CS 315 - Lecture 6 - Sep 9, 2015

Chapter 11: Requirements

- Lecture Slides
  - With Use Cases
  - Without Use Cases

- The Aim of the Requirements Workflow
  - To answer the question: What must the product be able to do?
  - Misconception
    - We must determine what the client wants

  - Reality
    - We must determine what the client needs
    - It is hard for systems analyst to visualize a software product and its functionality
      - The problem is far worse for the client

    - A skilled systems analyst is needed to elicit the appropriate information from the client
      - the client is the only source of this information

- Determining What the Client Needs
  - Obtain initial information from the client
  - Use this initial information as input to the Unified Process
  - Follow the steps of the Unified Process to determine the client's real needs

- Overview of the Requirements Workflow
  1. Gain an understanding of the *application domain* (or *domain*, for short)

     - The specific environment in which the target product is to operate

  2. Build a business model
     - Model the clients business processes

  3. Use the business model to determine the client's requirements
  4. Iterate the above steps ↺

- Definitions

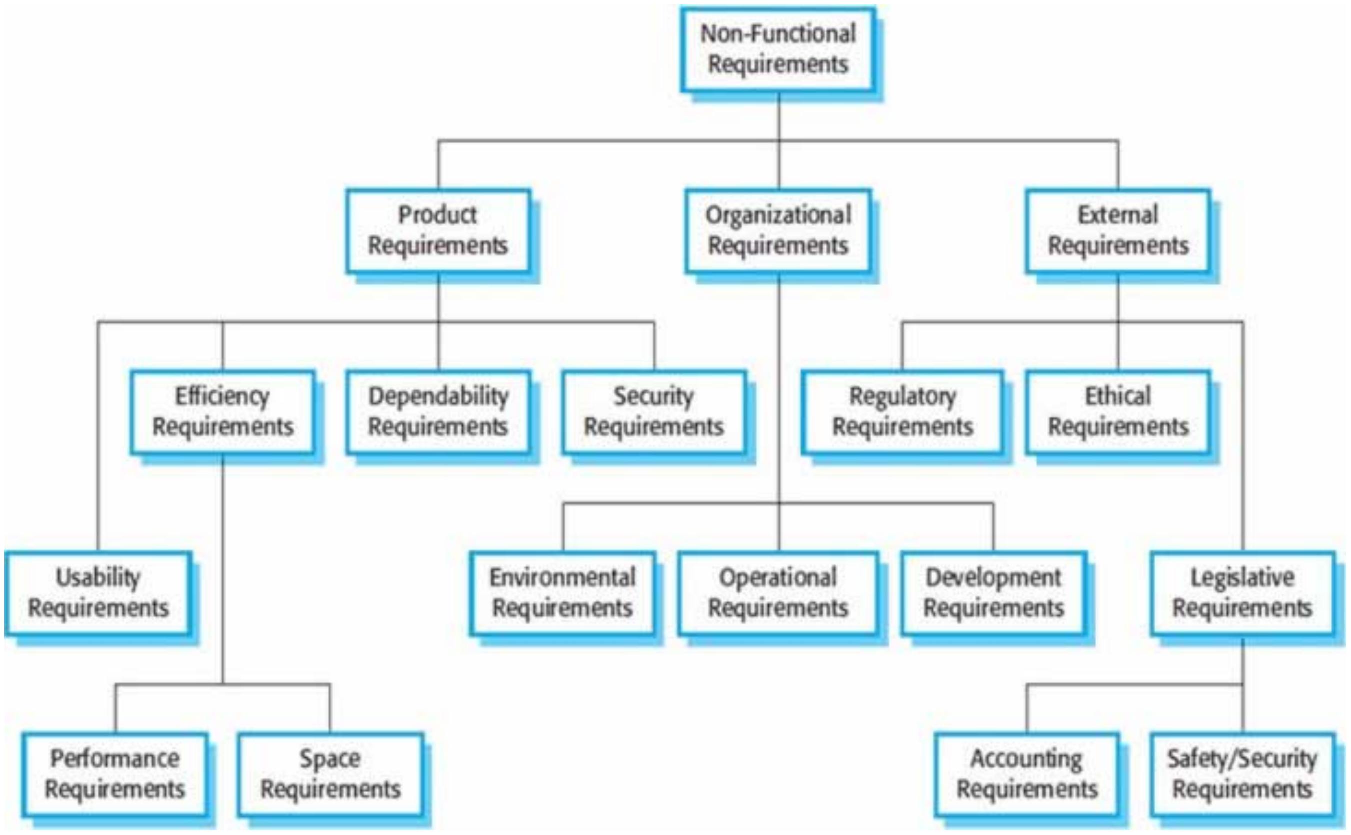
  - Discovering the client's requirements
    - *Requirements elicitation* (or *requirements capture*)
    - Methods include interviews and surveys
    - Agile is good at this because of frequent contact with the client
  - Refining and extending the initial requirements
    - *Requirements analysis*
- Understanding the Domain
  - Every member of the development team must become fully familiar with the application domain
    - Correct terminology is essential
  - Construct a *Glossary*
    - A list of technical words used in the domain, and their meanings
    - Spell out abbreviations and acronyms
- Business Model
  - A *business model* is a description of the business processes of an organization
  - The business model gives an understanding of the client's business as a whole
    - This knowledge is essential for advising the client regarding computerization
  - The systems analyst needs to obtain a detailed understanding of the various business processes
    - Different techniques are used, primarily interviewing
  - Interviewing
    - The requirements team meet with the client and users to extract all relevant information
    - There are two types of questions
      - Close-ended
        - Questions require a specific answer
      - Open-ended
        - Questions are posed to encourage the person being interviewed to speak out
    - There are two types of interviews
      - Structured
        - Specific, preplanned questions are asked
        - Frequently close-ended
      - Unstructured
        - Questions are posed in response to the answers received
        - Frequently open-ended

