

Modern Requirements and Business Analysis

Jean-Michel Bruel -- 2023/09/05

<https://bit.ly/jmbruel>



@SmartModelTeam



<https://github.com/smart-researchteam>

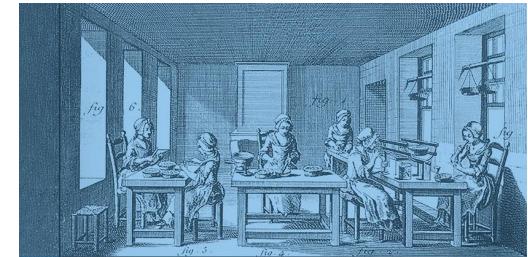


Get the 180 slides (pdf)



Outline

- Context
- Requirements Anatomy
 - Categories of requirements
 - Categories of inter-requirements relations
- Requirements tooling
 - There is more than Word and Excel
 - Concrete implementation (*of what comes next*)



Bertrand Meyer
Handbook of Requirements and Business Analysis

 Springer

<https://se.inf.ethz.ch/requirements/>

Outline

- 9h-10h30 Context / Brainstorming
- 11h-12h30 PEGS approach / Reqs taxonomy
- 14h-15h30 Tooling / Practice
- 16h-17h30 Case studies / Feedback

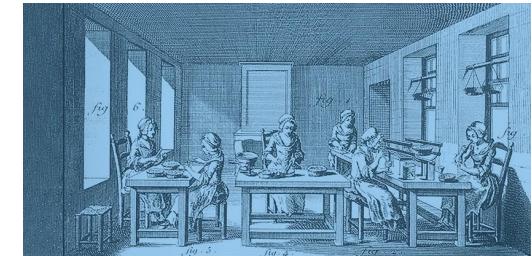




Why me?

- Professor at Toulouse University
 - Teaching **modeling** and **DevOps**
- Member of the CNRS-IRIT Laboratory
 - Model-Based **Systems Engineering**
- **Airbus** MBSE Chair of Toulouse
- Leader of the **companion book** (end of 2023)

<https://bit.ly/jmbruel>



Bertrand Meyer
Handbook of Requirements and Business Analysis

 Springer

<https://se.inf.ethz.ch/requirements/>



Why NOT me?



Disclaimer...

Sorry!

- ... for being the first attendees of this **first tutorial**
- ... for my **English**

We are not prescriptive!



<https://noharmspilt.com/2015/10/09/descriptive-vs-prescriptive-grammar/>

Outline

- Context
- Requirements anatomy
- Requirements tooling

Context



https://www.linkedin.com/posts/daniel-abrahams_reminder-people-dont-buy-products-they-ugcPost-7010015948820680704-CTJD?utm_source=share&utm_medium=member_android

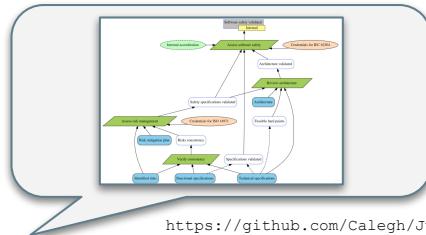


People don't buy **products**
They buy **solutions** to their **problem**

[...] they buy solutions to their problem



- **Play** with the product
 - Not so easy with an airplane...
- Don't need details
 - **Early** V&V
- Validation => **Rational**



<https://github.com/Calegh/JustificationDiagram>

Joint effort...

- Innopolis University
 - Alexandr 
 - Bertrand 
 - Manuel 
- Constructor Institute
 - Bertrand 
 - Li 
- IRIT/SM@RT team
 - Florian 
 - Sophie 
 - JMB 
 - Maria 
- CoCoVaD
 - Imen Sayar 
 - Thuy Nguyen 



Validation & Verification (V&V)

Does the right thing

- Validation
- « Building the right system »



<https://www.canon.co.nz/software-solutions/iw-sam>

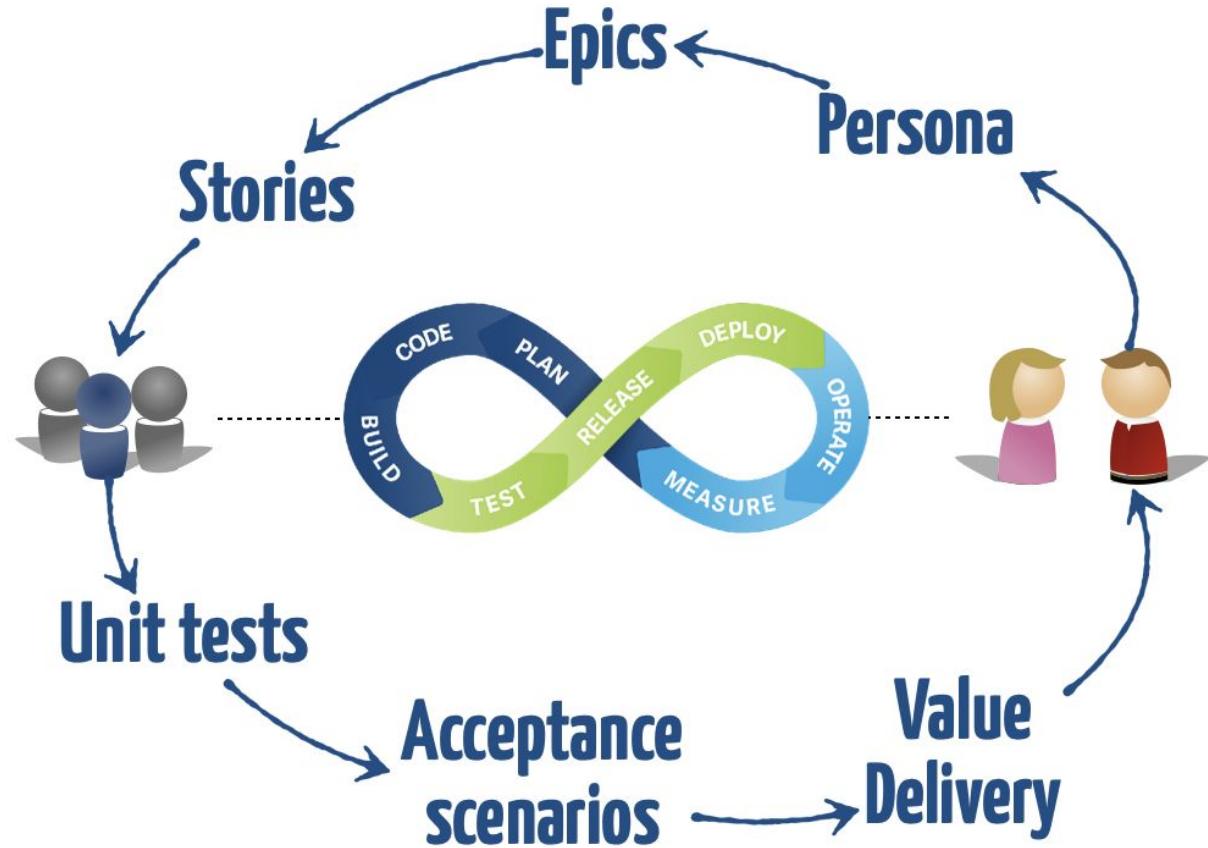
Does them right

- Verification
- « Building the system right »



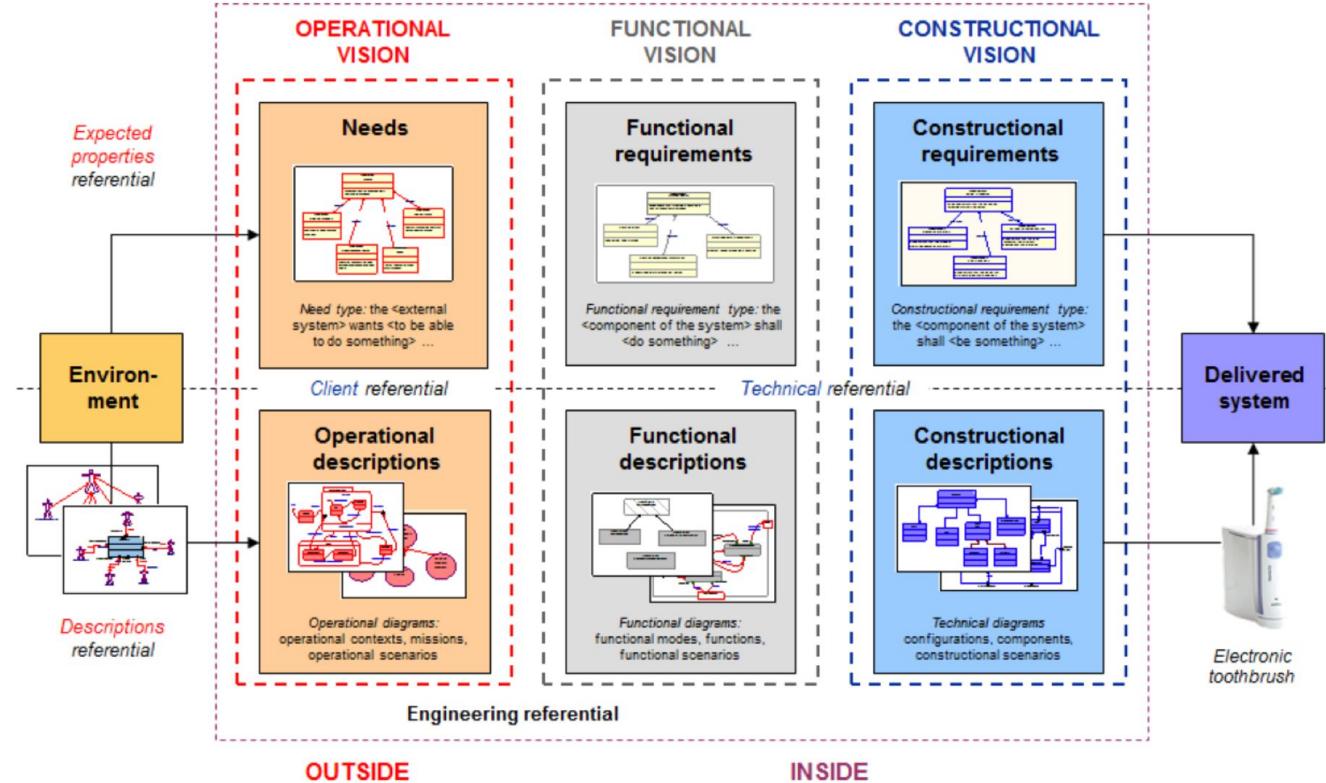
<https://www.techopedia.com>

Lean development

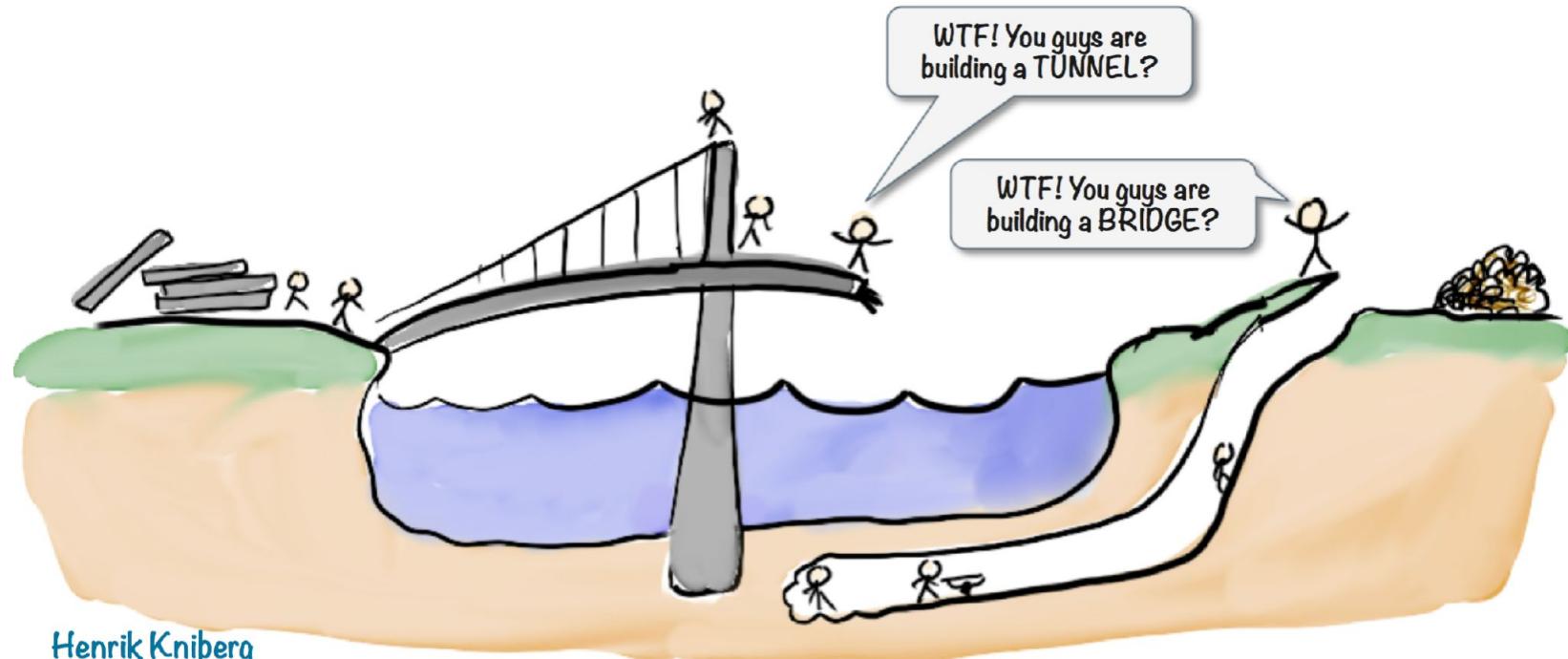


Source: <http://meshfields.de/continuous-integration-testing-delivery-ionic2-hybrid-mobile-apps-buddybuild/>

Systems Engineering

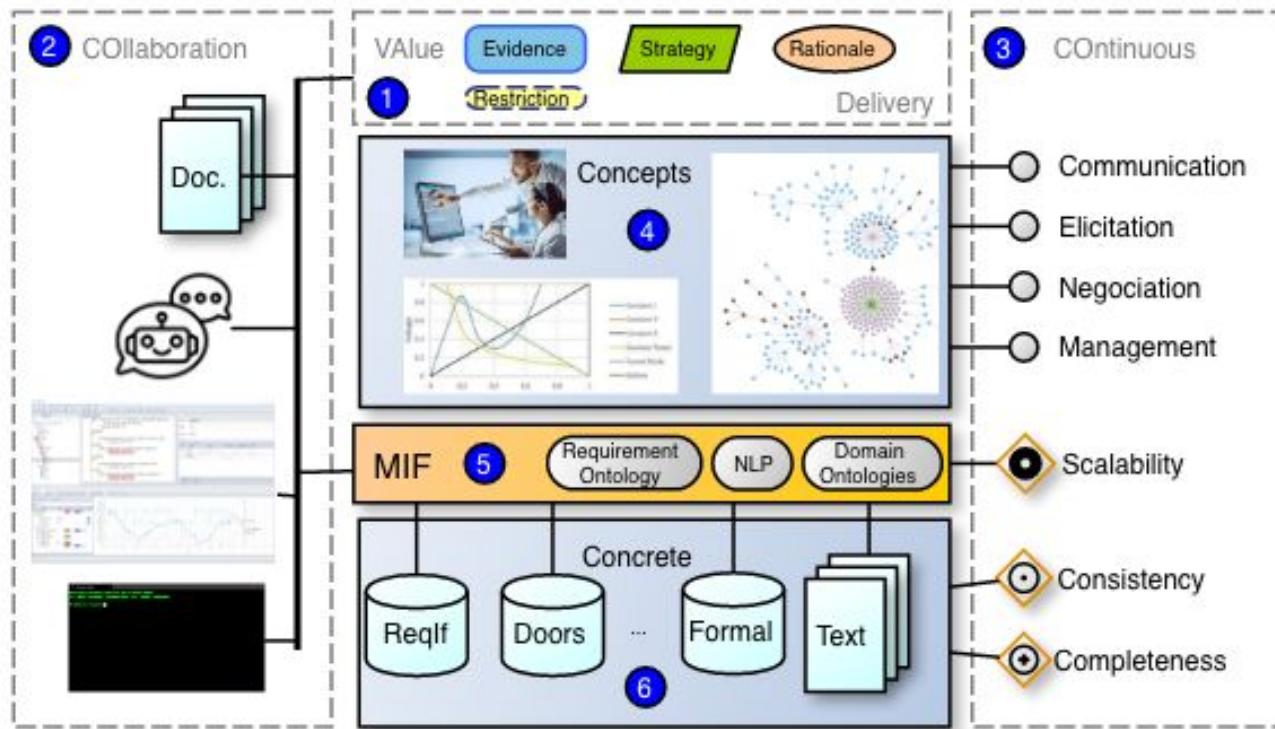


Misalignment



Henrik Kniberg

Requirements as first-class citizens



IEEE/SWEBOK/ISO definition of a Requirement

“A 1.1 Definition of a Software Requirement

At its most basic, a software requirement is a property that must be exhibited by something in order to solve some problem in the real world. It may aim to automate part of a task for someone to support the business processes of an organization, to correct shortcomings of existing software, or to control a device—to name just a few of the many problems for which software solutions are possible. The ways in which users, business processes, and devices function are typically complex. By extension, therefore, the requirements on particular software are typically a complex combination from various people at different levels of an organization, and who are in one way or another involved or connected with this feature from the environment in which the software will operate.

