CP317 Software Engineering

week 2-1: Requirement gathering, part-1

Shaun Gao, Ph.D., P.Eng.

Agenda

- Review week 1 topics
- Introduction
- The reasons why requirements are important
- The characteristics of good requirements
- The **MOSCOW method** for prioritizing requirements
- Requirements categories
  - Functional requirements vs. non-functional requirements
- **FURPS and FURPS+ methods** for categorizing requirements
- Summary

Review week 1-2

- Documentation
  - Document change control
  - Types of documentations
- Project management
  - Concept
- Project management tools
  - PERT charts
  - Critical path methods
  - Gantt charts
- Software Cost Estimation Models
  - COCOMO, Static Single Variable Model, Static Multi-Variable Model
- Risk management
  - Concept

Introduction

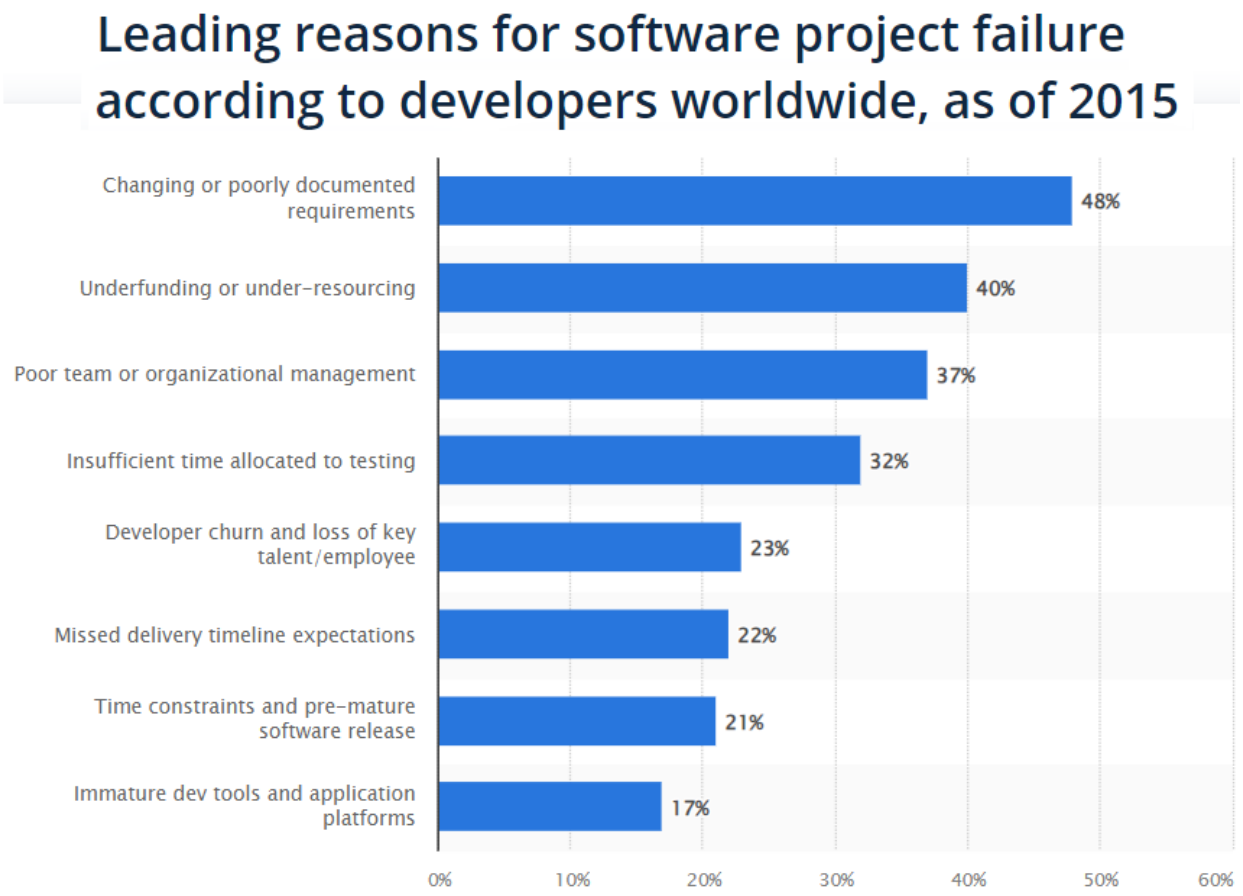
- Requirements
  - Definition: a **requirement** is a singular documented physical or functional need that particular design, product or process aims to satisfy.
  - In software engineering, requirements are documented customer needs or features that a computer system aims to satisfy.

