Software Engineering

Lesson #03 - Lecture

Lesson #03 - Lecture

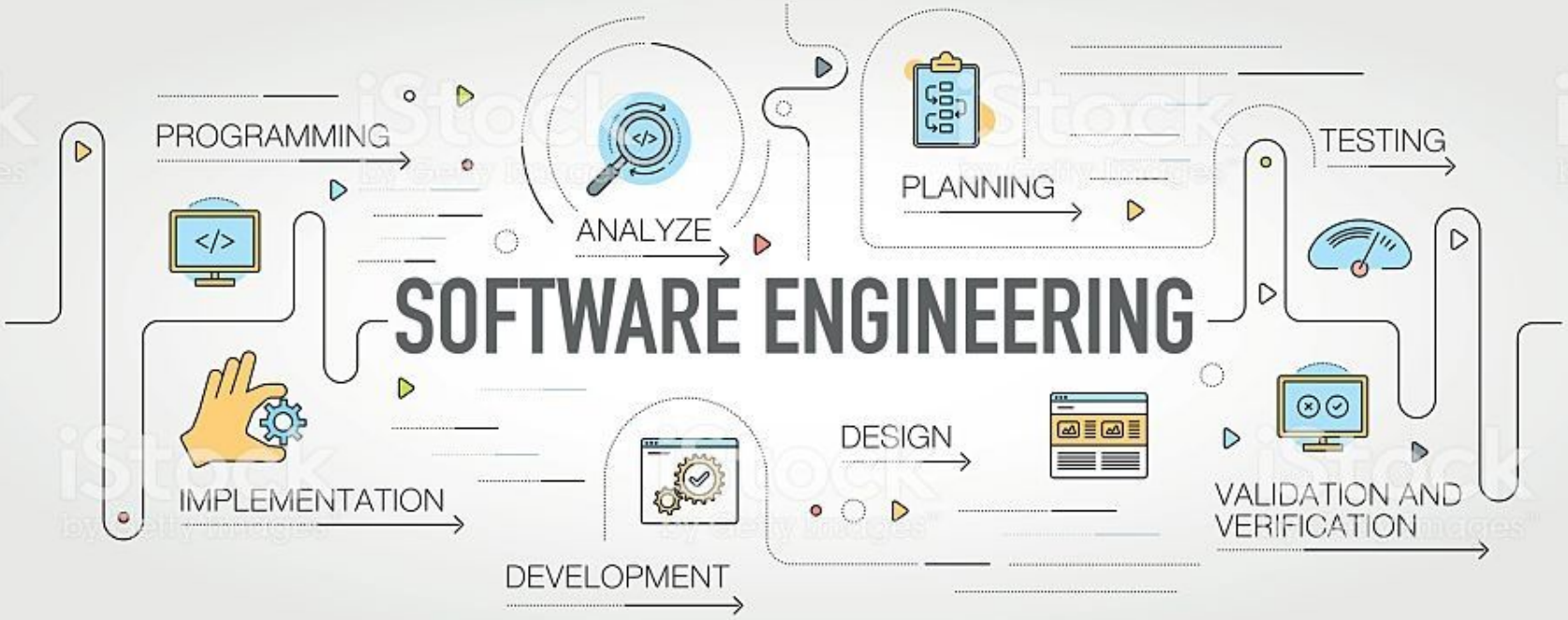
Your KBTU 202309 Software Engineering class information is updating …

