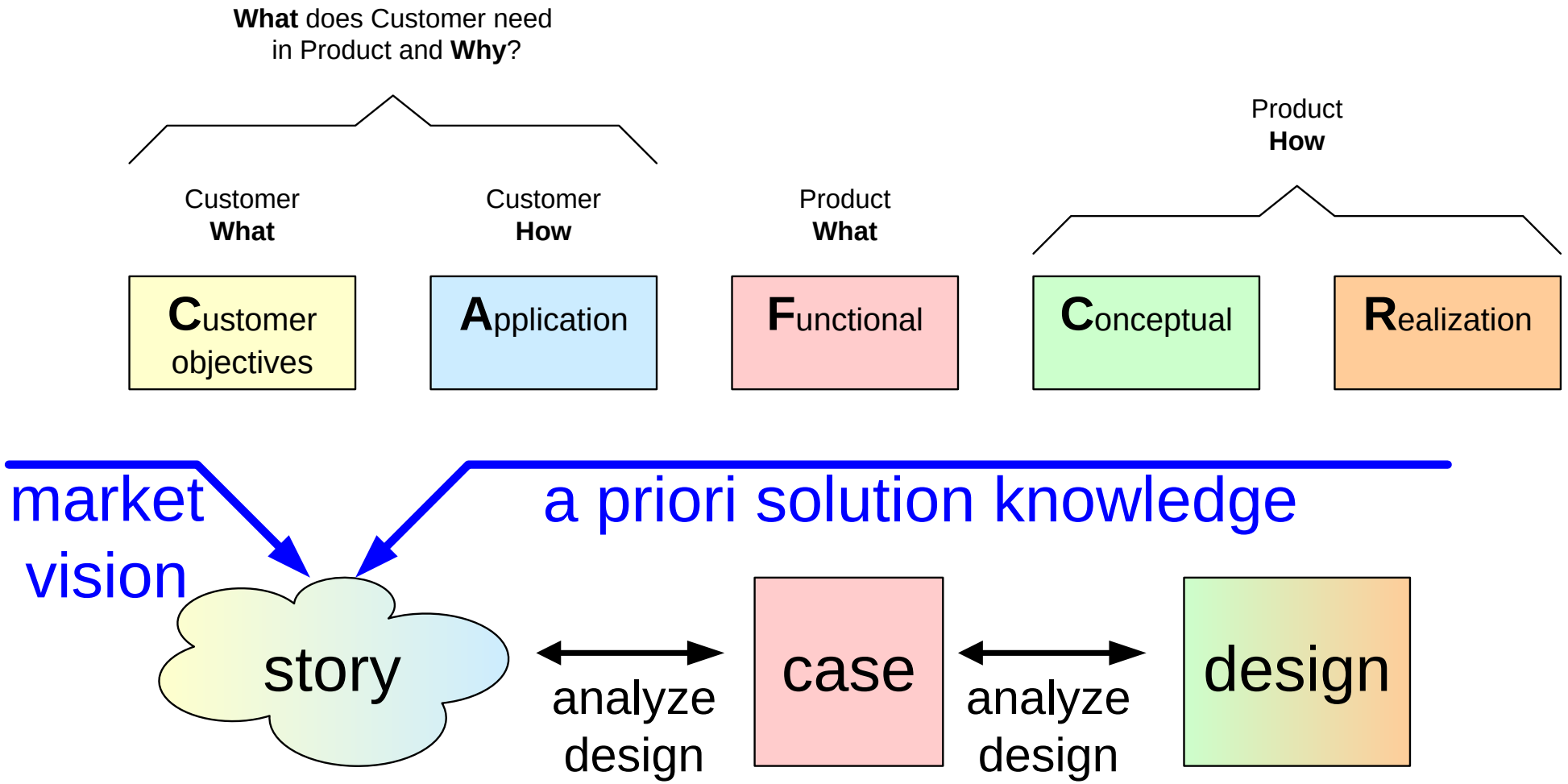
CAFCR viewpoints



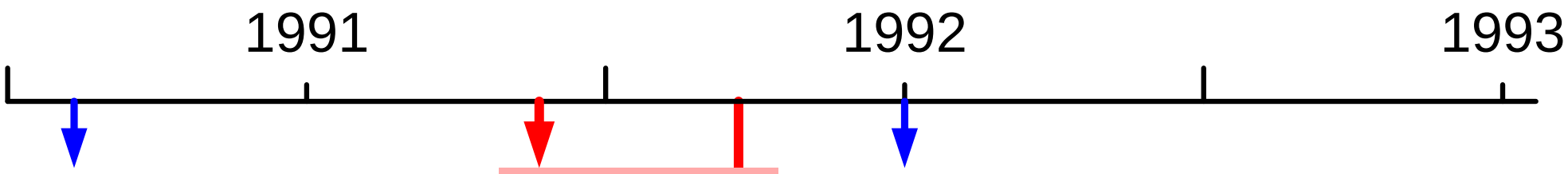
Quality needles as generic integrating concepts



From story to design



Chronology of Easyvision RF R1 development



basic application
toolboxes
100 kloc

interactive viewing

performance
problems

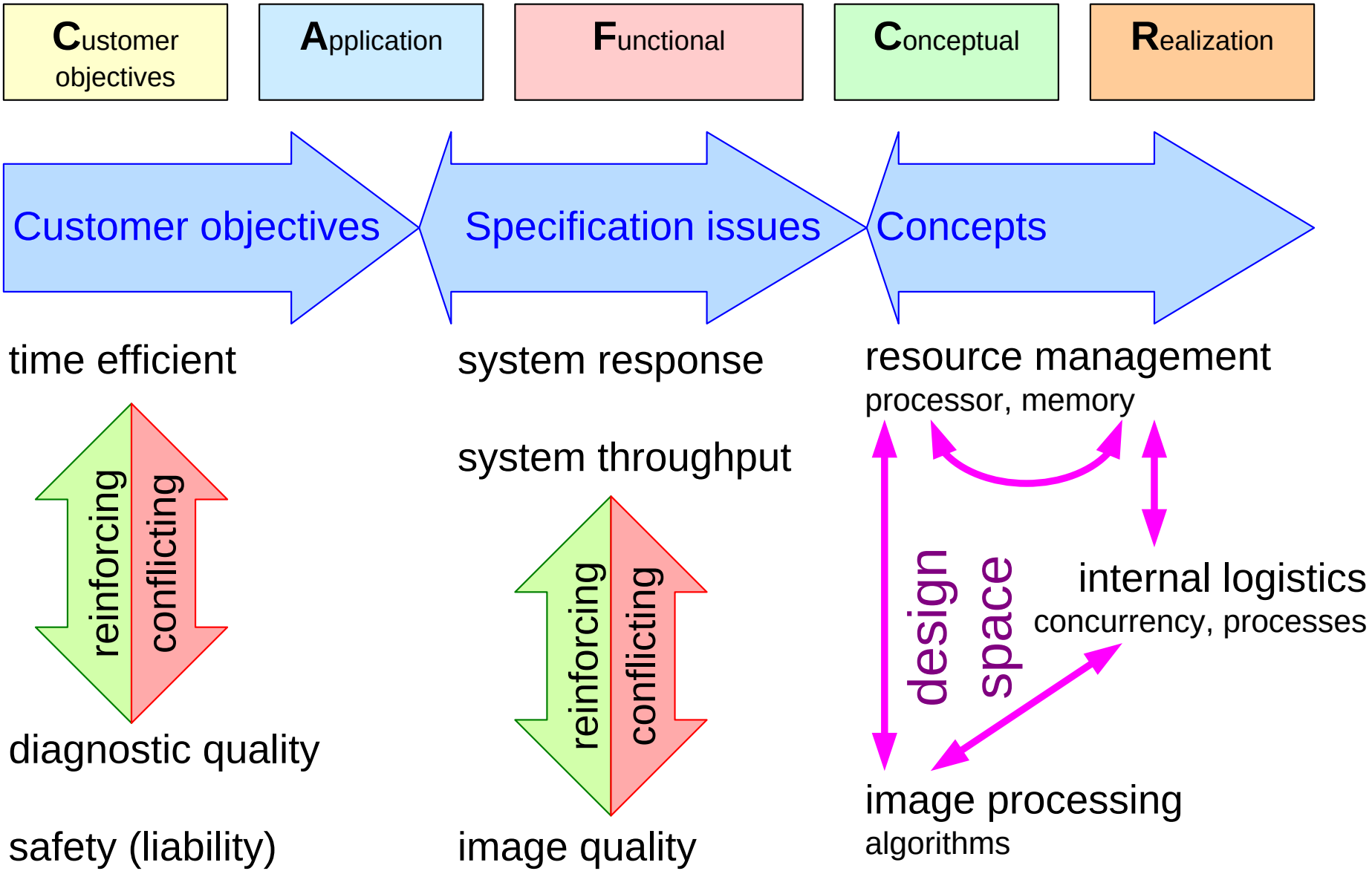
IQ
problems

Easyvision RF
integrated product
360 kloc

print server +
communication +
interactive viewing

marketing opinion:
"All the functionality is available,
we only have to provide a clinical UI"

