Chair Ecole Polytechnique – Thales
« Engineering of Complex Systems »

Elements of complex systems architecture

Daniel Krob

January 2009

Contents

Contents

Introduction
Philosophy

System approach is both a way of thinking and a standardized international engineering practice …

… whose objective is to master industrial systems complexity in order to optimize their quality, their cost, their time to market and their performance.

The term "system" refers both to the industrial object realized through an industrial process and the highest point of view that one can have when dealing with this industrial object.

- **Why such a course on system architecture fundamentals?**

  The *Elements of complex systems architecture* course intends to **present** and to **clarify** the **key systems architecture concepts** both in an intuitive and formalized (as well as possible) way.

- **What is it about?**
  - Explaining the **industrial background** of systems architecture
  - Introducing to the **system architecture paradigm**
  - Presenting the **key architectural concepts** (systems, architectural framework, model, abstraction, etc.) used in systems architecture
  - Giving an example of a (pseudo-formal) **architectural description language** (SysML)
  - Presenting **complexity measures** for complex industrial systems
  - Discussing the **key challenges** of systems architecture

Contents

Contents

Systems in practice
Some complex industrial systems

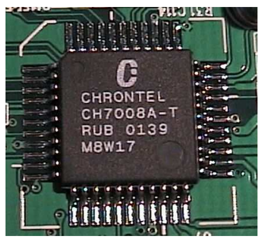
Automobile

Information system

Rocket

Air Traffic Management

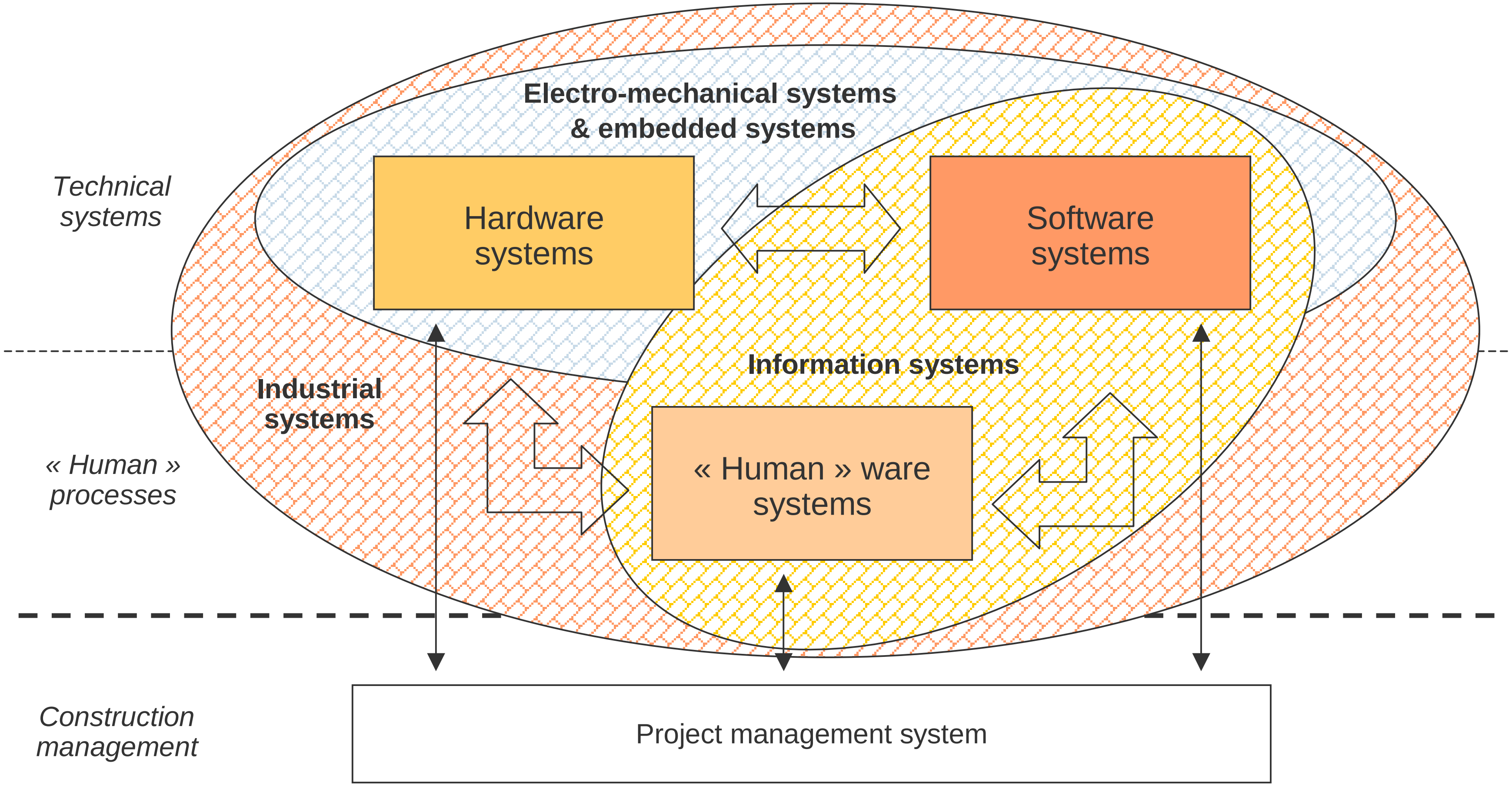
Aircraft



Systems on chip

Examples of complex industrial systems

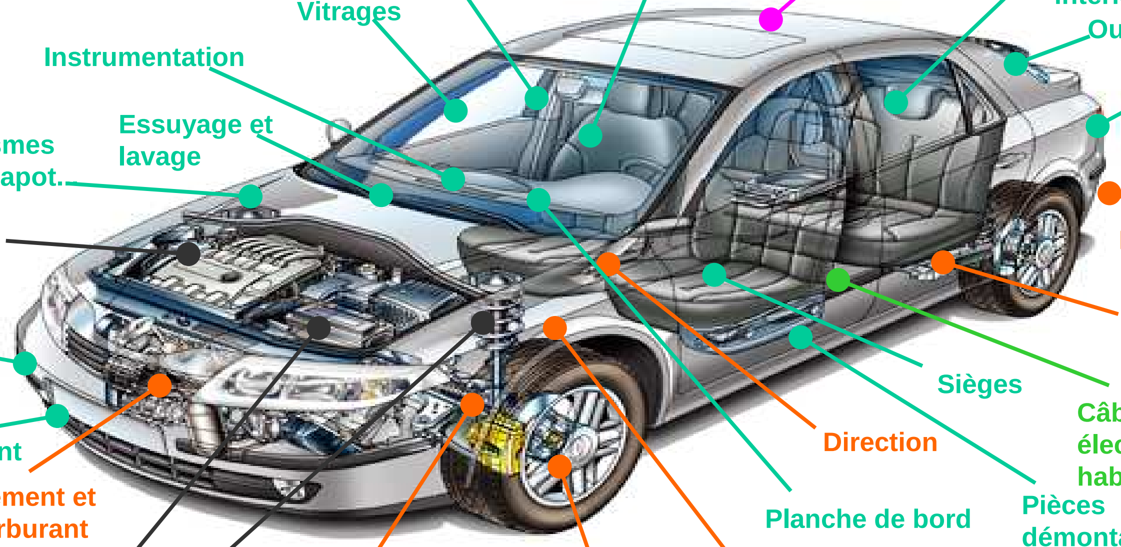
Systems in practice
Typologies of systems



Typical system decomposition of an industrial system

Systems in practice
What are the key systems characteristics in practice? (1)

Decomposition into sub-sub-systems



Source: C. Balle, Renault, 2004

RENAULT

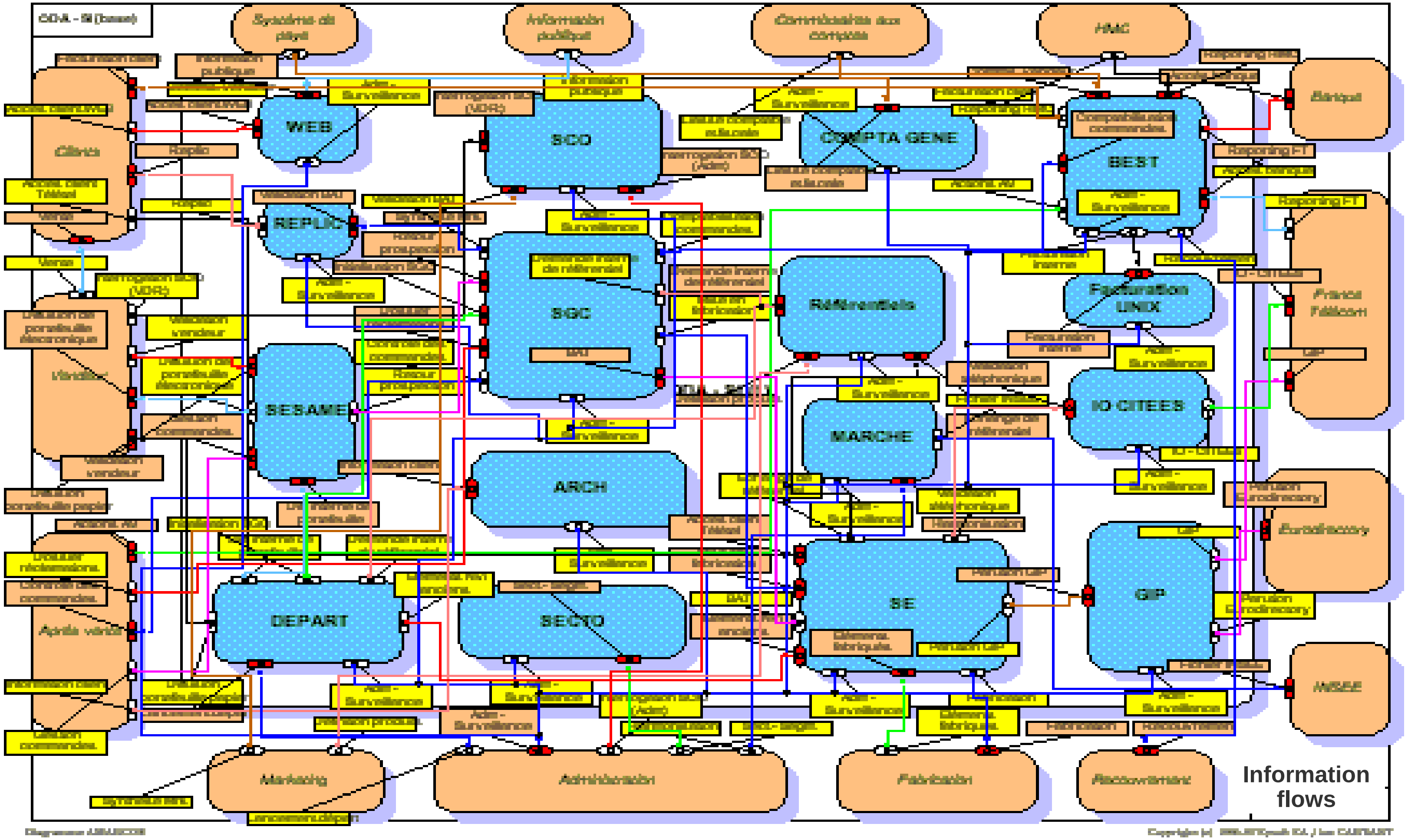
The « car » system

- Carrosserie
- Ground contact
- Engine and propulsion
- Electronic architecture
- Superstructure
- …

Automobile = { 400.000 state variables; 34 principal sub-systems; 14 levels of hierarchy }

Key characteristic 1: hierarchical integration

Systems in practice
What are the key systems characteristics in practice? (2)



Key characteristic 2: industrial systems are permanently evolving …

Systems in practice
Example of practical difficulties: redesign of an hardware system (1)

U.S. Navy Mission (1978)

fighter and attack
aircraft carrier based
3000 flight hours
90 min average sortie
max 7.5g positive
~15 year useful life

Swiss Mission (1993)

interceptor
land based
5000 flight hours
40 min average sortie
max 9.0g positive
~30 year useful life

Standard U.S. Navy F/A-18 C/D Configuration

"Redesign"

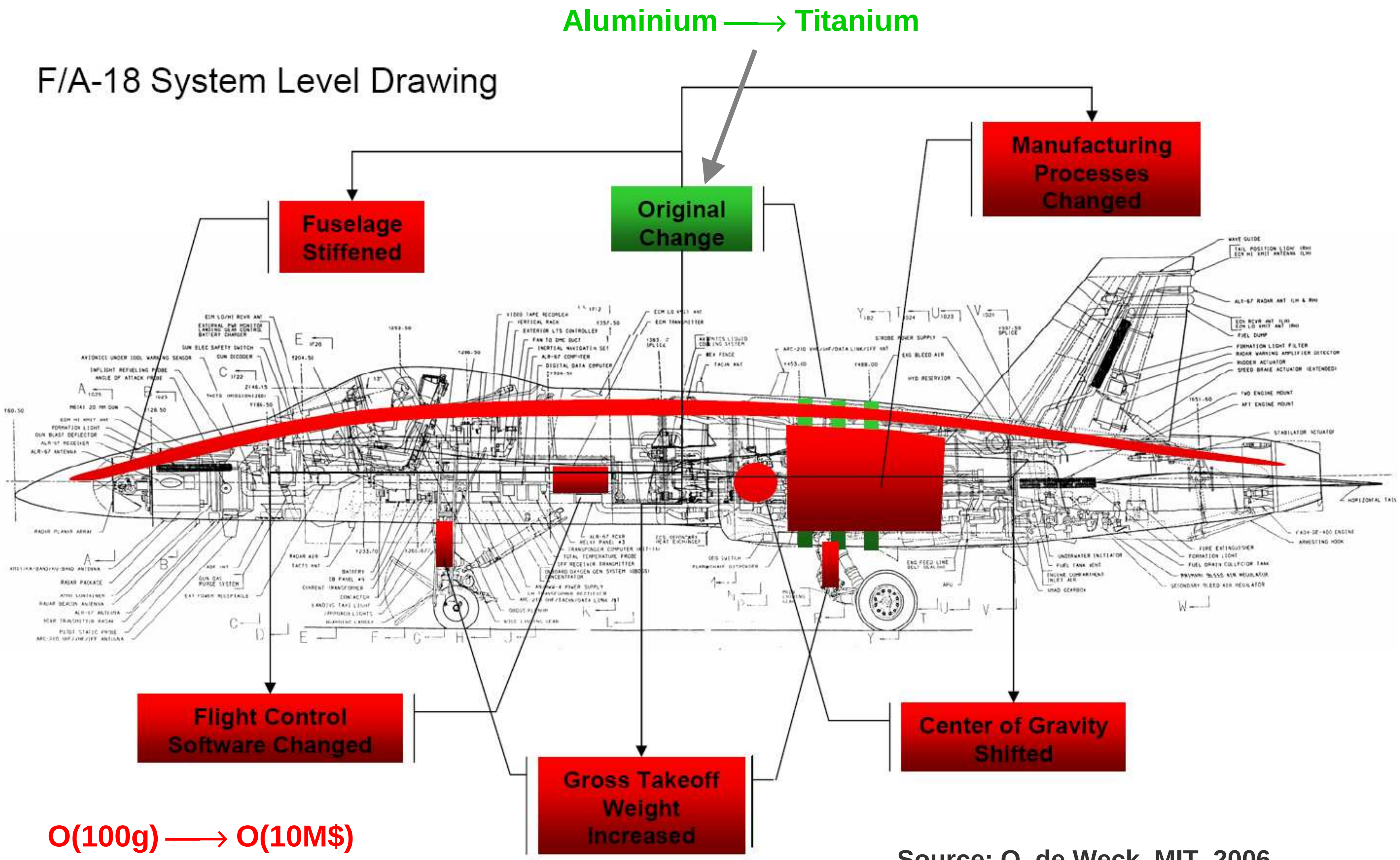
(Switch)

Modified Swiss F/A-18 C/D Configuration

Source: O. de Weck, MIT, 2006

Systems in practice

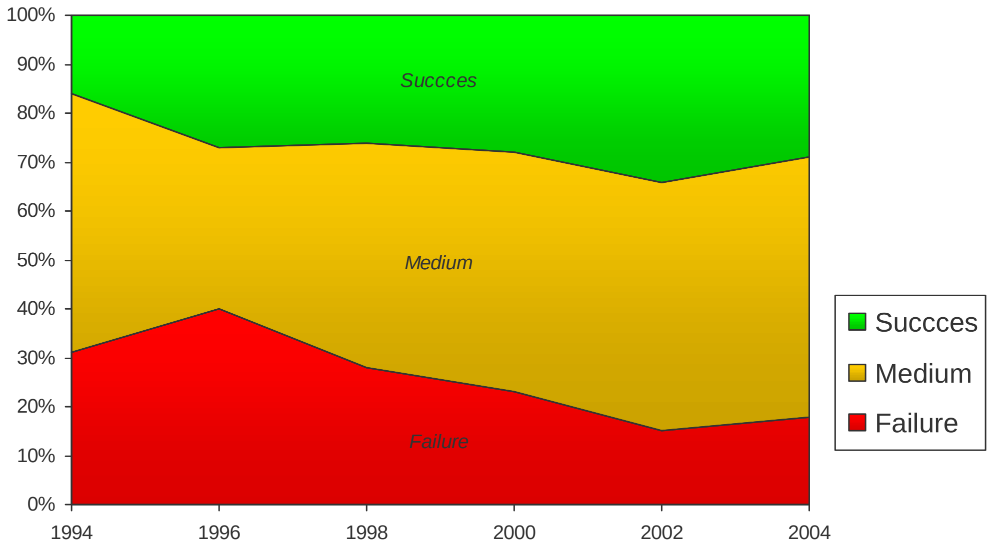
Example of practical difficulties:
redesign of an hardware system (2)



O(100g) ⟶ O(10M$)

Source: O. de Weck, MIT, 2006

Systems in practice
Example of pratical difficulties: information systems (1)



The Chaos study of the Standish Group

The only long term study
on the software failure in the world !

- *Success* = project ended by respecting the technical agenda without any time / budget overcrossing
- *Medium* = project ended without respecting neither the technical agenda, nor the time delays and/or scheduled budget
- *Failure* = project ended before the expected end or never put into operations

Types of projects:

- 71 % : *development projects*
  - 36 %: traditional legacy development
  - 19 %: oriented object legacy development
  - 16 %: mixt strategy (development + software)
- 29 % : *software integration*
  - 4 % : integrated software without modification
  - 13 %: light integrated software parametrization
  - 6 %: assembling of bought components
  - 6 %: heavy integrated sofware parametrization

Number of studied projects:

8.500 / 13.500 projects – every 2 years since 1994

Project origins:

- 45 %: international companies – 35 %: ME – 25 %: SE
- 60 %: USA – 25 %: Europe – 15%: rest of the world

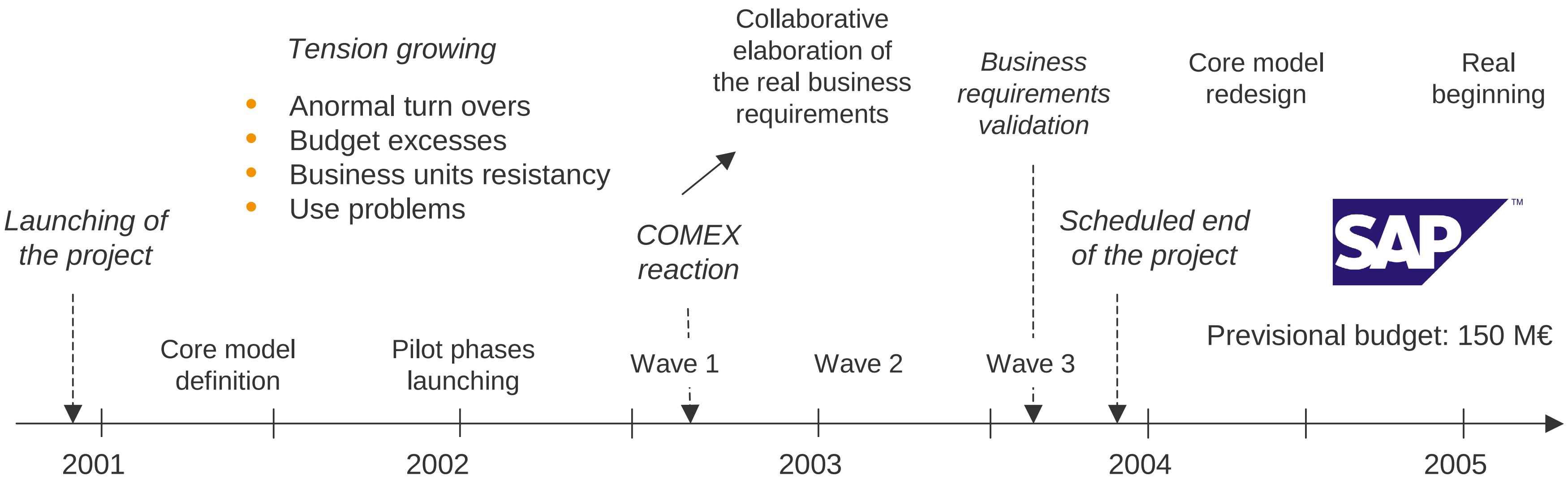
The information systems situation

Systems in practice
Example of pratical difficulties: information systems (2)

Objectives of the project:

- shareholders satisfaction through a worlwide management standard
- creation of homogeneous management processes at the group level
- possibility of permanent access to quality consolidated business figures
- suppression of 50 M€ of management expenses per year

