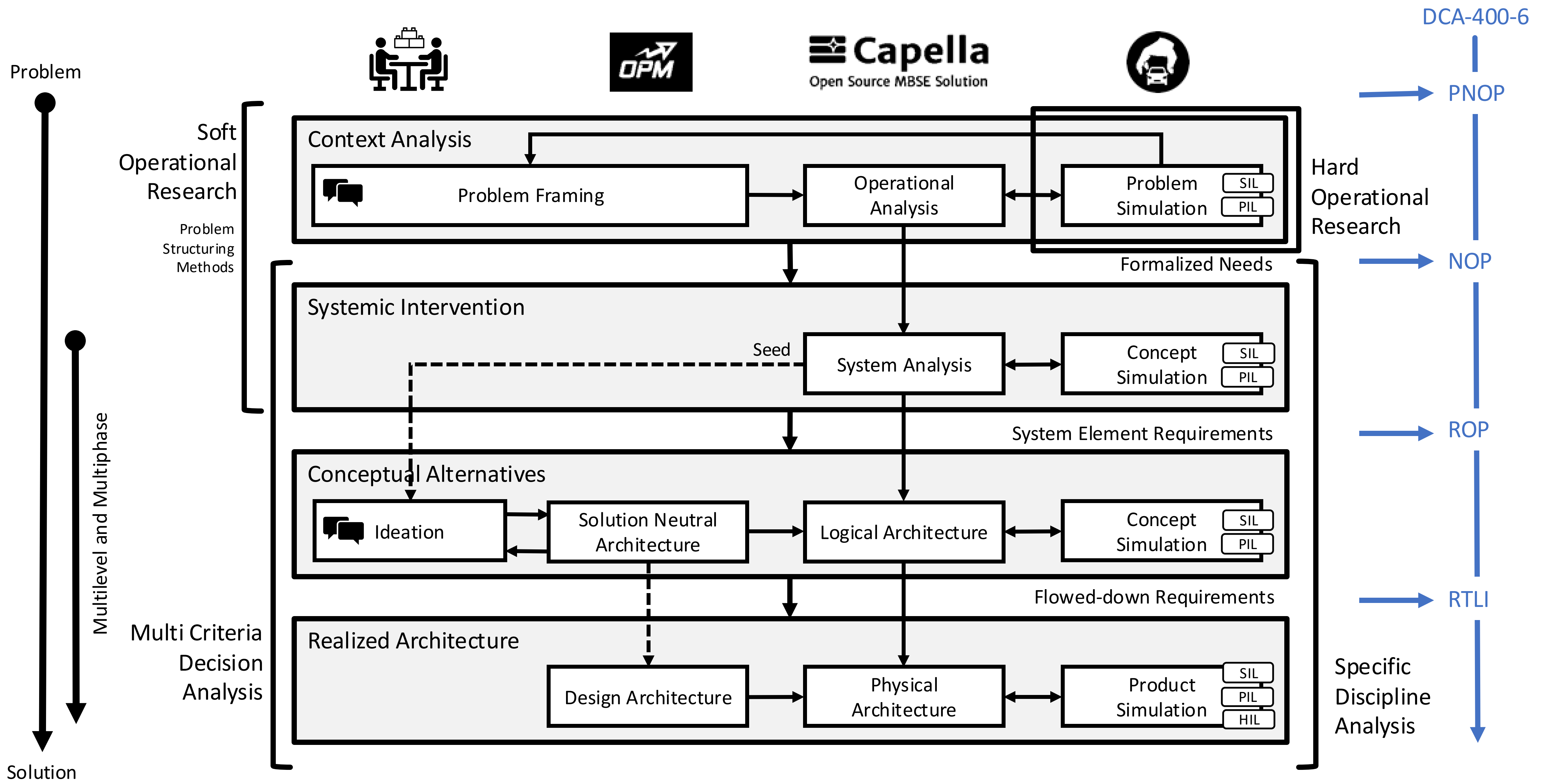
ARQUITETURA CONCRETA

SEMANA		TEORIA	INDIVIDUAL	PESO	GRUPO	PESO
1	1	Estrutura e Filosofia do Curso	AI-01 - Resumo Cap 1 - HB INCOSE	10%		
05-Aug	1	O que é Engenharia de Sistemas? INCOSE				
	1	Elementos da Eng Sis.				
	1	Introdução aos diagrams clássicos.				
2		* (Viagem ao EUA)	AI-02 - Leitura/Resumo paper sobre representações clássicas.	10%		
12-Aug						
3		* (Viagem ao EUA)	AI-03 - Exercício sobre arquitetura e escrita de requisitos.	10%		
19-Aug						
4	1	Metodologias de MBSE e uso de modelos.	AI-04 - Resumo Artigo de Metodologias	10%		
26-Aug	1	Revisão de UML-SysML.				
	1	OPM				
	1	Arcadia				
5	1	OPM	AI-05 - Lista de exercícios	10%		
02-Sep	1					
	1					
	1					
6	1	Blocos e Classes	AI-06 - Lista de Exercícios	20%		
09-Sep	1					
	1	Máquina de Estados				
	1					
7	1	Casos de Uso	AI-07 - Lista de Exercícios	20%		
16-Sep	1					
	1	Sequência				
	1					
8	1	Integração dos pontos de vistas em um	AI-08 - Resumo sobre Ciclo de Vida de Modelos	10%	AI-08 - Descrição e Contorno do Problema.	100%
23-Sep	1	Associação dos artefatos de SE com modelos				
	1	Análise Operacional				
	1					
				100%		**100%**
SEM						
30-Sep						

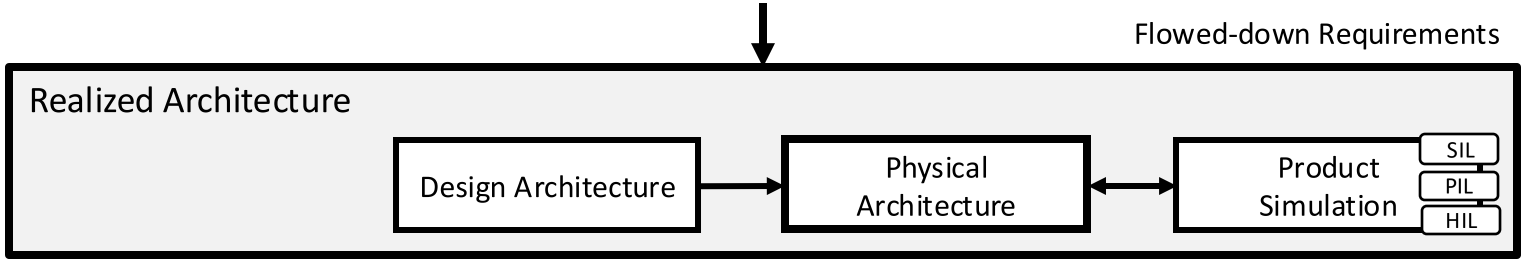
SEMANA		TEORIA	INDIVIDUAL	PESO	GRUPO	PESO
9	1	Apresentação das necessidades			AG-09 - Apresentação Necessidades	20%
07-Oct	1	Intervenção Sistêmica				
	1	Associação com Requisitos				
	1					
10	1	Apresentação da Arq e Req de sistema	AI-10 - Exercícios de Arquitetura Funcional	20%	AG-10- Apresentação Arq / Caixa Preta	20%
14-Oct	1	Conceitos de Arquitetura Funcional				
	1	Arquitetura Conceitual				
	1					
11	1	Utilização de modelos para outros processos			AG-11 - Geração de documentos	10%
21-Oct	1					
	1	Exportação automática de documentos				
	1					
12	1	Apresentação da arquitetura Conceitual	AI-12 - Explorar RCE lendo arquivo do Capella	20%	AG-12 - Apresentação Arq. Conceitual e Proposta de VV	20%
28-Oct	1	Co-Engineering / CDF / RCE				
	1	Arquitetura Concreta				
	1					
13		* (ADS-HLG)	AG.13 - Explorar Plugin M2DOC (extra)	20%		
04-Nov						
14		* (ADS-HLG)	AG-14 - Explorar Plug in P4C (extra)	20%		
11-Nov						
15	1	Metamodelo	AG=5 - Figura do Metamodelo	20%	AG-15 = Relatório de Proposta de plugin	20%
18-Nov	1	Capella Studio - Criação de plugins				
	1					
	1					
16	1	Apresentação final			AG-16 - Apresentação do Projeto Completo	20%
25-Nov	1					
	1					
	1	Encerramento do Curso				
				100%		**110%**
EXAME *02-Dec* *13-Dec*		**Grupo: Apresentação / Relatório / Gravação / Código de um: plugin ou doc**				**100%**

MMMF



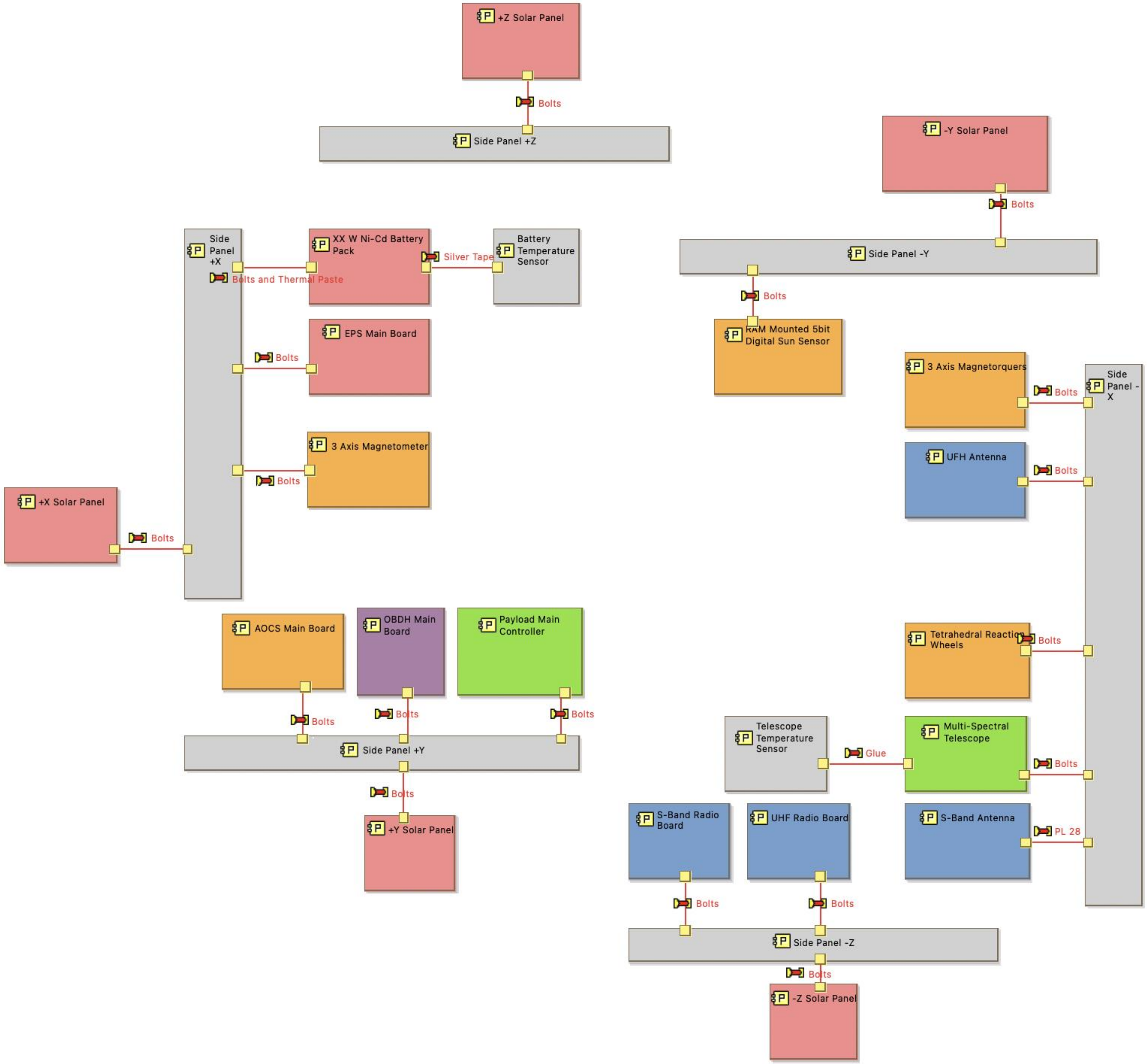
DCA-400-6
Problem
Soft Operational Research
Problem Structuring Methods
Capella
Open Source MBSE Solution
OPM
Context Analysis
Problem Framing
Operational Analysis
Problem Simulation
SIL
PIL
Hard Operational Research
PNOP
Formalized Needs
NOP
Systemic Intervention
Seed
System Analysis
Concept Simulation
SIL
PIL
System Element Requirements
ROP
Conceptual Alternatives
Ideation
Solution Neutral Architecture
Logical Architecture
Concept Simulation
SIL
PIL
Flowed-down Requirements
RTLI
Multilevel and Multiphase
Multi Criteria Decision Analysis
Realized Architecture
Design Architecture
Physical Architecture
Product Simulation
SIL
PIL
HIL
Specific Discipline Analysis
Solution