Thread of reasoning based on efficiency-quality tension

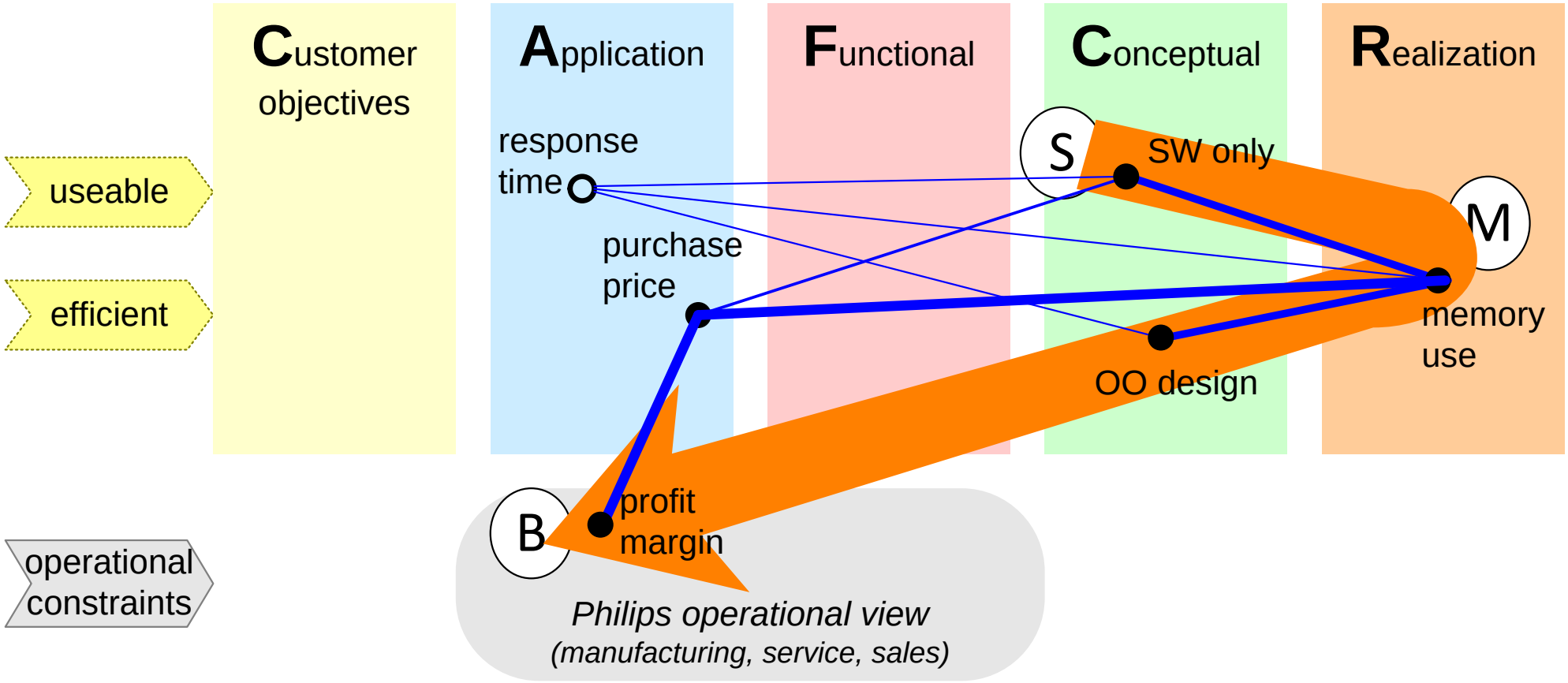


Technology innovations

performance
cost

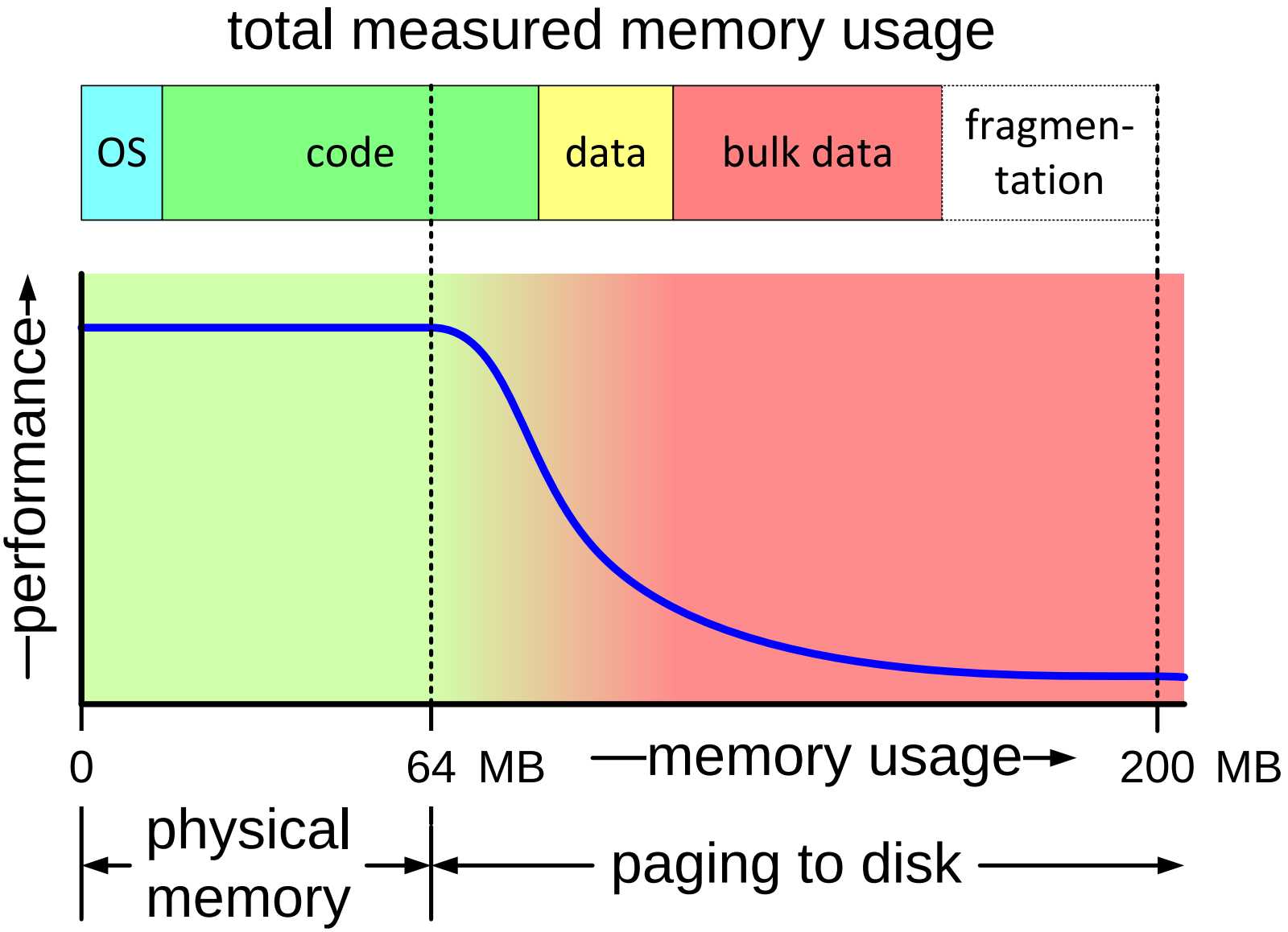
- standard UNIX based workstation
- full SW implementation, more flexible
- object oriented design and implementation (Objective-C)
- graphical User Interface, with windows, mouse etcetera
- call back scheduling, fine-grained notification
- data base engine, fast, reliable and robust
- extensive set of toolboxes
- property based configuration
- multiple coordinate spaces

