

Software Requirements Engineering

Foundations of Software Engineering

FSE v2020.1, Block 1 Module 3

Alexey Artemov, Fall 2020

1

FSE v2020.1, Block 1 Module 3

Lecture Outline

§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

2

FSE v2020.1, Block 1 Module 3

Lecture Outline

§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

3

FSE v2020.1, Block 1 Module 3

The Goal:
Think Before You Code

4

FSE v2020.1, Block 1 Module 3

§1. Software requirements overview

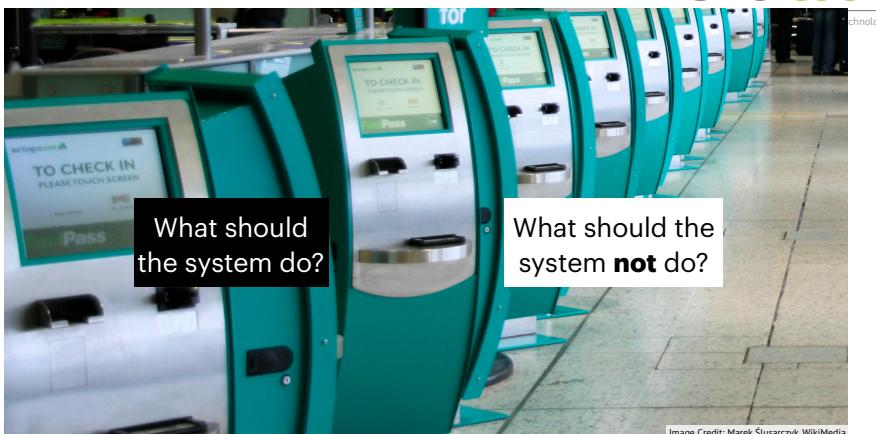
5

FSE v2020.1, Block 1 Module 3

What are software requirements?

6

FSE v2020.1, Block 1 Module 3



7

FSE v2020.1, Block 1 Module 3

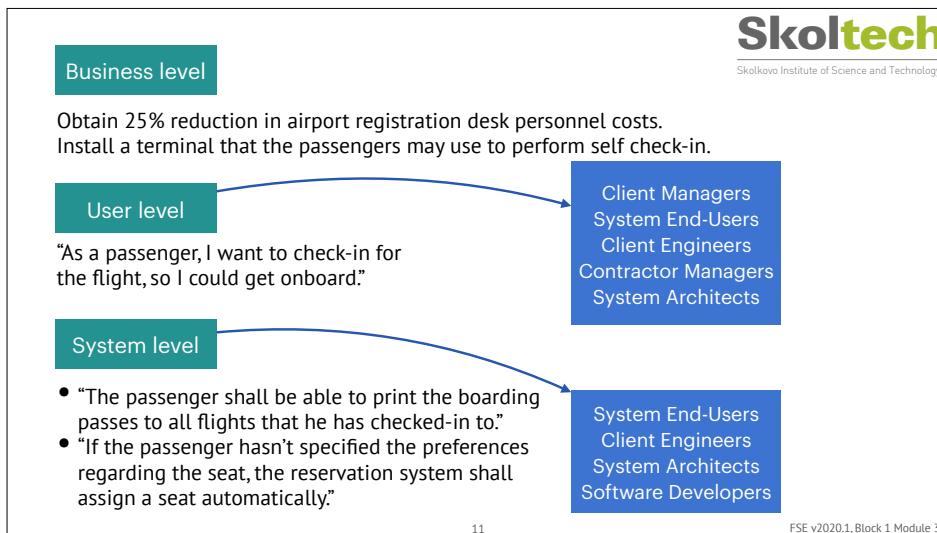
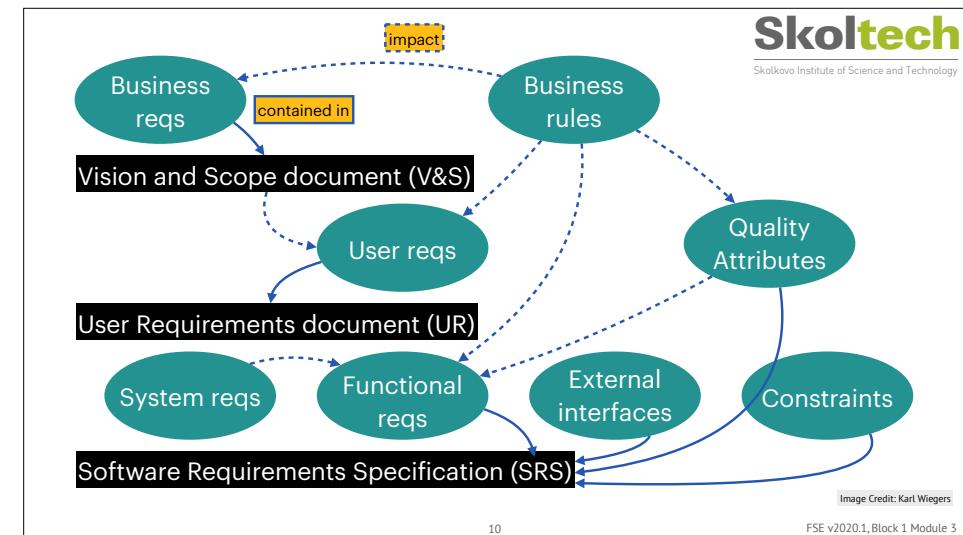
§1. Software requirements overview