”

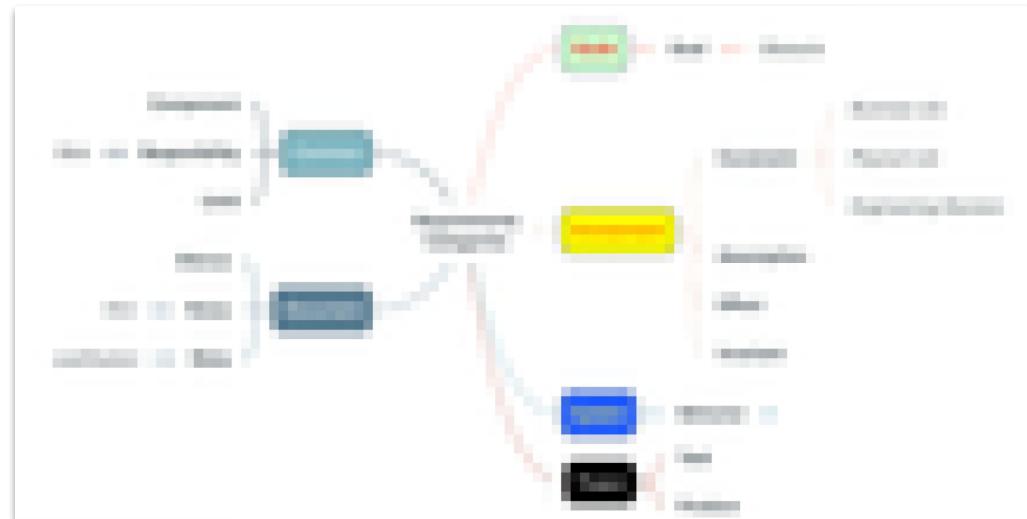
http://swebokwiki.org/Chapter_1:_Software_Requirements



Practice

Practice 1: Collaborative experience

1. Open collaborative mind map
2. Express categories you know/expect/manipulate
3. Organize and group
4. Discuss

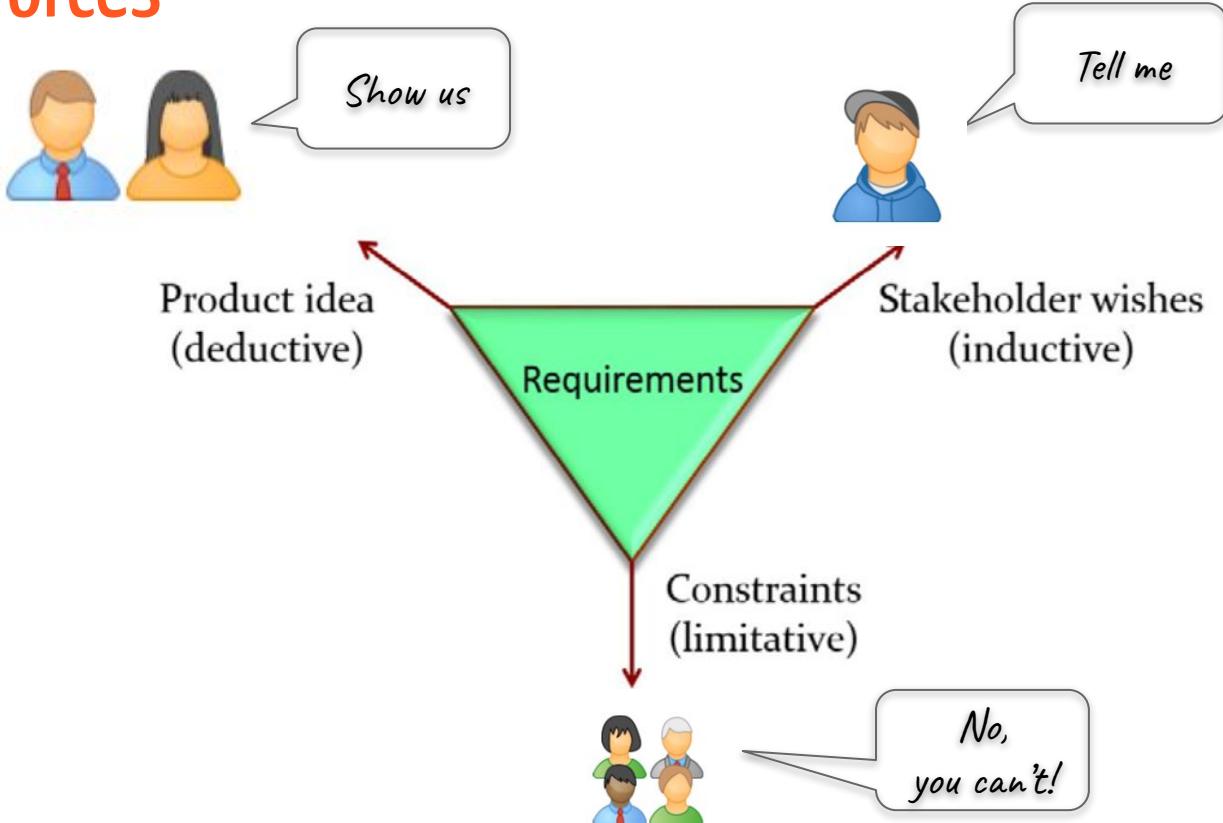


Outline

- Context
- Requirements anatomy
- Requirements tooling

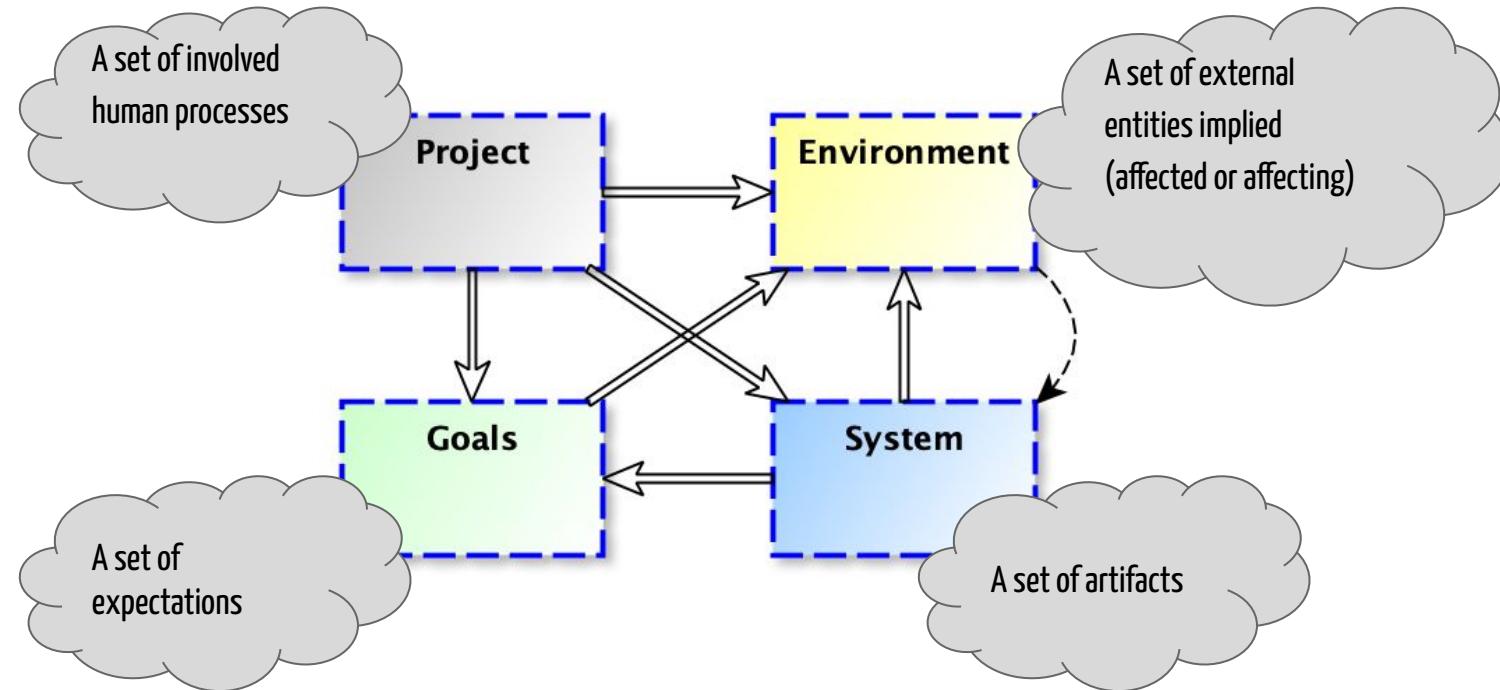
Requirements Anatomy

3 pulling forces



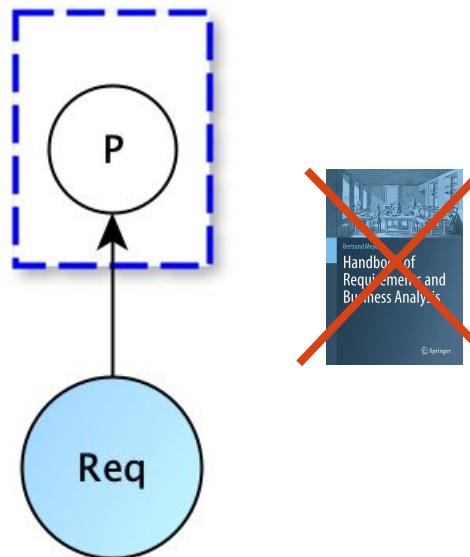
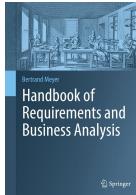
Context (universe of discourse)

“a project to develop a system, in a certain environment, to satisfy a set of goals”



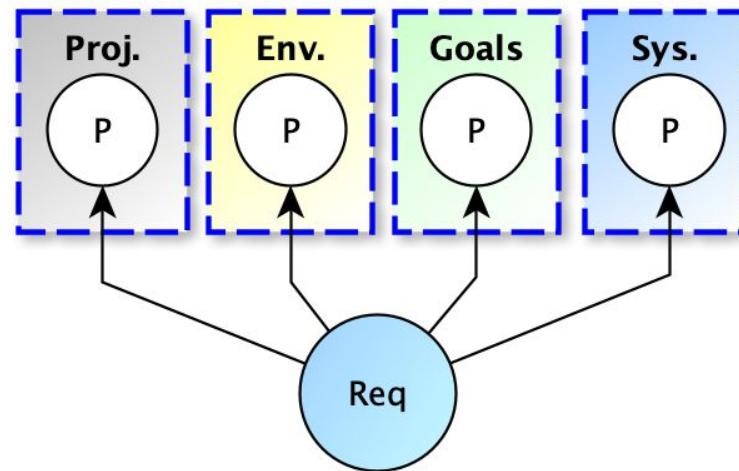
General definition of a Requirement

“A requirement is a (relevant) **statement** about a **property**”



General definition of a Requirement

“A requirement is a (relevant) **statement** about
a project, environment, goals or system **property**”



Some basic concepts

Property: boolean predicate (on a project, system or environment)

Statement: human-readable expression of a property

Relevant: ...

Relevance

Goals: always (by definition)

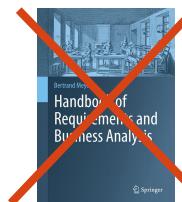
Environment: if it can affect or be affected

System: if it can affect or be affected by a stakeholder

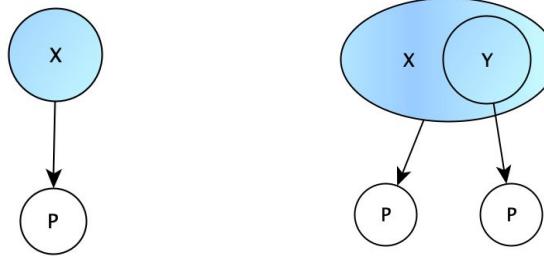
Project: if it can affect or be affected by a stakeholder

A statement of a property is relevant if the property is relevant

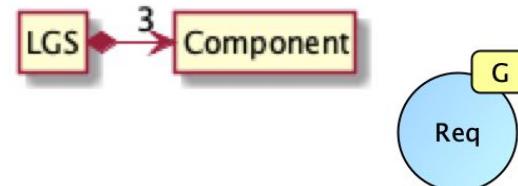
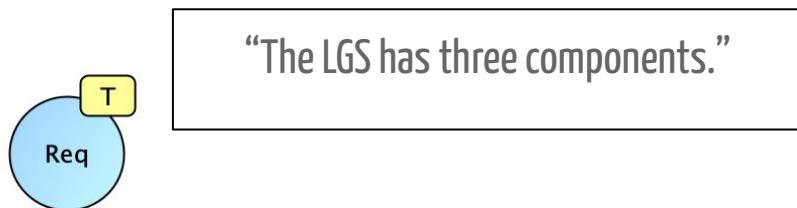
Elements of graphical representation



A requirement can be **Atomic** or **Composite**



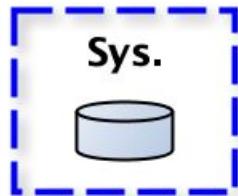
The **notation** of a requirement is the concrete syntax in which it is expressed (Text, Tabular, Graphical, formal)



Additional concepts

We distinguish the different **stages** of a System:

- The **system** itself (mainly to talk about its components)
- The **running system** (mainly to talk about its behavior)
- The system in **development** (mainly to talk about phases and artifacts)

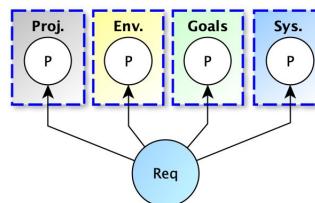




Practice

Practice 2: Find the corresponding Book

1. Some of the general constraints were defined in the preliminary meeting of 15 June 2022, available at [URL].
2. The login record shall be implemented using MongoDB.
3. Here is the basic scheme of interaction for ordering a product: [followed by the description of that scheme].
4. The project shall only use external software products available through an approved open-source license (GPL or Creative Commons).
5. The product shall be available on mobile platforms as well as through an API.
6. Any use of cookies shall conform to the GDPR.
7. As a result of the introduction of the new payroll system, pay periods shall be standardized to monthly for all employees.
8. As the system depends on Windows 11 facilities, meeting the schedule depends on Microsoft fully releasing Windows 11 by end of October, 2021.
9. This function is considered critical to the deployment of the project.
10. Upon exiting a session, the system shall memorize the last explored directory as the restart point for the next session.



