Object-Oriented Design

Lecture 3: Requirements Workflow
Review: Software Development Methodology (SDM)

A framework for applying software engineering practices with the specific aim of providing the necessary means for developing software-intensive systems.

Consisting of two main parts:

- A set of modeling conventions comprising a *Modeling Language* (syntax and semantics)
- A *Process*, which
  - provides guidance as to the order of the activities,
  - specifies what artifacts should be developed using the *Modeling Language*,
  - directs the tasks of individual developers and the team as a whole, and
  - offers criteria for monitoring and measuring a project’s products and activities.
Iterations and Workflows

**Workflows**
- Requirements
- Analysis
- Design
- Implementation
- Test

**Phases**
- Inception
- Elaboration
- Construction
- Transition

**Iterations**
- Preliminary Iteration(s)
- iter. #1
- iter. #2
- iter. #n
- iter. #n+1
- iter. #n+2
- iter. #m
- iter. #m+1

An iteration in the elaboration phase
Use Cases Drive the Iteration Process

Inception | Elaboration | Construction | Transition

Iteration 1 | Iteration 2 | Iteration 3

“Mini-Waterfall” Process
- Iteration Planning
- Rqmts Capture
- Analysis & Design
- Implementation
- Test
- Prepare Release

Sharif University of Technology
Four Steps of Requirements Capture

- List candidate requirements
- Understand system context
- Capture functional requirements
- Capture nonfunctional requirements
What are requirements?

• A requirement is a statement about an intended product that specifies what it should do or how it should perform.

• Goal: To make as specific, unambiguous, and clear as possible
1. List Candidate Requirements

- Prepare a ‘Features’ list:
  - Ideas that customers, users, analysts, and developers think are good for the systems

- Each feature has:
  - Status (proposed, approved, etc)
  - Estimated cost to implement (like man-hours)
  - Priority (critical, important, optional)
  - Level of risk in implementation
2. Understand system context

- **Domain model**
  - Important concepts of the context and relationships among them
  - A glossary of terms for better communication
  - Domain objects later transformed to classes

- **Business Model**
  - Model the business processes of the organization
  - Specify which processes are to be supported by the system
3. Capture functional requirements

- Use case model
  - Each use case describes a way of using the system by a user
  - Use case model contains all the use cases of the system
  - Interview users and customers to collect them
  - This model leads to analysis and design
4. Capture nonfunctional requirements

- **System properties:** environmental and implementation constraints, platform dependencies, reliability, timing constraints.
- **Some nonfunctional requirements are relevant only to a certain use case.**
- **Supplementary requirements:**
  - Nonfunctional requirements that cannot be applied to particular use cases.
Artifacts of requirements workflow: Use case model

- **Actor:**
  - Users who use the system, and
  - external systems that interact with the system

- **Use cases**
  - Flow of events
  - Special requirements

- **Use Case Priorities**

- **Glossary:**
  - Important and common terms used by analysts in describing the system

- **User Interface Prototype**
Example: Use case for shared calendar

A sequence of events to arrange a meeting:
1. The user chooses the option to arrange a meeting.
2. The system prompts user for the names of attendees.
3. The user types in a list of names.
4. The system checks that the list is valid.
5. The system prompts the user for meeting constraints.

More steps...

Note: Number indicates the steps, shows how user and system will interact
Use case diagram example

- Use case diagram for the shared calendar system
- 3 use cases and 2 actors (Using the UML tool)
Main Activity of Requirements Workflow: Capture Functional Requirements

1. Find actors and use cases
2. Prioritize use cases
3. Detail use cases
4. Prototype user interface
5. Structure the use-case model
1. Find actors and use cases -1

- Objectives:
  - Delimit the system from its environment
  - Outline who and what (actors) will interact with the system and what functionality is expected from the system
  - Capture and define in a glossary common terms that are essential for describing the system
1. Find actors and use cases - 2

• Four steps:
  • Finding the actors
    • At least one user who can enact the candidate actor
    • Min. overlap between the roles played by different actors
  • Finding the use cases
    • A use-case should deliver an observable result that is of value to the particular actor – the initiating actor
    • Avoid too small or too large use cases
1. Find actors and use cases

- Four steps:
  - Briefly describing each use case
    - A step-by-step description of what the system needs to do when interacting with the actor
  - Describing the use case model as a whole
    - Use diagrams and descriptions to explain the use-case model as a whole, and how they are related to each other
    - Let the users/customers approve the use-case model through an informal review
2. Prioritize use cases

• The purpose is to provide input to the realization of use cases to determine which need to be developed in early iterations.