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

8

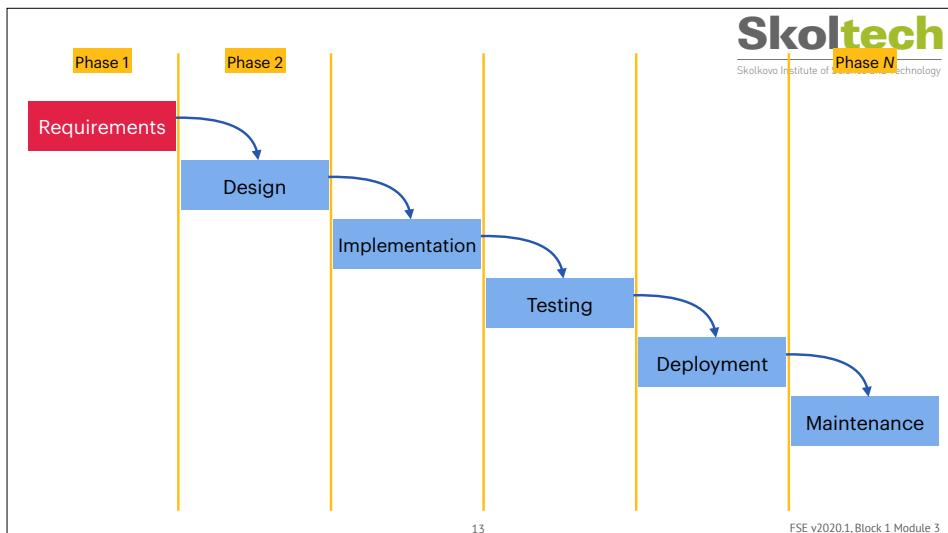
FSE v2020.1, Block 1 Module 3



Why spend time on software requirements?

12

FSE v2020.1, Block 1 Module 3



§1. Software requirements overview

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

14

FSE v2020.1, Block 1 Module 3

§1. Software requirements overview

1.2. Why spend time on software requirements?

Products that do what is expected of them.

- 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

15

FSE v2020.1, Block 1 Module 3

§1. Software requirements overview

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

16

FSE v2020.1, Block 1 Module 3

§2. Types and qualities of software requirements

17

FSE v2020.1, Block 1 Module 3

Types of software requirements

18

FSE v2020.1, Block 1 Module 3

§2. Types and qualities of software requirements

2.1. Types and qualities of software requirements

• Functional requirements:

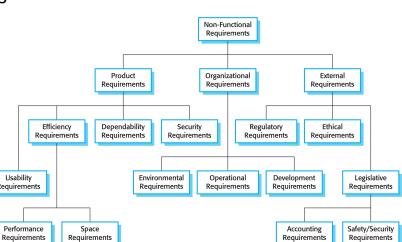
- services the system should provide
- how the system should react to particular inputs
- how the system should behave in particular

• Non-functional requirements:

- constraints on services offered by the system
- constraints imposed by development standards
- constraints on the development process

