Skoltech

Skolkovo Institute of Science and Technology

Software Requirements Engineering

Foundations of Software Engineering

FSE v2020.1, Block 1 Module 3

Alexey Artemov, Fall 2020

Lecture Outline

Skoltech
Skolkovo Institute of Science and Technology

§1. Software requirements overview [10 min]

1.1. What are software requirements?

1.2. Why spend time on software requirements?

§2. Types and qualities of software requirements [10 min]

2.1. Functional and non-functional requirements

2.2. Qualities of software requirements

Lecture Outline

Skoltech
Skolkovo Institute of Science and Technology

§3. Models for software requirements [10 min]

3.1. Why-what-how model

3.2. WRSPM: the reference model

3.3. MoSCoW requirements prioritization model

§4. Requirements development process [10 min]

4.1. Stages of developing requirements

4.2. Requirements specification documents

The Goal:
Think Before You Code

§1. Software requirements overview

What are software requirements?

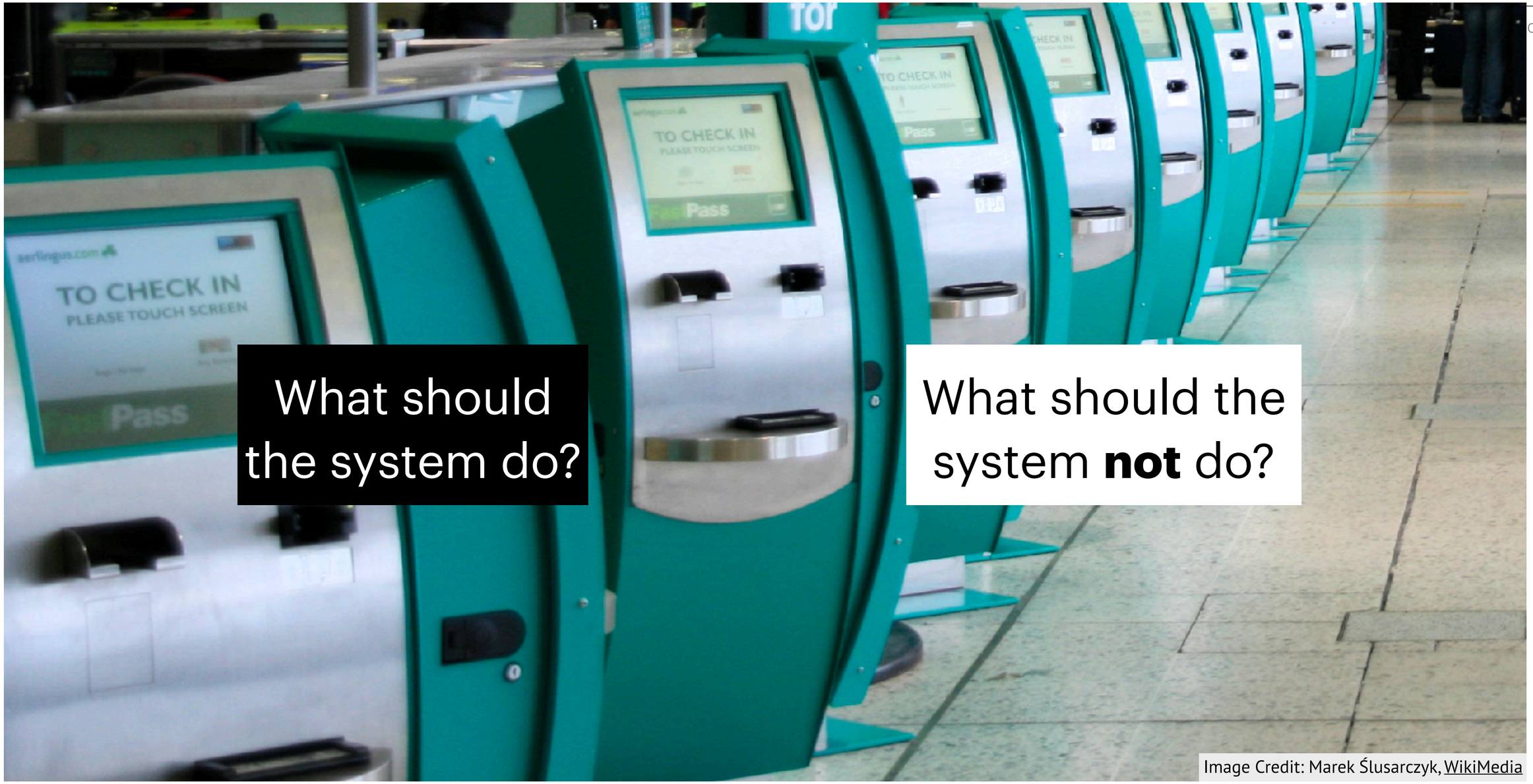


Image Credit: Marek Ślusarczyk, WikiMedia

§1. Software requirements overview

Skoltech
Skolkovo Institute of Science and Technology

1.1. What are software requirements? Why spend time on software requirements?

- Requirements: features that your application must provide. [1]
- Requirements: anything that determines the design choice. [2]
- Requirements: something that must be implemented
- Describe system behaviour, properties and attributes, may serve as constraints

- Variety of names: user, software, business, functional, system, product, project requirements
- Style and complexity highly depend on the project and the customer

