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

Rao Casturi
09/08/2015
http://cs.gsu.edu/~ncasturi1
Functional and Non Functional Requirement

• Functional
  – Specification a system should support
  – Need to have

• Non Functional
  – Constraint on the System
  – No direct relation to the function of the system
Software Engineering Development Activities

• Requirement Elicitation (Gathering)
• Analysis
• System Design
• Object Design
• Implementation
• Testing
Managing Software Development

• Communication
  – Time consuming activity
  – Critical for the project
• Rational Management
  – Justification – Complex activity
• Software Configuration
  – Version controls
  – Maintenance
• Project Management
  – Art
• Software Life Cycle
  – Putting it all together is the life cycle of the Software Development
Introduction to UML

• What is UML?
  – Unified Modeling Language
  – History
  – Why is it so important in Software Engineering?
  – Fundamental Notations of UML
Basic Notations of UML

- Use Case Diagrams
- Class Diagrams
- Interactive Diagrams
- State Machine Diagrams
- Activity Diagrams
1. Use Case Diagram:

Definitions:
- **Use Case:**
  - *Statement of functionality required by software*
  - 3 Main components of Use Case
    - **ACTOR, ACTION, SUBJECT**
  - Use Case is written in specific format
  - Use Case (UC) is a Functional Requirement

Subject represents the object on which the Action will take place.
Example of an Use Case

- Portfolio Manager
- Risk Manager
- Trader

- Balance Portfolio
- Risk Analysis
- Set Risk Limits
- Price The bond
- Valuation of the Bond
- Broker Quote
- End of Day Holdings

(System Boundary)

Data Warehouse
Broker

GSU: Software Engineering - CSC4350/6350 - Rao Casturi
## Use Case Name

<table>
<thead>
<tr>
<th>Use Case Name</th>
<th>UC_001_Bond_Risk_Analysis</th>
</tr>
</thead>
</table>

## Actors/Participants

<table>
<thead>
<tr>
<th>Actors/Participants</th>
<th>Risk Manager, Trader</th>
</tr>
</thead>
</table>

## Flow of Events/Scenario

1. Find the **historical P&L** of the bond
2. Depending on the P&L data decide if we can add this bond to the portfolio or not.
3. Set **Risk Target** if you need to add to the portfolio

## Entry Condition

When a Portfolio Manager requests a Bond to be analysed

## Exit Condition

The Risk appetite set

## Quality constraints

The Risk Limit set should be in line with the other similar instruments

## Relationships to Reduce Complexity:

- **Communication Relationships (Solid Line)**
- **<<Include>> (Dotted Line)**
- **<<Extend>> (Dotted Line)**
- **Inheritance (Solid Line with Triangle Head)**
2. Class Diagram:
   - Describes the structure of the system in terms of Classes and Objects
   - Classes: - Collection of objects
   - Objects: - The Entities to capture the state and behavior of the system
   - Consists of 3 Components

<table>
<thead>
<tr>
<th>Class / Object Name</th>
<th>Attributes</th>
<th>Methods or operations</th>
</tr>
</thead>
</table>

1. Link: Connection between 2 or more classes
2. Association: Relation between 2 or more classes
   - Roles, Multiplicity (1-1, 1-m, m-n)
3. Aggregation: Hierarchical model
4. Inheritance: Relationship to Root Class to many specialized classes
   - Super Class – Sub Class
3. Interaction Diagrams:
- Capture Communication between objects
- Messages between objects (Solid arrows)
- Timing Diagrams or Sequence Diagrams
- X – Axis Objects Y- Axis Time
4. State Machine Diagrams:

- Describes the sequence of states an object can go through when triggered by an external event.

- Active, Inactive, Closed etc.
5. Activity Diagrams:

- Technique to describe any procedural logic or business process or work flow
- Like flow charts but support parallel behavior
- Forks, Joins, Decisions
- Swim lane diagrams
Questions ?
Project Life Cycle
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>
<tr>
<td>Tech. Documen</td>
<td>Document the System Technical specifications</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)

<table>
<thead>
<tr>
<th>Task</th>
<th>Task Description</th>
<th>Duration in Days</th>
<th>Start</th>
<th>End</th>
<th>September</th>
<th>October</th>
<th>Nov</th>
<th>Dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>Project Setup</td>
<td>5</td>
<td>1-Sep</td>
<td>5-Sep</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 2</td>
<td>Requirement</td>
<td>5</td>
<td>6-Sep</td>
<td>10-Sep</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 3</td>
<td>Analysis</td>
<td>8</td>
<td>11-Sep</td>
<td>18-Sep</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- PERT (Program Evaluation Review Technique)
  - Critical Path
  - Graphical Representation
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 ?