The four PEGS

Standard Plan



Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview
- G.5 High-level usage scenarios
- G.6 Limitations and exclusions
- G.7 Stakeholders and requirements sources

Environment (E)

- E.1 Glossary
- E.2 Components
- E.3 Constraints
- E.4 Assumptions
- E.5 Effects
- E.6 Invariants

System (S)

- S.1 Components
- S.2 Functionality
- S.3 Interfaces
- S.4 Detailed usage scenarios
- S.5 Prioritization
- S.6 Verification and acceptance criteria

Goals

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview
- G.5 High-level usage scenarios
- G.6 Limitations and exclusions
- G.7 Stakeholders and requirements sources

Environment

Environment (E)

- E.1 Glossary
- E.2 Components
- E.3 Constraints
- E.4 Assumptions
- E.5 Effects
- E.6 Invariants

System

System (S)

S.1 Components

S.2 Functionality

S.3 Interfaces

S.4 Detailed usage scenarios

S.5 Prioritization

S.6 Verification and acceptance criteria

Project

Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

Outline

- Context
- Categories of requirements
- Categories of inter-requirements relations

Kind of requirements (overview)

Kind of requirements (common to all PEGS)

- Component
- Responsibility
 - *Role*
- Limit

Kind of requirements (Goals)

- Goal
 - *Obstacle*

Kind of requirements (Projects)

- Task
- Product

Kind of requirements (System)

- Behavior
 - *Functional*
 - *Non-functional*
 - *Example*

Kind of requirements (Environment)

- Constraint
 - *Business rule*
 - *Physical rule*
 - *Engineering decision*
- Assumption
- Effect
- Invariant

Kind of requirements (Document description)

- Silence
- Noise
 - *Hint*
- Meta-requirement
 - *Justification*

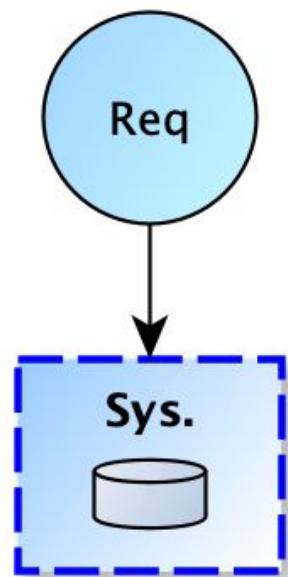
Kind of requirements (details)

Common to all PEGS

- Component
- Responsibility
 - *Role*
- Limit

Component

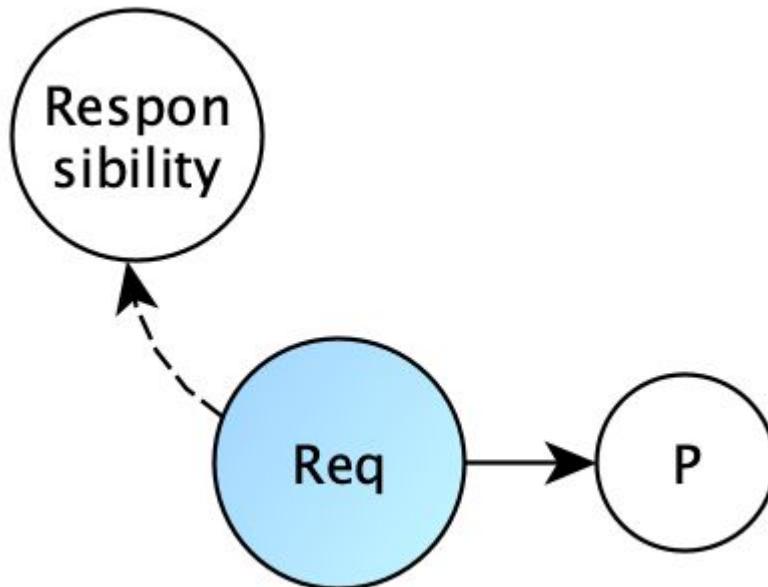
(Identification of a part of a whole)



“The Landing Gear System is composed of three parts.”

Responsibility

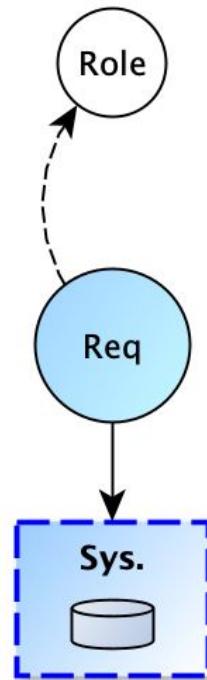
(Assignment of behavior or task to component)



“The control system is in charge of the opening/closing of the door.”

Role (kind of responsibility)

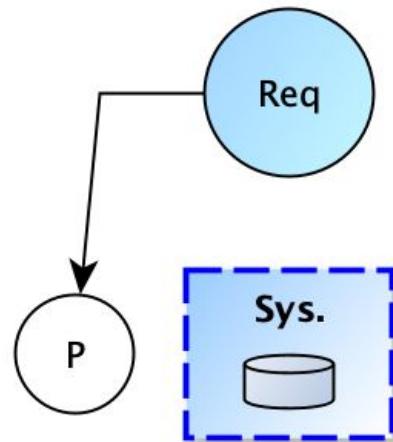
(A human or organizational responsibility)



“Authorizations are provided by the head of the audit department.”

Limit

(the property that the project, system or environment does *not* include a requirement of any of the preceding kinds)



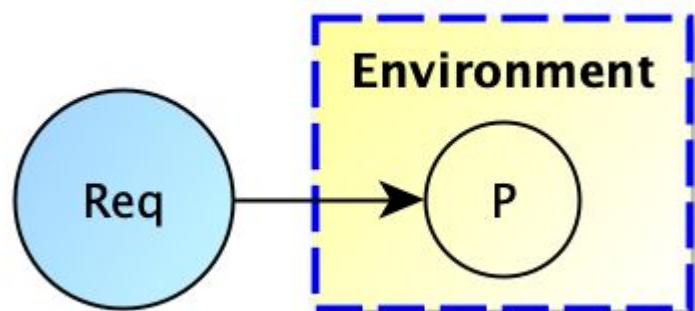
“Integration testing will be performed in a follow-up project.”

Goals Requirements

- Goal
 - *Obstacle*

Goal

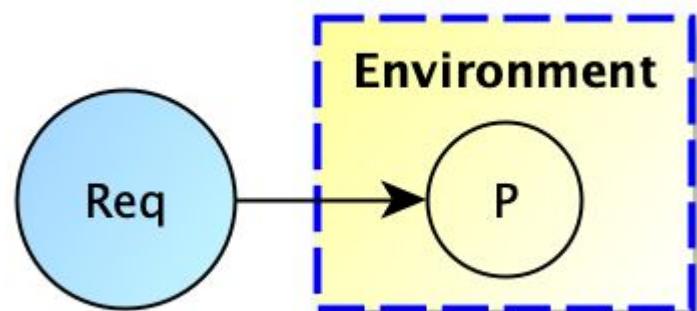
(Desired result for the target organization)



“The goal of the system is to allow any user to book a flight.”

Obstacle (kind of goal)

(Goal describing a property to be overcome)



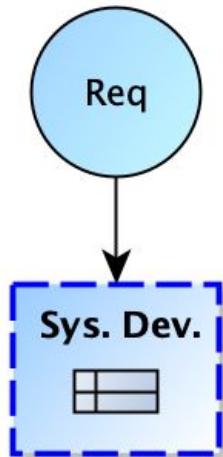
“The current manual operation makes it impossible to meet the expected growth of traffic over the next 10 years.”

Projects requirements

- Task
- Product

Task

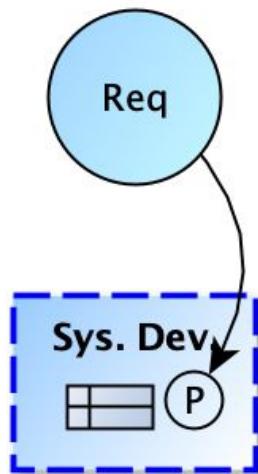
(The property that the project includes a certain activity)



“The team should meet in a daily basis, called daily meeting.”

Product

(Artifact produced or needed by a task)



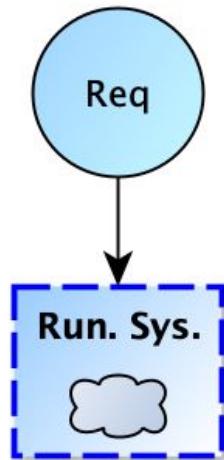
“The following test plan is provided:...”

System requirements

- Behavior
 - *Functional*
 - *Non-functional*
 - *Example*

Behavior

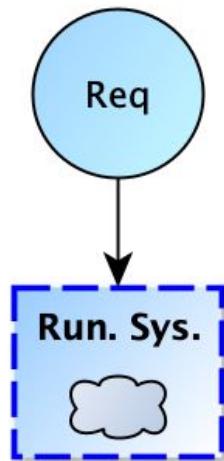
(A property of the effects of the operation of the system or some of its components)



“The system should allow to open
and close the door safely.”

Functional requirement (kind of behavior)

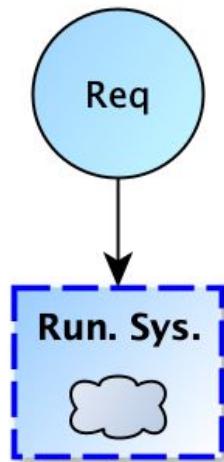
(What the system must do)



“The system should allow to open
and close the door safely.”

Non-functional requirement (kind of behavior)

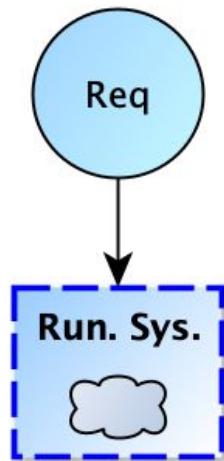
(How the system must perform)



“The identification process should
be secure.”

Example (kind of behavior)

(Illustrative/representative scenario)



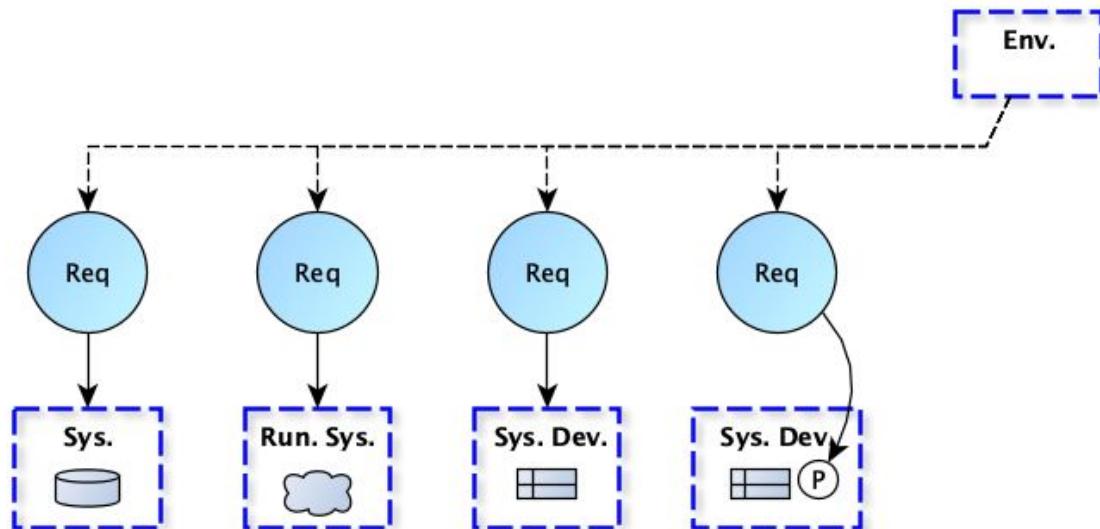
“Here is the description of the use case *cancel a previous order...*”

Environment requirements

- Constraint
 - *Business rule*
 - *Physical rule*
 - *Engineering decision*
- Assumption
- Effect
- Invariant

Constraint

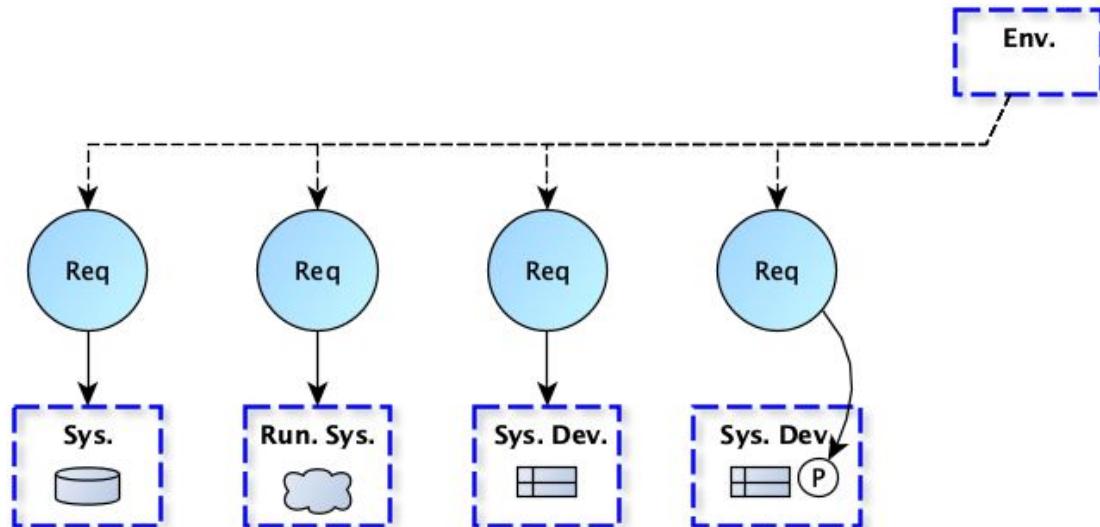
(A property imposed by the environment)



“Every transfer over 10.000€
requires an authorization.”

Business rules (kind of Constraint)

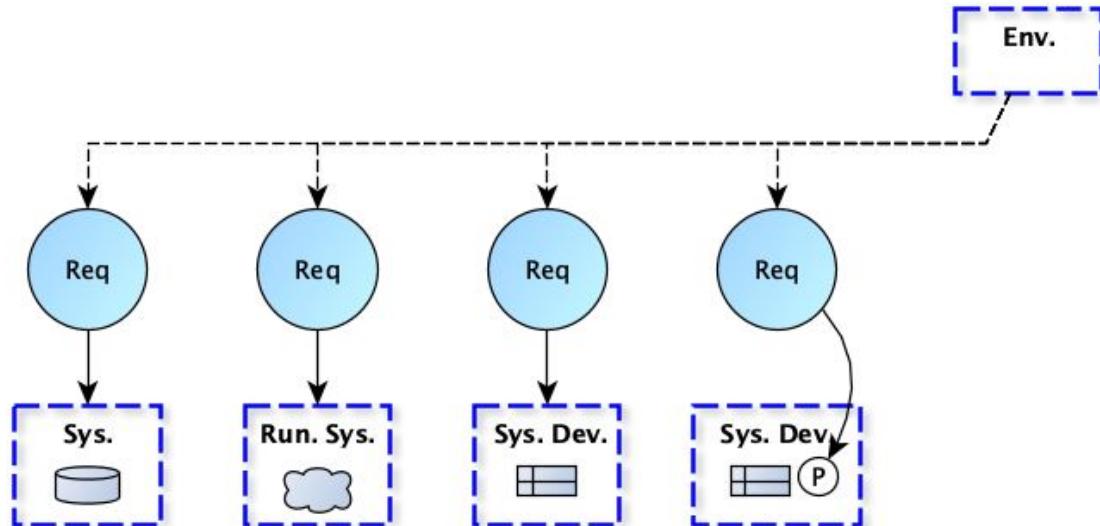
(A constraint imposed by an organization)



“According to the regulation rule X.45F53, the amount of the engine CO₂ emission must be less than...”