Image Credit: WikiMedia

Image Credit: WikiMedia

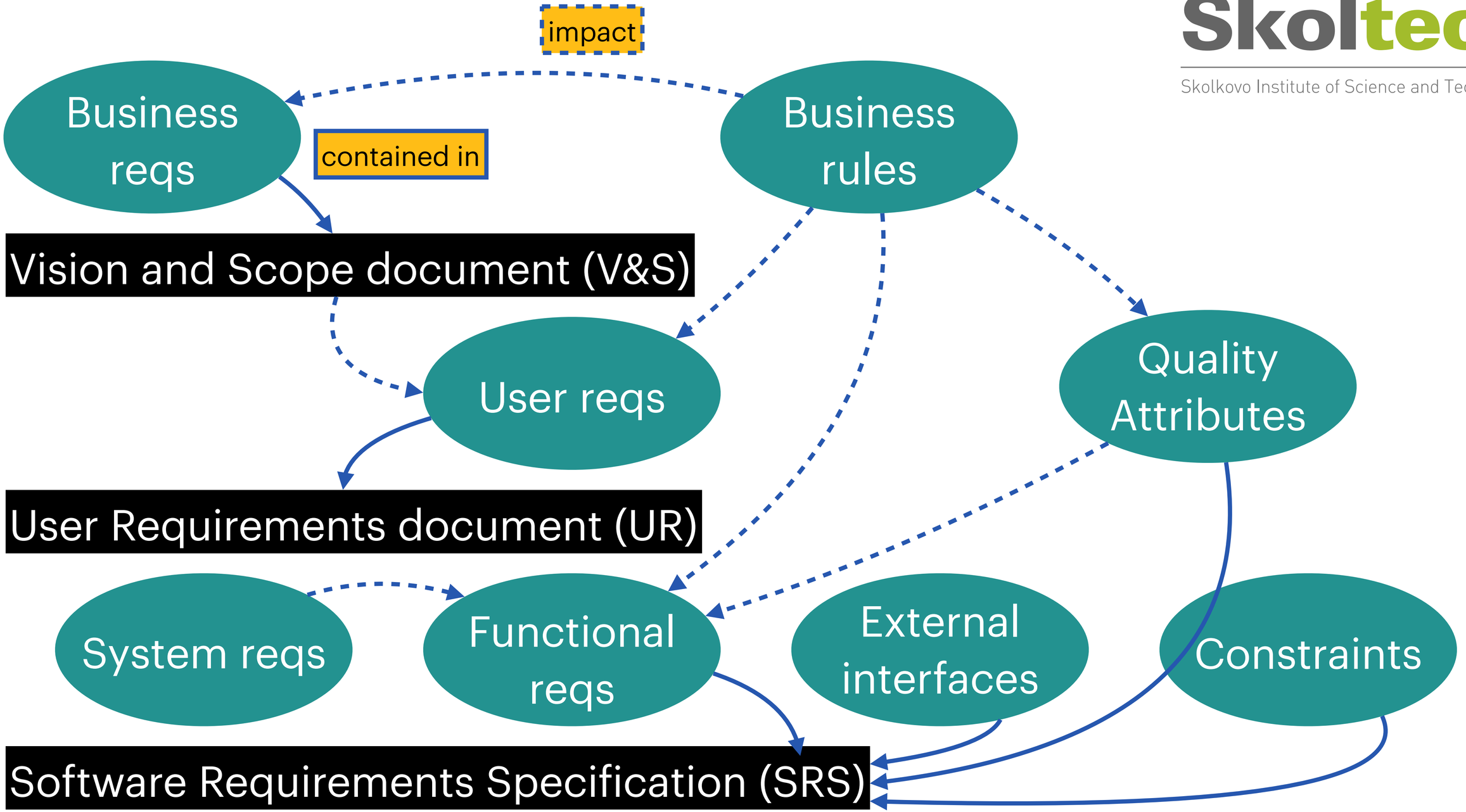


Image Credit: Karl Wiegers

Business level

Obtain 25% reduction in airport registration desk personnel costs.
Install a terminal that the passengers may use to perform self check-in.

User level

"As a passenger, I want to check-in for the flight, so I could get onboard."

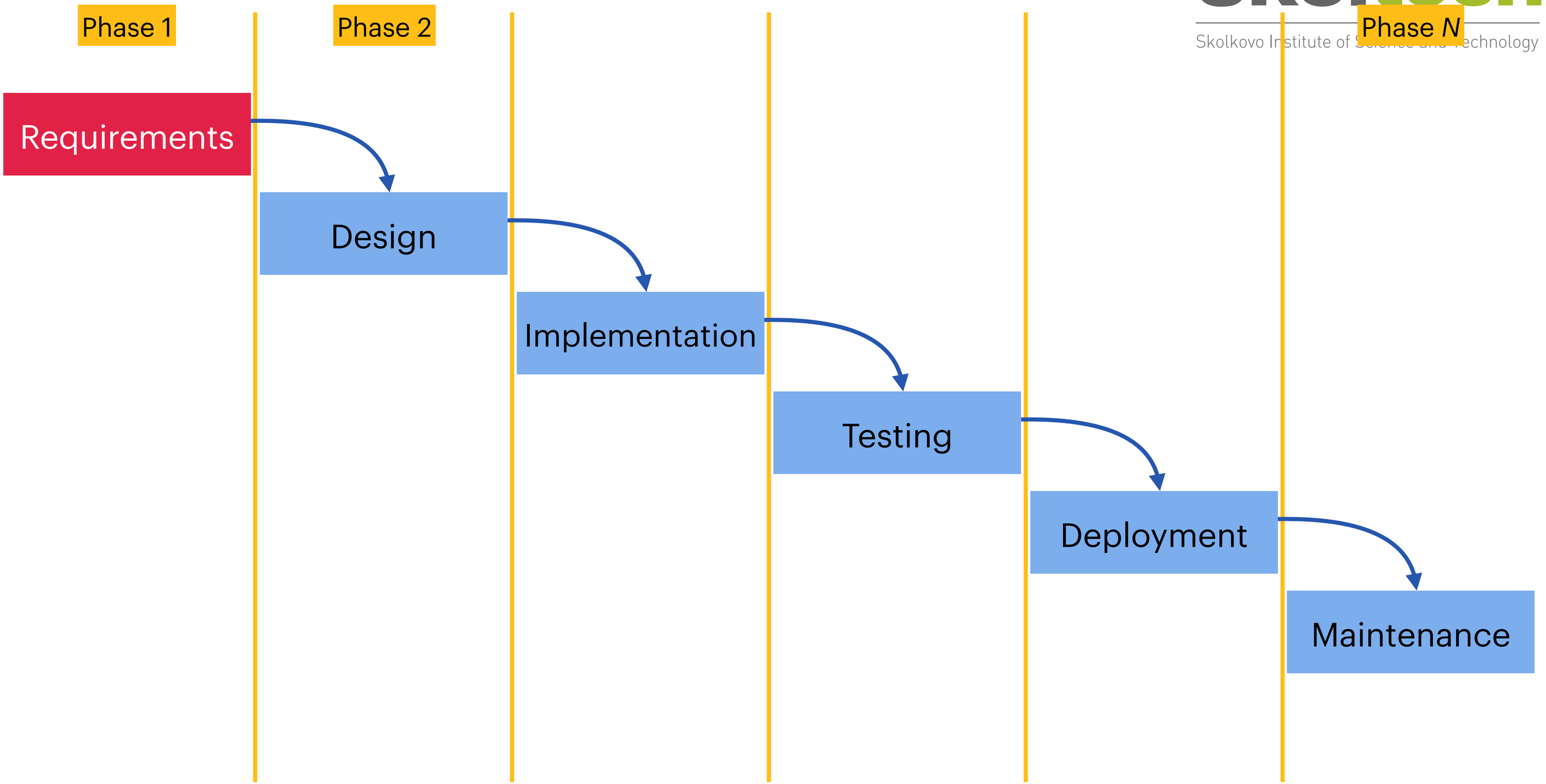
Client Managers
System End-Users
Client Engineers
Contractor Managers
System Architects

System level

- "The passenger shall be able to print the boarding passes to all flights that he has checked-in to."
- "If the passenger hasn't specified the preferences regarding the seat, the reservation system shall assign a seat automatically."

