Software Engineering
Session 4 – Main Theme
Requirements Model Engineering
Dr. Jean-Claude Franchitti
New York University
Computer Science Department
Courant Institute of Mathematical Sciences

Agenda

1 Session Overview
2 Requirements Model Engineering
3 Summary and Conclusion
What is the class about?

- Course description and syllabus:
  - http://www.nyu.edu/classes/jcf/g22.2440-001/
  - http://www.cs.nyu.edu/courses/spring13/G22.2440-001/

- Textbooks:
  - Software Engineering: A Practitioner’s Approach
    Roger S. Pressman
    McGraw-Hill Higher International
    - http://highered.mcgraw-hill.com/sites/0073375977/information_center_view0/

Requirements Model Engineering in Brief

- Requirements Engineering Processes
- Tools-Driven Approaches
- Summary and Conclusion
  - Readings
    - Individual Assignment #1 (due)
    - Team Assignment #1 (ongoing)
    - Course Project (ongoing)
The goal of the analysis phase is to truly understand the requirements of the new system and develop a system that addresses them -- or decide a new system isn’t needed.

The line between systems analysis and systems design is very blurry.
Gathering – Information Overview

- Interviews
- Joint Application Design (JAD)
- Questionnaires
- Document Analysis
- Observation

Interviews - Five Basic Steps

- Selecting Interviewees
- Designing Interview Questions
- Preparing for the Interview
- Conducting the Interview
- Post-Interview Follow-up
Selecting Interviewees

- Based on Information Needed
- Often Good to Get Different Perspectives
  - Managers
  - Users
  - Ideally, All Key Stakeholders

Types of Questions

<table>
<thead>
<tr>
<th>Types of Questions</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Closed-Ended Questions</td>
<td>* How many telephone orders are received per day?</td>
</tr>
<tr>
<td></td>
<td>* How do customers place orders?</td>
</tr>
<tr>
<td></td>
<td>* What additional information would you like the new system to provide?</td>
</tr>
<tr>
<td>Open-Ended Questions</td>
<td>* What do you think about the current system?</td>
</tr>
<tr>
<td></td>
<td>* What are some of the problems you face on a daily basis?</td>
</tr>
<tr>
<td></td>
<td>* How do you decide what types of marketing campaign to run?</td>
</tr>
<tr>
<td>Probing Questions</td>
<td>* Why?</td>
</tr>
<tr>
<td></td>
<td>* Can you give me an example?</td>
</tr>
<tr>
<td></td>
<td>* Can you explain that in a bit more detail?</td>
</tr>
</tbody>
</table>
Designing Interview Questions

- Unstructured interview
  - Broad, Roughly Defined Information
- Structured interview
  - More Specific Information

Questioning Strategies

<table>
<thead>
<tr>
<th>Level</th>
<th>Top Down</th>
<th>Bottom Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Level</td>
<td>Very General</td>
<td></td>
</tr>
<tr>
<td>Medium-Level</td>
<td>Moderately Specific</td>
<td></td>
</tr>
<tr>
<td>Low-Level</td>
<td>Very Specific</td>
<td></td>
</tr>
</tbody>
</table>

EXAMPLES?
Interview Preparation Steps

- Prepare General Interview Plan
  - List of Question
  - Anticipated Answers and Follow-Ups
- Confirm Areas of Knowledge
- Set Priorities in Case of Time Shortage
- Prepare the Interviewee
  - Schedule
  - Inform of Reason for Interview
  - Inform of Areas of Discussion

Conducting the Interview

- Appear professional and unbiased
- Record all information
- Check on organizational policy regarding tape recording
- Be sure you understand all issues and terms
- Separate facts from opinions
- Give interviewee time to ask questions
- Be sure to thank the interviewee
- End on time
Conducting the Interview - Practical Tips

- Don’t Worry, Be Happy
- Pay Attention
- Summarize Key Points
- Be Succinct
- Be Honest
- Watch Body Language

Post-Interview Follow-Up

- Prepare Interview Notes
- Prepare Interview Report
- Look for Gaps and New Questions
### Interview Report