19

FSE v2020.1, Block 1 Module 3



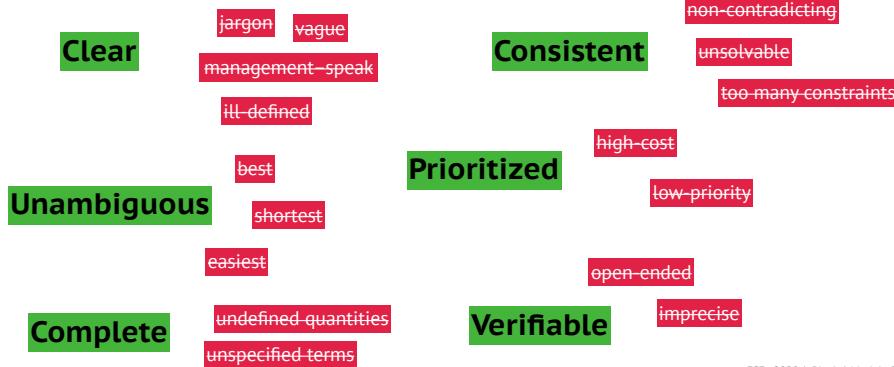
Qualities of software requirements

20

FSE v2020.1, Block 1 Module 3

§2. Types and qualities of software requirements

2.2. Qualities of software requirements



21

FSE v2020.1, Block 1 Module 3

§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

22

FSE v2020.1, Block 1 Module 3

§3. Requirements models

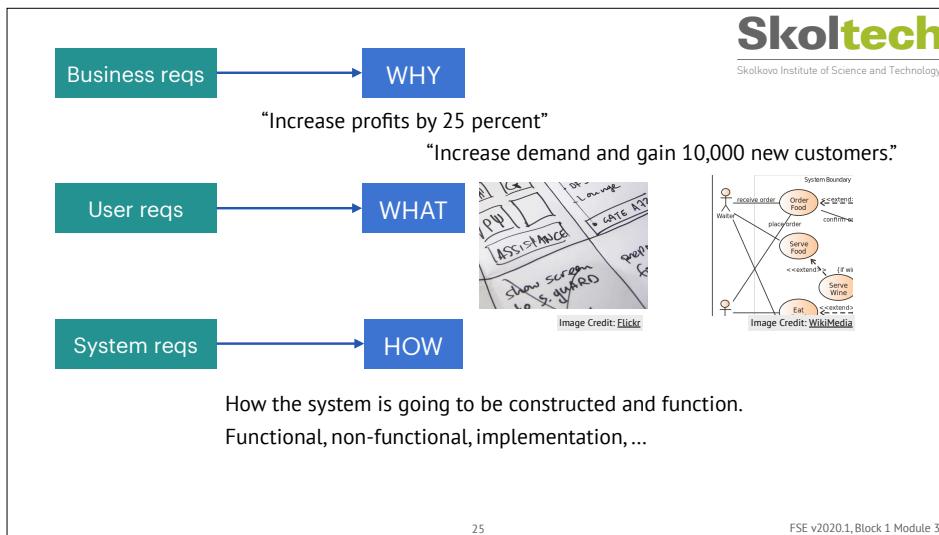
23

FSE v2020.1, Block 1 Module 3

Why-what-how

24

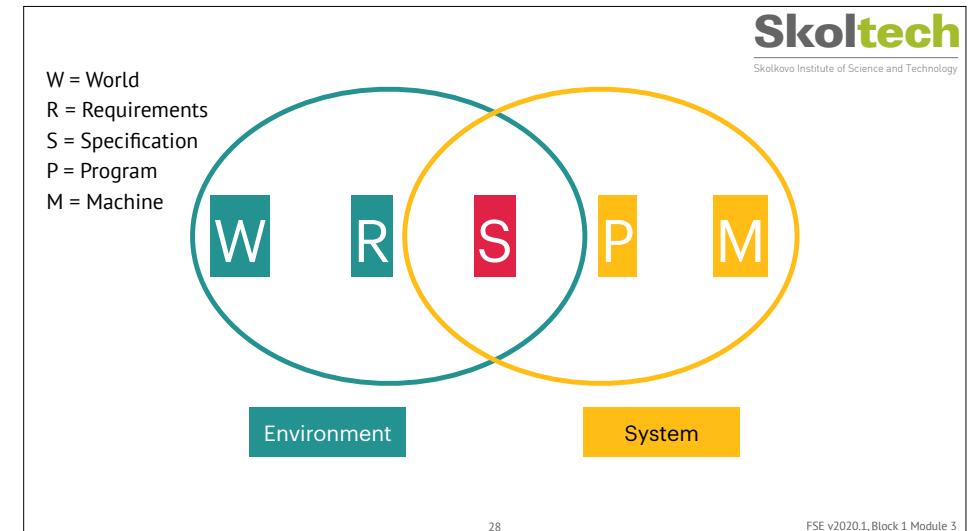
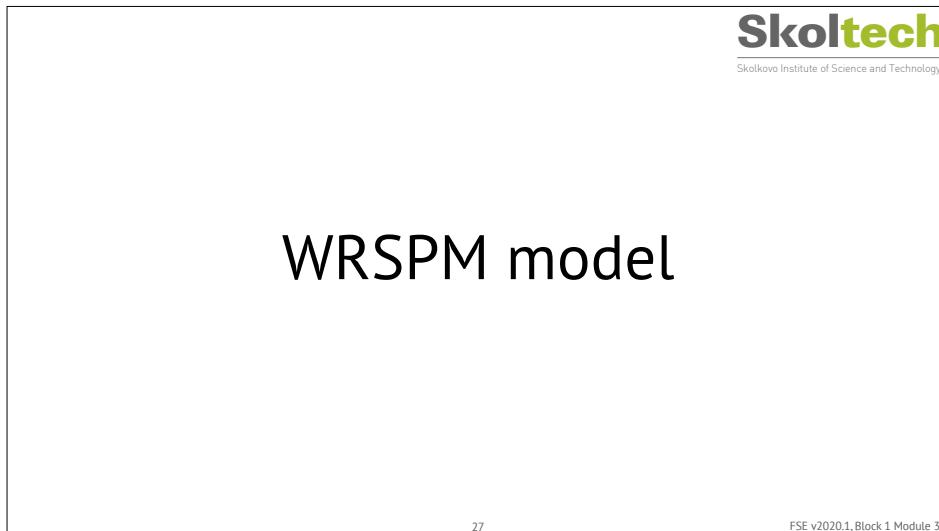
FSE v2020.1, Block 1 Module 3