Example

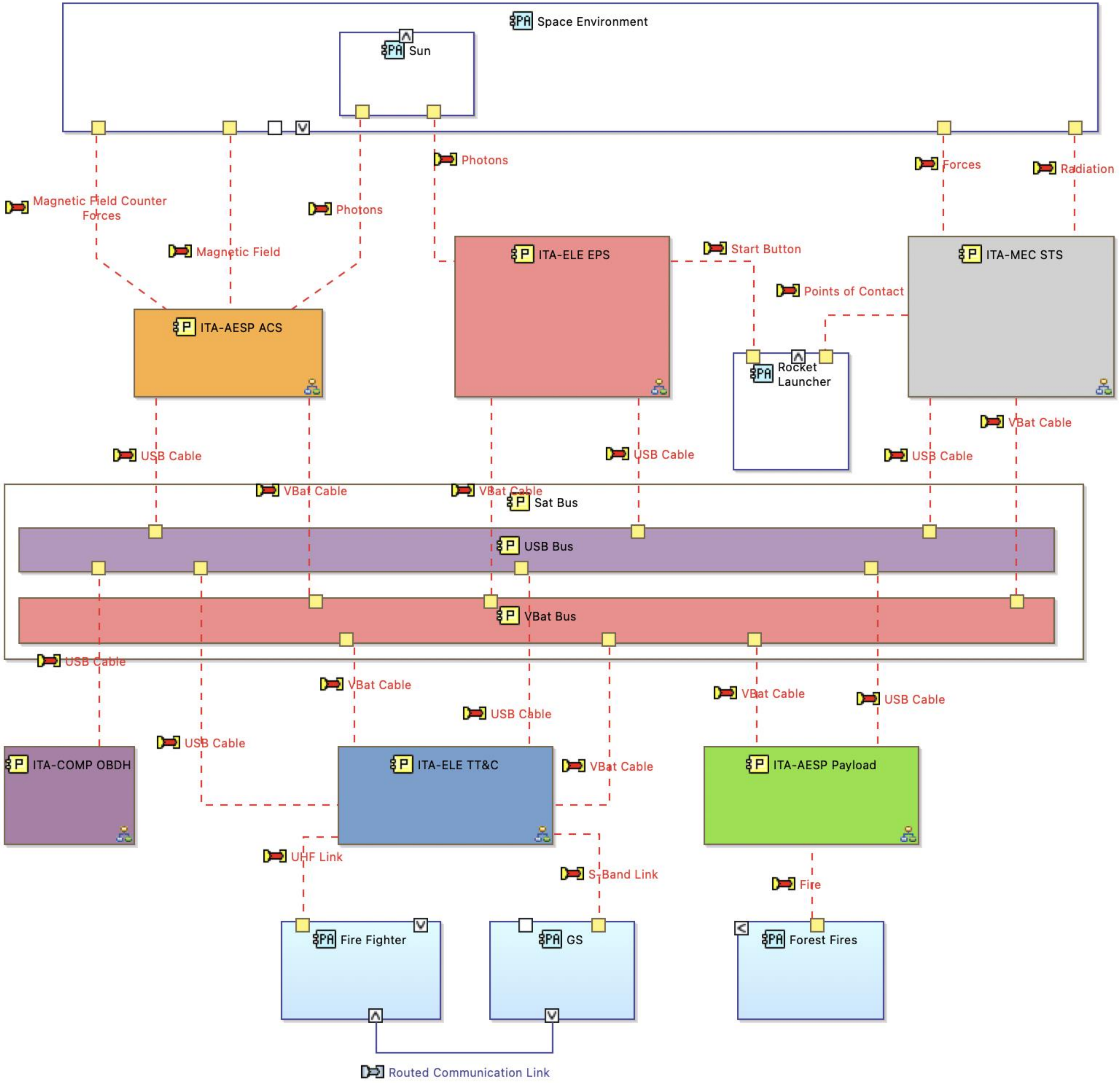


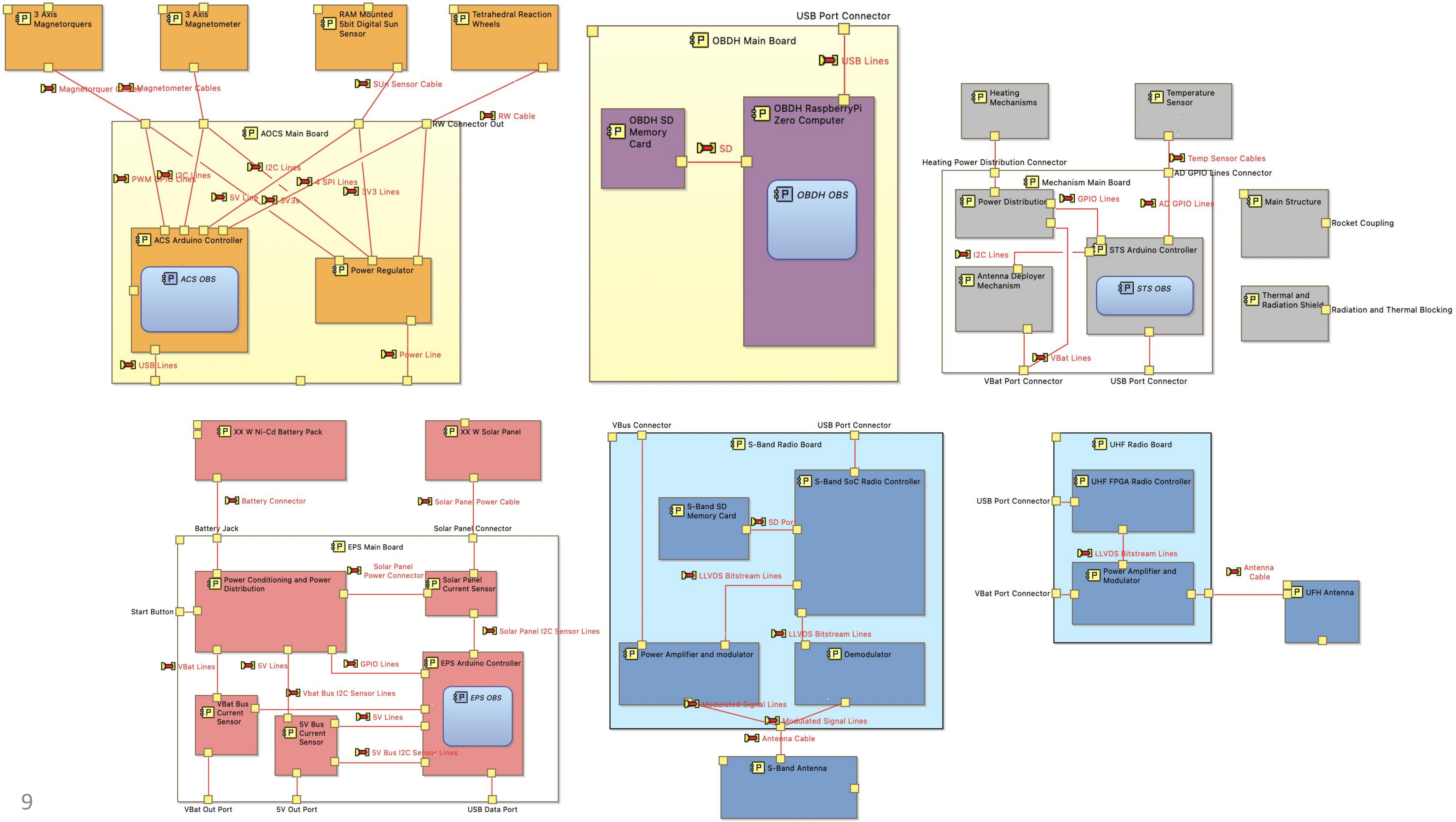
Specific Discipline Analysis

RTLI



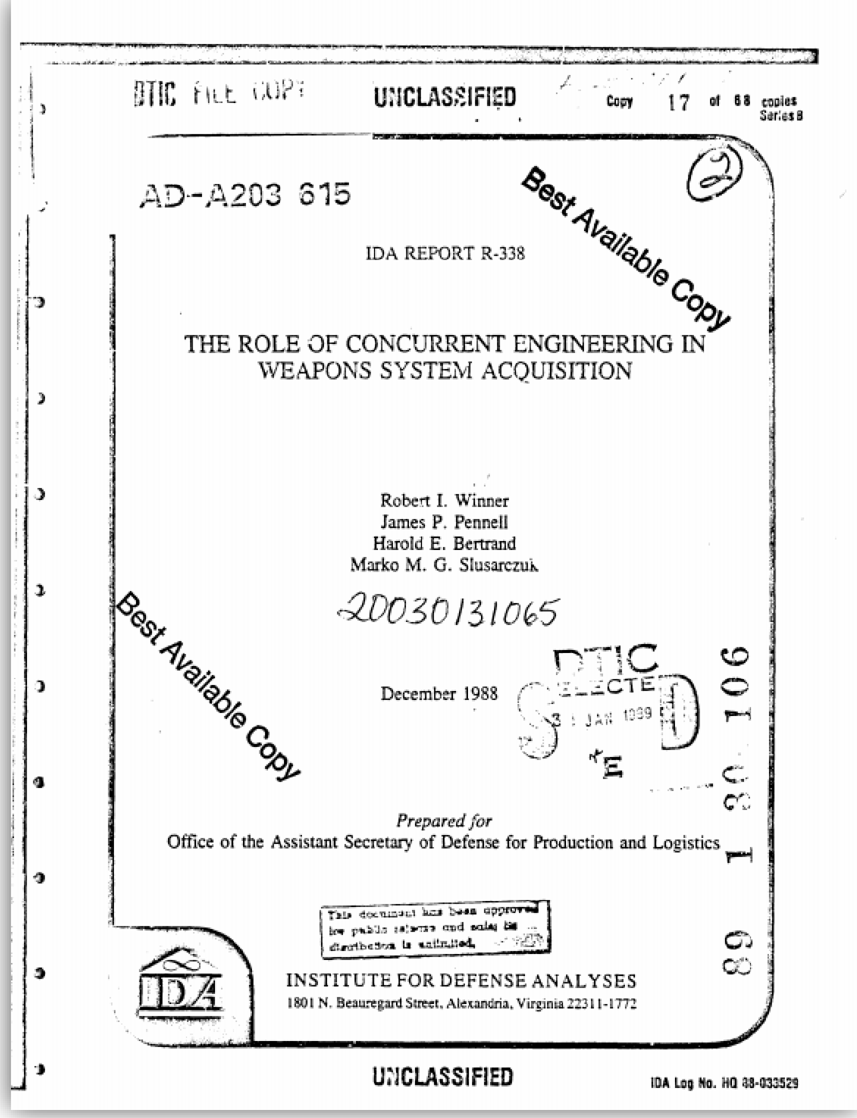
+Z Solar Panel
Bolts
Side Panel +Z
-Y Solar Panel
Bolts
Side Panel -Y
Bolts
RAM Mounted 5bit Digital Sun Sensor
Side Panel +X
XX W Ni-Cd Battery Pack
Silver Tape
Battery Temperature Sensor
Bolts and Thermal Paste
EPS Main Board
Bolts
3 Axis Magnetometer
Bolts
+X Solar Panel
Bolts
3 Axis Magnetorquers
Bolts
Side Panel -X
UFH Antenna
Bolts
AOCS Main Board
OBDH Main Board
Payload Main Controller
Bolts
Bolts
Bolts
Side Panel +Y
Bolts
+Y Solar Panel
Tetrahedral Reaction Wheels
Bolts
Telescope Temperature Sensor
Glue
Multi-Spectral Telescope
Bolts
S-Band Radio Board
UHF Radio Board
S-Band Antenna
PL 28
Bolts
Bolts
Side Panel -Z
Bolts
-Z Solar Panel


Space Environment
Sun
Magnetic Field Counter Forces
Magnetic Field
Photons
Photons
Forces
Radiation
ITA-AESP ACS
ITA-ELE EPS
Start Button
ITA-MEC STS
Points of Contact
Rocket Launcher
USB Cable
VBat Cable
VBat Cable
USB Cable
USB Cable
VBat Cable
Sat Bus
USB Bus
VBat Bus
USB Cable
VBat Cable
USB Cable
VBat Cable
USB Cable
USB Cable
VBat Cable
ITA-COMP OBDH
ITA-ELE TT&C
ITA-AESP Payload
UHF Link
S-Band Link
Fire
Fire Fighter
GS
Forest Fires
Routed Communication Link


3 Axis Magnetorquers
3 Axis Magnetometer
RAM Mounted 5bit Digital Sun Sensor
Tetrahedral Reaction Wheels
Magnetorquer Cables
Magnetometer Cables
SUn Sensor Cable
RW Cable
RW Connector Out
AOCS Main Board
PWM GPIO Lines
I2C Lines
I2C Lines
4 SPI Lines
3V3 Lines
5V Line
3V3s
ACS Arduino Controller
ACS OBS
Power Regulator
Power Line
USB Lines
USB Port Connector
OBDH Main Board
USB Lines
OBDH SD Memory Card
SD
OBDH RaspberryPi Zero Computer
OBDH OBS
Heating Mechanisms
Temperature Sensor
Temp Sensor Cables
Heating Power Distribution Connector
AD GPIO Lines Connector
Mechanism Main Board
Power Distribution
GPIO Lines
AD GPIO Lines
I2C Lines
STS Arduino Controller
STS OBS
Antenna Deployer Mechanism
VBat Lines
VBat Port Connector
USB Port Connector
Main Structure
Rocket Coupling
Thermal and Radiation Shield
Radiation and Thermal Blocking
XX W Ni-Cd Battery Pack
XX W Solar Panel
Battery Connector
Solar Panel Power Cable
Battery Jack
Solar Panel Connector
EPS Main Board
Solar Panel Power Connector
Power Conditioning and Power Distribution
Solar Panel Current Sensor
Start Button
Solar Panel I2C Sensor Lines
VBat Lines
5V Lines
GPIO Lines
EPS Arduino Controller
Vbat Bus I2C Sensor Lines
EPS OBS
VBat Bus Current Sensor
5V Bus Current Sensor
5V Lines
5V Bus I2C Sensor Lines
VBat Out Port
5V Out Port
USB Data Port
VBus Connector
USB Port Connector
S-Band Radio Board
S-Band SoC Radio Controller
S-Band SD Memory Card
SD Port
LLVDS Bitstream Lines
LLVDS Bitstream Lines
Power Amplifier and modulator
Demodulator
Modulated Signal Lines
Modulated Signal Lines
Antenna Cable
S-Band Antenna
UHF Radio Board
UHF FPGA Radio Controller
USB Port Connector
LLVDS Bitstream Lines
Power Amplifier and Modulator
VBat Port Connector
Antenna Cable
UFH Antenna

Concurrent Engineering

Things started to speed-up


DTIC FILE COPY UNCLASSIFIED Copy 17 of 68 copies Series B

AD-A203 615

IDA REPORT R-338

THE ROLE OF CONCURRENT ENGINEERING IN WEAPONS SYSTEM ACQUISITION

Robert I. Winner
James P. Pennell
Harold E. Bertrand
Marko M. G. Slusarczuk

20030131065

December 1988

DTIC ELECTE 31 JAN 1989 E

89 1 30 106

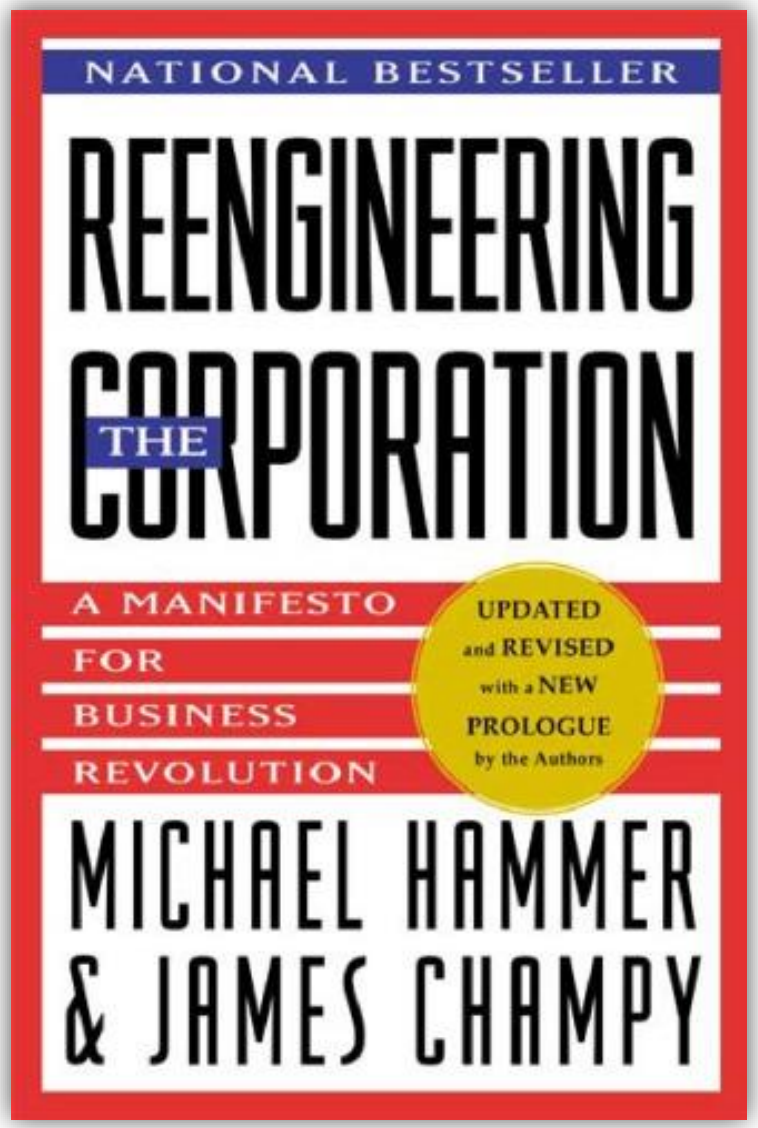
Best Available Copy

Prepared for
Office of the Assistant Secretary of Defense for Production and Logistics

This document has been approved for public release and sale; its distribution is unlimited.

IDA
INSTITUTE FOR DEFENSE ANALYSES
1801 N. Beauregard Street, Alexandria, Virginia 22311-1772

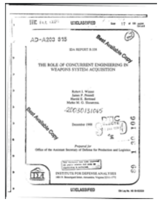
UNCLASSIFIED IDA Log No. HQ 88-033529



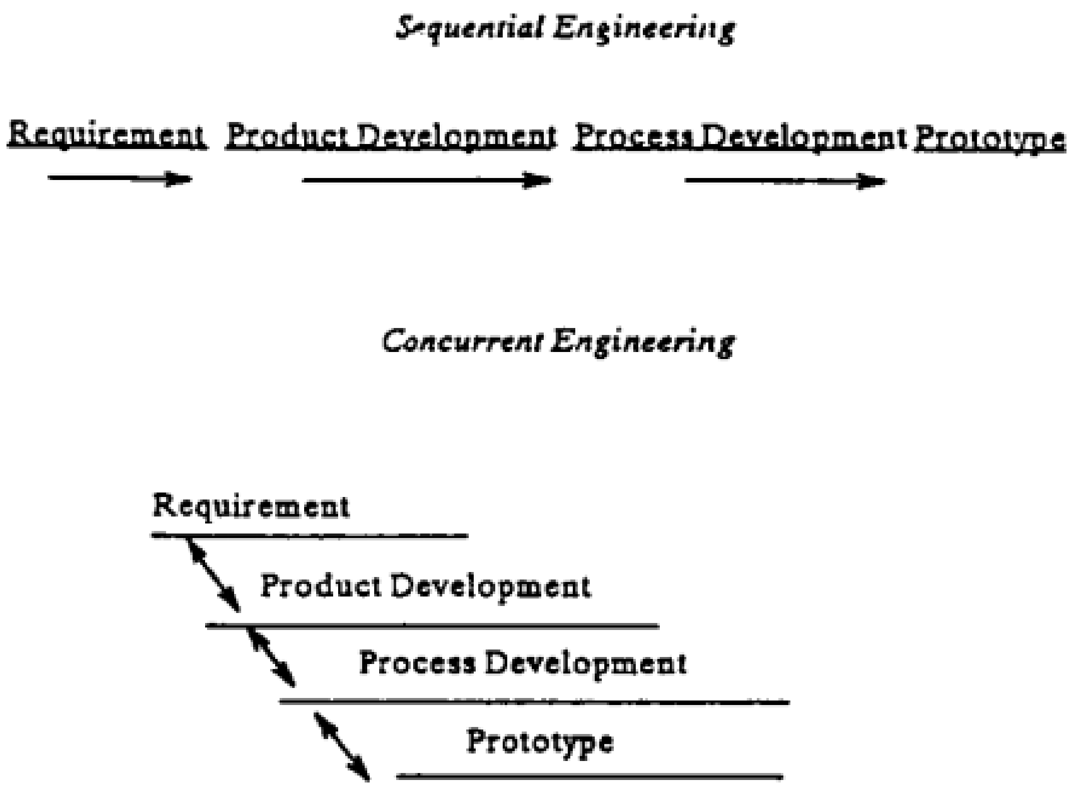
Participants at the first IDA concurrent engineering workshop discussed concurrent engineering practices in several U.S. companies. They described the use of methods that included traditional system engineering practices and new engineering and management approaches. DoD and Air Force initiatives to improve the acquisition process were also presented. Based on the discussion at that workshop, on further contributions from participants, and consultation with various reviewers, the following definition was developed:

> Concurrent engineering is a systematic approach to the integrated, concurrent design of products and their related processes, including manufacture and support. This approach is intended to cause the developers, from the outset, to consider all elements of the product life cycle from conception through disposal, including quality, cost, schedule, and user requirements.

Winner, Robert I., Pennell, James P., Bertrand, Harold E., and Slusarczuk, Marko M. G. (1991). "The Role of Concurrent Engineering in Weapons System Acquisition", *Institute for Defense Analyses Report R-338*, December 1988, p v.



Parallelization/INTEGRATION of work



[Concurrent engineering: an overview for Autotestcon | IEEE Conference Publication | IEEE Xplore]()

[The Role of Reduced Latency in Integrated Concurrent Engineering | Center for Integrated Facility Engineering (stanford.edu)]()

THIS IS A WORKING PAPER. Please contact the authors for permission to cite or circulate.

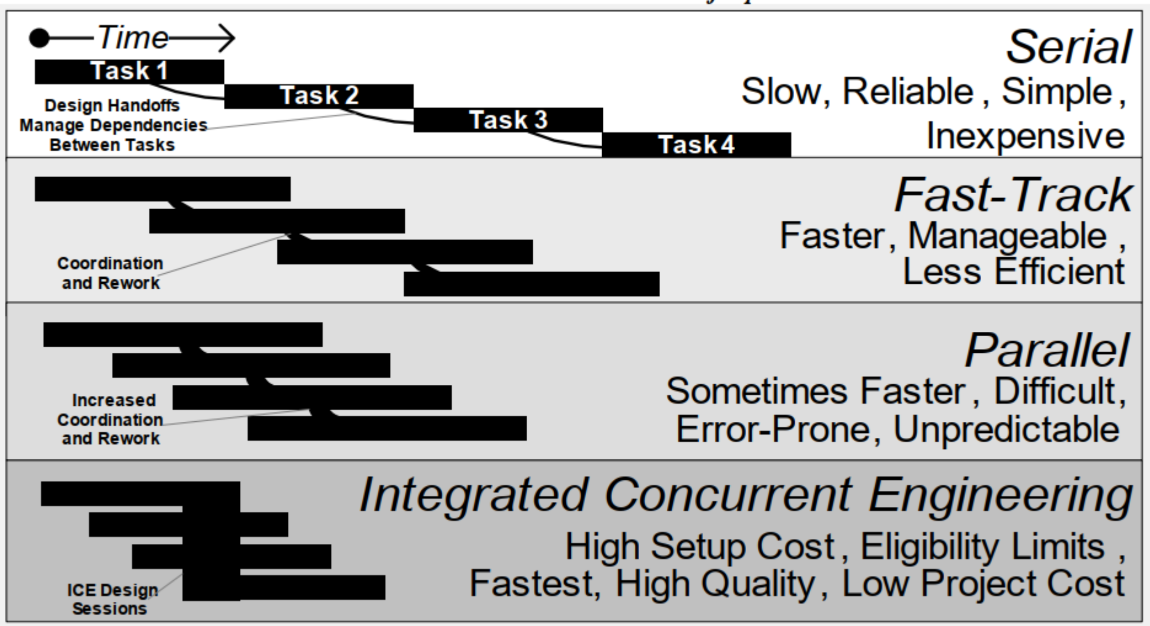
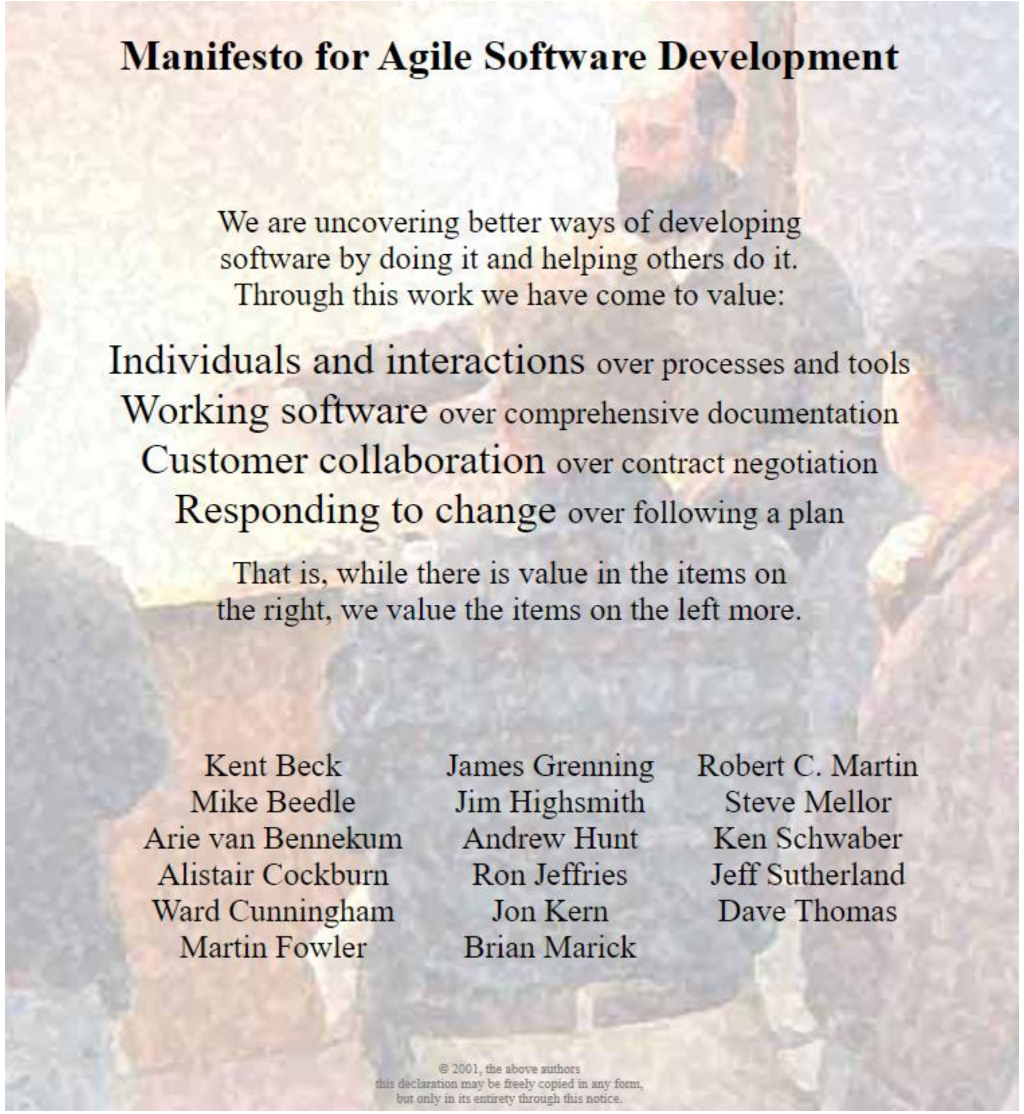


Figure 3: Degrees of Parallelism ICE radically increases task parallelism and operates to facilitate effective and efficient coordination with little waiting and rework. This diagram shows a schematic Gantt bar chart with four tasks arranged with increasing parallelism. Projects under increasing pressure to meet tight schedules often overlap tasks that once executed serially. Compressing schedule in this way is costly, difficult, and risky for teams failing to anticipate the complex interactions between product, organization, process, and technology. Many industries are broadly parallelizing design tasks, but few teams have experimented with ICE yet. ICE represents the most accelerated of these engineering methods, in which the full organization gathers and executes the most interdependent work together. We predict and observe in ICE that the coordination and rework effort equals or exceeds the effort given to direct work. ICE works well when the individual subtask durations are short (a few minutes) and coordination latency is reliably exceptionally short (waits to initiate discussion rarely exceed one minute), and coordination duration is short (a few minutes at most).

