Announcements

- Quiz 2 will release tomorrow
  - Single attempt quiz with Honorlock; Closed book; Duration of 10 minutes
  - Syllabus: Everything upto Requirements Engineering
- Project 1 Progress Report is Due on 9/12
- You should now start working on the Project 1 Requirements Assignment
- REST assignment was out last Thursday - This is an individual assignment
- Extra Credit Opportunity in today's class

CS3300 Introduction to Software Engineering

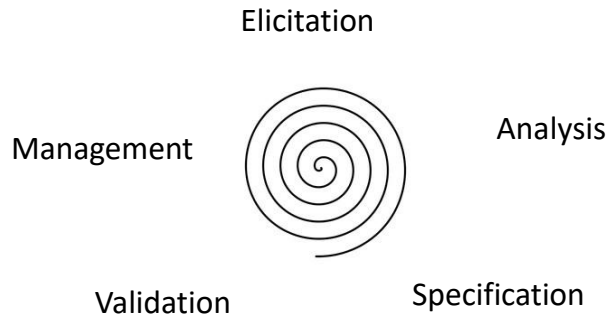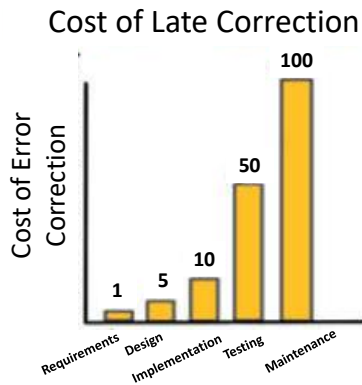# Lecture 07: Requirements Engineering

Dr. Nimisha Roy ▸ nroy9@gatech.edu
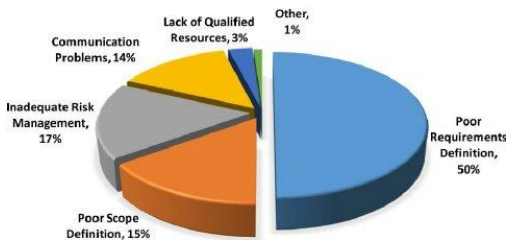
Slides adapted from Alessandro Orso

Requirements Engineering (RE)

RE is the process of establishing the needs of stakeholders that are to be solved by software
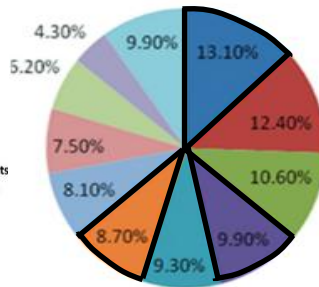
Cost of Late Correction

Cost of Error Correction

| Requirements | Design | Implementation | Testing | Maintenance |

1 5 10 50 100

Elicitation

Management

Analysis

Validation

Specification

Software Failures due to RE

Role of Requirements in Software Project Failures (Agile)

Source: Abdou, T., Kamthan, P., & Shahmir, N. (2014). User Stories for Agile Business: INVEST, Carefully!. Social Media and Publicity, 141.

- Incomplete Requirements
- Lack of user involvement
- Lack of Resources
- Unrealistic Expectations
- Lack of Executive Support
- Changing Requirements
- Lack of planning
- Not needed any longer
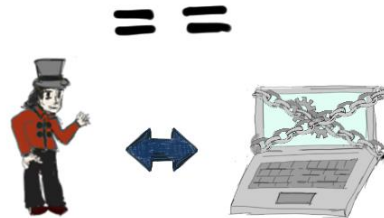- Lack of IT Management
- Technology Illiteracy
- Others

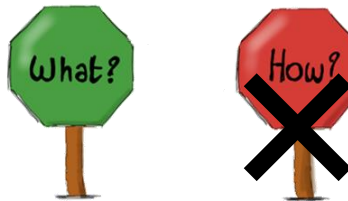**Role of Requirements in Software Project Failures.**

Source: Hussain, A., Mkpojiogu, E. O., & Kamal, F. M. (2016). The role of requirements in the success or failure of software projects. International Review of Management and Marketing, 6(7S), 306-311

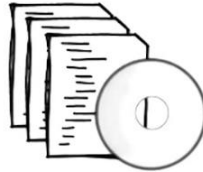# Requirements Engineering (RE)

=> Software Requirements Specification (SRS)

understanding the services that the customer requires from the software system. & the constraints under which the system can be developed.