Lesson #03 update is in progress

This will take around 2 hours to complete

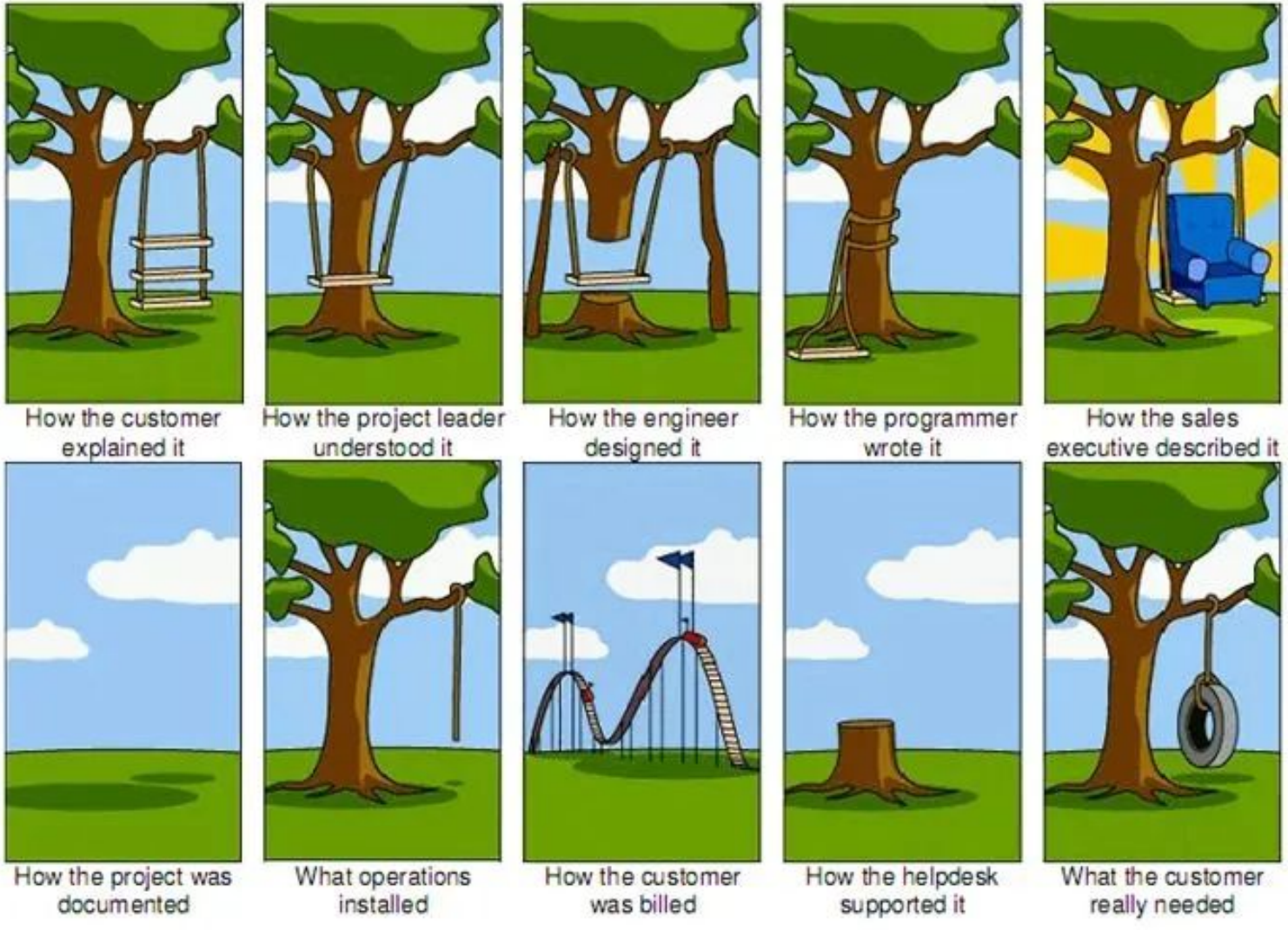
Please, don’t turn off your head

Requirements Engineering

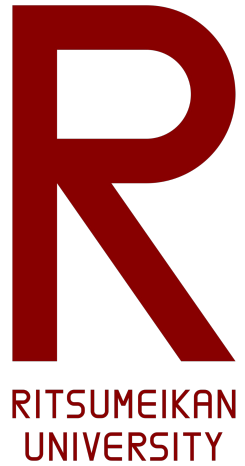


Requirements Engineering

Requirements Engineering


How the customer explained it
How the project leader understood it
How the engineer designed it
How the programmer wrote it
How the sales executive described it
How the project was documented
What operations installed
How the customer was billed
How the helpdesk supported it
What the customer really needed

Instructor's Master Degree Thesis



- University: Ritsumeikan University
- Thesis Name: Negotiating method among alternatives (sub-goals) in Goal-Oriented Requirements Analysis (2007-06-21)
- Professor: prof. Atsushi Ohnishi (淳　大西)
- Link: https://www.ieice.org/ken/paper/20070621eAVo/eng/
- Related Topics: Requirements Analysis /
  Requirements Elicitation /
  Goal-Oriented Requirements Analysis /
  Analytic Hierarchy Process /

Requirements Engineering

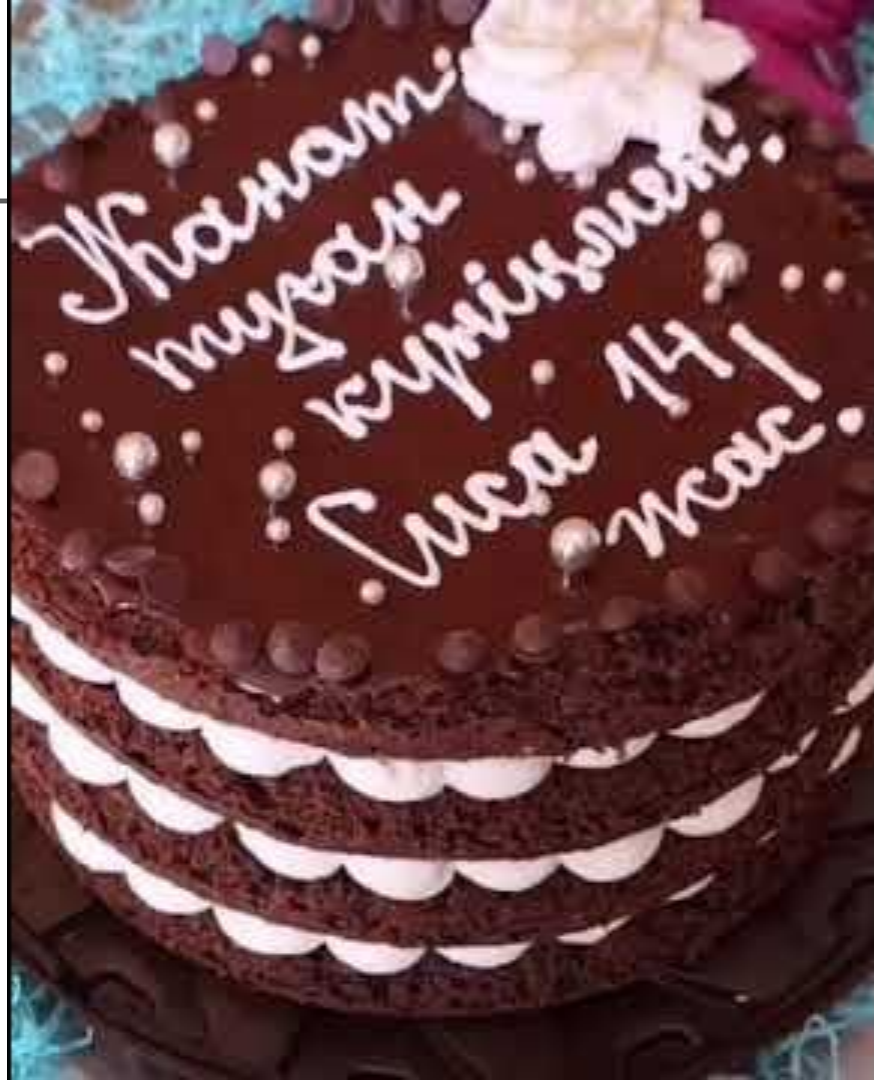
Pandemic period

Requirements Engineering

Zhanat, Happy Birthday!