2001 – Agile manifesto


Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

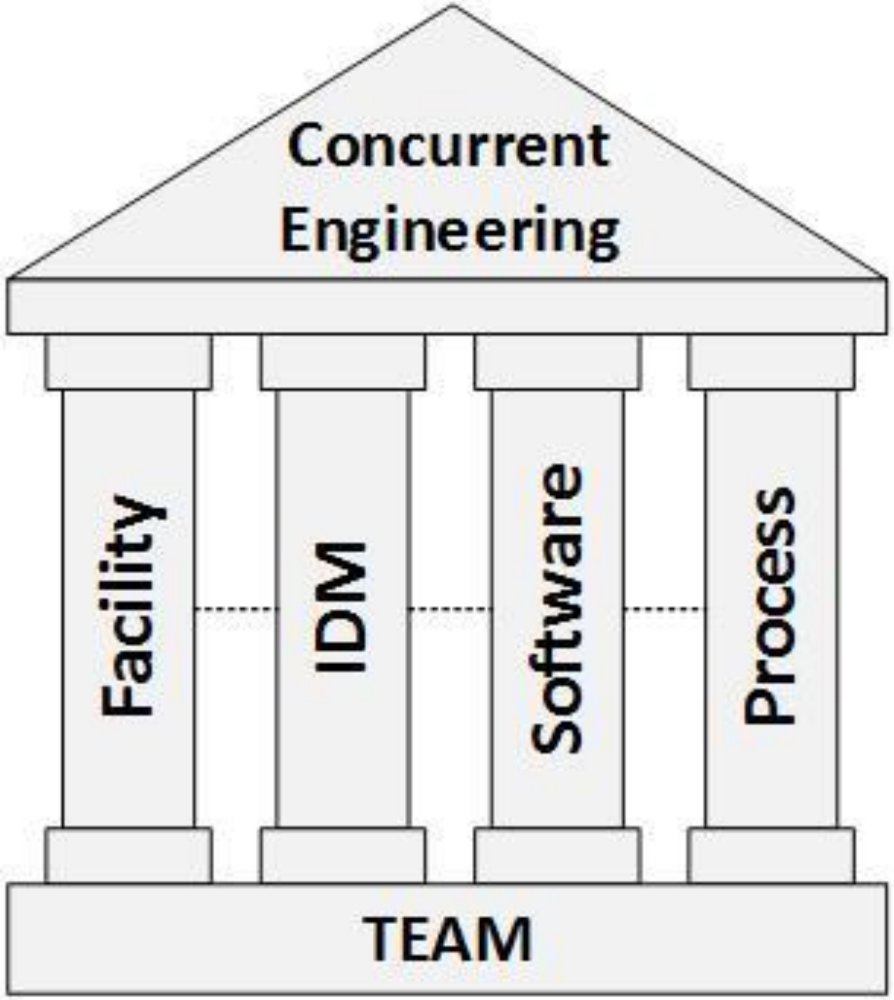
That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas

- We follow these principles:
  - Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
  - Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
  - Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
  - Businesspeople and developers must work together daily throughout the project.
  - Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.
  - The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
  - Working software is the primary measure of progress.
  - Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
  - Continuous attention to technical excellence and good design enhances agility.
  - Simplicity--the art of maximizing the amount of work not done--is essential.
  - The best architectures, requirements, and designs emerge from self-organizing teams.
  - At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

[Manifesto for Agile Software Development (agilemanifesto.org)](https://agilemanifesto.org)

Modern concurrent engineering



"Modern Concurrent Engineering"

Bidirectional Graphical Modelling Supporting Concurrent Spacecraft Design | Semantic Scholar

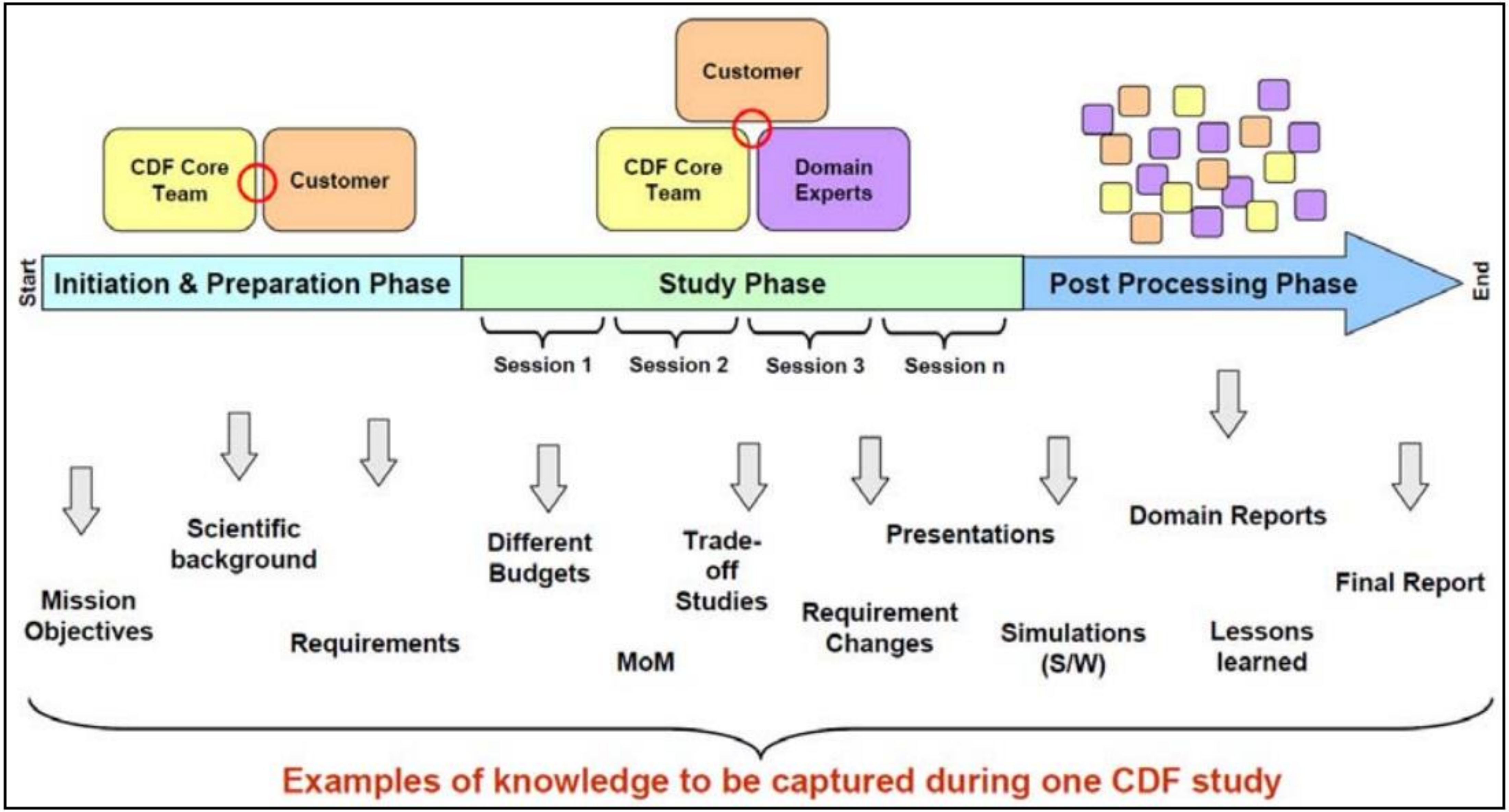
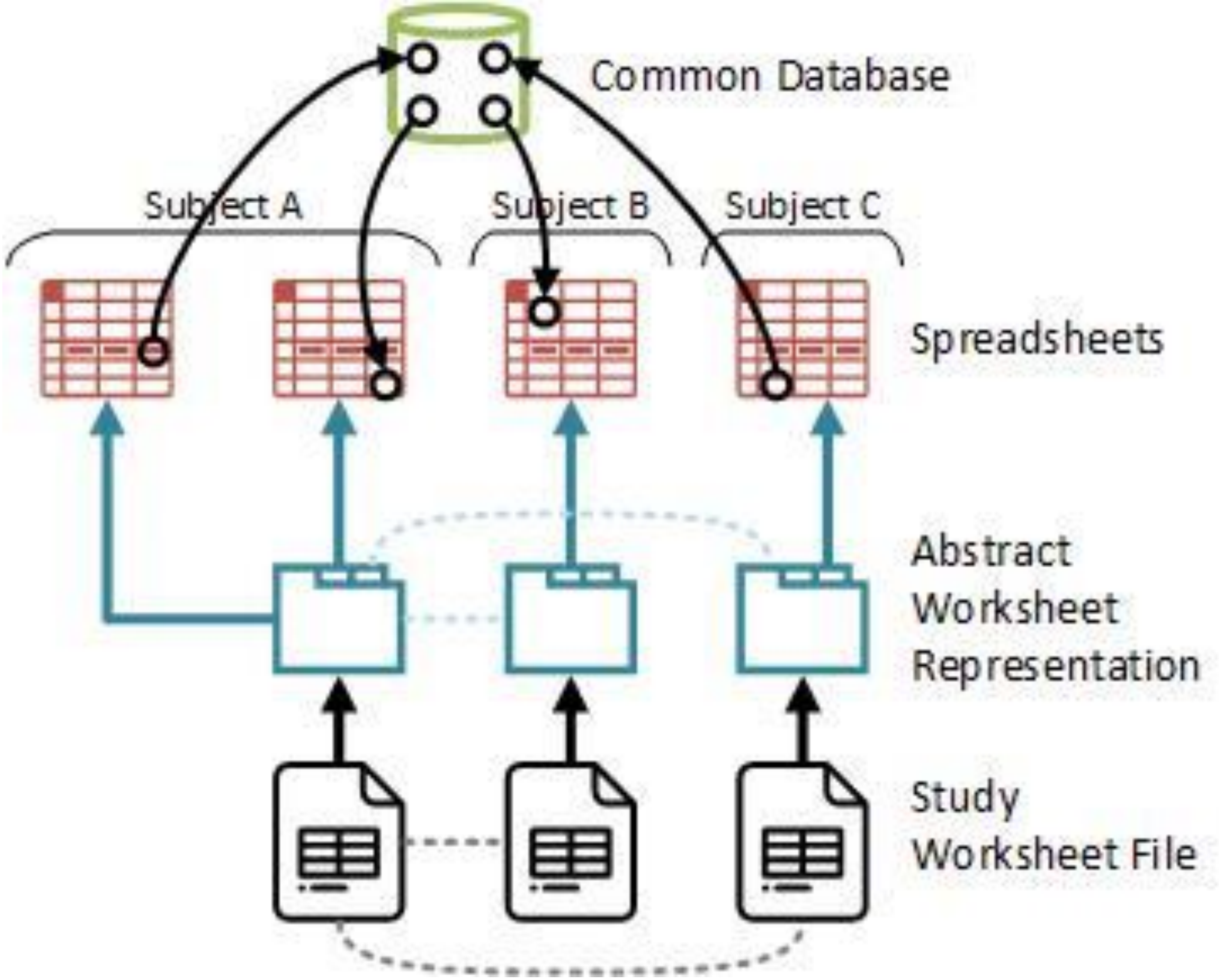


Figure 2: Concurrent Engineering Process with respect to Knowledge Management

[(PDF) Concurrent Engineering Knowledge Management Architecture (researchgate.net)]

[ESA - What is the CDF?]

Multiple spreadsheet through a shared bd




ECSS-E-TM-E-10-25A
20 October 2010

EUROPEAN COOPERATION
ECSS
FOR SPACE STANDARDIZATION

Space engineering

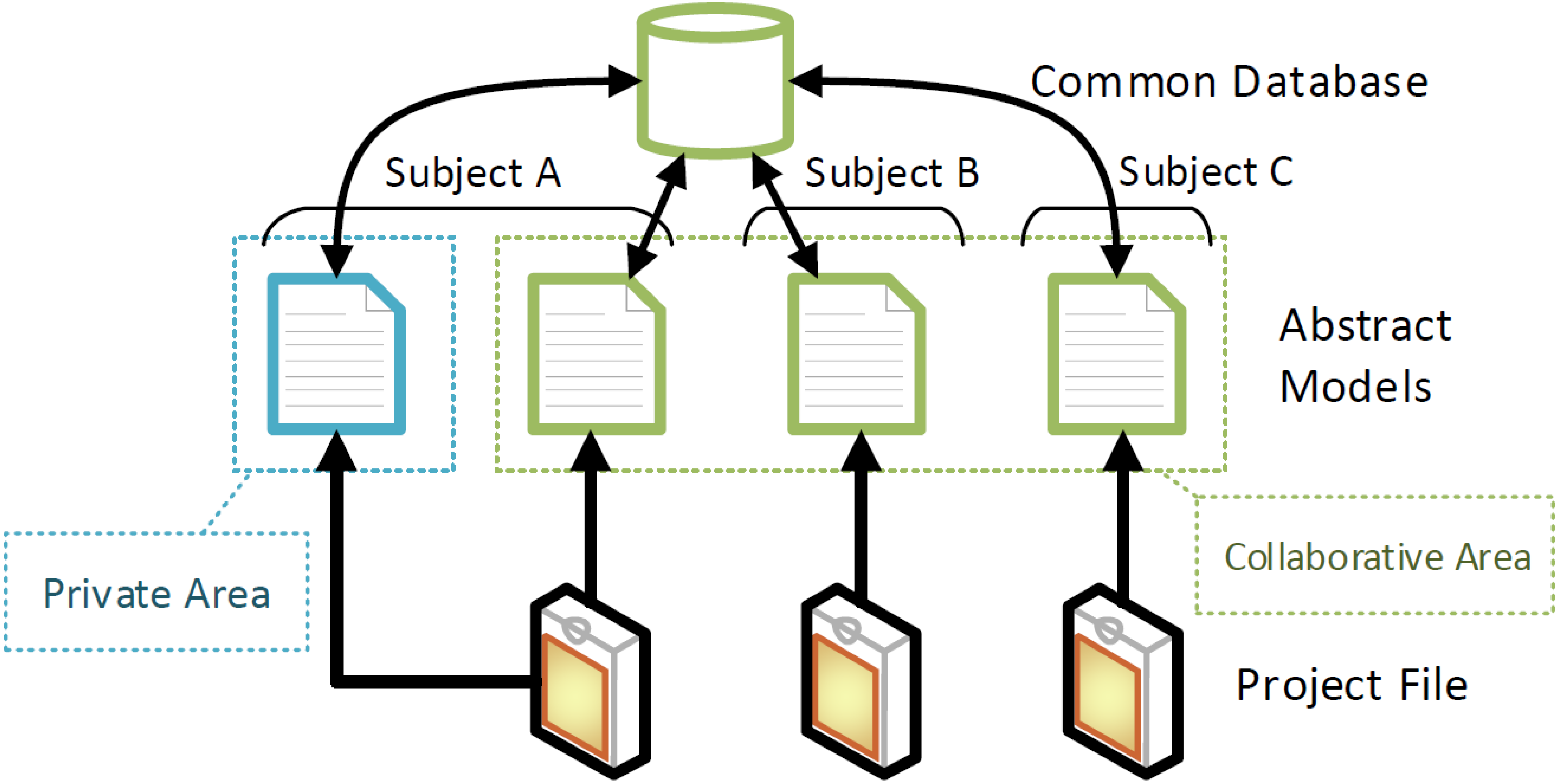
Engineering design model data exchange (CDF)

ECSS Secretariat
ESA-ESTEC
Requirements & Standards Division
Noordwijk, The Netherlands

(joke) motto


Paige Bailey @DynamicWebPaige · 9 de jul
1) Pick an industry
2) Ask people in that industry what they use spreadsheets for
3) Build something better
Traduzir do inglês
54
825
2,2 mil

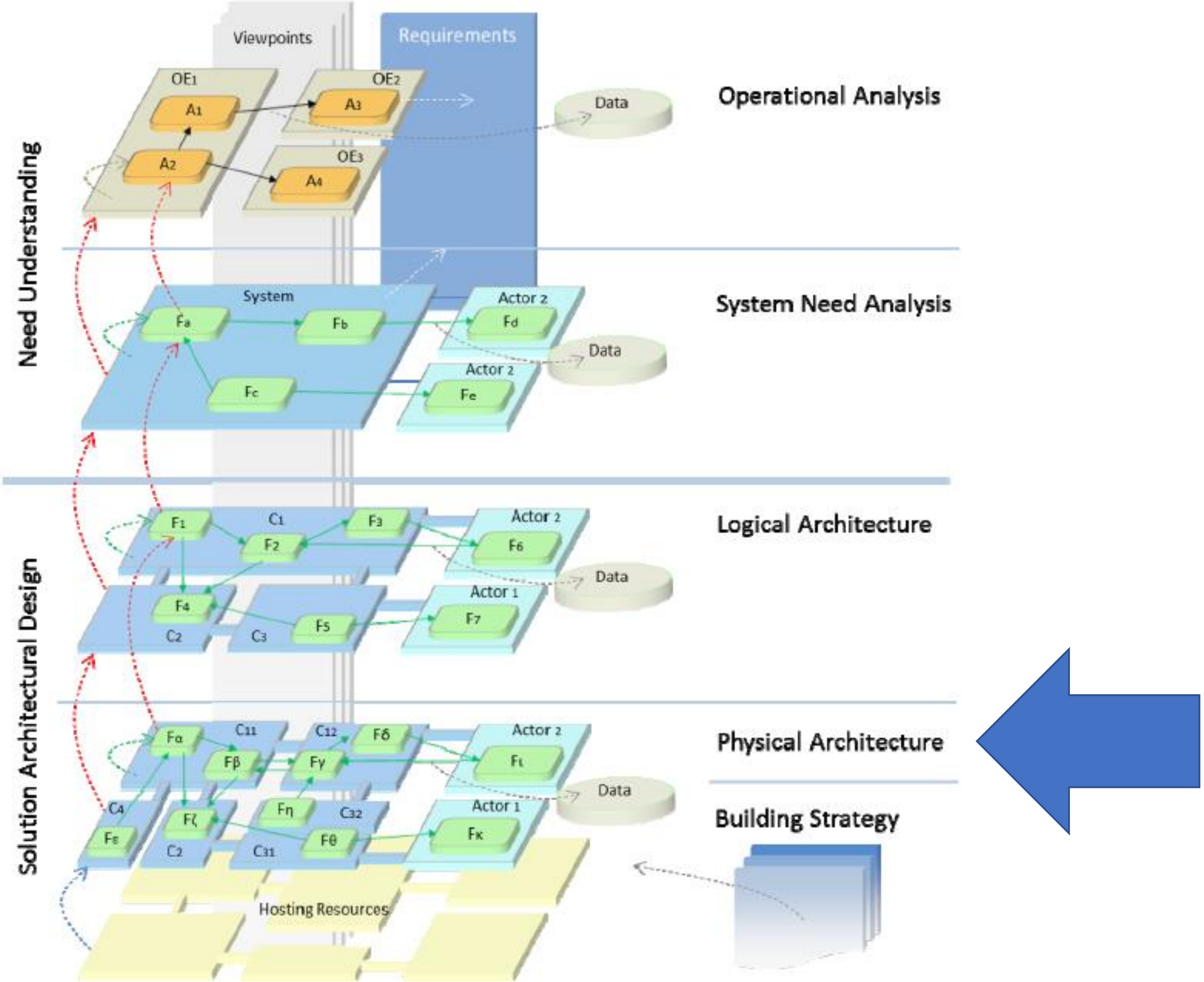
From doc to model



Challenges – co-engineering

- Collaborative work (SystemToSubsystem / T4C)
- Data-base of solutions (reuse) (Libraries)
- Strong Interface Definition/Management
- Optimization (man-made / automatic)
- Coupled analysis – Simulation (P4C and other connectors)
- Model has **SEVERAL THOUSANDS** of advantages, however the spreadsheet metaphor is easy and known.