Software Intensive Systems

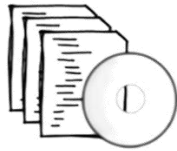Software

Software Intensive System = Software + Hardware + Context

abstract description of a set of computations that becomes concrete, and therefore useful, only when we run the software on some hardware, and that, in the context of some human activity that it can support.

Software Quality
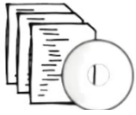
Software runs on some hardware and is developed for a purpose that is related to human activities

Quality ≠ f( )   Quality = f(,)

RE is mostly about identifying the purpose

What system will be used. Low quality not only if it doesn't work well, also when it does not fulfill its purpose

Identifying Purpose = Defining Requirements

Extremely Hard Task

- Sheer Complexity of the purpose/requirements
- "Often, people don't know what they want until you show it to them" – Steve Jobs
- Changing requirements
- Multiple stakeholders with conflicting requirements

Hard to extract this purpose from humans and make it explicit. Quote by steve jobs

Software Quality Example and Relevance of RE: Healthcare Monitoring System in an ICU

Software: A real-time patient monitoring system that tracks vital signs like heart rate, blood pressure, and oxygen saturation, providing alerts to healthcare professionals if any parameter deviates from normal ranges.

Hardware: The system relies on specialized medical-grade sensors, servers, and display monitors that can handle real-time data processing and provide uninterrupted operation in the ICU environment.

Context: The ICU is a high-stakes medical environment where patients require constant monitoring, and any delay or inaccuracy in the system's alerts can lead to severe health consequences or fatalities. The system must be able to operate 24/7 with minimal downtime, maintain data integrity, and integrate with other hospital information systems.

Software Quality Example and Relevance of RE: Healthcare Monitoring System in an ICU

How Quality Depends on Software, Hardware, and Context:

Software Dependency: The monitoring software must be designed to handle large amounts of real-time data efficiently, generate accurate alerts, and provide a user-friendly interface for healthcare staff. It must also comply with medical regulations and data privacy laws.

Hardware Dependency: The sensors and monitors must be highly accurate, reliable, and capable of operating continuously without failure. The hardware should have the necessary processing power and connectivity to support the software's real-time data processing requirements.

Context Dependency: In the context of an ICU, the system must meet stringent requirements for uptime (e.g., less than 0.1% downtime), response time (e.g., alerts should be generated within 1 second of detecting an abnormal reading), and compatibility with existing medical infrastructure. The system must be resilient to power outages, electromagnetic interference from other medical devices, and the physical constraints of the ICU environment.

Software Quality Example and Relevance of RE: Healthcare Monitoring System in an ICU

Role of Requirements Engineering:

•**Understanding Context Constraints:** Requirements Engineering helps identify and define the specific constraints of the ICU environment, such as the need for minimal downtime, fast response times, data integration with existing hospital systems, and compliance with health regulations.

•**Defining Quality Requirements:** It ensures that both functional (e.g., real-time monitoring, alert generation) and non-functional requirements (e.g., system reliability, response time, compliance) are clearly defined, taking into account the software, hardware, and contextual needs.

•**Managing Trade-offs:** Requirements Engineering aids in understanding the trade-offs between different quality attributes, such as balancing high accuracy and low latency of alerts against hardware limitations or cost constraints.

Definition of Requirements Engineering

Requirements Engineering (RE) is a set of activities concerned with identifying and communicating the purpose of software-intensive system, and the context in which it will be used. Hence, RE acts as the bridge between the real-world needs of users, customers, and other constituencies affected by a software system, and the capabilities and opportunities afforded by software – intensive technologies.