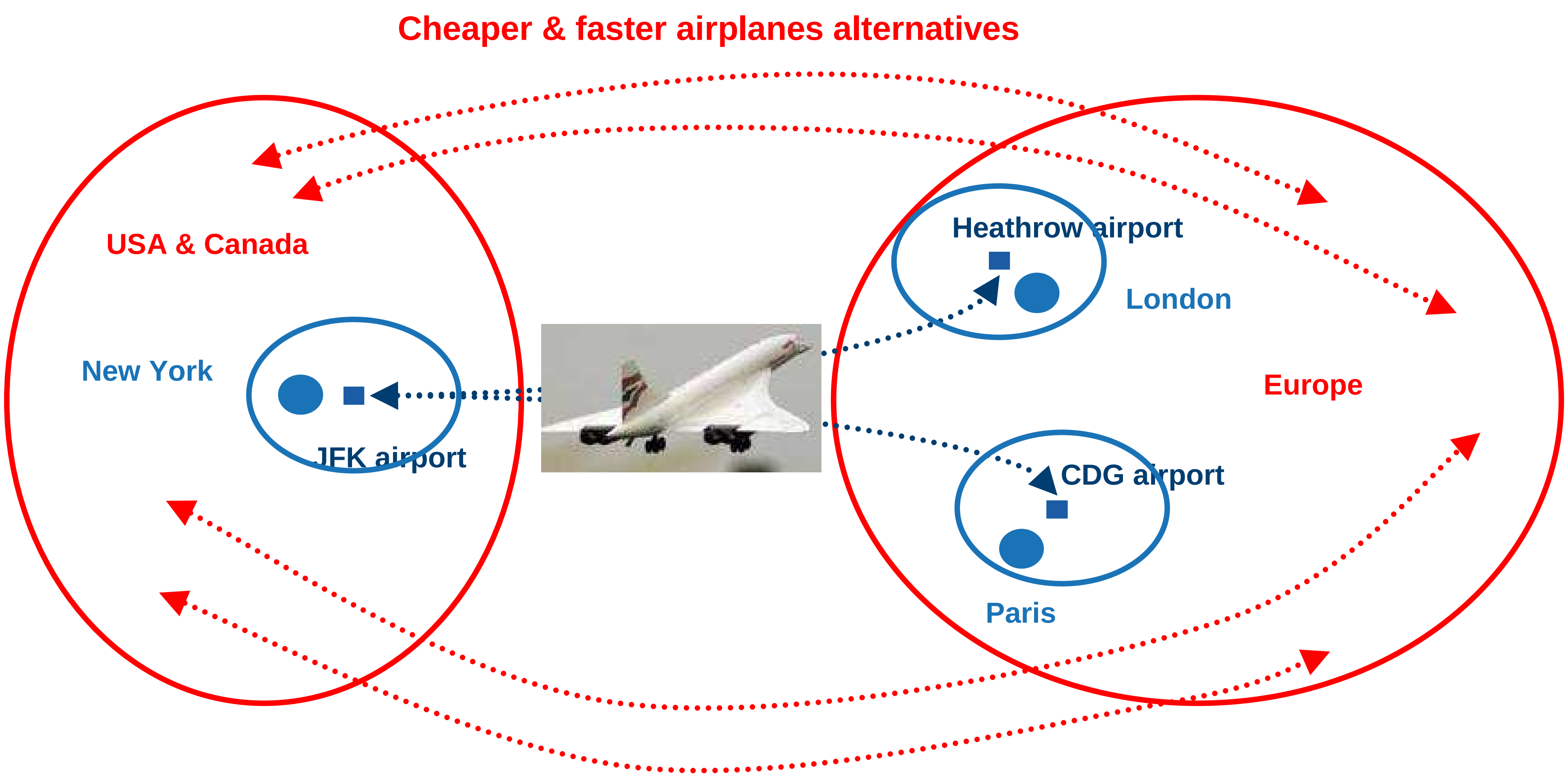
They are non technical !



The (real) story of an ERP project in a big international company

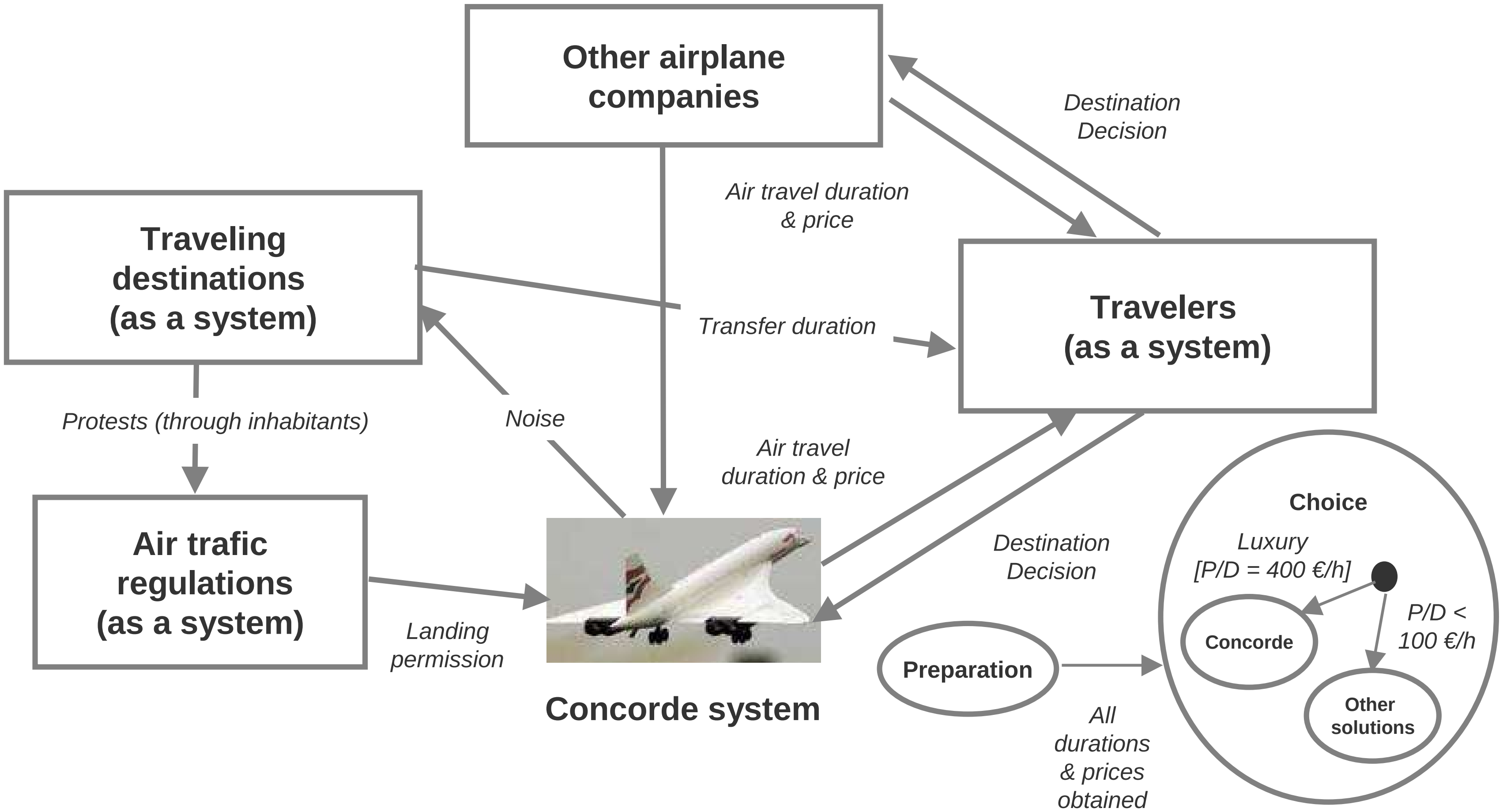
- 80 business units worldwide
- 3 different core activities
- 100.000 people in the world

Contents



A « good » technical solution depends on the considered system !

Systems engineering & architecture
Thinking in terms of “systems” (2)

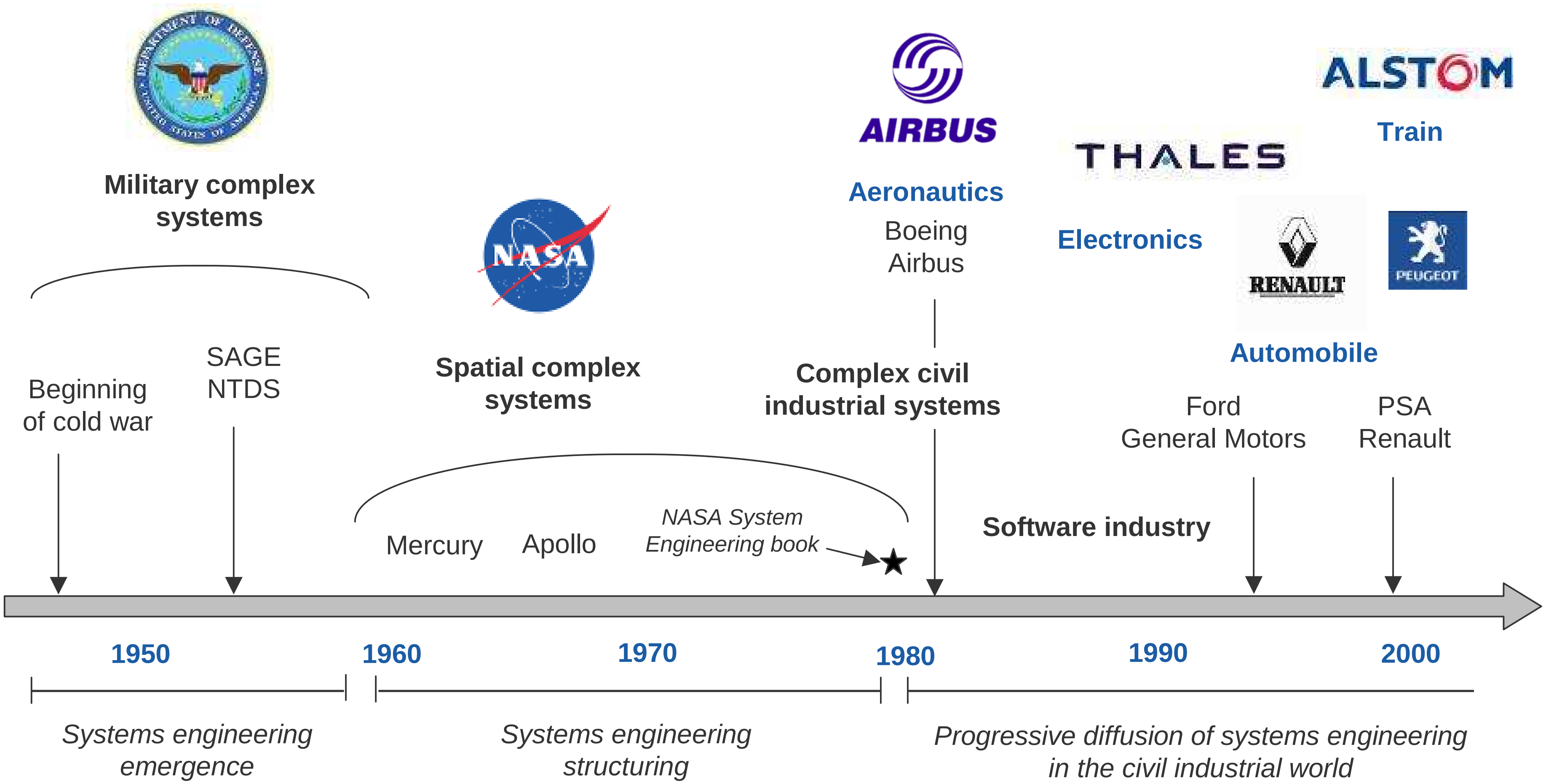


The Concorde case systemic analysis

Traveler behavior model

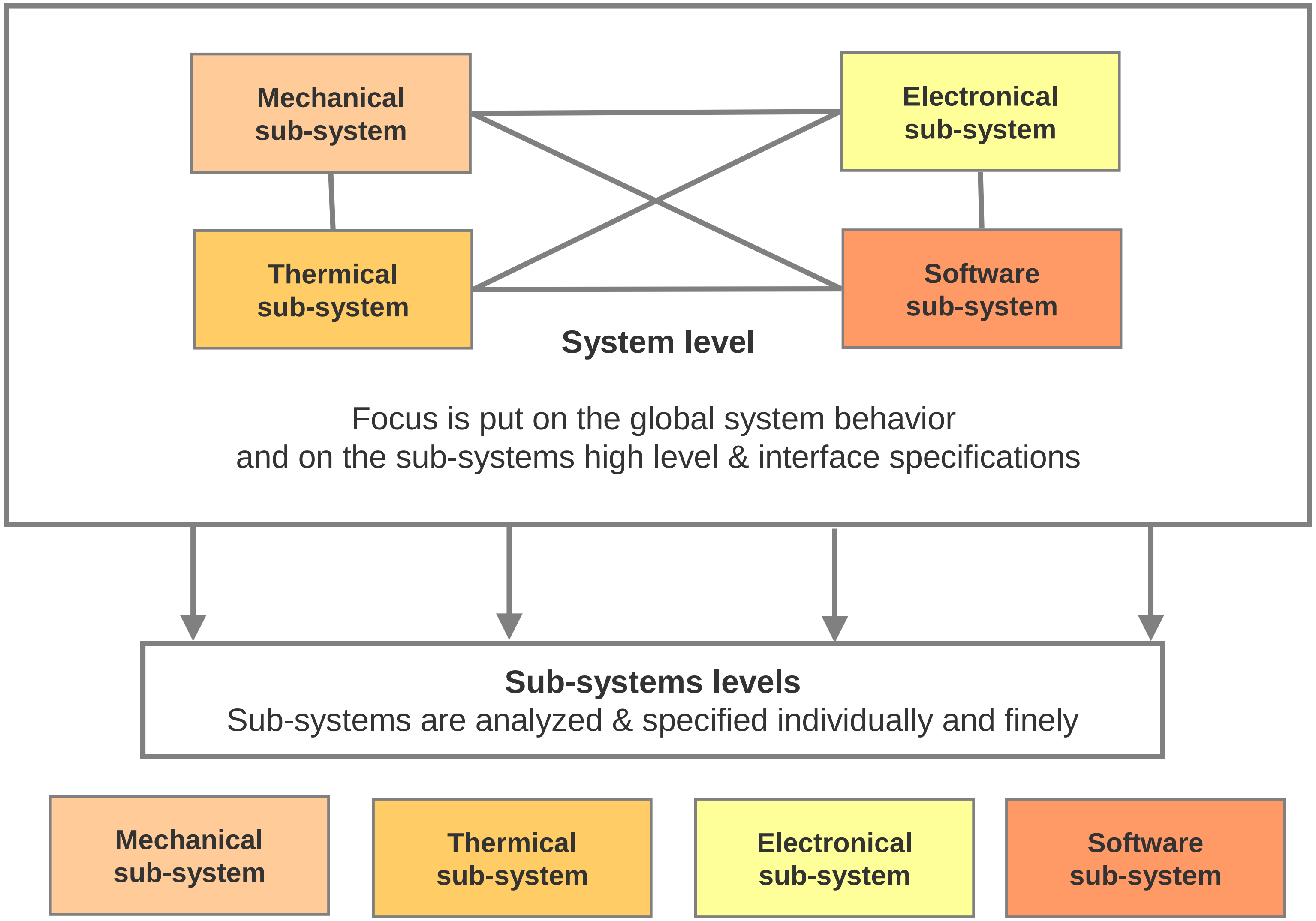
D = Transfer duration + Air travel duration

Systems engineering & architecture
Systems engineering history

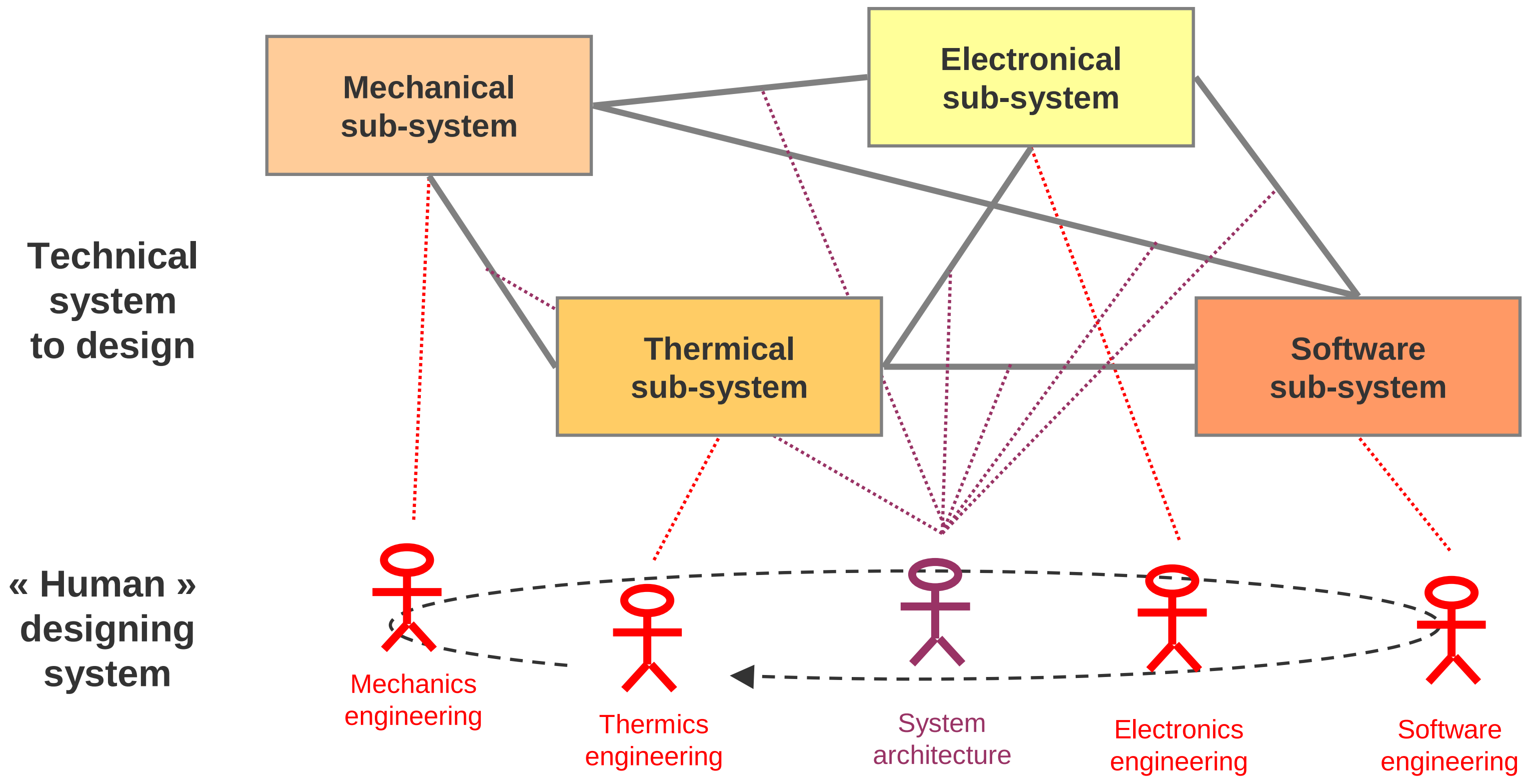


- SAGE = Semi-Automatic Ground Environment = 1st American anti-aircraft defense system
- NTDS = Navy Tactical Data System = = 1st American naval defense system

Systems engineering & architecture
The two sides of system design (1)



Systems engineering & architecture
The two sides of system design (2)

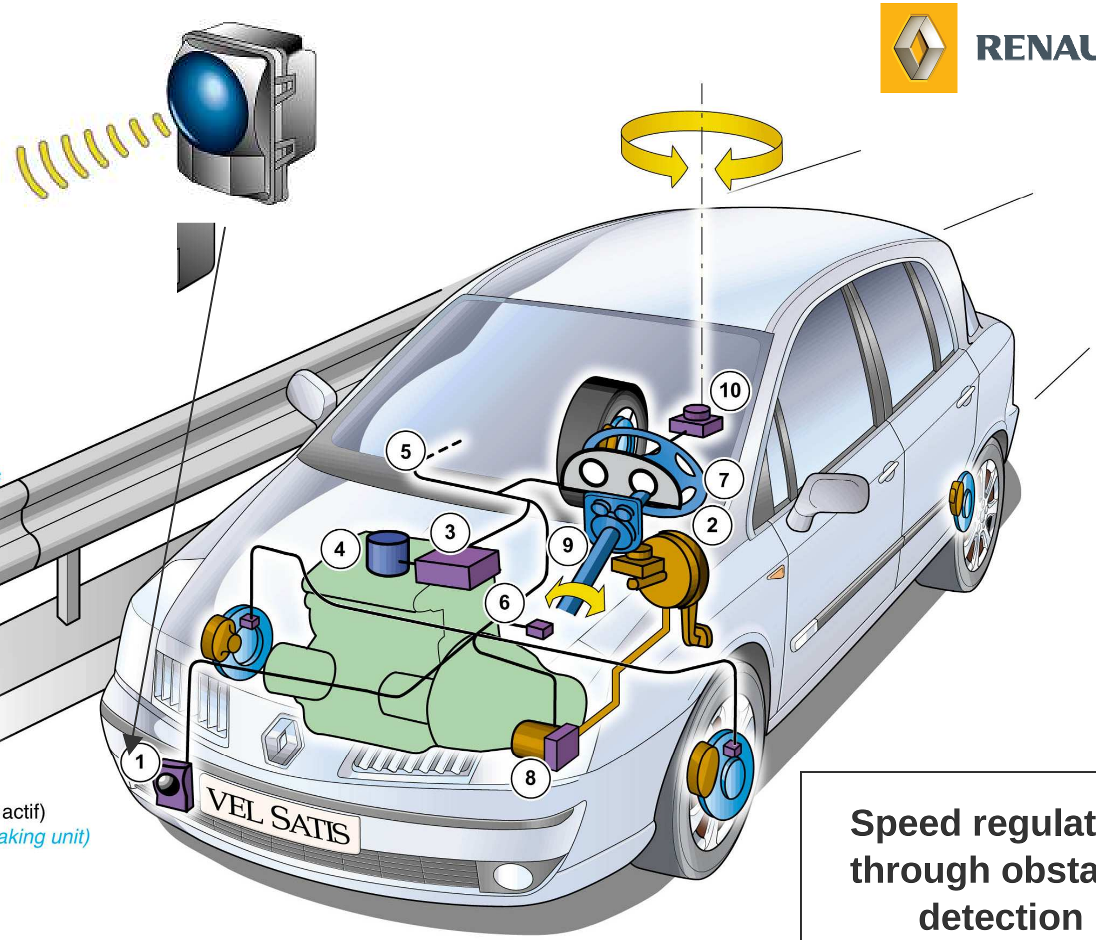


Sub-systems engineering is in charge of the homogenous boxes when system architecture is responsible of the heterogeneous arrows