Thread of reasoning; introvert phase

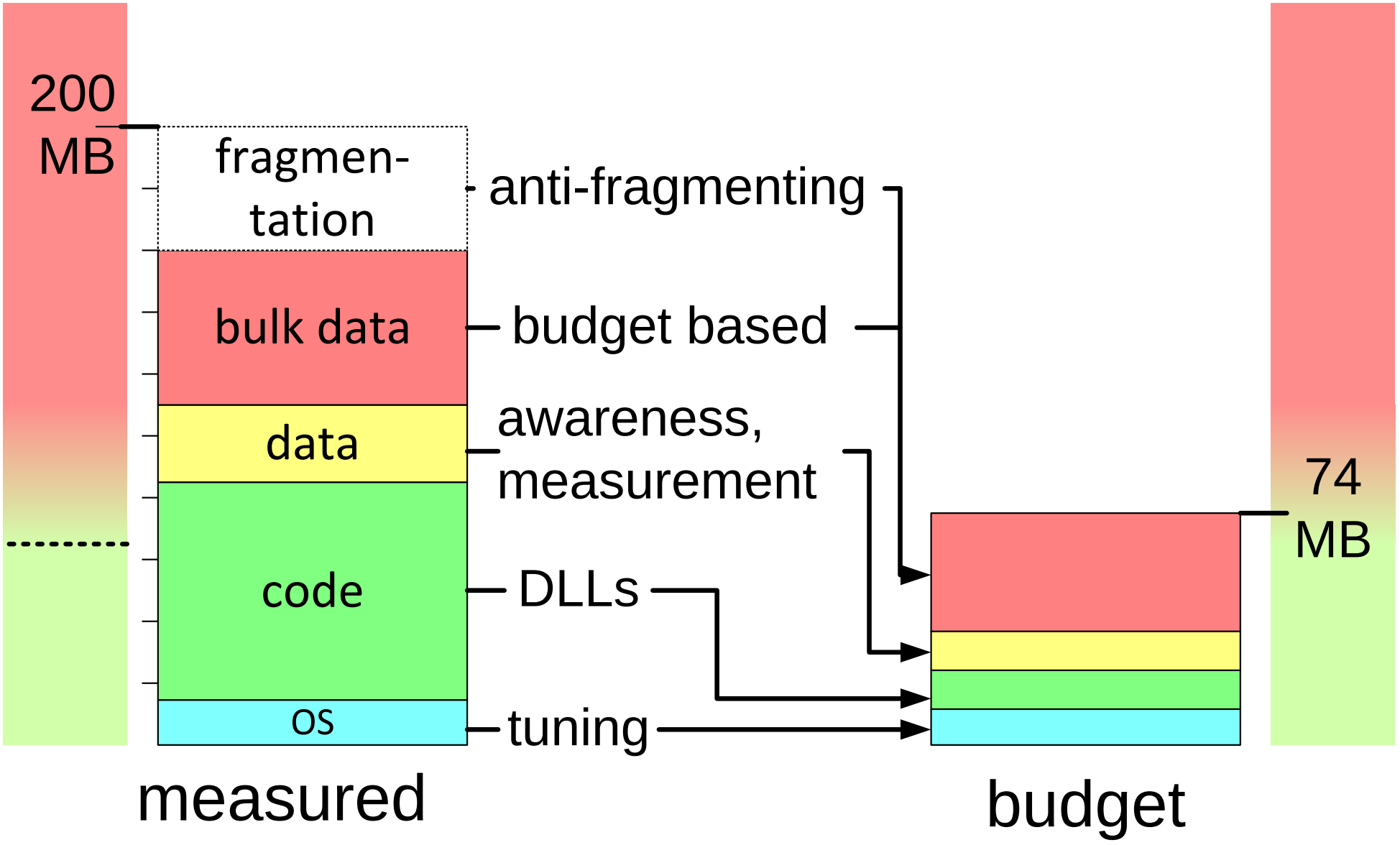


Introvert view: cost and impact of new technologies

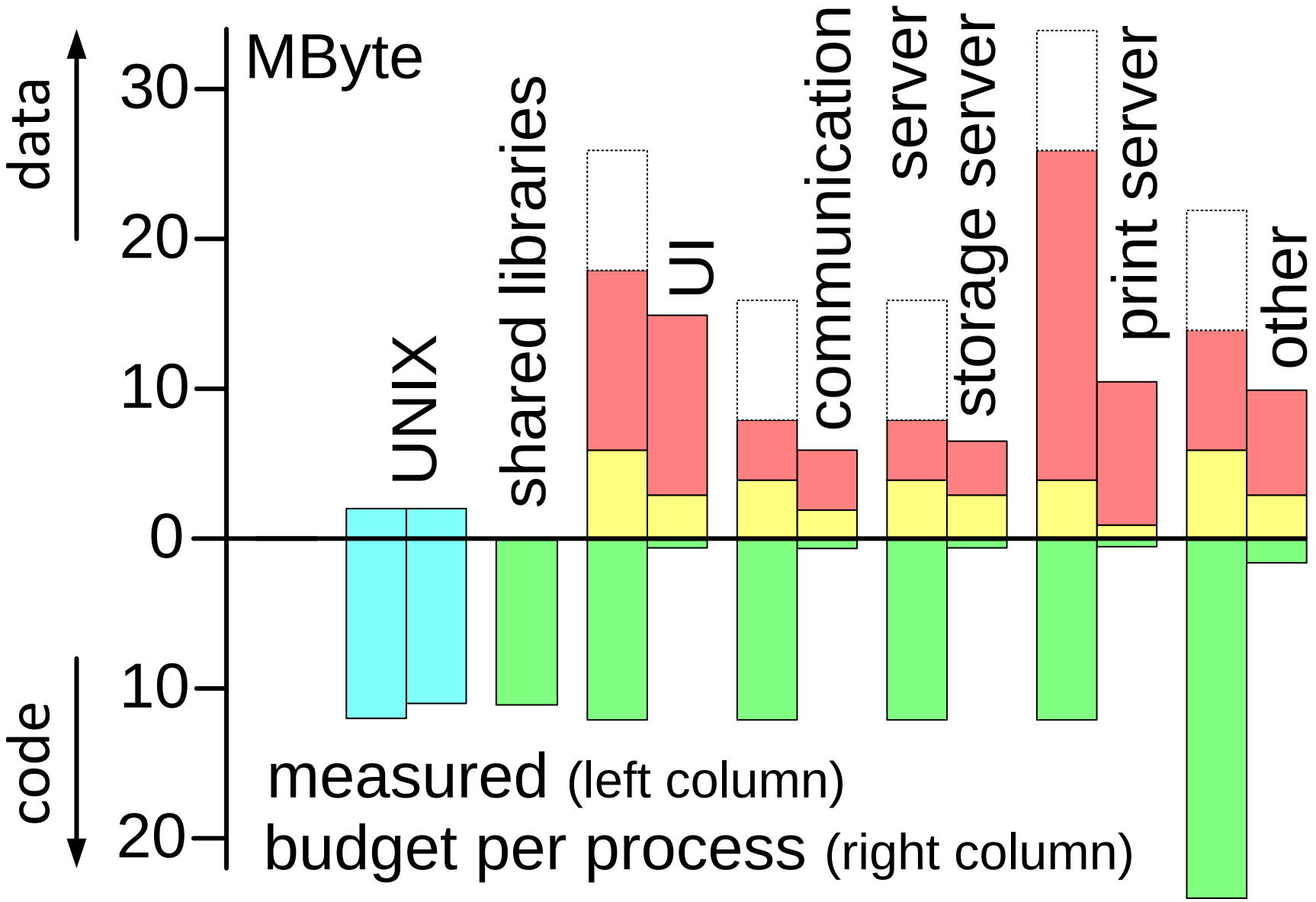
Memory usage half way R1



Solution of memory performance problem

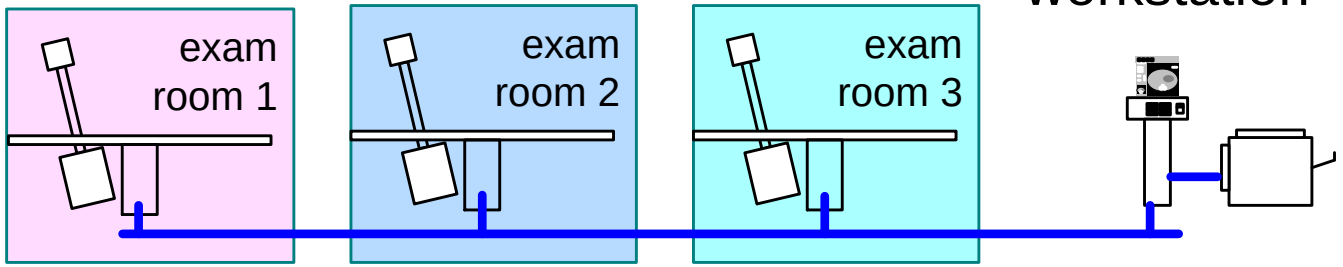


Visualization memory use per process



Typical case URF examination

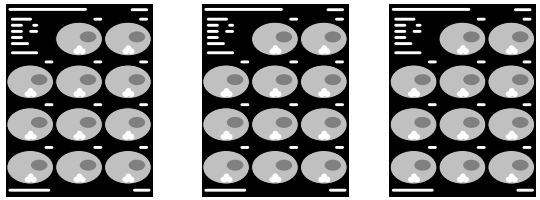
3 examination rooms connected to 1 medical imaging workstation + printer



examination room: average 4 interleaved examinations / hour

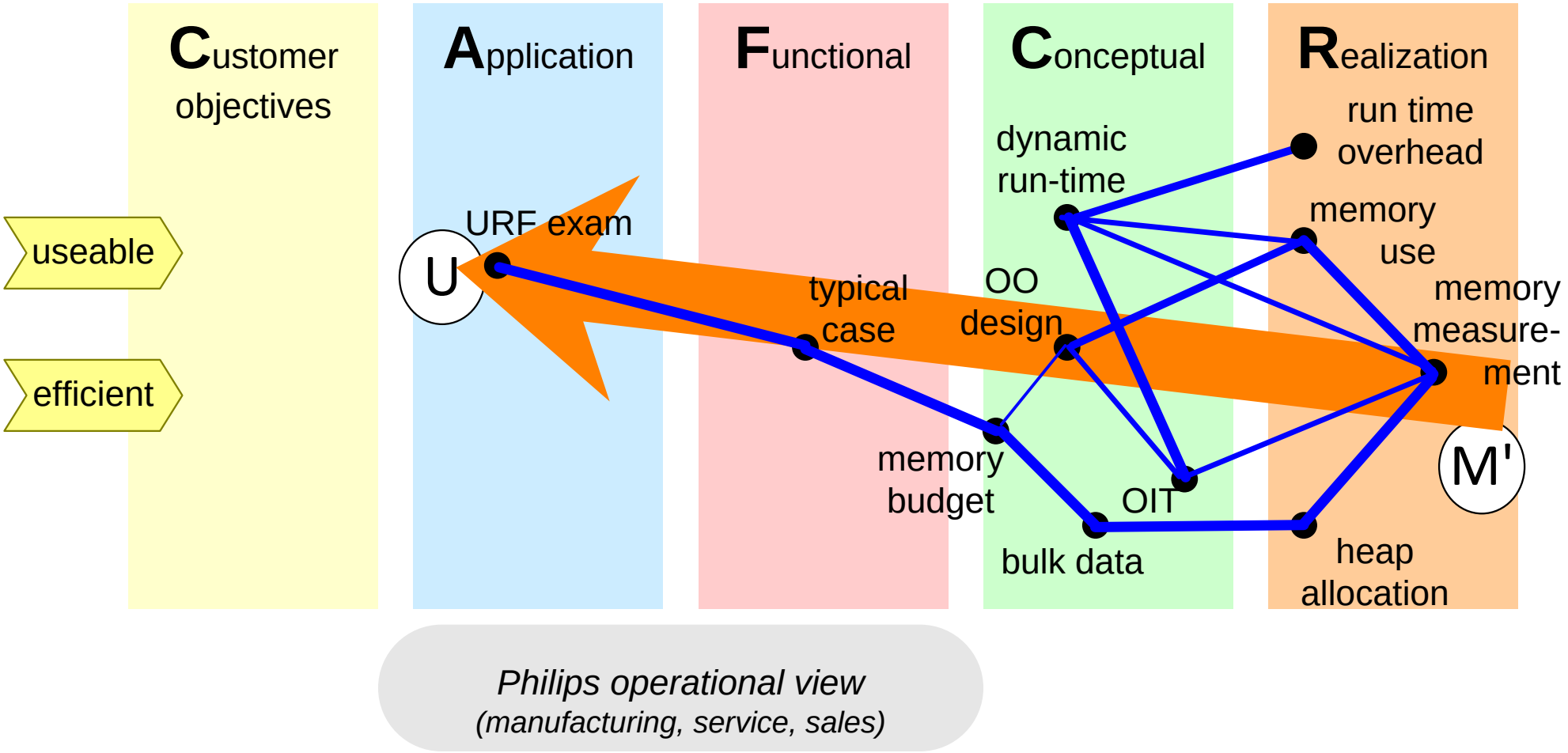
image production: 20 1024^2 8 bit images per examination

