Requirements
Introduction

- Sloppy requirements specification and management is a major risk in software development.
  - Therefore, worth the effort to do very well.

- The UP defines several artifacts:
  - Vision document
  - Use-case model
  - Supplementary Specifications
    - Includes requirement lists and attributes
  - Stakeholder requests
  - UI prototype
  - Use-case UI storyboard
  - Glossary

- Advice: Create every one!
Scope of our Seminar

- We could easily spend 5 days exploring these artifacts.

- However, we will just focus on practicing use case modeling.
  - The others are important.
  - Carefully learn the art of requirements specification and management.

- See the Resources section for good references.
Types of Requirements

- FURPS+ model:
  - Functionality
  - Usability
  - Reliability
  - Performance
  - Supportability

- The "+" in FURPS+:
  - design constraints
  - implementation requirement
  - interface requirements
  - physical requirements
Use Cases

- A use case tells a story of actors using a system.
  - “Rent Videos”
  - A use-case is a sequence of actions a system performs that yields an observable result of value to a particular actor.

- One artifact to express (especially) functional requirements.

- Emphasizes thinking about the valuable objectives-oriented viewpoint of the users.
Supplementary Specifications

1. Functionality
   Requirement list and attributes
2. Usability
3. Reliability
4. Performance
5. Supportability
6. Design Constraints
7. Online User Documentation and Help System Requirements
8. Purchased Components
9. Interfaces
10. Licensing Requirements
11. Legal, Copyright and Other Notices
12. Applicable Standards
Requirement Lists & Attributes

- For those not easily captured in use cases (which only emphasize functional requirements).
  - Fault tolerance, supportability, documentation, . . .

- Includes requirement attributes.

<table>
<thead>
<tr>
<th>ID</th>
<th>Feature</th>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>FEAT-1</td>
<td>Record video rental.</td>
<td>Priority</td>
<td>Must have</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Stability</td>
<td>New forms of media rental, at least 1/year, is likely.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Subject matter expert</td>
<td>Jill Smith</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cross-references</td>
<td>use case &quot;Rent Videos&quot;</td>
</tr>
</tbody>
</table>
Use Cases and Requirement Lists

- **Advice:** Create in parallel.

- Enhance each other.

- Alternate ways to explore, think, talk.

- Use cases are a simple, familiar communication vehicle for non-analysts.
  - Reduces barriers to communication—very important!
Use Cases and Requirement Lists

- Use cases help us focus on
  - the business value of using a system
  - the priorities of development

- If only create requirement lists, it is easier to lose sight of the important elements and their priority in terms of business value.

- Use cases are awkward to record non-functional requirements.
  - The Supplementary Specification is suitable
Tracebility

Key ideas

- Everything has codes. FEA-1, . . .
- Track how requirements trace to other artifacts, components, issues, change requests, tests, . . .

Advice: Use a real requirements management and change request tool.

- Alternative: Use a simple tool such as MS Access to create a database of elements and their relationships.
Resources

- In all cases
  - Rational Unified Process Web Tool has some advice or templates.
  - *The Unified Software Development Process*
  - *(fall 2000) Applying UML and Patterns in the Unified Process*
- Vision document
  - *Practical Software Requirements (A)*
  - *Managing Software Requirements (B)*
- Use-case model
  - *Applying UML and Patterns (C)*
  - *Applying Use Cases*
- Supplementary Specifications
  - A, B, C
  - *Exploring Requirements: Quality Before Design*
- Stakeholder requests
  - A, B
- UI prototype
  - *Usability Engineering*
  - *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design*
- Glossary
  - C