Physical rules (kind of Constraint)

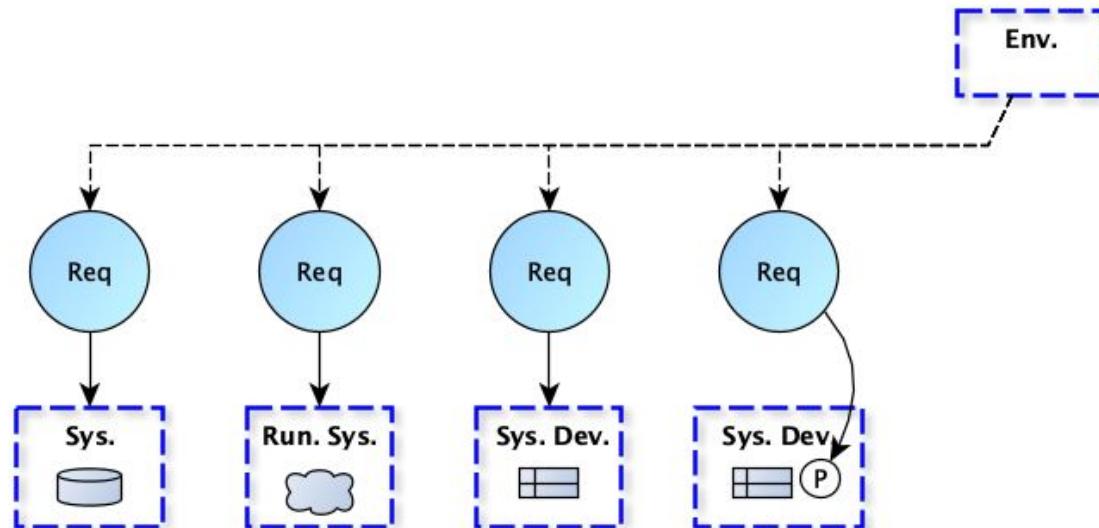
(A constraint imposed by an organization)



“According to the regulation rule X.45F53, the amount of the engine CO₂ emission must be less than...”

Engineering decisions (kind of Constraint)

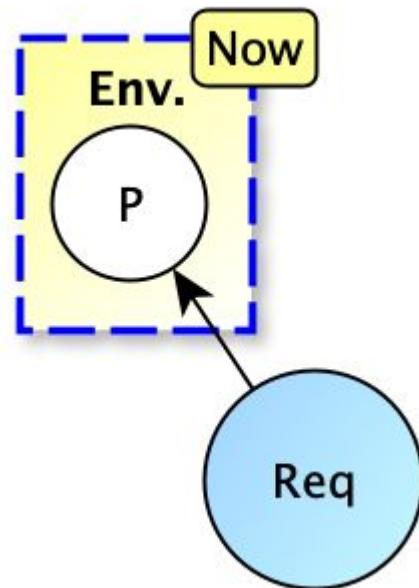
(A constraint imposed by an organization)



“According to the regulation rule X.45F53, the amount of the engine CO₂ emission must be less than...”

Assumption

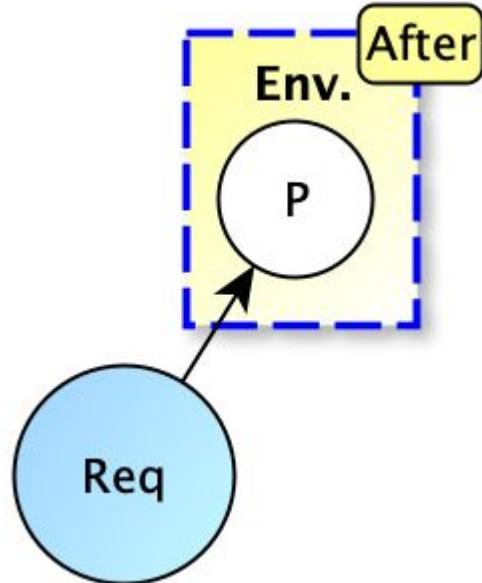
(Expected property of the environment)



“The available bandwidth will be 1 Mbit/s or more.”

Effect

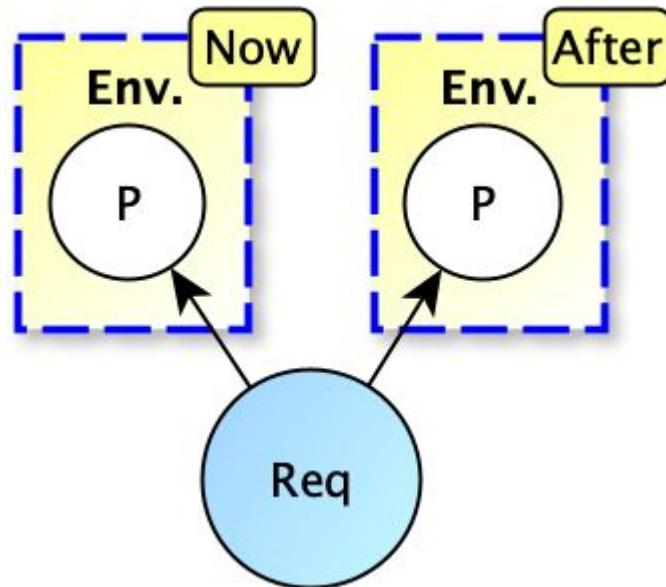
(Property of the environment affected by the system)



“When the system is put into operation, employees will be paid on the last working day of the month.”

Invariant

(Environment property that must be maintained)



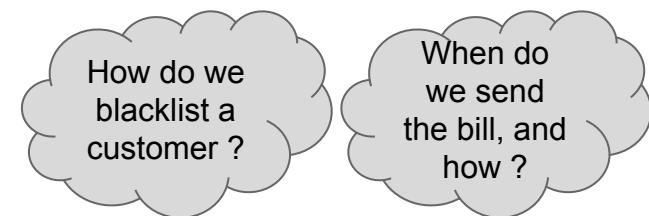
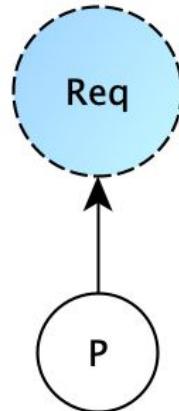
“The system expects a temperature between 18 to 25 degrees Celsius (precondition) and maintains it in that range.”

Document description

- Silence
- Noise
 - *Hint*
- Meta-requirement
 - *Justification*

Silence

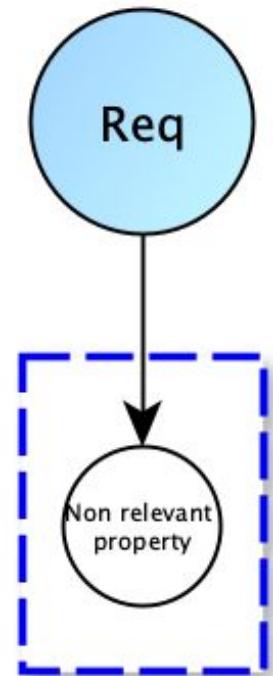
(a property that should have a requirement, but does not)



“The system should send the bill to the non blacklisted customers.”

Noise

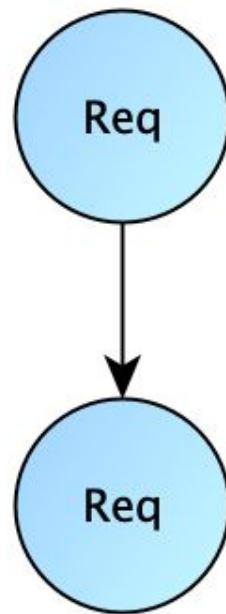
(something that should not be in the requirement document but is there)



“The director is not consistent in his decision making.”

Meta-requirement

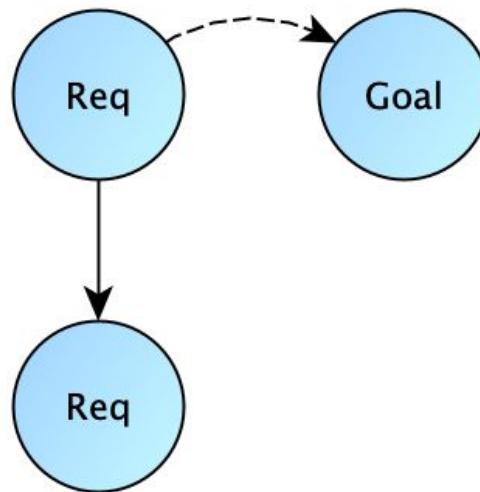
(a property of requirements themselves)



“The details are provided in Fig7.”

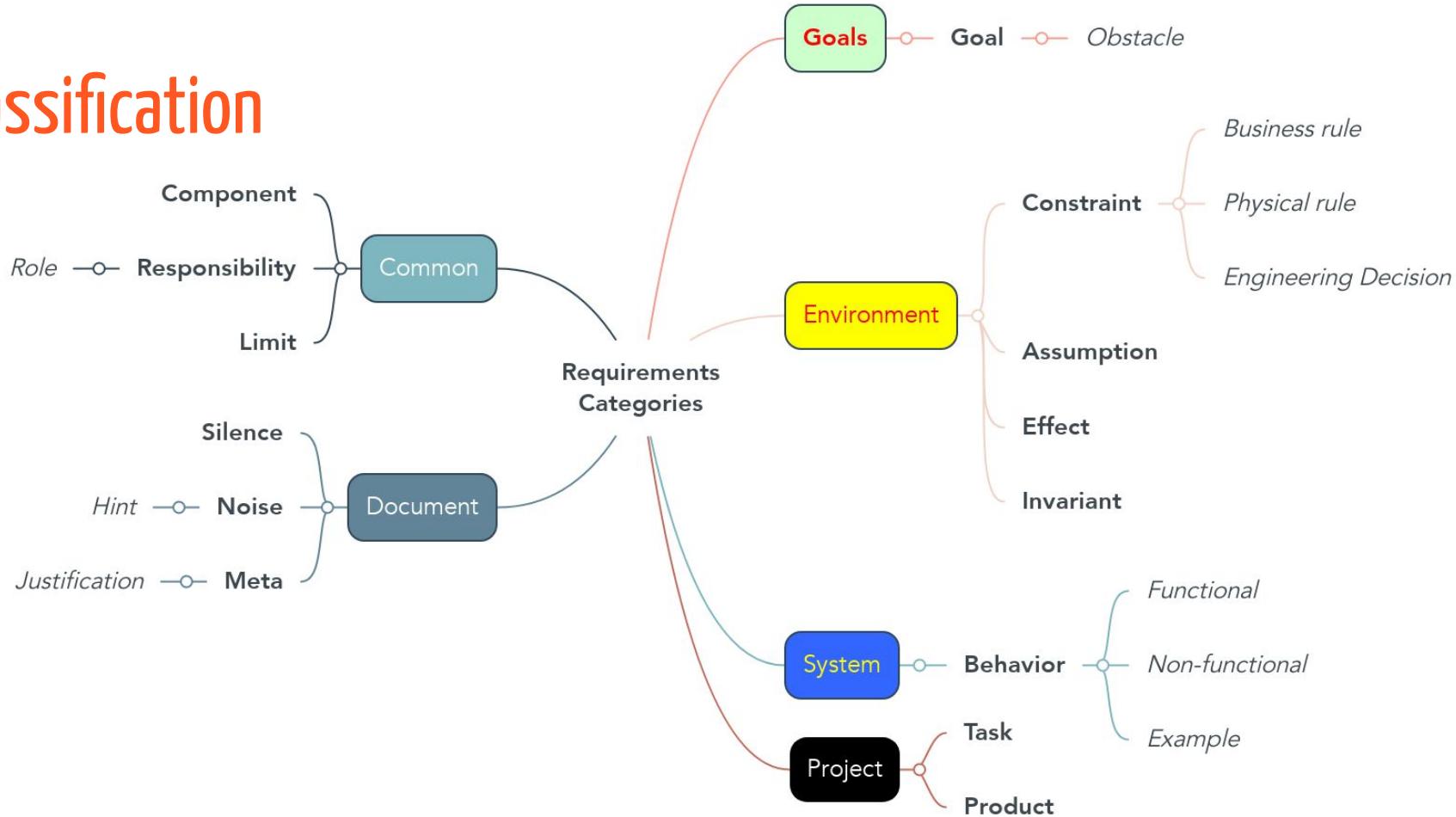
Justification (kind of Meta-requirement)

(Explanation of a project or system property, in reference to a goal or environment property)



“ The presence of two signature fields follows from the rule on purchases higher than € 5000 (section E.3.X).”

Classification



Categories of requirements (derived)

- **Justification** (from Meta)
- **Role** (from Responsibility)
- **Business rule** (from Constraint)
- **Physical rule** (from Constraint)
- **Engineering decision** (from Constraint)
- **Hint** (from Noise)
- **Obstacle** (from Goal)
- **Functional** (from Behavior)
- **Non-Functional** (from Behavior)
- **Example** (from Behavior)

Guideline for category identification

1. Which PEGS (this shortens the possibilities)
2. Check if specific (not component/resp/limit or document)
3. Pick the best among the remaining ones



Practice

<https://bit.ly/re23-tutoJMB>

EXERCICE

Practice 3: Collaborative experience

- [Elaastic Platform](#)
- <https://www.irit.fr/elaastic/>
- [Find examples](#)
- Classification



Outline

- Context
- Requirements anatomy
- Requirements tooling

The four PEGS (details)

Standard Plan



Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview
- G.5 High-level usage scenarios
- G.6 Limitations and exclusions
- G.7 Stakeholders and requirements sources

Environment (E)

- E.1 Glossary
- E.2 Components
- E.3 Constraints
- E.4 Assumptions
- E.5 Effects
- E.6 Invariants

System (S)

- S.1 Components
- S.2 Functionality
- S.3 Interfaces
- S.4 Detailed usage scenarios
- S.5 Prioritization
- S.6 Verification and acceptance criteria

Goals

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview
- G.5 High-level usage scenarios
- G.6 Limitations and exclusions
- G.7 Stakeholders and requirements sources

Goals Book

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview
- G.5 High-level usage scenarios
- G.6 Limitations and exclusions
- G.7 Stakeholders and requirements sources

Describes the **needs** of the target organization,
which the system to be developed **will address**

G.1 Context and overall objectives

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview
- G.5 High-level usage scenarios
- G.6 Limitations and exclusions
- G.7 Stakeholders and requirements sources

High-level view of the project: organizational context and reason for building a system

G.2 Current situation

Current state of processes to be addressed by the project and the resulting system

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview
- G.5 High-level usage scenarios
- G.6 Limitations and exclusions
- G.7 Stakeholders and requirements sources

NOT EMPTY!