film production: 3 films of 4k*5k pixels each



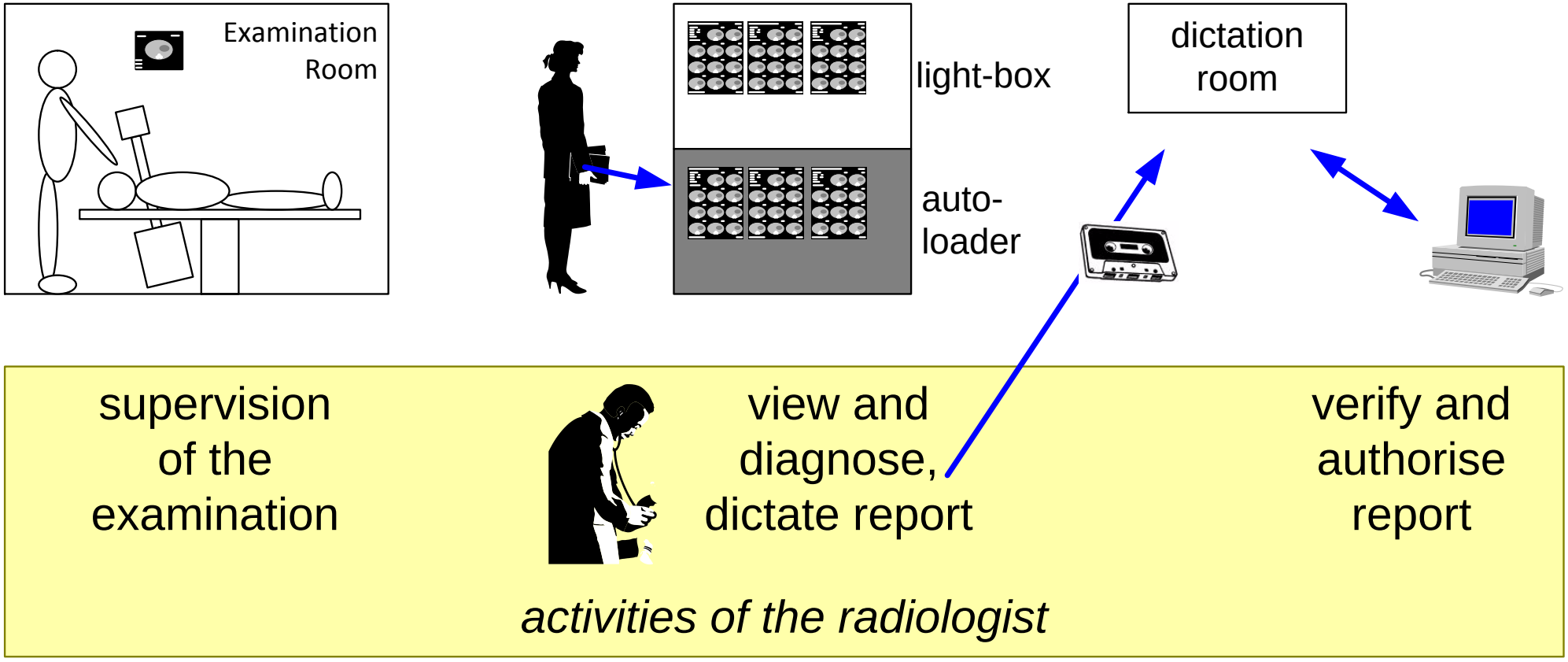
high quality output
(bi-cubic interpolation)

Thread of reasoning; phase 2

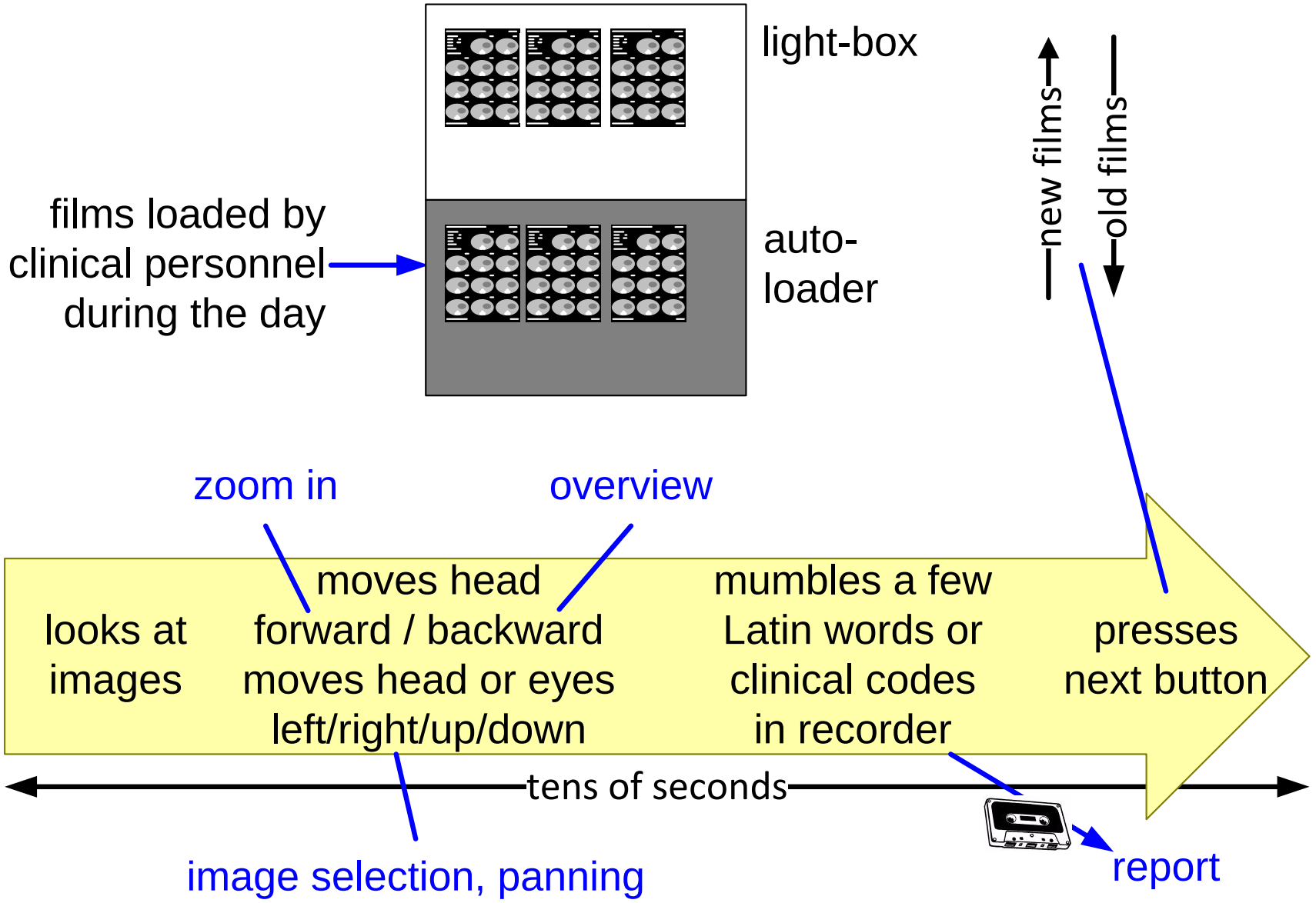


How to measure memory, how much is needed?
from introvert to extrovert

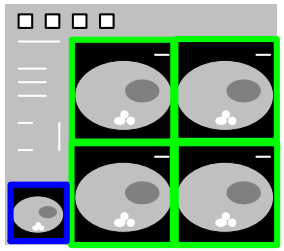
Radiologist workspots and activities



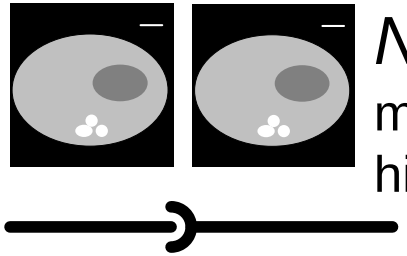
Diagnosis in tens of seconds



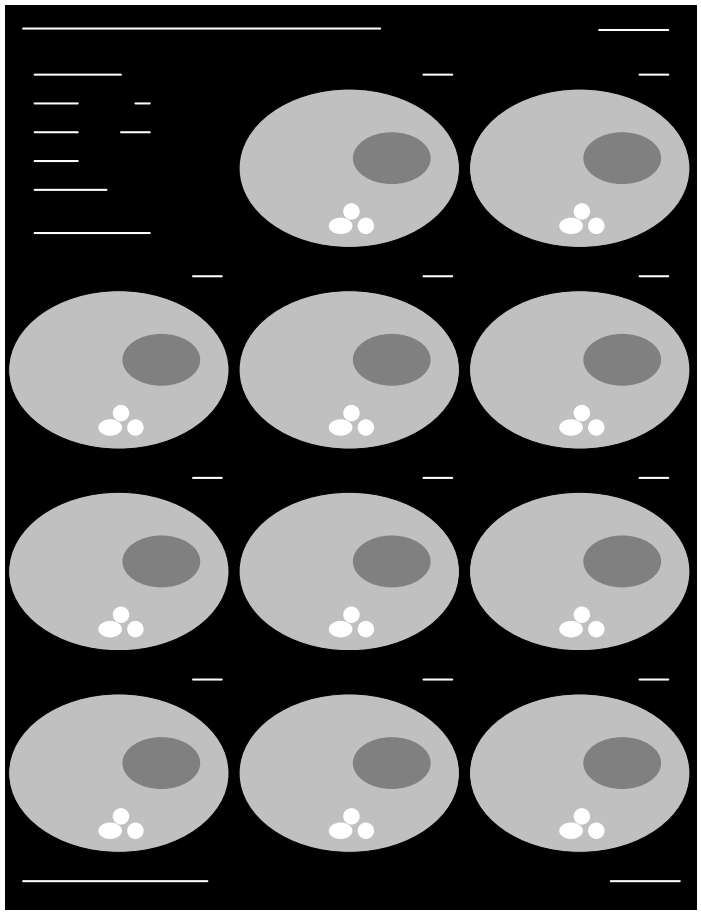
Rendered images at different destinations