System End-Users
Client Engineers
System Architects
Software Developers

Why spend time on software requirements?


Phase 1
Phase 2
Phase N
Requirements
Design
Implementation
Testing
Deployment
Maintenance

§1. Software requirements overview

Skoltech
Skolkovo Institute of Science and Technology

1.2. Why spend time on software requirements?

- Gathering software requirements helps mitigate risks early in the project:
  - Insufficient user involvement
  - Poor planning
  - Growing user requirements
  - Ambiguous requirements
  - Gold plating requirements
  - Missing user categories

§1. Software requirements overview

1.2. Why spend time on software requirements?

- Less defects in the requirements and in the product
- Less rework
- Faster development and product release
- Less unused and unnecessary functionality
- Higher modifiability
- Less project scope creep
- Less misunderstanding
- More organized (aka less messy) development pipelines
- Higher customer and team satisfaction

Products that do what is expected of them.

§1. Software requirements overview

Skoltech
Skolkovo Institute of Science and Technology

Summary

- **Requirements:** a coherent enumeration of the system properties and functionality
- Different levels of requirements: business, user, and system
- Vision & Scope, User Requirements Doc, Software Requirements Specification

§2. Types and qualities of software requirements

Types of software requirements

§2. Types and qualities of software requirements

2.1. Types and qualities of software requirements

- **Functional requirements:**
  - services the system should provide
  - how the system should react to particular inputs
  - how the system should behave in particular
- **Non-functional requirements:**
  - constraints on services offered by the syster
  - constraints imposed by development standa
  - constraints on the development process

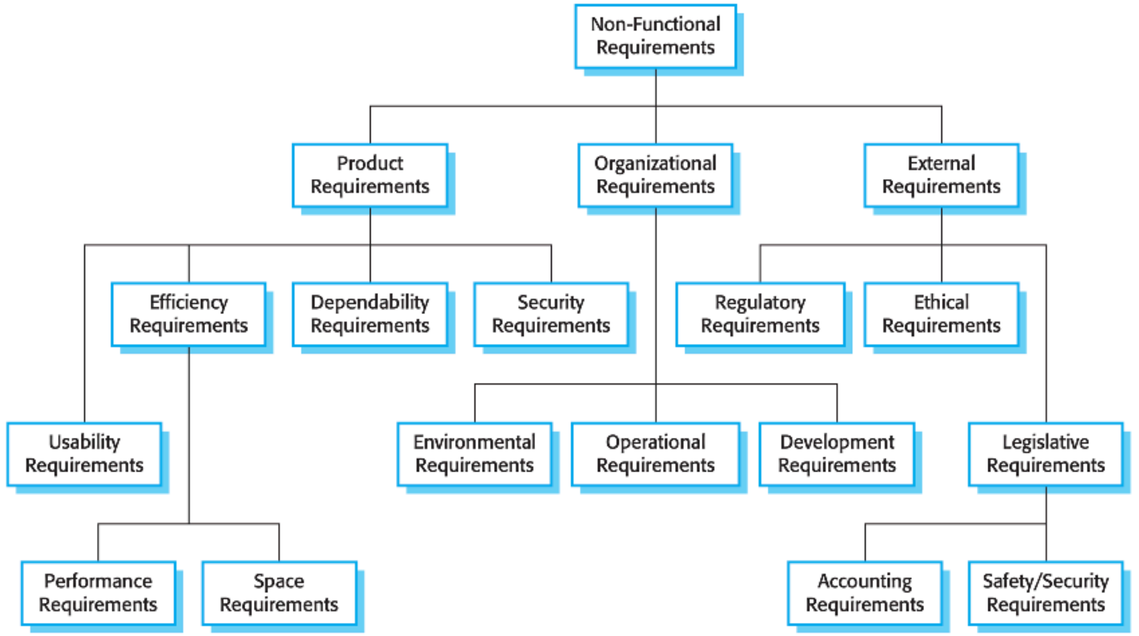
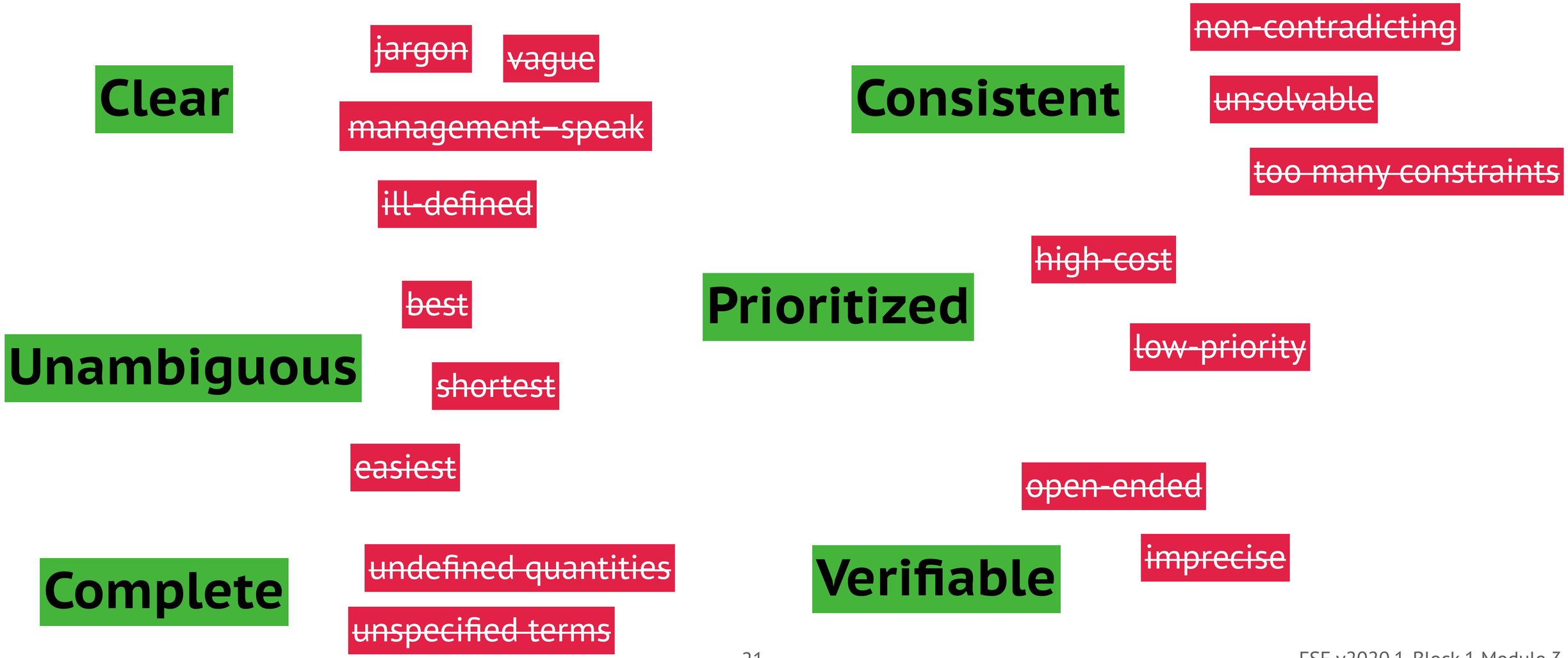


