

Software Engineering: Requirements

CPSC 2150

Requirements Engineering

- “The hardest single part of building a software system is deciding what to build. No part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later.” —Fred Brooks
- “You start coding. I’ll go find out what they want.” —Computer analyst to programmer

Requirements Drift

- If developers are not careful, their code will slowly drift away from the actual requirements
 - Ends with a final product that was not what was asked for
- The key to preventing this is to keep the requirements in focus
- This happens often with students
 - They get an assignment and read the prompt
 - Think of a one sentence summary
 - “OK, I’m making a connect 4 game.”
 - They start coding, and never look at the prompt again
 - Just keep thinking of their one sentence summary
 - In the end, they submit a solution that matches the one sentence summary, but misses key requirements from the prompt, which causes them to lose a lot of points
 - Yeah, it plays connect 4, but you forgot to _____ and _____ and _____

Requirements Engineering

- Helps software engineers understand the problem they are to solve.
- Must understand the domain or business context
 - How will the user use the system?
 - How will the system be used on a day to day basis?
- Fact finding
 - How is the problem currently solved
 - Is that the best way to solve it?
- Envisioning
 - How will the planned system work
 - Not too much detail
- It's a long process that will be covered more in 3720. I just want to introduce some basics

Functional vs Non-functional Requirements

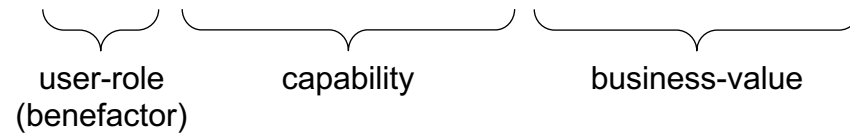
- Functional Requirements
 - Describe interactions between system and its environment and users
 - EX. The cashier can add items to the order
- Non-functional Requirements
 - Describe aspects of the system not directly related to functional behavior of the system
 - Not something the user can do
 - Still a requirement of the system
 - The system must be coded in Java
 - The system must run on Unix
 - These are often the ones that students forget

User Stories

- User stories help us capture our functional requirements
- User stories are short sentences with three parts
 - The user role
 - Sometimes it's just user, but it's helpful to be more specific
 - The Capability
 - The action the user can take in the system
 - The business value, or benefit
 - Why does the user want to take this action?
 - How does it help them?

User Stories

As a tenant, I can unlock the doors to enter my apartment.



User stories

- How short should they be?
 - Very short
 - They should only describe one action
- Once user stories are created, they will be assigned one at a time to team members
 - So you want a simple, specific task to work on.

Examples

- Consider a program that would get the size of a matrix from the user, have the user fill in the matrix with data and then provide:
 - A print out of the matrix
 - The Transpose of the matrix
 - The product sum of each row
 - The average of all numbers in the matrix
 - The sum of all numbers for each column in the matrix
 - The sum of all numbers for each row in the matrix

What functional requirements?

- We are looking for actions that the user or the program takes
 - The user enters the size of the matrix
 - The user fills the matrix with data
 - The program prints the matrix
 - The program solves and displays the transpose of the matrix
 - The program solves and displays the product sum of each row
 - The program solves and displays the average of all numbers in the matrix
 - The program solves and displays the sum of all numbers for each column in the matrix
 - The program solves and displays the sum of all numbers for each row in the matrix
- We need to write these as user stories

Convert to User Stories

- Functional Requirement: The user enters the size of the matrix
- Role: User
- Action: enters the size of the matrix
- Benefit: So they can have whatever size matrix they want
- User Story: As a user, I can enter the size of the matrix, so I can have whatever size matrix I want

Convert to User Stories

- Functional Requirement: The program prints the matrix
- Role: User (The program is not a user!)
- Action: View the matrix
- Benefit: to see what data is in it
- User Story: As a user, I can view the matrix so I can see what data is in it.

Convert to User Stories

- Functional Requirement: The program solves and displays the product sum of each row
- Role: User
- Action: get the product sum of each row
- Benefit: to know what the product sum is
- User Story: As a user, I can get the product sum of each row in my matrix, so I can know what the product sum is

Non-functional requirements

- Any requirements *for the program* that are **not an action**
- Note: some requirements on the prompt are just requirements for the assignment
 - Don't use magic numbers
 - Use good comments
 - Write contracts
 - Make a lab report

Non-functional Requirements

- FURPS+

- Functionality

- Usability

- Reliability

- Performance

- Supportability

Quality Requirements

- +

- Implementation

- Interface

- Operations

- Packaging

- Legal

Constraints / Psuedo
Requirements

Non-Functional Requirements

- Usability
 - Conventions adopted by UI
 - Scope of online help
 - User Documentation
 - How easy it is to use the system
- Reliability
 - Robustness
 - Safety
 - Fault tolerances
 - Security
 - AKA Dependability

Non-functional Requirements

- Performance
 - Response time
 - Throughput
 - Availability
 - Accuracy
- Supportability
 - Adaptability (after deployment)
 - Maintainability
 - Portability

Non-functional Requirements (+)

- Implementation
 - Specific tools, languages, hardware platforms
- Interface
 - Constraints imposed by external systems
 - Formats to exchange data
 - Not GUI
- Operations
 - Constraints on admin and management of system while operational
- Packaging
 - Delivery of system
 - Installation
- Legal
 - Licensing
 - Accessibility
 - Privacy
 - Certification

Requirements Analysis

- In this course you will be given assignment prompts that outline your requirements
- You will need to analyze that prompt and make your list of user stories and non-functional requirements
- This will help ensure you better understand the requirements, and give you an easy to reference list to check as you work on the assignment