Systems engineering & architecture
What does it mean in pratice: the example of a “transversal” function

Main sub-systems
- Carrosserie
- Ground contact
- Engine and propulsion
- Electronic architecture

RENAULT



1. Radar
2. Commandes au volant
   Steering wheel mounted controls
3. Calculateur contrôle moteur
   Engine management unit
4. Papillon motorisé
   Motorized throttle valve
5. CAN
6. Boite de vitesse automatique
   Automatic transmission
7. Tableau de bord
   Instrument panel
8. Calculateur ABS / ESP (freinage actif)
   ABS / ESP control unit (active braking unit)
9. Capteur d'angle au volant
   Steering wheel angle sensor
10. Capteur de vitesse de lacet et d'accélération transversale
    Yaw and lateral acceleration speed sensor

Régulateur de vitesse à contrôle de distance
Adaptive cruise control

Speed regulation through obstacle detection

Source: C. Balle, Renault, 2004

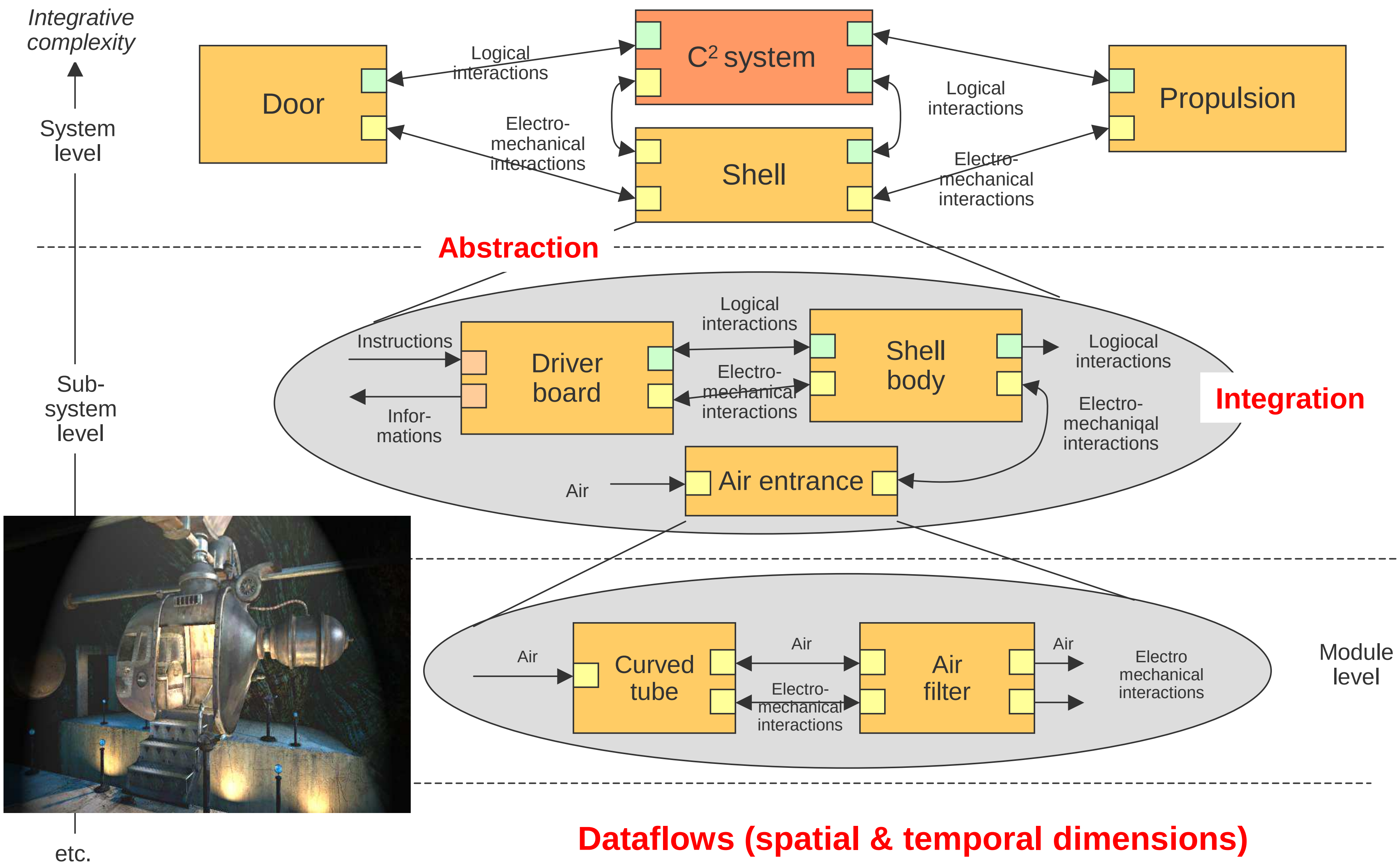
Systems engineering & architecture
Synthesis: architecture versus analysis

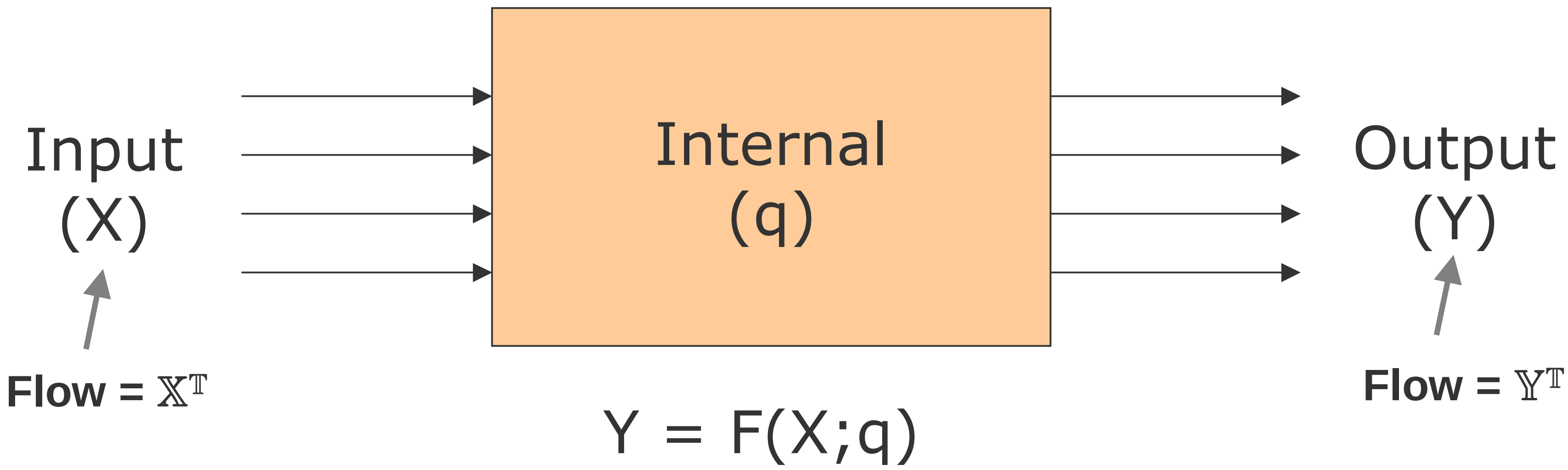
Paradigm	Analytical	Architectural
Key principle	Exhaustive understanding	Global understanding
Perimeter	Homogeneous system	Heterogeneous system
Building blocks	Disciplinary knowledge	Systems & interfaces
Mindset	Uniqueness & certainty	Diversity & relativity
Description mode	Detailed representation	Perceptions & viewpoints
Working mode	Assembling	Integration
Interaction mode	Expertise & local	Collaborative & global
Industrial specialist	Engineer	Architect

Contents

Contents

What is a system?
What do we want to model?

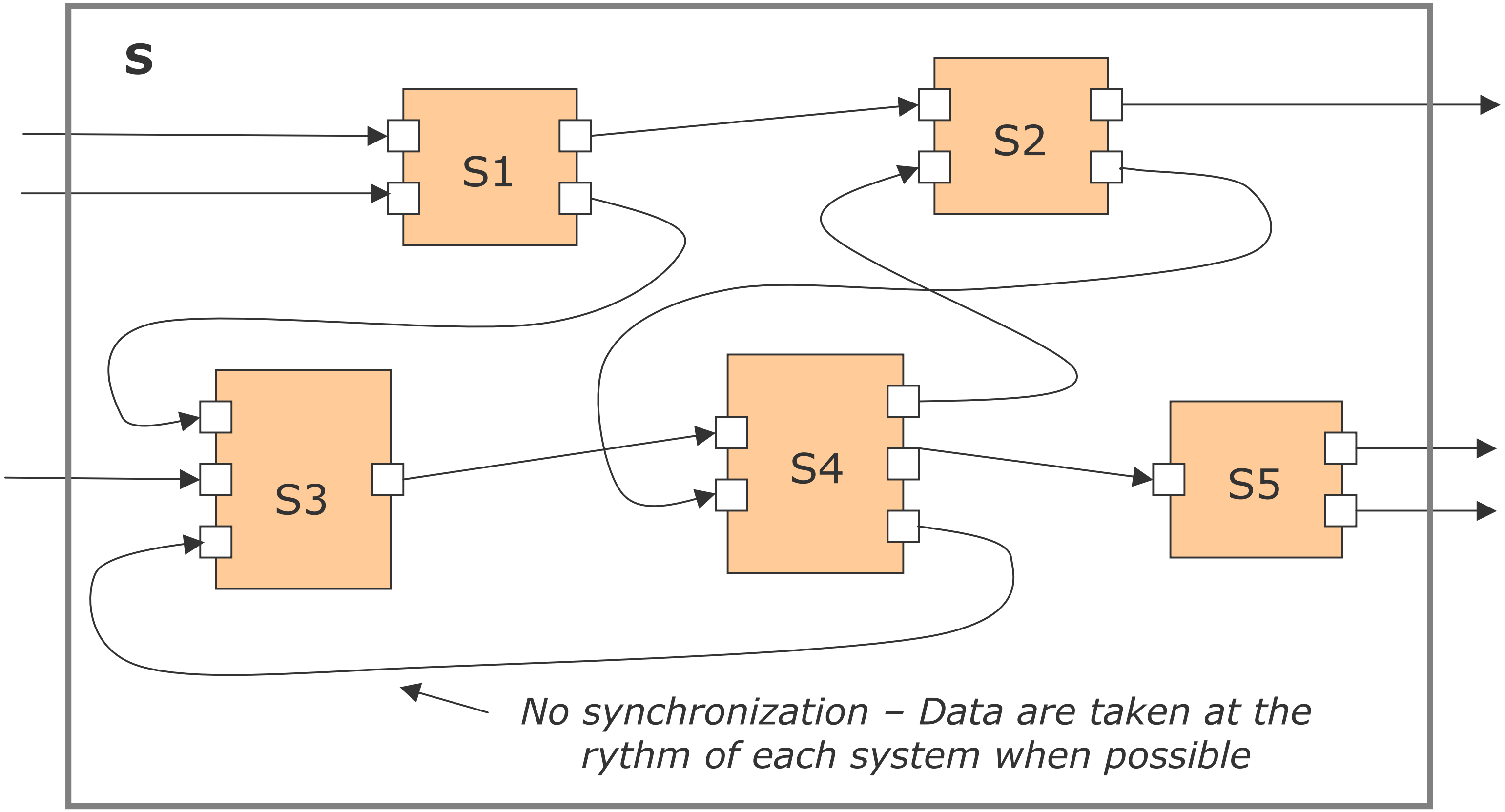




A system is a continuous partial function on dataflows that may transform an input dataflow (X) into an output dataflow (Y) depending on its internal state (q)

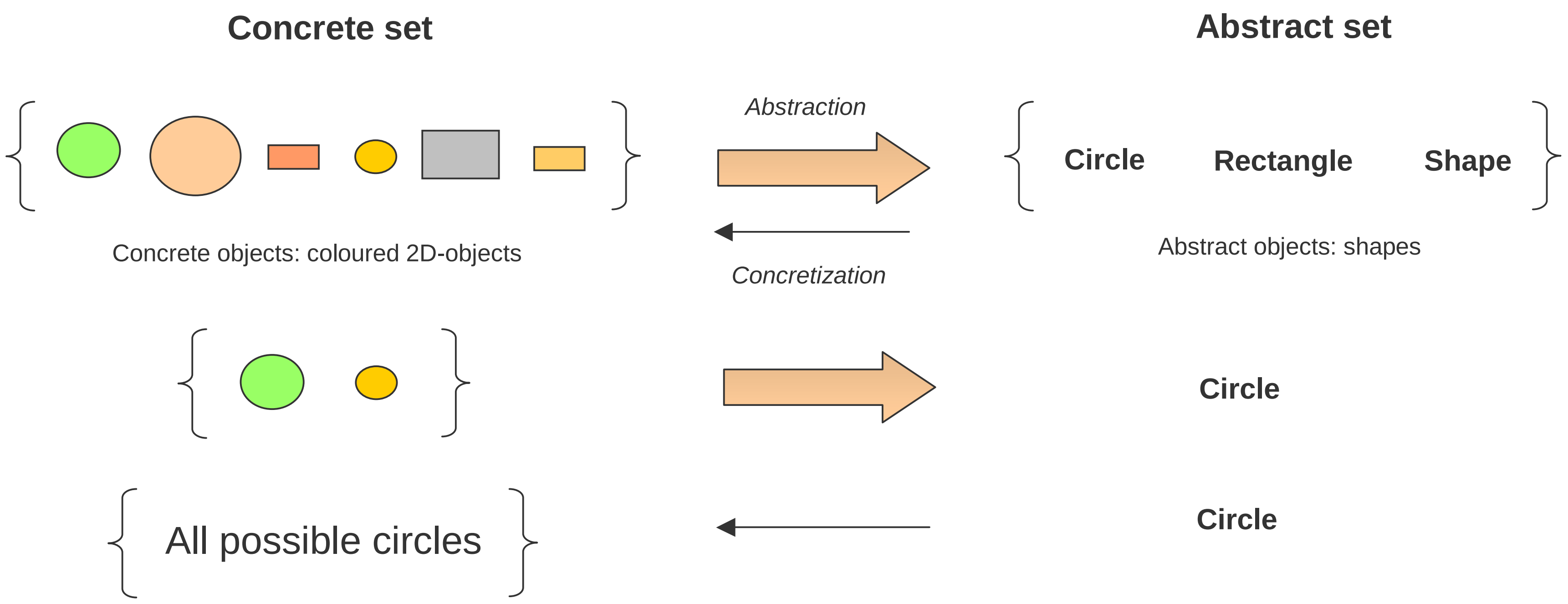
Key note: a **behavioral definition** is mandatory due to the fact that **logical behaviors** are the only **common points** between **all different types of homogeneous systems** at the level of abstraction that we must use from a **systems architecture** point of view

What is a system?
Key operator 1: integration



The integration operator is defined by a fixed point semantics (Kahn; 1974)
Continuity is the only technical property used to prove system stability

What is a system?
Key operator 2: abstraction

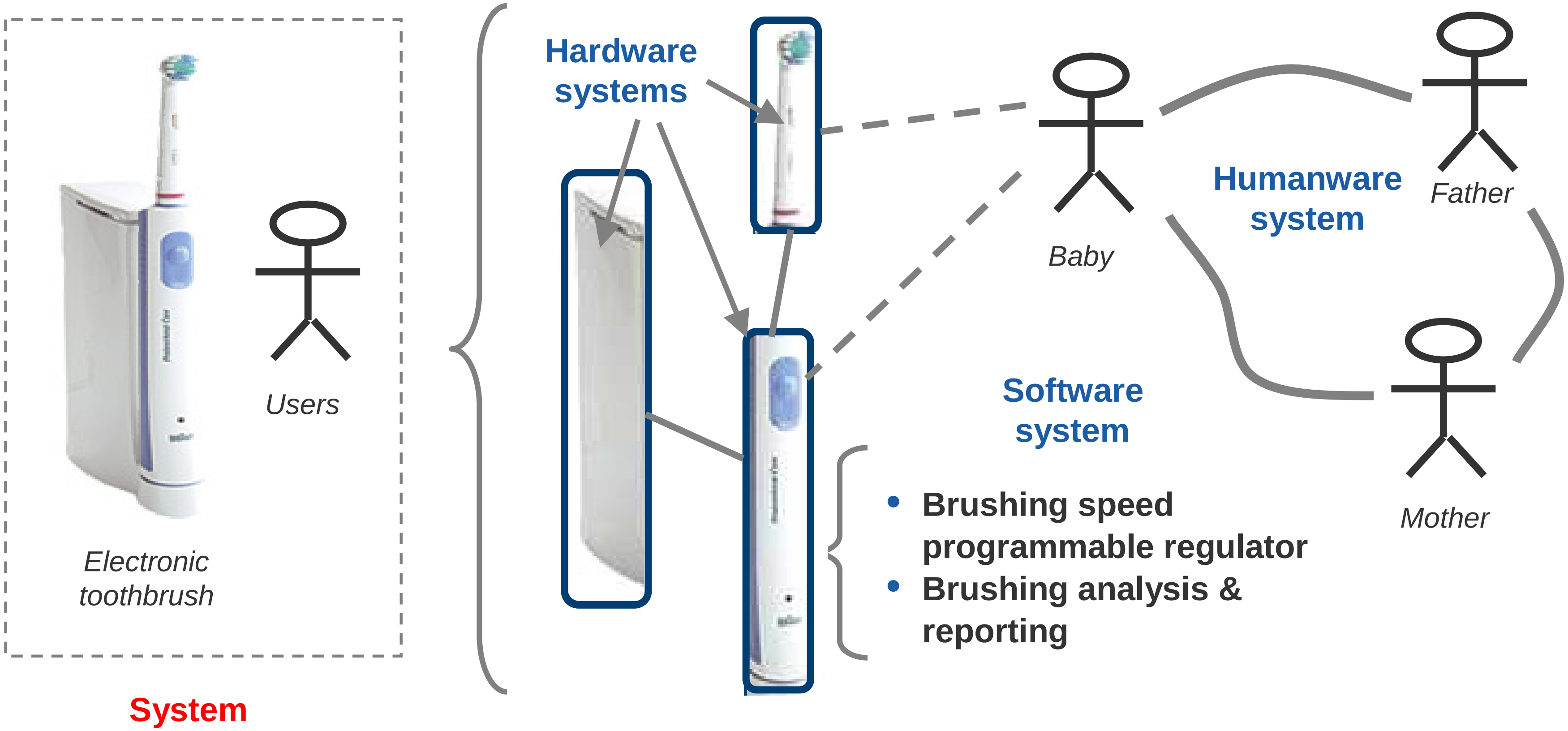


Formal definition
(Galois correspondance;
P. & R. Couzot)

- $\text{Abstraction}(\text{Concretization}(A)) \subseteq A$
- $C \subseteq \text{Concretization}(\text{Abstraction}(C))$