Communication is as important as analysis

Quality means fitness-for-purpose. Cannot say anything about quality unless you understand the purpose
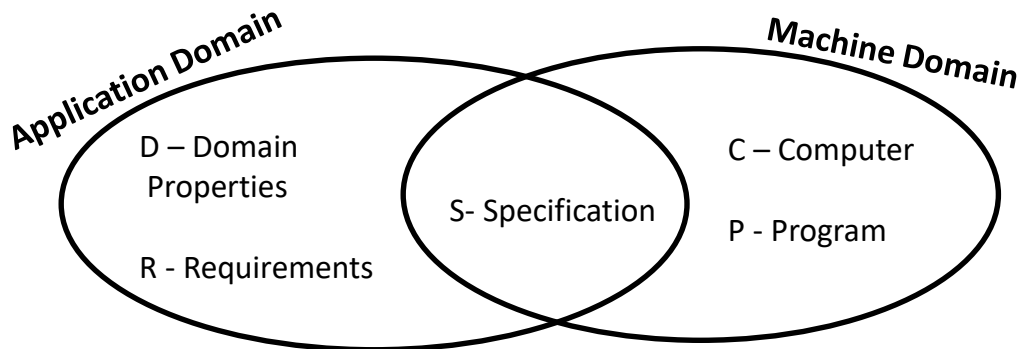
Needed to identify all stakeholders – not just the customer or user

Designers need to know how and where the system will be used

Requirements are partly about what is needed…

… and partly about what is possible

RE is a shared phenomenon between domains

Application Domain

Machine Domain

D – Domain Properties

R - Requirements

S- Specification

C – Computer

P - Program

- Machine Domain - hardware/OS/libraries
- Application Domain - world in which software will operate
- Events in real world that machine can detect – buttons pushed
- Actions in real world that machine can cause- image appearing on screen

D- things true of the world. Assumptions on the domain.

R- things we would like to achieve.

S- description of system- what it should do to meet the requirements.. Shared phenomena

Requirements Elicitation: Where do Requirements come from?
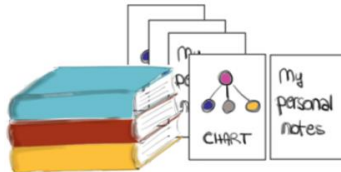
Stakeholders

Application Domain

Documentation

Anybody affected by system or functionality
Constraints posed by application domain, e..g. bank or school regulations

Requirements Elicitation: Other Techniques

Collaborative Techniques - Brainstorming

A software company is developing a new mobile game and invites both developers and gamers for a brainstorming session. During the session, a suggestion for a unique game mode emerges that becomes a central feature of the final product.

Social Approaches - Ethnographic techniques

Analysts spend a day in a busy restaurant to observe how servers use the point-of-sale system. They notice that servers often bypass a time-consuming screen, jotting orders down on paper to enter later, indicating a need for a more efficient user interface.

Requirements Elicitation: Other Techniques

Cognitive techniques - Problem solving methods, Prototyping, Cognitive Task Analysis

In designing software for air traffic controllers, analysts study the controllers' processes for tracking multiple planes. They find that certain visual cues and alerts can reduce mental strain
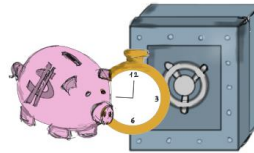
Requirements Elicitation: Functional and Non-functional Requirements

Functional

Non-Functional

Non-Functional requirements: criteria that can be used to judge the operation of a system, rather than specific behaviors. In essence, they describe how the system works, rather than what it does. E.g. security, accuracy, performance, cost, usability, adaptability, interoperability, reusability and so on.

Does not have a clear satisfaction criteria, elevator fast, refine the requirements to make verifiable, reach requested floor in 30 seconds.