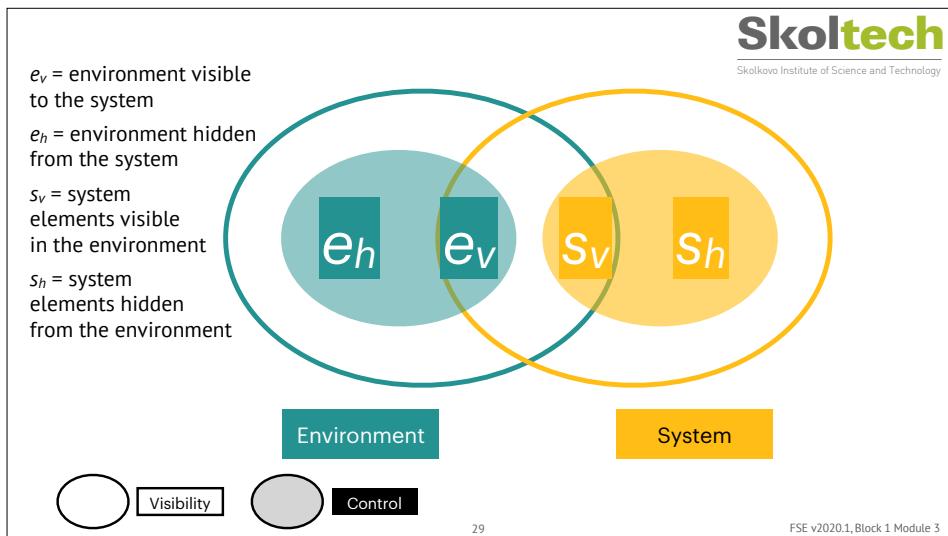


§3. Requirements models

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

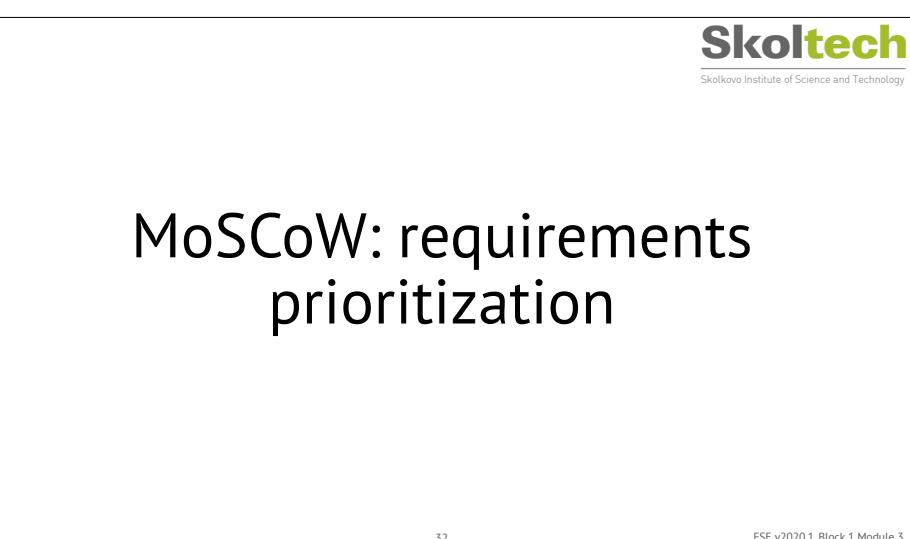
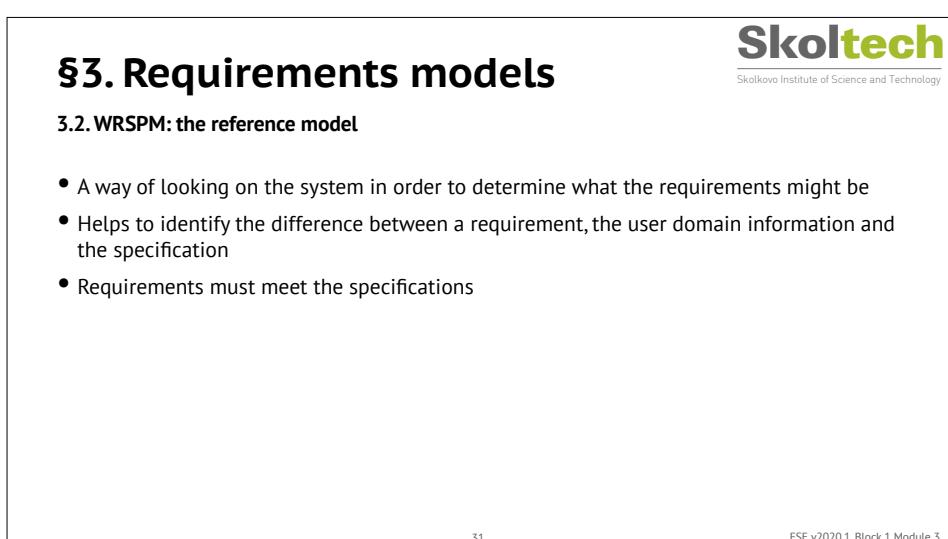


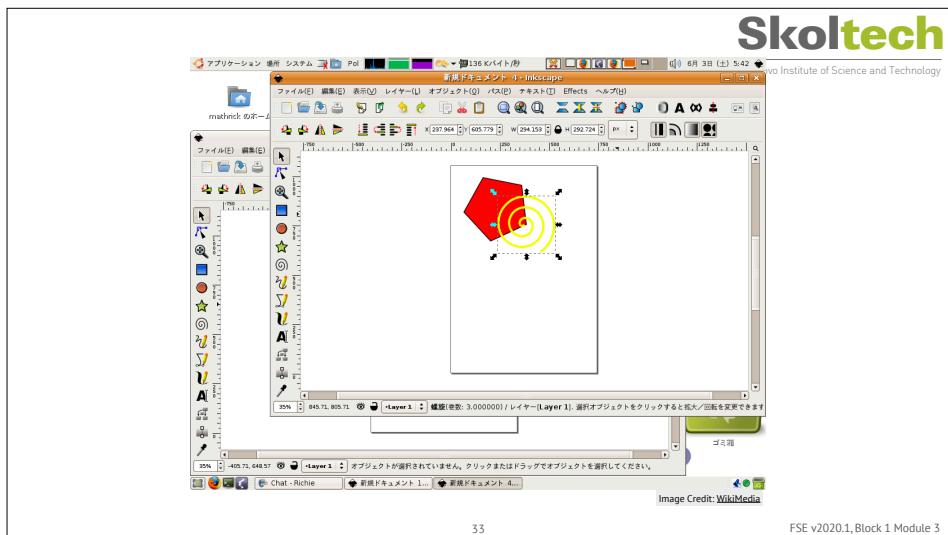


§3. Requirements models

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





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

34

FSE v2020.1, Block 1 Module 3

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]