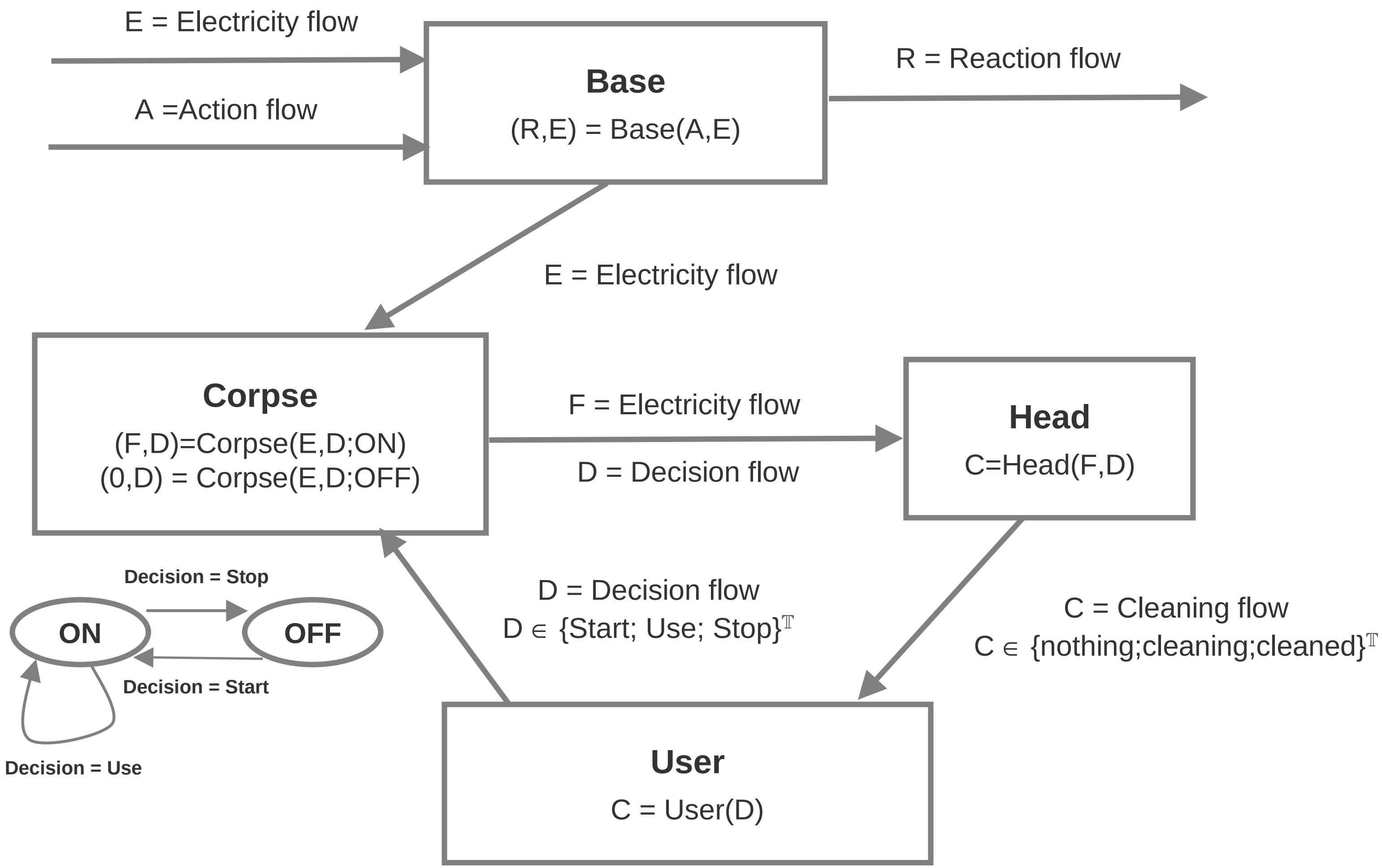
An abstraction is a non (too) destructive idealization of a set of objects

What is a system?
Example of a formal system modeling (1)



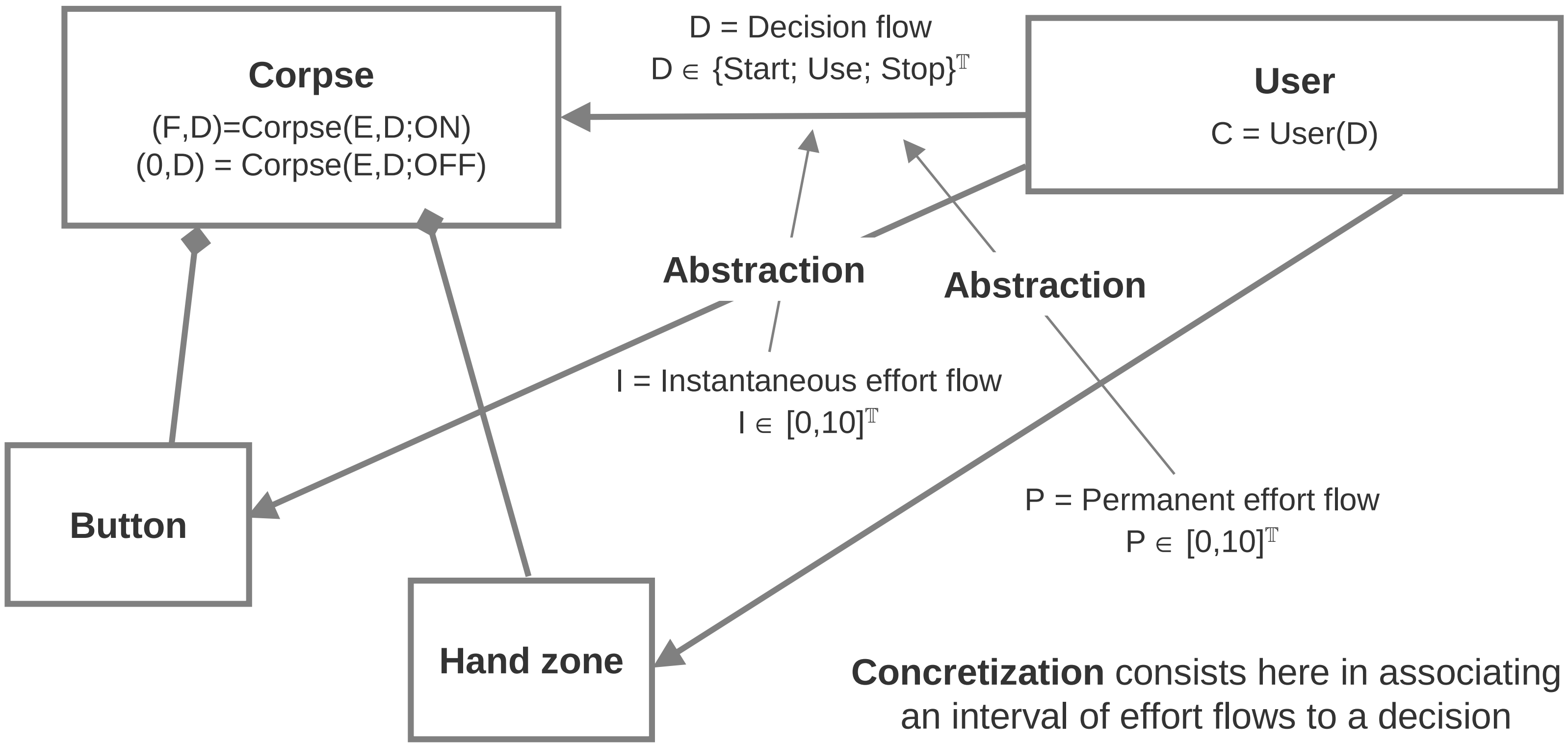
The considered system: an electronic toothbrush + its users

What is a system?
Example of a formal system modeling (2)



The full system is obtained by integration from this description

What is a system?
Example of a formal system modeling (3)



Concretization consists here in associating an interval of effort flows to a decision

To enter into a detailed modeling of the considered system, one must use abstraction / concretization operators

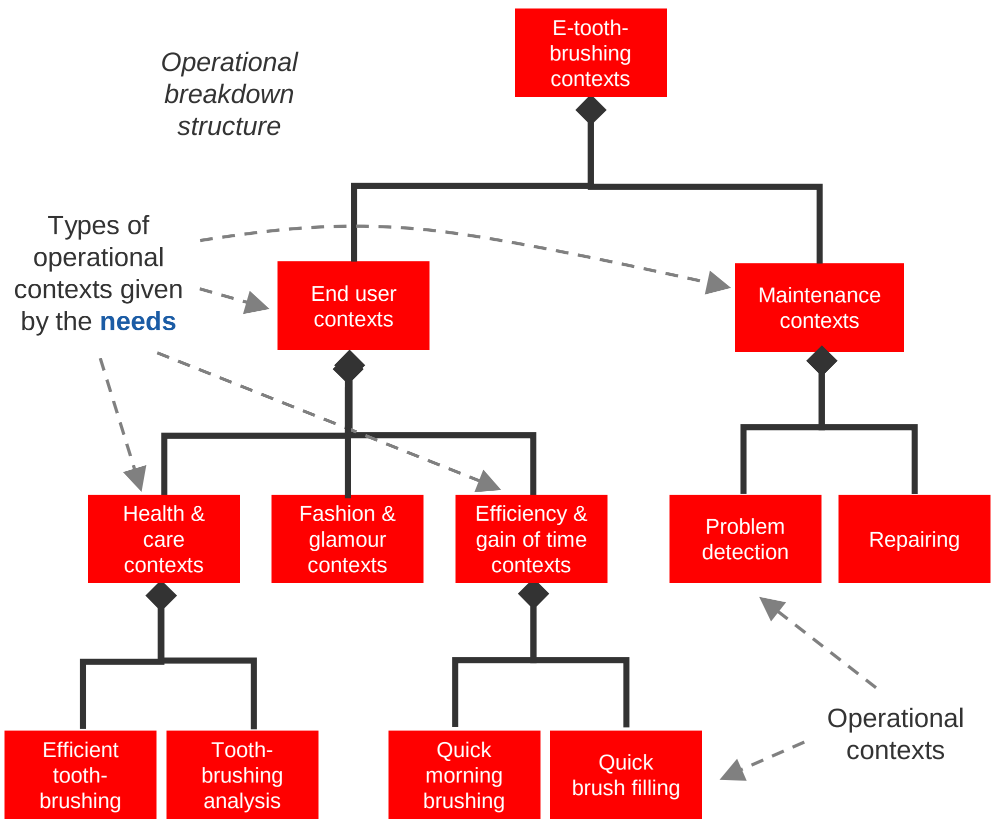
Contents

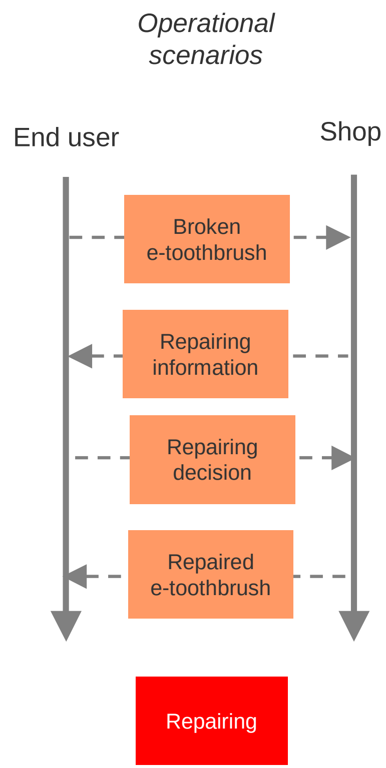
Electronic toothbrush

Viewpoint	Answers to the question	Some associated keywords	Examples (e-toothbrush)
Operational	Why ?	Operational context, mission, use case	Clean & healthy teeth, gain of time, fashion bathroom
Functional	What ?	Service, function, task, operation, mode of operation	Brushing, speed regulating, brushing strength programming
Constructional	How ?	Component, device, configuration	Head, base, corpse, speed regulator

Systems architecture frameworks
Operational vision

Electronic toothbrush



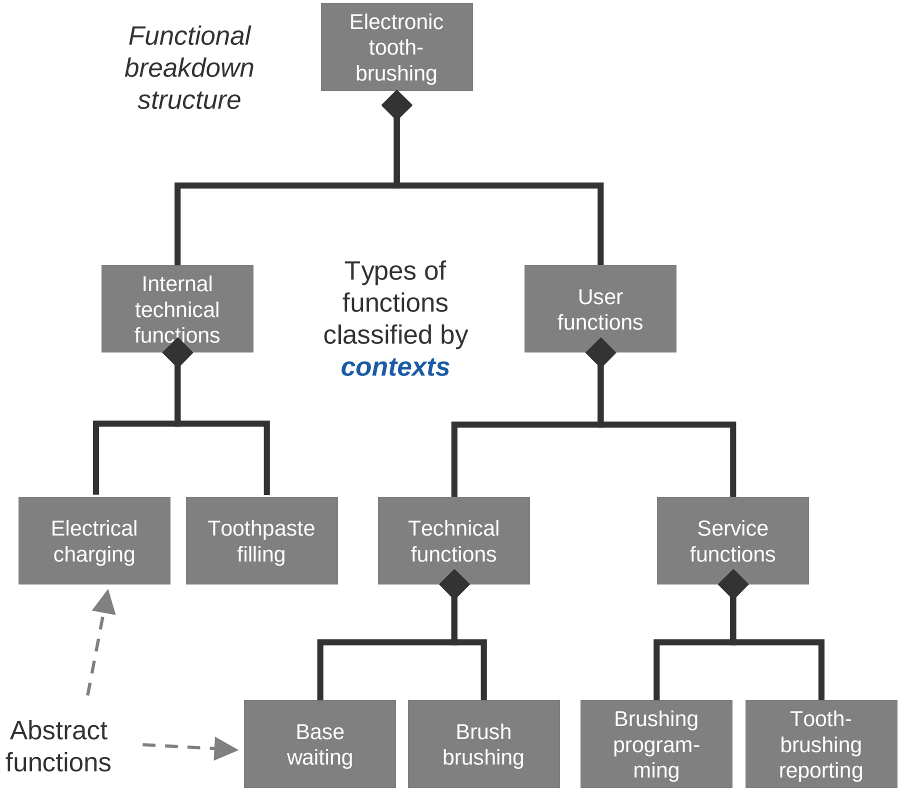


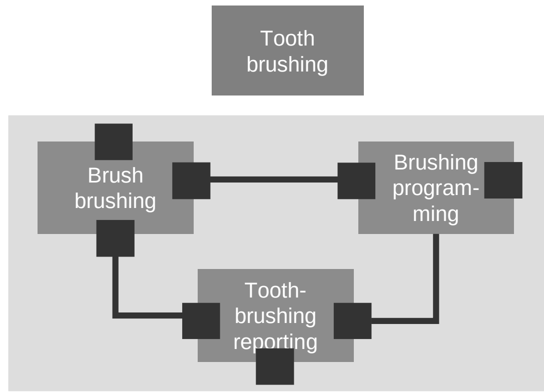
Operational vision: ***defines the intended objectives & uses and the ways of operating of the system relatively to the externally interfaced systems (customers, end users, etc.)***

Systems architecture frameworks
Functional vision

Electronic toothbrush

Functional breakdown structure





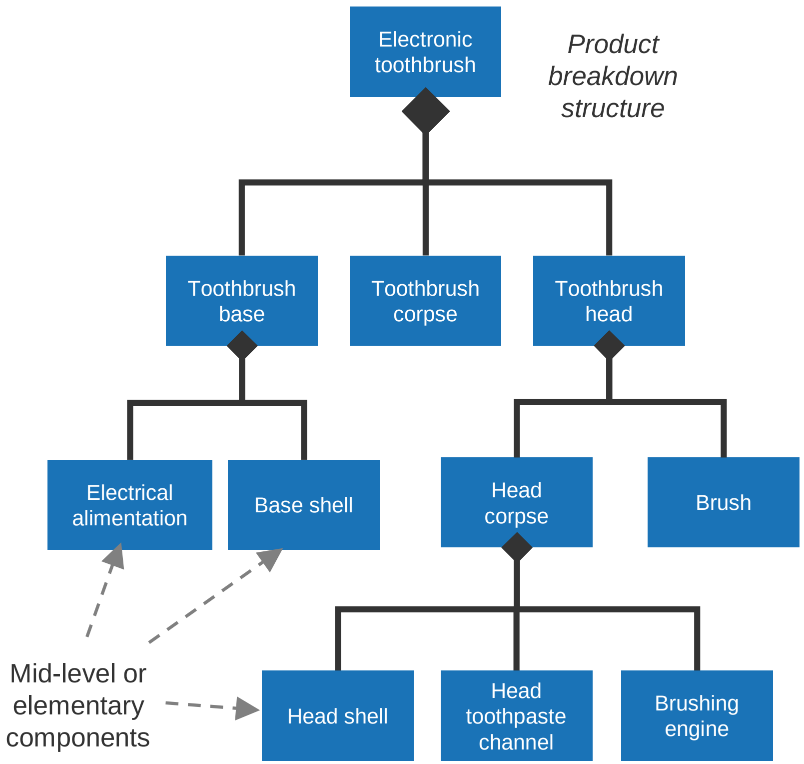
Process organization

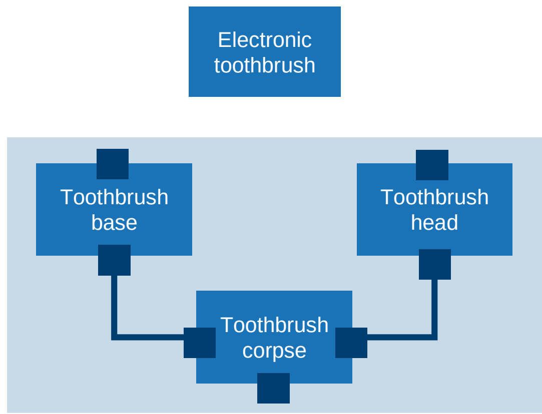
- **Brush brushing** is a function
- **Brush brushing using QTB123 brush** is not a function

That is to say not directly associated with a technical solution

Functional vision: ***defines the abstract functions that are required to perform the missions of the system***

Electronic toothbrush



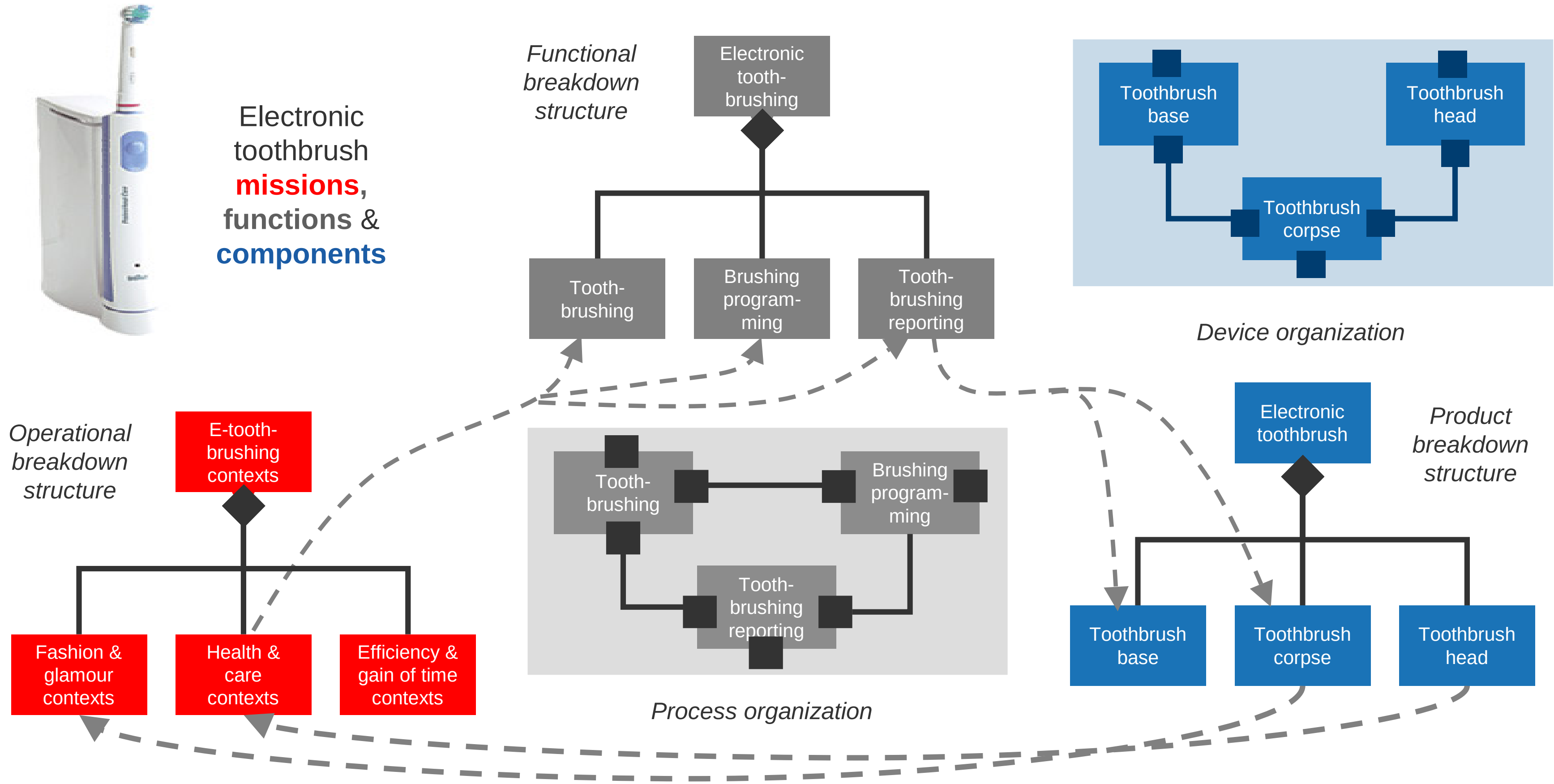


Device organization

- **Hardware** components
- **Software** components
- **People**

Constructional vision: ***defines the real components that are going to perform the abstract functions***