Requirements Elicitation: Functional and Non-functional Requirements

Non-Functional Requirement may be more critical than functional requirement
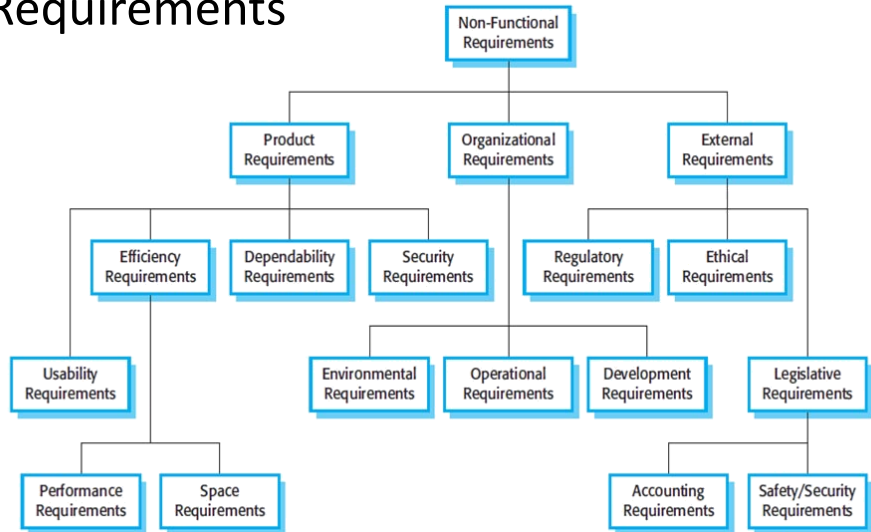
Requirements should be measurable

Property	Measure
Speed	Processed transactions/second User/event response time Screen refresh time
Size	Mbytes Number of ROM chips
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

An example: MHC-PMS: A patient management system for mental health care

- This case study describes a real system (although that is not its real name) which was used (and may still be used) in a number of UK hospitals, including hospitals in Scotland.
- The system is designed for use in clinics attended by patients suffering from mental health problems and records details of their consultations and conditions.
- It is separate from a more general patient records system as more detailed information has to be maintained and the system has to be set up to generate letters and reports of different types and to help ensure that the laws pertaining to mental health are maintained by staff treating patients.

MHC-PMS: Non functional Requirements

- ***Product requirement*** The MHC-PMS shall be available to all clinics during normal working hours (Mon–Fri, 0830–17.30). Downtime within normal working hours shall not exceed five seconds in any one day.
- ***Organizational requirement*** Users of the MHC-PMS system shall authenticate themselves using their health authority identity card.
- ***External requirement:*** The system shall implement patient privacy provisions as set out in the country's law.

Example Quiz

A software application designed for a hospital is running smoothly on its intended hardware. However, the nurses using the system report that it takes too long to enter patient information, causing delays in patient care. Based on the definition of software quality discussed, what type of requirement might need improvement?

A) Functional Requirement
✅ Non-Functional Requirement
C) Maintenance Requirement
D) Hardware Requirement

Requirements Analysis: Traditional Techniques

Background Reading

Hard Data and Samples

Interviews

Surveys

Meetings

Company reports, policy manuals, documentation of existing systems,
Facts and figures, forms, invoices, marketing data, sampling
Structured with agenda or open-ended
Remotely, quickly and from large number of people
Summarization of findings and collection of feedback, confirm or refute what you learned

Requirements Analysis: Completeness and Pertinence
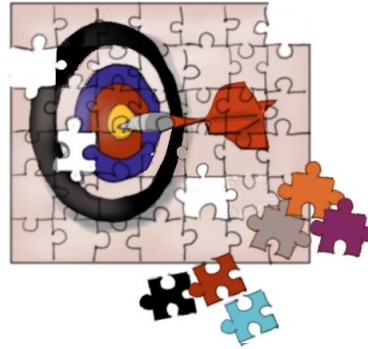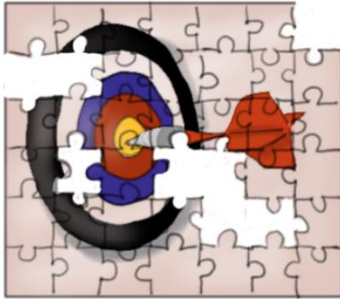
Difficult to identify all requirements, incomplete requirements collected and S/W missing functionality

