Requirements: Scalability Issues

Jean-Michel Bruel

GT IE / INFORSID -- 2024/03/29

https://bit.ly/jmbruel

@SmartModelTeam

https://github.com/smart-researchteam

How to cite:

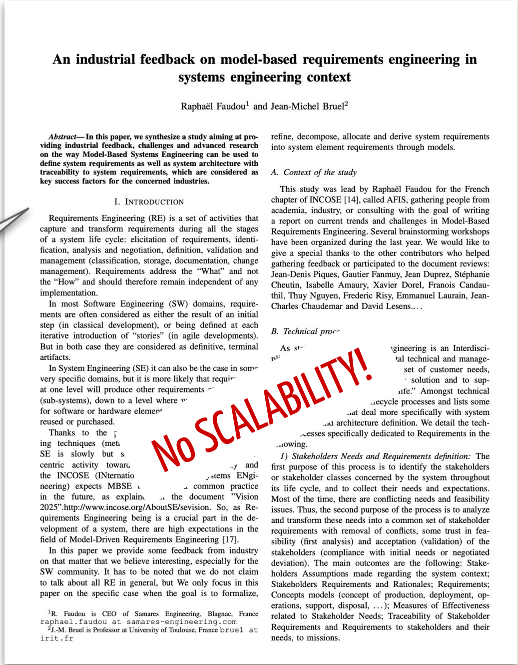
Jean-Michel Bruel, “Journée du GT Exigence en collaboration avec Inforsid”. Toulouse, mar. 2024.

Get my 50+ slides (pdf)

Don't PANIC!

Why me?

- Professor at Toulouse University
- Member of the CNRS-IRIT Laboratory
  - Model-Based Systems Engineering
- Airbus MBSE Chair of Toulouse



An industrial feedback on model-based requirements engineering in systems engineering context

Raphaël Faudou[1] and Jean-Michel Bruel[2]

Abstract— **In this paper, we synthesize a study aiming at providing industrial feedback, challenges and advanced research on the way Model-Based Systems Engineering can be used to define system requirements as well as system architecture with traceability to system requirements, which are considered as key success factors for the concerned industries.**

I. Introduction

Requirements Engineering (RE) is a set of activities that capture and transform requirements during all the stages of a system life cycle: elicitation of requirements, identification, analysis and negotiation, definition, validation and management (classification, storage, documentation, change management). Requirements address the "What" and not the "How" and should therefore remain independent of any implementation.

In most Software Engineering (SW) domains, requirements are often considered as either the result of an initial step (in classical development), or being defined at each iterative introduction of "stories" (in agile developments). But in both case they are considered as definitive, terminal artifacts.

In System Engineering (SE) it can also be the case in some very specific domains, but it is more likely that requir… at one level will produce other requirements … (sub-systems), down to a level where … for software or hardware element… reused or purchased.

Thanks to the … ing techniques (meth… SE is slowly but s… centric activity towar… …y and the INCOSE (INternatio… …ystems ENgineering) expects MBSE … common practice in the future, as explain… the document "Vision 2025".http://www.incose.org/AboutSE/sevision. So, as Requirements Engineering being is a crucial part in the development of a system, there are high expectations in the field of Model-Driven Requirements Engineering [17].

In this paper we provide some feedback from industry on that matter that we believe interesting, especially for the SW community. It has to be noted that we do not claim to talk about all RE in general, but We only focus in this paper on the specific case when the goal is to formalize, refine, decompose, allocate and derive system requirements into system element requirements through models.

A. Context of the study

This study was lead by Raphaël Faudou for the French chapter of INCOSE [14], called AFIS, gathering people from academia, industry, or consulting with the goal of writing a report on current trends and challenges in Model-Based Requirements Engineering. Several brainstorming workshops have been organized during the last year. We would like to give a special thanks to the other contributors who helped gathering feedback or participated to the document reviews: Jean-Denis Piques, Gautier Fanmuy, Jean Duprez, Stéphanie Cheutin, Isabelle Amaury, Xavier Dorel, Franois Candauthil, Thuy Nguyen, Frederic Risy, Emmanuel Laurain, Jean-Charles Chaudemar and David Lesens…

B. Technical proc…

As st… …gineering is an Interdisci… …tal technical and manage… set of customer needs, … solution and to sup… …ife." Amongst technical …ecycle processes and lists some …at deal more specifically with system …d architecture definition. We detail the tech… …cesses specifically dedicated to Requirements in the …llowing.

1) Stakeholders Needs and Requirements definition: The first purpose of this process is to identify the stakeholders or stakeholder classes concerned by the system throughout its life cycle, and to collect their needs and expectations. Most of the time, there are conflicting needs and feasibility issues. Thus, the second purpose of the process is to analyze and transform these needs into a common set of stakeholder requirements with removal of conflicts, some trust in feasibility (first analysis) and acceptation (validation) of the stakeholders (compliance with initial needs or negotiated deviation). The main outcomes are the following: Stakeholders Assumptions made regarding the system context; Stakeholders Requirements and Rationales; Requirements; Concepts models (concept of production, deployment, operations, support, disposal, …); Measures of Effectiveness related to Stakeholder Needs; Traceability of Stakeholder Requirements and Requirements to stakeholders and their needs, to missions.

[1]R. Faudou is CEO of Samares Engineering, Blagnac, France raphael.faudou at samares-engineering.com
[2]J.-M. Bruel is Professor at University of Toulouse, France bruel at irit.fr

https://doi.org/10.1109/REW.2016.042

R. Faudou and J. -M. Bruel, "An Industrial Feedback on Model-Based Requirements Engineering in Systems Engineering Context," **2016** IEEE 24th International Requirements Engineering Conference Workshops (REW), Beijing, China, 2016, pp. 190-199.