<table>
<thead>
<tr>
<th>INTERVIEW REPORT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interview notes approved by: __________</td>
</tr>
<tr>
<td>Person interviewed</td>
</tr>
<tr>
<td>Interviewer</td>
</tr>
<tr>
<td>Date</td>
</tr>
<tr>
<td>Primary Purpose:</td>
</tr>
</tbody>
</table>

| Summary of Interview: |
| Open Items: |
| Detailed Notes: |

---

### JAD: Introduction

- Invented by IBM late 1970s
- Structured Meeting of 10-20 users
- ~30 minutes per agenda item
- frequent breaks
### JAD: Overview

- Selecting participants
- Designing the session
- Preparing for the session
- Conducting the session
- Follow-Up

### JAD Key Ideas

- Allows project managers, users, and developers to work together
- May reduce scope creep by 50%
- Avoids requirements being too specific or too vague
Joint Application Design (JAD) Important Roles

- Facilitator
- Scribe

Joint Application Design (JAD) Setting

- U-Shaped seating
- Away from distractions
- Whiteboard/flip chart
- Prototyping tools
- e-JAD
The JAD Session

- Tend to last 5 to 10 days over a three week period
- Prepare questions as with interviews
- Formal agenda and ground rules
- Facilitator activities
  - Keep session on track
  - Help with technical terms and jargon
  - Record group input
  - Help resolve issues
- Post-session follow-up
Managing Problems in JAD Sessions

- Reducing domination
- Encouraging non-contributors
- Side discussions
- Agenda merry-go-round
- Violent agreement
- Unresolved conflict
- True conflict
- Use humour

JAD : Summary

- Structured Meeting
- Facilitator and scribe + 10-20 users
- Attempts to overcome usual problems with groups
- Only one person talks at once
- Every opinion is valued
Questionnaire Steps

- Selecting participants
  - Using samples of the population
- Designing the questionnaire
  - Careful question selection
- Administering the questionnaire
  - Working to get good response rate
- Questionnaire follow-up
  - Send results to participants

Good Questionnaire Design

- Begin with non-threatening and interesting questions
- Group items into logically coherent sections
- Do not put important items at the very end of the questionnaire
- Do not crowd a page with too many items
- Avoid abbreviations
- Avoid biased or suggestive items or terms
- Number questions to avoid confusion
- Pretest the questionnaire to identify confusing questions
- Provide anonymity to respondents
Document Analysis

- Provides clues about existing “as-is” system
- Typical documents
  - Forms
  - Reports
  - Policy manuals
- Look for user additions to forms
- Look for unused form elements

Observation

- Users/managers often don’t remember everything they do
- Checks validity of information gathered other ways
- Behaviours change when people are watched
- Careful not to ignore periodic activities
  - Weekly ... Monthly ... Annual
Criteria for Selecting the Appropriate Techniques

- Type of information
- Depth of information
- Breadth of information
- Integration of information
- User involvement
- Cost
- Combining techniques

Selecting the Appropriate Techniques

<table>
<thead>
<tr>
<th></th>
<th>Interviews</th>
<th>JAD</th>
<th>Questionnaires</th>
<th>Document Analysis</th>
<th>Observation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of Information</td>
<td>As-Is</td>
<td>As-Is</td>
<td>As-Is</td>
<td>As-Is</td>
<td>As-Is</td>
</tr>
<tr>
<td>Depth of Information</td>
<td>High</td>
<td>High</td>
<td>Medium</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>Breadth of Information</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Integration of Info.</td>
<td>Low</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>User Involvement</td>
<td>Medium</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>Cost</td>
<td>Medium</td>
<td>Low-Medium</td>
<td>Low</td>
<td>Low</td>
<td>Low-Medium</td>
</tr>
</tbody>
</table>
Sub-Section Objectives

- Describe Requirements Model Engineering Activities for a Selected Tools-Driven Approach
Reasoning About Business Entities and Their Dependencies and Goals

Pattern Language Structure for Agent Patterns Selection
(http://www.scs.carleton.ca/~weiss/papers/aois03-revised.pdf)
Use of Sparx Systems EA to Engineer an EAMF Requirements Model

Requirements Model Engineering

- Analysis input: Requirements Model within EA EAMF-Compliant Project Template
- Define Project Business Vocabulary
- Define Requirements Model
- Define Concept Definition
- Requirements Business Entity Requirements
- Project Requirements Model and Enterprise Solutions Catalog
- Enterprise Requirements Model
- Enterprise Products (EAMF-Compliant)
Use of IBM Rational ReqPro for the Requirements Model Engineering Phase