G.3 Expected benefits

New processes, or improvement to existing processes, made possible by the project's results

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview
- G.5 High-level usage scenarios
- G.6 Limitations and exclusions
- G.7 Stakeholders and requirements sources

G4. Functionality overview

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview**
- G.5 High-level usage scenarios
- G.6 Limitations and exclusions
- G.7 Stakeholders and requirements sources

Overview of the **functions** (behavior) of the system

Principal **properties** only (*details are in the System book*)

G.5 High-level usage scenarios

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview
- G.5 High-level usage scenarios**
- G.6 Limitations and exclusions
- G.7 Stakeholders and requirements sources

Fundamental usage paths through the system

G.6 Limitations and exclusions

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview
- G.5 High-level usage scenarios
- G.6 Limitations and exclusions**
- G.7 Stakeholders and requirements sources

Aspects that the system need **not address**

G.7 Stakeholders and requirements sources

Goals (G)

- G.1 Context and overall objective
- G.2 Current situation
- G.3 Expected benefits
- G.4 Functionality overview
- G.5 High-level usage scenarios
- G.6 Limitations and exclusions
- G.7 Stakeholders and requirements sources

Groups of **people** who can affect the project or be affected by it, and other places to consider for **information about** the project and system

Environment

Environment (E)

- E.1 Glossary
- E.2 Components
- E.3 Constraints
- E.4 Assumptions
- E.5 Effects
- E.6 Invariants

Environment Book

Environment (E)

- E.1 Glossary
- E.2 Components
- E.3 Constraints
- E.4 Assumptions
- E.5 Effects
- E.6 Invariants

Describes the application **domain** and **external context**, physical or virtual (or a mix), in which the system will operate

 NOT EMPTY!

E.1 Glossary

Environment (E)

- E.1 Glossary
- E.2 Components
- E.3 Constraints
- E.4 Assumptions
- E.5 Effects
- E.6 Invariants

Clear and precise **definitions** of all the **vocabulary** **specific** to the application domain, including technical terms, words from ordinary language used in a special meaning, and acronyms

E.2 Components

Environment (E)

- E.1 Glossary
- E.2 Components
- E.3 Constraints
- E.4 Assumptions
- E.5 Effects
- E.6 Invariants

List of elements of the environment that **may affect or be affected by the system** and project

Includes other systems to which the system must be interfaced

NOT EMPTY!

E.3 Constraints

Obligations and limits imposed on the project and system by the environment

Environment (E)

- E.1 Glossary
- E.2 Components
- E.3 Constraints**
- E.4 Assumptions
- E.5 Effects
- E.6 Invariants

E.4 Assumptions

Environment (E)

- E.1 Glossary
- E.2 Components
- E.3 Constraints
- E.4 Assumptions**
- E.5 Effects
- E.6 Invariants

Properties of the environment that may be **assumed**, with the goal of facilitating the project and simplifying the system

E.5 Effects

Environment (E)

- E.1 Glossary
- E.2 Components
- E.3 Constraints
- E.4 Assumptions
- E.5 Effects**
- E.6 Invariants

Elements and properties of the environment that
the system will affect

E.6 Invariants

Environment (E)

- E.1 Glossary
- E.2 Components
- E.3 Constraints
- E.4 Assumptions
- E.5 Effects
- E.6 Invariants

Properties of the environment that the system's operation **must preserve**

System

System (S)

S.1 Components

S.2 Functionality

S.3 Interfaces

S.4 Detailed usage scenarios

S.5 Prioritization

S.6 Verification and acceptance criteria

System Book

System (S)

- S.1 Components
- S.2 Functionality
- S.3 Interfaces
- S.4 Detailed usage scenarios
- S.5 Prioritization
- S.6 Verification and acceptance criteria

Refines the Goal one by focusing on more detailed requirements about the system under development, mainly its **constituents, behaviors and properties**

NOT EMPTY!

S.1 Components

System (S)

- S.1 Components
- S.2 Functionality
- S.3 Interfaces
- S.4 Detailed usage scenarios
- S.5 Prioritization
- S.6 Verification and acceptance criteria

Overall **structure** expressed by the list of major software and, if applicable, hardware **parts**

 NOT EMPTY!

S.2 Functionality

System (S)

- S.1 Components
- S.2 Functionality
- S.3 Interfaces
- S.4 Detailed usage scenarios
- S.5 Prioritization
- S.6 Verification and acceptance criteria

One section, S.2.n, for each of the components identified in S.2, describing the corresponding **behaviors** (functional and non-functional properties)

S.3 Interfaces

System (S)

- S.1 Components
- S.2 Functionality
- S.3 Interfaces**
- S.4 Detailed usage scenarios
- S.5 Prioritization
- S.6 Verification and acceptance criteria

How the system makes the functionality of S.2 available to the rest of the world, particularly user interfaces and program interfaces (**APIs**)

S.4 Detailed usage scenarios

System (S)

- S.1 Components
- S.2 Functionality
- S.3 Interfaces
- S.4 Detailed usage scenarios**
- S.5 Prioritization
- S.6 Verification and acceptance criteria

**Examples of interaction between the environment
(or human users) and the system: use cases and/or
user stories**

S.5 Prioritization

System (S)

- S.1 Components
- S.2 Functionality
- S.3 Interfaces
- S.4 Detailed usage scenarios
- S.5 Prioritization
- S.6 Verification and acceptance criteria

Classification of the behaviors, interfaces and scenarios (S.2, S.3 and S.4) by their degree of criticality

S.6 Verification and acceptance criteria

System (S)

- S.1 Components
- S.2 Functionality
- S.3 Interfaces
- S.4 Detailed usage scenarios
- S.5 Prioritization
- S.6 Verification and acceptance criteria

Specification of the **conditions** under which an implementation will be deemed **satisfactory**

Project

Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

Project Book

Describes all the constraints and expectations **not about the system itself, but about how to develop and produce it**

Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

P.1 Roles and personnel

Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

Main **responsibilities** in the project; required project staff and their needed qualifications

P.2 Imposed technical choices

Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

Any **a priori** choices **binding** the project to specific tools, hardware, languages or other technical parameters

NOT EMPTY!

P.3 Schedule and milestones

Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

List of tasks to be carried out and their scheduling

 NOT EMPTY!

P.4 Tasks and deliverables

Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

Details of individual tasks listed under P.3 and their expected outcomes

P.5 Required technology elements

Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

External systems, hardware and software, expected to be necessary for building the system

P.6 Risk and mitigation analysis

Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

Potential **obstacles** to meeting the schedule of P.4, and measures for **adapting the plan** if they do arise

P.7 Requirements process and report

Project (P)

- P.1 Roles and personnel
- P.2 Imposed technical choices
- P.3 Schedule and milestones
- P.4 Tasks and deliverables
- P.5 Required technology elements
- P.6 Risk and mitigation analysis
- P.7 Requirements process and report

Initially, description of what the requirements process will be; later, report on its steps

Terms often found in RE

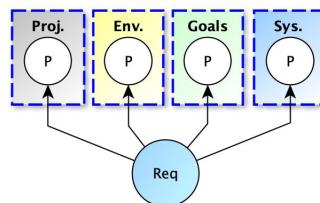
- Actor (=> Role or Responsibility)
- See brainstorming
- Obligation (=> Constraint)



Practice

Practice 4: Find the corresponding chapter

1. Some of the general constraints were defined in the preliminary meeting of 15 June 2022, available at [URL].
2. The login record shall be implemented using MongoDB.
3. Here is the basic scheme of interaction for ordering a product: [followed by the description of that scheme].
4. The project shall only use external software products available through an approved open-source license (GPL or Creative Commons).
5. The product shall be available on mobile platforms as well as through an API.
6. Any use of cookies shall conform to the GDPR.
7. As a result of the introduction of the new payroll system, pay periods shall be standardized to monthly for all employees.
8. As the system depends on Windows 11 facilities, meeting the schedule depends on Microsoft fully releasing Windows 11 by end of October, 2021.
9. This function is considered critical to the deployment of the project.
10. Upon exiting a session, the system shall memorize the last explored directory as the restart point for the next session.



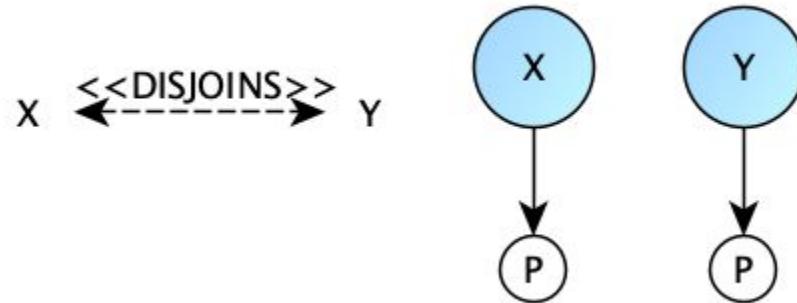
Categories of inter-requirements relations

Relations between requirements

- **Disjoins** ($x \parallel y$)
- **Belongs** ($x \subseteq y$)
- **Repeats** ($x \Leftrightarrow y$)
- **Contradicts** ($x \oplus y$)
- **Extends** ($x > y$)
- **Excepts** ($x \setminus\setminus y$)
- **Constrains** ($x \triangleright y$)
- **Characterizes** ($x \rightarrow y$)

X || Y

X and Y are unrelated



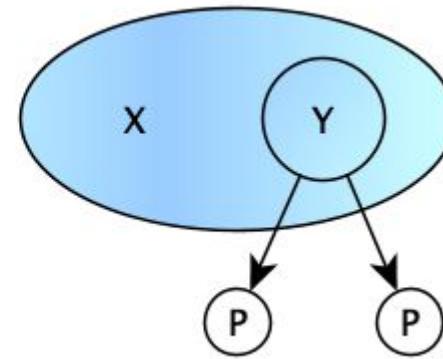
“The system is composed of three components.”

“The car should be as economic in fuel consumption as possible.”

$Y \subseteq X$

Y is a sub-requirement of X

Y $\xrightarrow{<<\text{BELONGS}>>}$ X

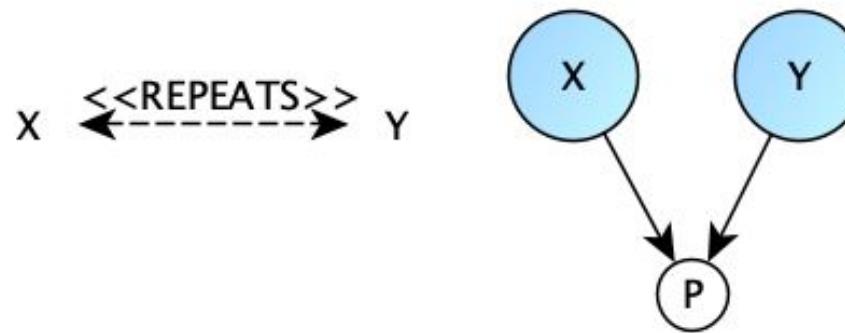


“4.3 System Externals”

“A customer is any user of the system that has not identified himself as an SBE employee.”

$X \leftrightarrow Y$

X specifies the same property as Y



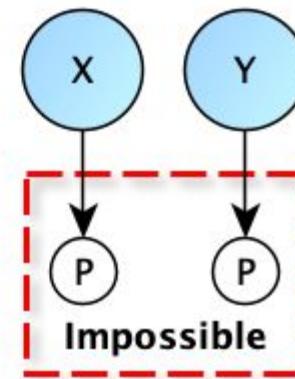
“The system is composed of three components.”

“Here are the descriptions of the three parts of the system:”

$X \oplus Y$

X specifies a property in a way not compatible with Y

X <<CONTRADICTS>> Y

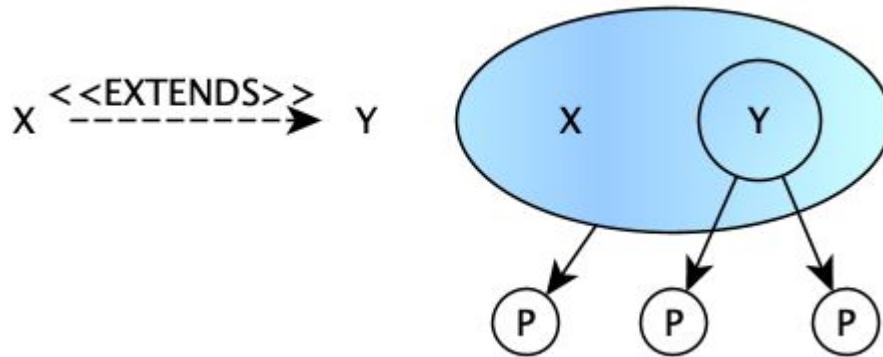


“The system has no interaction
with human.”

“The user should login
interactively with the system.”

X > Y

X assumes Y and specifies a property not specified by Y

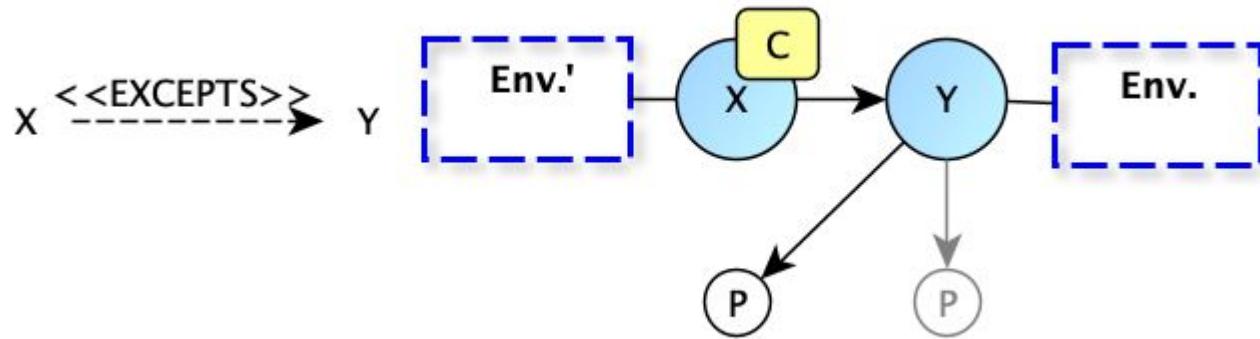


“The online product ordering should allow direct access to the confirmation page.”

“The system shall allow for online product ordering by either the customer or the sales agent.”

X \ Y

X changes or removes, for a specified case,
a property specified by Y

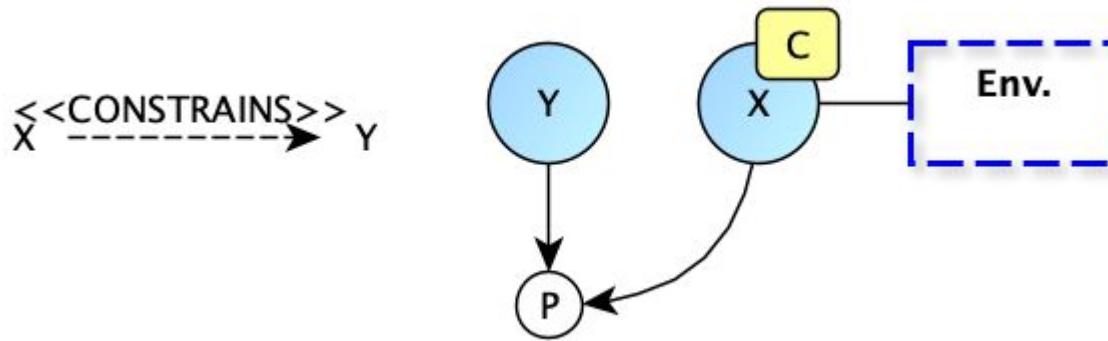


“In case of emergency braking,
the system should prevent the
wheels from freezing.”

“The wheel can be frozen by
braking.”

$X \triangleright Y$

X specifies a constraint on a property specified by X

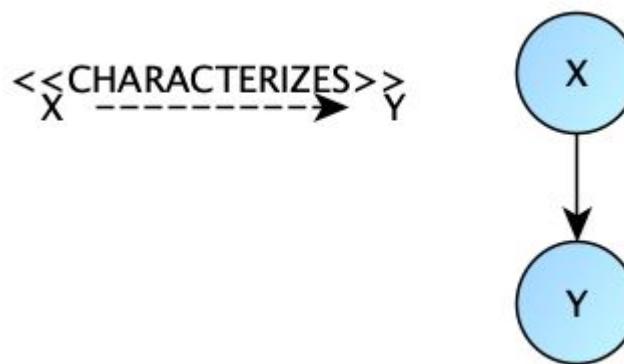


“The user is registered.”