Disclaimer (and assumptions)

Scalability = Large amount of (not so) organized data

"MBSE" in this talk = **Model-Based** Software Intensive Systems Engineering

... I am **NOT** an expert in data

... I have more **questions** than answers!

Claim of this talk:

Dealing with RE **without Scalability** in mind

...can **ruin your efforts**

Outline

1. Concrete examples
2. Context: the “CoCoVaD Airbus chair”
3. Why is there a concern?
4. Requirements for requirements’ scalability
5. Conclusion

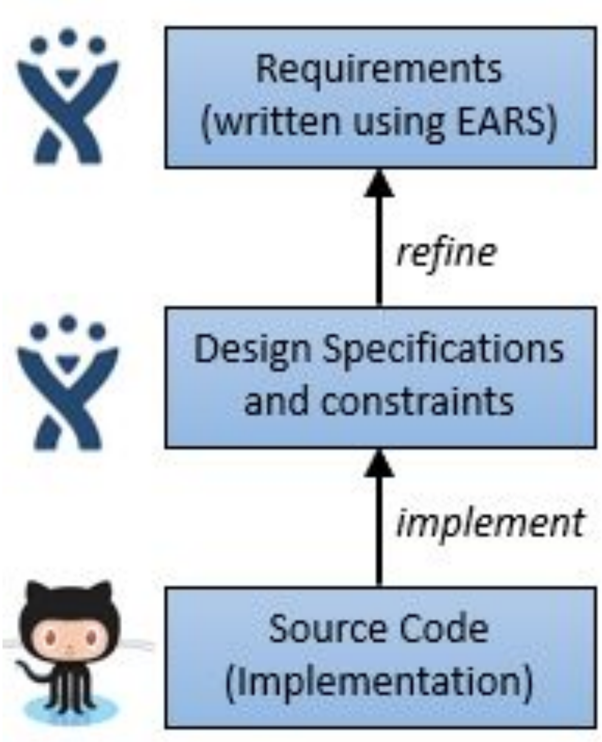
Outline

1. Concrete examples
2. Context: the “CoCoVaD Airbus chair”
3. Why is there a concern?
4. Requirements for requirements’ scalability
5. Conclusion

Dronology

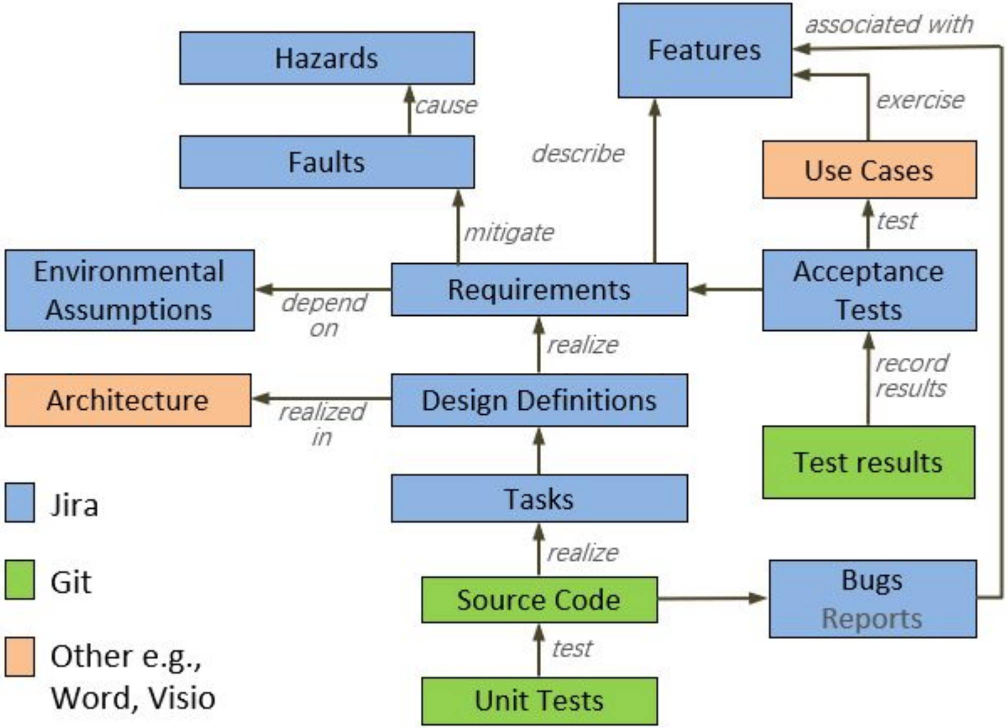
https://dronology.info/

Focus on traceability



https://dronology.info/

Traceability



https://dronology.info/

Useful requirements document

https://dronology.info/

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

(big) Companies data in real-life

1 program (variability on composite choice)
=> 20 To
=> 80% for engineering only!

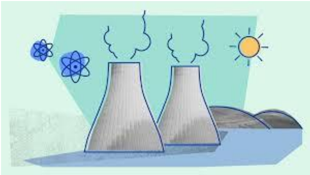
Flight test 380 (4 engines)
=> 4 Go/h (per engine)

Data lake
=> ~ 10 Po

E-commerce study
=> 250 To
=> 10 K€/month in storage only!
=> down to 250 Go by scalability tricks

Requirements in real-life

≈ 10^1 at Aircraft level
≈ 10^3 at Functional level
≈ 10^5 at ATA (system) level



EPR

=> only regulation requirements
=> no explicit hypothesis
=> only design documents

Outline

1. Concrete examples
2. **Context**: the "CoCoVaD Airbus chair"
3. Why is there a concern?
4. Requirements for requirements' scalability
5. Conclusion

Context

Joint effort...