PHYSICAL ARCHITECTURE


Viewpoints
Requirements
Operational Analysis
System Need Analysis
Logical Architecture
Physical Architecture
Building Strategy
Need Understanding
Solution Architectural Design
Data
Hosting Resources
System
Actor 2
Actor 1

WHAT IS IN THE PHYSICAL ARCHITECTURE (PA)?

Physical Architecture

"how the system will be built"

- This perspective has the same objective as the logical architecture, except that it **defines the finalized architecture of the system**, as it should be completed and integrated. It **adds the functions required by the implementation and technical choices and reveals the behavioral components that perform these functions**. These behavioral components are then implemented using host implementation components that offer them the necessary material resource.
- Defines the solution at a ***sufficient level of detail to specify the developments and acquisitions of all subsystems (or components) to be implemented, and to define and orientate the system integration, verification and validation (IVV) phases***.

- It is often at this level only that **choices and constraints are introduced** related to implementation and production technologies, **to existing elements to be re-used**.
- Any ambiguities or inaccuracies that could still exist in the logical architecture (LA), if they did not impact its structuring, should this time be resolved, in order to constitute clear development contracts for the identified components.
- PA is the **privileged place of co-engineering** with subsystem engineering and software or hardware components.

The main activities to be undertaken for the definition of the finalized PA

- to define the structuring principles of the architecture and behavior;
- to detail and finalize the expected system behavior;
- to build and rationalize one or more possible system architectures;
- to select, complete and justify the system architecture retained.

Definition of the structuring principles of the architecture and behavior

- The major objective of the PA is to **minimize complexity through rationalizing**.
  - One of the most used means of rationalization consists of **reducing diversity and heterogeneity** within the solution, by searching for similarities and therefore possible architecture invariants (sometimes called "**patterns**") that can be applied more than once in the same manner – or configurable.
  - Another classic way to overcome complexity is based on the **separation of concerns and their containment** within parts of the architecture as separate as possible from each other.

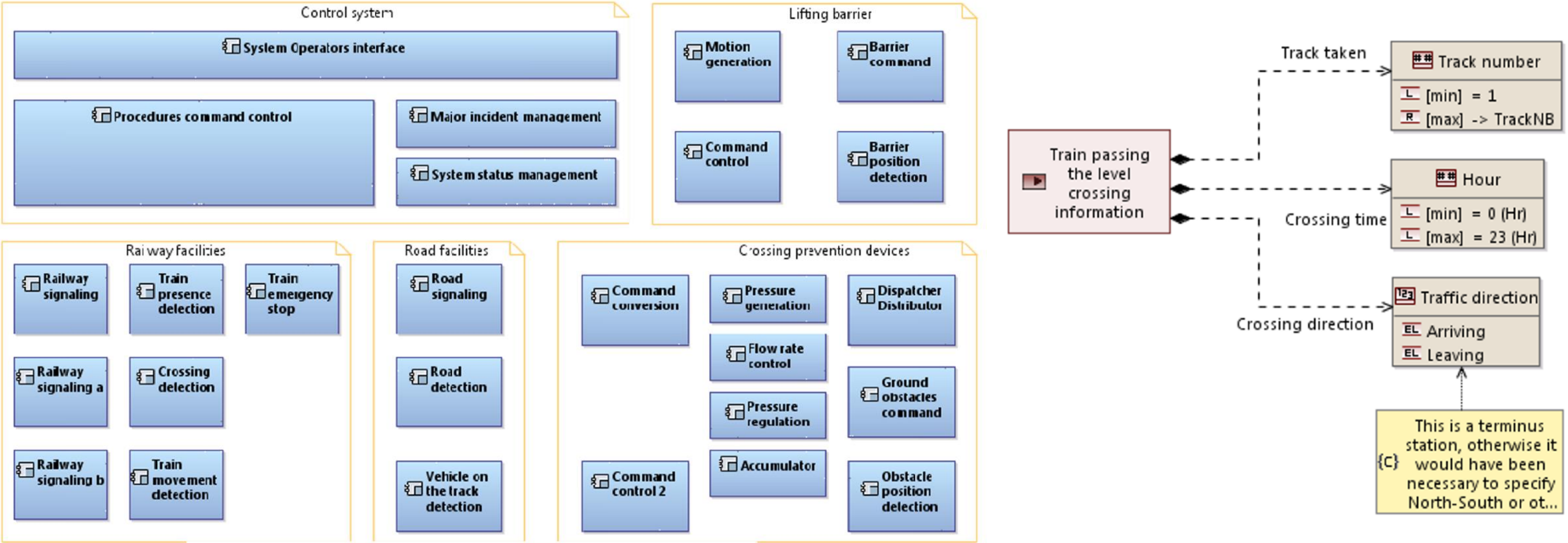
Detail and finalization of the expected system behavior

- Define the expected behavior of the system, to a **level of detail and validation** enough so that each of its components can be implemented (or selected and purchased), without any further risk or major questioning; this definition must of course demonstrate compliance with constraints, especially nonfunctional constraints, by which the system will have to abide when being used under operational conditions.

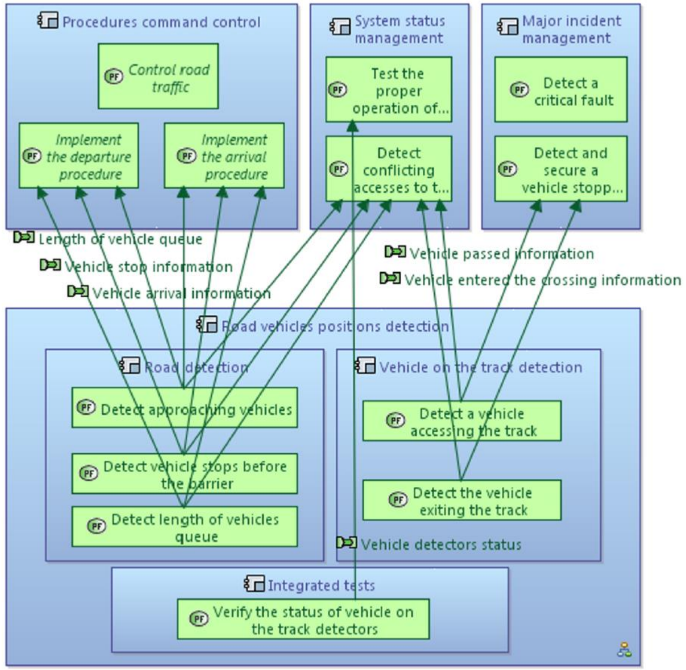
- In particular, the finalized behavior should **not necessarily be considered as a simple refinement** of that defined in the LA. The finalization of the chosen behavior in fact often constitutes a **re-designing**, which must result from the comparison between the principle behavior of the LA, and the implications of the principles chosen in the PA: **technological choices and adoption of standards, previous structuring principles, etc.**

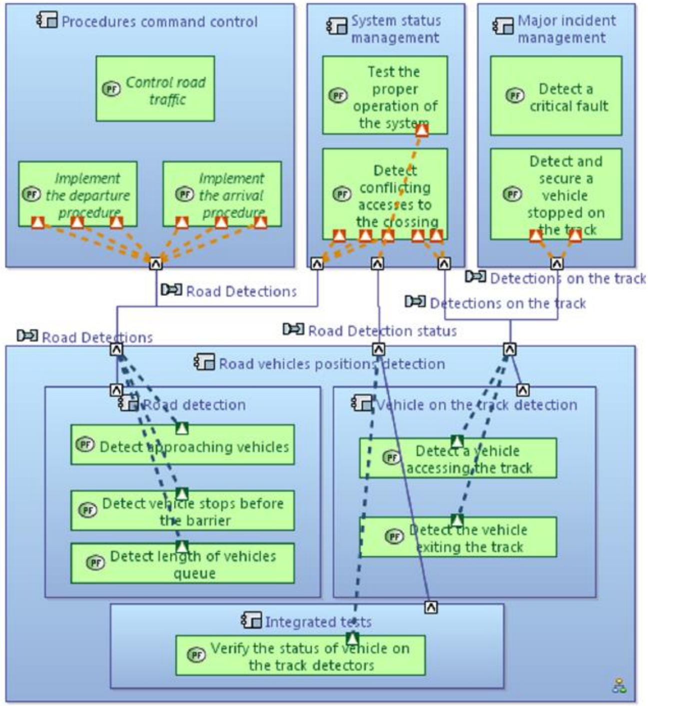
Construction and rationalization of one or more possible system architectures

- This step is intended to **define one or more solutions reflecting the structuring principles** defined in the LA, the previous finalized behavior, satisfying the expected non-functional constraints and applying technology and reuse choices decided in accordance with the structuring principles adopted.

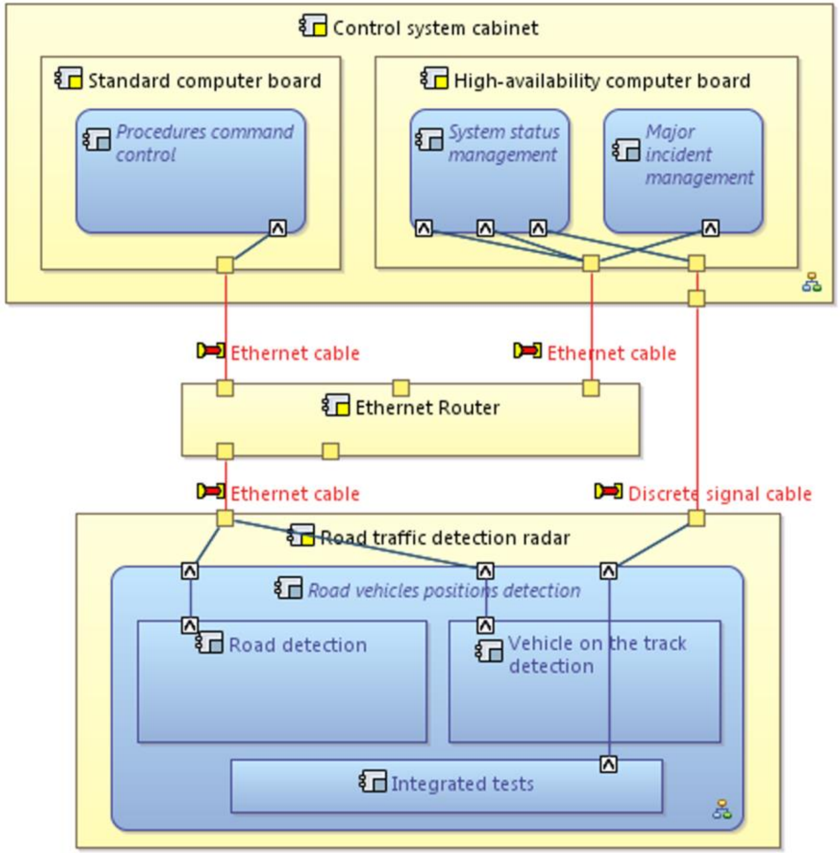


- In the simplest cases, or in systems with a physical or electrical dominant, the exchange items are often simple in their description and usage at this engineering and modeling level. However, for more complex exchange items, involving large numbers of exchange contents elements, it is desirable to be able to **structure a list of exchange items that can be extensive, by grouping them by type of service achieved**, for example. This is the role of the concept of an interface (also mainly present in software design).





- The PA complements this behavioral description by way of the definition of implementation components, or **hosting physical components, containing behavioral components and forming the infrastructure of the system; the behavioral components are deployed on these host components**, which provide necessary resources for their behavior and hardware vectors (links) for their communications. It may thus consist of high-performance computers, resources for digital or analog processing, mechanical systems, evaporators, furnaces, chemical reactors, etc.
- **Hosting physical components are themselves connected by physical links**, reflecting the media that channel exchanges between behavioral components (a cabled network, a satellite link, a pipe or a mechanical shaft, for example).
- The same rationalization processes have to be performed for hosting physical components as for the behavior and behavioral components, in compliance with the established structuring principles.

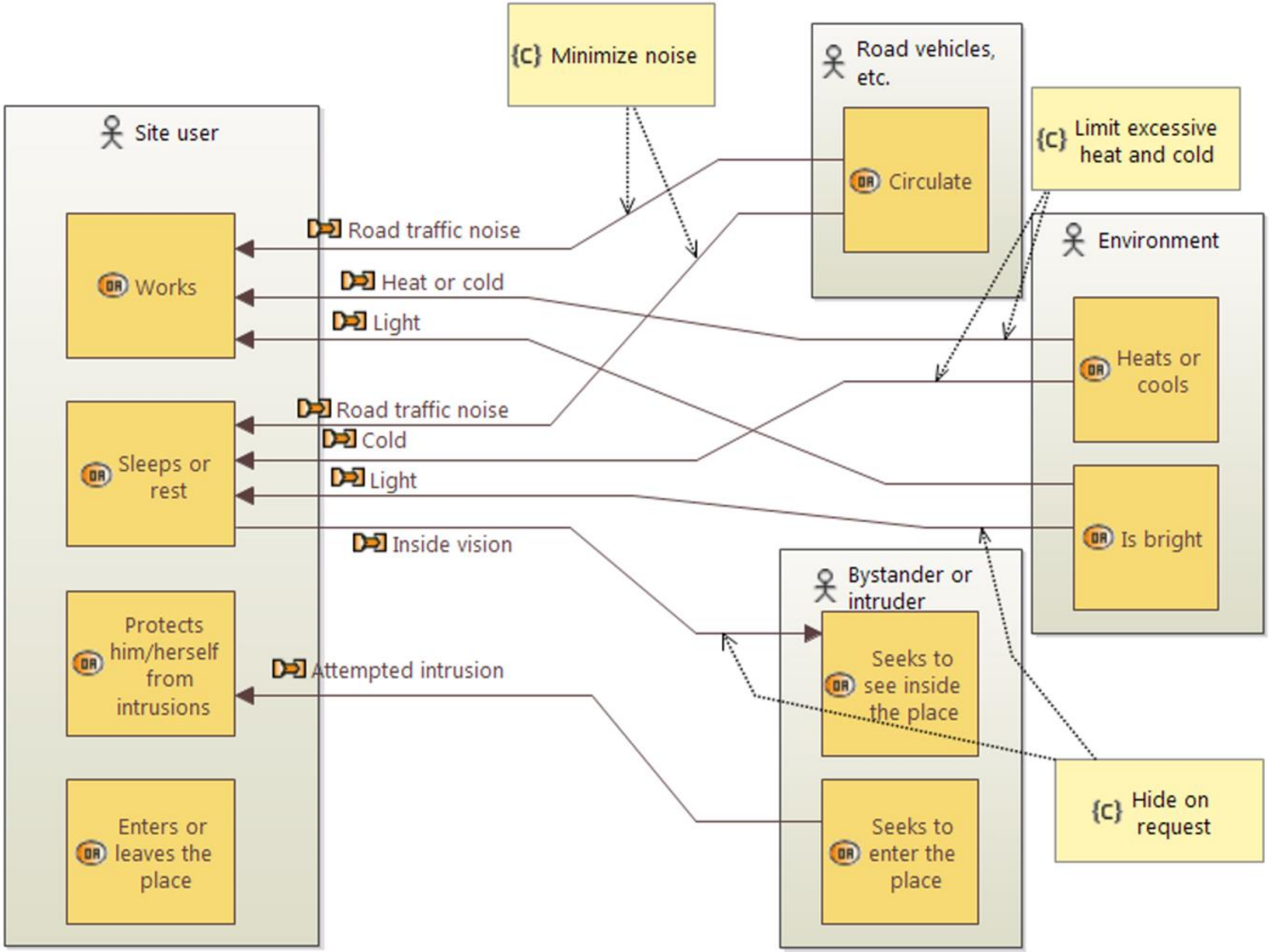


Selection, completion and justification of the system architecture

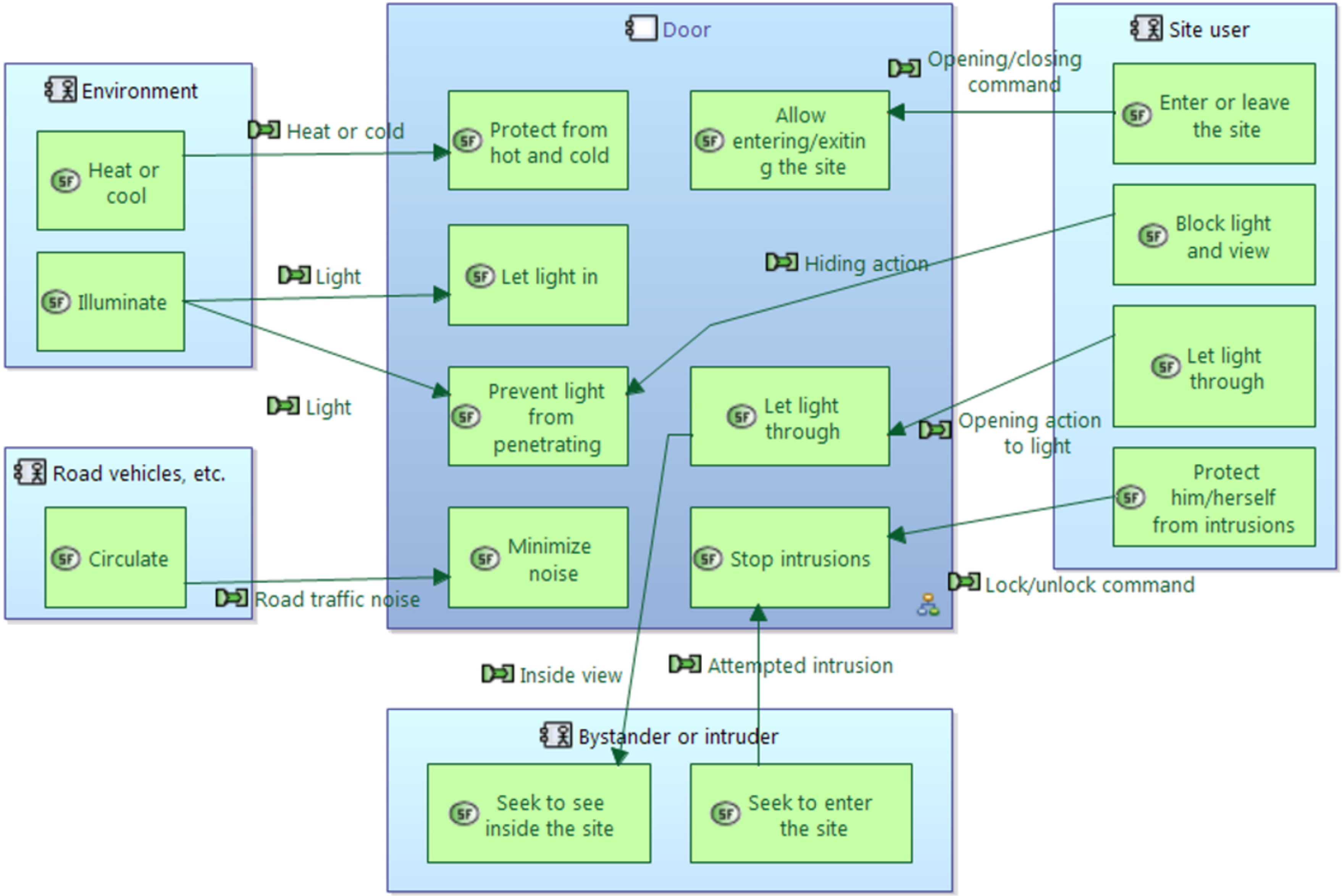
- Finalize the choices among potential alternatives, and **verify that the retained alternative satisfies, possibly by means of an acceptable trade-off, all of the needs and constraints that have been imposed thereon**.
- For example, the implementation resources available may not be sufficient to support an expected behavior or associated properties (computational load too high for a given process in computers supporting it, temperature and pressure too high for a given pipe, etc.). This will lead to a redesigning of the architecture, including a redecomposition and a different distribution of behavioral components, or the use of other implementation resources (more powerful computers, more robust pipes).