Image Credit: Ian Sommerville

Qualities of software requirements

§2. Types and qualities of software requirements

2.2. Qualities of software requirements



§2. Types and qualities of software requirements

Summary

- Functional and non-functional requirements
- Ideally, the user and system requirements should be clear, unambiguous, complete, and consistent, prioritized, and verifiable

§3. Requirements models

Why-what-how

Business reqs → WHY

"Increase profits by 25 percent"

"Increase demand and gain 10,000 new customers."

User reqs → WHAT

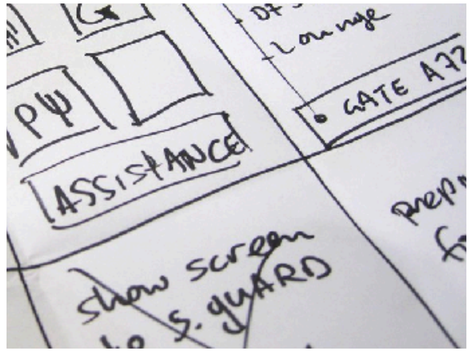


Image Credit: Flickr

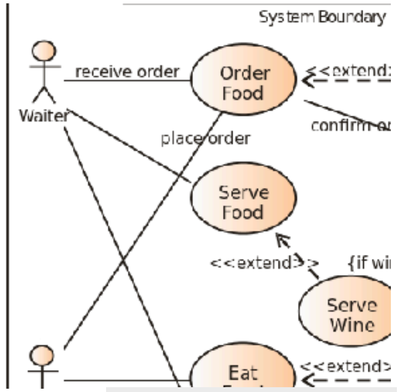


Image Credit: WikiMedia

System reqs → HOW

How the system is going to be constructed and function.

Functional, non-functional, implementation, …

§3. Requirements models

Skoltech
Skolkovo Institute of Science and Technology

3.1. Why-what-how

- Why: define the business need in customer terms
- What: define the system features that will serve and satisfy the business purpose
- How: define the way the system will be constructed that guarantees to provide these features