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

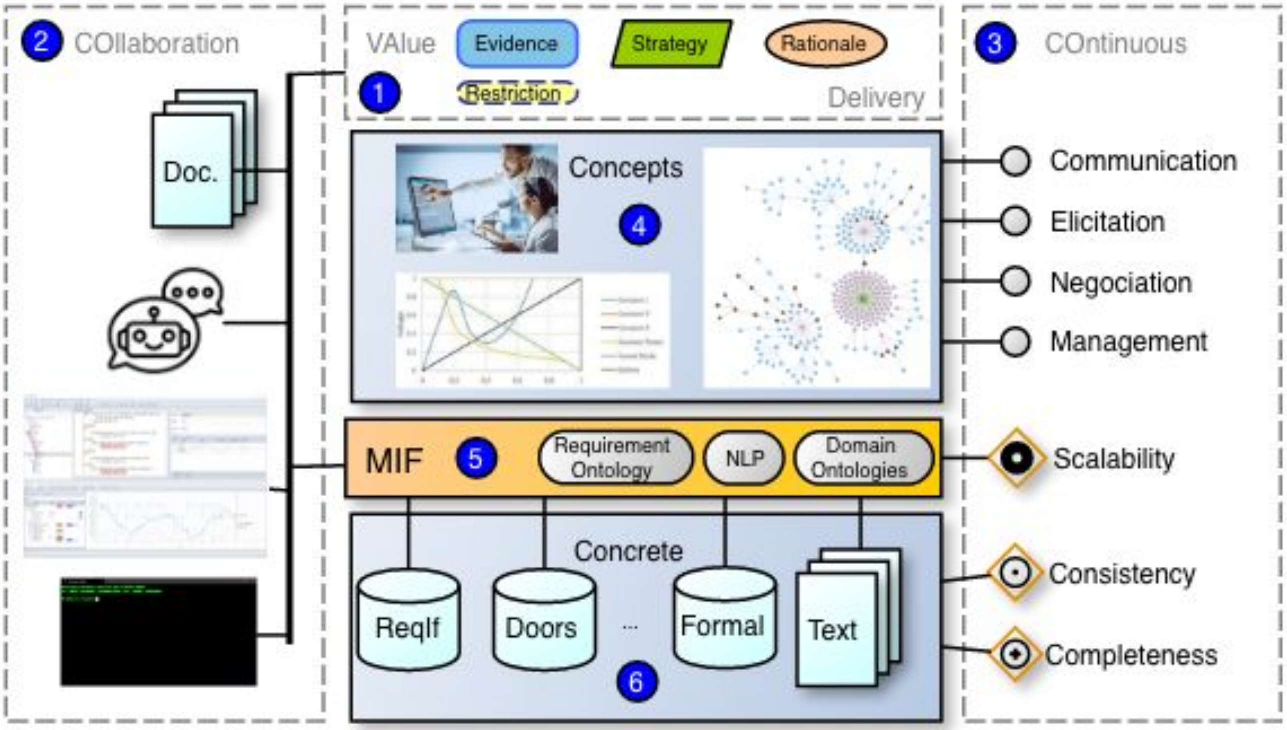
IEEE/SWEBOK/ISO definition of a Requirement

