Khulna University
Computer Science & Engineering Discipline

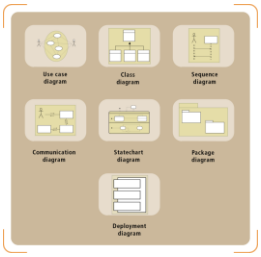
SK Alamgir Hossain

Assistant Professor
Computer Science & Engineering Discipline
Khulna University, Khulna, Bangladesh

http://alamgirhossain.com

CSE3203 - Software Engineering and Information System Design

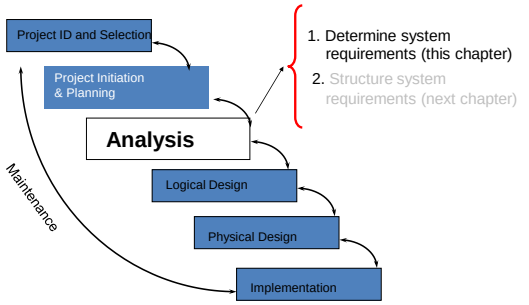
UML Revise



Chapter 7

- Requirement Engineering
- Determining System Requirements

SDLC - Where are we?



Lecture 12

Requirement Engineering

- What is RE ?
- Characteristics of a Good Requirement
- Requirements Engineering Tasks
- Initiating Requirements Engineering Process
- Software Prototype
- Software Specification

Requirements Engineering

- Requirement: A function, constraint or other property that the system must provide to fill the needs of the system's intended user(s)
- Engineering: implies that systematic and repeatable techniques should be used
- Requirement Engineering means that requirements for a product are defined, managed and tested systematically

Requirements Engineering

- It is essential that the software engineering team understand the requirements of a problem before the team tries to solve the problem.
- In some cases requirements engineering may be abbreviated, but it is never abandoned.
- RE is software engineering actions that start with communication activity and continues into the modeling activity.
- RE establishes a solid base for design and construction. Without it, resulting software has a high probability of not meeting customer needs.

Characteristics of a Good Requirement

- Clear
  - standard structure
  - has only one possible interpretation
  - Not more than one requirement in one sentence
- Correct
  - A requirement contributes to a real need
- Understandable
  - A reader can easily understand the meaning of the requirement
- Verifiable
  - A requirement can be tested
- Complete
- Consistent
- Traceable

Why is Getting Good Requirements Hard?

- Stakeholders don't know what they really want.
- Stakeholders express requirements in their own terms.
- Different stakeholders may have conflicting requirements.
- Organisational and political factors may influence the system requirements.
- The requirements change during the RE process. New stakeholders may emerge and the business environment change.

Requirements Engineering Tasks

- 1. Inception —Establish a basic understanding of the problem and the nature of the solution.
- 2. Elicitation —Draw out the requirements from stakeholders.
- 3. Elaboration (Highly structured)—Create an analysis model that represents information, functional, and behavioral aspects of the requirements.
- 4. Negotiation—Agree on a deliverable system that is realistic for developers and customers.
- 5. Specification—Describe the requirements formally or informally.
- 6. Validation —Review the requirement specification for errors, ambiguities, omissions, and conflicts.
- 7. Requirements management —Manage changing requirements.

1. Inception

- Inception— Ask "context-free" questions that establish …
  - Basic understanding of the problem
  - The people who want a solution
  - The nature of the solution that is desired, and
  - The effectiveness of preliminary communication and collaboration between the customer and the developer

2. Elicitation

- Elicitation - elicit requirements from customers, users and others.
  - Find out from customers, users and others what the product objectives are
  - what is to be done
  - how the product fits into business needs, and
  - how the product is used on a day to day basis

Why Requirement elicitation is difficult?

- Problems of scope:
  - The boundary of the system is ill-defined.
  - Customers/users specify unnecessary technical detail that may confuse rather than clarify objectives.
- Problem of understanding:
  - Customers are not completely sure of what is needed.
  - Customers have a poor understanding of the capabilities and limitations of the computing environment.
  - Customers don't have a full understanding of their problem domain.
  - Customers have trouble communicating needs to the system engineer.
  - Customers omit detail that is believed to be obvious.
  - Customers specify requirements that conflict with other requirements.
  - Customers specify requirements that are ambiguous or not able to test.
- Problems of volatility:
  - Requirement change over time.

3. Elaboration

- Focuses on developing a refined technical model of software functions, features, and constraints using the information obtained during inception and elicitation
- Create an analysis model that identifies data, function and behavioral requirements.
- It is driven by the creation and refinement of user scenarios that describe how the end-user will interact with the system.
- Each event parsed into extracted.
- End result defines informational, functional and behavioral domain of the problem

4. Negotiation

- Negotiation - agree on a deliverable system that is realistic for developers and customers
  - Requirements are categorized and organized into subsets
  - Relations among requirements identified
  - Requirements reviewed for correctness
  - Requirements prioritized based on customer needs
  - Negotiation about requirements, project cost and project timeline.
  - There should be no winner and no loser in effective negotiation.