WRSPM model

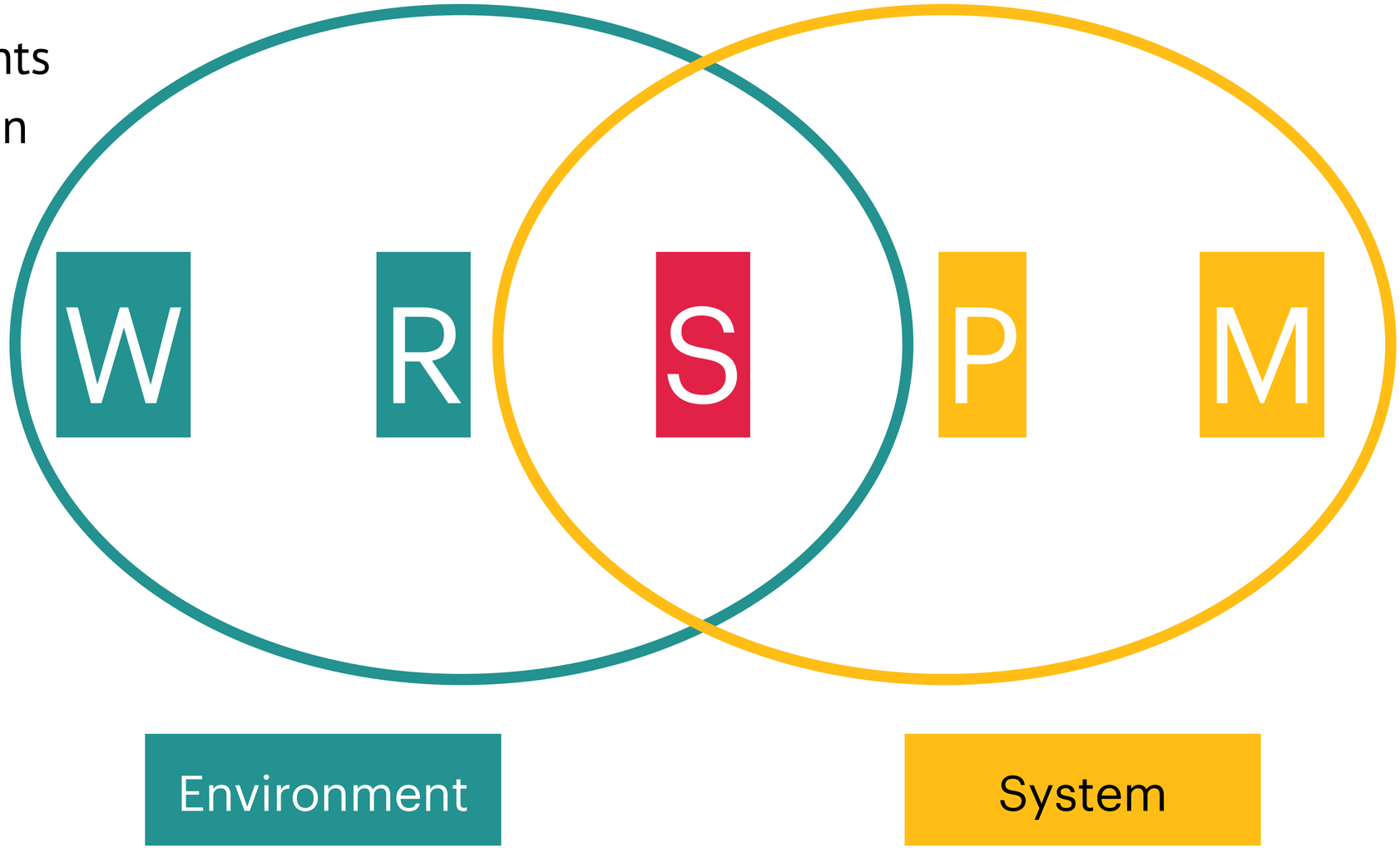
W = World

R = Requirements

S = Specification

P = Program

M = Machine

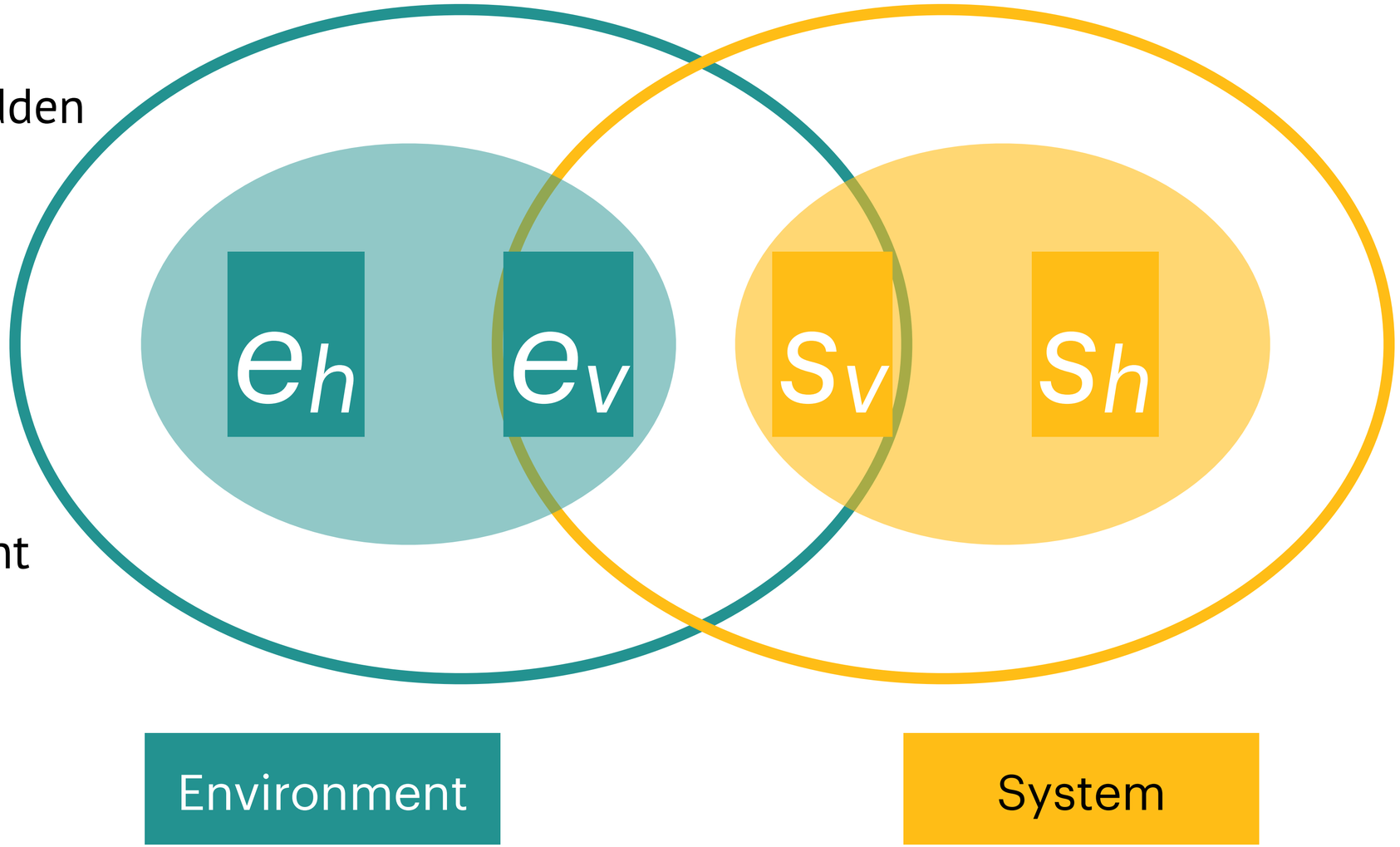


e_v = environment visible to the system

e_h = environment hidden from the system

s_v = system elements visible in the environment

s_h = system elements hidden from the environment



§3. Requirements models

Skoltech
Skolkovo Institute of Science and Technology

3.2. WRSPM: the reference model

- Patient Monitoring System
- **R:** warning if the patient's heartbeat stops
- **M:** platform with sensors and actuators
- **P:** program that sounds a buzzer
- **W:** — nurse close enough to hear the buzzer;
  — if patient's heart stops, chest sound falls down
- e_h**:** the nurse and the heartbeat of the patient
- e_v**:** sounds from the patient's chest
- s_v**:** the buzzer at the nurse's station
- s_h**:** internal representation of data from the sensor

Image Credit: ecgmonitorcable.com

§3. Requirements models

Skoltech
Skolkovo Institute of Science and Technology

3.2. WRSPM: the reference model

- A way of looking on the system in order to determine what the requirements might be
- Helps to identify the difference between a requirement, the user domain information and the specification
- Requirements must meet the specifications