“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

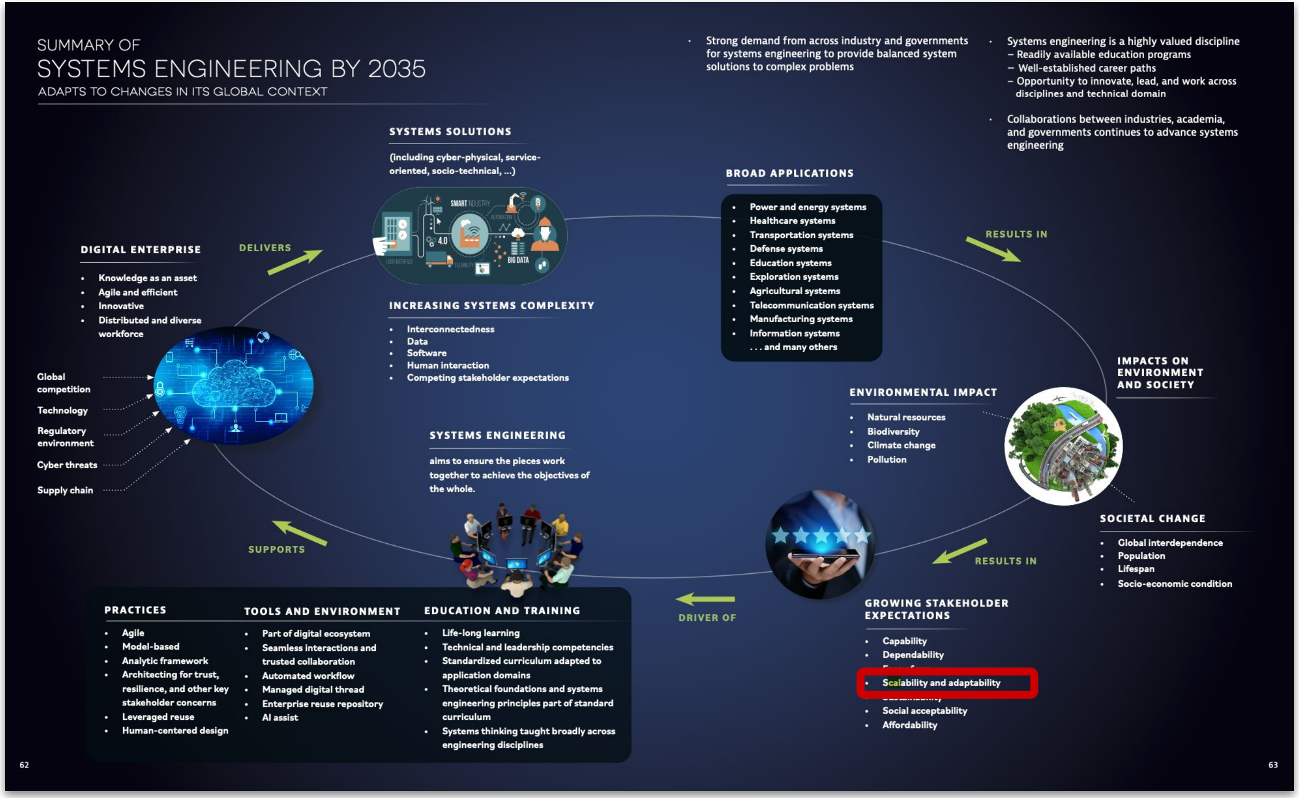
Requirements as first-class citizens



Outline

1. Concrete examples
2. Context: the “CoCoVaD Airbus chair”
3. Why is there a concern?
4. Requirements for requirements’ scalability
5. Conclusion

INCOSE SE Vision 2035



https://www.incose.org/docs/default-source/se-vision/incose-se-vision-2035.pdf

INCOSE SE Vision 2035

9. SCALABLE Scalable systems are adaptable to a range of performance and capabilities without breaking their fundamental architecture. This is an important trait because of the high cost associated with initial infrastructure investments or non-recurring engineering costs.

Scalability and adaptability must be a consideration from system inception and be reconciled with the conflicts that scalability often presents for products optimized for single applications.

Growing Stakeholder Expectations

1. SIMPLE System solutions must provide expected capability but hide as much design complexity as possible, have simple user interfaces, be understandably failure tolerant, and easy to use. Employing human-centered design and taking into account the entire user experience will be increasingly important to system acceptance.

2. TIMELY Systems must be developed and placed into use in a timely fashion to assure customer demand and market conditions are conducive to systems success and provide sponsor value.

3. SAFE Systems, driven by software-intensive designs, are increasingly being used in applications in which human, environmental, and property safety is a significant concern.

Ever increasing levels of safety and resilience must be assured in the face of increasing systems complexity.

4. SECURE System complexity, global connectivity, and IT dependence give rise to system vulnerabilities. The challenges for averting unwanted intrusions or for mitigating the results of intrusions have grown enormously.

Threats must be continuously assessed throughout the system life cycle and solutions implemented, ensuring security and cyber-defense against both ad hoc and organized (national actor) threats.

5. STABLE AND PREDICTABLE Systems of the future must be stable, reliable and predictable in order to meet operational needs, achieve customer acceptance, operate efficiently, minimize unintended consequences, avoid liability, and provide expected value. Systems must be validated to be consistent with customer stability expectations across a wide variety of use cases and stress conditions.

6. SMART Smart systems are able to cope with a changing and unknown environment, assist human operators, or self-organize to provide products and services. Social, functional and physical demands must be integrated to create valuable systems solutions that are resilient in their operational environment.

7. SUSTAINABLE Stakeholders will demand, as a result of global imperatives and market forces, that systems and services be environmentally sustainable – such as minimizing waste and undesirable impacts to climate change. Sustainability as a system characteristic will be stressed as well as the sustainability ethic of the responsible enterprises.

8. MAINTAINABLE Systems developers must take into account maintenance costs over the full product life cycle, management of product diversity, pre-planned product evolution and disposal, capture and disposition of knowledge gained from fielded systems, and the ability to perform upgrades while operational. Engineers must be able to balance the often contradictory technologically driven demands of support for deployed systems.

9. SCALABLE Scalable systems are adaptable to a range of performance and capabilities without breaking their fundamental architecture. This is an important trait because of the high cost associated with initial infrastructure investments or non-recurring engineering costs.

Scalability and adaptability must be a consideration from system inception and be reconciled with the conflicts that scalability often presents for products optimized for single applications.

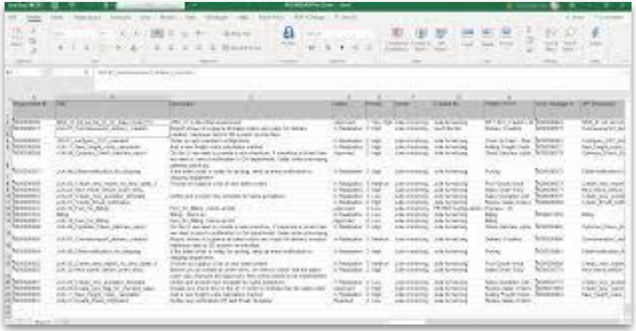
10. AFFORDABLE For systems to be viable they must be affordable within the context of the total cost of ownership. They must provide value to systems sponsors and users, and, very often, the general public. Developers must understand systems value from the perspective of all stakeholders and incorporate these, often competing values, into design decisions.