Sparx Systems EA Template

Enterprise Project Reqs Model types are the same as Project Reqs Model Types

Enterprise Project Req Model Docs types are the same as Project Req Model Docs Types
jUCMNav GRL Modeling Constructs
(http://www.scs.carleton.ca/~weiss/papers/MCeTech05.pdf)

(a) GRL Elements

(b) GRL Satisfaction Levels

(c) Link Composition

(d) GRL Links

(e) GRL Contributions Types

jUCMNav GRL Modeling Constructs (continued)
(http://www.jrpit.acs.org.au/jrpit/JRPITVolumes/JRPIT36/JRPIT36.4.259.pdf)

Contribution Links

Correlation Links

Entities

Softgoal

Goal

Task

Resource

Belief
Business Architecture Analysis Using EAMF

Pattern Language Structure for Agent Patterns Selection
**UCM Notations Summarized**

(http://www.usecasemaps.org/pub/sugarloafplop01.pdf)

(a) UCM Path Elements

- **Start Point**
- **Path**
- **End Point**
- **Responsibility**
- **Direction Arrow**
- **Timestamp Arrow**
- **Failure Point**
- **Shared Responsibility**

(b) UCM Forks and Joins

- **OR Fork** & Guarding Conditions
- **AND Fork**
- **OR Join**
- **AND Join**

(c) UCM Components

- **Team**
- **Agent**

- **Waiting Place** (with trigger path)
- **Continuation Path**
- **Timeout**
- **Continuation Figure Path**

(d) UCM Stubs and Plug-ins

- **Static Stub & Segments ID**
- **Dynamic Stub**

(e) UCM Waiting Places and Timers

---

**UCM Notations Summarized (continued)**

(http://www.scs.carleton.ca/~francis/Thesis/phdthesis.pdf)

<table>
<thead>
<tr>
<th>UCM Notation</th>
<th>Notation Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Start</strong></td>
<td>Represents the start of the UCM.</td>
</tr>
<tr>
<td><strong>Path</strong></td>
<td>Connects the start point to the end point.</td>
</tr>
<tr>
<td><strong>End</strong></td>
<td>Represents the end of the UCM.</td>
</tr>
<tr>
<td><strong>Responsibility</strong></td>
<td>Connects start points to responsibilities.</td>
</tr>
<tr>
<td><strong>Direction</strong></td>
<td>Arrow indicating direction.</td>
</tr>
<tr>
<td><strong>Timestamp</strong></td>
<td>Arrow indicating timestamp.</td>
</tr>
<tr>
<td><strong>Failure</strong></td>
<td>Arrow indicating failure point.</td>
</tr>
<tr>
<td><strong>Shared</strong></td>
<td>Arrow indicating shared responsibility.</td>
</tr>
</tbody>
</table>

**OR Fork** and **AND Fork**

- **OR Fork** and **AND Fork**: Represents the point where two or more paths diverge. The path to the left is conditioned, the path to the right is unconditional. The left path is executed only if the condition is met, and the right path is executed if the condition is not met. Segments ID, dynamic stub, and case stub are connected to the OR fork. Segments ID, dynamic stub, and case stub are connected to the AND fork.

**Waiting Place**

- Represents a place where a path is blocked. When a path is blocked, the path waits until it can continue. Waiting places can be triggered by a timer or another path.

**Timeout**

- Represents a timeout event that occurs when a path takes too long to complete.

**AND Fork & AND Join**

- **AND Fork** and **AND Join**: Represents the point where two or more paths converge. The path to the left is executed only if all the conditions are met, and the path to the right is executed if any of the conditions are not met. Segments ID, dynamic stub, and case stub are connected to the AND fork. Segments ID, dynamic stub, and case stub are connected to the AND join.

**Static Stub & Segments ID**

- Represents a static stub that cannot be modified. The path is forced to follow a specific segment. Segments ID are connected to the static stub. Segments ID can be used to control the flow of the UCM.

**Dynamic Stub**

- Represents a dynamic stub that can be modified based on runtime conditions. The path is forced to follow a specific segment depending on the condition. Dynamic stubs are connected to the dynamic stub.

**Case Stub**

- Represents a case stub that can be used to control the flow of the UCM based on runtime conditions. The path is forced to follow a specific segment depending on the condition. Case stubs are connected to the case stub.