MoSCoW: requirements prioritization

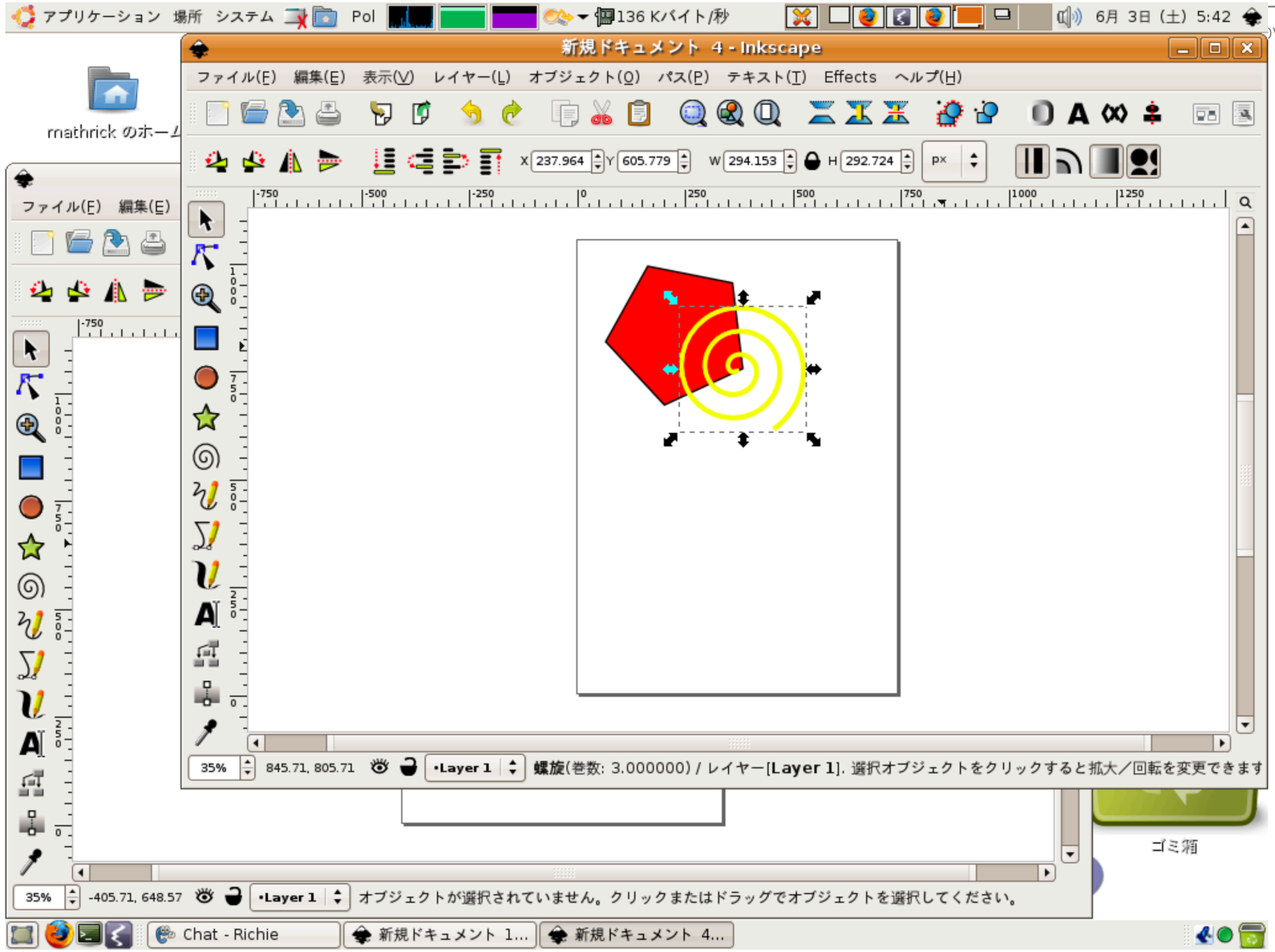


Image Credit: WikiMedia

Draw shapes	Click & drag to select	Colors from a palette	Rearrange palette/toolbar
Save/load	Click+Shift to edit selection	Custom colors	Auto-save
Protect drawing (close w/o save)	Edit line/fill styles	Support transparency	Export to LaTeX
Line/fill styles/ colors	Move objects	Copy/cut/paste selected object	Online help
Selecting objects	Delete objects	Copy/cut/paste bitmaps	Online tutorials

THE MOSCOW METHOD

MOSCOW is an acronym to help you remember a common system for prioritizing application features. The consonants in MOSCOW stand for the following:

M—Must. These are *required* features that must be included. They are necessary for the project to be considered a success.

S—Should. These are *important* features that should be included if possible. If there's a work-around and there's no room in the release 1 schedule, these may be deferred until release 2.

C—Could. These are *desirable* features that can be omitted if they won't fit in the schedule. They can be pushed back into release 2, but they're not as important as the "should" features, so they may not make it into release 2, either.

W—Won't. These are *completely optional* features that the customers have agreed will not be included in the current release. They may be included in a future release if time permits. (Or they may just be included in the requirements list to make a particularly loud and politically connected customer happy, and you have no intention of ever including these features.)

Image Credit: Rod Stephens [1]

Draw shapes	Click & drag to select	Colors from a palette	Rearrange palette/toolbar
Save/load	Click+Shift to edit selection	Custom colors	Auto-save
Protect drawing (close w/o save)	Edit line/fill styles	Support transparency	Export to LaTeX
Line/fill styles/ colors	Move objects	Copy/cut/paste selected object	Online help
Selecting objects	Delete objects	Copy/cut/paste bitmaps	Online tutorials

Must: cannot omit, program not usable without

Should: significant common enhancement

Could: useful or makes us special

Won't: unnecessary, confusing

§3. Requirements models

Skoltech
Skolkovo Institute of Science and Technology

3.4. MoSCoW requirements prioritization model

- Identify necessary and auxiliaxy features for a system to have
- *Must:* required features, must be included (initial release 1)
- *Should:* important features, should be included if possible (release 2)
- *Could:* desirable features, can be omitted if don't fit in the schedule (release 2)
- *Won't:* completely optional features (not in release 1 or 2)

§4. Requirements development process

§4. Requirements development process

2.1. Stages of developing requirements

- Overall process:
  - (Feasibility study) ⟶ Elicitation ⟶ Analysis ⟶ Specification ⟶ Validation