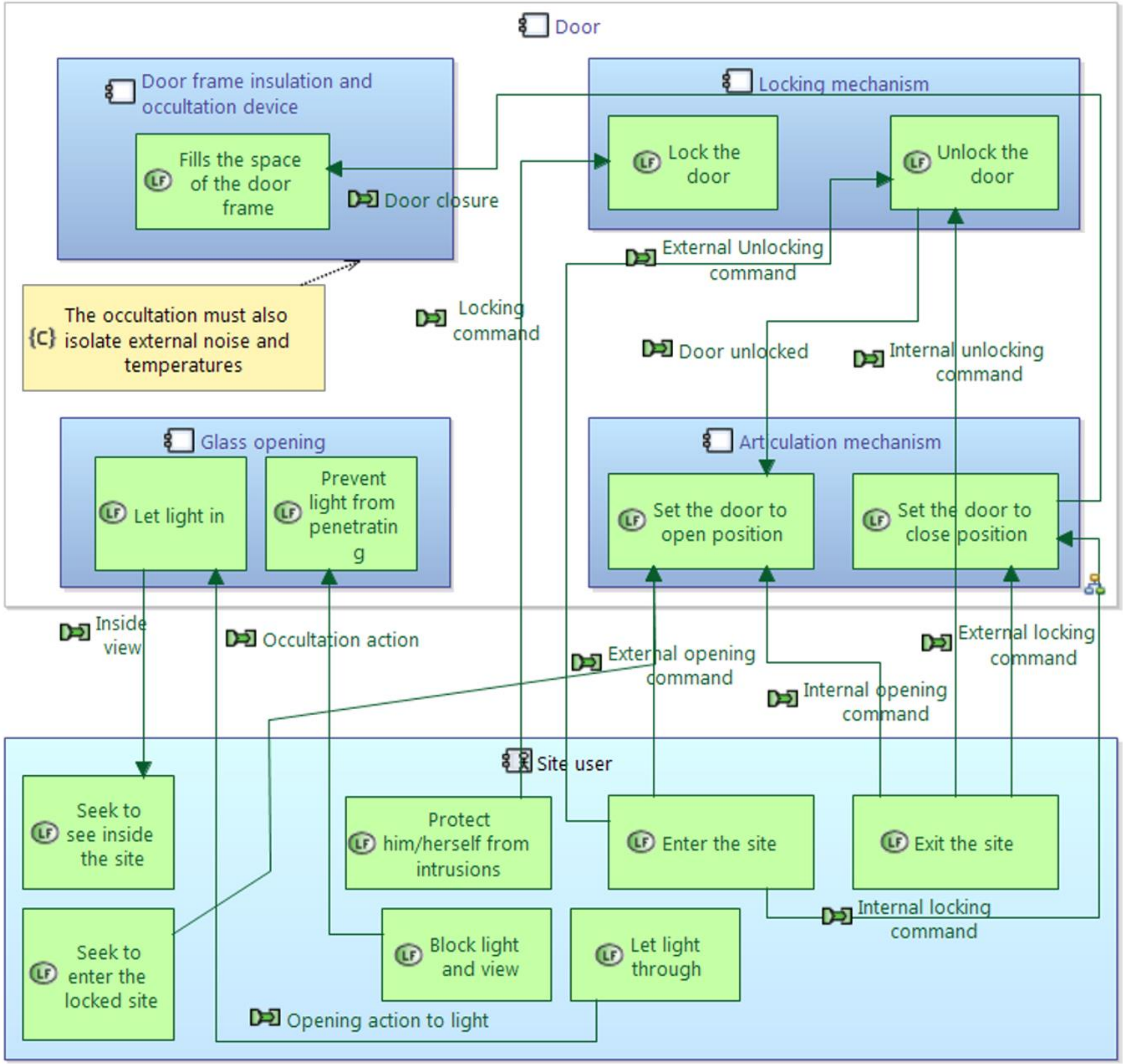
SUMMARY

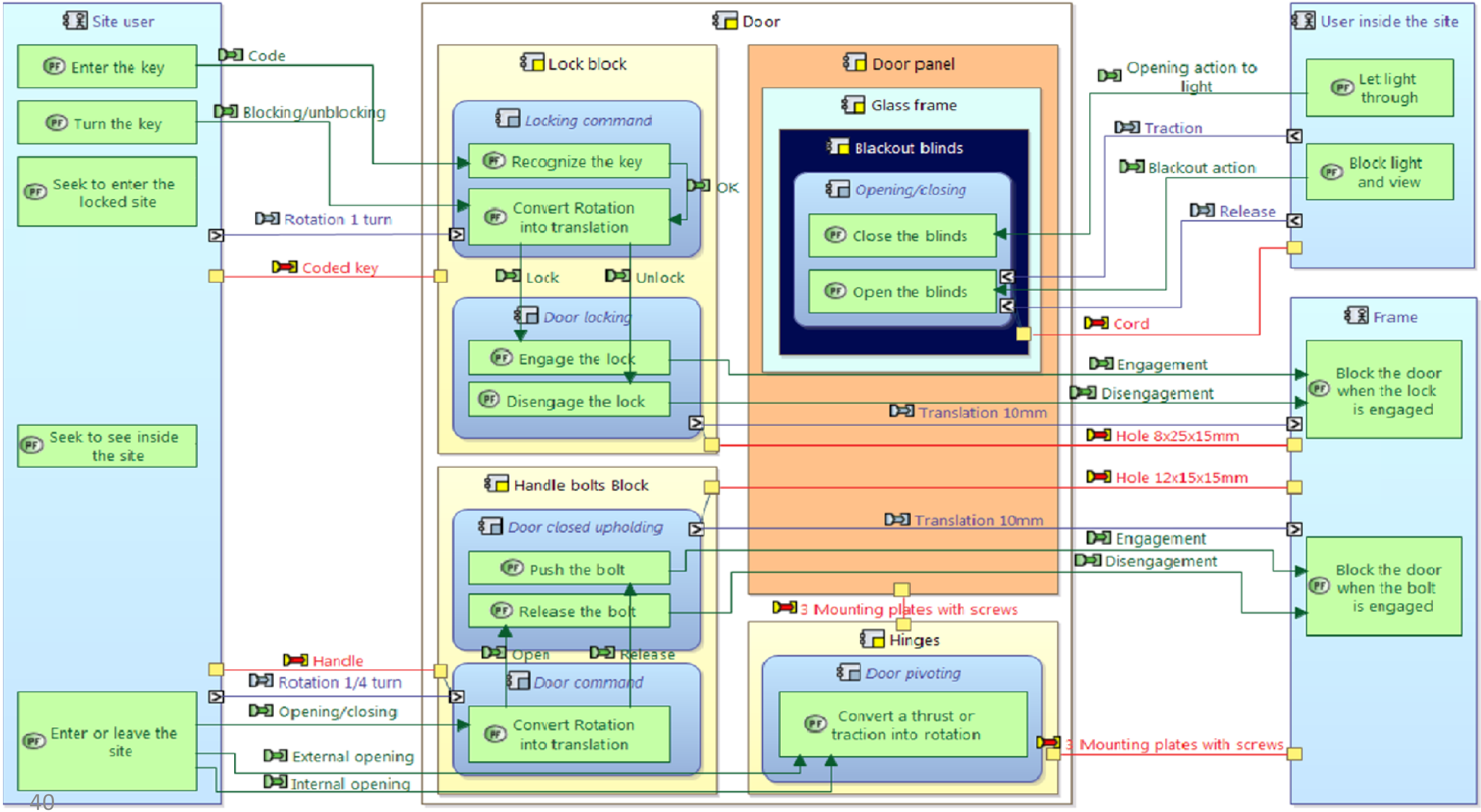
define the structuring principles of the architecture and behavior;	Factors impacting or constraining the definition of the architecture, as well as the viewpoints and structuring design choices mentioned earlier, equally apply to this level of architecture and are to be taken into account in a similar way.
detail and finalize the expected system behavior;	define the expected behavior of the system, to a level of detail and validation enough so that each of its components can be implemented (or selected and purchased), without any further risk or major questioning
build and rationalize one or more possible system architectures	define one or more solutions reflecting the structuring principles defined in the LA, the previous finalized behavior, satisfying the expected non-functional constraints and applying technology and reuse choices decided in accordance with the structuring principles adopted.
select, complete and justify the system architecture retained.	finalize the choices among potential alternatives, and verify that the retained alternative satisfies, possibly by means of an acceptable trade-off, all of the needs and constraints that have been imposed thereon.



{C} Minimize noise
Road vehicles, etc.
Circulate
{C} Limit excessive heat and cold
Site user
Works
Road traffic noise
Heat or cold
Light
Environment
Heats or cools
Sleeps or rest
Road traffic noise
Cold
Light
Is bright
Inside vision
Bystander or intruder
Seeks to see inside the place
Protects him/herself from intrusions
Attempted intrusion
Enters or leaves the place
Seeks to enter the place
{C} Hide on request


Door
Environment
Heat or cool
Illuminate
Road vehicles, etc.
Circulate
Site user
Enter or leave the site
Block light and view
Let light through
Protect him/herself from intrusions
Protect from hot and cold
Allow entering/exiting the site
Let light in
Prevent light from penetrating
Let light through
Minimize noise
Stop intrusions
Bystander or intruder
Seek to see inside the site
Seek to enter the site
Heat or cold
Light
Light
Road traffic noise
Opening/closing command
Hiding action
Opening action to light
Lock/unlock command
Inside view
Attempted intrusion


Door
Door frame insulation and occultation device
Fills the space of the door frame
Door closure
Locking mechanism
Lock the door
Unlock the door
External Unlocking command
The occultation must also isolate external noise and temperatures
Locking command
Door unlocked
Internal unlocking command
Glass opening
Let light in
Prevent light from penetratin g
Articulation mechanism
Set the door to open position
Set the door to close position
Inside view
Occultation action
External opening command
Internal opening command
External locking command
Site user
Seek to see inside the site
Protect him/herself from intrusions
Enter the site
Exit the site
Internal locking command
Seek to enter the locked site
Block light and view
Let light through
Opening action to light


Site user
Enter the key
Turn the key
Seek to enter the locked site
Seek to see inside the site
Enter or leave the site
Code
Blocking/unblocking
Rotation 1 turn
Coded key
Handle
Rotation 1/4 turn
Opening/closing
External opening
Internal opening
Door
Lock block
Locking command
Recognize the key
Convert Rotation into translation
OK
Lock
Unlock
Door locking
Engage the lock
Disengage the lock
Handle bolts Block
Door closed upholding
Push the bolt
Release the bolt
Open
Release
Door command
Convert Rotation into translation
Door panel
Glass frame
Blackout blinds
Opening/closing
Close the blinds
Open the blinds
3 Mounting plates with screws
Hinges
Door pivoting
Convert a thrust or traction into rotation
User inside the site
Opening action to light
Let light through
Traction
Blackout action
Block light and view
Release
Cord
Frame
Engagement
Disengagement
Translation 10mm
Hole 8x25x15mm
Hole 12x15x15mm
Block the door when the lock is engaged
Translation 10mm
Engagement
Disengagement
Block the door when the bolt is engaged
3 Mounting plates with screws

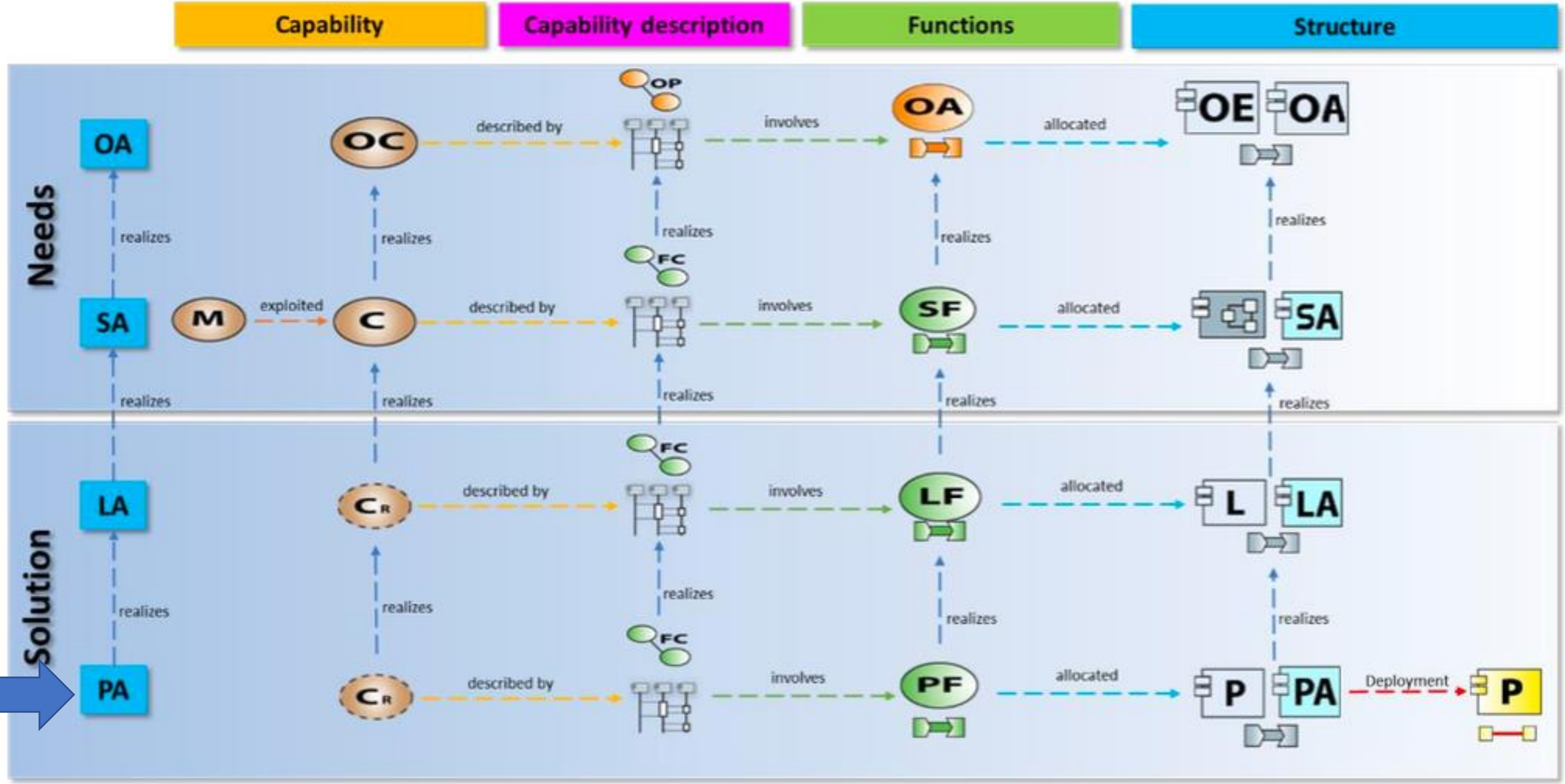


Figure 2.3: Arcadia ontology traceability

https://www.slideshare.net/HelderCastro3/mbse-with-arcadia-methodpdf-256664096

Arcadia layer	Requirements	Capability	Capability description	Functional	Structure	Modes and States	Data	Interfaces
Operational Analysis	**R-OA** Capture stakeholder requirements	**OA1** Define Operational Capabilities	**OA2** Define processes and scenarios	**OA3** Define Operational Activities and interactions	**OA4** Capture Operational Entities and Actors. Allocate Operational Activities to Operational Actors, Entities	**M&S-OA5** Define operational modes and states	**D-OA6** Define operational data model	**I-OA7** Define interfaces and describe interfaces scenarios
System Analysis	**R-SA** Derive Stakeholder requirements and capture System requirements	**SA1** Define System Missions and System Capabilities	**SA2** Define Functional Chains and Scenarios.	**SA3** Define System Functions. Define Functional Exchanges and components	**SA4** Allocate System Functions to System and Actors	**M&S-SA5** Define system modes and states	**D-SA6** Define system data model	**I-SA7** Define interfaces and describe interfaces scenarios Enrich Logical Scenarios.
Logical Architecture	**R-LA** Derive system requirements and Capture components requirements	**LA1** Transition Capabilities Realization from system layer	**LA2** Define Functional Chains and scenarios	**LA3** Derive System Functions and define Logical Functions. Define Functional Exchanges and components.	**LA4** Allocate Logical Functions to Logical Components	**M&S-LA5** Define logical components modes and states	**D-LA6** Define logical data model	**I-LA7** Delegate System Interfaces and create Logical Interfaces. Enrich Logical Scenarios.
Physical Architecture	**R-PA** Derive logical requirements and capture physical requirements	**PA1** Transition Capabilities Realization from logical layer	**PA2** Define Functional Chains, Scenarios, and Physical Path	**PA3** Derive Logical Functions and define Physical Functions. Define Functional Exchanges and components.	**PA4** Define Physical Nodes and refine Behavioural Physical Components. Allocate Behavioural Components.	**M&S-PA5** Define physical nodes modes and states	**D-PA6** Define physical data model	**I-PA7** Delegate Logical Interfaces and create Physical Interface. Enrich Physical Scenarios.