“In order to get personalized or restricted information, place orders or do other specialized transactions a user must login so that the system can determine his access level.”

$X \rightarrow Y$

X is a meta-requirement involving Y



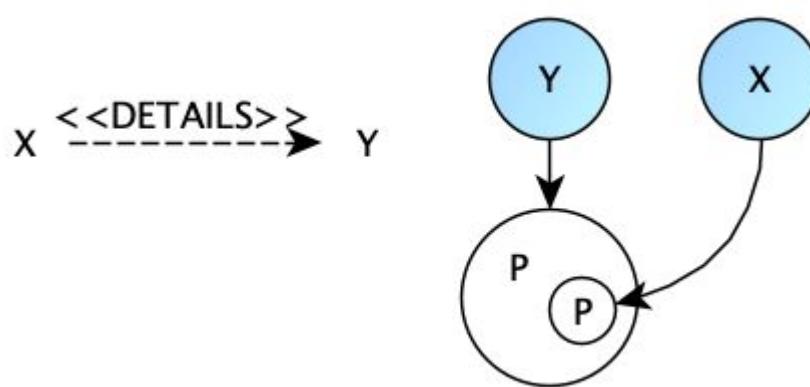
"The following requirement is optional:"

"The car should looks like a Ferrari."

Derived (but useful) relations

X » Y

X adds detail to a property specified by Y

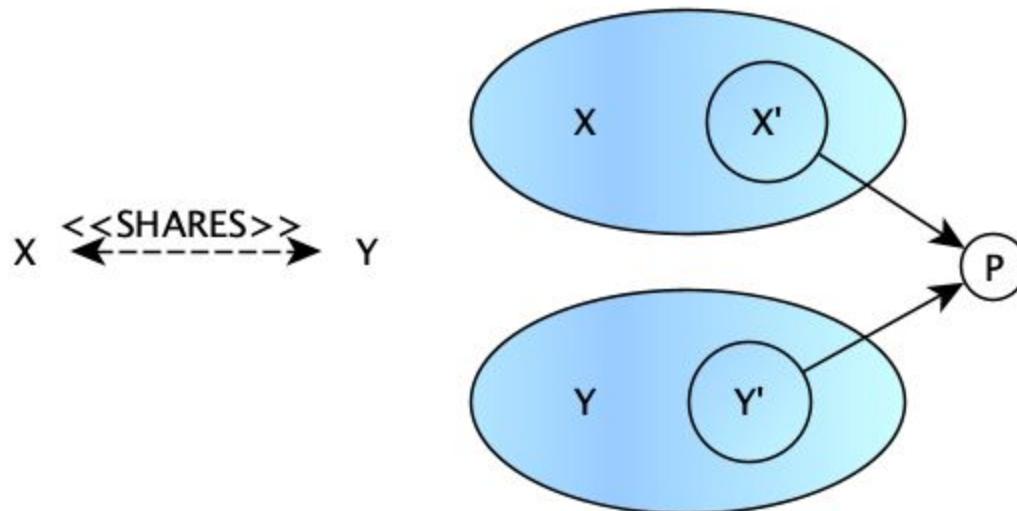


“The hot water should be
between 27°C and 37°C.”

“The shower should deliver hot
water.”

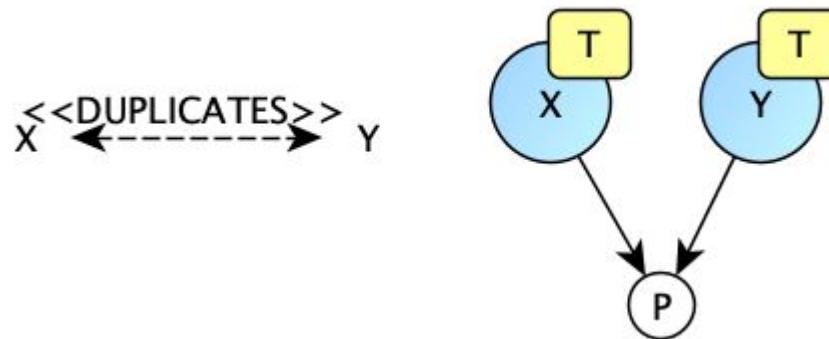
$X \cap Y$

$X' \Leftrightarrow Y'$ for some sub-requirements X' and Y' of X and Y



$X \equiv Y$

$X \Leftrightarrow Y$, and X has the same type as Y

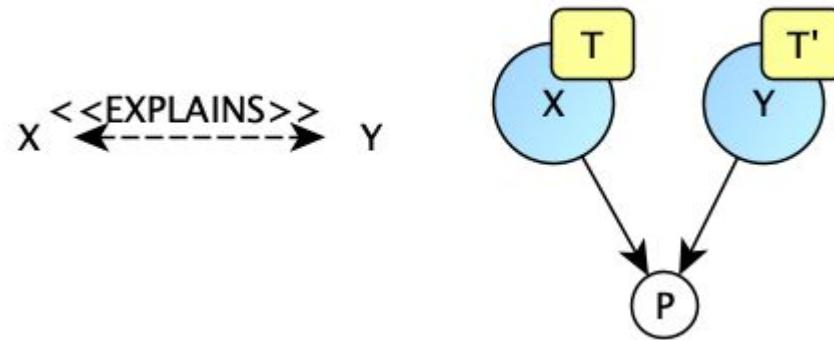


“The system is composed of
three components.”

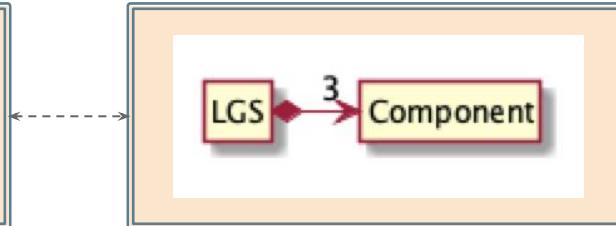
“Here are the descriptions of
the three parts of the system:”

$X \cong Y$

$X \Leftrightarrow Y$, and X has a different type from Y

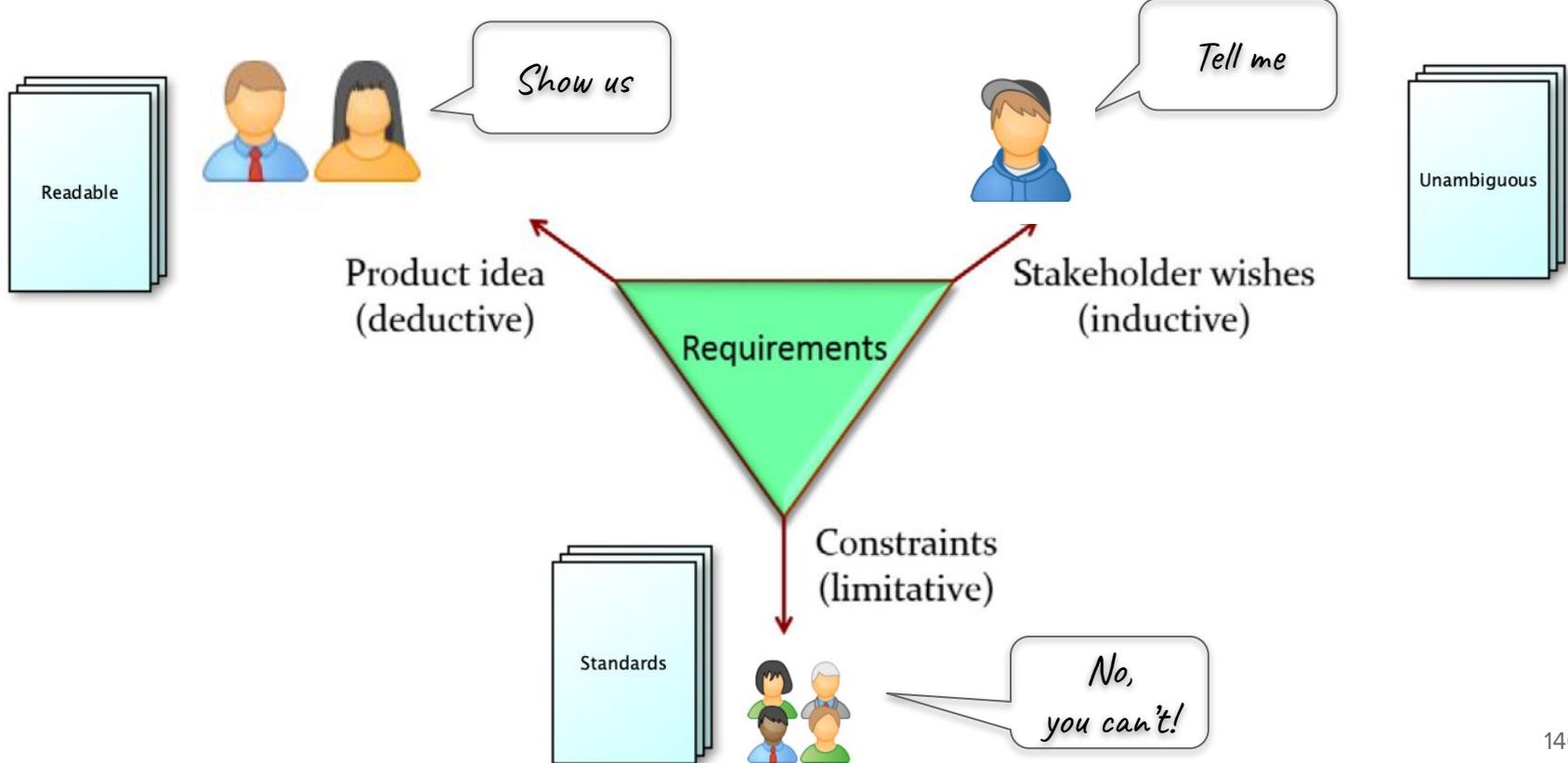


“The LGS has three components.”



Quality Assessment

Quality criteria for requirements



Quality criteria for requirements

Quality criteria for requirements			
Attribute	Applies to	Attribute	Applies to
Correct (4.1)	<i>Environment, System.</i>	Traceable (4.8)	<i>all</i>
Justified (4.2)	<i>Project, System</i>	Delimited (4.9)	<i>all</i>
Complete (4.3)	<i>all</i>	Readable (4.10)	<i>all</i>
Consistent (4.4)	<i>all</i>	Modifiable (4.11)	<i>all</i>
Unambiguous (4.5)	<i>all</i>	Verifiable (4.12)	<i>Project, System</i>
Feasible (4.6)	<i>Project, System</i>	Prioritized (4.13)	<i>system</i>
Abstract (4.7)	<i>System</i>	Endorsed (4.14)	<i>all</i>

Correctness

An Environment or System requirement is correct if it is compatible with actual project parameters, properties of the environment, organizational goals, and stakeholder expectations.

APPLIES TO:
PROJ & SYS

Justifiability

A Project or System requirement is justified if it helps reach a goal or satisfy a constraint.



APPLIES TO:
ALL

Completeness

A set of requirements is complete, or not, along six criteria: document, goal, scenario, environment, interface and command-query completeness.



APPLIES TO:
ALL

Consistency

A set of requirements is consistent if it contains no contradiction.



APPLIES TO:
ALL

Non-ambiguity

A set of requirements is unambiguous if none of its elements is so expressed as to lend itself to two significantly different understandings.

Feasibility

A **System** (resp. Project) requirement is feasible if it is **possible**, within the constraints of the Environment and Goals, to produce an **implementation** (resp. schedule) that satisfies it.

Abstractness

A **System** requirement is abstract if it specifies a desired system property **without prescribing or favoring specific design or implementation choices.**



APPLIES TO:
ALL

Traceability

A Goals, System, Project or Environment requirement is traceable if it is possible to follow its consequences, both ways, in other project artifacts including design, implementation and verification elements.

Delimitedness

A set of **Goals** or **System** requirements is delimited if it specifies the **scope** of the future system, making it possible to determine what functionality lies beyond that scope.

APPLIES TO:
ALL

Readability

A requirement is readable if it can be **readily understood** by its intended audience.



APPLIES TO:
ALL

Modifiability

A set of requirements is modifiable if it can be adapted in case of **changes** to Project, Environment, Goals or System properties, through an effort commensurate with the extent of the changes.

Verifiability

A **System** (resp. Project) requirement is verifiable if it is expressed in such a way as to allow determining whether a proposed implementation (resp. the sequence of events in the actual project) satisfies it.

Prioritization

A set of **System** requirements is prioritized if it includes for each of them a **specification of its importance** relative to the others, making it possible to make informed decisions if events in the course of the project make it necessary to renounce some functionality.



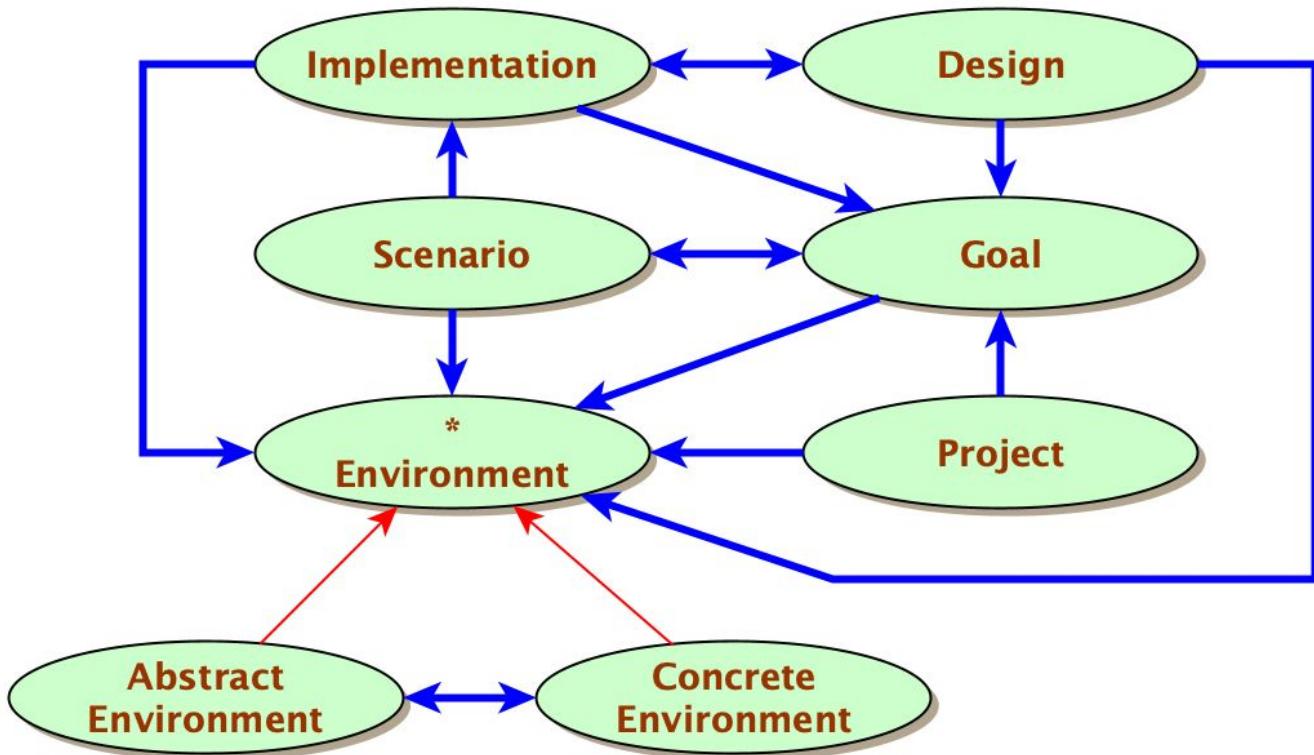
APPLIES TO:
ALL

Endorsement

A requirement is endorsed if it has been **approved** by all the relevant decision-makers.

What are the benefits ?

Seven kinds of classes



Examples of possible prescriptions

No **Duplicates**

Few **Excepts**

Discussions and choices made **explicit**

...