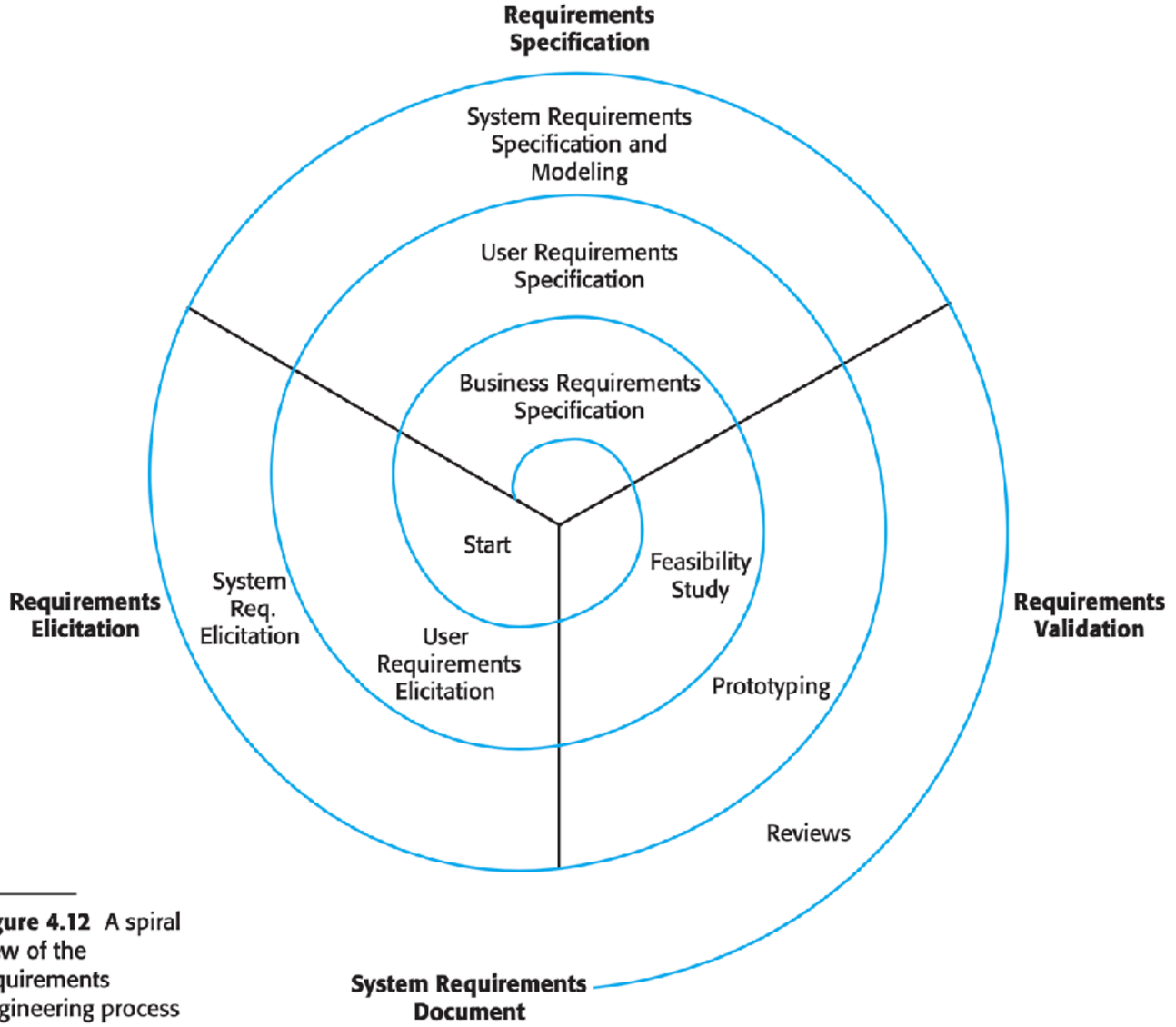


Figure 4.12 A spiral view of the requirements engineering process

Image Credit: Ian Sommerville

§4. Requirements development process

2.1. Stages of developing requirements

- Documentation:
  - Defining business objectives ⟶ Vision and Scope document
  - Developing user requirements ⟶ User Requirements document
  - Developing software requirements ⟶ Software Requirements Specification document (ISO/IEC/IEEE, 2011)

§4. Requirements development process

2.1. Stages of developing requirements: Elicitation

- Customers and end-users involved
- **What:** application domain, services, performance, hardware
- **Sources:** documentation, stakeholders, specifications of similar systems
- **How:** interview, observe, use scenarios, device prototypes

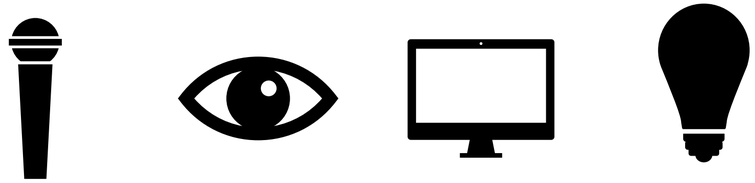


Image Credit: pxhere.com

§4. Requirements development process

2.1. Stages of developing requirements: Analysis

- Classification and organization
  - Group related
  - Use architectural design
- Prioritization and negotiation
  - Resolve conflicts
  - Compromise

