Software Engineering

Session 4 – Main Theme
Requirements Model Engineering
Dr. Jean-Claude Franchitti

New York University
Computer Science Department
Courant Institute of Mathematical Sciences
What is the class about?

Course description and syllabus:

- http://www.nyu.edu/classes/jcf/g22.2440-001/
- http://www.cs.nyu.edu/courses/spring14/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)
Icons / Metaphors

- Information
- Common Realization
- Knowledge/Competency Pattern
- Governance
- Alignment
- Solution Approach
Agenda – Requirements Engineering Processes

2. Requirements Model Engineering
   - Requirements Engineering Processes
   - Tools-Driven Approaches
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>
</table>
| **Closed-Ended Questions** | * How many telephone orders are received per day?  
* How do customers place orders?  
* What additional information would you like the new system to provide? |
| **Open-Ended Questions**   | * What do you think about the current system?  
* What are some of the problems you face on a daily basis?  
* How do you decide what types of marketing campaign to run? |
| **Probing Questions**      | * Why?  
* Can you give me an example?  
* Can you explain that in a bit more detail? |
Designing Interview Questions

- Unstructured interview
  - Broad, Roughly Defined Information
- Structured interview
  - More Specific Information
<table>
<thead>
<tr>
<th>Questioning Strategies</th>
<th>EXAMPLES?</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Level</td>
<td>TOP DOWN</td>
</tr>
<tr>
<td>Very General</td>
<td></td>
</tr>
<tr>
<td>Medium-Level</td>
<td>BOTTOM UP</td>
</tr>
<tr>
<td>Moderately Specific</td>
<td></td>
</tr>
<tr>
<td>Low-Level</td>
<td></td>
</tr>
<tr>
<td>Very Specific</td>
<td></td>
</tr>
</tbody>
</table>
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
<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>
<tr>
<td>Summary of Interview:</td>
</tr>
<tr>
<td>Open Items:</td>
</tr>
<tr>
<td>Detailed Notes:</td>
</tr>
</tbody>
</table>
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><strong>Type of Information</strong></td>
<td>As-Is</td>
<td>As-Is</td>
<td>As-Is</td>
<td>As-Is</td>
<td>As-Is</td>
</tr>
<tr>
<td></td>
<td>Improve.</td>
<td>Improve.</td>
<td>Improve.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>To-Be</td>
<td>To-Be</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Depth of Information</strong></td>
<td>High</td>
<td>High</td>
<td>Medium</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td><strong>Breadth of Information</strong></td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td><strong>Integration of Info.</strong></td>
<td>Low</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td><strong>User Involvement</strong></td>
<td>Medium</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td><strong>Cost</strong></td>
<td>Medium</td>
<td>Low-Medium</td>
<td>Low</td>
<td>Low</td>
<td>Low-Medium</td>
</tr>
</tbody>
</table>
Agenda – Requirements Engineering Processes

2. Requirements Model Engineering

- Requirements Engineering Processes
- Tools-Driven Approaches
Describe Requirements Model Engineering Activities for a Selected Tools-Driven Approach
Requirements and Definitions Traceability Graph

- Strategy Definition
- Concept Definition

- Non-Functional Requirements
- Functional Requirements

- Business Use Case Requirements
- Business Model Requirements
  - Location Requirements
  - Organizational Requirements
  - Process Model Requirements
  - Business Rules Requirements
  - Workflow Rules Requirements
Reasoning About Business Entities and Their Dependencies and Goals

Legend:
- : GRL Modeling Constructs

High-Level Business Requirements (bus. obj. & informal reqs)

Business Model Reqs (org, location, process)

Motivated By

Value Flows Between Actors

Modeled by

EAMF Process Patterns

Help Engineer

Refined into

High-Level Tasks

Goals & Subgoals

Have

Actors

Have (Based on Goals Achievement)

Dependencies

Represented In Terms of

Legend:
- EAMF Process Patterns
  - Help Engineer
  - Refined into

GRL Modeling Constructs

: 1

: 2

: 3
Pattern Language Structure for Agent Patterns Selection
(http://www.scs.carleton.ca/~weiss/papers/aois03-revised.pdf)
Requirements Model Engineering Activities

1. Map Business Requirements into Requirements Model
2. Document Traceability between Project Requirements and Req, Model Definitions/Requirements
3. Draw Actor and Dependency Diagram(s)
4. Draw Goal and Task Diagram(s)
5. Document Model Constructs
6. Document Traceability between Modeling Constructs and Req, Model Definitions
7. Populate Requirements Model Categories
8. Document New Requirements Traceability