5. Specification

- Specification – Different things to different people.
- It can be –
  - Written Document
  - A set of graphical models,
  - A formal mathematical models
  - Collection of usage scenario.
  - A prototype
  - Combination of above.
- The Formality and format of a specification varies with the size and the complexity of the software to be built.
- For large systems, written document, language descriptions, and graphical models may be the best approach.
- For small systems or products, usage scenarios

6. Validation

- Requirements Validation - formal technical review mechanism that looks for
  - Errors in content or interpretation
  - Areas where clarification may be required
  - Missing information
  - Inconsistencies (a major problem when large products or systems are engineered)
  - Conflicting or unrealistic (unachievable) requirements.

7. Requirement Management

- Set of activities that help project team to identify, control, and track requirements and changes as project proceeds
- Requirements begin with identification. Each requirement is assigned a unique identifier. Once requirement have been identified, traceability table are developed.
- Traceability Table:
- Features traceability table - shows how requirements relate to customer observable features
- Source traceability table - identifies source of each requirement
- Dependency traceability table - indicate relations among requirements
- Subsystem traceability table - requirements categorized by subsystem
- Interface traceability table - shows requirement relations to internal and external interfaces
- It will help to track, if change in one requirement will affect different aspects of the system.

Collaborative Requirement Gathering

- Meetings are attended by all interested stakeholders.
- Rules established for preparation and participation.
- Agenda should be formal enough to cover all important points, but informal enough to encourage the free flow of ideas.
- A facilitator controls the meeting.
- A definition mechanism (blackboard, flip charts, etc.) is used.
- During the meeting:
  - The problem is identified.
  - Elements of the solution are proposed.
  - Different approaches are negotiated.
  - A preliminary set of solution requirements are obtained.
  - The atmosphere is collaborative and non-threatening.
- Flow of event – Outline the sequence of events occurs
  - Requirement gathering meeting (initial meeting)
  - During meeting
  - Follow the meeting.

Collaborative requirement gathering (contd.)

- In initial meeting, distribute "Product request" (defined by stakeholder) to all attendee.
- Based on product request, each attendee is asked to make
  - List of objects (Internal or external system objects)
  - List of services(Processes or functions)
  - List of constraints (cost, size, business rules) and performance criteria(speed, accuracy) are developed.
- Collect lists from everyone and combined.
- Combined list eliminates redundant entries, add new ideas , but does not delete anything.
- Objective is to develop a consensus list in each topic area (objects, services, constraints and performance).
- Based on lists, team is divided into smaller sub-teams : each works to develop mini-specification for one or more entries on each of the lists.

User Scenario

- It is difficult to move into more software engineering activities until s/w team understands how these functions and features will be used by diff. end-users.
- Developers and users create a set of usage threads for the system to be constructed
- A use-case scenario is a story about how someone or something external to the software (known as an actor) interacts with the system.
- Describe how the system will be used
- Each scenario is described from the point-of-view of an "actor"—a person or device that interacts with the software in some way

Lecture 13+14

Requirement Engineering

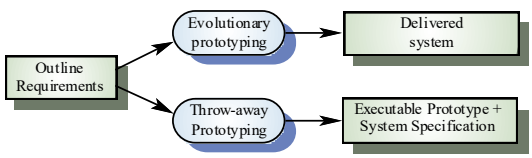
- What is RE ?
- Characteristics of a Good Requirement
- Requirements Engineering Tasks
- Initiating Requirements Engineering Process
- Software Prototype
- Software Specification

Software Prototype

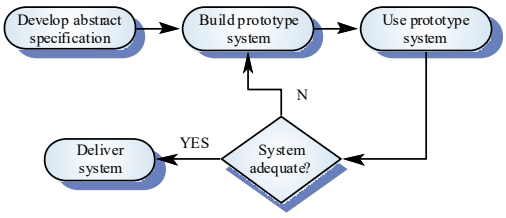
- Prototype constructed for customer and developer assessment.
- Prototype Approach
  - Close ended (Throw-away Approach)
  - Open ended (Evolutionary Approach)

  - Close Ended – It serves as a rough demonstration of requirement. It is then discarded, and the software engineered using a different paradigm.
  - Open Ended - uses the prototype as the first part of an analysis activity that will be continued into design and construction. The prototype of the software is the first evolution of the finished system.