Introduction – cont.

- Requirement gathering is the most important part of SE.
- Why requirements are important?
  - To understand what we are going to do
    - Accountability and responsibility
  - We build software systems for customers
    - Reputation and integrity
  - Getting accurate requirements is crucial
    - Earnest and meticulous and no assumption

Introduction – cont.

- Why requirements are important?
  - https://www.statista.com



10 Reasons why software projects fail

1. Unclear project requirements
2. Wrong tech stack
3. Lack of communication
4. Underestimated timeline
5. Scope creep
6. Gaps in developer skill set & project requirements
7. Poor project management
8. Unrealistic expectations
9. No end-user involvement
10. Lack of proper testing

https://solveit.dev/blog/why-software-projects-fail (2023)

Think – Pair – Share

- what skills are required for understanding requirements in SE?

Characters of good requirements

- Use of Correct Terms
  - Shall = requirement
  - Will = facts or declaration of purpose
  - Should = goal
- Example System Requirements
  - The system shall operate at a power level of 12 voltage
  - The software shall acquire data from the sensors that are located at the motor
  - The data structure shall contain a linked list.

Characters of good requirements –cont.

- Understandable – we know what's needed
- Correct / Precise – it will do what's needed
- Unambiguous – everyone interprets it in the same way
- Complete – it can be fully documented
- Consistent – it is not contradictory with other requirements
- Interoperable – dependencies among requirements are known
- Verifiable / Valid – it can be tested
- Singular –it only appears once
- Traceable – it can be traced through design, test, and delivery
- Prioritized – relative importance is understood
- Achievable – we have the capability and the resources to do it

How to prioritize requirements

- MOSCOW method

Why do requirements need prioritization

- Requirements prioritization

Why prioritize requirements?

- Priorities help you
  - concentrate on the most important user and customer requirements
  - focus the development effort
  - manage projects more effectively
  - plan for staged deliveries
- It can also help you
  - make acceptable trade-offs among conflicting goals
  - allocate resources

Requirements categories

- Business requirements
  - Business requirements lay out the project's high-level goals. They explain what the customer hopes to achieve with the project.
- Example:
  - Automated Teller Machine (ATM)
    - cash withdraw
    - deposit
    - a transfer of money
    - balance inquiry
    - ……

Requirements categories – cont.