Screen:
low resolution
fast response

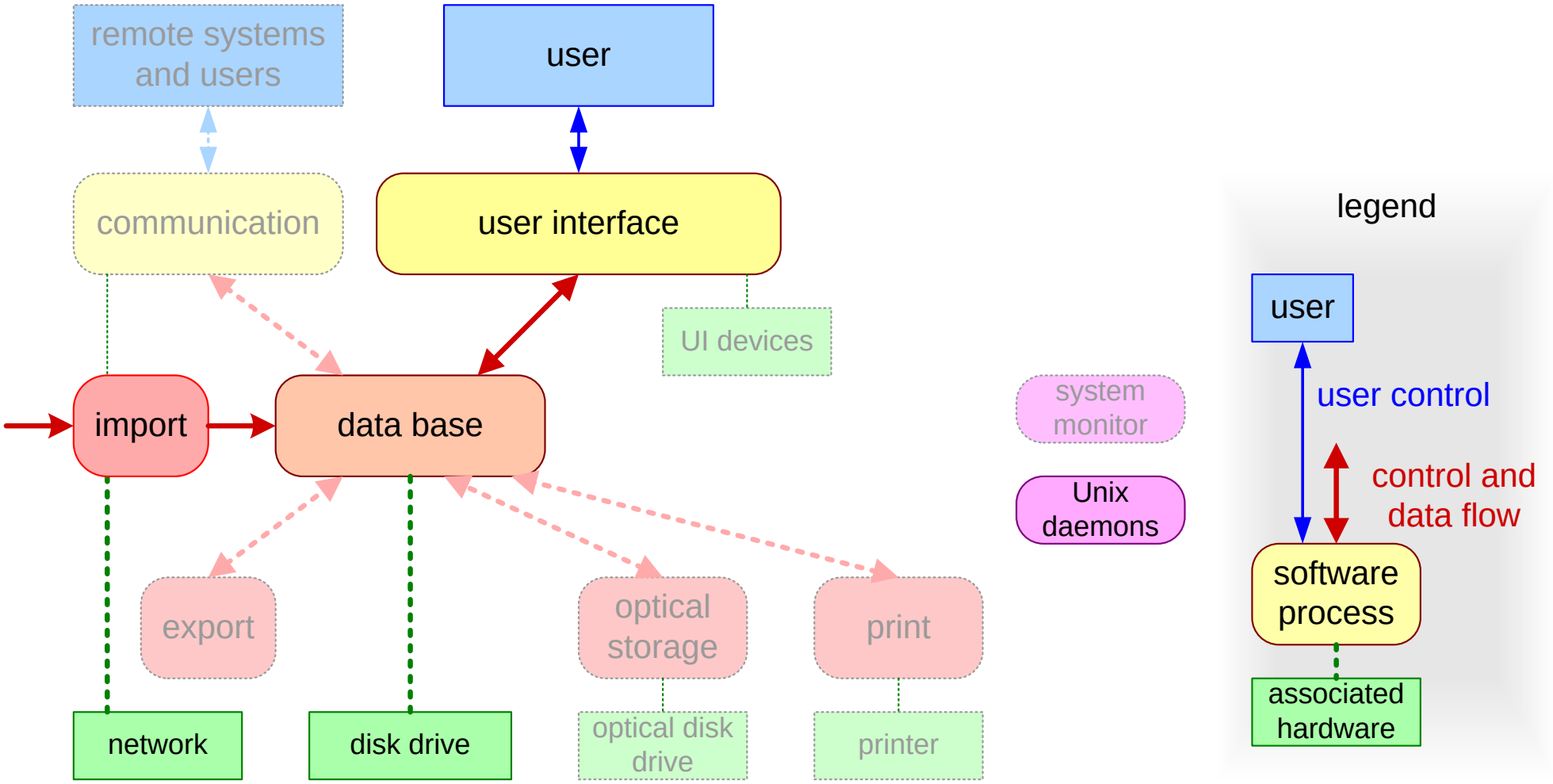


Network:
medium resolution
high throughput

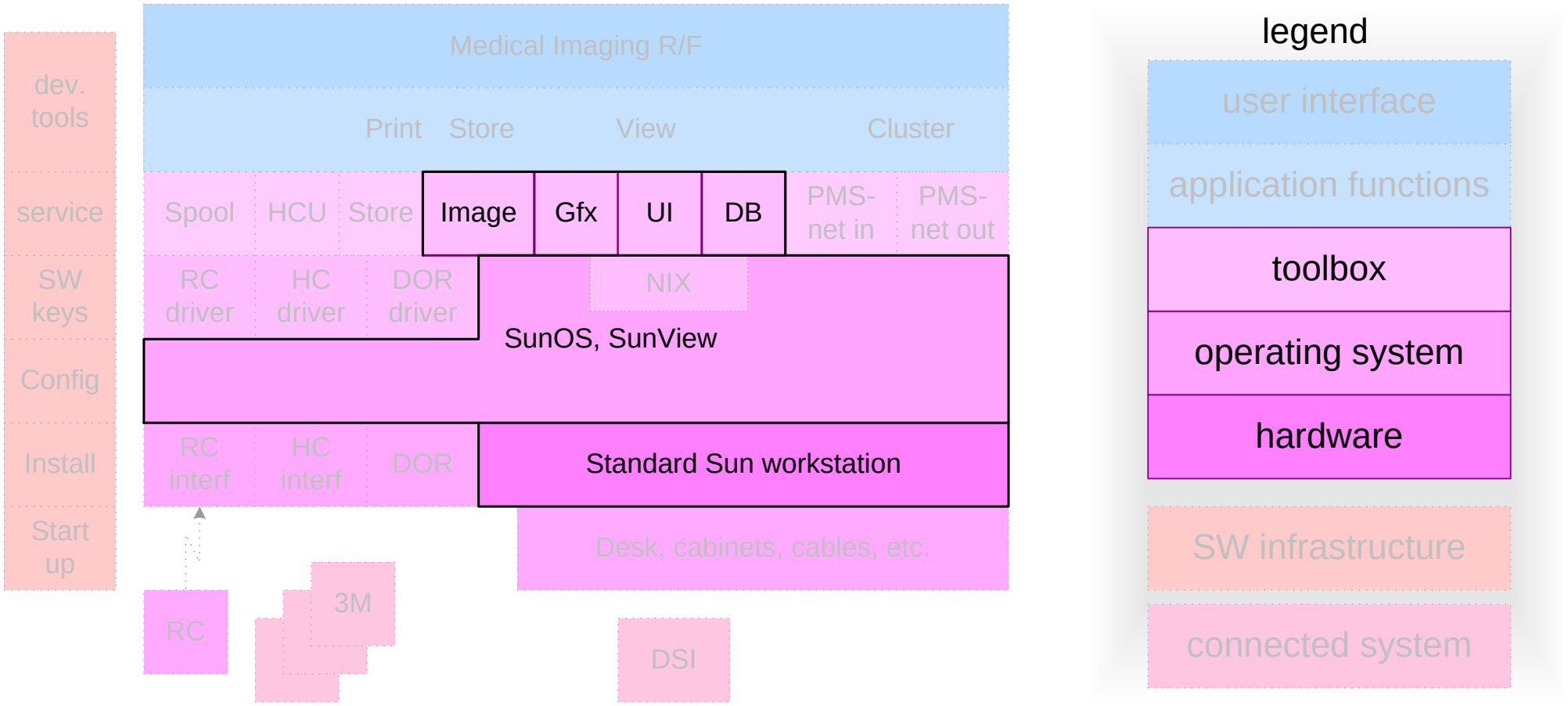


Film:
high resolution
high throughput

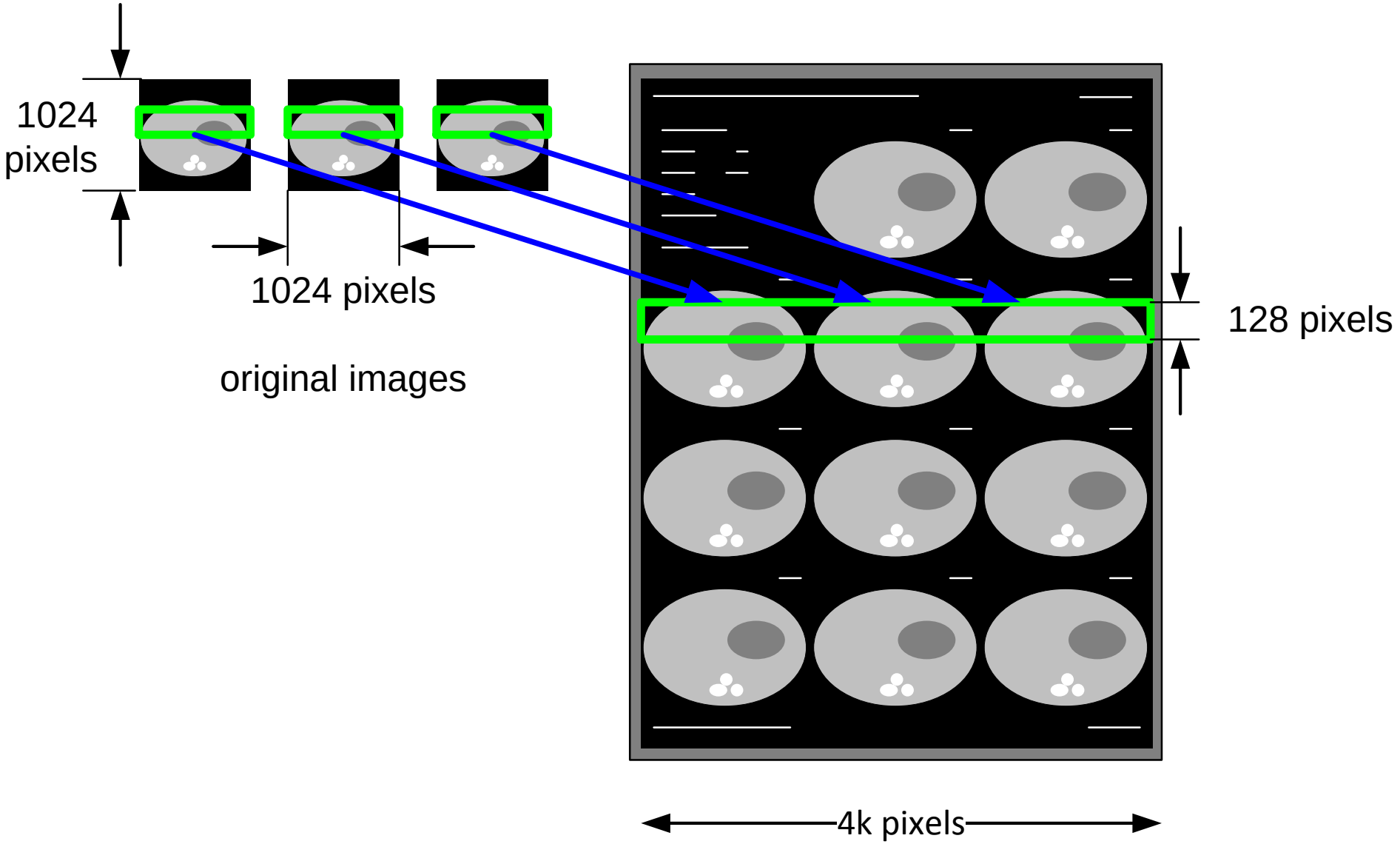
SW Process structure 1991



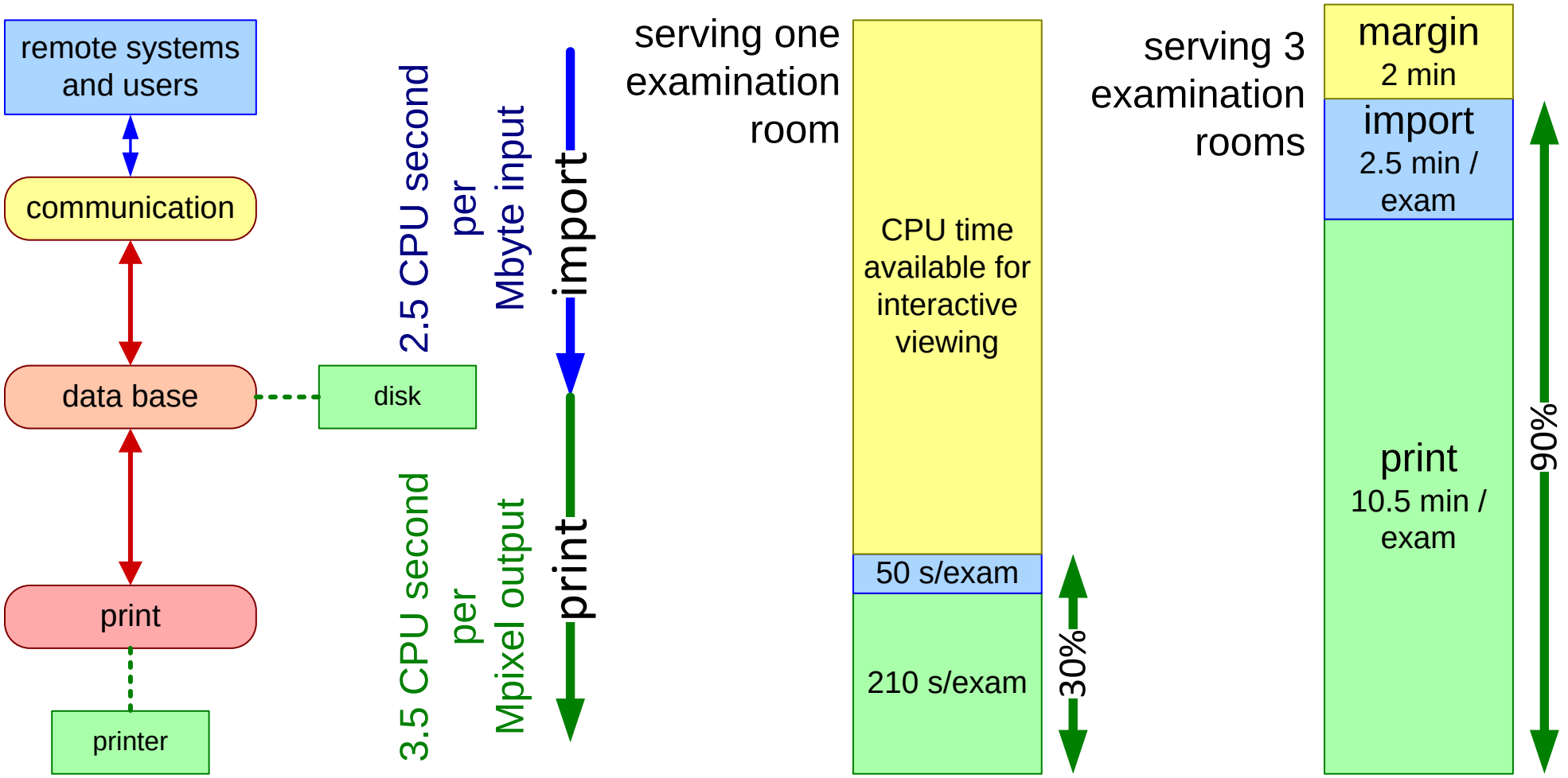
SW layers 1991