Legend:
1. ReqPro Doc(s)
2. ReqPro Req(s)
3. ReqPro Traceab. Info*
4. EA Doc Ref(s)*
5. EA Def(s)
6. EA Req(s)
7. EA Traceab. Info*
8. JUCMNav Diagram
9. EAMF Patterns

Domain Models:
- Sparx Systems EA
- Shared File System
- Sparx Systems EA Repository
- EAMF Catalog
- ReqPro Req(s) Repository

Requirements Model:
- Business Analyst
- Req(s) Traceability Info
- Req/
- Def
- Process Patterns
- Business Entity
- Business Use Case
- Organizational
- Location
- Req(s) Traceability Info

Legend:
- <definition> all
- <requirement> some

ReqPro
JUCMNav
Use of Sparx Systems EA to Engineer an EAMF Requirements Model

- **Analysis Input**: Requirements Model Within EA EAMF-Compliant Project Template
- **Definition**: Project Business Vocabulary
- **Definition**: Strategy Definition
- **Definition**: Concept Definition
- **Requirement**: Business Entity Requirements
- **Requirement**: Business Use Case Requirements
- **Requirement**: Business Model Requirements
- **Requirement**: Requirements Definition

**Requirements Model Engineering**

- Project Bus. Vocabulary Definition
- Strategy Definition
- Concept Definition
- Proj. Bus. Directives
- Business Objective
- Features and Events
- Business Use Case Requirements
- Business Entity Requirements
- Location Requirements
- Organizational Requirements
- Business Process Requirements
- Bus/Workflow Rules Reqs
- Process Model Requirements
- Functional Requirements
- Non-Functional Requirements
- Requirements Definition
- Project Requirements Model Categories

**EAMF Pattern, Product and Enterprise Solutions Catalogs**

- Pattern/Product/Enterprise Solutions Catalog
- Enterprise Glossary
- Enterprise Business Rules
- Enterprise Solution Patterns
- Enterprise Strategies
- Enterprise Projects (e.g., Ent. Worker Services)

**EAMF Enterprise Requirements Model**

- EAMF Catalogs and Enterprise Requirements Model Categories

**Enterprise Projects (EAMF-Compliant)**
Sparx Systems EA Template
Use of IBM Rational ReqPro for the Requirements Model Engineering Phase
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
Sample Actor-Dependency Diagram
(Early Requirements Discipline)
Pattern Language Structure for Agent Patterns Selection

Legend:
- GRL Modeling Constructs
- UCM Modeling Constructs

1. High-Level Business Requirements (bus. obj. & informal reqs) Modeled by Business Model Reqs (org, location, process)
2. Motivated By
3. Value Flows Between Actors Represented In Terms of Dependencies
5. Goals & Subgoals Meet High-Level Tasks
6. Have Have (Based on Goals Achievement)
7. Allocated to (based on belief) Refined into
8. Motivates Newly Identified Requirements
9. Modeled by Role Components
10. Allocated to (Teams) Components
11. (Teams) Components
12. Help Engineer
13. Implemented via Low-Level Responsibilities
14. Contain Plug-in Maps
15. Assigned to Couplings Between Paths
16. Applied to Path Segments
17. GRL Modeling Constructs
18. UCM Modeling Constructs

Legend:
- EAMF Process Patterns
- Help Engineer
UCM Notations Summarized
(https://www.usecasemaps.org/pub/sugarloafplop01.pdf)

Figure 27

(a) UCM Path Elements
- Start Point
- Path
- End Point
- Responsibility
- Direction Arrow
- Timestamp Point
- Failure Point
- Shared Responsibility

(b) UCM Forks and Joins
- OR-Fork
- Guarding Conditions
- [C1] [C2] [C3] OR-Join
- AND-Fork
- AND-Join

(c) UCM Components
- Team
- Agent

(d) UCM Stubs and Plug-ins
- Static Stub & Segments ID
- Dynamic Stub
- S{IN1} Plug-in Map
- E{OUT1}

(e) UCM Waiting Places and Timers
- Waiting Path
- Waiting Place
- Trigger Path (asynchronous)
- Continuation Path
- Timer
- Timer Release (synchronous)
- Continuation Path
- Timeout Path
### UCM Notations Summarized (continued)