Systems architecture frameworks
Relationships between systems visions

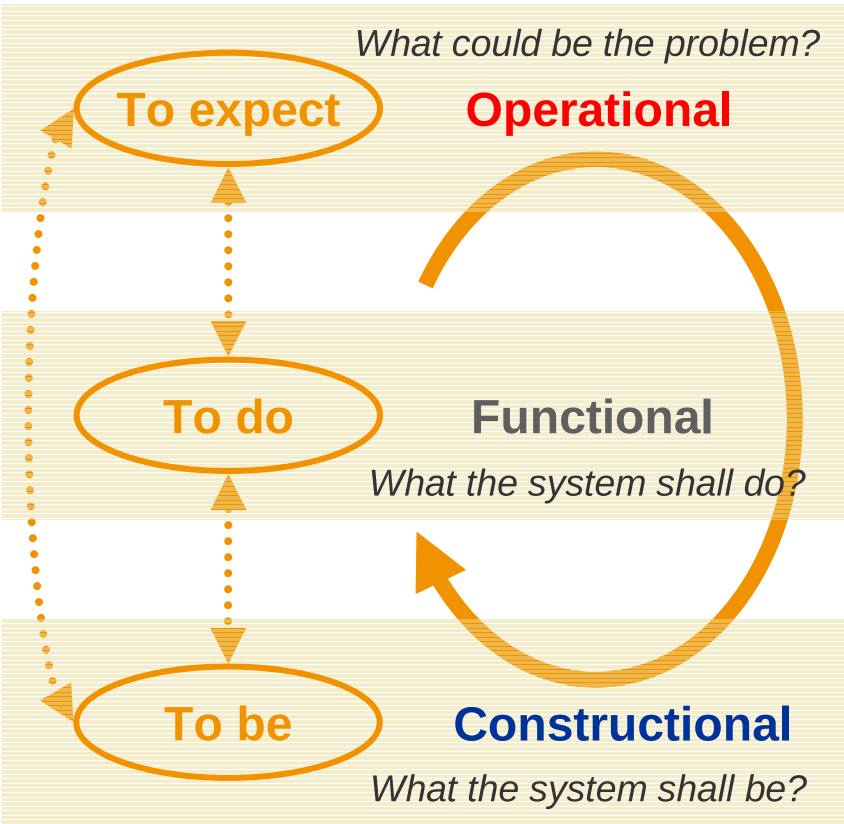


Viewpoint traceability: ***tracing the coherent organization of functions, components & missions of a given system***

Systems architecture frameworks
Example of architectural framework (1)

Classified according to their supervision frequency

SAGACE framework

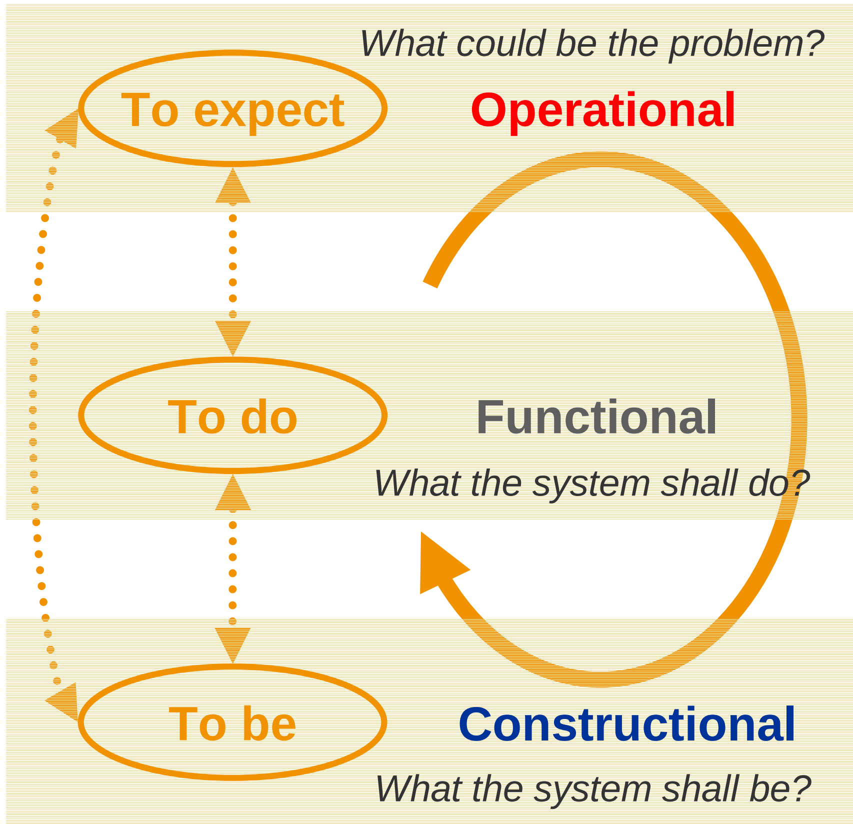


Action	Tactics	Strategy
Control context (scenario)	Management context (scenario)	Strategic context (scenario)
Function (dependency)	Task (operation)	Operation mode (scenario)
Resource (architecture)	Device (organization)	Configuration (scenario)

Systems architecture frameworks
Example of architectural framework (2)

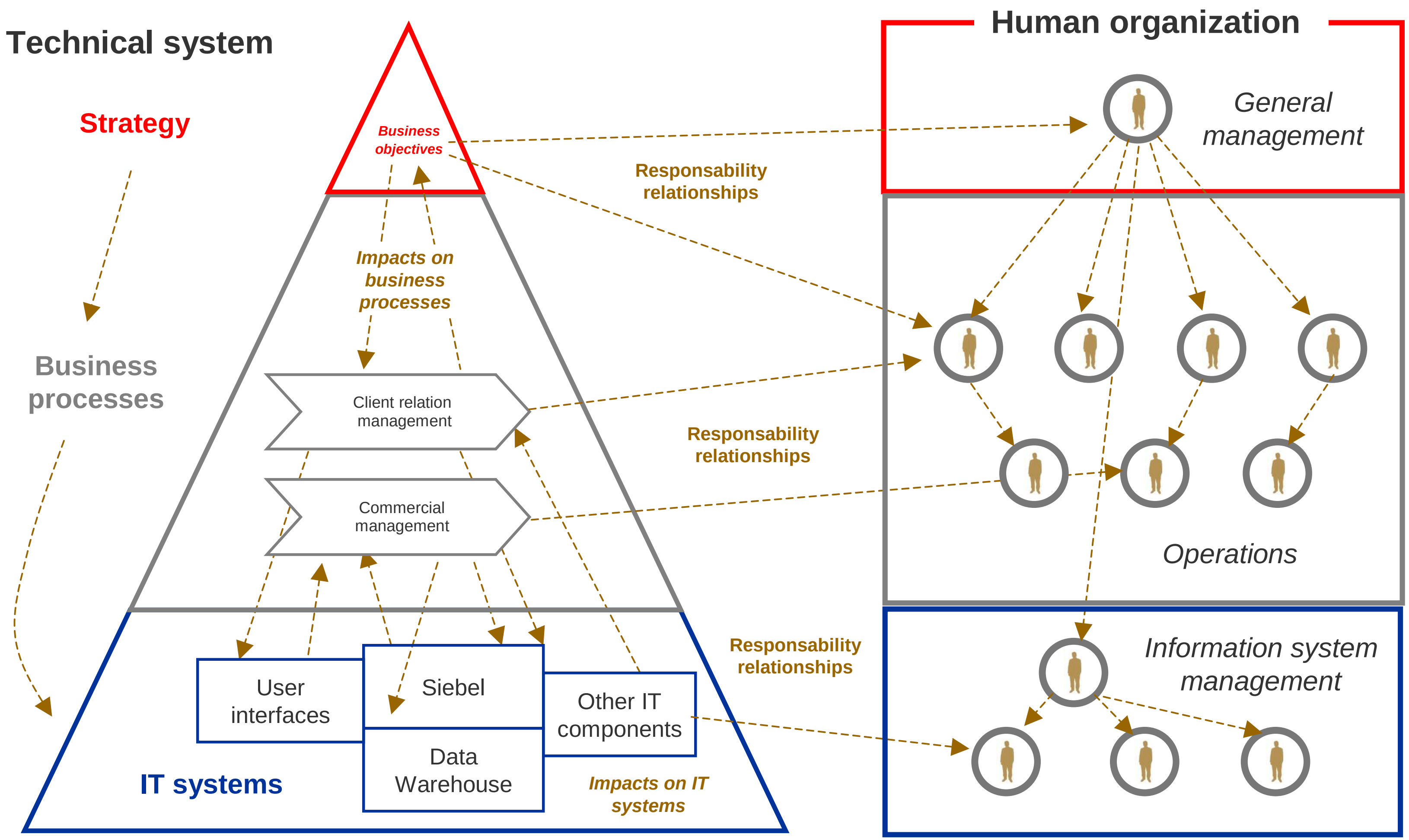
Classified according to the SysML expressivity

SysML framework



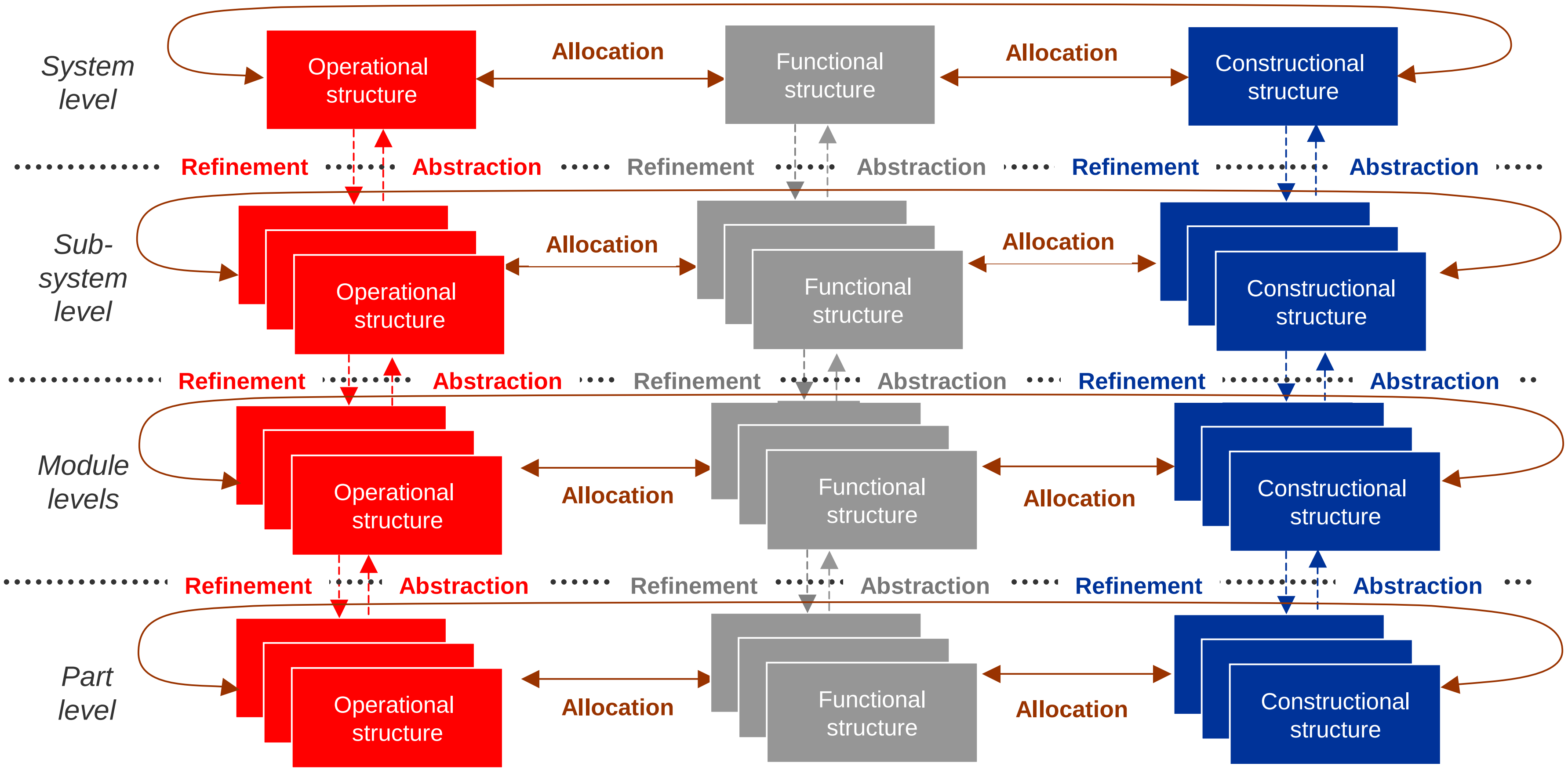
	States	Statics	Dynamics
Operational	Life cycle (behavior)	Operational contexts (mission)	Operational scenarios (scenario)
Functional	Operation modes (behavior)	Functions (operation)	Function dynamics (scenario)
Constructional	Configurations (behavior)	Resources (organization)	Resource dynamics (scenario)

Systems architecture frameworks
Example of architectural framework (3)



A classical enterprise architecture framework for information systems design

Systems architecture frameworks
Organization of a system model

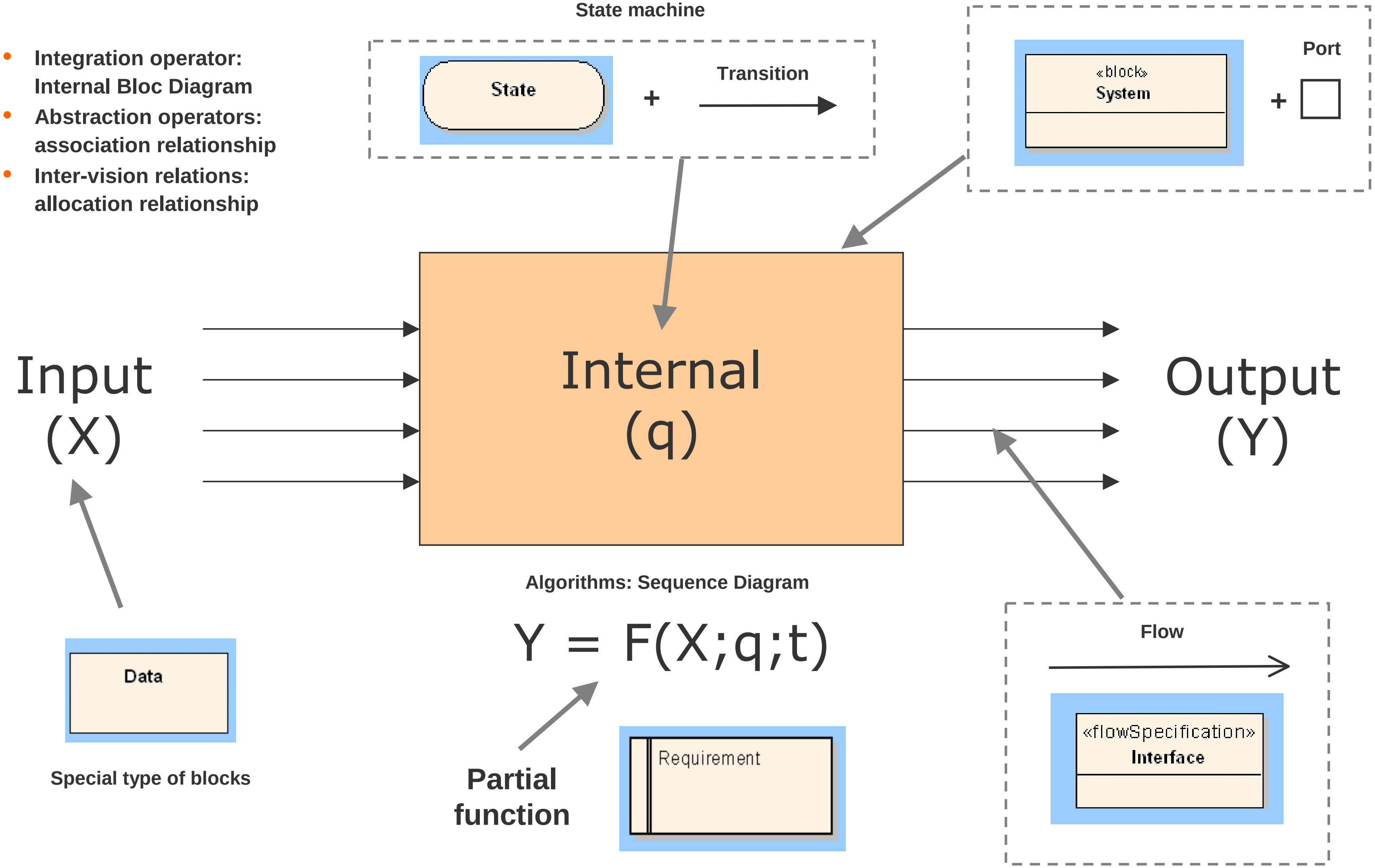


Two system modeling dimensions given respectively by the system hierarchy & the architectural framework

Contents

Systems architecture description
What should be find in a system model?

- Integration operator: Internal Bloc Diagram
- Abstraction operators: association relationship
- Inter-vision relations: allocation relationship



Architectural Visions	Requirements	States	Static structure	Dynamical behavior	Business data
Operational vision	Operational requirements (Requirements diagram)	Operational contexts synthesis (State machine)	Operational contexts (Bloc definition & internal bloc diagrams)	Operational scenarios (Sequence diagram for each operational context)	Operational data (Bloc definition diagram)
Functional vision	Functional requirements (Requirements diagram)	Functional modes synthesis (State machine)	Functional decomposition & interactions (Bloc definition & internal bloc diagrams)	Functional behaviors (Sequence diagram for each function)	Functional data (Bloc definition diagram)
Constructional vision	Constructional requirements (Requirements diagram)	Configurations synthesis (State machine)	Constructional decomposition & interactions (Bloc definition & internal bloc diagrams)	Constructional behaviors (Sequence diagram for each resource)	Constructional data (Bloc definition diagram)

Typical structure of a SysML oriented system model at a given systemic level

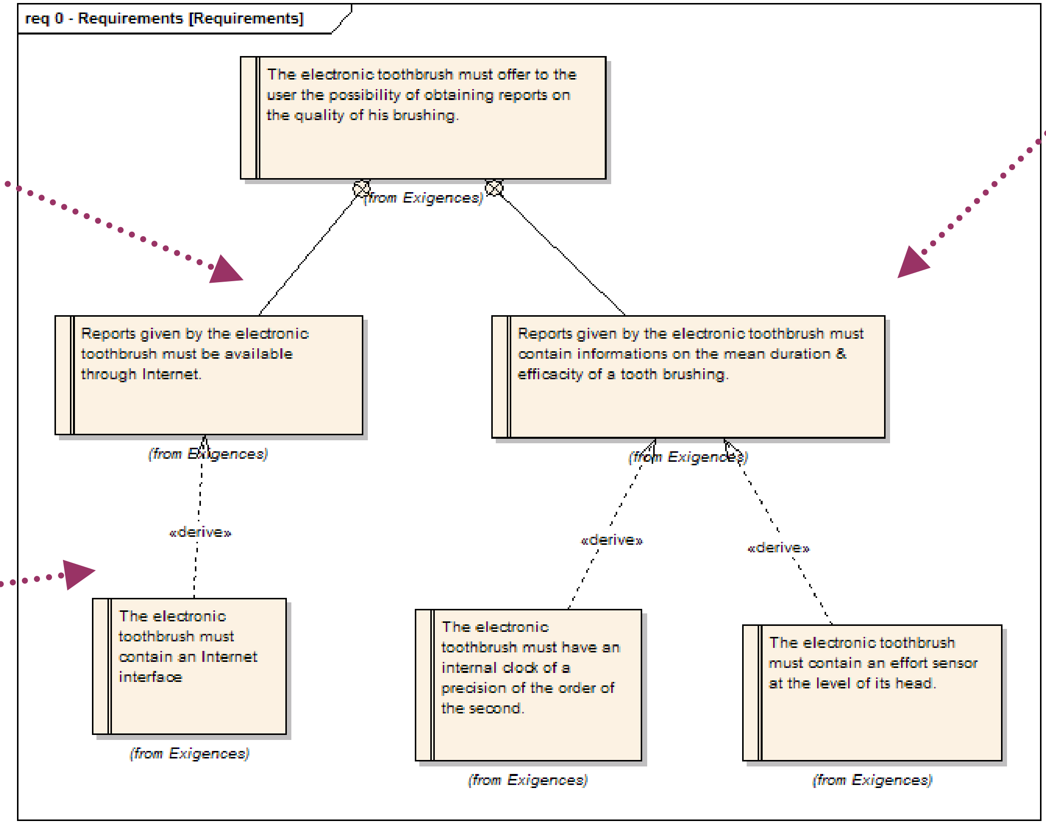
Systems architecture description
Requirements diagram

Requirements diagram
Models requirements & their interdependencies

Requirement
Models a requirement

Nesting
Models a conjonctive decomposition relationship
$(A \wedge B \rightarrow P)$

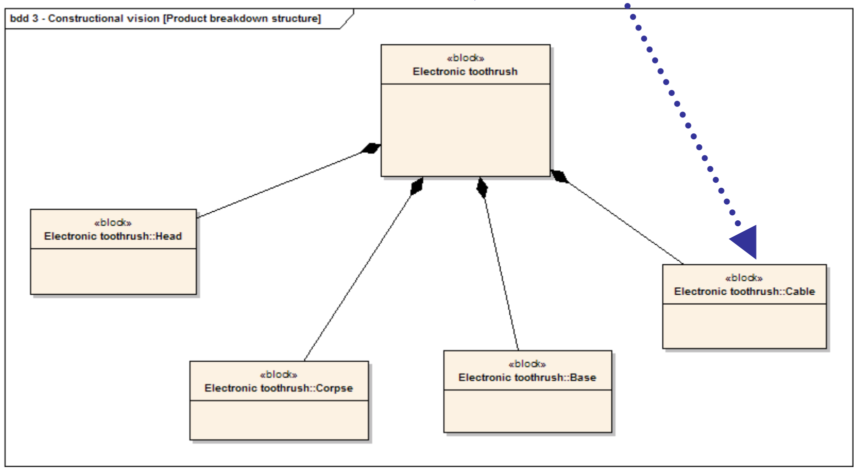
Derivation
Models a logical consequence relationship
$(P \rightarrow Q)$



Bloc definition diagram

Models a systemic organization
(here a constructional decomposition)