- Relevance of requirements;
- Irrelevant conflicting requirements collected for sake of completeness
- Worse case: completeness issue not solved, irrelevant requirements with information harmful to system

Requirements Analysis: Requirements Prioritization

Limited Resources

=> Inability to satisfy all the requirements

⇒Need to prioritize them

- Mandatory
- Nice to have
- Superfluous

Example Quiz: Completeness and Pertinence

A software development team is creating a mobile app for online grocery shopping. They collected several requirements, including the ability to browse items, add items to a cart, make payments, and view order history. However, they also included a requirement to integrate a weather forecast feature that shows the weather in the user's location. Which issue does this extra feature represent?

A) Lack of Completeness
✅ Lack of Pertinence
C) Incomplete Requirements
D) Redundant Requirements

Requirements Specification: User and System Requirements

User Requirements

- Written for customers
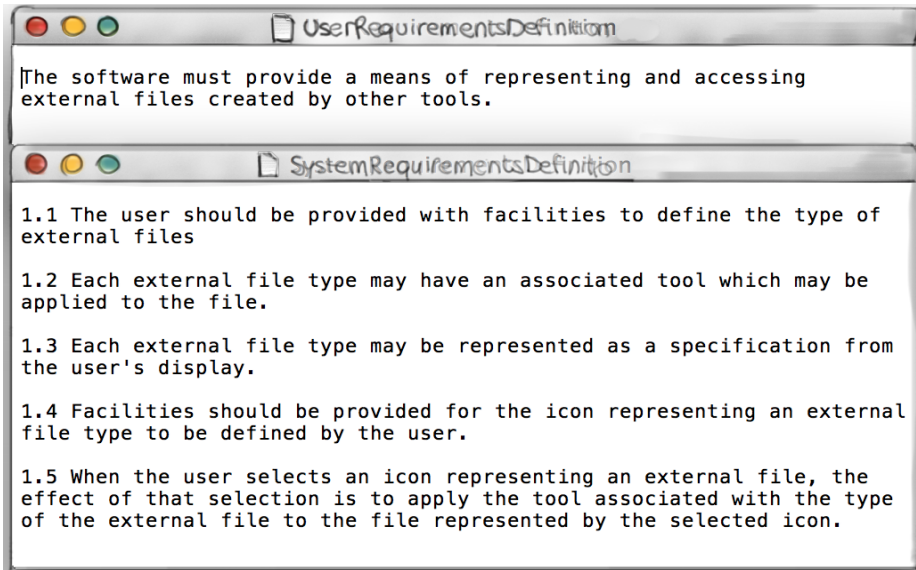- Often in natural language, no technical details

System Requirements

- Written for developers
- Detailed functional and non-functional requirements
- Clearly and more rigorously specified

User requirements: to communicate with the stakeholders or nontechnical audience

User and System Requirements

UserRequirementsDefinition

The software must provide a means of representing and accessing external files created by other tools.

SystemRequirementsDefinition

1.1 The user should be provided with facilities to define the type of external files

1.2 Each external file type may have an associated tool which may be applied to the file.

1.3 Each external file type may be represented as a specification from the user's display.

1.4 Facilities should be provided for the icon representing an external file type to be defined by the user.

1.5 When the user selects an icon representing an external file, the effect of that selection is to apply the tool associated with the type of the external file to the file represented by the selected icon.

Requirements Specification: SRS

- The software requirements specification document is the official statement of what is required of the system developers.

- Should include both a definition of user requirements and a specification of the system requirements.

- It is NOT a design document. As far as possible, it should set of WHAT the system should do rather than HOW it should do it.

- Your project assignment uses the lightweight IEEE format of Software Requirements Specifications Document

Example Quiz

A team is developing a new financial software tool for a bank. After conducting initial interviews, they find that different departments have conflicting requirements: one wants a simplified user interface for quick transactions, while another requires detailed analytics tools that clutter the interface. What should the team focus on next according to the principles of Requirements Engineering?