[Link to PDF](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>Start Point</td>
<td>Basic path. The basic path is the most basic, complete unit. The path represents scenario flow. Paths connect start points, responsibilities, and end points. A path may have any shape as long as it is continuous (can cross itself). The start points represent preconditions or triggering causes. The end points represent post-conditions or resulting effects.</td>
</tr>
<tr>
<td>Do something</td>
<td>Responsibility point. Represents generic processing (actions, tasks, or functions to be performed). Responsibilities may be bound to a component.</td>
</tr>
<tr>
<td>Direction</td>
<td>Direction (optional). In general, the positioning of the start and end points of a path indicate direction. In certain cases, it is useful to show the direction on a complicated map.</td>
</tr>
<tr>
<td>Waiting place</td>
<td>Waiting place. Represents a waiting place along a path. Propagation along the path stops at the waiting place until the trigger arrives. Waiting places can be triggered by a trigger path as shown or by the environment.</td>
</tr>
<tr>
<td>Timer</td>
<td>Timer. A special waiting place that expresses the idea that there is a time limit on waiting. When propagation along the waiting path reaches the timer, the timer is set. Propagation along the continuation path continues if the timer release arrives. Propagation along the timeout path continues if the timeout occurs.</td>
</tr>
<tr>
<td>AND Fork and AND Join</td>
<td>For concurrent paths (two or more).</td>
</tr>
<tr>
<td>OR Fork and OR Join</td>
<td>OR Fork and OR Join. An OR fork indicates that the path proceeds in only one out of two or more directions. Labels may identify alternative paths or guarding conditions. An OR Join indicates a common causal segment of two or more paths.</td>
</tr>
<tr>
<td>Static stub</td>
<td>Static stub. Contains only one plug-in (sub UCM), hence enabling hierarchical decomposition of complex maps.</td>
</tr>
<tr>
<td>Dynamic stub</td>
<td>Dynamic stub. May contain several plug-ins, whose selection can be determined at run-time according to a selection policy (often described with pre-conditions).</td>
</tr>
<tr>
<td>Generic component</td>
<td>Generic component. Represents an architectural entity.</td>
</tr>
<tr>
<td>Slot Placeholder</td>
<td>Slot. Placeholder for dynamic components as operational units. Dynamic responsibilities can move dynamic components from a path into a slot or out of a slot onto a path.</td>
</tr>
</tbody>
</table>

[Button]| Create | Delete | Move out | Move into |
Inter-Scenario Relationships Design Patterns Used in EAMF BPM Approach

- Scenario Interactions
  - Mutually Exclusive Scenarios
  - Scenario Composition
  - Scenario Aborting
- Scenario Dependency
- Scenario Clustering
- Timed Waiting Place
- Concurrent Scenarios

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

Software Development Lifecycle Phases

- Analysis
  - Relationships
  - Entities
    - BUC Mdl
    - Loc. Mdl
    - Org. Mdl
    - URN Mdl
  - Capab. Matrix
  - Bus. Problem Force Hierarchies
  - Candidate Bus Pattern Hierarchies
  - Candidate Bus. Arch. Styles
  - Candidate Reference Projects
  - Bus. Arch. Analysis Artifacts

- Design
  - Domain Model
    - BUC Mdl
    - Loc. Mdl
    - Org. Mdl
    - Process Mdl
    - Bus. Arch. Model
  - Bus. Pattern Hierarchies
  - Reference Business Arch(s)
  - Reference EAMF Project(s)
  - Bus. Arch. Patterns/Reuse Constraints
  - Bus. Arch. Design Artifacts
Use of Sparx Systems EA to Design the Business Architecture

Bus. Arch. Engineering
Analysis

Design

Relationships

Entities

Domain Model

BUC Mdl

Loc. Mdl

Org. Mdl

URN Mdl

Capab. Matrix

Bus. Problem

Force

Hierarchies

Candidate Bus Pattern
Hierarchies

Candidate Bus. Arch. Styles

Candidate Reference Projects

Bus. Arch. Analysis Artifacts

Bus. Arch. Design Artifacts

<view> Analysis within
<perspective> Business Architecture of EAMF-Compliant Project Template

<view> Design within
<perspective> Business Architecture of EAMF-Compliant Project Template

Within EAMF Framework

Within EAMF Enterprise
Solutions Catalog
From Requirements Engineering to BA Engineering Using EAMF
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
Agenda

1. Session Overview
2. Planning and Managing Requirements
3. Summary and Conclusion
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 a 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
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
Next Session: Introduction to Software Analysis and Design