If there is enough space, please write 14 year!

:-)



Requirements Engineering

The process of finding out, analyzing, documenting and checking these services and constraints is called requirements engineering

In some cases, a requirement is simply a high-level, abstract statement of a service that a system should provide or a constraint on a system

Requirements Engineering

An introduction to Requirements Engineering

https://www.youtube.com/watch?v=Ec0s0z5uXQ8&t=108s

Requirements Engineering

User Requirements

- In a natural language plus diagrams, no technical details
- Written for customers

System Requirements

- More detailed descriptions of the software system’s functions, services, and operational constraints
- Written for developers

Requirements Engineering

User and System Requirements - Georgia Tech - Software Development Process

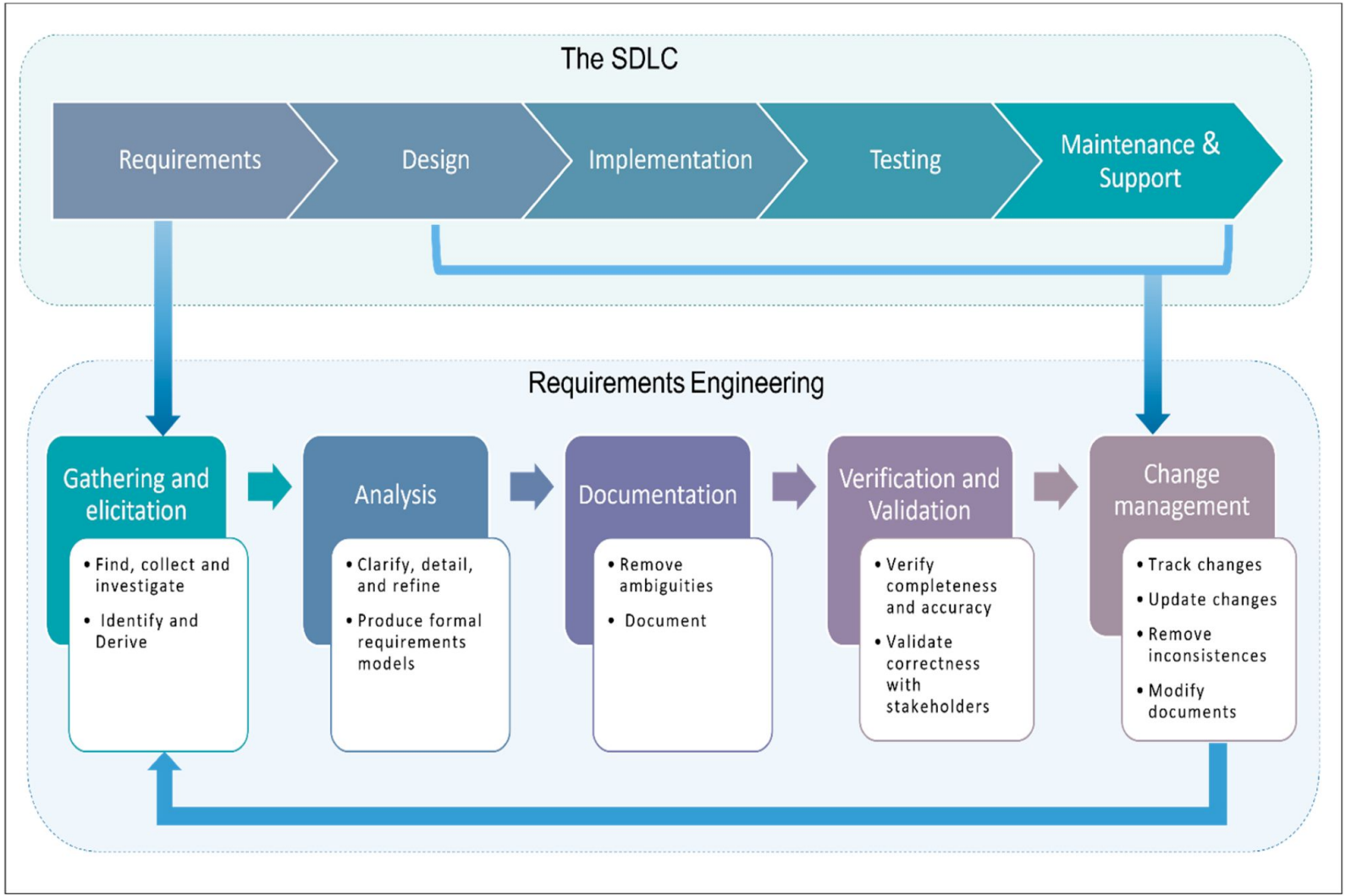
https://www.youtube.com/watch?v=vpNnZDwC_vs