Table 3.2: Arcadia matrix activities

https://www.slideshare.net/HelderCastro3/mbse-with-arcadia-methodpdf-256664096

Arcadia layer	Requirements	Capability	Capability description	Functional	Structural	Modes and States	Data	Interfaces
Operational Analysis	**R-OA** No dedicated diagram	**OA1** [OCB] Operational Capabilities	**OA2** [OAS] Operational Activity Scenario [OPD] Operational Process Scenario [OES] Operational Entity Scenario	**OA3** [OABD] Operational Activity Breakdown Diagram [OAIB] Operational Activity Interaction Blank	**OA4** [OEBD] Operational Entities Blank Diagram [ORB] Operational Roles Blank [OAB] Operational Architecture Blank	**M&S-OA5** [MSM] Modes and States	**D-OA6** [CDB] Class Diagram	**I-OA7** [IDB] Interface Definition Blank [CEI] Component External Interfaces [IS] Interface Scenario [CDI] Component Detailed Interface
System Analysis	**R-SA** No dedicated diagram	**SA1** [MCB] Mission and Capabilities Blank [CC] Contextual Capability	**SA2** [FS] System Functional Scenario [ES] System Entity Scenario [SFCD] System Functional Chain Description	**SA3** [SFBD] System Functional Breakdown Diagram [SDFB] System Data Flow Blank	**SA4** [CSA] Contextual System Actor [SAB] System Architecture Blank	**M&S-SA5** [MSM] Modes and States	**D-SA6** [CDB] Class Diagram	**I-SA7** [IDB] Interface Definition Blank [CEI] Component External Interfaces [IS] Interface Scenario [CDI] Component Detailed Interface
Logical Architecture	**R-LA** No dedicated diagram	**LA1** [CRB] Capabilities Realization Blank [CRI] Contextual Capability Realization Involvement	**LA2** [FS] Logical Functional Scenario [ES] Logical Entity Scenario [LFCD] Logical Functional Chain Description	**LA3** [LFBD] Logical Functional Breakdown Diagram [LDFB] Logical Data Flow Blank	**LA4** [LCBD] Logical Component Breakdown Diagram [LAB] Logical Architecture Blank	**M&S-LA5** [MSM] Modes and States	**D-LA6** [CDB] Class Diagram	**I-LA7** [IDB] Interface Definition Blank [CEI] Component External Interfaces [IS] Interface Scenario [CDI] Component Detailed Interface
Physical Architecture	**R-PA** No dedicated diagram	**PA1** [CRB] Capabilities Realization Blank [CRI] Contextual Capability Realization Involvement	**PA2** [FS] Physical Functional Scenario [ES] Physical Entity Scenario [PFCD] Physical Functional Chain Description	**PA3** [PFBD] Physical Functional Breakdown Diagram [PDFB] Physical Data Flow Blank	**PA4** [PCBD] Physical Component Breakdown Diagram [PAB] Physical Architecture Blank	**M&S-PA5** [MSM] Modes and States	**D-PA6** [CDB] Class Diagram	**I-PA7** [IDB] Interface Definition Blank [CEI] Component External Interfaces [IS] Interface Scenario [CDI] Component Detailed Interface

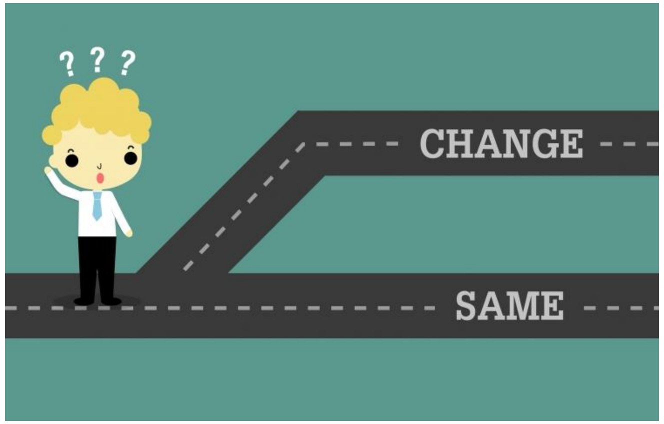
Table 3.3: Arcadia diagrams matrix

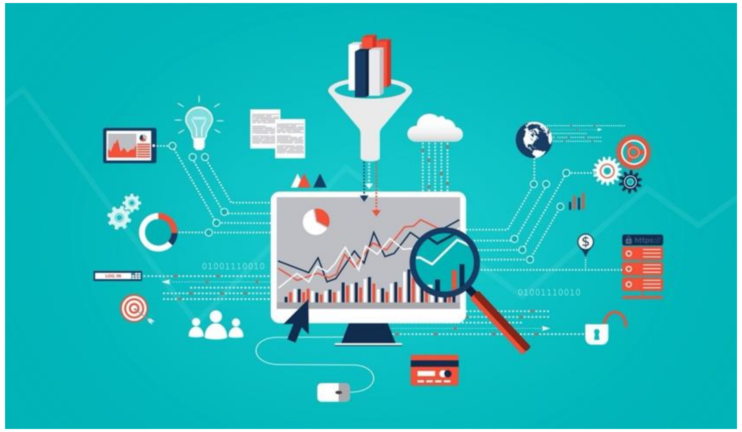
https://www.slideshare.net/HelderCastro3/mbse-with-arcadia-methodpdf-256664096

PHYSICAL ARCHITECTURE CONCEPTS

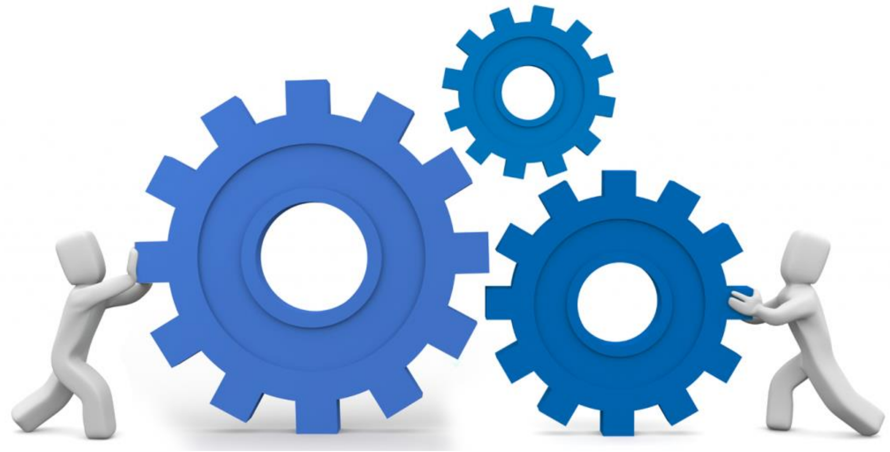
- At this level, the main concepts proposed by Arcadia are similar to those of the Logical Architecture: Physical Function, Functional Exchange, Physical Component, Physical Actor, etc. However, there are some additional concepts, notably:

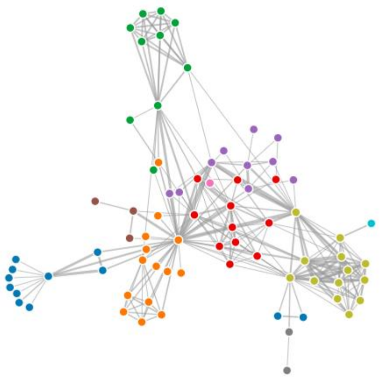
- **Behavior Physical Component:** **Physical Component tasked with Physical Functions** and therefore carrying out part of the **behavior** of the System (for example software component, data server, etc.);



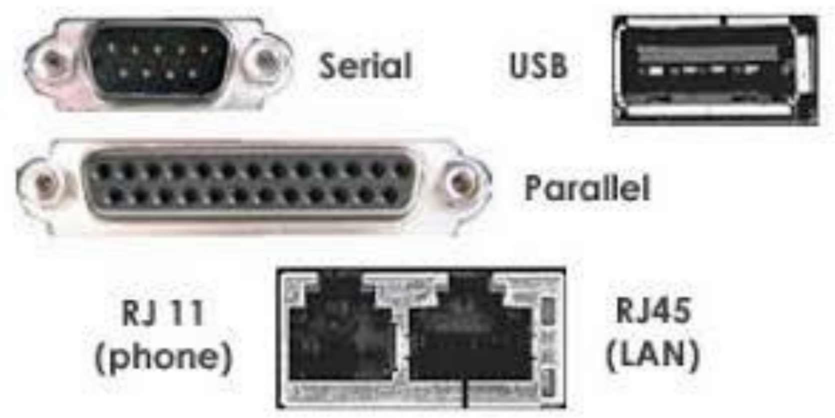


- **Node (or Implementation) Physical Component**: Physical Component that provides the **material resources needed for one or several Behavior Components** (for example processor, router, OS, etc.).

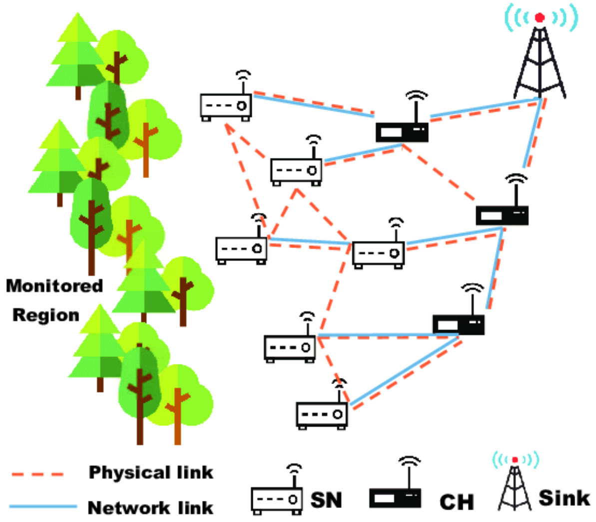




- **Physical Port**: non-oriented **port that belongs to an Implementation Component** (or Node). The structural port (Component Port), on the other hand, has to belong to a Behavior Component;

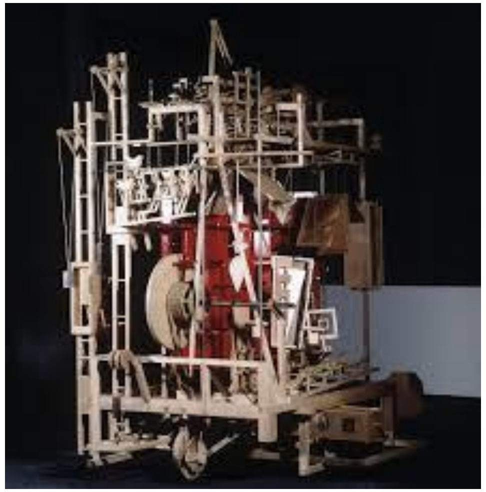
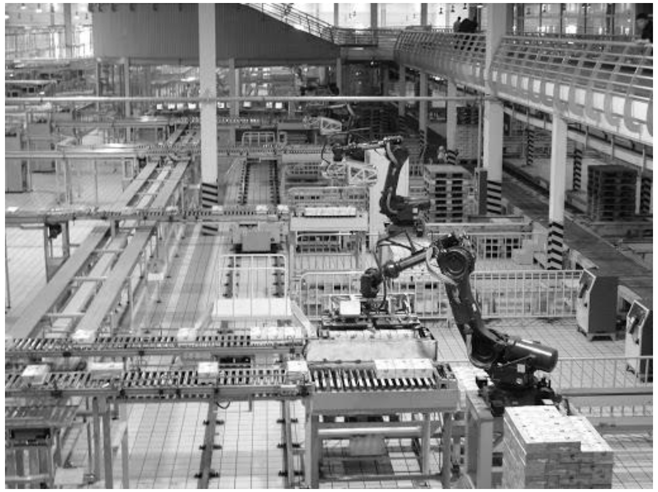


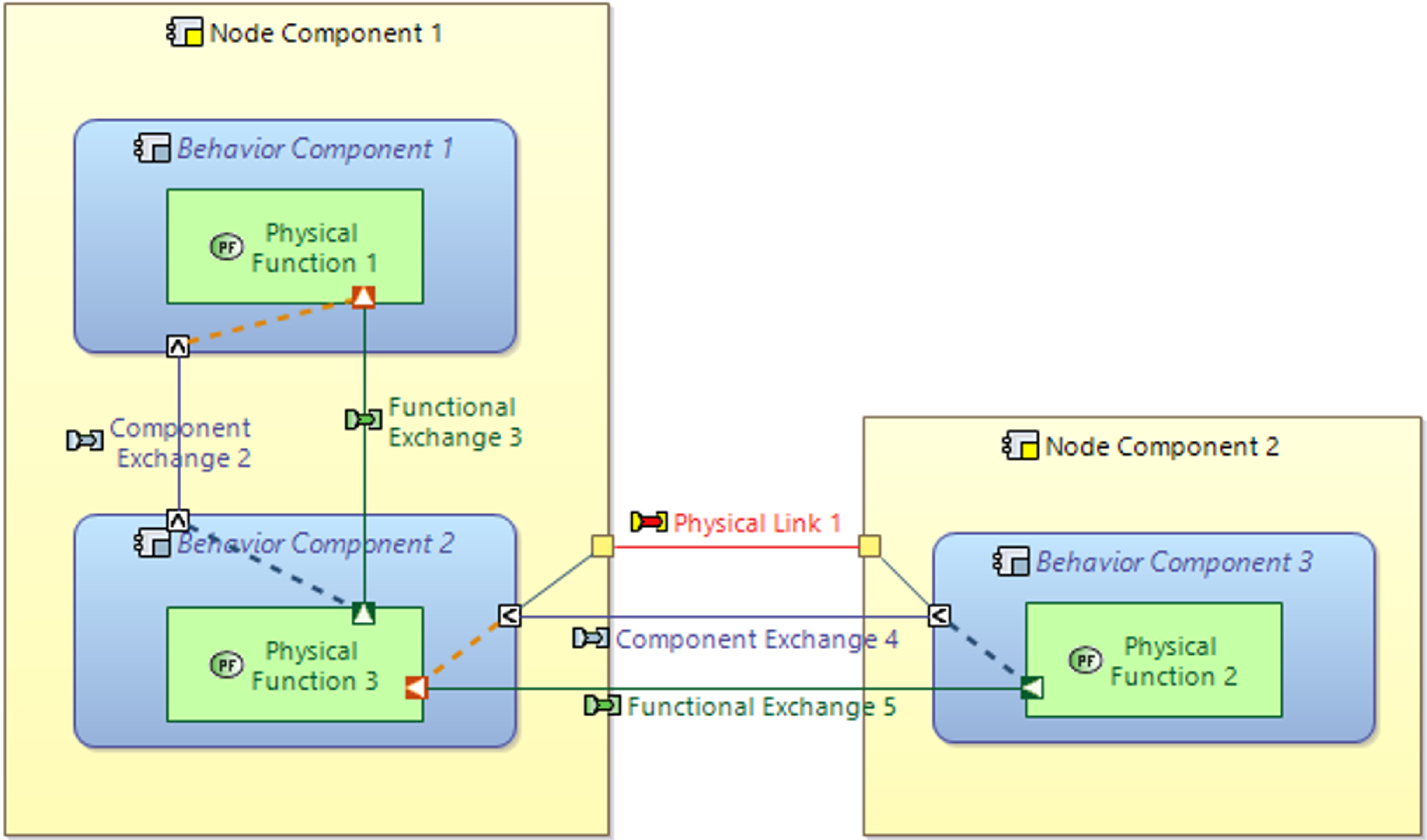
- **Physical Link**: non-oriented **material connection between Implementation Components** (or Nodes). The Component Exchange remains a connection between Behavior Components. A Physical Link allows one or several Component Exchanges to take place (for example Ethernet cable, USB cable, etc.);



- **Physical Path**: **organized succession of Physical Links enabling a Component Exchange** to go through several Implementation Components (or Nodes).




Node Component 1
Behavior Component 1
Physical Function 1
Component Exchange 2
Functional Exchange 3
Behavior Component 2
Physical Function 3
Physical Link 1
Component Exchange 4
Functional Exchange 5
Node Component 2
Behavior Component 3
Physical Function 2

PHYSICAL ARCHITECTURE DIAGRAMS

Transition from Logical Functions

- Perform an automated transition of Logical Functions
- Create Traceability Matrix

Initialization and automated update of the physical functions according to the logical functions

The transition tools create a first 1-1 traceability mapping between Physical Architecture and Logical Architecture. Use dedicated traceability matrices to modify the traceability relationships.

Refine Physical Functions, describe Functional Exchanges

- [PFBD] Create a new Functional Breakdown diagrams
- [PDFB] Create a new Functional Dataflow Blank diagram
- [FS] Create a new Functional Scenario

Enrich and details the functional breakdown with new physical functions.

Describe the data flows between physical functions and identify specific functional chains.

Define Physical Components and Actors, Manage deployments

- Perform an automated transition of External Logical Actors
- Perform an automated transition of Logical System
- [PCBD] Create a new Physical Component Breakdown diagram
- [PAB] Create a new Physical Architecture diagram
- Create a new Physical Component / Logical Component Matrix

The initialization and automated updated of the physical actors can be automatically performed according to logical actors.

Define the physical components. A physical component is a physical representation of an entity in the system (hardware, software, firmware, personnel, facilities, data, materials, services and processes). It is in charge of the implementation of one or several logical components. A physical component can be Node or Behavior.

Allocate Physical Functions to Physical Components

[PAB] Create a new Physical Architecture diagram

[ES] Create a new Exchange Scenario

Create a new allocation Physical Component / Physical Function Matrix

Delegate Logical Interfaces and create Physical Interfaces

[CII] Create a new Contextual Internal Interface diagram on the Physical System

Enrich Physical Scenarios

Perform an automated transition of Logical Architecture Capabilities

[IS] Create a new Interface Scenario

The behavioral physical components are responsible for implementing the physical functions. Manage these allocations using an architecture diagram and deduce component exchanges implementing the functional exchanges.

Manage the deployment of behavior components on node components and deduce physical links and paths. Create dataflows scenarios to illustrate functional exchanges between the components.

Delegate each logical interface to one physical component. Create new physical interfaces between components.

Specify the dynamical behavior of the physical components by completing the interaction sequences coming from the Logical Architecture. The enrichment of the interaction sequences and the identification of the new physical interfaces are two very tight and iterative activities.

The scenario refinement process is iterative, each update on a source can be automatically propagated to the target.