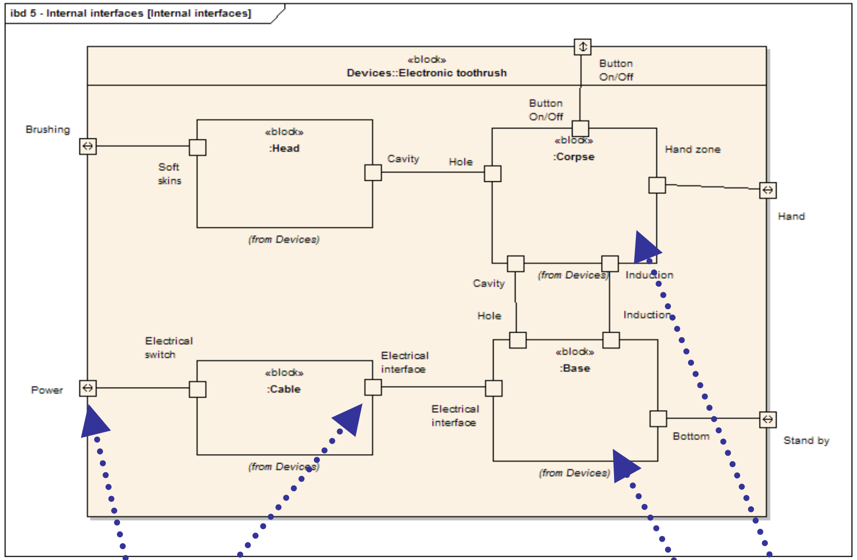
Bloc
Models a systemic component



To describe static elements & relationships

Internal bloc diagram

Models the internal relationships between different systemic components (here constructional relations)



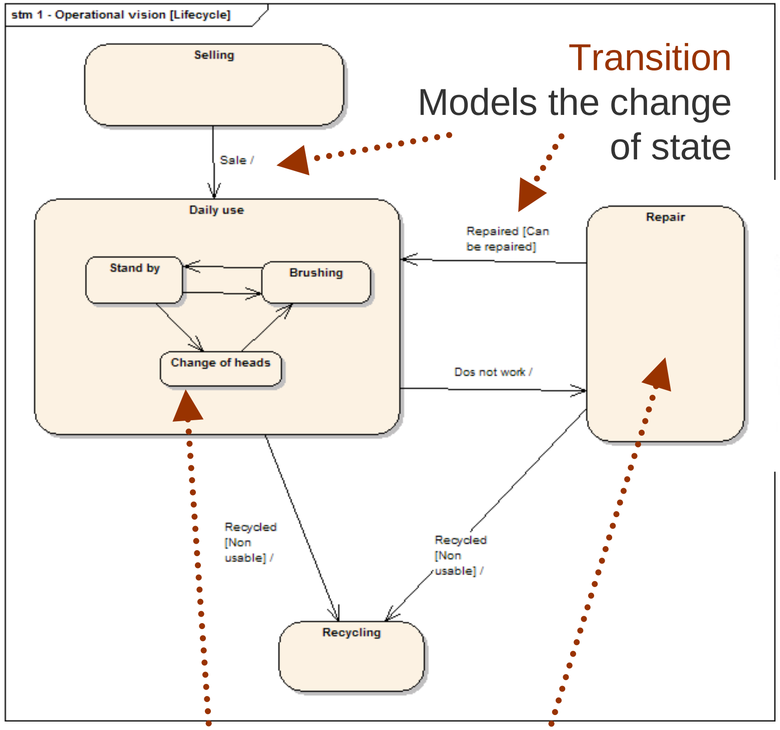
Flow port
Models interaction points

Block instance
Instances of blocs modeling specific systemic components

To express dynamic behaviors

State machine

Models the evolution of the states of a system

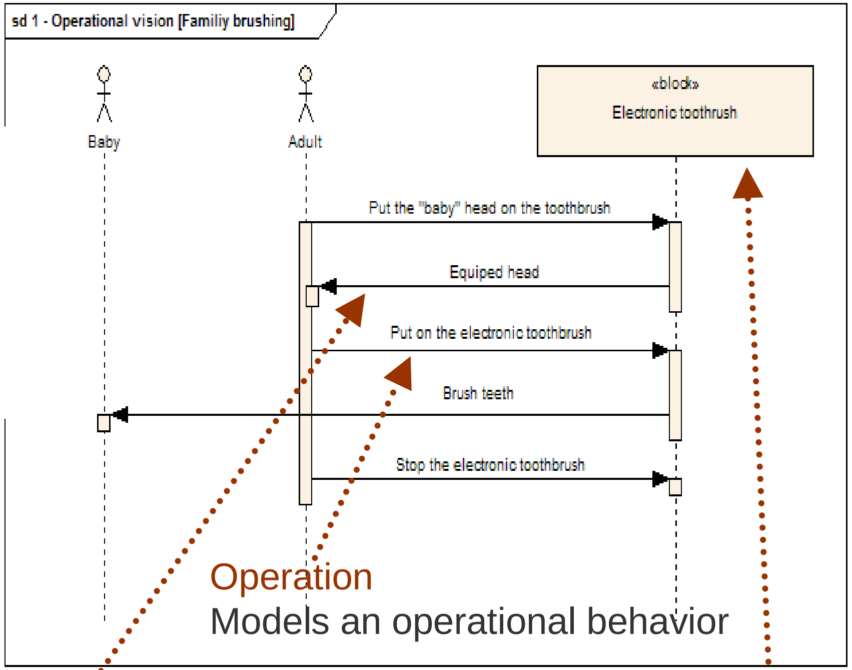


Transition
Models the change of state

State
Models an operational context, a functional mode or a constructional configuration

Sequence diagram

Models the interactions between system components



Operation
Models an operational behavior

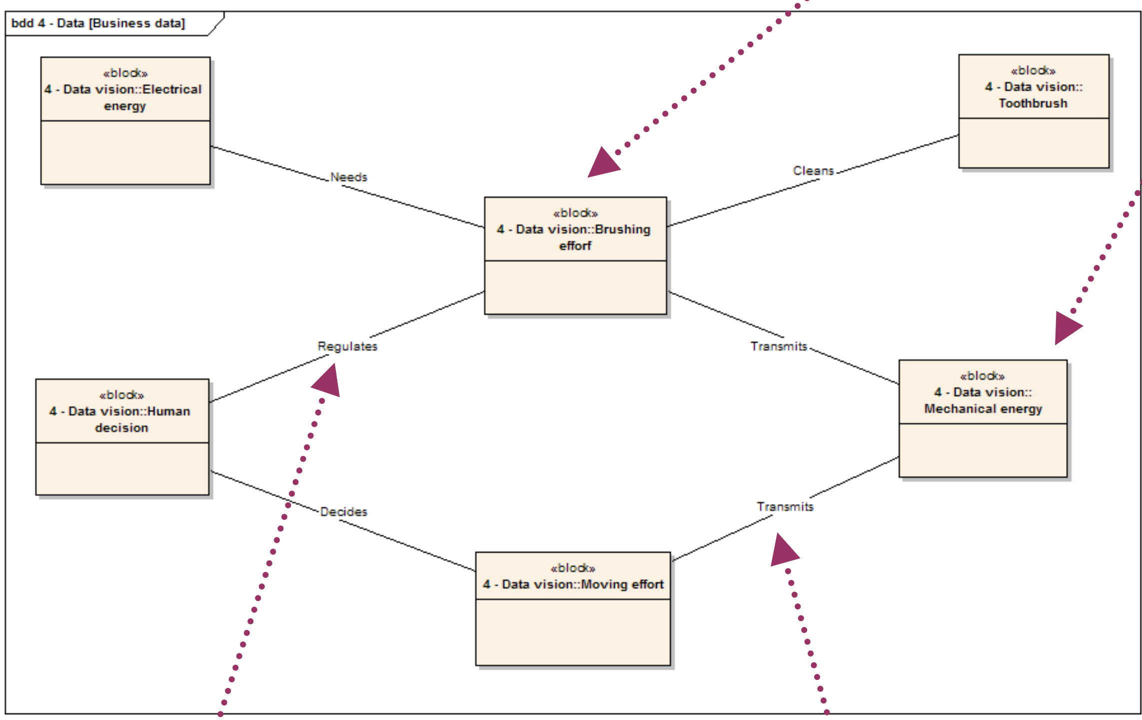
Message
Models an exchange of energy or information between two systemic components

Bloc
Models a systemic component

Bloc definition diagram

Models business data & objects managed by the system

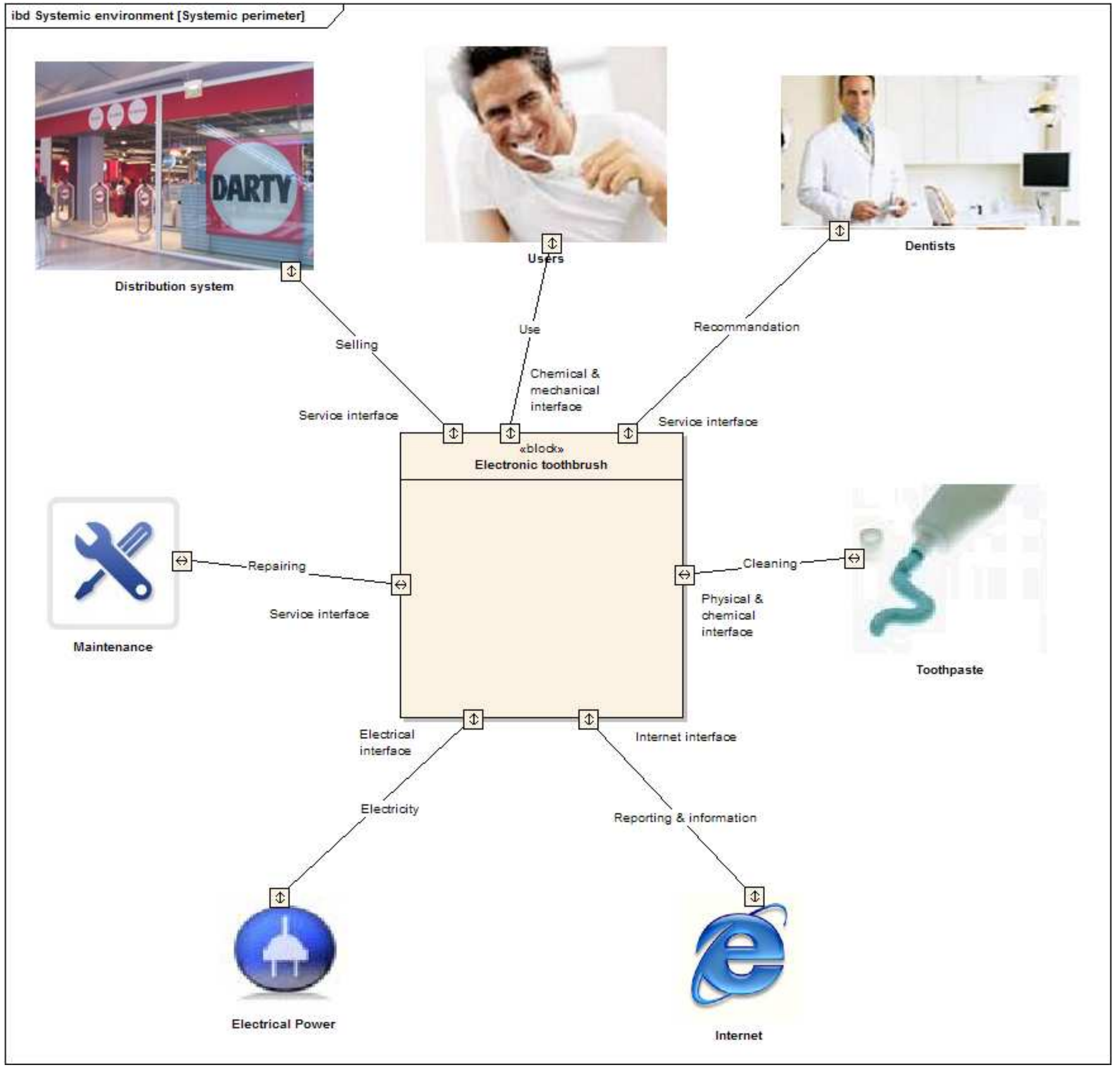
Bloc
Models a business object or data



Association
Models a relationship existing between two business data or objects

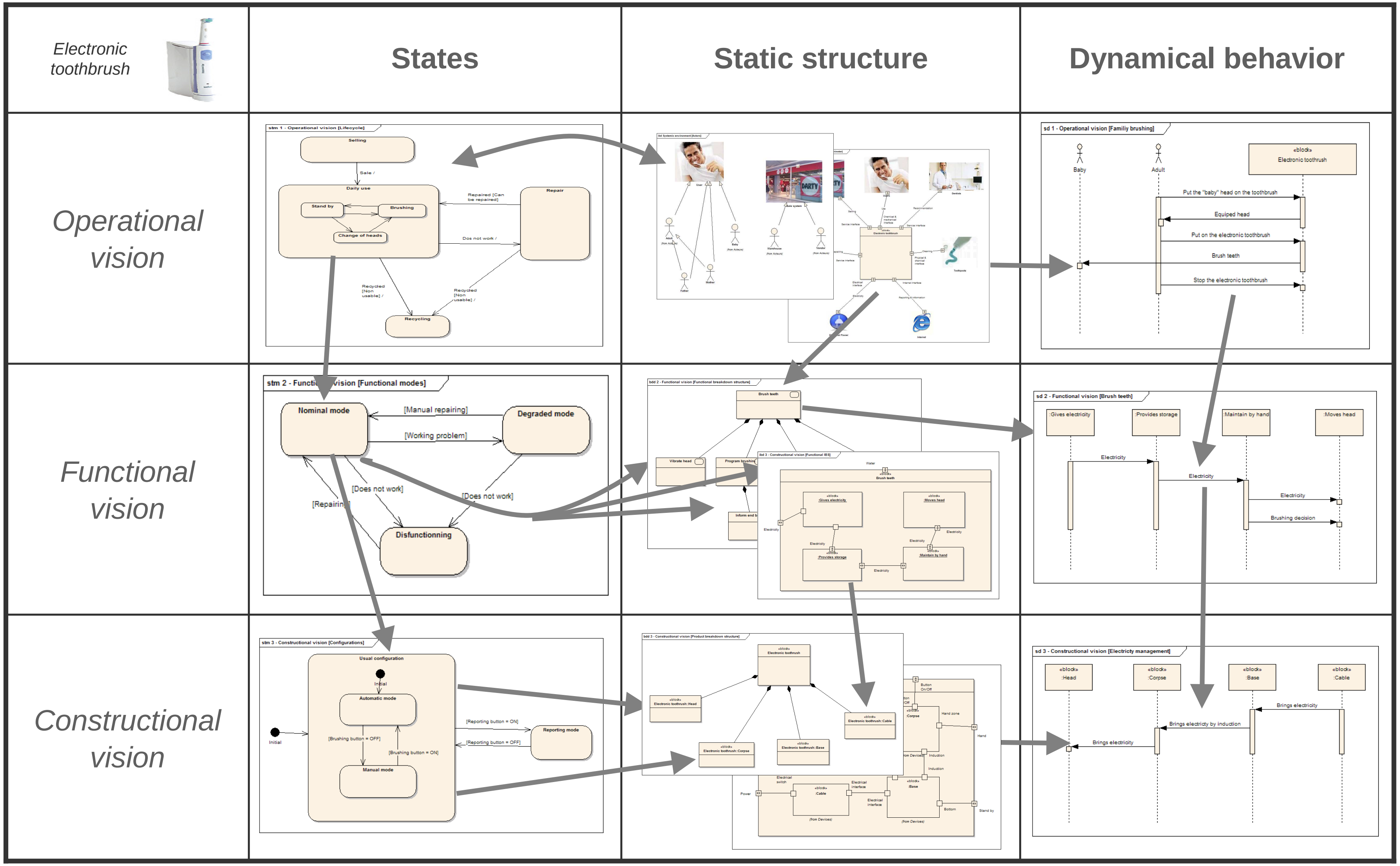
Systems architecture description
Example of a system model (1)

Electronic toothbrush



The starting point: analyzing & defining the systemic perimeter of the system studied from an operational point of view

Systems architecture description
Example of a system model (2)



Contents

ÉCOLE POLYTECHNIQUE

Contents

Design of families of systems
Abstraction is a key lever for scientific progress …

Abstraction

Modern physics — *Relativity*

Einstein relativity laws + *Light speed c is independant of the Galilean observation referential*

speed(x) << c

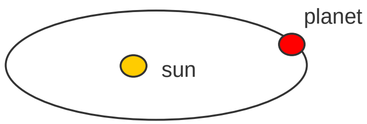
Classical Galilean relativity & Newtonian mechanics

speed(x) ≈ c

Classical physics

$F(x) = m(x)\,\overrightarrow{\gamma(x)}$ + $Attraction(a,b) = G\, m(a)m(b) / d(a,b)^2$