Feasibility Study

Requirements engineering

Requirements Engineering - Georgia Tech - Software Development Process

https://www.youtube.com/watch?v=miuUIfpaC40&t=13s


The SDLC
Requirements
Design
Implementation
Testing
Maintenance & Support
Requirements Engineering
Gathering and elicitation
• Find, collect and investigate
• Identify and Derive
Analysis
• Clarify, detail, and refine
• Produce formal requirements models
Documentation
• Remove ambiguities
• Document
Verification and Validation
• Verify completeness and accuracy
• Validate correctness with stakeholders
Change management
• Track changes
• Update changes
• Remove inconsistences
• Modify documents

Bad Requirements

Good Requirements

Agenda: Lesson #03 - Software Engineering - Lecture

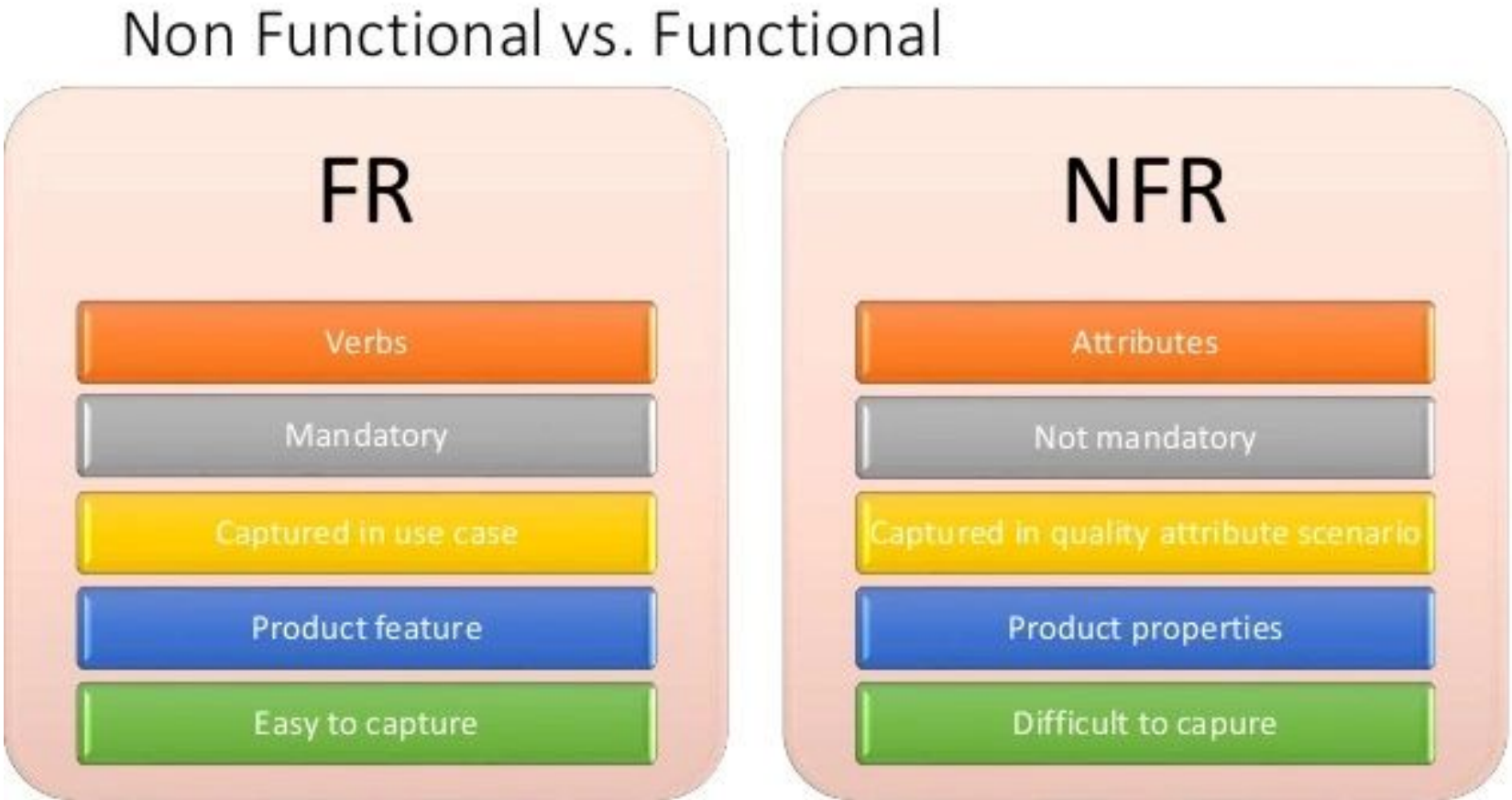
1. Functional and non-functional requirements
2. Requirements engineering processes
3. Requirements elicitation, specification and validation
4. Requirements change

Agenda: Lesson #03 - Software Engineering - Lecture

1. **Functional and non-functional requirements**
2. Requirements engineering processes
3. Requirements elicitation, specification and validation
4. Requirements change

Functional and non-functional requirements

Functional and non-functional requirements



FUNCTIONAL vs NONFUNCTIONAL REQUIREMENTS

	Functional requirements	Nonfunctional requirements
Objective	Describe what the product does	Describe how the product works
End result	Define product features	Define product properties
Focus	Focus on user requirements	Focus on user expectations
Documentation	Captured in use case	Captured as a quality attribute
Essentiality	They are mandatory	They are not mandatory, but desirable
Origin type	Usually defined by user	Usually defined by developers or other tech experts
Testing	Component, API, UI testing, etc. Tested before nonfunctional testing	Performance, usability, security testing, etc. Tested after functional testing
Types	External interface, authentication, authorization levels, business rules, etc.	Usability, reliability, scalability, performance, etc.