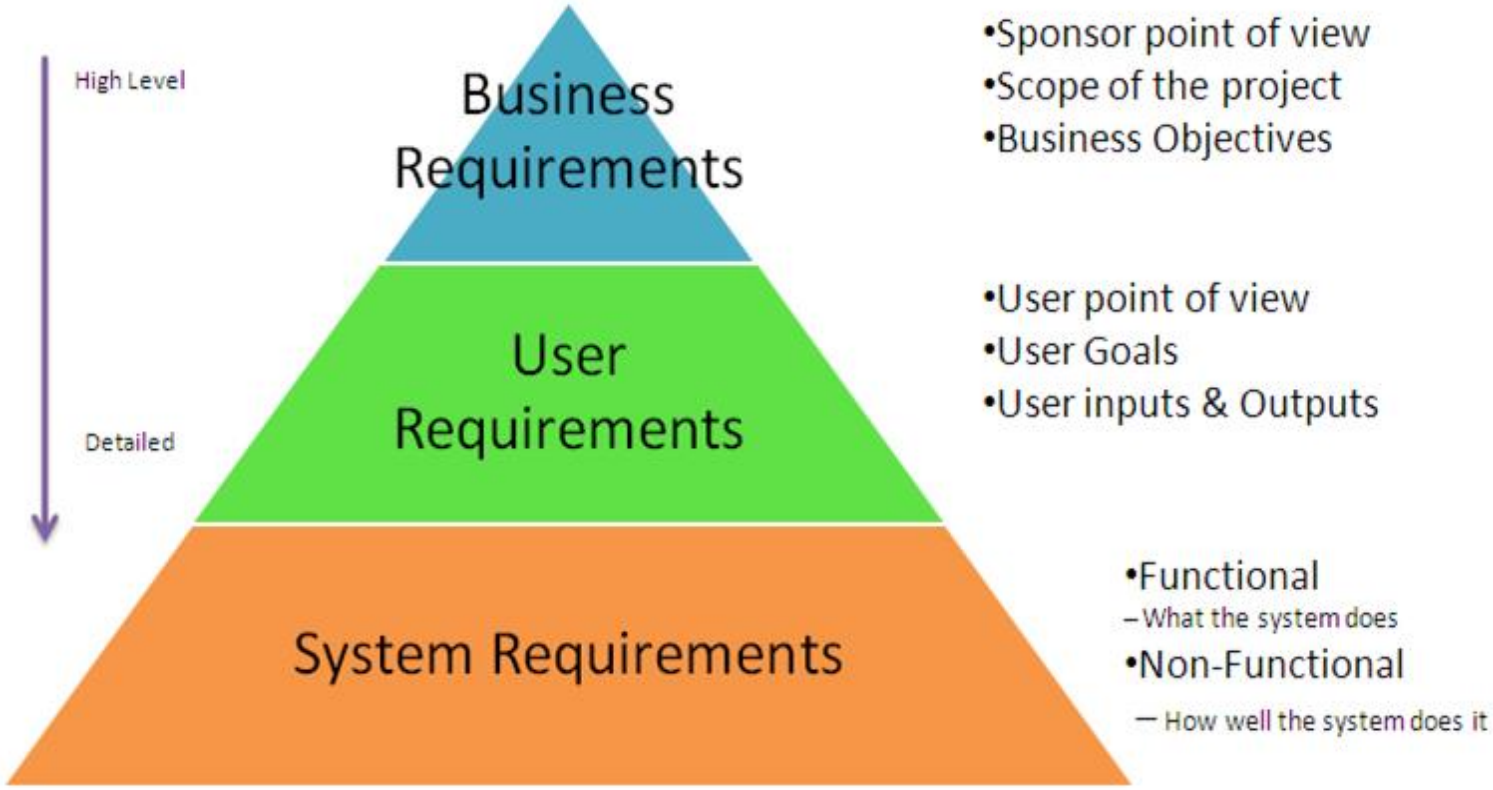
- User requirements
  - User requirements describe how the software product of the project will be used by the end users.
- Example:
  - ATM
    - required to insert an ATM card
    - enter a personal identification number (PIN)
    - amount selection
    - ……

Requirements categories – cont.

- System requirements
  - System requirements are the configuration that a system must have in order for a hardware or software application to run smoothly and efficiently such that to achieve business requirements and user requirements.
- Example:
  - ATM
    - Touch screen or keypad
    - Computer languages, python, C++, C#,…
    - Physical measurements – size
    - ……

Requirements categories – cont.

- Requirements



Requirements categories – cont.