Main challenges
(compilation)

- The top 5 challenges of software requirements management
- A Systematic Literature Review on the Scalability Issues in Software Requirements Prioritization
- Scaling Up Requirements Engineering
- And... chatGPT!

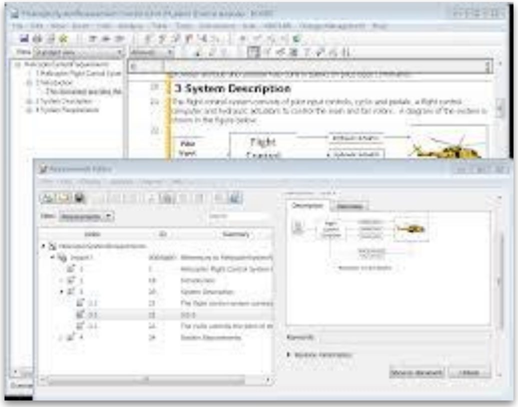
Tool Inadequacy

Word & Excel!!



- Poor versioning management
- Heavy review process
- Poor variability support
- Error prone

DOORS



=> 8 days without access for a version change in the tool!

=> “Portability team”

=> No semantic

=> No smart modification

=> No interoperability

IBM DOORS Next 7.x performance considerations

News

Abstract

IBM DOORS Next 7.x changed the underlying data store from Jena to a relational database. As a result, underlying queries might run differently in 7.x.

Requirements

- Strong version control
- Strong access control
- Support for static verification (e.g., coverage/compliance matrices)
- Support for animation/simulation/execution
    - Not just “test based”
- Better integration with other tools (e.g., project management, **IDE**)

Communication & Collaboration

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

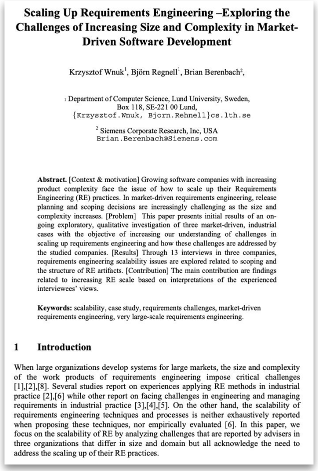
Requirements

- Central hub to view all requirements (with easy access by everybody involved)
- Repetitive processes to foster communication (i.e. regular meetings)
- Work by iteration (to revisit regularly requirements checklist)
- Accurate and up-to-date requirements
- Management tool (to support trial and error)

Complexity management

Factors to be considered

- Number of comparisons
- Time
- Scarcity of the automation
- Human efforts required
- Scoping
- Structure of RE Artifacts



Scaling Up Requirements Engineering –Exploring the Challenges of Increasing Size and Complexity in Market-Driven Software Development

Krzysztof Wnuk[1], Björn Regnell[1], Brian Berenbach[2],

[1] Department of Computer Science, Lund University, Sweden, Box 118, SE-221 00 Lund, {Krzysztof.Wnuk, Bjorn.Rehnell}cs.lth.se

[2] Siemens Corporate Research, Inc, USA Brian.Berenbach@Siemens.com

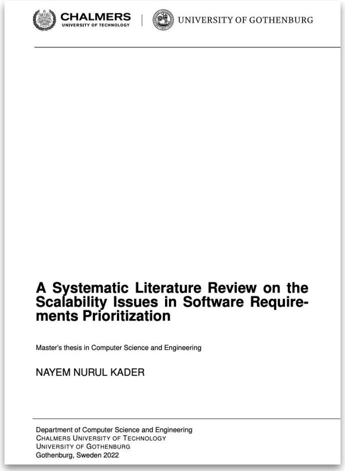
Abstract. [Context & motivation] Growing software companies with increasing product complexity face the issue of how to scale up their Requirements Engineering (RE) practices. In market-driven requirements engineering, release planning and scoping decisions are increasingly challenging as the size and complexity increases. [Problem] This paper presents initial results of an on-going exploratory, qualitative investigation of three market-driven, industrial cases with the objective of increasing our understanding of challenges in scaling up requirements engineering and how these challenges are addressed by the studied companies. [Results] Through 13 interviews in three companies, requirements engineering scalability issues are explored related to scoping and the structure of RE artifacts. [Contribution] The main contribution are findings related to increasing RE scale based on interpretations of the experienced interviewees' views.

Keywords: scalability, case study, requirements challenges, market-driven requirements engineering, very large-scale requirements engineering.

1 Introduction

When large organizations develop systems for large markets, the size and complexity of the work products of requirements engineering impose critical challenges [1],[2],[8]. Several studies report on experiences applying RE methods in industrial practice [2],[6] while other report on facing challenges in engineering and managing requirements in industrial practice [3],[4],[5]. On the other hand, the scalability of requirements engineering techniques and processes is neither exhaustively reported when proposing these techniques, nor empirically evaluated [6]. In this paper, we focus on the scalability of RE by analyzing challenges that are reported by advisers in three organizations that differ in size and domain but all acknowledge the need to address the scaling up of their RE practices.

https://cs.lth.se/fileadmin/cs/KrzysztofWnuk/Wnuk_Berenbach_Regnell_CameraREADY.pdf



CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

A Systematic Literature Review on the Scalability Issues in Software Requirements Prioritization

Master's thesis in Computer Science and Engineering

NAYEM NURUL KADER

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2022

https://gupea.ub.gu.se/handle/2077/74139

Requirements

- Better **overview** of the size and dynamics of scope changes
- Methods of **prioritizing** requirements
- Importance of requirements **architecture**
- Efficient method of **knowledge** management (that can speed up complex investigations)