F&NF R.

Parameters	Functional Requirement	Non-Functional Requirements
Requirement	It is mandatory	It is non-mandatory
Capturing type	It is captured in use case	It is captured as a quality attribute
End-result	Product feature	Product properties
Capturing	Easy to capture	Hard to capture
Objective	Helps you verify the functionality of the software	Helps you to verify the performance of the software
Area of focus	Focuses on user requirement	Concentrates on the user`s expectation and experience
Documentation	Describe what the product does	Describes how the product works
Product Info	Product features	Product properties

FR & NFR

Functional and Nonfunctional Requirements - Georgia Tech - Software Development Process

https://www.youtube.com/watch?v=zCX-N1H8Vps&t=1s

Functional Requirements

These are statements of services the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations

In some cases, the functional requirements may also explicitly state what the system should not do

Functional Requirements

Functional Requirement No.	Function Requirement Description
FR 1	User should be able to enter Sales Data
FR 2	Sales Reports should be generated every 24 hours
FR 3	API interface to Invoice System

Functional Requirements

The functional requirements for a system describe what the system should do

When expressed as user requirements, functional requirements should be written in natural language so that system users and managers can understand them

Functional system requirements expand the user requirements and are written for system developers

Non-Functional Requirements

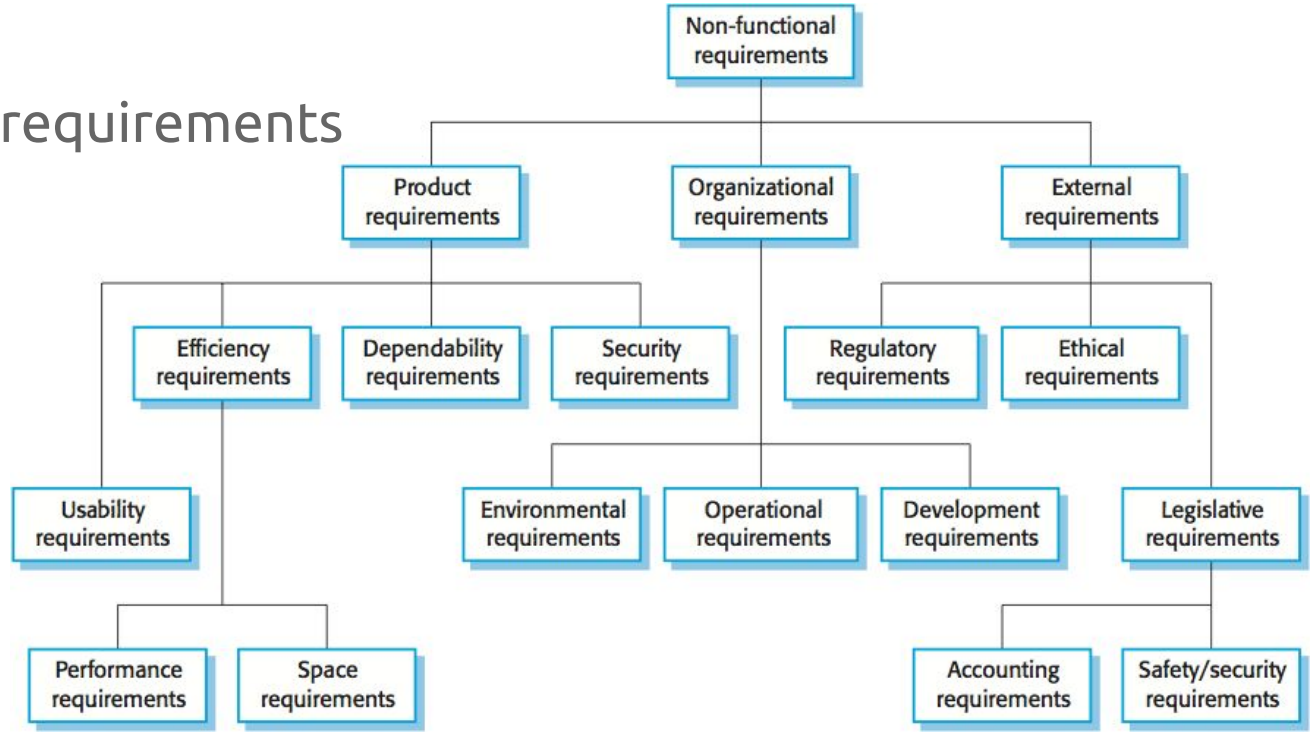
These are constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process, and constraints imposed by standards

Non-functional requirements often apply to the system as a whole rather than individual system features or services

They may relate to emergent system properties such as reliability, response time, and memory use

Functional and non-functional requirements

Non-Functional requirements



Non-Functional Requirements

Non-Functional Requirement Examples

- Response Time and Net Processing Time
- Capacity, Growth and Scalability
- Service Levels and Service Level Agreements
- Maintainability
- Security
- System Management
- Legal / Regulatory
- Application Architecture
- Data Architecture
- Technology Architecture
- Network Architecture
- Organizational Design and Change Management
- Training, Target Audience And Communications
- On-Going Support And Maintenance Requirements
- Usability Objectives
- Locations / Physical Environment
- Delivery, Deployment And Timing Requirements
- Development Standards
- Software Configuration Management

Non-Functional Requirements

Product requirements

- Requirements which specify that the delivered product must behave in a particular way e.g. execution speed, reliability, etc