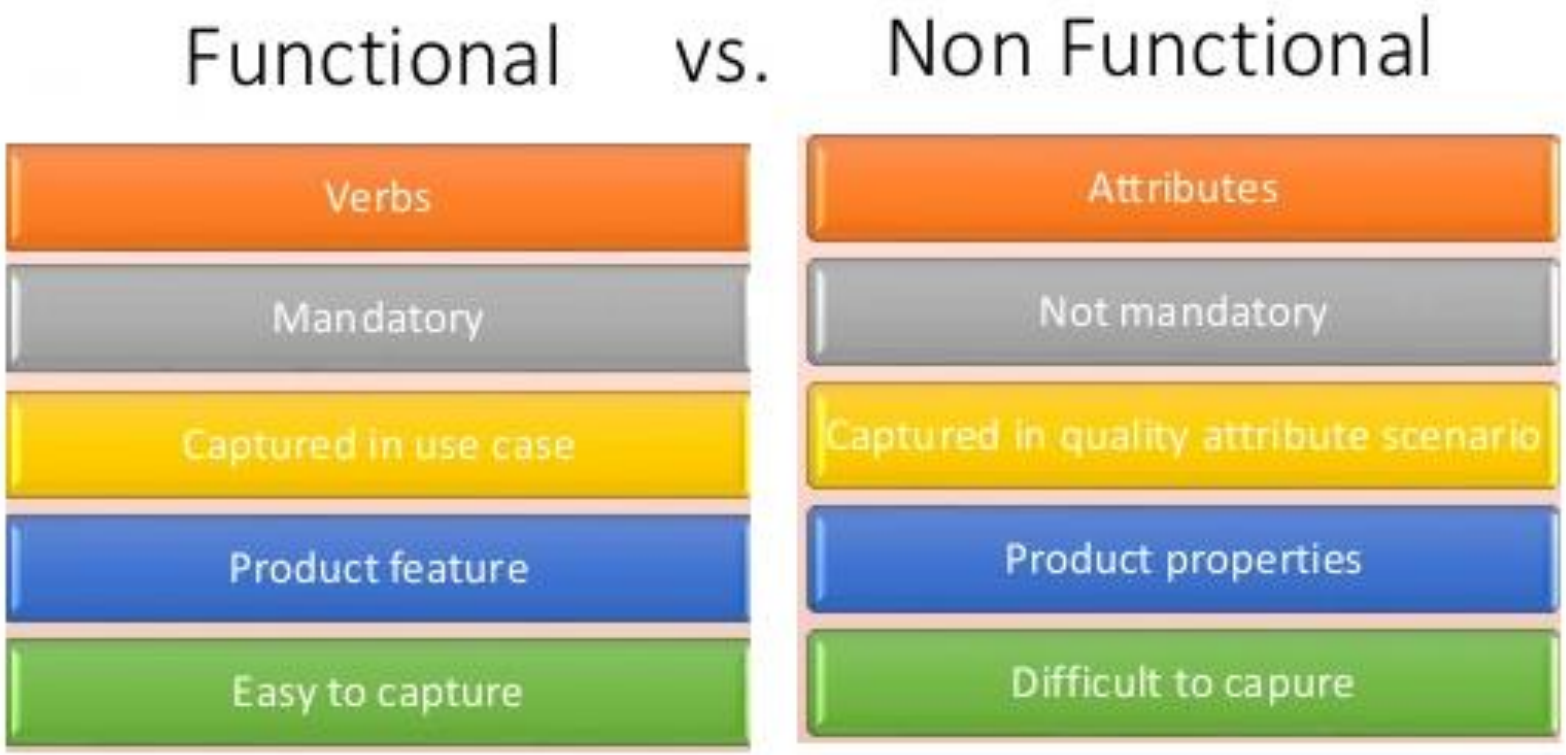
- Functional requirements
  - Functional requirements are detailed statements of the project's desired capabilities.
  - Examples of functional requirements
    - Business Rules.
    - Transaction corrections, adjustments and cancellations.
    - Administrative functions.
    - Authentication.
    - Authorization levels.
    - Audit Tracking.
    - External Interfaces.
    - Certification Requirements.

Requirements categories – cont.

- Non-functional requirements
  - Non-functional requirements are statements about the quality of the product's behavior or constraints on how it produces a desired results.
  - Examples – based on IEEE-Std 830 - 1993
    - Performance requirements
    - Operational requirements
    - Resource requirements
    - Verification requirements
    - Acceptance requirements
    - Documentation requirements
    - Security requirements
    - Portability requirements

Requirements categories – cont.

- Functional requirements vs. Non-functional requirements



FURPS

- FURPS is an acronym for categorizing the system requirements.
  - Functionality requirement
  - Usability requirement
  - Reliability requirement
  - Performance requirement
  - Supportability requirement

  - Developed by HP.