Contributions

Clarification of reqs concepts

Basic for reqs methodology

Basics for critical analysis of reqs docs

Basis for NLP

...

Enough concepts,
let's get practical

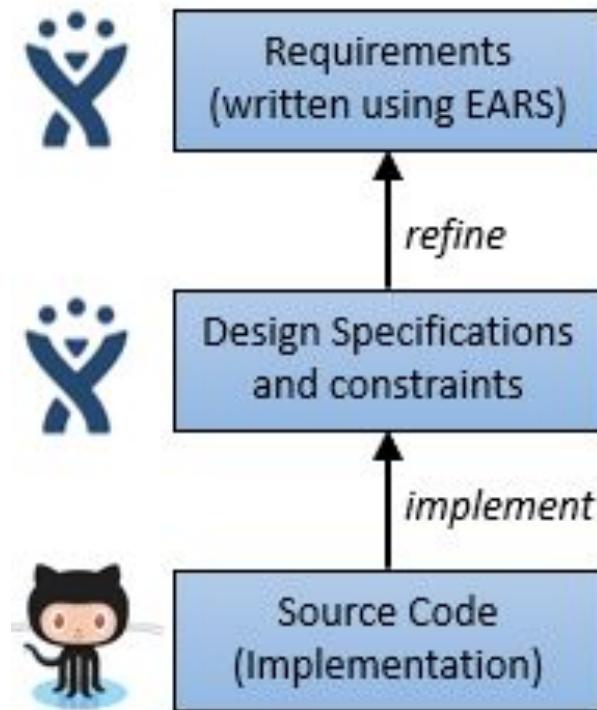
Modern versions

- Dronology: a traceability masterpiece (<https://dronology.info/>)
- Companion material for an upcoming book... (<https://requirements.university>)

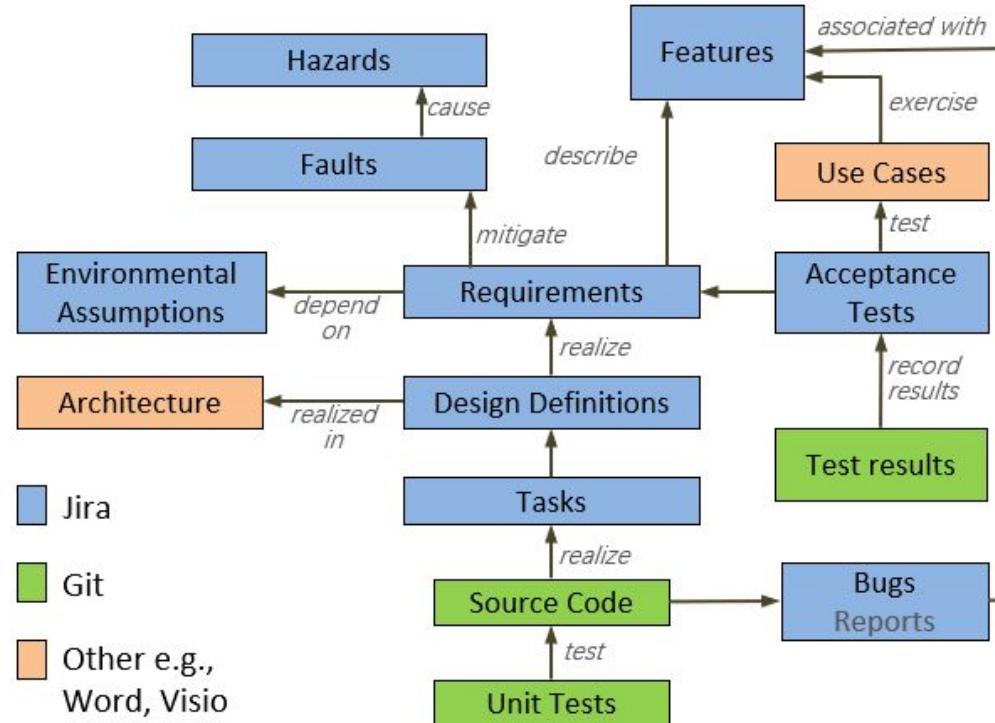
Dronology



Focus on traceability



Traceability



Useful requirements document

Total Entries:	398
Components:	25
Open:	23
Closed:	2
Requirements:	99
Open:	32
Closed:	67
Design Definitions:	211
Open:	52
Closed:	159
Sub-Tasks:	63
Open:	0
Closed:	63
Links to Code:	892
Manual created Links:	338
Committed Links:	554

CO-90 -- GCS Middleware	[Component]
Status: Open	
Description:	
Handles connections between Dronology and Ground Control Stations (GCS). Forwards commands monitoring and other messages from Dronology to its registered GCS and passes messages describing the state of the UAVs managed by each GCS back to dronology.	
Contained Elements:	DD-354 - DD-361 - DD-710 - DD-711 - DD-712 - DD-713 - DD-715 - DD-716 - DD-718 - DD-719 - DD-720 - DD-721 - DD-723 - DD-724 - DD-727 - DD-728 - DD-730 - DD-731 - DD-732 - DD-733 - DD-734 - DD-735 - DD-737 - DD-763 - DD-768 - RE-160 - RE-709 - RE-714 - RE-722 - RE-729 - RE-736
CO-91 -- GCS	[Component]
Status: Open	
Description:	
Python based system that manages and controls UAVs. Communicates with Dronology via the Ground Station middleware. Each GCS is responsible for communicating directly with each UAV sending it commands and monitoring its state including its current position flight mode and health.	
Contained Elements:	DD-740 - DD-742 - DD-743 - DD-744 - DD-745 - DD-747 - DD-748 - DD-749 - DD-750 - DD-752 - DD-753 - DD-755 - DD-756 - DD-757 - RE-235 - RE-739 - RE-741 - RE-746 - RE-751 - RE-754
CO-105 -- UI Real-Time Flight View	[Component]
Status: Open	
Description:	
Manages all aspects of displaying flights and UAVs in real-time and interacting with them. The flight view displays active routes UAV coordinates and their current health. The map uses zoom and panning features to follow one or more selected UAV.	
Contained Elements:	DD-113 - DD-121 - DD-229 - DD-682 - DD-683 - DD-684 - DD-685 - DD-686 - DD-687 - DD-688 - DD-690 - DD-692 - DD-694 - DD-696 - DD-697 - DD-699 - RE-114 - RE-120 - RE-681 - RE-689 - RE-691 - RE-693 - RE-695 - RE-698
CO-184 -- Internal Simulator	[Component]
Status: Closed	
Description:	
The internal simulator provides low-fidelity features for supporting quick initial tests of a virtual UAV. Features include takeoff goto land and battery health.	
Contained Elements:	RE-593 - RE-594 - RE-595 - RE-596 - RE-597

Companion material



Templates (docx, LaTeX, Google Doc, ...)

Goals

Goals are "needs of the target organization, which the system will address". While the development team is the principal user of the other books, the Goals book addresses a wider audience: essentially, all stakeholders (see [Handbook](#)).

 It must contain enough information to provide — if read just by itself — a general sketch of the entire project. To this effect, chapter G.3 presents a short overview of the system and [G.1](#) will typically include some key properties of the environment. As it addresses a wide readership, it should be clear and minimize the use of specialized technical terms. Together, [G.1](#), [G.2](#) and [G.3](#) describe the rationale for the project. It is important to state these justifications explicitly. Typically, they are well understood at the start of the project, but management and priorities can change (see [Handbook](#)).

G.1 Context and overall objectives

 High-level view of the project: organizational context and reason for building a system (see [Handbook](#)).

 This section should not be empty (following the *Minimum Requirements Outcome Principle*, p.27 of the [Handbook](#)).

¹ Example of numbered requirement that can be [referenced](#).

G.2 Current situation

 Current state of processes to be addressed by the project and the resulting system (see [Handbook](#)).

1 Goals

Contents

1.1	G.1 Context and overall objective	4
1.2	G.2 Current situation	4
1.3	G.3 Expected benefits	4
1.4	G.4 Functionality overview	5
1.5	G.5 High-level usage scenarios	5
1.6	G.6 Limitations and exclusions	5
1.7	G.7 Stakeholders and requirements sources	5

Comment: Goals are "needs of the target organization, which the system will address". While the development team is the principal user of the other books, the Goals book addresses a wider audience: essentially, all stakeholders.

1.1 G.1 Context and overall objective

Comment: High-level view of the project: organizational context and reason for building a system. This chapter should not be empty!

Goal 1.1.1. This is a goal example. If you need explicit (and automatic) numbering, you can use the definitions in the `.tex` template. Is is refined by [1.2.1](#)

More than Word!

- Markdown-like format
- GitHub itself
- Quality metrics & rules **implemented**

Github repo template

The screenshot shows a GitHub repository interface for a template named "HandBookTemplate". The repository is public and has one branch, "master", with no tags. The commit history shows a single update by user "jmbruel" on October 13, 2021, titled "Update README.adoc". The repository contains several files, all of which are marked as "First draft": ".github", "features", ".gitignore", "Environment.adoc", "Gemfile", "Goals.adoc", "LICENSE", "Makefile", "Project.adoc", "README.adoc", "System.adoc", "changelog.adoc", "config.json", and "definitions.adoc".

File	Status
.github	First draft
features	First draft
.gitignore	First draft
Environment.adoc	First draft
Gemfile	First draft
Goals.adoc	First draft
LICENSE	First draft
Makefile	First draft
Project.adoc	First draft
README.adoc	Update README.adoc
System.adoc	First draft
changelog.adoc	First draft
config.json	First draft
definitions.adoc	First draft

PEGS chapters to organize requirements writing

The screenshot shows a web browser displaying a Zenhub Kanban board for a project titled "ATCO Eats - Requirements Elicitation". The board has four columns representing different stages: "Todo" (22 items), "In Progress" (5 items), "In Review" (3 items), and "Done" (2 items). Below these columns, two milestones are visible: "Milestone #1" (10 items) and "Milestone #2" (11 items). Each item on the board is represented by a card with a user icon, a title, and a description. A search bar and filter options are at the top, and a "Sian in now to use Zenhub" button is at the bottom right.

Column	Count	Description
Todo	22	This item hasn't been started
In Progress	5	This is actively being worked on
In Review	3	Work is done and pending reviewer approval
Done	2	This has been completed

Milestone	Count	Description
Milestone #1	10	(G.7) Stakeholders and requirements sources (E.1) Glossary (P.7) Requirements process and report atco-eats #1 (G.1) Context and Overall Objectives atco-eats #2 (G.2) Current situation atco-eats #4 (G.3) Expected Benefits atco-eats #10 (G.6) Limitations and Exclusions atco-eats #5 (G.4) Functionality overview atco-eats #8 (E.5) Effects atco-eats #9 (E.6) Invariants
Milestone #2	11	(G.5) High-level usage scenarios atco-eats #6

Thanks to Sébastien Mosser for sharing. More at <https://github.com/ace-lectures/atco-eats/>

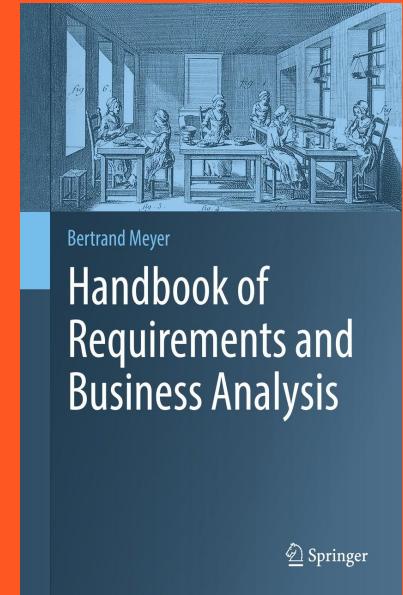
Requirements documents can be tested!

```
#-----  
# language: en  
Feature: Book mutual references  
    The books should follow the mutual references rules.  
  
Scenario: The Environment book must not refer to the Goals and Project books  
    Given The Environment book  
    Then No reference should include the Goals book  
    And No reference should include the Project book  
    And Only E.5 section can refer to the System book  
  
Scenario: The Goals book must not refer to the Project and System books  
    Given The Goals book  
    Then No reference should include the Project book  
    And No reference should include the System book  
  
Scenario: The System book must not refer to the Project book  
    Given The System book  
    Then No reference should include the Project book
```

Requirements documents can be tested!

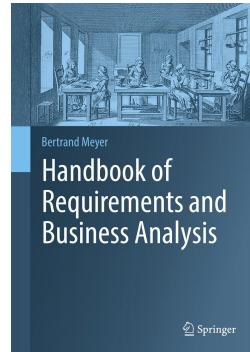
```
4 #-----  
5 # language: en  
6 Feature: Minimum Requirements Outcome Principle  
7 | The requirements effort must always produce the following elements.  
8  
9 Scenario: The Project book must have P3 P4 chapters  
10 | Given The Project book  
11 | Then P3 chapter must not be empty  
12 | And P4 chapter must not be empty  
13  
14 Scenario: The Environment book must have E3 chapter  
15 | Given The Environment book  
16 | Then E3 chapter must not be empty  
17  
18 Scenario: The Goals book must have G1 G3 G7 chapters  
19 | Given The Goals book  
20 | Then G1 chapter must not be empty  
21 | And G3 chapter must not be empty  
22 | And G7 chapter must not be empty  
23  
24 Scenario: The System book must have S1 S2 chapters
```

What's more in the book...



<https://se.inf.ethz.ch/requirements/>

Chapters from the Handbook



- 1. Requirements: Basic concepts and definitions
- 2. Requirements: General principles
- 3. Standard plan for requirements
- 4. Requirements quality and verification
- 5. How to write requirements
- 6. How to gather requirements
- 7. Scenarios: use cases, user stories
- 8. Object-oriented requirements
- 9. Benefiting from formal methods
- 10. Abstract data types
- 11. Are my requirements complete?
- 12. Requirements in the software lifecycle

- Covered
- Overview
- Not covered

Doggy bag

What to remember from all of this?

- Requirements are way more **complex** than simply
“The system shall work.”
- Organizing and classifying requirements helps **Q&A**
- Quality metrics & rules can be **implemented** and hence useful



What's next?



- Feedback (more than) welcome!
- Stay tuned (companion is coming)
- Contribute

<https://requirements.university>



Bertrand Meyer

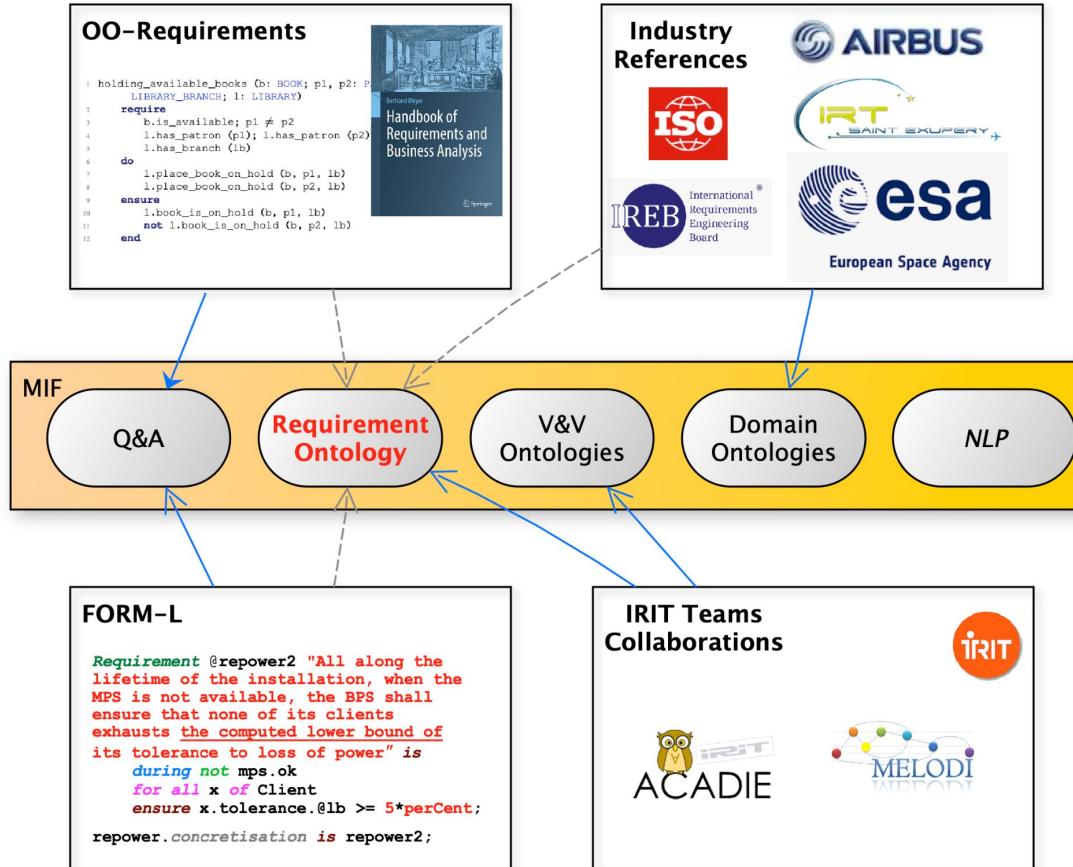
Handbook of
Requirements and
Business Analysis

Springer

<https://se.inf.ethz.ch/requirements/>

One last thing...