Non-Functional Requirements

Organisational requirements

- Requirements which are a consequence of organisational policies and procedures e.g. process standards used, implementation requirements, etc

Non-Functional Requirements

External requirements

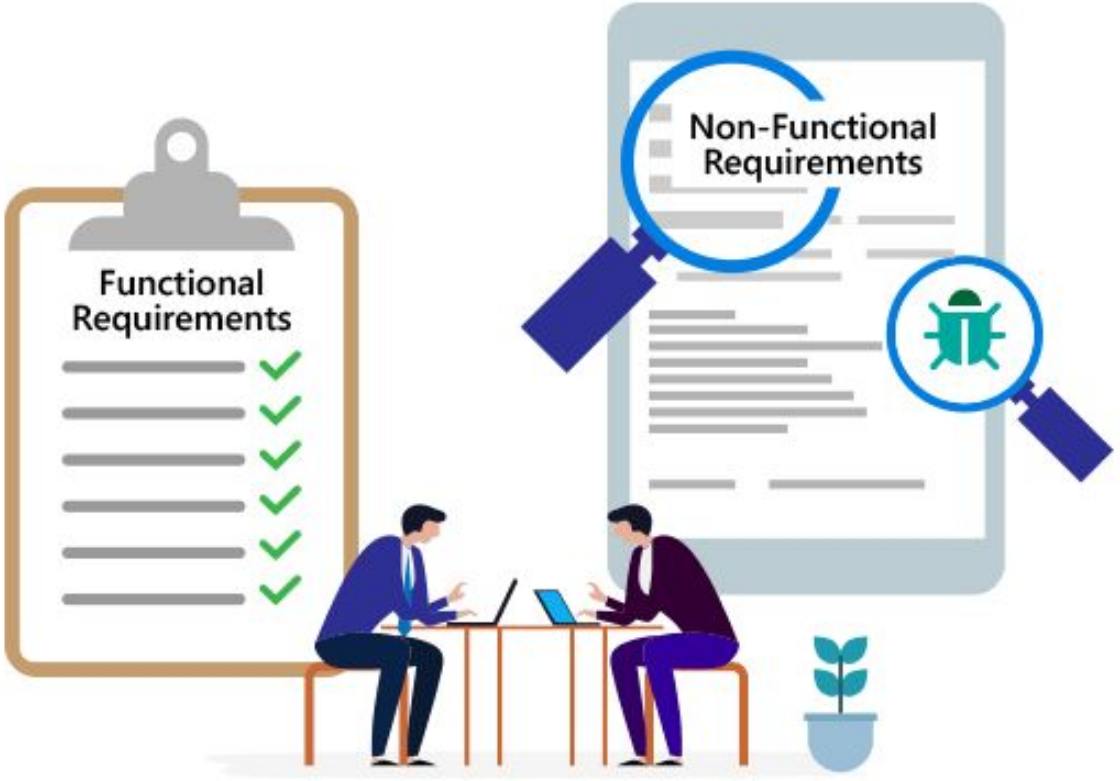
- Requirements which arise from factors which are external to the system and its development process e.g. interoperability requirements, legislative requirements, etc

Requirements engineering

Requirements Engineering lecture 1: Overview

https://www.youtube.com/watch?v=qENBiYaAXNE&t=331s

Functional and non-functional requirements



Functional and non-functional requirements

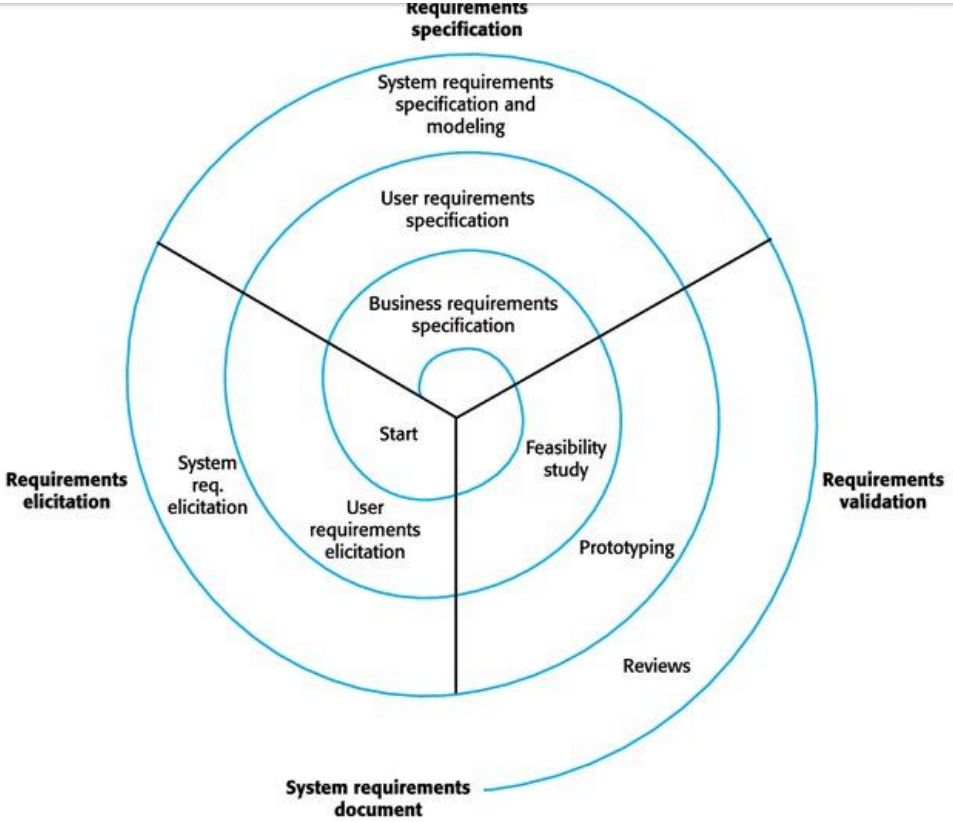
Types of software engineering requirements