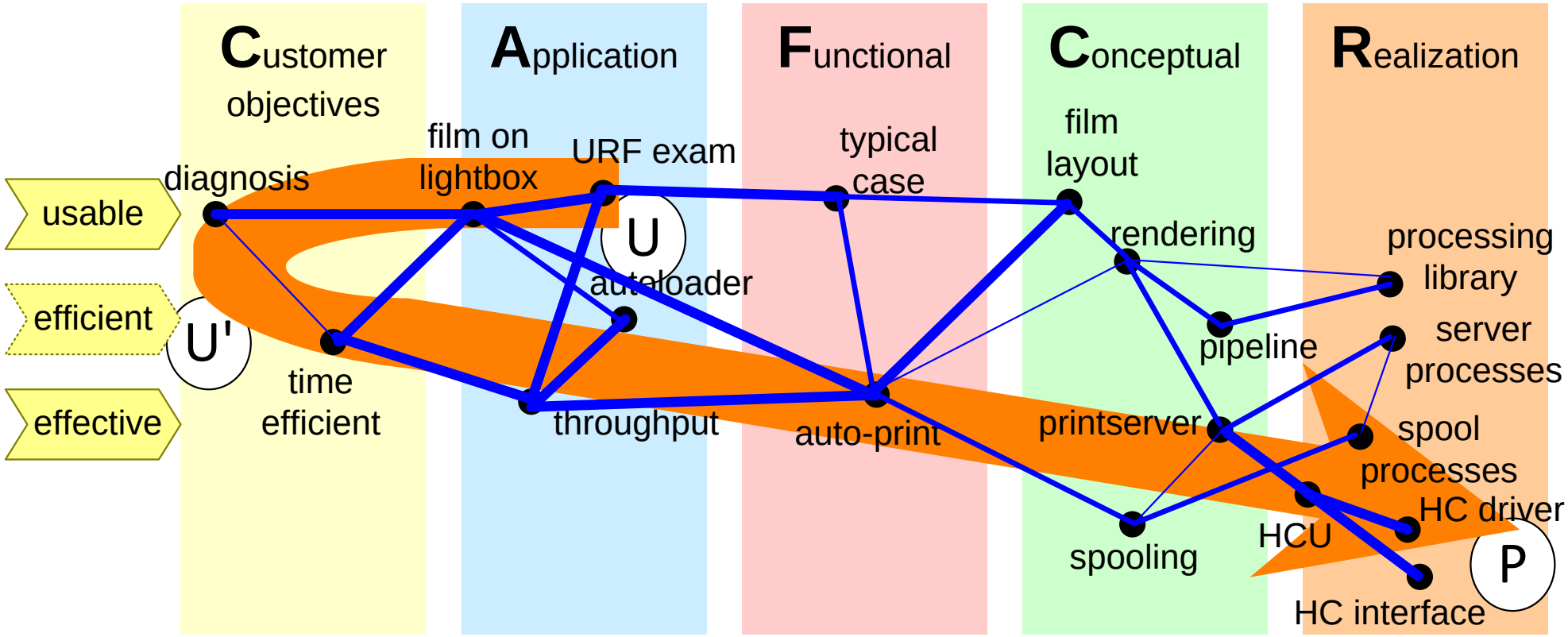
Print server is based on banding



Server CPU load



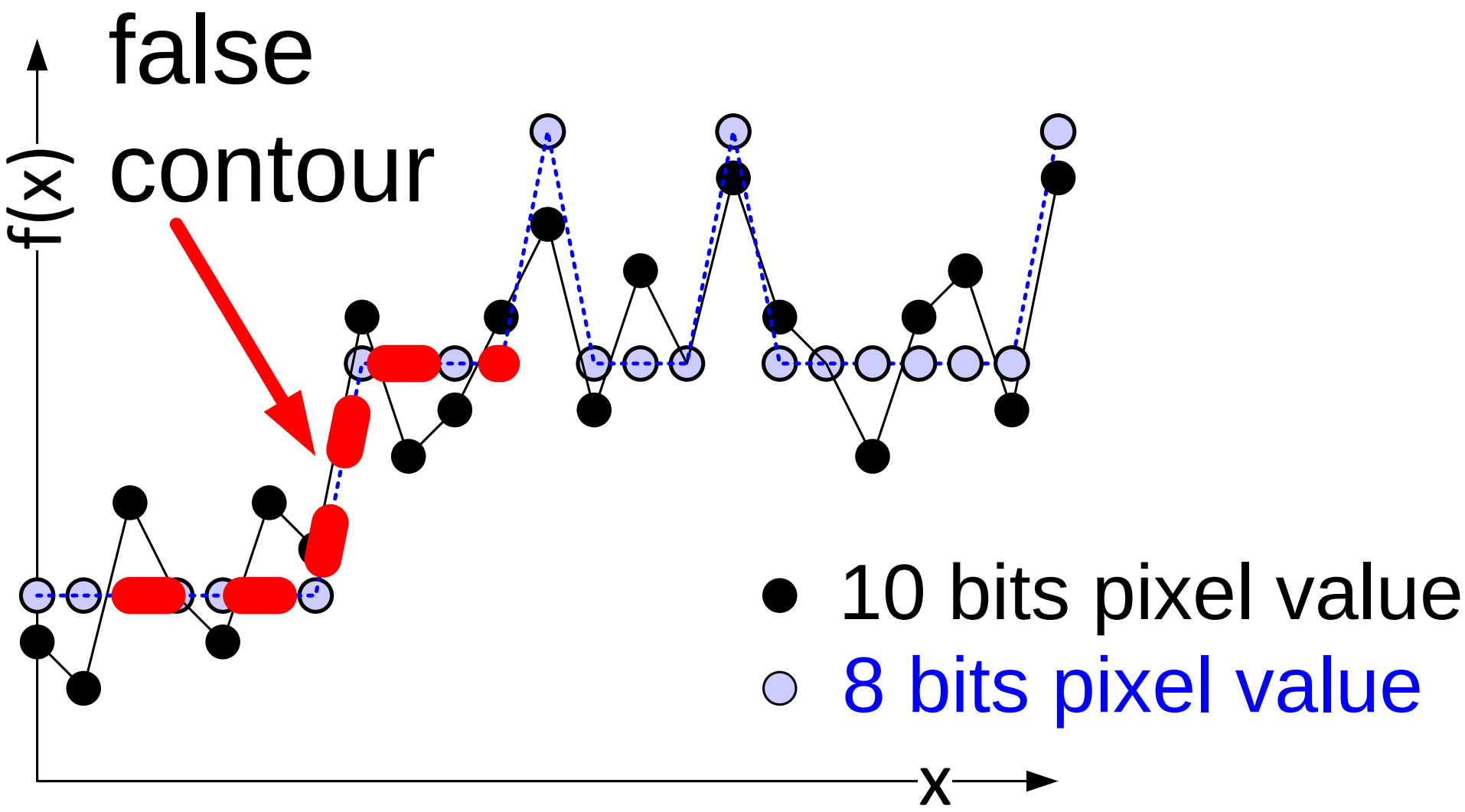
Thread of reasoning; phase 3



Philips operational view
(manufacturing, service, sales)

Radiologists diagnose from film, throughput is important
Extrovert view shows conceptual and realization gaps!