Variability and Prioritization

Impact of change

- Complexity
  - => more requirements ?
  - => more complex requirements ?
- More requirements
  - => management of impacts
  - => more interfaces
  - => impact on the overall process

> “Change is the only constant in life”
>
> Heraclitus

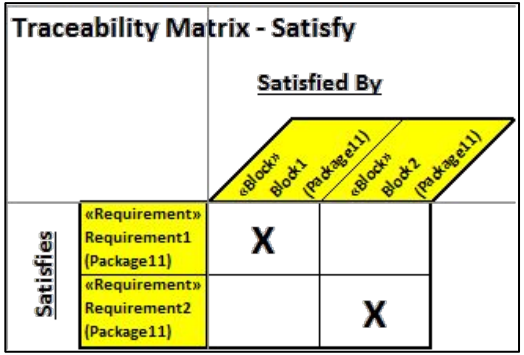
Requirements

- Customer feedback sessions
- PLM (Product Lifecycle Management)

Traceability

Requirements

- A key mechanism
- Need semantic (for the links)
- Right level of granularity
- Ontologies ?
- NoSQL ?



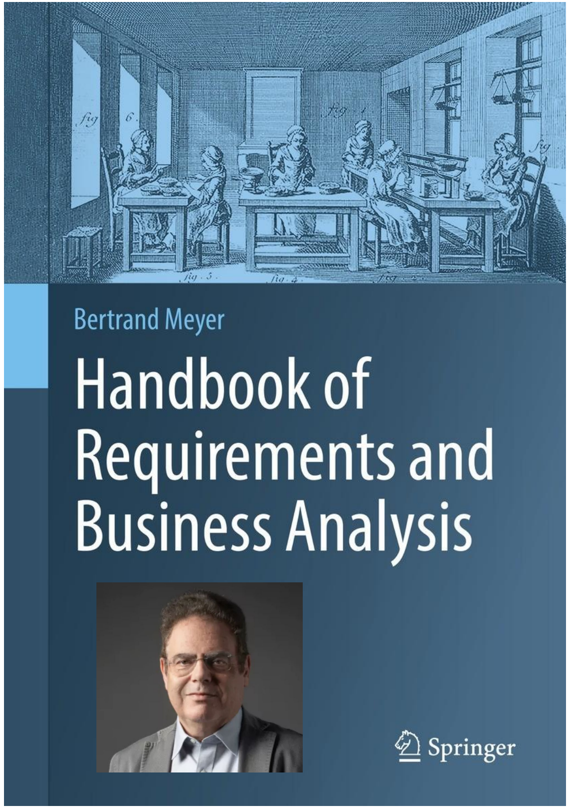
Outline

1. Concrete examples
2. Context: the “CoCoVaD Airbus chair”
3. Why is there a concern?
4. Requirements for requirements’ scalability
5. Conclusion

Requirements for requirements' scalability

- Improved traceability (semantic meaning)
- Zoom in / Zoom out capabilities
- Animation capabilities ("what-if scenarios")