        - Interviewing is not easy
            - An interview that is too unstructured will not yield much relevant information
            - The interviewer must be fully familiar with the application domain
            - The interviewer must remain fully open-minded at all times
        - After the interview, the interviewer must prepare a written report
            - It is strongly advisable to give a copy of the report to person who was interviewed
            - Providing feedback to the person you interviewed keeps them interested
    - Other Techniques
        - A questionnaire is useful when the opinions of hundreds of individuals needs to be determined
        - Examination of business forms shows how the client currently does business
        - Direct observation of the employees while they perform their duties can be useful
            - Video cameras are a modern version of this technique
            - It can take a long time to analyze the tapes
            - Employees may view the cameras as an unwarranted invasion of privacy
                - Should ensure that the employees know you are there
- High-Level Requirements
    - Executive document, business case, constraints on the software product and project
        - Opportunity and Need
            - Inventory system losing 50% of customer orders, there is $2M extra inventory, need to increase customer orders by 30%
    - Justification
        - Scope
            - Inventory control and order processing
        - Major Constraints
            - Budget (if known), Schedule (if known), Risks
        - Major Functionality
            - Improved inventory control via automating order and shipping process
            - Online customer orders
            - Online Delivery/Shipping control
        - Success Factor
            - Must reduce inventory
            - Must not lose customer orders
            - Comes from the client's domain

    - User Characteristics

- Functional Requirements
    - Most obvious group, starting point
        - What should the system do?
    - Specifies an action that the system must be able to perform as an interaction between the system and its environment
        - Often expressed in terms of inputs and outputs
        - Independent from its implementation
    - Typically handled during requirement and analysis workflows
    - Business Flow
        - Functionality needs to be explained in the context of a business flow
        - Step-by-step scenarios
        - Use Case
            - Sequence of actions that a system should perform within the business flow context of the user
            - Specific steps necessary to accomplish a specific task
    - Data & Formats
        - Determine the application's input and output data
            - What needs to be entered into the system?
            - For what purpose?
        - Some input data may trigger a process
        - Output may be in form of a query response or a report
        - Allowed format of information input
        - Required format of information output
        - Error messages, warnings, help text
    - User Interface
        - How the input and output are presented
        - Look and Feel
        - Flow should follow business flow as close as possible
        - Screenshots, mocking, **rapid prototyping**
    - Interface with Other Systems
        - Existing applications, network systems
        - Some requirements may require a modification in how users operate adjacent systems
            - Software should adapt to existing systems

        - Or even adapt the other system to the new software
    - Dimensions to consider
        - Transfer of Control
        - Transfer of data
        - Receipt of responses
        - Error recovery, retry capabilities, messages
    - Should be aware of other systems as early as possible in the process

- Non-Functional Requirements
  - 

  - Specifies properties of the software product itself
    - Platform constraints
    - Response times
    - Performance
    - Reliability
    - Security
    - Adaptability
    - Availability
    - Transportability
    - Maintainability

    - Typically wait until design workflow
    - Constraints on programming language, OS, tooling, license

- Requirements Elicitation Activities
    - Actors identification
        - Types of users to support

    - Scenarios identification
        - Observe users in their daily activities
        - Concrete example per functionality
        - Deepen understanding in application domain

    - Use Cases identification
        - Generalize scenarios to completely represent the system
        - Abstraction from scenarios describing all possible cases
        - This defines the scope of the system

    - Use Case refinement
        - Make sure the requirements are complete
        - Detailed description of behavior
        - Handle error and and exceptional conditions

    - Use Case relationship identification
        - find dependencies between use cases
        - Factor out common functionality
        - Ensure requirements are consistent

    - Non-Functional Requirements Identification
        - Agree on aspects visible to user
        - Performance
        - Documentation
        - Resource
        - Consumption
        - Quality