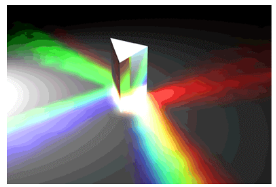
a = sun, b = planet



Galilean relativity

a = earth, x = b = apple

Weight (x) = m(x) g

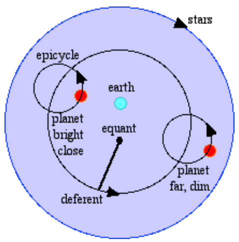


Lorentz relativity

Maxwell equations

Visible light is just a part of the electro-magnetic spectrum

Aristotle physics

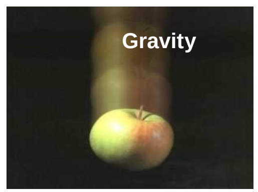


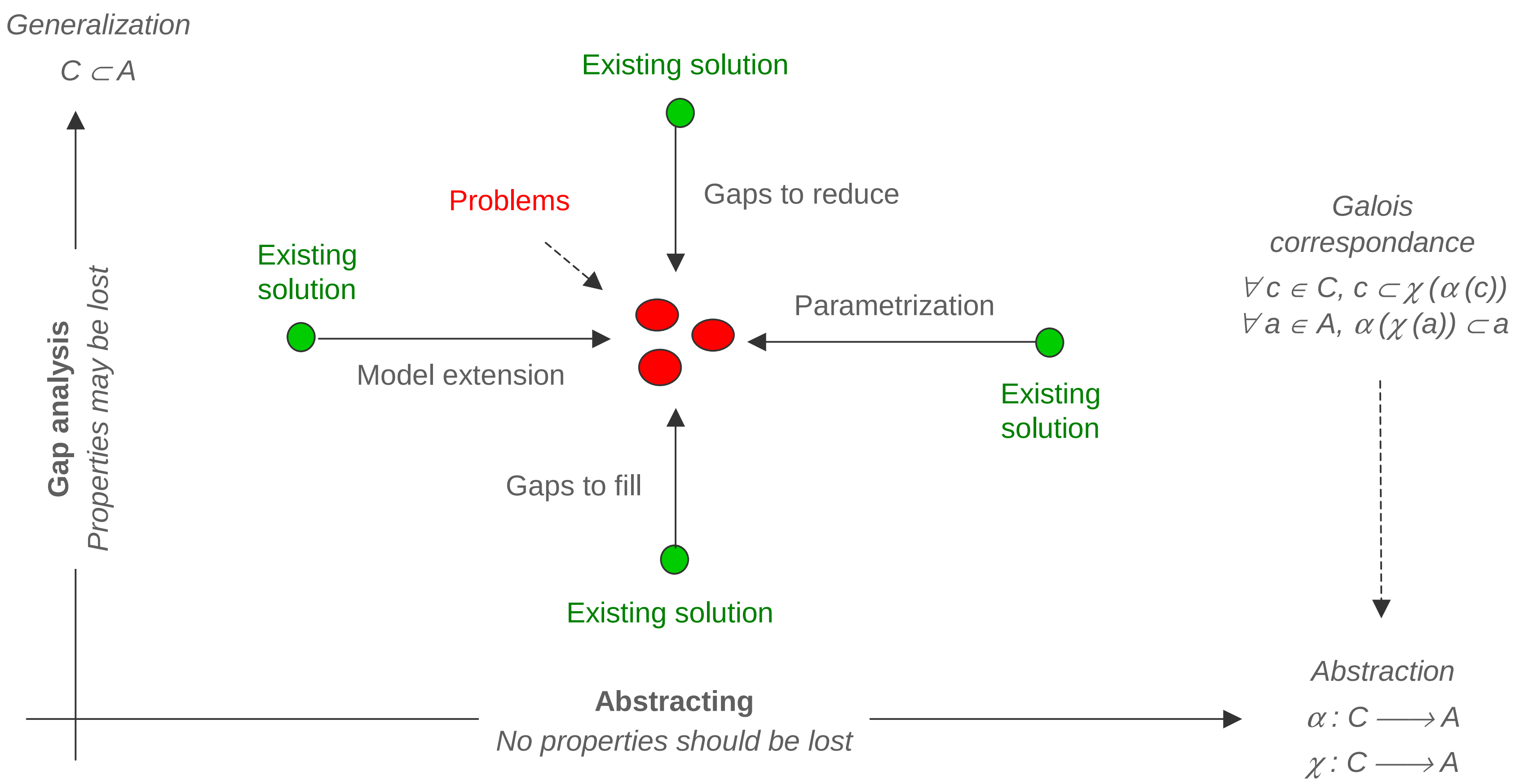
Teleonomy

An apple belongs to earth and tries therefore always to get back to the earth

Light belongs to sky and tries therefore always to get back to the sky

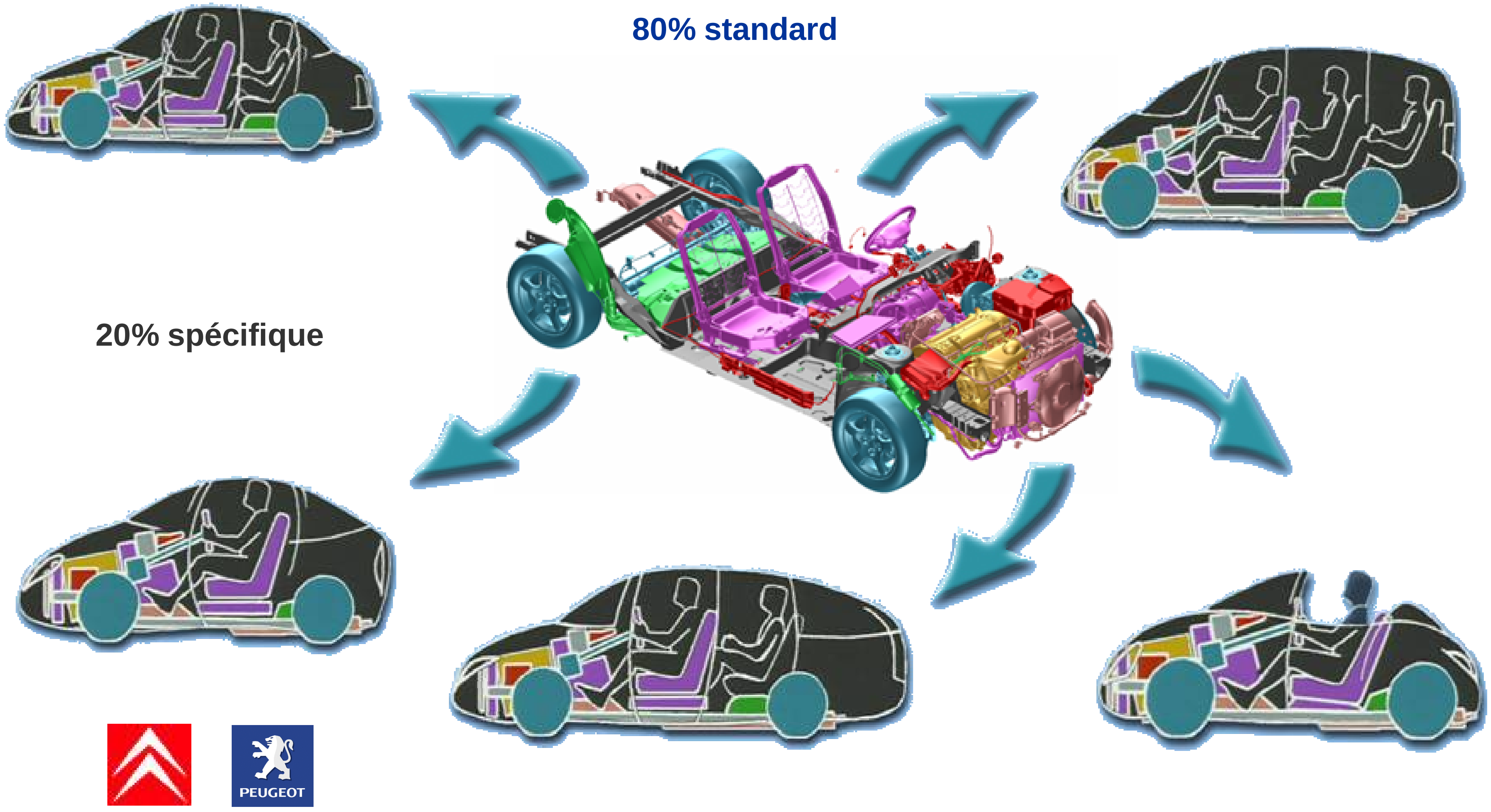
Facts





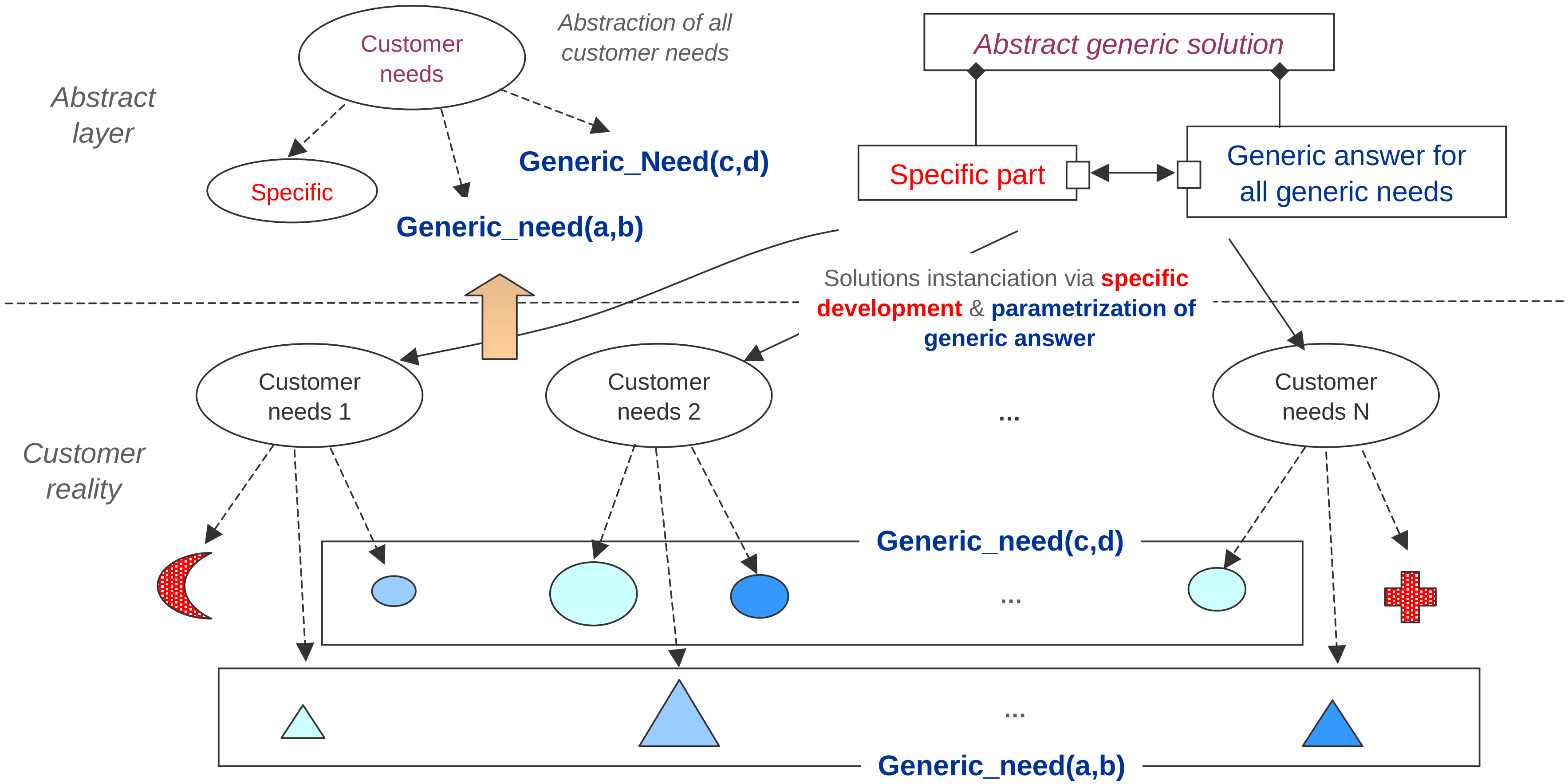
Two complementary ways for solving problems starting from known solutions

Design of families of systems
An example of families of systems



Source: N. Lartigue, PSA Peugeot Citroën, 2004

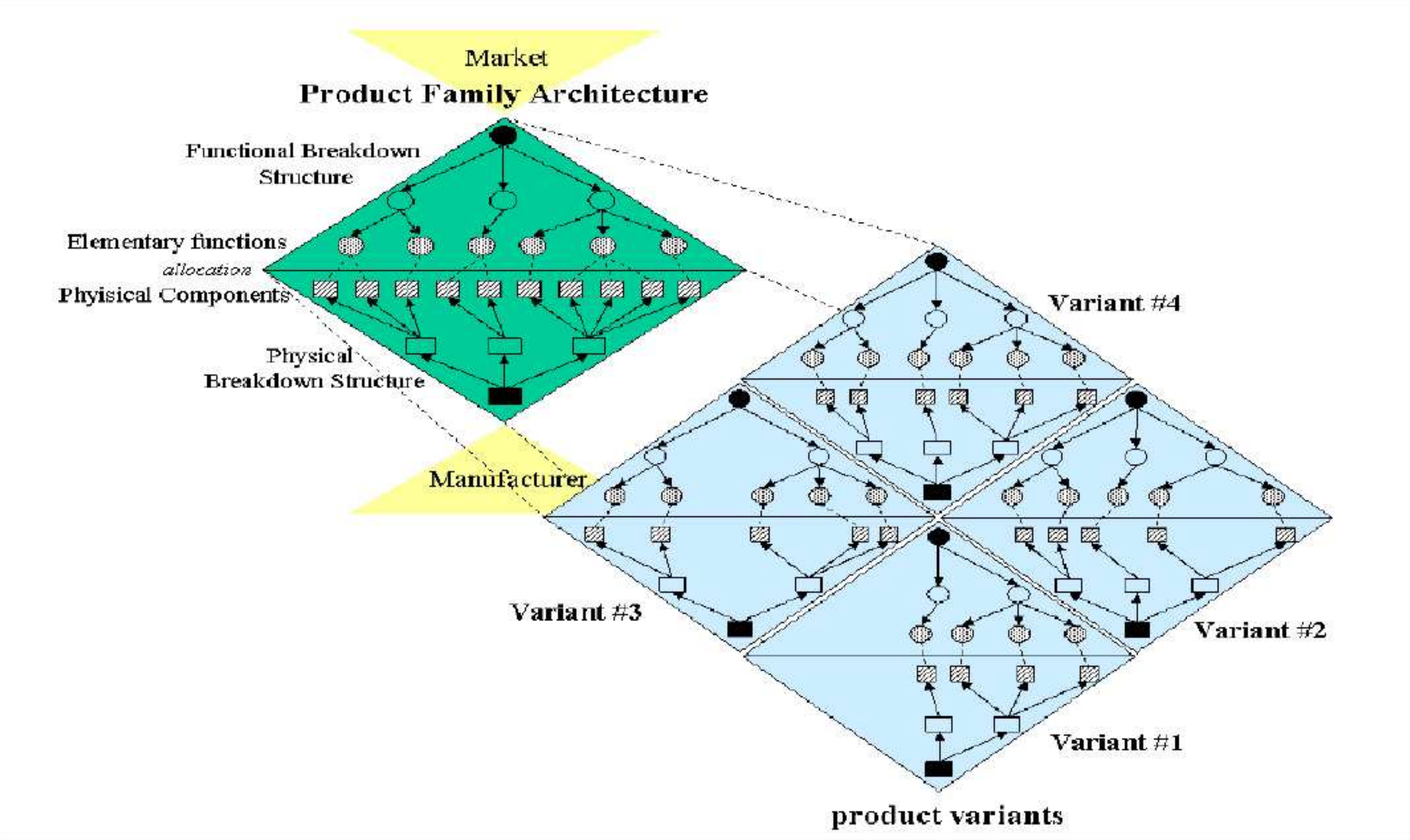
Design of families of systems
What is behind: abstract architectures



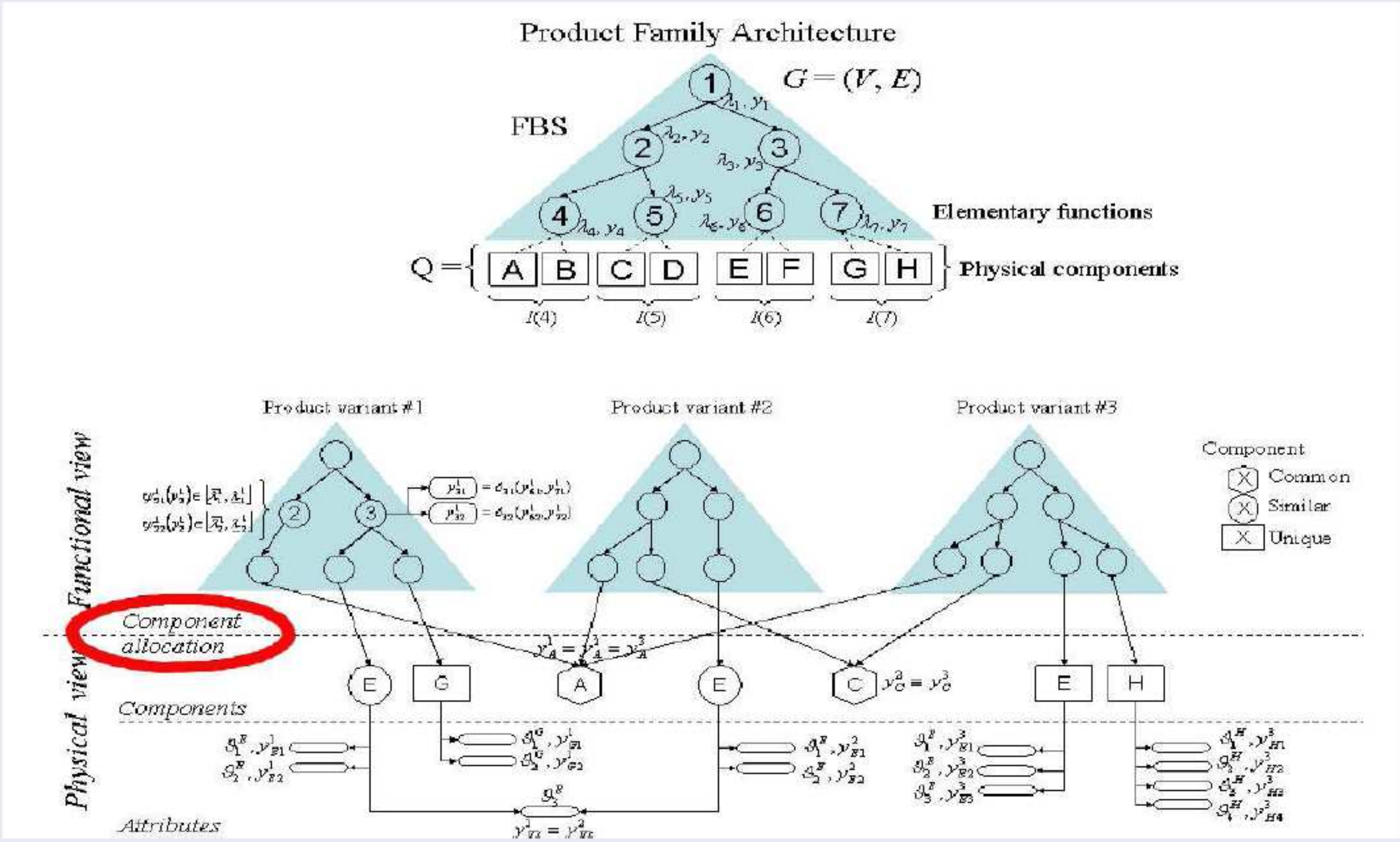
Abstraction allows to **avoid product diversity** which is the traditional problem of the traditional way of working (**one context-optimized solution per customer**)

Design of families of systems
An optimization reformulation (1)

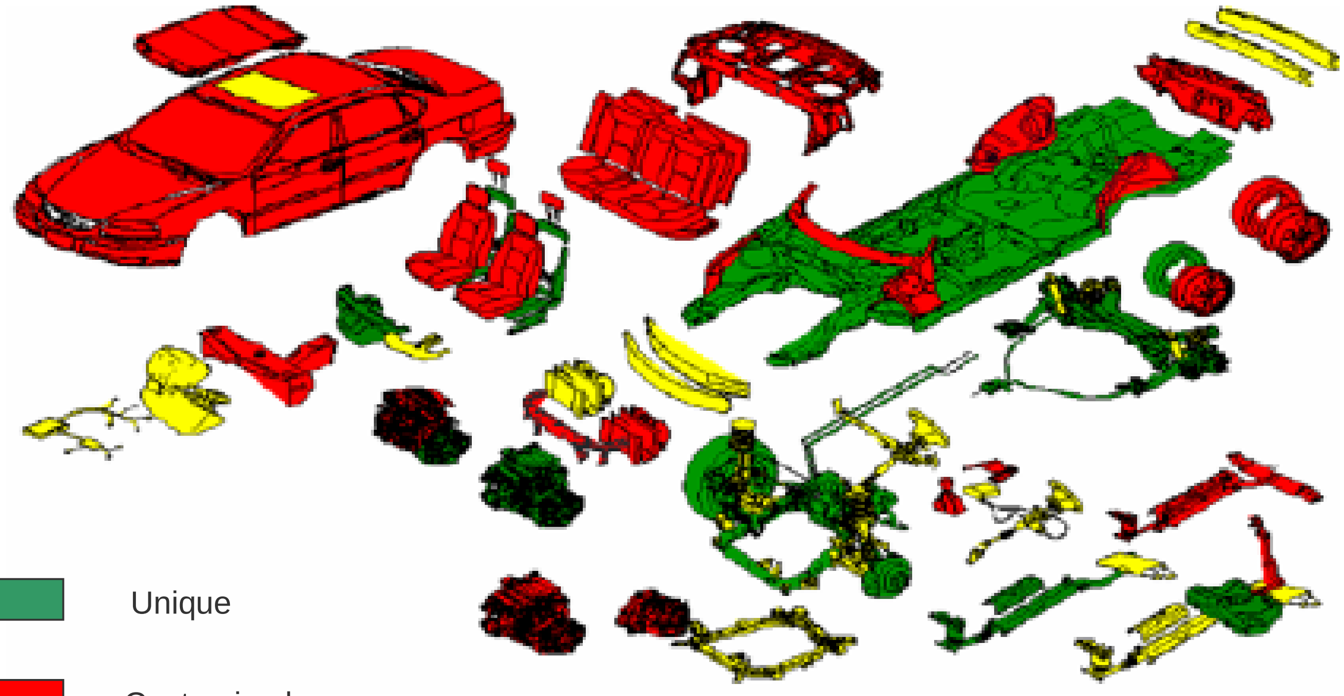
Mapping of FBS/PBS to platform



Optimal allocation w.r.t. commonality



Design of families of systems
Our hardware example revisited



Unique

Customized

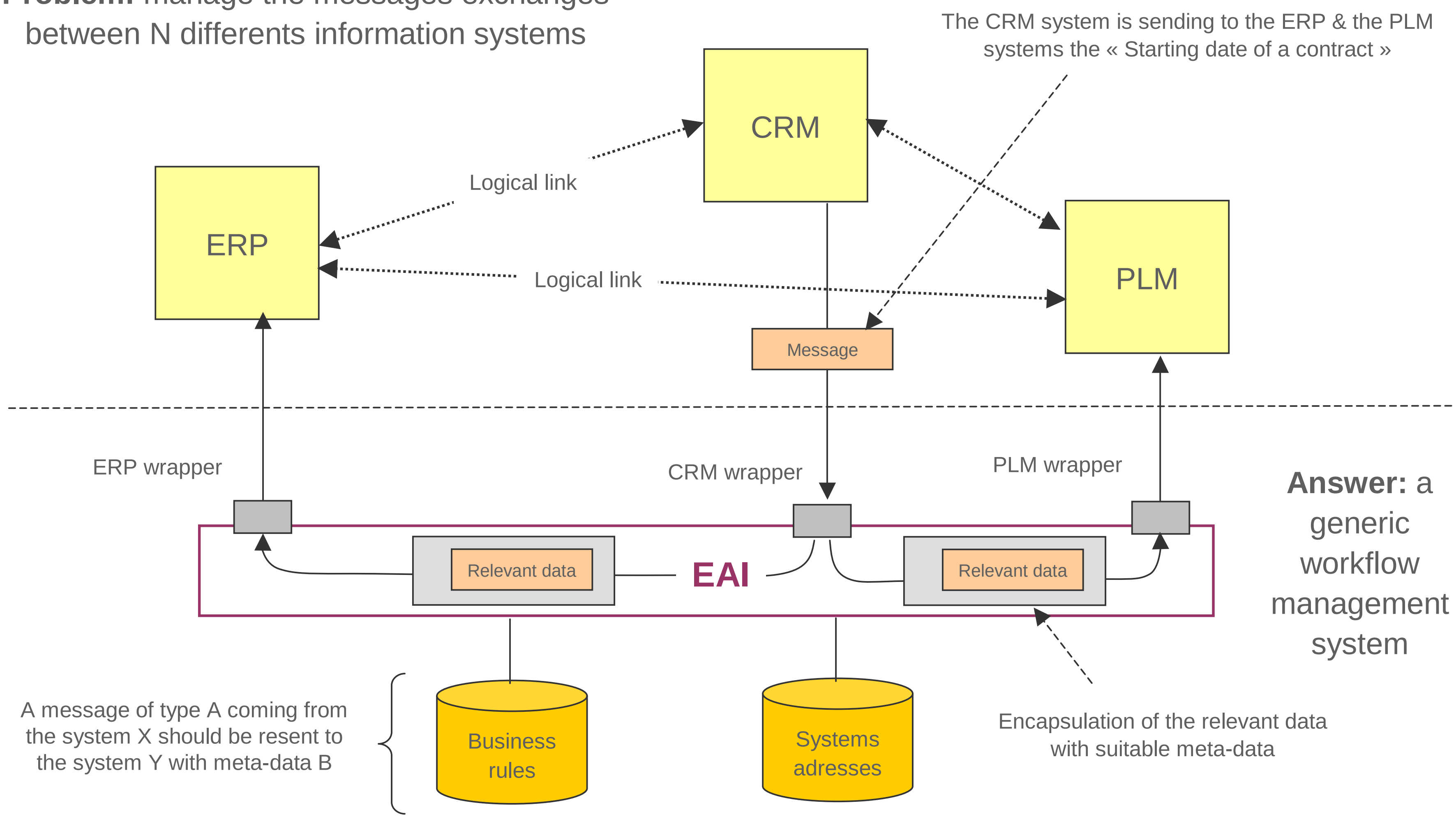
Options

Flexible architecture

How to organize a **constructional architecture** to implement the **abstraction paradigm** for **families of systems** in an hardware context already discussed