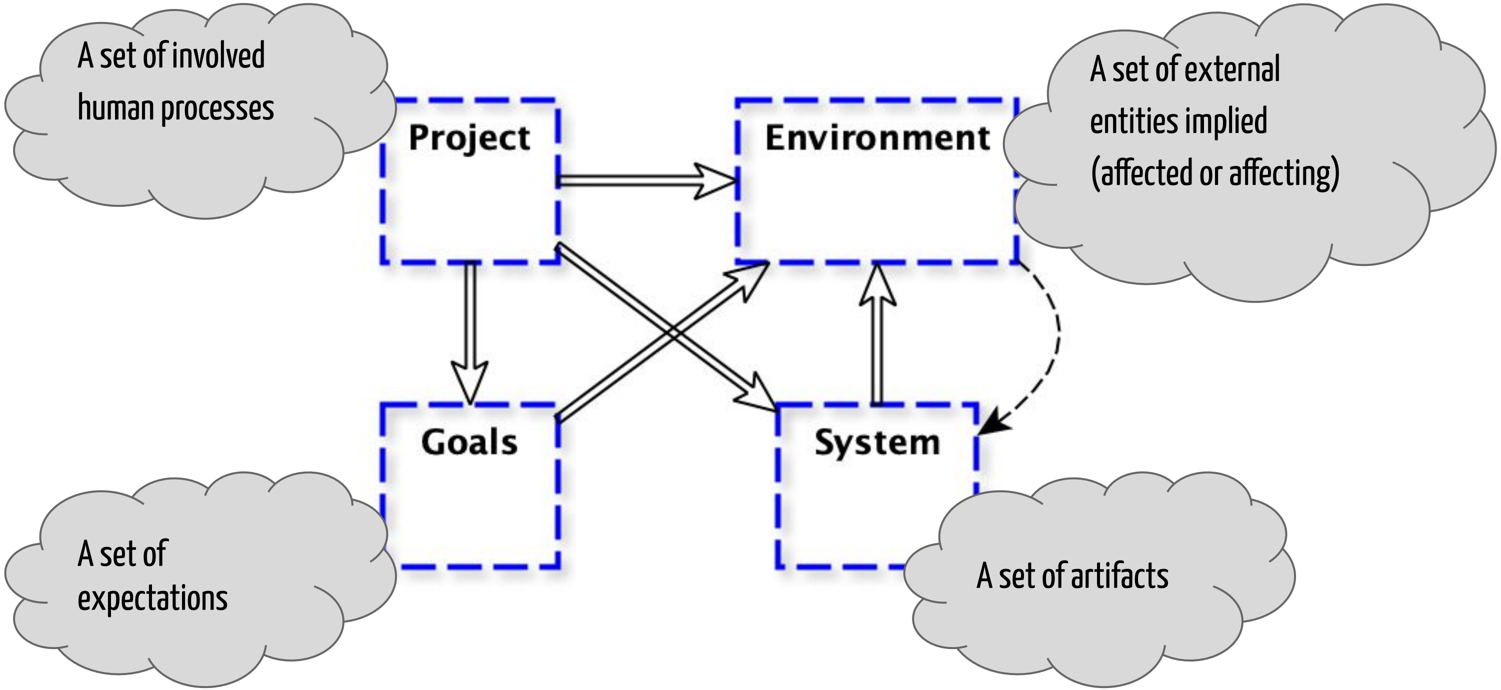
Ongoing efforts



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

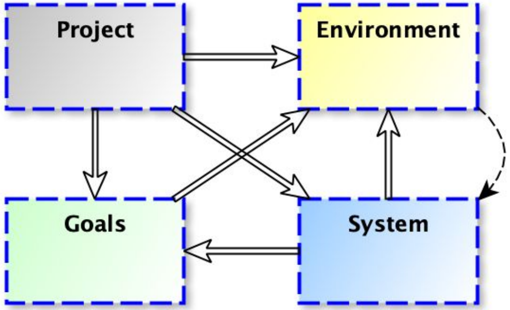
Context (universe of discourse)

"a **project** to develop a **system**, in a certain **environment**, to satisfy a set of **goals**"



PEGS Cheatsheet

v 2024-03-16



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*

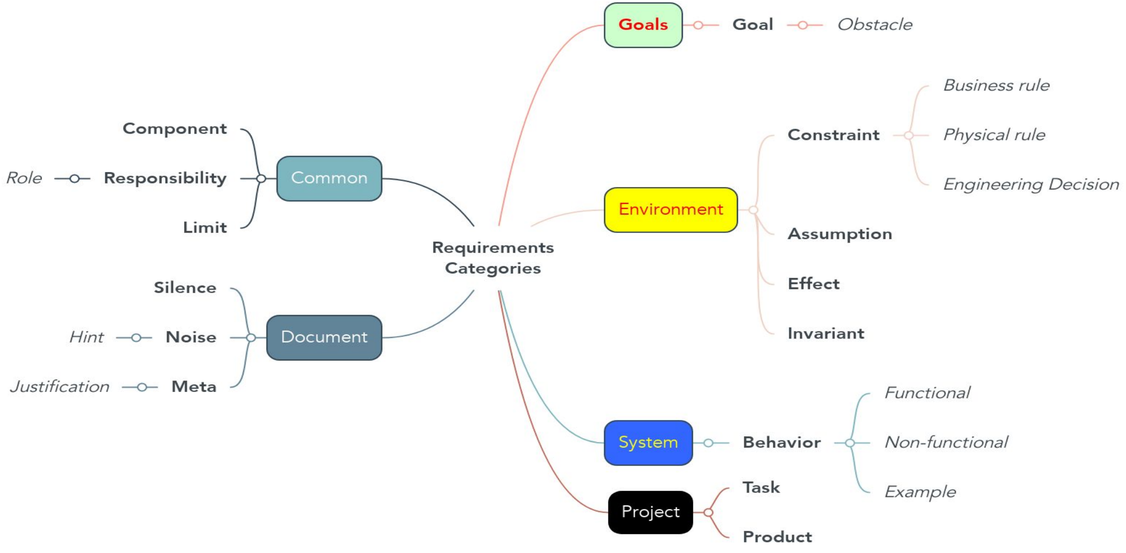
Environment (E)

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

* These chapters should not be empty (following the Minimum Requirements Outcome Principle)



https://requirements.university/

Relations between requirements

- Disjoins ($X \,||\, Y$)
- Belongs ($X \subseteq Y$)
- Repeats ($X \Leftrightarrow Y$)
- Contradicts ($X \oplus Y$)
- Extends ($X > Y$)
- Excepts ($X \backslash\backslash Y$)
- Constrains ($X \blacktriangleright Y$)
- Characterizes ($X \longrightarrow Y$)

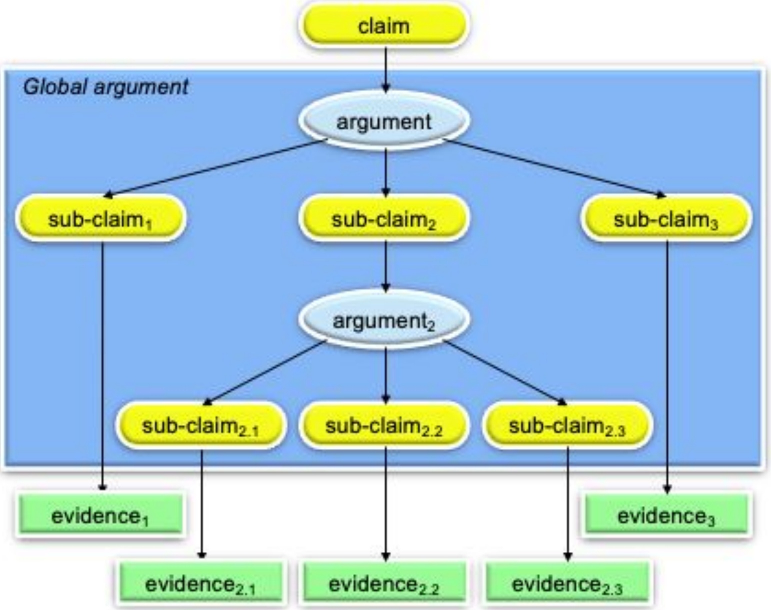
Requirements documents can be tested!

```
#------------------------------
# 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
```

Use of Justifications as Requirements Architecture?