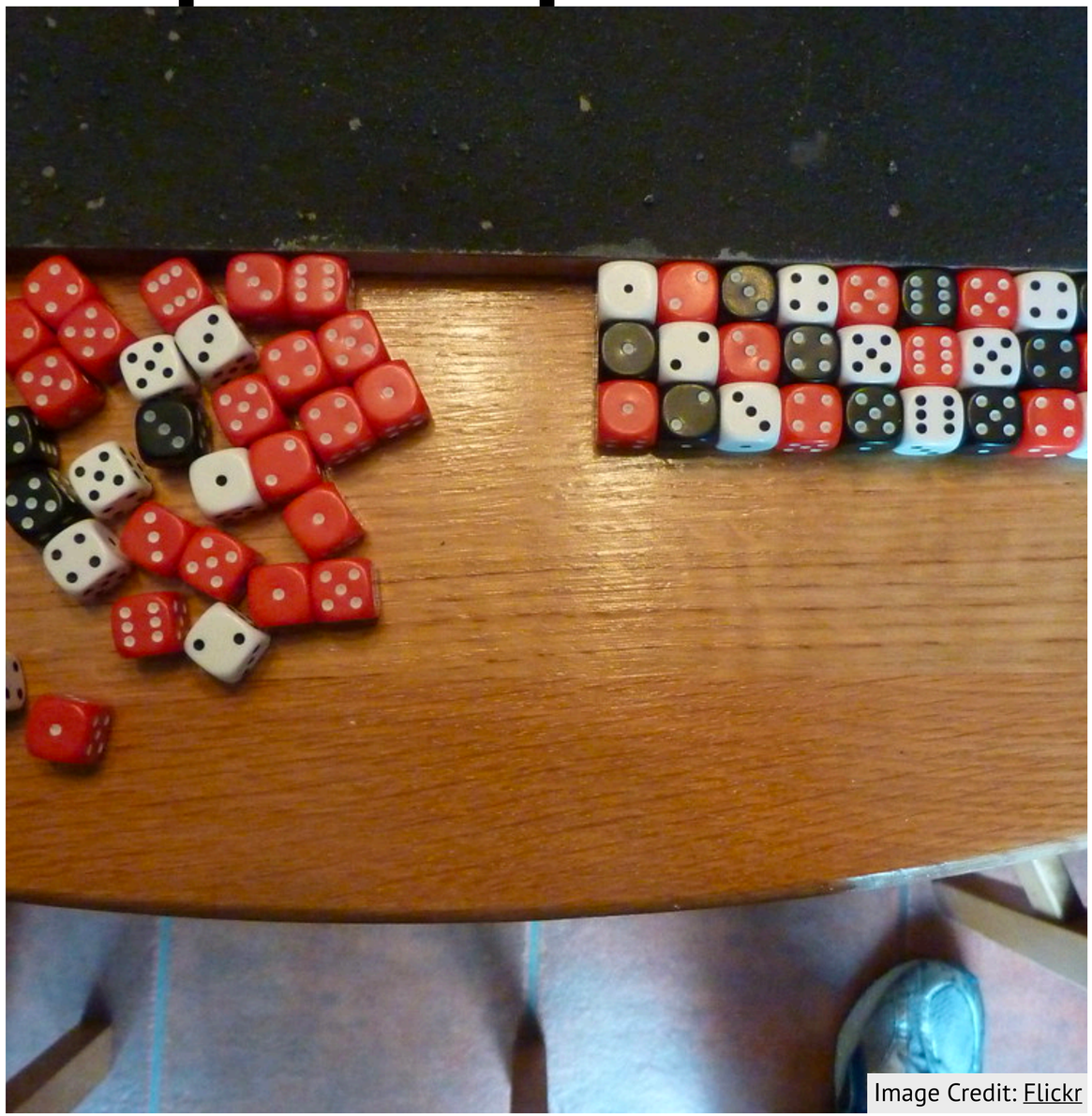


Image Credit: Flickr

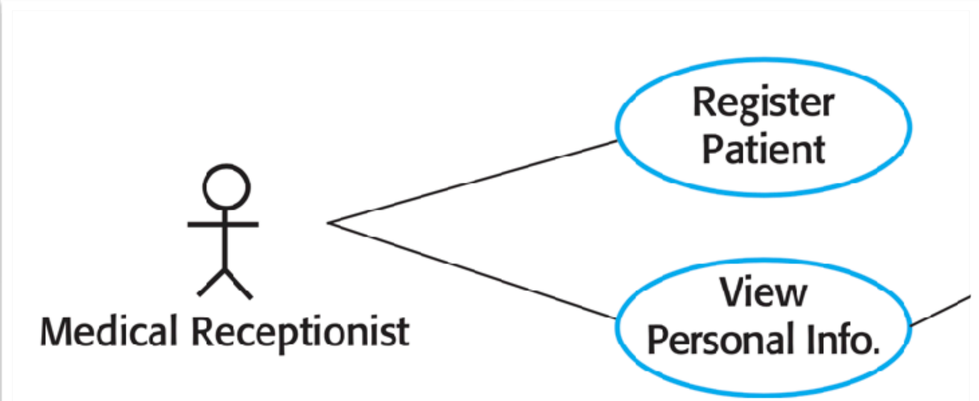
§4. Requirements development process

2.1. Stages of developing requirements: Specification

Natural language sentences

"3.2 The system shall measure the blood sugar and deliver insulin, if required, every 10 minutes."

Graphical notations



Example Credit: Ian Sommerville

Structured natural language

Insulin Pump/Control Software/SRS/3.3.2

Function	Compute insulin dose: Safe sugar level.
Description	Computes the dose of insulin to be delivered when the current measured sugar level is in the safe zone between 3 and 7 units.
Inputs	Current sugar reading (r2), the previous two readings (r0 and r1).
Source	Current sugar reading from sensor. Other readings from memory.
Outputs	CompDose—the dose in insulin to be delivered.

Design description languages

Mathematical specifications

Example Credit: Ian Sommerville

§4. Requirements development process

2.1. Stages of developing requirements: Validation

- Validity
- Consistency
- Completeness
- Realism
- Verifiability

- Requirement reviews
- Prototyping
- Test-case generation

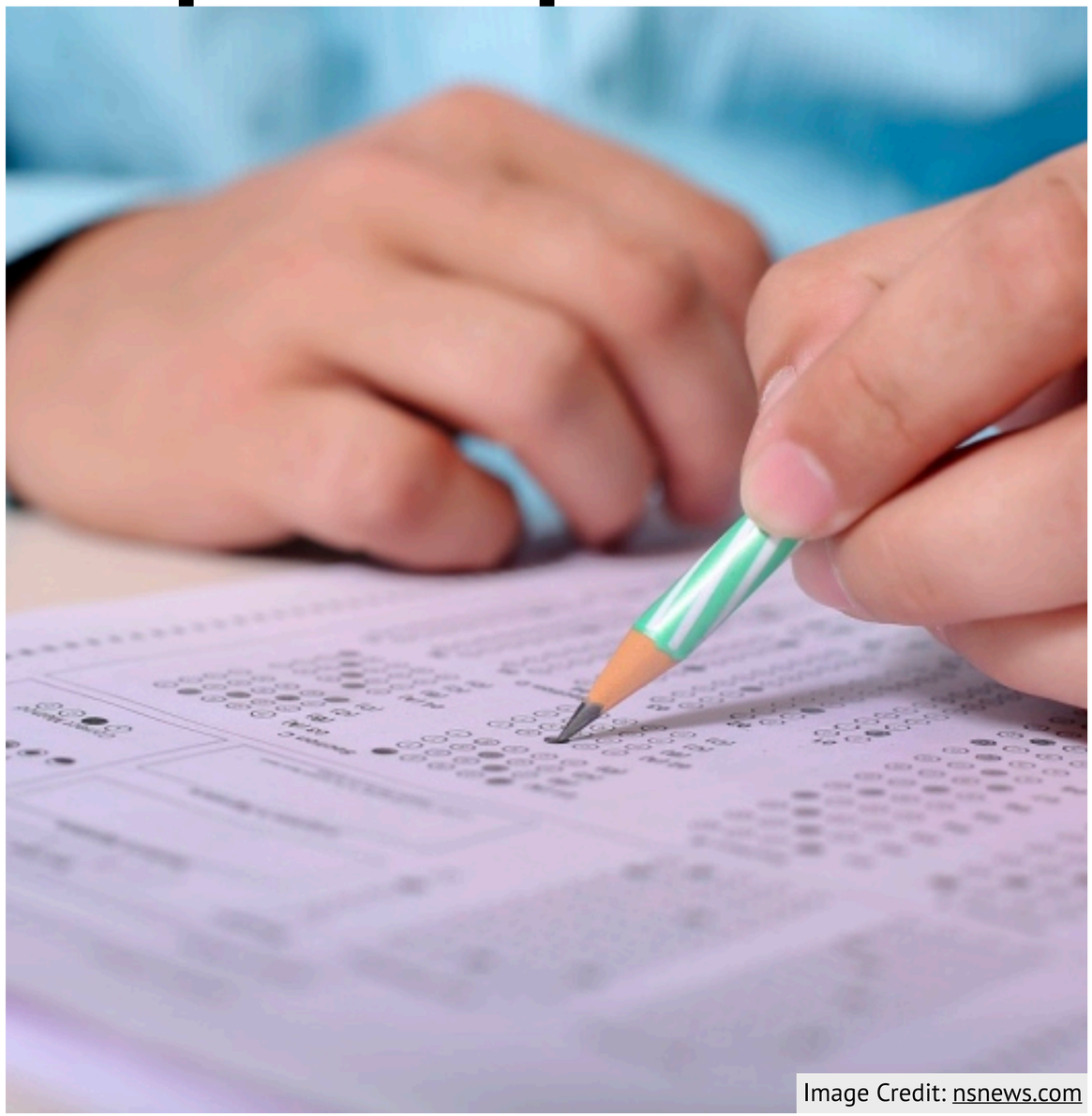


Image Credit: nsnews.com

§4. Requirements development process

Skoltech
Skolkovo Institute of Science and Technology

Summary

- Elicitation: discover the application domain, services the system should provide, the required performance of the system, hardware constraints, ...
- Analysis: group subsets of requirements together and resolve conflicting requirements
- Specification: use structured approaches to write down the description of the system
- Validation: check that requirements actually define the system that the customer really wants

References

1. Stephens, R. (2015). Beginning software engineering. John Wiley & Sons.
2. Wiegers, K., & Beatty, J. (2013). *Software requirements*. Pearson Education.
3. Sommerville I. *Software engineering 9th Edition*.