TABLE II. THE CONTENT OF FURPS MODEL

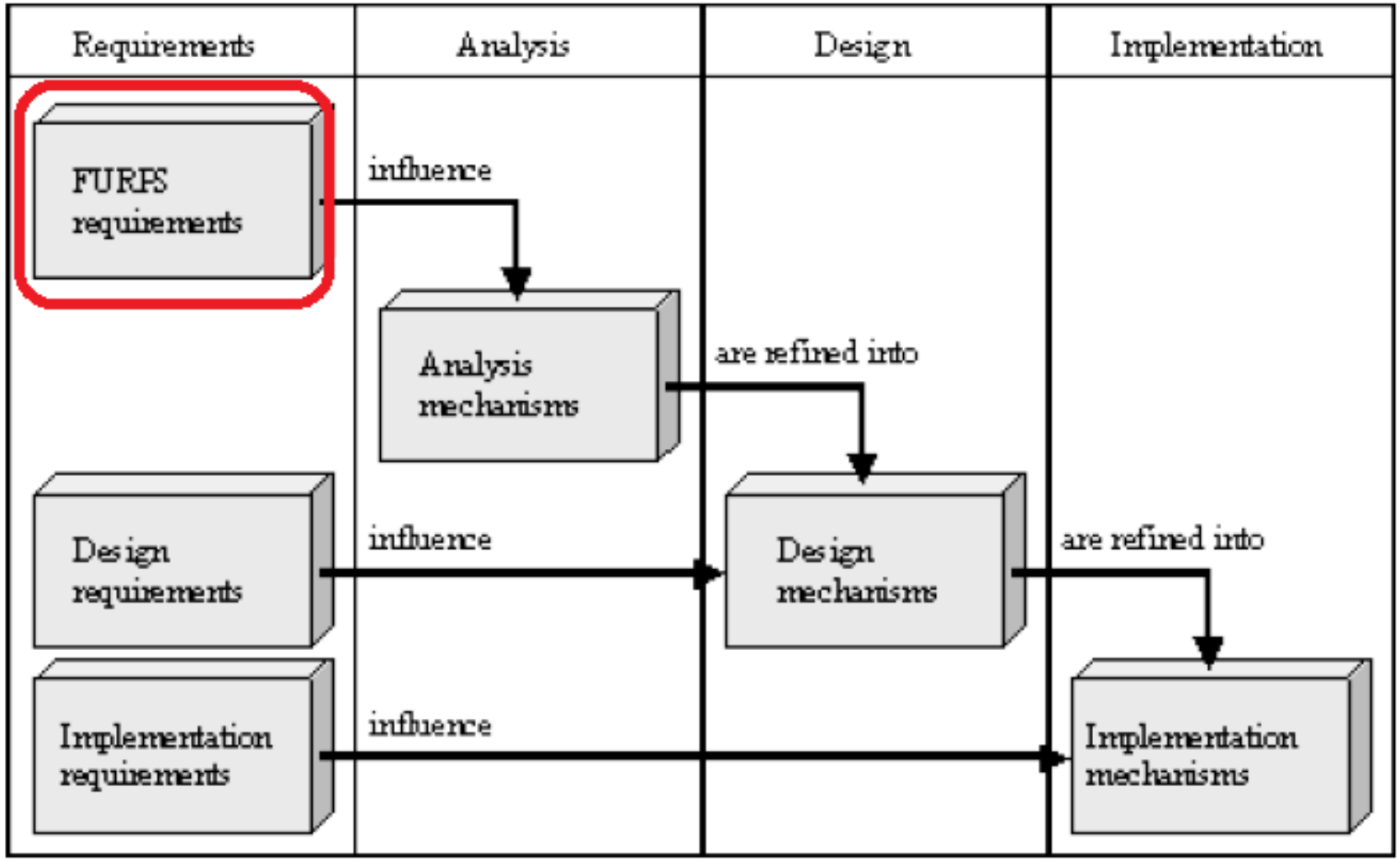
Characteristics	Description
Functionality	Include feature sets, capabilities, and security.
Usability	Human Factors, overall aesthetics, consistency, and documentation
Reliability	Frequency and severity of failure, recoverability, predictability, accuracy, and mean time between failures (MTBF).
Performance	Processing speed, response time, resource consumption, throughput and efficiency.
Supportability	Testability, extensibility, adaptability, maintainability, compatibility, configurability, serviceability, installability, and localizability.

FURPS+

- FURPS+
  - Functionality requirement
  - Usability requirement
  - Reliability requirement
  - Performance requirement
  - Supportability requirement
  - *Design constraints*
  - *Implementation requirements*
  - *Interface requirements*
  - *Physical requirements*

Requirement categories	FURPS + categories	Example requirements
Functional	Functions	Business rules and processes
Nonfunctional	Usability Reliability Performance Security + Design constraints Implementation Interface Physical Support	User interface, ease of use Failure rate, recovery methods Response time, throughput Access controls, encryption Hardware and support software Development tools, protocols Data interchange formats Size, weight, power consumption Installation and updates

An Example of FURPS



Summary

- Requirements
- The reasons why requirements are important
- The characteristics of good requirements
- The MOSCOW method for prioritizing requirements
- Requirements categories
  - Functional vs. non-functional requirements
- FURPS and FURPS+ methods for categorizing requirements

Announcement

- Group for project.
    - Please find a group as soon as possible by end of September 2024.
    - Please let me know if you need help.