A) Begin coding the software to meet the needs of one department.
B) Develop a prototype incorporating all requested features.
✅ Conduct further stakeholder analysis to prioritize and clarify requirements.
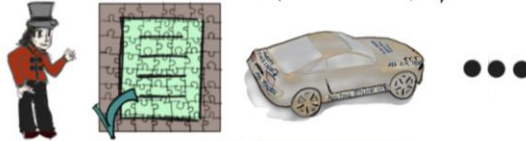D) Decide independently which department's needs are more critical.
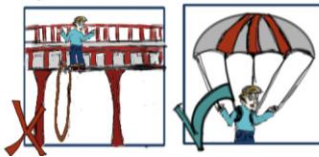
Requirements Validation

Verification
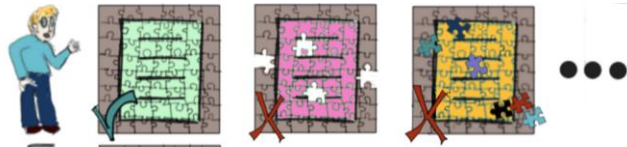
Validation

Risk Analysis

After modelling, analysis-
verification- accurately reflect customer needs as per the developer's knowledge,
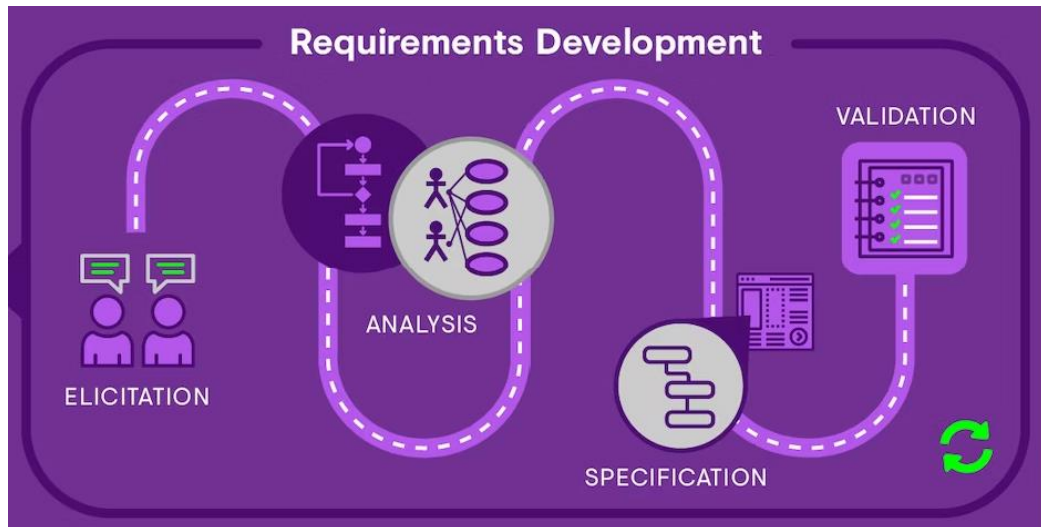complete and pertinent, irrelevant
Validation – system reflects stakeholders request, they can check themselves if they
understand the document, or check prototype
Risk analysis – too risky

Requirements Engineering Process

1. Requirements Elicitation
   1. Consider functional and non-functional requirements
2. Requirements Analysis
   1. User research
   2. Pertinence and Completeness
   3. Prioritize and Categorize
3. Requirements Specification
   1. User and system requirements
4. Requirements Validation
   1. Verification, Validation, Risk analysis
5. Management

Requirements Engineering Process

Requirements Development

ELICITATION — ANALYSIS — SPECIFICATION — VALIDATION

Extract requirements
Model into formalisms
Identify issues

Quizizz

https://www.utc.edu/sites/default/files/2021-04/4900-4-requirements-engineering.pdf