Justification Framework in a Nutshell

- **Justification frameworks may be used to explain, step-by-step, the rationales behind engineering decisions**
  - And also to justify the legitimacy of assumptions
  - Can express rigorous and objective aspects, but also qualitative and subjective aspects
  - May refer to elements external to the modelling framework
    - Historical data, international standards, regulations ...
  - More informative than simple traceability links
  - May also be used to justify the adequacy and correctness of models and supporting tools
- **Different types of elementary arguments**
  - Concretisation, Substitution, Decomposition, Calculation
- **ISO - IEC - IEEE 15026-2 (2011)**



Self-promotion

SPRINGER LINK

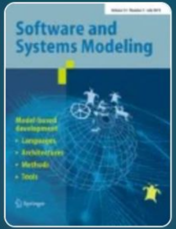
Find a journal | Publish with us | Track your research | Search

Home › Software and Systems Modeling › Article

Identifying and fixing ambiguities in, and semantically accurate formalisation of, behavioural requirements

Theme Section Paper | Published: 16 March 2024

(2024) Cite this article



Software and Systems Modeling

Aims and scope →

Submit manuscript →

Thuy Nguyen, Imen Sayar, Sophie Ebersold ✉ & Jean-Michel Bruel

Access this article

https://link.springer.com/article/10.1007/s10270-023-01142-0

Outline

1. Concrete examples
2. Context: the “CoCoVaD Airbus chair”
3. Why is there a concern?
4. Requirements for requirements’ scalability
5. Conclusion

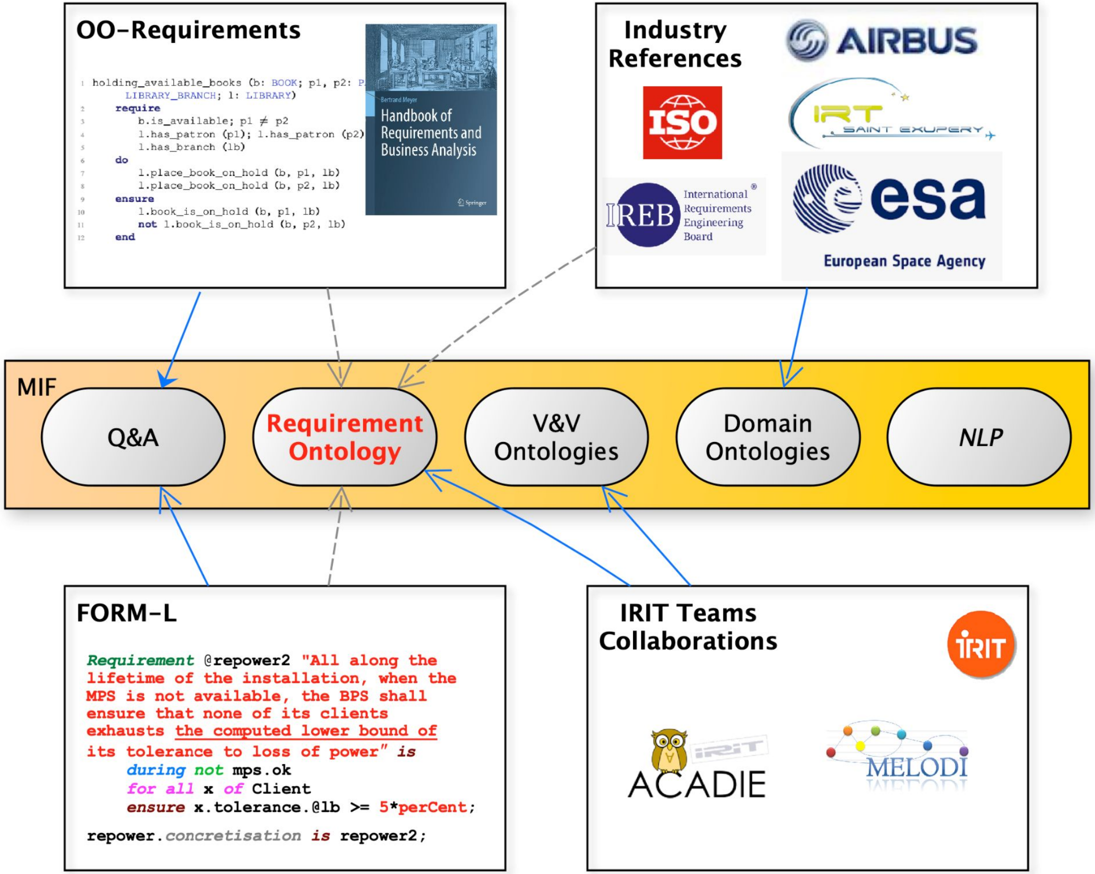
What to remember from all of this?

- Requirements are way more **complex** than just *“The system shall work.”*
- Organizing and classifying requirements helps **Q&A**
- Quality metrics & rules can be **implemented** and hence useful
- Scalability issues are **not specific** but **amplified**

One last thing...

We are hiring! (and looking for collaborations)

INTERESTED ?



Discussions time!

https://bit.ly/jmbruel

@jmbruel