Software Engineering – Fall 2015
(CSC 4350/6350)
TR. 5:30 pm – 7:15 pm

Rao Casturi
09/10/2015
http://cs.gsu.edu/~ncasturi1
Recap – UML Notation

Use Case Diagrams

Class Diagrams

Interaction Diagrams

Use Case ID:
Use Case Name:
History:
Date:
Actors:
Description:
Trigger:
Enter Conditions:
Exit Conditions:
Normal Flow:
Alternative Flows:
Exceptions:
Includes:
Frequency:
Notes and Issues:


GSU: Software Engineering - CSC4350/6350 - Rao Casturi
Recap – UML Notation

State Diagrams

Activity Diagrams


GSU: Software Engineering - CSC4350/6350 - Rao Casturi
Project Life Cycle
Project Organization and Communication

Communication

- Structured / Unstructured

Projects consists of 4 major components.

1. Work product
2. Schedule
3. Participants
4. Task

Project:
- Formal – Contact / $ Amount / Time
- Informal – Word / Trust
Project Organization Concepts

1. Project Organization
2. Roles
3. Tasks and Work Products
4. Schedules
1. Project Organization

- Team Based Organization
  - Hierarchical
  - Cross-Functional

- Interactions (3 Types)
  - Reporting
    - Used for status information
    - Between Develops or Team Lead to Project manager
  - Decision Making
    - Project Manager can take a decision to move the timelines
    - Team Lead can decide to change the logic to implement a scenario
  - Communication
    - Many forms
    - Formal and informal
Hierarchical & Cross-Functional

- **Hierarchical**
  - Reaction Time Slow
  - Wrong people making decisions
  - Lowest level participation has no control of the project timelines
  - Budget decisions efficiently

- **Cross-Functional**
  - Fast Reaction time
  - Right people at different decision levels
  - Budget control can be an issue
  - Complex communication leads to some slip through gaps
## 2. Roles

<table>
<thead>
<tr>
<th>Role</th>
<th>Responsibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Management</td>
<td>Client - Problem Description, Project Managers - Overall incharge, Team Leaders - Incharge for a small teams and assign tasks</td>
</tr>
<tr>
<td>Development</td>
<td>Design of Systems, architect, Object Designers etc.</td>
</tr>
<tr>
<td>Tester</td>
<td>Testing the user cases, Unit Testing etc</td>
</tr>
<tr>
<td>Support</td>
<td>Gives System Support</td>
</tr>
</tbody>
</table>
3. Task and Work Products

- Task is a well defined unit of work
- Work Product is a outcome from a task
  - Document
  - Object
  - Subsystems
  - Test Cases
  - Use Cases

<table>
<thead>
<tr>
<th>Work Product</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Plan</td>
<td>Document</td>
<td>Gives a test plan for the system or unit</td>
</tr>
<tr>
<td>Design of Objects</td>
<td>Class Model</td>
<td>Shows all the class objects in the system</td>
</tr>
<tr>
<td>Subsystem (Code)</td>
<td>Source Code</td>
<td>Produced by development team and submitted for review</td>
</tr>
</tbody>
</table>
4. Schedule

- Mapping of task on a timeline or plot
- Each task has a life of its own (Start and End)
- 2 Types of Schedule Charts are widely used
- PERT and Gantt Charts

Gantt Chart (TASK vs Time)
PERT Charts

- PERT (Program Evaluation Review Technique)
  - Critical Path
  - Graphical Representation
  - Milestones, Sequence of tasks

![PERT Chart Example](image.png)

PERT chart for a project with five milestones (10 through 50) and six activities (A through F). The project has two critical paths: activities B and C, or A, D, and F – giving a minimum project time of 7 months with fast tracking. Activity E is sub-critical, and has a float of 1 month.

**Project Charter**

**Goal**
To streamline the current XXXXXXXXXX process to a Standardized Enterprise Architecture with robust process control and monitoring mechanism.

**Sponsors**
Risk Management, Front Office

**Problem Statement**
The current Risk Reporting for XXXXXXX process depends on the overnight process for Clients Analytics. With the process dependency, any delay or failure on overnight process impacts the Risk Reporting of XXXXXXX. Risk Team is spending a lot of hours in maintaining or fixing the process. The existing process can’t be supported for a prolonged period of time.

**Team**
Risk Management, IT & Infrastructure Team (DBA), Front office Team

**Out of Scope**
- Sunset existing Process
- Data Quality Issues from the Source Systems

**Proposed Solution**
Separate current process for the XXXXX from the existing overall overnight process. Set up data controls around the process and to identify data issues earlier. Set up clear ownership on the source data. Redesign the reports and data structure where it is needed to optimize the response time.