FUNCTIONAL	NONFUNCTIONAL
Detail product features	Detail product properties
Describe the work that is done	Describe the character of the work
Describe the user actions	Describe the user experience
Characterized by verbs	Characterized by adjectives

Agenda: Lesson #03 - Software Engineering - Lecture

1. Functional and non-functional requirements
2. **Requirements engineering processes**
3. Requirements elicitation, specification and validation
4. Requirements change

Requirements engineering processes



Requirements engineering processes

Requirements engineering processes

https://www.youtube.com/watch?v=GSe4xIy-iBE&t=63s

Agenda: Lesson #03 - Software Engineering - Lecture

1. Functional and non-functional requirements
2. Requirements engineering processes
3. **Requirements elicitation, specification and validation**
4. Requirements change

Requirements elicitation

The aims of the requirements elicitation process are to understand the work that stakeholders do and how they might use a new system to help support that work

It is, sometimes, called requirements discovery

Requirements elicitation

Stages include:

- Requirements discovery
- Requirements classification and organization
- Requirements prioritization and negotiation
- Requirements specification

Requirements elicitation

- Interviewing

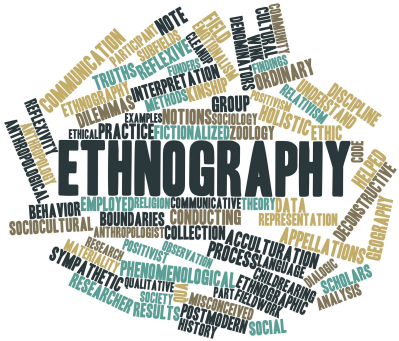
Requirements elicitation

- Interviewing

Requirements elicitation

Ethnography

- An observational technique that can be used to understand operational processes and help derive requirements for software to support these processes



Stories and scenarios

Scenarios and user stories are real-life examples of how a system can be used

Stories and scenarios are a description of how a system may be used for a particular task