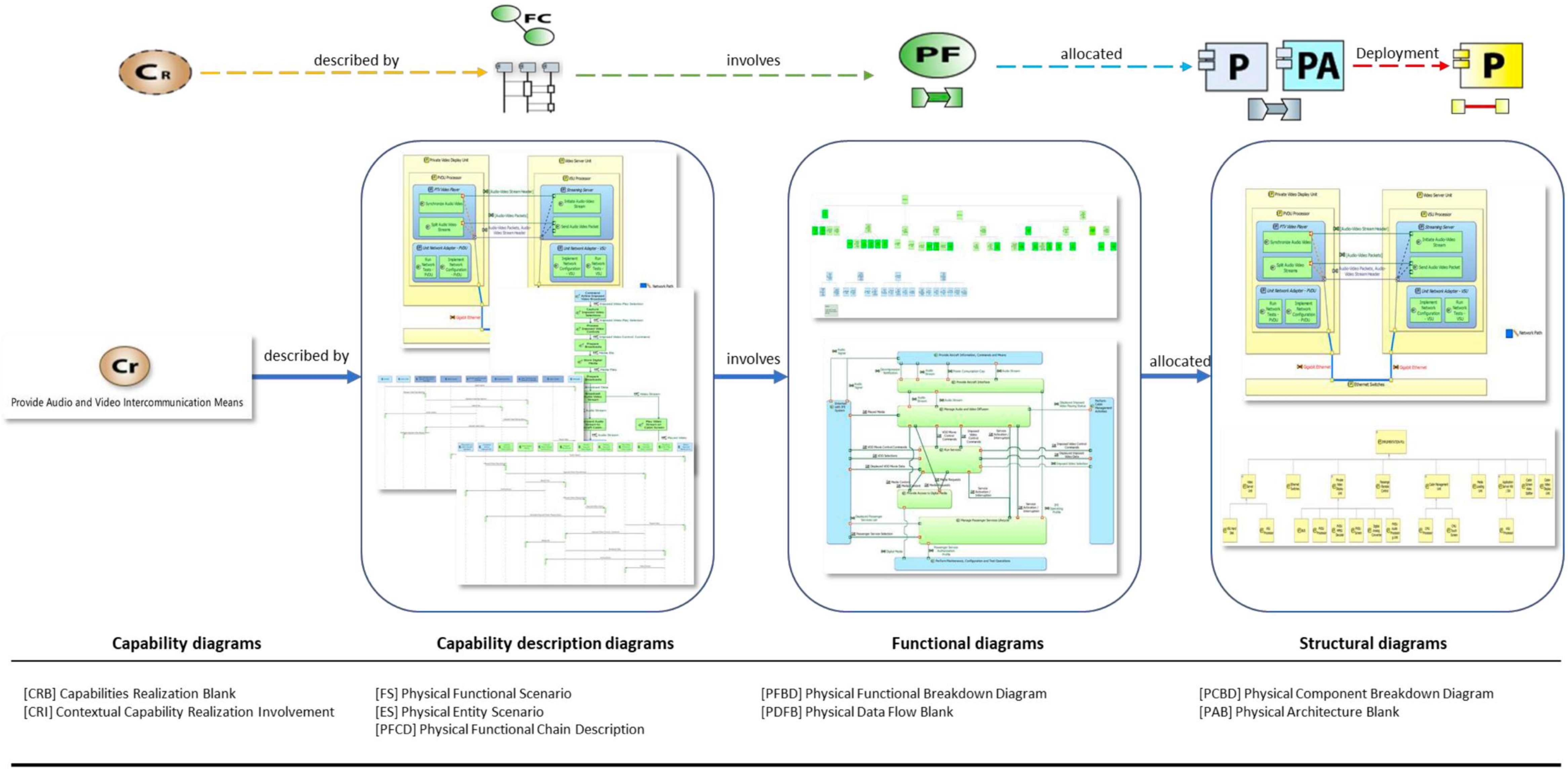


Figure 7.7: Physical Architecture traceability flow

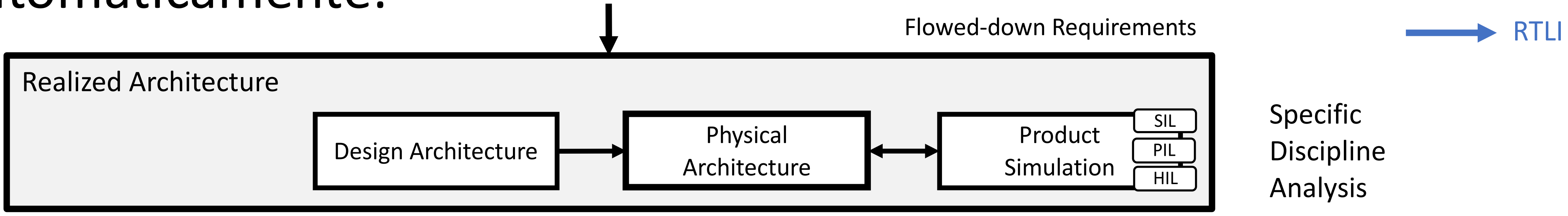
Final Considerations

SOME THOUGHTS

- Physical architecture answers:
  - ***"how the system will be built"***
- In the physical architecture the system choices are concretized.
- Focus on technological trades that implements the functions
- Focus in the realize the "to-be"
- Maps the technologies into the designed functions.

Atividades para a próxima aula

- Fazer a etapa da formalização do sistema construído
- Apresentar como o sistema logístico vai ser construído.
- Apresentar o modelo da arquitetura final:
  - Características mínimas: realizar 2 nós com 1 comportamento em cada (mínimo). Anexar os comportamentos vindos da LA. Indicar como o sistema foi “totalmente construído / entregue pelo fabricante”.
- Extra-fun: Gerar um doc com no mínimo uma arquitetura de cada camada (AO-PA), trazer o título do diagrama e o diagrama automaticamente.



Document Generation

https://www.m2doc.org/

conceitos

- O projeto M2Doc fornece a **geração de documentos** do Word (arquivos .docx) com base em um modelo de documento e modelos EMF.
- A abordagem geral consiste na criação de modelos no formato OOXML em que a criação de texto estático se beneficia dos recursos WYSIWYG do Microsoft Word. Partes dinâmicas são inseridas usando um vocabulário dedicado de código de campos OOXML.
- Os campos são usados principalmente para inserir números de página, referências, etc. O M2Doc faz uso de diretivas de geração de documentação. Isso permite uma separação total entre o documento e as diretivas M2Doc.


Model
Document Template
M2Doc
Generated Document

Physical Model Data Dictionary {m:db.name}

Tables

1.2 Tables au niveau du modèle

{m:for table | db.allTables()}

1.2.1 Table {m:table.name}

1.2.1.1 Table {m:table.name} description

Name	{m:table.name}
SGDB	{m:db.DBLibrary()}
Record number	{m:table.recordNumber()}

1.2.1.2 {m:table.name} columns list

Name	*Type*

{m:for column | table.columns}

{m:column.name}	{m:column.typeName()}

{m:endfor }

{m:for column |table.columns}

1.2.1.3 Column {m:column.name} from table {m:table.name}

1.2.1.3.1 Column {m:column.name} description

Nom	{m:column.name}
Type de données	{m:column.typeName()}
Obligatoire	{m:column.isMandatory()}
Commentaire	{m:column.comments}
Clé primaire	{m:column.isPrimaryKey()}
Clé étrangère	{m: column.isForeignKey()}
Nombre d'enregistrements	{m: column.recordNumber()}

{m:endfor}

Saut de page

Physical Model Data Dictionary serie-oracle

Tables

1.2 Tables au niveau du modèle

1.2.1 Table GS_SERIE

1.2.1.1 Table GS_SERIE description

Name	GS_SERIE
SGDB	physicalTypes
Record number	10000

1.2.1.2 GS_SERIE columns list

Name	*Type*
GS_SERIE_ID	**INTEGER**
RF_GENRE_ID	**INTEGER**
RF_PAYS_ID	**INTEGER**
GS_SERIE_NOM	**VARCHAR2**
GS_SERIE_ANNEECRE	**VARCHAR2**
GS_SERIE_ANNEEFIN	**VARCHAR2**
GS_SERIE_DESCRIPTION	**VARCHAR2**
GS_SERIE_LOGO	**VARCHAR2**
GS_XTOPSUP	**VARCHAR2**
GS_XDMAJ	**DATE**

1.2.1.3 Column GS_SERIE_ID from table GS_SERIE

1.2.1.3.1 Column GS_SERIE_ID description

Nom	GS_SERIE_ID
Type de données	INTEGER
Obligatoire	Oui
Commentaire	PK de la table GS_SERIE
Clé primaire	Non
Clé étrangère	Oui
Nombre d'enregistrements	5000

1.2.1.4 Column RF_GENRE_ID from table GS_SERIE

1.2.1.4.1 Column RF_GENRE_ID description

Nom	RF_GENRE_ID
Type de données	INTEGER
Obligatoire	Oui

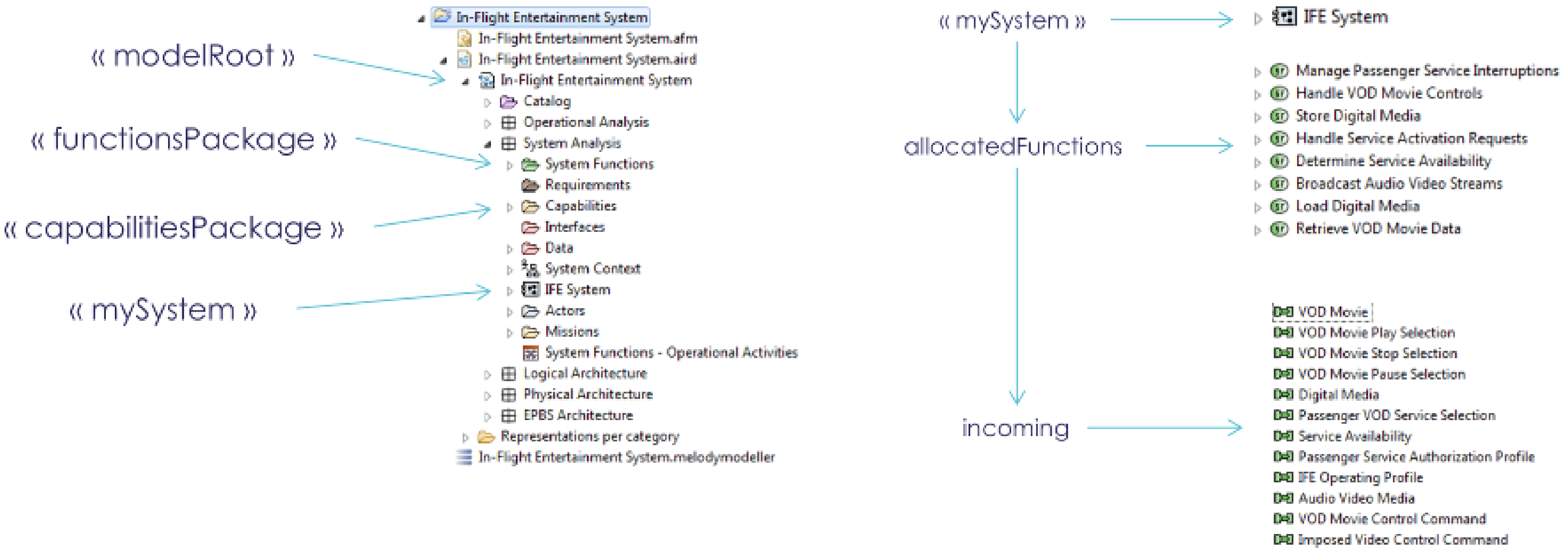
Princípios

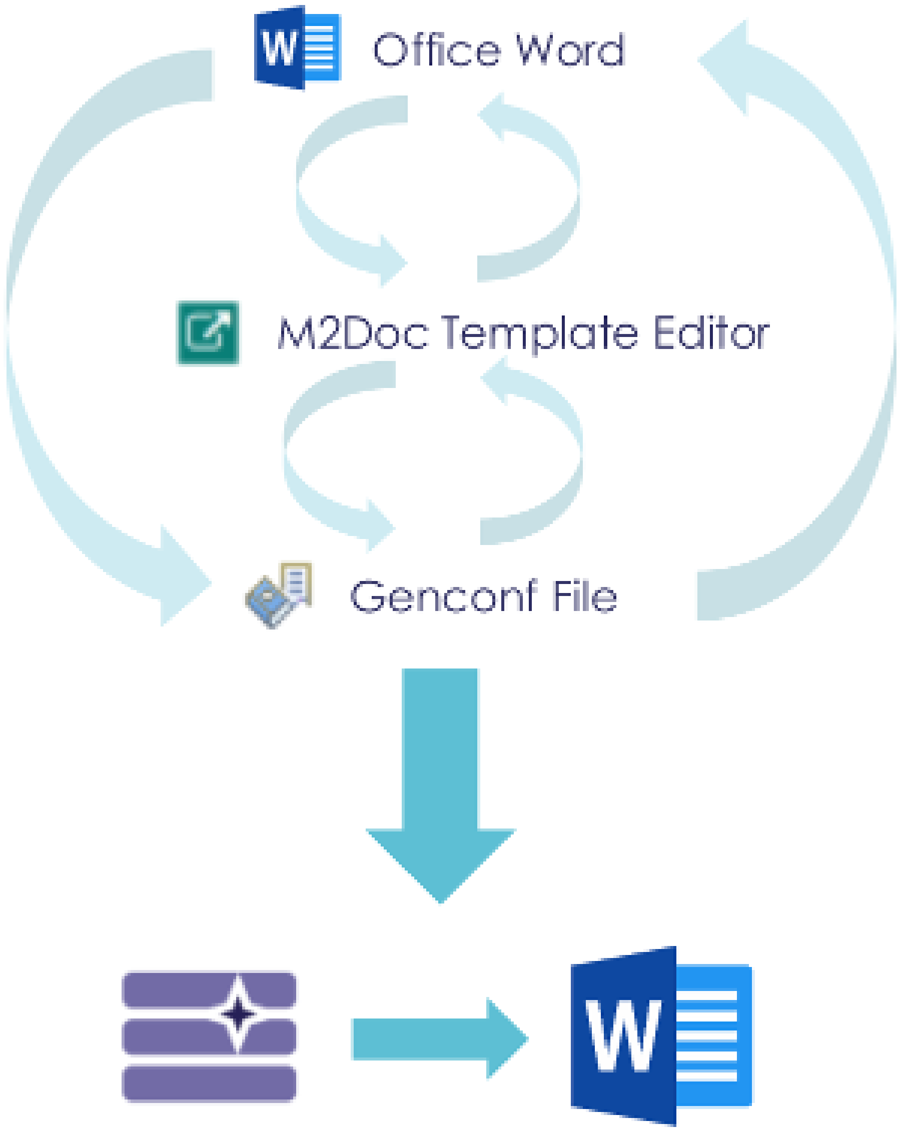
- A linguagem de modelo faz um uso extensivo da **Acceleo Query Language**, que fornece uma linguagem de consulta de modelo.

- Os modelos M2Doc podem ser validados. Se forem encontrados erros, um modelo anotado será produzido descrevendo os problemas encontrados.

Princípios

- Definição dos pontos de entrada do modelo.
- Extração de informações navegando no modelo.





- Definition of content, navigation, and format
- Declaration of variables
- Mapping of variables with model elements, definition of input model and output file
- Generation of output document

Não é fácil e necessita de pessoas específicas:

- **Template user**: que já possuem o modelo e desejam gerar o documento.
  - Edita modelos e quer produzir documentos
- **Template developer**: que querem criar seu próprio modelo
  - Templates podem ser usados em diferentes documentos.
  - Varios templates podem ser usados em um modelo.
  - Conhece a estrutura de navegação AQL
- **Integrator (Meta-model Expert)**: que desejam fornecer geração de documentos em seu próprio projeto usando M2Doc.
  - Cria serviços AQL para criadores de templates.

Básico de metamodelagem

https://www.eclipse.org/modeling/emf/

Metamodelagem

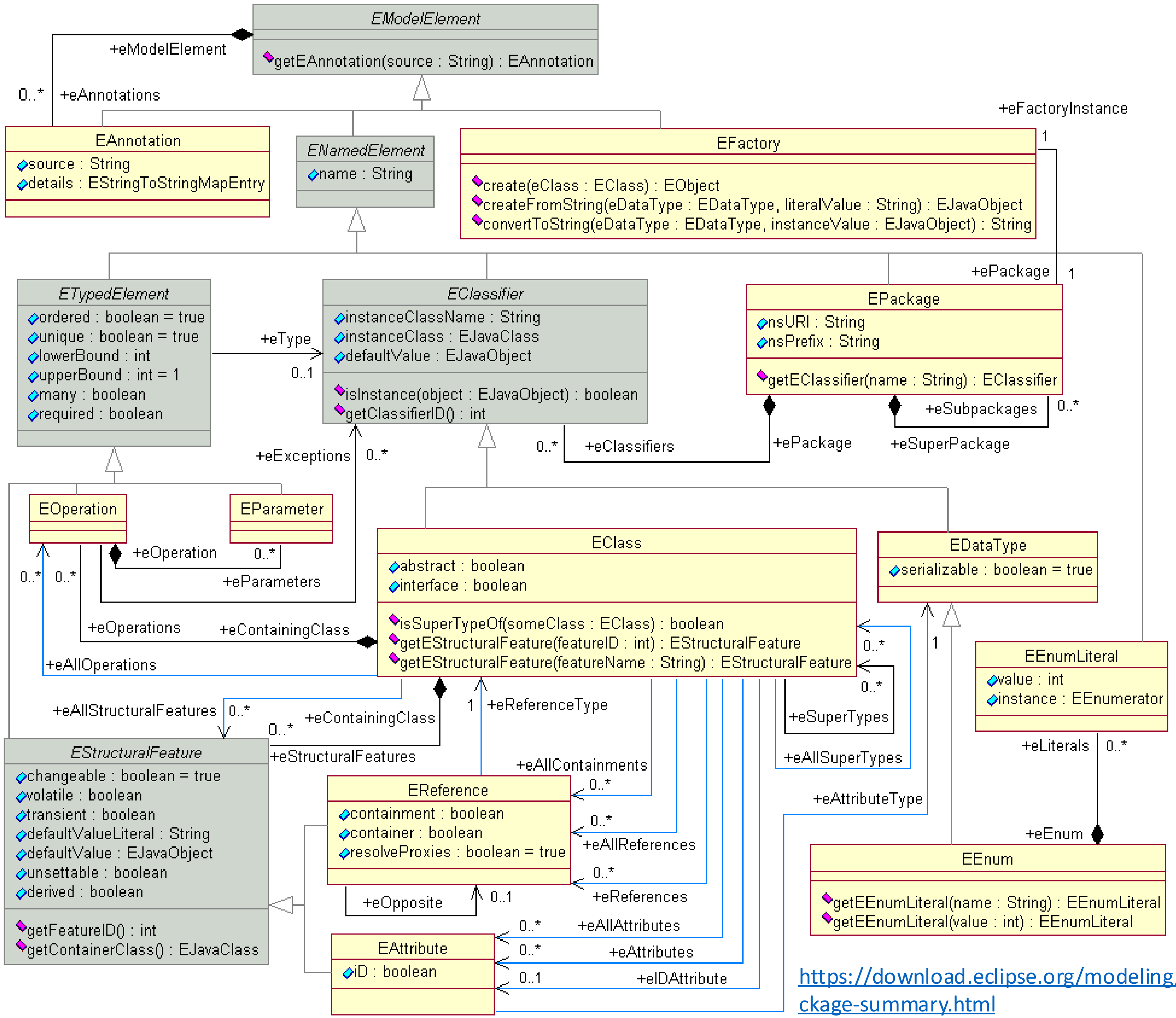
- EMF : Eclipse Modeling Framework
  - Modelagem de dados Java e framework de integração
- Fornece a infraestrutura para realmente usar os modelos em um aplicativo
- Facilmente acessível
- Open source – EPL licence
  - Necessário conhecimento em Java
  - Aliás também em UML, XML Schema
    - Baseado em modelagem > geração de código
- Possibilita a produção rápida de modeladores
- Utiliza toda a facilidade trazida pelo Eclipse

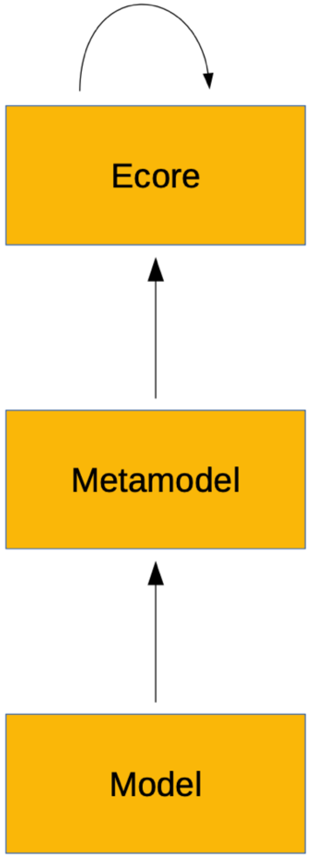
Meta modelo

- Um modelo EMF é um conjunto de dados
  - Conformação a um metamodelo
  - Composto por objetos com propriedades e relações
- Serializado em um arquivo XMI
- Conceitos próximos aos do diagrama de classes UML
  - Elements ~ Class
  - Attributes
  - References ~ Associations

Benefícios

- Um link entre o mundo do desenvolvimento e o mundo da modelagem
  - Converte modelos em código e vice-versa
  - Fornece toda a infraestrutura necessária de M / MM / M2M
- Gratuito
  - Licença EPL
  - Poucos pré-requisitos
  - Pode operar no modo autônomo (sem acesso externo)
- Estabilidade
  - Desenvolvido desde 2002
  - No centro da infraestrutura do Eclipse





https://download.eclipse.org/modeling/emf/emf/javadoc/2.9.0/org/eclipse/emf/ecore/package-summary.html

EclipseSource

Services · Technology · Products · Company · Blogs · Contact · Let's Talk

EMF Tutorial

24 min Read

What every Eclipse developer should know about EMF

This tutorial is an introduction to EMF and explains the basics of EMF. We start by showing you how to build a very simple data-centric application, including the UI, based on EMF. We explain how to define a model in EMF and generate code from it. We explore the API of the generated code, that is, how to create, navigate and modify model instances.