We are hiring! (and looking for collaborations)



Concrete positions

- 2 Ph.D. positions on
 - “Requirements Analysis in the Aeronautic Industry: Enhancing Quality and Usability”
 - “V&V in an Industrial DevOps Context”
- Summer Internships (B.Sc/M1) on DLS/MDE

Backup slides

CoCoVaD

Requirements' Early V&V

Airbus Chair

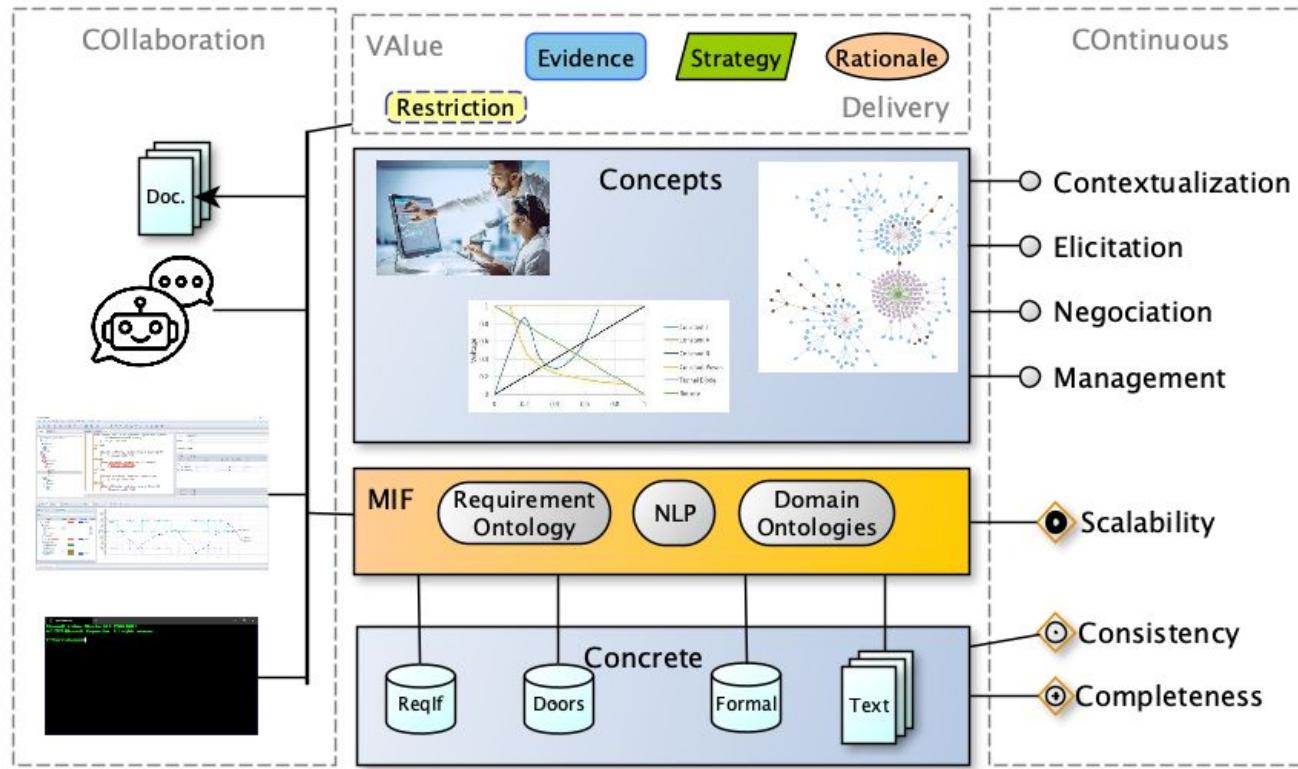


@SmartModelTeam



<https://github.com/smart-researchteam>

Requirements as first-class citizens

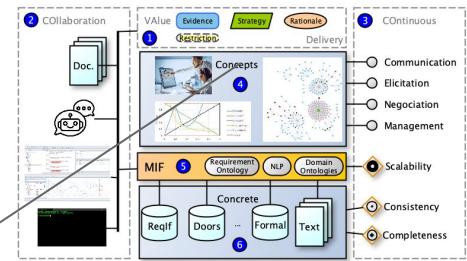


Value Delivery



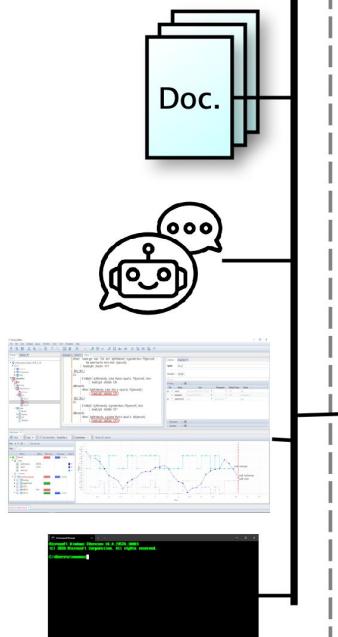
- Who are the clients
- Why do they need this product

Not the primary target
of our research, but crucial



Collaborative effort

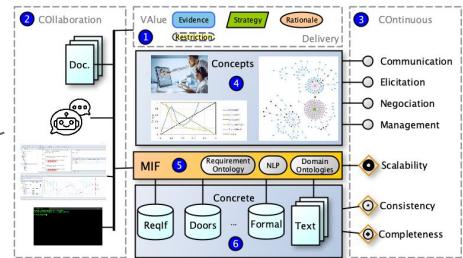
2 Collaboration



- Different Viewpoints
- Different Knowledge
- Different Purposes
- Collaborative effort

Ph.D. #1

- Model Alignment
- DSLs
- Abstraction/Ontologies



Continuous effort

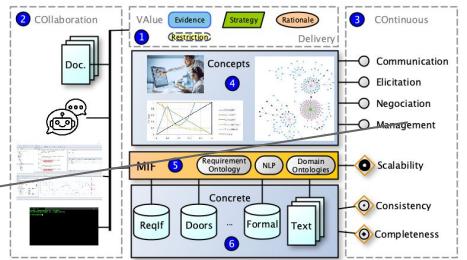
3 COntinuous

- Communication
- Elicitation
- Negotiation
- Management
- ◇ Scalability
- ◇ Consistency
- ◇ Completeness

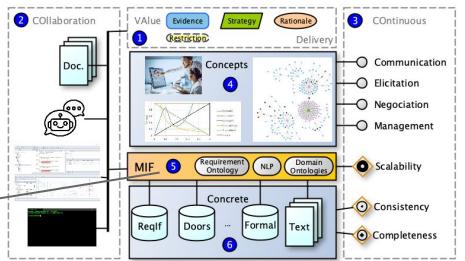
- Check Q&A
- Check Value Delivery
- Allow Collaboration
- Support activities

Ph.D. #2

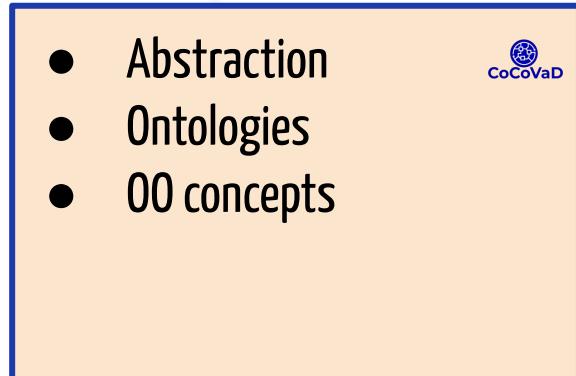
- DevOPS
- Incremental V&V



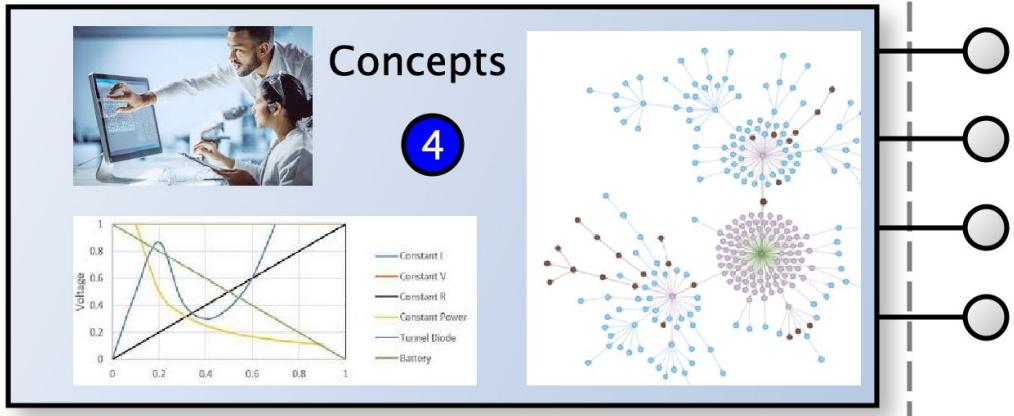
Federation of models



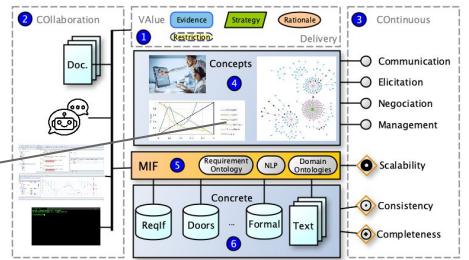
- Model As A Service
- Semantic alignment
- NoSQL
- Model mining



Single source of truth (data lake)



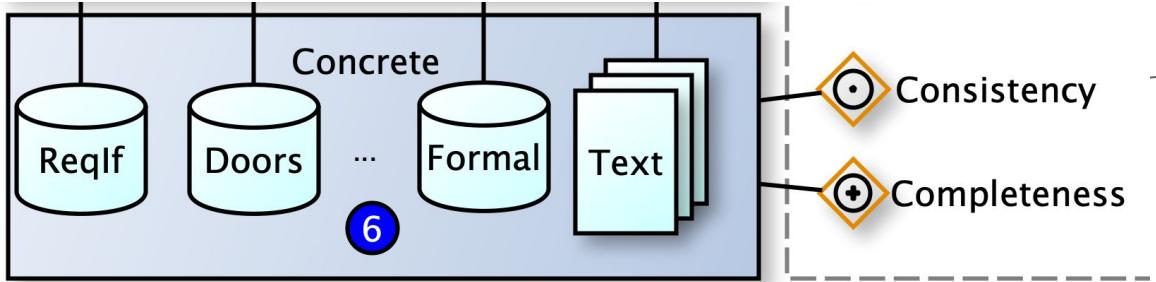
- Model As A Service
- Model & Data collaboration
- Massive data sets



- Abstraction
- Data extraction
- MBSE framework

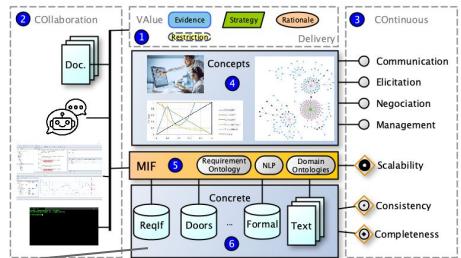


Digitalization



- DDMS
- Zillions of licences and formats

Out of our scope!



Ambiguities

Identifying ambiguities



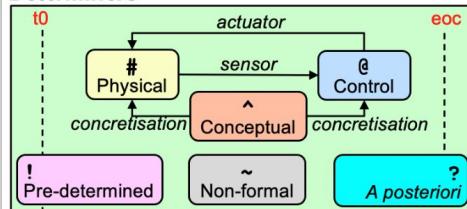


Working on Form-L (CRML)

Formal Requirements Modelling Language

FORM-L in a Nutshell

Determiners



Behavioural Items

Variables (Booleans, statecharts, Integers, Reals, quantities, Strings)
Events
Sets (of items or of values) → 1 st order logic
Properties, Assumptions, Objectives, Requirements, Guarantees, Guards
Objects (static, or dynamic creation / deletion)

Time Domains (in Newtonian time)

One single **Continuous Time Domain** for physical processes & human actions

Multiple Discrete Time Domains
for Globally Asynchronous but Locally Synchronous (GALS) digital systems

Elementary Instructions

When (Temporal locators)	Who / Which (Selectors: universal or existential quantifiers in 1 st order logic)	Where (Spatial locators)
↑↑ time duration		

What, How, How well

Value assignments	A posteriori constraints	Invariance	Achievement
	Operation-time constraints	Invariance constraints	Achievement constraints
Object creation / deletion	Systematic constraints	ensure	achieve
	Capability constraints	assert X, Y can ensure	assert X, Y can achieve

Composite Instructions

Time exclusion	Sequence
Set exclusion	Concurrency
Selection	Iteration
(Deterministic or probabilistic)	

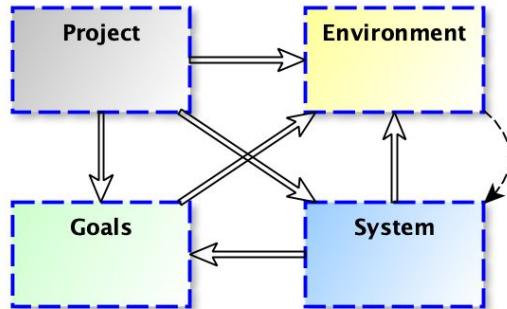
Interfaces (coordination, co-simulation)

Contracts (Engineered, mutually agreed interfaces between concerned parties)	Encroachments (Undesired, forceful failure propagation)	Bindings (enabling co-simulation of FORM-L models developed independently or with non FORM-L models)
Party Deliverables (variables, events, sets, objects) Properties of deliverables		

Cheatsheet

PEGS Cheatsheet

v.2023-09-05



Project (P)	Goals (G)
P.1 Roles and personnel P.2 Imposed technical choices P.3 Schedule and milestones P.4 Tasks and deliverables P.5 Required technology elements P.6 Risk and mitigation analysis P.7 Requirements process and report	G.1 Context and overall objective G.2 Current situation G.3 Expected benefits G.4 Functionality overview G.5 High-level usage scenarios G.6 Limitations and exclusions G.7 Stakeholders and requirements sources
Environment (E)	System (S)
E.1 Glossary E.2 Components E.3 Constraints E.4 Assumptions E.5 Effects E.6 Invariants	S.1 Components S.2 Functionality S.3 Interfaces S.4 Detailed usage scenarios S.5 Prioritization S.6 Verification and acceptance criteria