Image quality and safety problem



Presentation pipeline for X-ray images

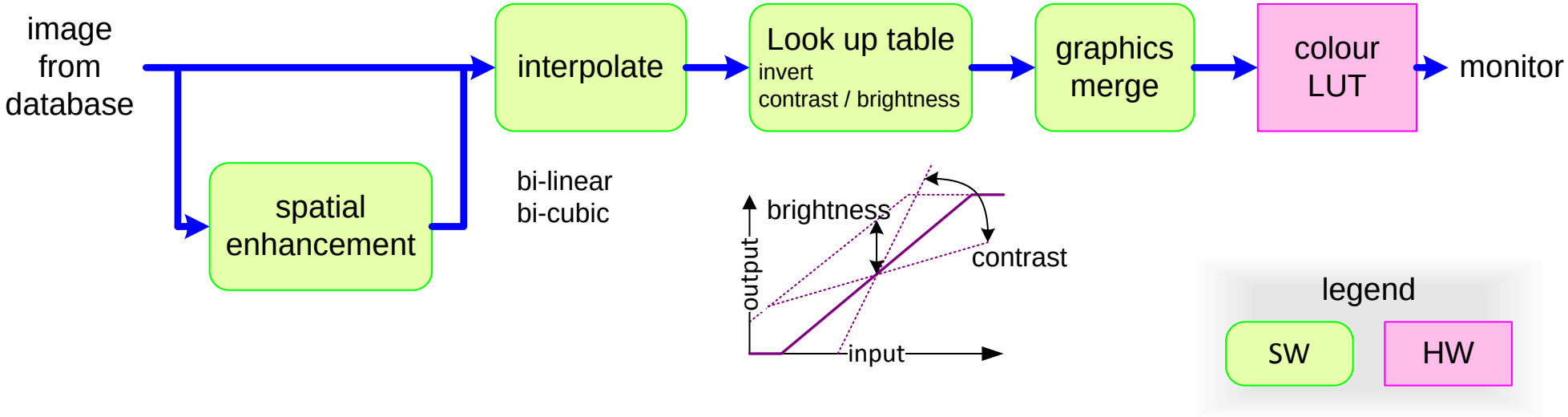
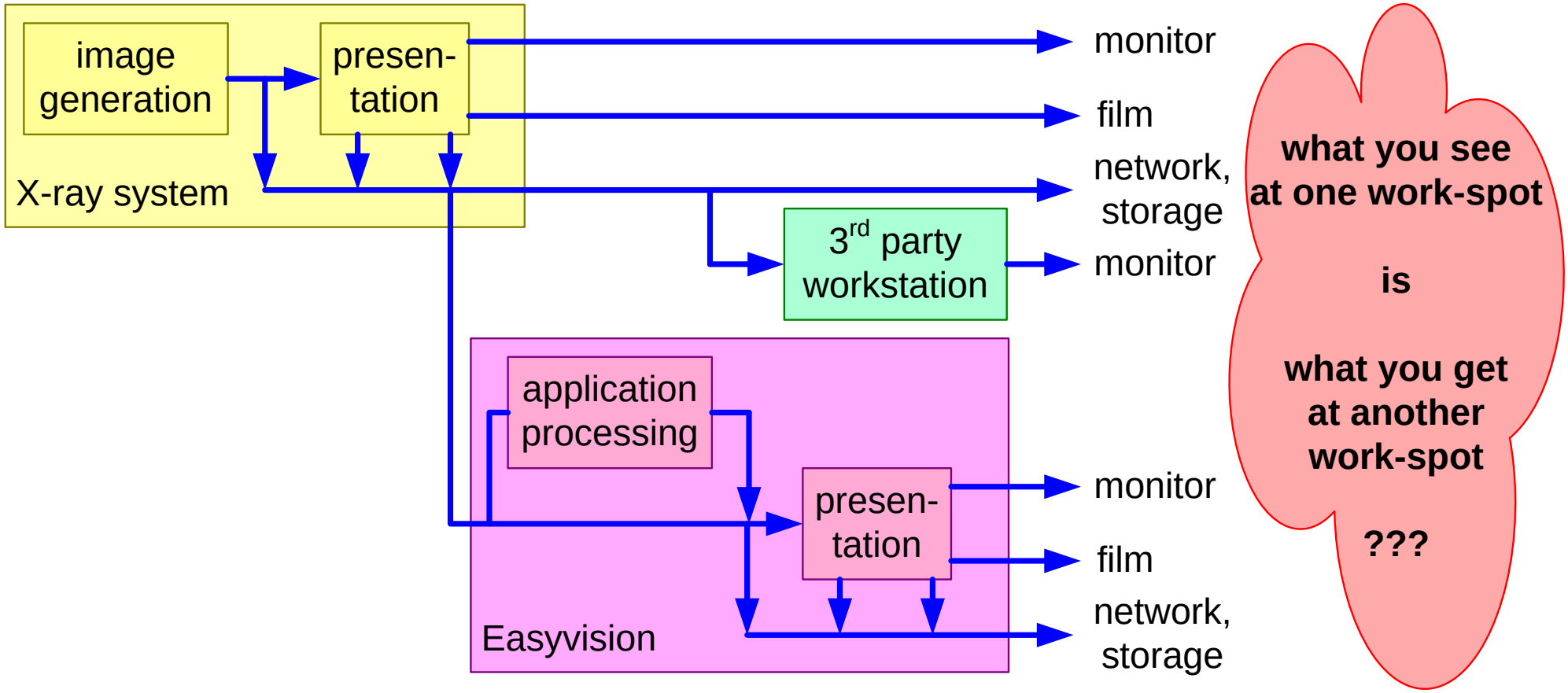
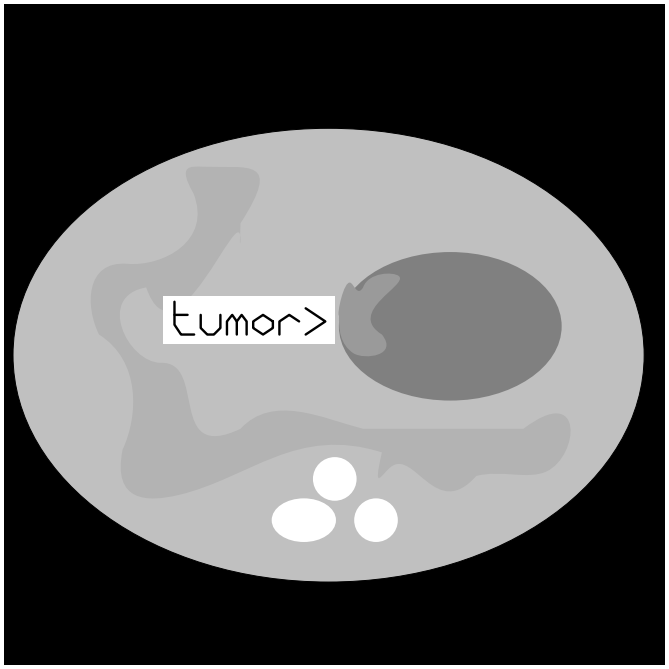


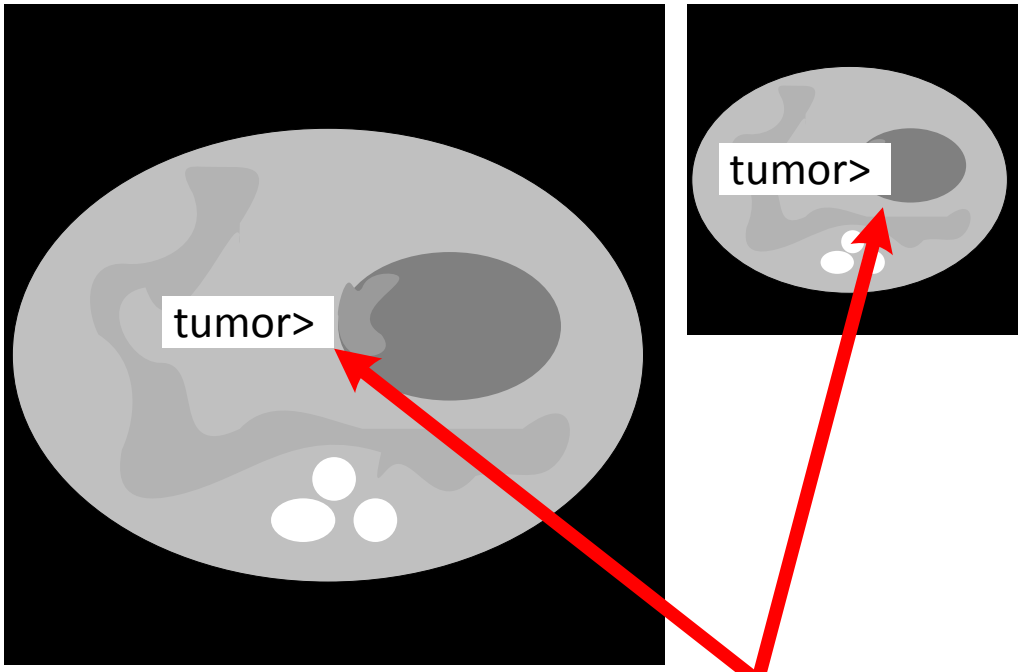
Image Quality expectation WYSIWYG



Safety problem



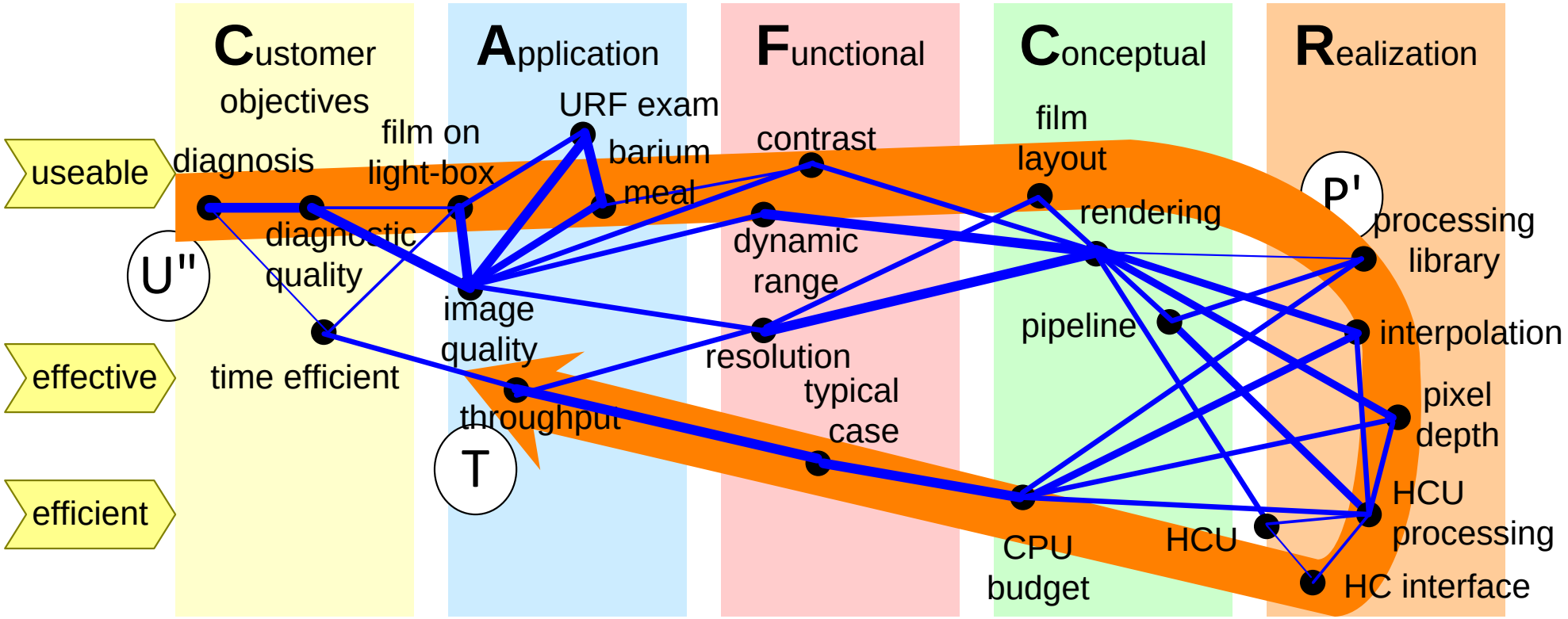
URF monitor output:
fixed size letters at fixed grid



for user readability the font-size was determined "intelligently"; causing a dangerous mismatch between text and image