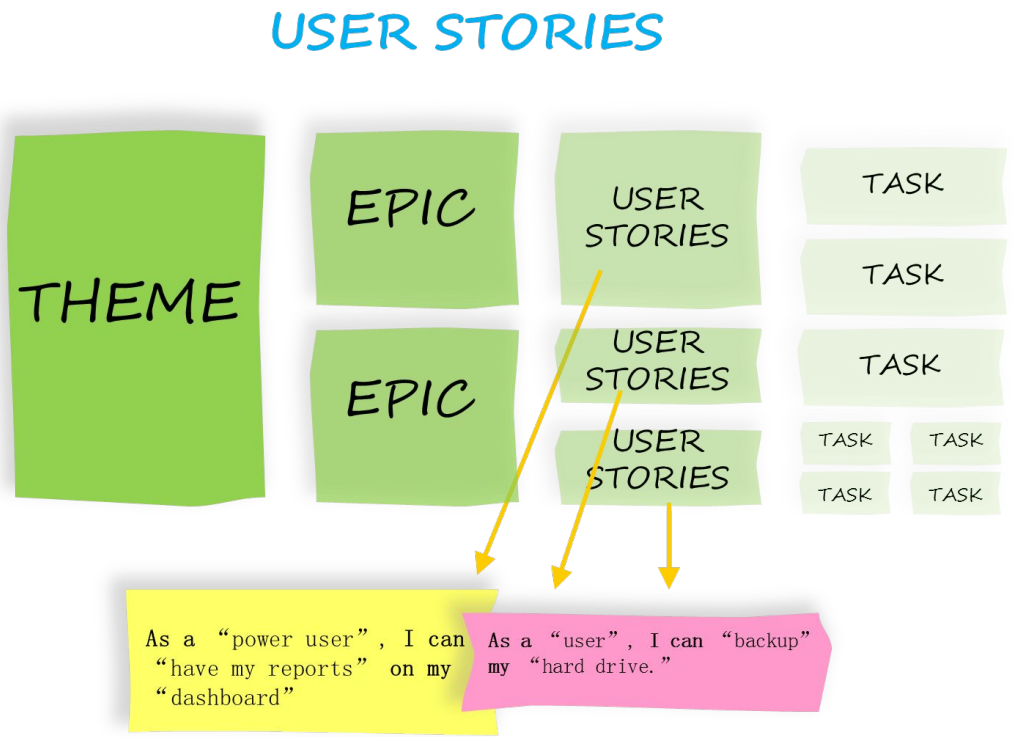
Stories and scenarios

User stories

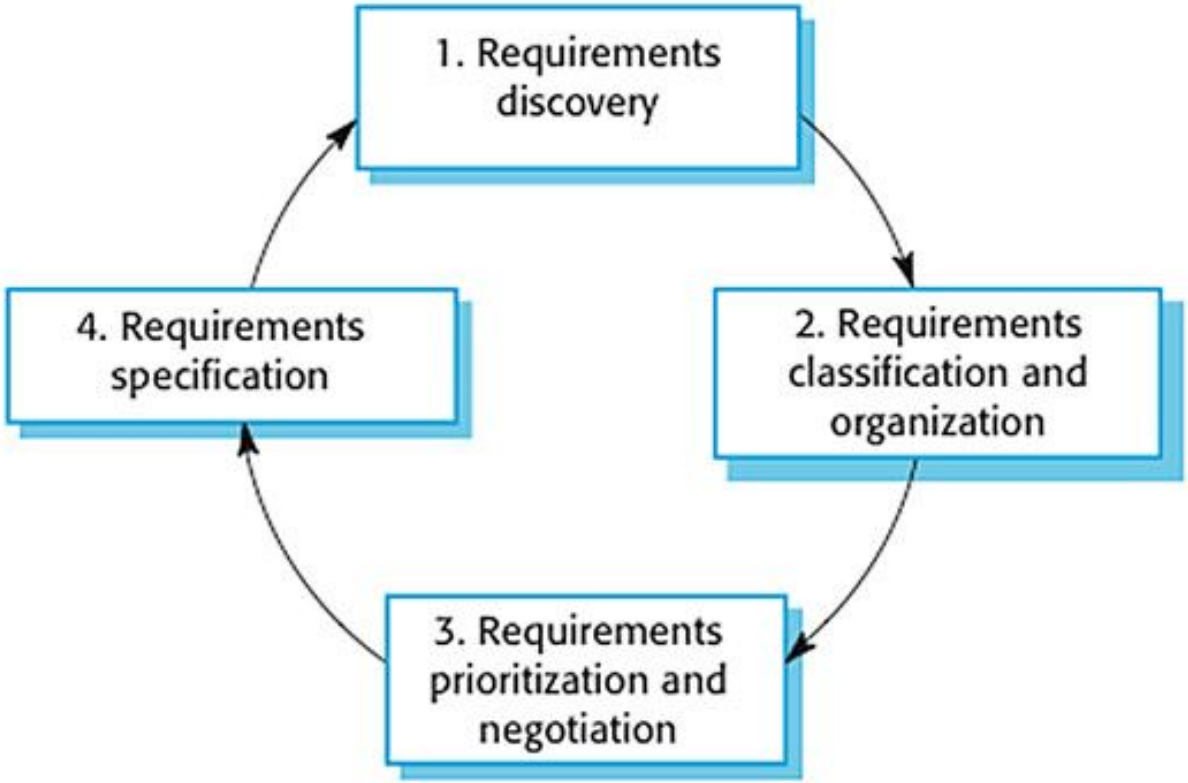
https://www.youtube.com/watch?v=UpYdVSV3dG8&t=42s

Stories and scenarios

User stories



Requirements elicitation



Agenda: Lesson #03 - Software Engineering - Lecture

1	Functional and non-functional requirements
2	Requirements engineering processes
3	Requirements elicitation
4	**Requirements specification**
5	Requirements validation
6	Requirements change

Requirements specification

The process of writing down the user and system requirements in a requirements document

Requirements specification

Stakeholders interpret the requirements in different ways, and there are often inherent conflicts and inconsistencies in the requirements

User requirements should be written in natural language, with simple tables, forms, and intuitive diagrams

System requirements are expanded versions of the user requirements that software engineers use as the starting point for the system designing including software design, system architecture

Requirements specification

Natural language specification

- Natural language will continue to be the most widely used way of specifying system and software requirements

Requirements specification

Structured specifications

- Structured natural language is a way of writing system requirements where requirements are written in a standard way rather than as free-form text

Requirements specification

Use cases

- Use cases are a way of describing interactions between users and a system using a graphical model and structured text

Requirements specification

The software requirements document

- The software requirements document (sometimes called the software requirements specification or SRS) is an official statement of what the system developers should implement

- It may include both the user requirements for a system and a detailed specification of the system requirements

Requirements specification

The software requirements document

- Rather than a formal document, agile approaches often collect user requirements incrementally and write these on cards or whiteboards as short user stories

- The user then prioritizes these stories for implementation in the next increment of the system

Requirements specification

The software requirements document

- When the system is to be developed by a separate company (e.g., through outsourcing), the system specifications need to be detailed and precise

- If an in-house, iterative development process is used, the requirements document can be less detailed

Requirements specification

The process of checking that requirements define the system that the customer really wants

It overlaps with elicitation and analysis, as it is concerned with finding problems with the requirements

Agenda: Lesson #03 - Software Engineering - Lecture

1. Functional and non-functional requirements
2. Requirements engineering processes
3. Requirements elicitation, specification and validation
4. **Requirements change**

Requirements change

The requirements for large software systems are always changing

One reason for the frequent changes is that these systems are often developed to address “wicked” problems—problems that cannot be completely defined (Rittel and Webber 1973)

Requirements change

Once a system has been installed and is regularly used, new requirements inevitably emerge

This is partly a consequence of errors and omissions in the original requirements that have to be corrected

Requirements management planning

Requirements management planning is concerned with establishing how a set of evolving requirements will be managed

Requirements management needs automated support, and the software tools for this should be chosen during the planning phase

Requirements change management

Requirements change management should be applied to all proposed changes to a system’s requirements after the requirements document has been approved

Change management is essential because you need to decide if the benefits of implementing new requirements are justified by the costs of implementation

Requirements engineering challenges

Requirements engineering challenges

https://www.youtube.com/watch?v=bK-y0CaGkhU


COMING SOON

System modeling

- Introduction & Context models
- Interaction models & Structural models
- Behavioral models
- Model-driven engineering

Agenda: Lesson #03 - Software Engineering - Lecture

Q & A