Approaches to prototyping



Evolutionary prototyping



Evolutionary prototyping advantages

- Accelerated delivery of the system
  - Rapid delivery and deployment are sometimes more important than functionality or long-term software maintainability
- User engagement with the system
  - Not only is the system more likely to meet user requirements, they are more likely to commit to the use of the system

Evolutionary prototyping problems

- Management problems
  - Existing management processes assume a waterfall model of development
  - Specialist skills are required which may not be available in all development teams
- Maintenance problems
  - Continual change tends to corrupt system structure so long-term maintenance is expensive
- Contractual problems
  - Due to cost or time line agreed

Throw-away prototyping

- Used to reduce requirements risk
- The prototype is developed from an initial specification, delivered for experiment then discarded
- The throw-away prototype should NOT be considered as a final system
  - Some system characteristics may have been left out
  - There is no specification for long-term maintenance
  - The system will be poorly structured and difficult to maintain

Traditional Methods in Collecting/Determining Requirements

1. Interviewing and Listening
2. Questionnaires
3. Interviewing Groups
4. Direct Observations
5. Analyzing Procedures and Documents

1. Interviewing and Listening

- Plan for the Interview
  - make an appointment, framing questions
  - prepare a checklist and meeting agenda
- Listen carefully and take notes
- Observe body language and emotions
- Review notes within 48 hours
- Be neutral
- Seek diverse views

Types of Questions used in Interviewing

- Open Ended (i.e., no pre-specified answers)
  - Unknown information may surface
  - Interviewees can respond in their own words – sense of involvement and control
  - Take more time and responses may be difficult to summarize
- Closed Ended (i.e., choose from set of answers)
  - Take shorter time since major answers to questions are well known
  - Some useful information may be overlooked as respondents try to make a choice

Interview Characteristics

- Information Richness is High
- Time and Expense is High
- Easy to Follow-up
- Interviewee is Known (not anonymous)
- Interviewee is involved and committed
- Limited Sample

2. Questionnaire

- More cost-effective than interviews
- Questionnaire can be administered over the phone, mail, or in person
- Mostly closed-ended questions

Questionnaire Characteristics

- Information Richness is Low (relatively)
- Time and Expense is Low (relatively)
- Hard to do Follow-up
- Respondents can be Anonymous
- Respondents is passive or not committed
- Large Sample

3. Interviewing Groups

- Advantages
  - More effective use of time
  - Enables people to hear opinions of others and to agree or disagree
- Disadvantages
  - Difficulty in scheduling

4. Direct Observing Users

- Serves as a good method to supplement interviews
- Often difficult to obtain unbiased data
  - People often work differently when being observed

5. Document Analysis

- Four types of useful documents
  - Written work procedures
    - Describes how a job is performed
    - Includes data and information used and created in the process of performing the job or task
  - Business form
    - Explicitly indicate data flow in or out of a system
  - Report
    - Enables the analyst to work backwards from the report to the data that generated it
  - Description of current information system

Document Analysis Characteristics

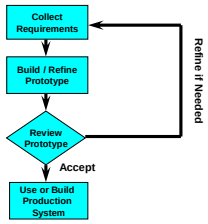
- Time and Expense is Low (relatively)
- Limited follow-up
- May not be confidential (depends on document)
- Potentially biased sample -- what is kept / shared may not reflect reality

Modern Methods in Collecting / Determining Requirements

1. Prototyping
2. Joint Application Design (JAD)

1. Prototyping

- Quickly converts requirements to working version of system
- Once the user sees requirements converted to system, will ask for modifications or will generate additional requests



Reasons for Prototyping

- Most useful when:
  - User requests are not clear
  - Few users are involved in the system
  - Designs are complex and require concrete form
  - History of communication problems between analysts and users
  - Tools are readily available to build prototype
- Drawbacks
  - Tendency to avoid formal documentation
  - Difficult to adapt to more general user audience
  - Sharing data with other systems is often not considered

Types of Prototypes

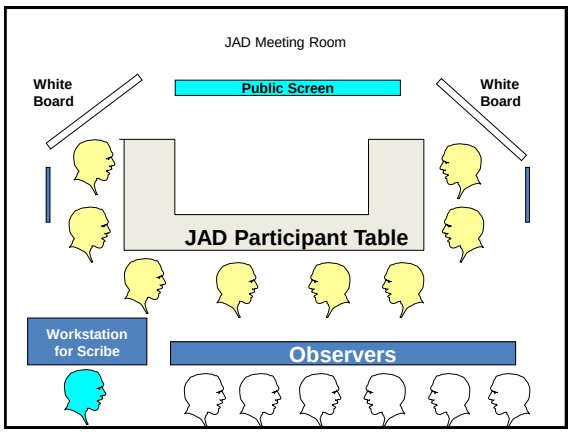
- Simulated Prototype (slide show)
  - Throw-away prototyping(Proof-of-concept)
  - quick and dirty
- Partial-function Prototype
  - Pilot prototyping
  - Could be expanded to full

Prototyping Characteristics

- Speed
- Iterative and interactive
- Close working relationship with customers
- Good for system requirements which must be “discovered”

2. Joint Application Design (JAD)

- A Special type of Group Meeting to:
  - Define system requirements simultaneously from key people
  - Approve designs
- Conducted off-site
- Brings together:
  - key users
  - systems analysts
  - managers / sponsors
  - IS staff



JAD Workshop

- May last several days
- Lots of pre-planning
- Retreat Atmosphere
- No more than 15 people
- Can technology help? (Yes, using GSS)

THANK YOU