35

FSE v2020.1, Block 1 Module 3

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

36

FSE v2020.1, Block 1 Module 3

§3. Requirements models

3.4. MoSCoW requirements prioritization model

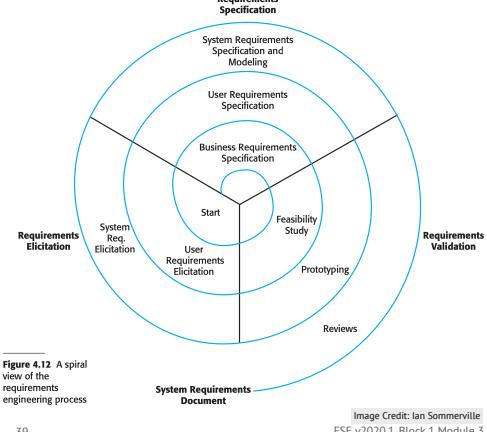
- Identify necessary and auxiliary features for a system to have
- *Must*: required features, must be included (initial release 1)
- *Should*: important features, should be included if possible (release 2)
- *Could*: desirable features, can be omitted if don't fit in the schedule (release 2)
- *Won't*: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



§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



Skoltech
Science and Technology

Image Credit: pxhere.com
FSE v2020.1, Block 1 Module 3

41

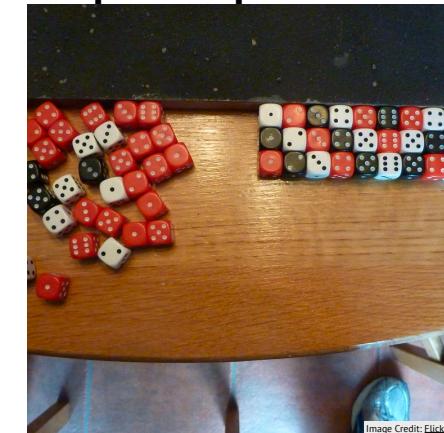
FSE v2020.1, Block 1 Module 3

Skoltech Science and Technology

§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



42

FSE v2020.1, Block 1 Module 3

§4. Requirements development process

2.1. Stages of developing requirements: Specification

Natural language sentences

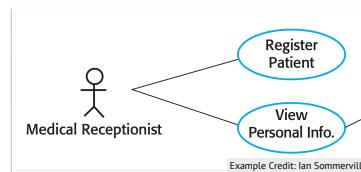
"3.2 The system shall measure the blood sugar and deliver insulin, if required, every 10 minutes."

Structured natural language

Insulin Pump/Control Software/SRS/3.3.2

Function	Description
Compute insulin dose: Safe sugar level.	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 (r_2), the previous two readings (r_0 and r_1).
Source	Current sugar reading from sensor. Other readings from memory.
Outputs	CompDose—the dose of insulin to be delivered.

Graphical notations



Design description languages

Mathematical specifications

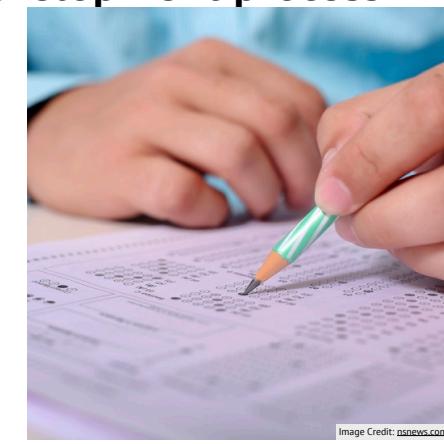
Example Credit: Ian Sommerville
FSE v2020.1, Block 1 Module 3

Skoltech Science and Technology

§4. Requirements development process

2.1. Stages of developing requirements: Validation

- Validity
- Consistency
- Completeness
- Realism
- Verifiability
- Requirement reviews
- Prototyping
- Test-case generation



44

Image Credit: psnews.com

FSE v2020.1, Block 1 Module 3

§4. Requirements development process

Summary

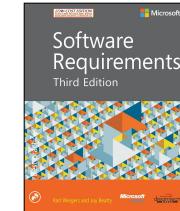
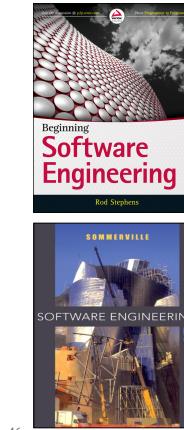
- Elicitation: discover the application domain, services the system should provide, the required performance of the system, hardware constraints,...
- Analysis: group subsets of requirements together and resolve conflicting requirements
- Specification: use structured approaches to write down the description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

45

FSE v2020.1, Block 1 Module 3

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*.



46

FSE v2020.1, Block 1 Module 3