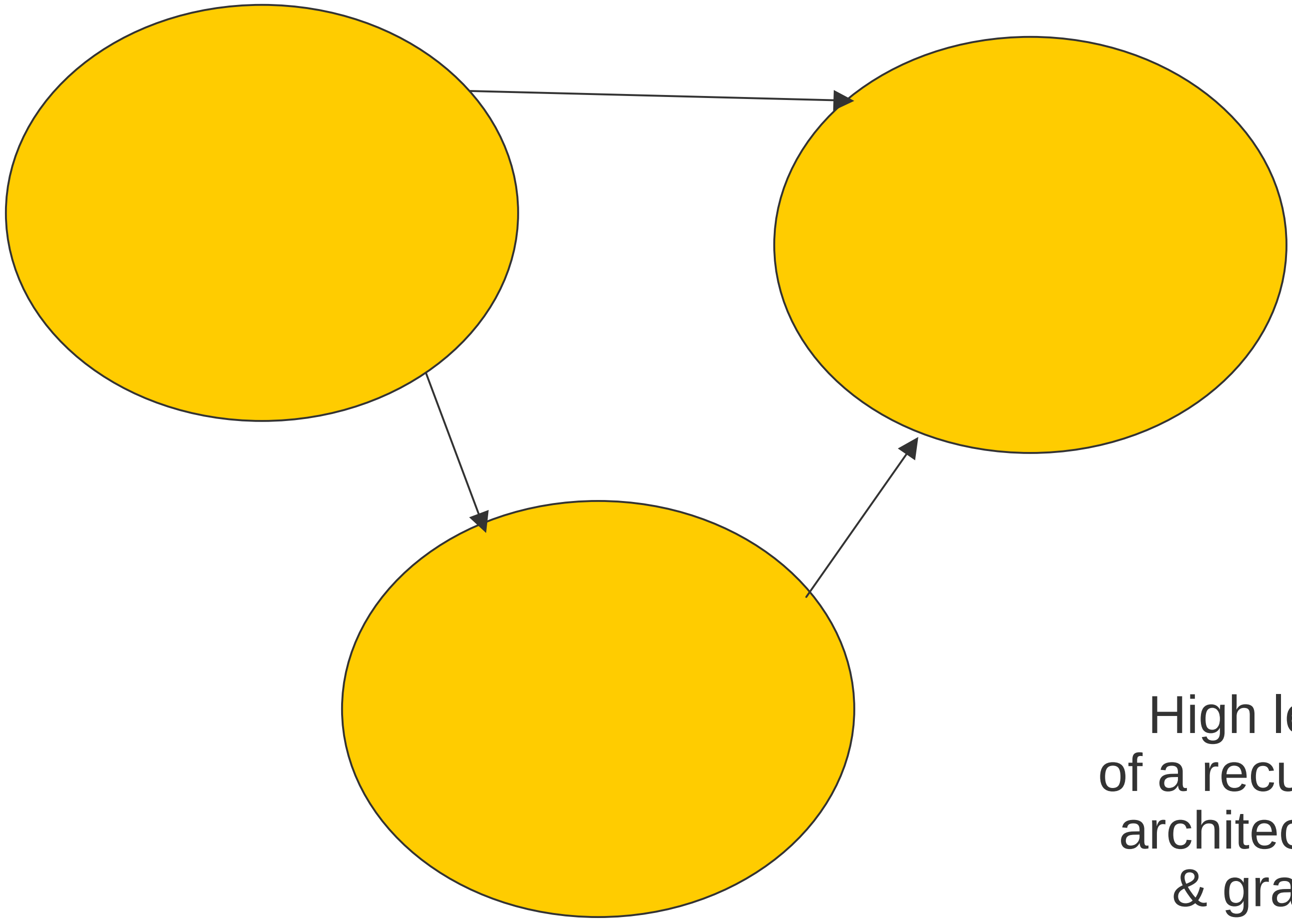
Design of families of systems
A software example

Problem: manage the messages exchanges between N differents information systems

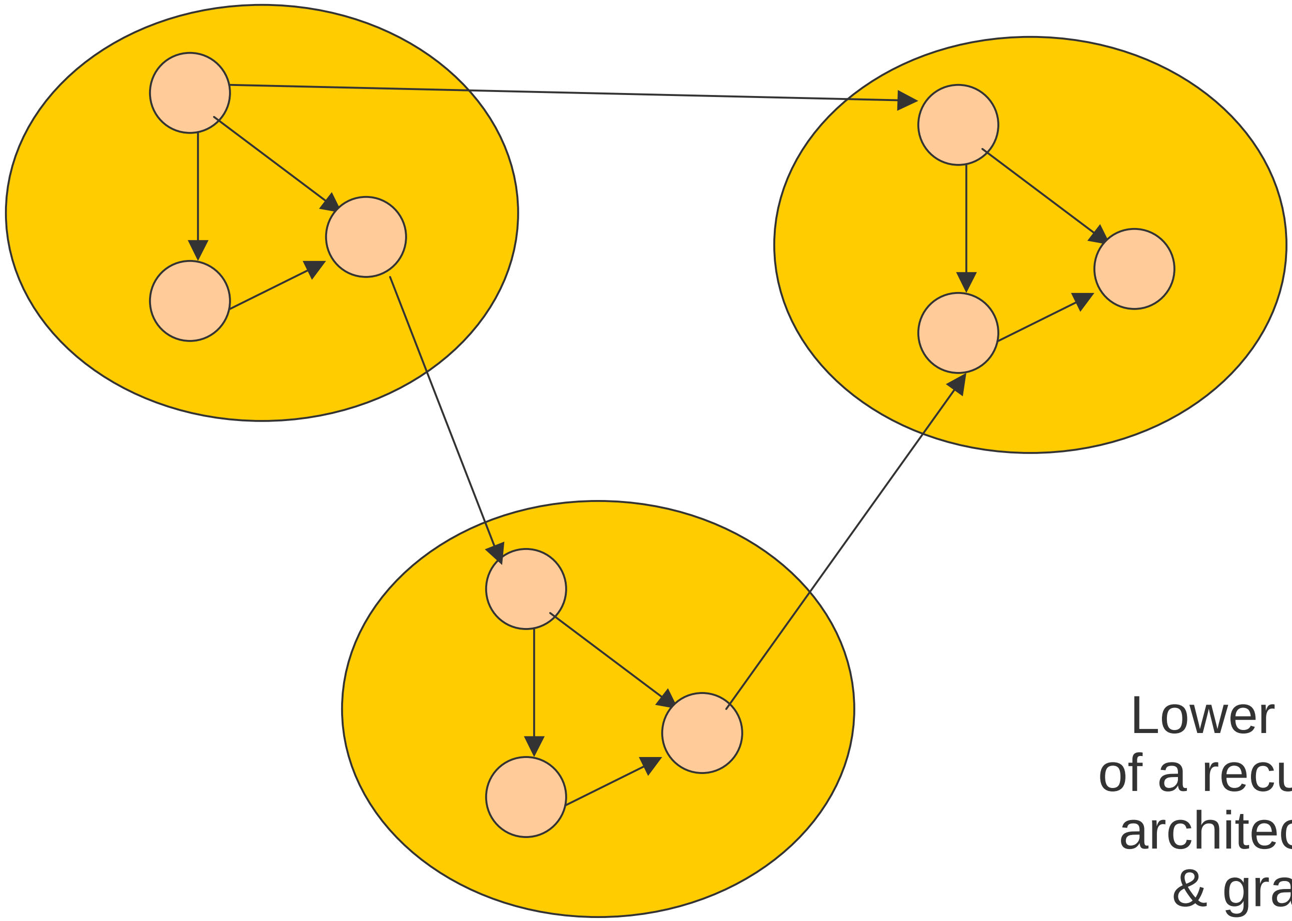


Answer: a generic workflow management system

Contents



High level
of a recursive
architecture
& graph



Lower level
of a recursive
architecture
& graph

Definition: a recursive graph G of order N is defined by setting:

- when N = 1, G is just an ordinary graph,
- when N > 1, G is a family of N usual graphs $(G_i)_{i=1..N} = (V_i,A_i)$ such that one has for every i:
  - V_i is a partition of V_{i-1},
  - $A_i \subset A_{i-1}$.
- Hence, the vertices of G of order i are obtained by partitionning the vertices of G of order i-1 when an arrow of G of order i is always an arrow of G of order i-1.

Passing from level i to level i-1 of in a recursive graph
is called zooming within the graph

- **Definition:** A weighted recursive graph is defined by:
  - a recursive graph $G = (V_i, A_i)_{i=1..N}$,
  - a family $(\pi_i)_{i=1..N}$ of vertex weight functions associated with each level of G which are defined by asking that:
    - π_1 is any vertex weight function on the lower level G_1 of G,
    - for i > 1, π_i is the vertex weight function on the i-th level G_i of G which is recursively defined by setting

$$\pi_i(V) = \sum_{x \in V} \pi_i(x) \; .$$

Weights are on the vertices !

The weight of a « gluing » of vertices is the sum of the weights of the « glued » vertices …

- **Complexity measure of a recursive graph:**
  Let G be a weighted recursive graph whose lower (and first) level $G_1 = (V_1, A_1)$ is weighted by a weight function π. The complexity measure $m_n(G)$ of G of order is then the value defined by:

$$m_n(G) = \Big(\sum_{(x_1, \ldots, x_n) \text{ path in } G_1} \pi(x_1) \ldots \pi(x_n) \Big)^{1/n}$$

- **Architectural complexity measure(s) of a system:**
  Let S be a system defined from any architectural (that can typically be operational, functional or constructional) point of view. The architectural complexity measure of order n of S is then the complexity measure of order n of the weighted recursive graph underlying to the considered architecture equiped with an initial weigthing of its primitive components which is proportional to what one wants to measure (effort, cost, etc.).

1. The following properties of the complexity measures of a recursive graph weighted by a **positive** weight function π hold:

   - One always has $m_n(G)^n \leq m_i(G)^i\, m_{n-i}(G)^{n-i}$ for every $i \geq 1$,
   - By consequence, there exist a value λ such that:

   $$m_n(G) \xrightarrow[n \to +\infty]{} \lambda$$

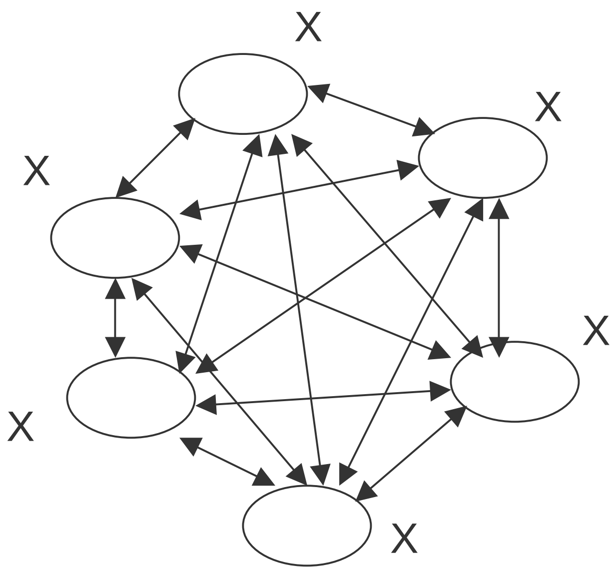
   λ can hence be interpreted as a kind of **intrinsic complexity** of G.

2. The complexity measures $m_n(G)$ are **linear**, **zoom independent** and **non destructive** which means that one has:

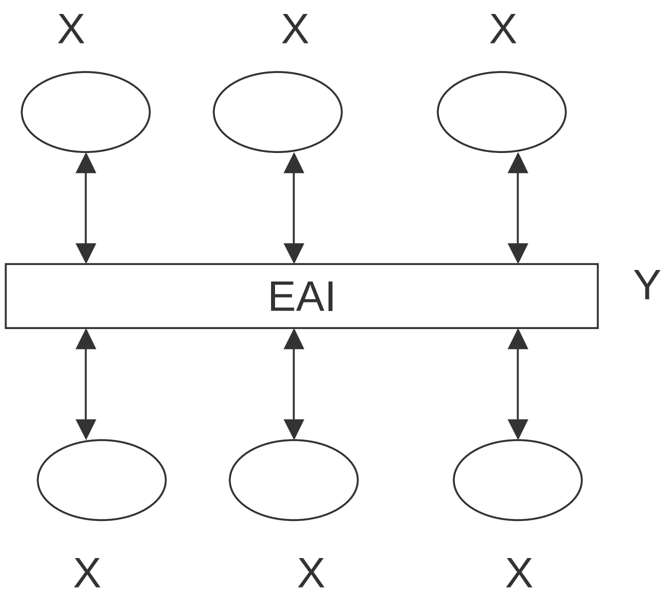
   $$m_n((G_i)_{i=1..N}) = m_n((G_i)_{i=1..N-1})$$

   when the passage from G_{N-1} to G_N is realized by creating a complete graph with all the vertices of level N-1.

In practice, our complexity measures can be used in order to compare different architectural choices for chosing the « less » complex (see below for a typical architecture comparison example in an information systems context).



A n-th order spaghetti architecture (K_n)



A n-th order EAI = (Enterprise Application Integration) architecture (K_n)

Systems complexity measures
Example of complexity measure use (2)

The limit complexity of the n-th order spaghetti architecture is:

$$\lambda(K_n) = n\,X$$

The limit complexity of the n-th order EAI architecture is:

$$\lambda(E_n) = (X+Y)/2 + (\,(X-Y)^2 + 4\,(n-1)X^2)^{1/2}$$

The EAI architecture is better than the spaghetti architecture iff one has:

$$(n^2 - 5n + 13/4)X^2 + XY(5/2 - n) - 3/4Y^2 > 0$$

Assymptotically (i.e. when n is big), the condition translates into:

$$n^2 X^2 > 5n\,X^2 + n\,XY, \text{ i.e. to } nX > 5X + Y \text{ or } (n-5)\,X > Y$$

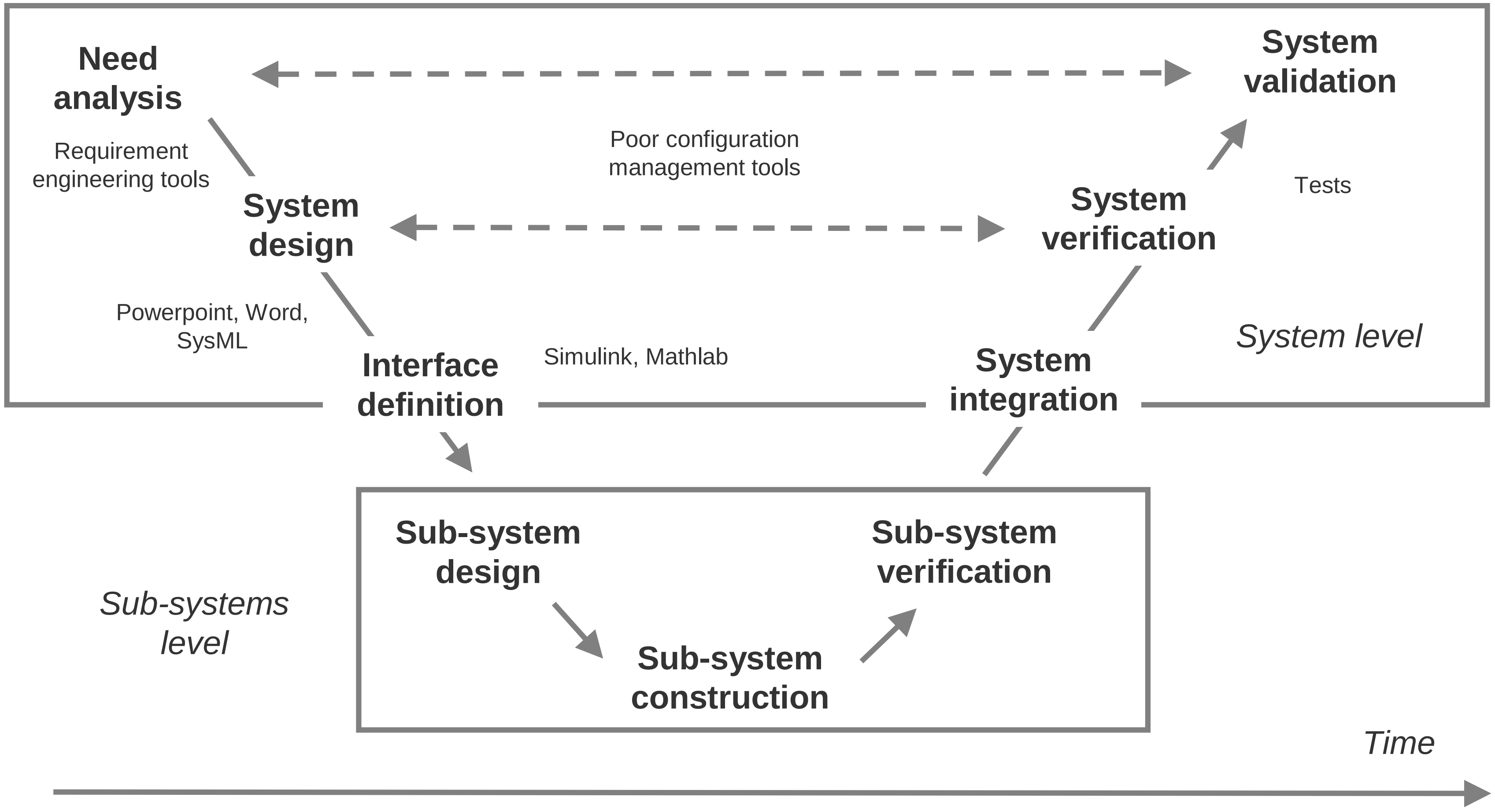
or equivalently to the following relationship between X, Y and n:

$$Y/n < X$$

Hence if the average complexity of the EAI bus per system connected to the bus is strictly less that the complexity of the systems, it is a good choice to choose an EAI architecture (which seems reasonable).

Contents

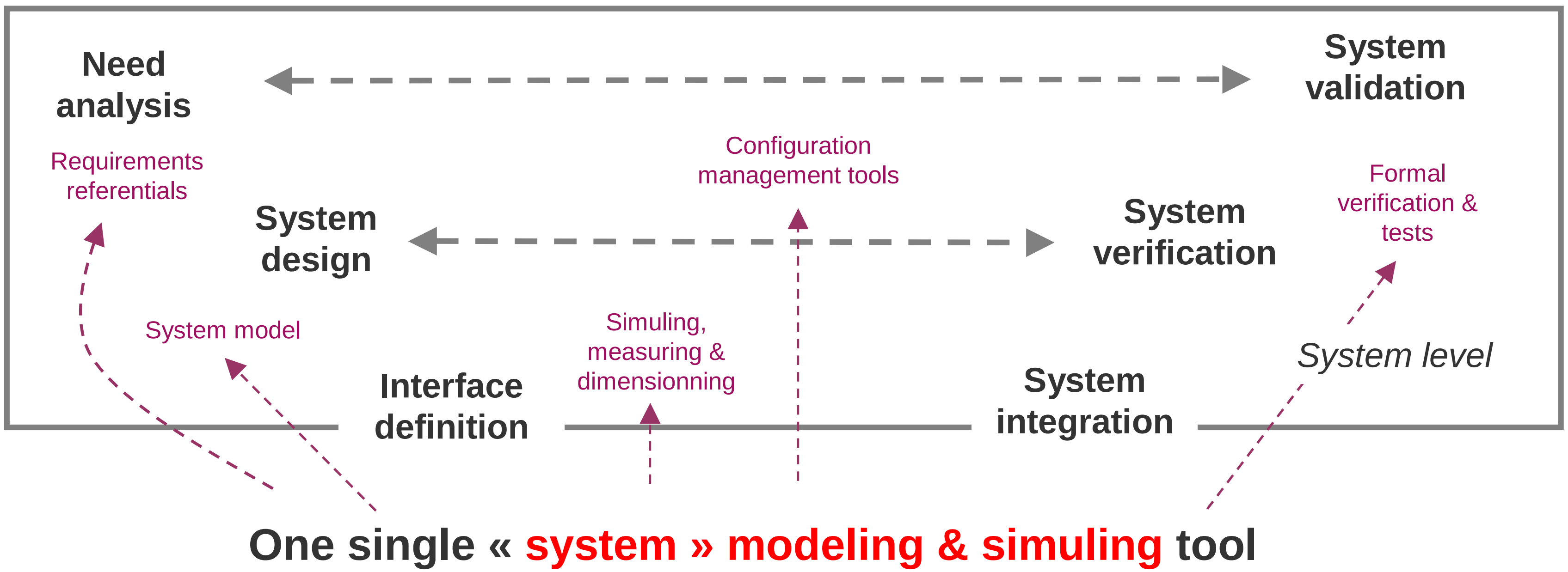
Conclusion
What is the main system architecture challenge? (1)



Key engineering challenge: a smooth design process at system level

Conclusion
What is the main system architecture challenge? (2)

- **Key theoretical challenge: constructing an unified system theory based on an architectural perspective …**
  - An unified formal behavioral system model
  - An unified formal point of view on architectural frameworks
- **… leading to unified system modeling tools at system level**



End of the course