Next we demonstrate how to build a UI based on this model using databinding. For our example, we build an application to manage a bowling league, including matches and players. Later on in the tutorial, we explore the advantages of using AdapterFactories and briefly look at data management in EMF. We also include a few pointers on the most important add-on technologies for EMF. If you are interested in getting fast results building an application based on EMF, maybe EMF Client Platform is also a good starting point for you, see this tutorial.

PDF Download: This tutorial is also available for download as a PDF on our website.

Installation Requirements: To work through the examples, you'll need to download and install a fresh version of the Eclipse Modeling Tools from the Eclipse Download Page.

Introduction

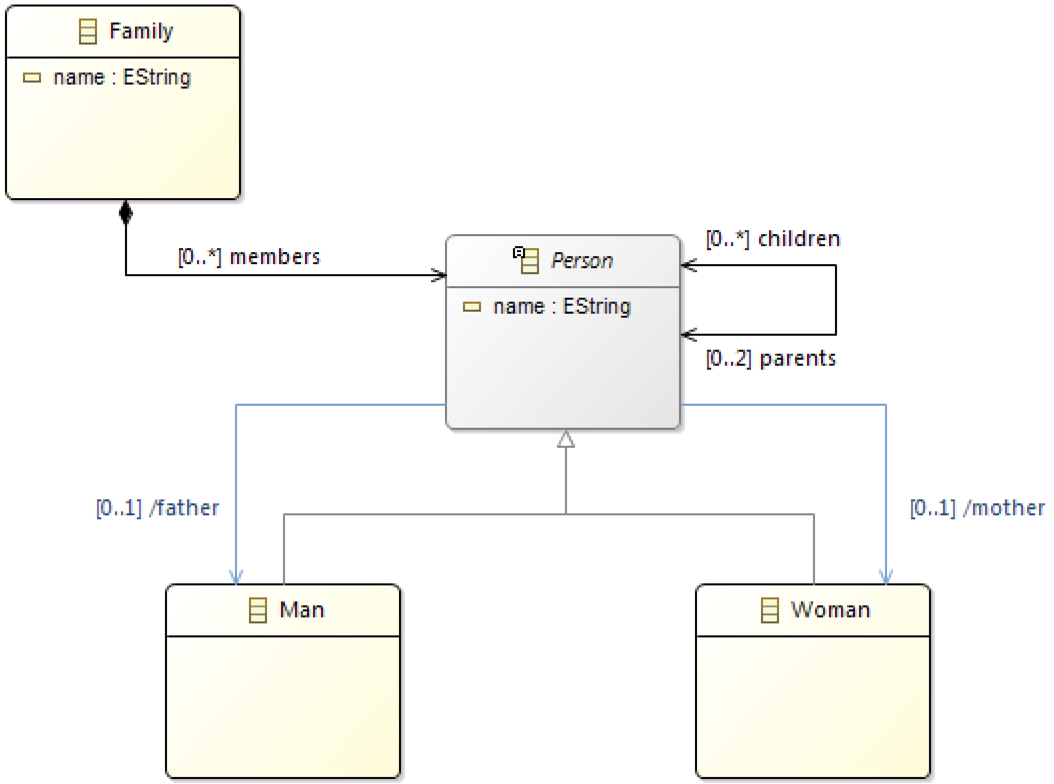
https://eclipsesource.com/blogs/tutorials/emf-tutorial/

Acceleo Query Language

introdução

- A **Acceleo Query Language** (AQL) é uma linguagem usada para navegar e consultar um modelo EMF.
- O AQL como mecanismo de consulta é pequeno, simples, rápido, extensível e traz uma validação mais rica do que o interpretador MTL (Model to Text Language)
- O interpretador AQL é usado no Sirius com o prefixo «aql:»

- From a variable one can access field or reference values using the **.** separator.
- With self being an instance of Person, self.name returns the value of the attribute name and self.father return the father of the person.
- If the attribute or the reference is multi-valued, then self.parents will return a collection.
- Calls can be chained, as such self.parents.name will return a collection containing the names of the parents.
- If one want to access the collection itself, then the separator -> must be used, as such self.parents.name->size() will return the number of elements in the collection whereas self.parents.name.size() will return a collection containing the sizes of each name.



Sintaxe

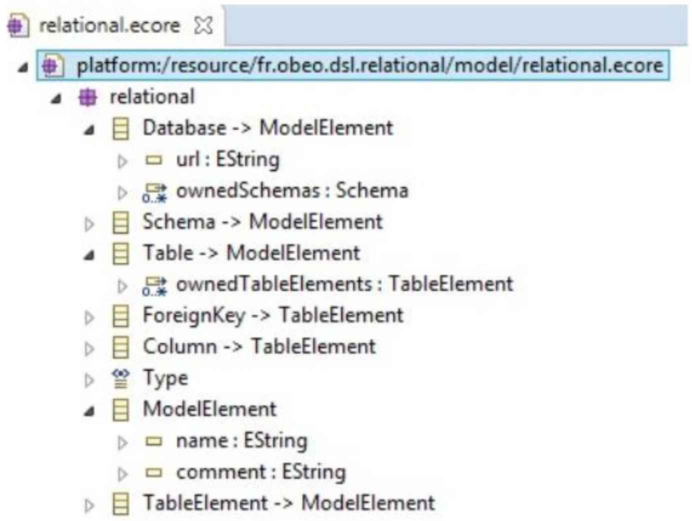
- A noção de contexto
  - Uma consulta AQL se aplica ao seu contexto (denotado por “self”)
- A navegação está em conformidade com o metamodelo usando a notação de pontos
  - Acesso a referências e atributos
  - Exemplo : aql:self.ownedSchemas.name
  - No contexto de um Banco de Dados, recupere os nomes dos esquemas de propriedade do banco de dados atual.

Sintaxe

- Outros exemplos de consulta
  - aql:self.name in a Table context → Name of the Table
  - aql:self.ownedTableElements.name in the context of

    Table → Column names list
  - aql:self.ownedTableElements→size() in the context of a

    Table → Number of columns of the Table
- Elementos estáticos vs elementos dinâmicos
  - aql:'Table_ ' + self.name in the context of a Table named 'Vehicle' → “Table_Vehicle”
  - aql:’Prefixe' + self.name + ‘Sufixe' → “PrefixeVehicleSufixe”



Estruturando as buscas: interpreter

- Disponível em Window > Show View > Interpreter
- Trabalha com representações do Sirius
- Mas também em todos os modelos EMF
- Muito útil para ajustar consultas

Acceleo Query Language

Query and navigate in EMF models

Overview

The Acceleo Query Language (AQL) is a language used to navigate and query an EMF model. In this document, you will find the description of all the services of the standard library of AQL.

Introduction

The Acceleo Query Language (AQL) is a language used to navigate and query an EMF model. In this document, you will find the description of the syntax, all the services and the standard library of AQL.
AQL as a query engine is small, simple, fast, extensible and it brings a richer validation than the MTL interpreter.

For those looking for a simple and fast interpreters for your EMF models, AQL can provide you with a lot of features, including:

- Support for static and dynamic Ecore models, no query compilation phase required.
- The least possible overhead at evaluation time. During this phase, the evaluation goes forward and will not even try to validate or compile your expressions. Errors are tracked and captured along the way.
- Strong validation: types are checked at validation time and the metamodels used are analyzed to do some basic type inference

https://www.eclipse.org/acceleo/documentation/

(super) Pequeno exemplo...

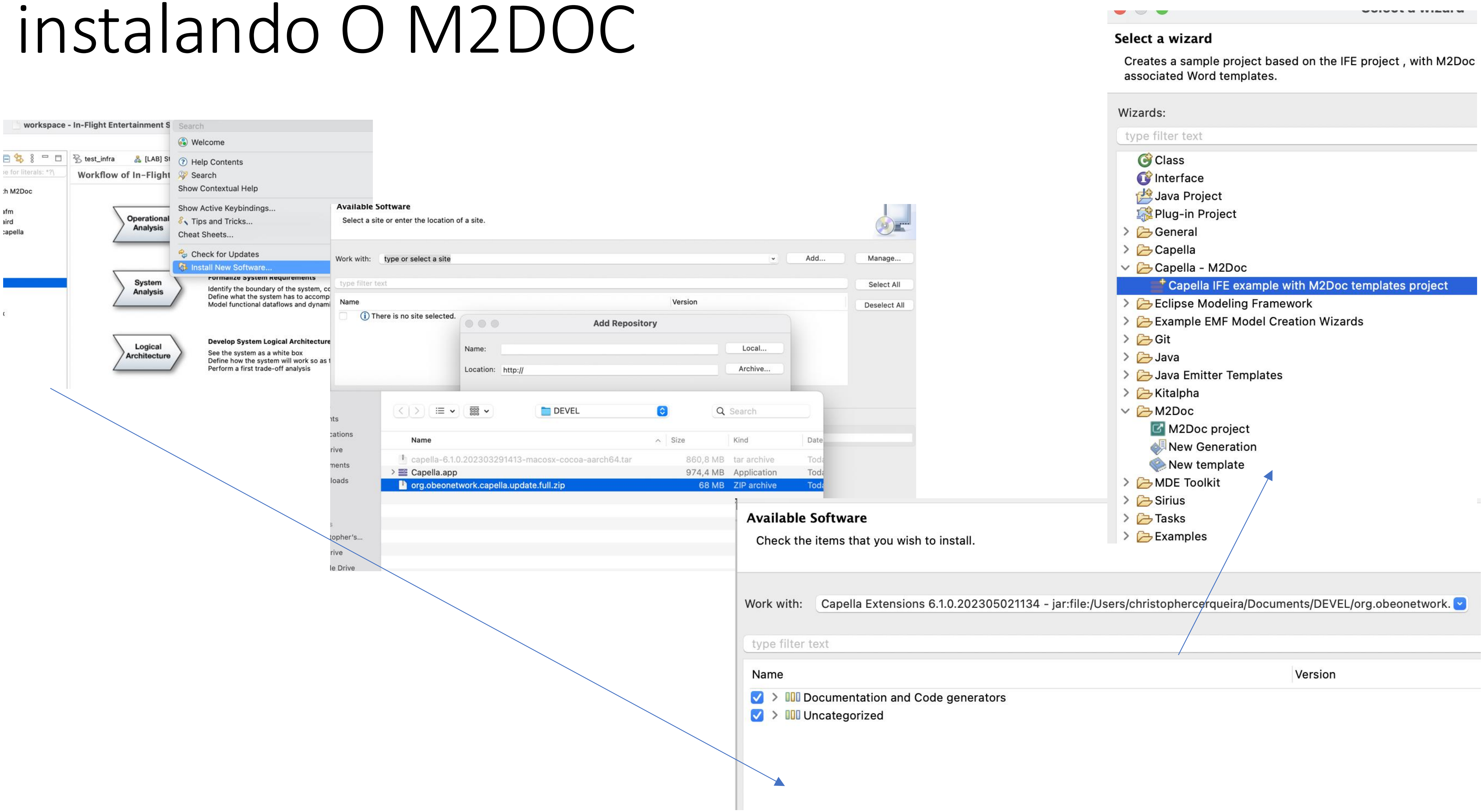
https://www.m2doc.org/

Instalação

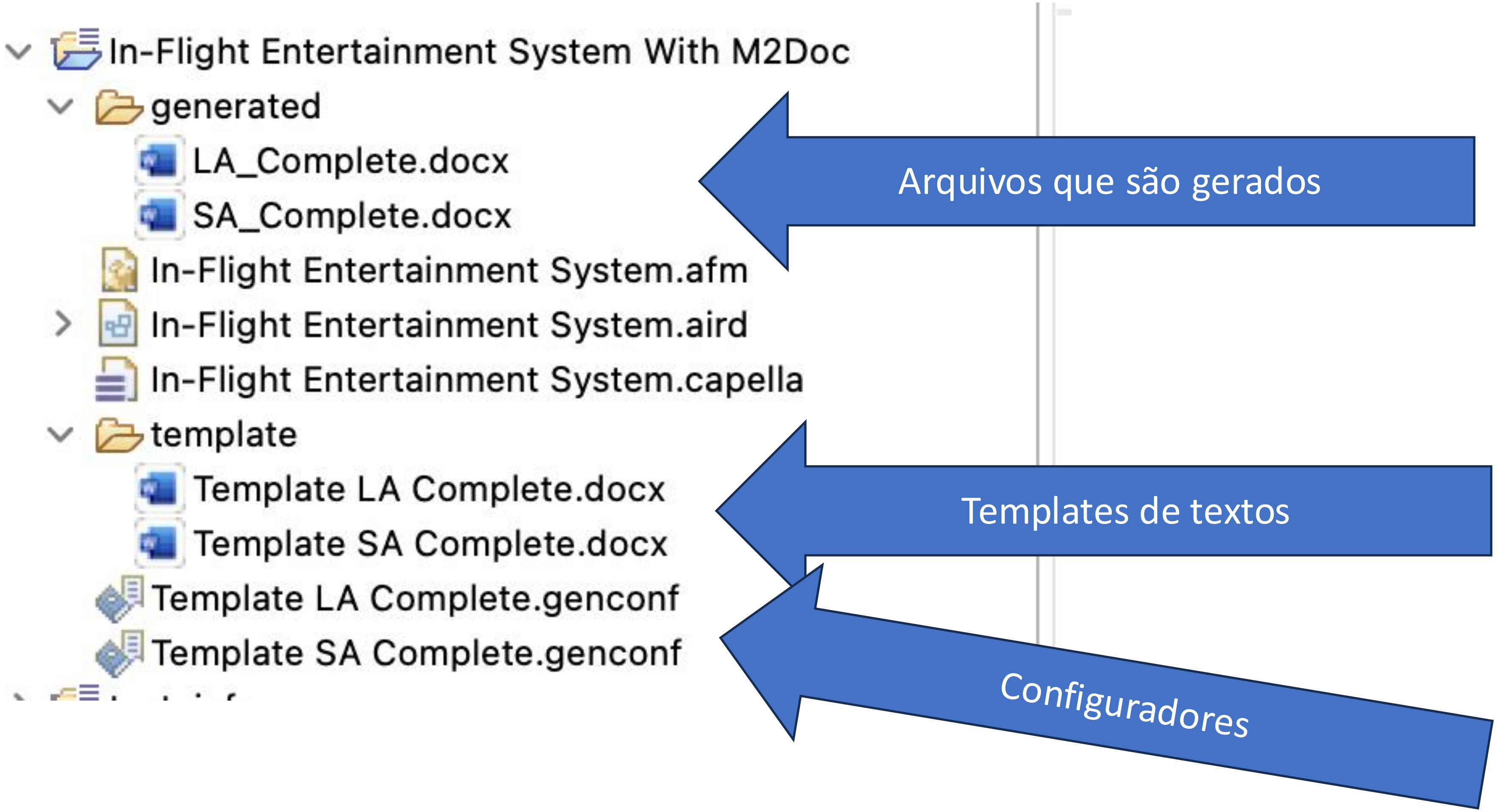
- Primeiro você precisa baixar Capella.
- Quando o download estiver concluído, extraia o arquivo baixado e execute o executável Eclipse na subpasta Eclipse.

- **INSTALLATION FOR CAPELLA 6.1.X**
  - https://s3-eu-west-1.amazonaws.com/obeo-networkaggregation-releases/capella-extensions/6.1.0_M2Doc3.3.0/full **zip** (M2Doc 3.3.0)

instalando O M2DOC



Abrindo o exemplo



ctrl+f9

3 → Description of the system Missions

{m:if self.containedSystemAnalysis.ownedMissionPkg.eAllContents(ctx::Mission)->size() > 0}

{m:if self.containedSystemAnalysis.ownedMissionPkg.eAllContents(ctx::MissionPkg)->size() > 0}

The system Missions are sorted in the following packages:

Mission package	Missions
{m:self.containedSystemAnalysis.ownedMissionPkg.name}	{m:for mission | self.containedSystemAnalysis.ownedMissionPkg.ownedMissions} {m:mission.name.asBookmarkRef(mission.id)} {m:endfor}

{m:for package \| self.containedSystemAnalysis.ownedMissionPkg.eAllContents(ctx::MissionPkg)}

{m:package.name}	{m:for mission |package.ownedMissions} {m:mission.name.asBookmarkRef(mission.id)} {m:endfor}

{m:endfor}

{m:endif}

{m:for mission \| self.containedSystemAnalysis.ownedMissionPkg.eAllContents(ctx::Mission)}

3.1 → Mission : {m:mission.name.asBookmark(mission.id)}

Description:

{m:if mission.description.trim().size() <> 0}

{m:mission.description.trim().fromHTMLBodyString().replaceLink(mission)}

{m:else}

No description

{m:endif}

Involved actors:

{m:if mission.involvedSystemComponents->select(sc \| sc.actor)->size() > 0}

{m:for actor \| mission.involvedSystemComponents->select(sc \| sc.actor)}

{PAGE}

3 → Description of the system Missions

3.1 → Mission : Provide Entertainement Solutions

Description:

No description

Involved actors:

- → { REF c8b78c78-5b11-4fc0-87b7-3ca84622efea \h }
- → { REF 181a678c-dca9-46c1-9d18-b5a0c457c0de \h }

Exploited capabilities:

3.1.1 → Capability : Provide Moving-Map Services

Description:

[Wikipedia]
A moving-map system is a real-time flight information video channel broadcast through to cabin project/video screens and personal televisions (PTVs). In addition to displaying a map that illustrates the position and direction of the plane, the system gives the altitude, airspeed, outside air temperature, distance to the destination, distance from the origination point, and local time. The moving-map system information is derived in real time from the aircraft's flight computer systems.

Capability inclusion relations:

Including capabilities	Current capability	Included capabilities
No including capability	Provide Moving-Map Services	*No included capability*

Capability extension relations:

Extended capabilities	Current capability	Extending capabilities
No extended capability	Provide Moving-Map Services	*No extending capability*

Capability generalization relations:

Super capabilities	Current capability	Sub capabilities

{PAGE}

Alt+f9

3 → Description of the system Missions

3.1 → Mission : Provide Entertainement Solutions

Description:

No description

Involved actors:

- → Aircraft
- → Passenger

Exploited capabilities:

3.1.1 → Capability : Provide Moving-Map Services

Description:

[Wikipedia]
A moving-map system is a real-time flight information video channel broadcast through to cabin project/video screens and personal televisions (PTVs). In addition to displaying a map that illustrates the position and direction of the plane, the system gives the altitude, airspeed, outside air temperature, distance to the destination, distance from the origination point, and local time. The moving-map system information is derived in real time from the aircraft's flight computer systems.

Capability inclusion relations:

Including capabilities	Current capability	Included capabilities
No including capability	Provide Moving-Map Services	*No included capability*

Capability extension relations:

Extended capabilities	Current capability	Extending capabilities
No extended capability	Provide Moving-Map Services	*No extending capability*

Capability generalization relations:

Super capabilities	Current capability	Sub capabilities

8