**Time Line**

- **Initiation**
  - 08/01/2014
  - Completed

- **Analysis & Solution**
  - 08/1 to 8/15
  - Completed

- **Implementation & Development**
  - 8/15 to 9/10
  - Completed

- **Testing**
  - 9/10 to 9/20
  - In Progress

- **Production**
  - 9/21 to 9/27
  - Not Started

- **Control & Closeout**
  - 09/30
  - Not Started
Project Communication – Planned & Un planned

 Planned:
  • Problem presentation
  • Client reviews
  • Project reviews
  • Peer reviews
  • Status reviews
  • Brainstorming
  • Releases
  • Postmortem reviews.

 Un Planned:
  • Requests for clarification
  • Requests for changes
  • Issue resolution.
Questions ?
Requirement Elicitation
**Requirement Engineering**

- **First step** for understanding the “System”
- **Very Critical** for the System Success
- **2 Components**
  - Requirement Specification (Gathering)
  - Analysis Model

**Diagram:***

- **Problem Statement**
- **Requirements Specification**
  - Functional
  - Non-Functional
- **Analysis Model**
  - Dynamic Model
  - Analysis Object Model
- **Requirement Elicitation**
- **Analysis**
Key Ideas:

– This is about the communication among the resources working on the project
– Human interaction
– Story board about the System Proposed
– Understanding the problem and articulate it back
– Ability to establish the boundaries
**Requirement Elicitation Concepts:**

- **Functional Requirements**
  - Complete with all variations
  - Unambiguous
  - Correct

- **Non Functional Requirements**
  - Usability
  - Reliability
  - Robustness
  - Supportability
  - Performance

- **Completeness, Consistency, Clarity, Accuracy**
- **Realism, Verifiability, Traceability**
  - Should be realistic
  - Able to verify the outcome
  - Should match with the original requirements

- **Greenfield, Re-Engineering, Interface Development**
- **From Start**
  - Develop or work on existing system
  - Build UI/Interfaces
- **MUST to have**
  - Interactions with the system
Requirement Elicitation Activities:

• Identify the Actors (Roles)
  – During this phase the system developers will try to layout the various users and their roles
  – This help in a better understanding of the Application Domain

• Identify Scenarios:
  – Have discussions on some example how the system will work
  – Helps in understanding the system better

• Identify Use Case:
  – This is a detailed document of the various scenarios
  – Capture all the possible functions the system should able to perform
Requirement Elicitation Activities:

- Refine the Use Cases:
  - This phase is utilized to refine and eliminate any of the duplicate
  - Re verify that all the requirements re captured and each one has its own Use Case

- Identify the Relations:
  - During this phase the relationship between the various Use Cases
  - Dependencies are identified
  - Identify the common functions in the system

- Identify the Non-Functional Requirements
  - Get a confirmation from the client
  - Document the non functional requirements
Requirement Elicitation Activities: (Actor)

- Actors are **Role Abstraction** - not necessary to map to a persons
- One Actor can take multiple roles
- Subsystems can be Actors

**Questions for identifying actors**

- Which user groups are supported by the system to perform their work?
- Which user groups execute the system’s main functions?
- Which user groups perform secondary functions, such as maintenance and administration?
- With what external hardware or software system will the system interact?
Questions for identifying scenarios

✓ What are the tasks that the actor wants the system to perform?
✓ What information does the actor access? Who creates that data? Can it be modified or removed? By whom?
✓ Which external changes does the actor need to inform the system about? How often? When?
✓ Which events does the system need to inform the actor about? With what latency?
Aggregation of the Scenarios
Use Case is initiated by an ACTOR
Use Case can interact with other Actors
Writing Use Case is an ART

Best Practices of Use Case
✓ Number each Use Case Uniquely
✓ Name the User Case to represent the ACTION (Verb)
✓ Name the ACTORS with the Role (Noun)
✓ Exit and Entry conditions should be clear
✓ Don’t write about the user interface
✓ Exceptions should be outlined clearly
✓ Flow of activity should be clear and numbers for easy identification
Goal of this activity is to capture all the “Shall” statement sentences from the requirement documents into a table format.

End of this activity the “Requirements Trace Matrix” is a deliverable to the project team

Why is this activity important?
Scope Creep
Foundation for next phases (Use Cases, Categories, Classes, Methods)
Cost & Time lines of project
Future development (Incremental) Issues
Questions ?