EV output: scaleable fonts in graphics overlay

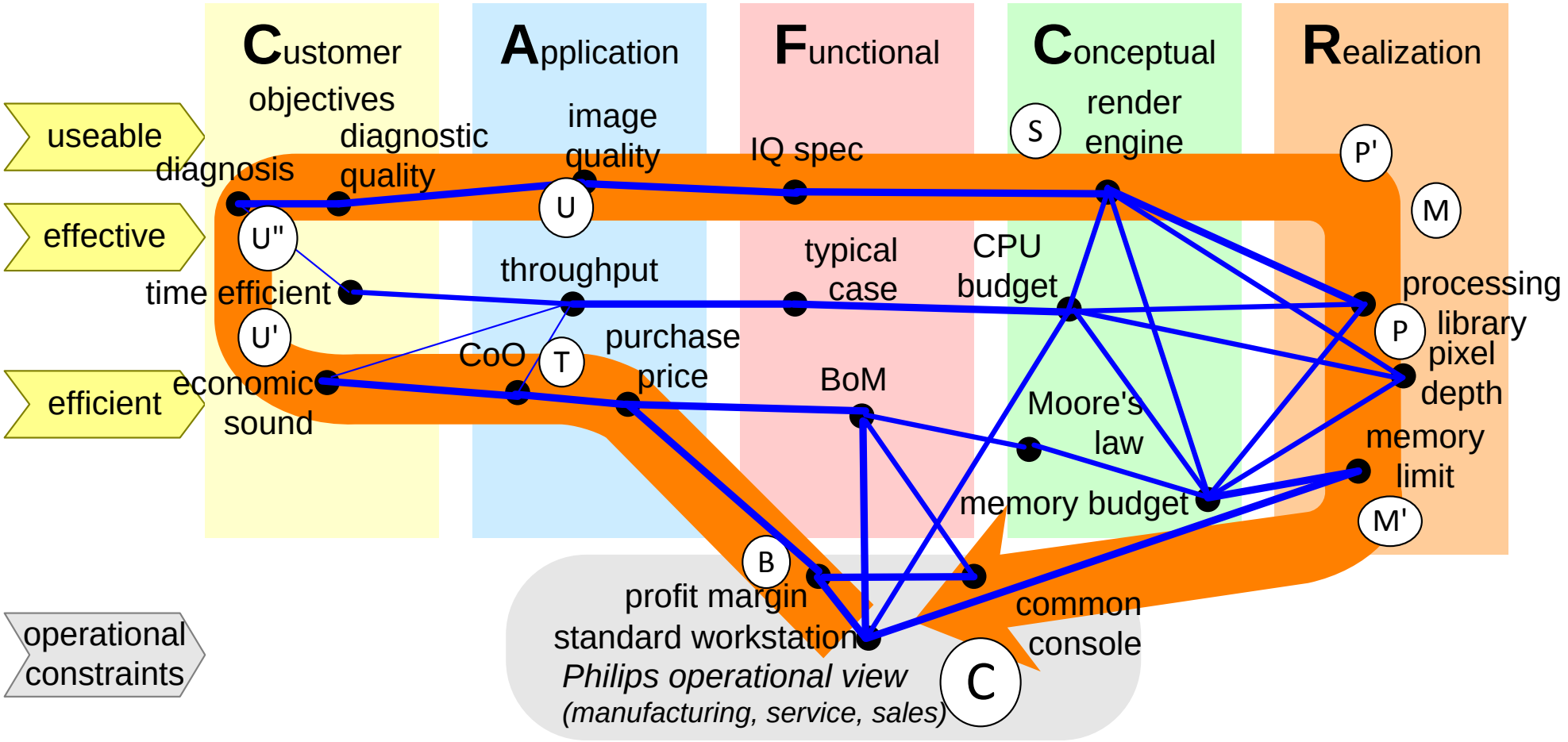
Thread of reasoning; phase 4



Philips operational view
(manufacturing, service, sales)

from extrovert diagnostic quality, via image quality, algorithms and load, to extrovert throughput

Thread of reasoning; phase 5

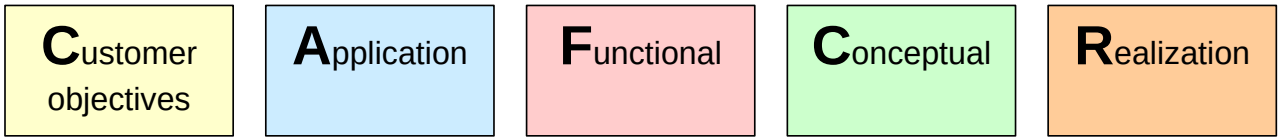


cost revisited in context of clinical needs and realization constraints; note: original threads are significantly simplified

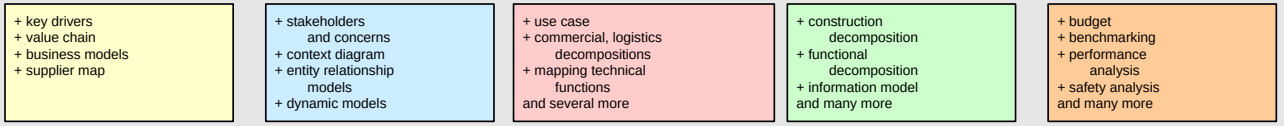
Overview of architecting method

method outline *method visualization*

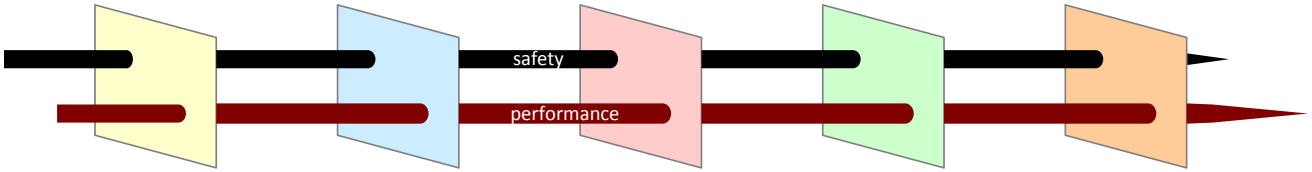
framework



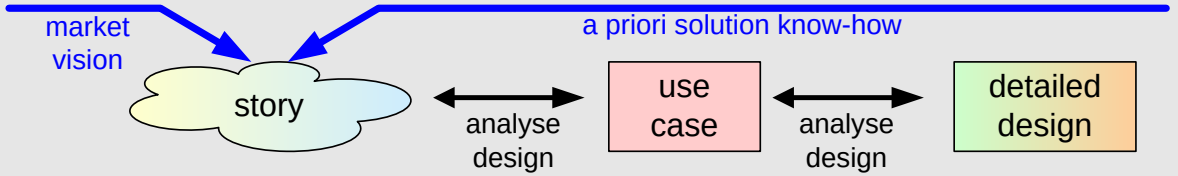
submethods



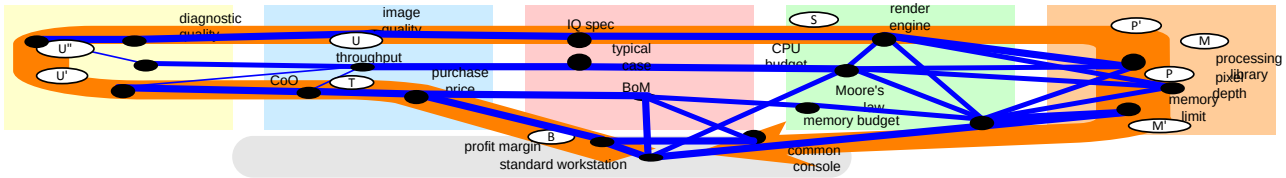
integration via qualities



explore specific details



reasoning



Exercise

- Make a key driver graph

  Use the key driver approach

  Take the recommendations into account

Reflection on Exercise

+ Key drivers put requirements in broader perspective
+ Discussion creates shared understanding

~ The graph needs external feedback

- Are the key drivers really from the customer?
- Are the key drivers *sharp* enough?

Summary Threads of Reasoning

Conclusions

Key Driver graph connects customer objectives to system requirements

Threads of Reasoning connects Customer and Operational Objectives to design and technology choices

The overview is maintained by focusing on valuable, important, critical or sensitive aspect; Look for tensions!

Techniques, Models, Heuristics of this module

Key driver graph

Thread of reasoning

Why, What and How

Tensions

Colophon

The Boderc project contributed to Key drivers and Threads of Reasoning. Especially the work of

Lou Somers, Peter van den Bosch, Zhaouri Yuan (Océ),

Berry van der Wijst (Philips),

Adriaan van den Brand (Centric TSolve),

Heico Sandee and *Maurice Heemels* (TU/e, ESI)

has been valuable.

Module CAFCR wrap up

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

Abstract

This module addresses what we did so far and what still has to be done.

October 11, 2020
status: preliminary draft
version: 0

logo
TBD

Exercise Wrap Up

- Determine architecture status:
  - What do we have?
  - What are the important gaps?
  - What are the urgent gaps?
- How to obtain architecture feedback?
- Determine an integration sequence.