---

**Schechter - Friesen**

- Schechter - Friesen is a tool for creating and managing UCMs. It provides a user-friendly interface for creating and editing UCMs. It also supports the creation of UCMs from existing models.
Inter-Scenario Relationships Design Patterns Used in EAMF BPM Approach

Inter-Scenario Relationships

- Scenario Interactions
- Scenario Dependency
- Scenario Clustering
- Concurrent Scenarios

- Mutually Exclusive Scenarios
- Scenario Composition
- Scenario Aborting
- Waiting Place
- Timed Waiting Place

Combining Goal Oriented & Scenario-Based Modeling

(http://www.cs.toronto.edu/km/GRL/from-r2a/fromr2a/straw01.pdf)
Use of Sparx Systems EA to Analyze the Business Architecture

Business Architecture Design Using EAMF
Use of Sparx Systems EA to Design the Business Architecture

From Requirements Engineering to BA Engineering Using EAMF
Composite UCM Root Map for ESLM

Plug-In Map for HandleAllEvents in Root Map
Generic Plug-In Map for HandlePolicyServiceEvents and HandleInvoiceServiceEvents

Generic Plug-In Map for ProcessAllRequests Stub in Root Map
Conclusions: Gathering Information & Tools-Driven Approach

- Gathering Information Involves
  - Interviews
  - Joint Application Design (JAD)
  - Questionnaires
  - Document Analysis
  - Observation
- Tools-Driven Approach Involves
  - Requirements Engineering Tools
  - Users and Goals Elicitation Tool
  - Use Case Maps
Course Assignments

- Individual Assignments
  - Reports based on case studies / class presentations
- Project-Related Assignments
  - All assignments (other than the individual assessments) will correspond to milestones in the team project.
  - As the course progresses, students will be applying various methodologies to a project of their choice. The project and related software system should relate to a real-world scenario chosen by each team. The project will consist of inter-related deliverables which are due on a (bi-) weekly basis.
  - There will be only one submission per team per deliverable and all teams must demonstrate their projects to the course instructor.
  - A sample project description and additional details will be available under handouts on the course Web site.

Team Project

- Project Logistics
  - Teams will pick their own projects, within certain constraints: for instance, all projects should involve multiple distributed subsystems (e.g., web-based electronic services projects including client, application server, and database tiers). Students will need to come up to speed on whatever programming languages and/or software technologies they choose for their projects - which will not necessarily be covered in class.
  - Students will be required to form themselves into "pairs" of exactly two (2) members each; if there is an odd number of students in the class, then one (1) team of three (3) members will be permitted. There may not be any "pairs" of only one member! The instructor and TA(s) will then assist the pairs in forming "teams", ideally each consisting of two (2) "pairs", possibly three (3) pairs if necessary due to enrollment, but students are encouraged to form their own 2-pair teams in advance. If some students drop the course, any remaining pair or team members may be arbitrarily reassigned to other pairs/teams at the discretion of the instructor (but are strongly encouraged to reform pairs/teams on their own). Students will develop and test their project code together with the other member of their programming pair.
Team Project Approach - Overall

- Document Transformation methodology driven approach
  - Strategy Alignment Elicitation
    - Equivalent to strategic planning
      - i.e., planning at the level of a project set
  - Strategy Alignment Execution
    - Equivalent to project planning + SDLC
      - i.e., planning at the level of individual projects + project implementation
- Build a methodology Wiki & partially implement the enablers
- Apply transformation methodology approach to a sample problem domain for which a business solution must be found
- Final product is a wiki/report that focuses on
  - Methodology / methodology implementation / sample business-driven problem solution

Team Project Approach – Initial Step

- Document sample problem domain and business-driven problem of interest
  - Problem description
  - High-level specification details
  - High-level implementation details
  - Proposed high-level timeline
Assignments & Readings

- Readings
  - Slides and Handouts posted on the course web site
  - Textbook: Part Two-Chapter 5
- Individual Assignment (due)
  - See Session 3 Handout: “Assignment #1”
- Team Project #1 (ongoing)
  - Team Project proposal (format TBD in class)
  - See Session 2 Handout: “Team Project Specification” (Part 1)
- Team Exercise #1 (ongoing)
  - Presentation topic proposal (format TBD in class)
- Project Frameworks Setup (ongoing)
  - As per reference provided on the course Web site

Any Questions?
Next Session: Introduction to Software Analysis and Design