• MoSCoW rules are prevalently used for this purpose.
2. Prioritize use cases

<table>
<thead>
<tr>
<th>Letter</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>MUST</td>
<td>Describes a requirement that must be satisfied in the final solution for the solution to be considered a success.</td>
</tr>
<tr>
<td>S</td>
<td>SHOULD</td>
<td>Represents a high-priority item that should be included in the solution if it is possible. This is often a critical requirement but one which can be satisfied in other ways if strictly necessary.</td>
</tr>
<tr>
<td>C</td>
<td>COULD</td>
<td>Describes a requirement which is considered desirable but not necessary. This will be included if time and resources permit.</td>
</tr>
<tr>
<td>W</td>
<td>WON'T</td>
<td>Represents a requirement that stakeholders have agreed will not be implemented in a given release, but may be considered for the future. (note: occasionally the word &quot;Would&quot; is substituted for &quot;Won't&quot; to give a clearer understanding of this choice).</td>
</tr>
</tbody>
</table>
3. Detail use cases -1

• Describe the flow of events for each use case
• Structuring the use-case description
  • Choose a complete basic path from the start state to the end state and describe it in one section
  • Basic path: “normal” path
  • Describe the rest of the paths as alternatives of deviation from the basic path
  • Alternative paths are described in a separate section
3. Detail use cases - 2

• What to include in use-case descriptions
  • Define the start state and end states as precondition and post-conditions, respectively
  • How and when the use case starts and ends
  • The required order of actions
  • Paths of execution that are not allowed
  • Alternative path descriptions
  • System interactions with the actor, explicitly specify what the system does and what the actor does
  • Usage of objects, values, and resources of the system
3. Detail use cases - 3

- Formalizing the use-case description
  - For simple use cases with fewer states, textual description may be used
  - For complex use cases
    - Use Activity diagrams to describe sequence of activities
    - Use state charts to describe the states and transitions between those states
    - Use Interaction Diagrams to describe how the actor (or actors) interacts with the system in the context of the use case
4. Prototype user interface - 1

• Creating a logical user interface design
  • Determine what elements are needed from the user interfaces to enable the use cases for each actor
  • How should they be related to each other
  • What should they look like
  • How should they be manipulated
  • Use sticky notes (for elements) on a whiteboard
4. Prototype user interface - 2

- Creating a physical user-interface design and prototype
  - Sketch the constellation of user interface elements
  - Additional elements may be added to organize the elements (like windows, menus, etc)
  - Each actor should be provided with a well-integrated, easy-to-use, and consistent interface
  - Prototypes may be built for user validation
5. Structure the use-case model

• Identify shared descriptions of functionality
  • The actions that are common to or shared by several use cases (Gen./Spec. Relationships)

• Identify additional and optional description of functionality
  • Identify *Extend* relationships: Additions to a use case’s sequence of actions
  • Identify *Include* relationships between use cases: Commonalities among different use cases
نمودار موارد کاربرد نمونه

شریف مدرسه علوم و فناوری
مورد کاربرد نمونه

<table>
<thead>
<tr>
<th>شماره</th>
<th>توصیف اجمالی</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>کاربر اطلاعات مربوط به ثبت اختراع را وارد سامانه می‌کند تا پس از ویرایش و نهایی کردن آن، آن را برای کارشناسی بفرستد.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>عامل اصلی</th>
<th>عامل فرعی</th>
</tr>
</thead>
<tbody>
<tr>
<td>کاربر عادی</td>
<td>ندارد</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>شرایط اولیه</th>
<th>شرایط نهایی</th>
</tr>
</thead>
<tbody>
<tr>
<td>ورود به سامانه</td>
<td>تشکیل پرونده اختراع</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>نقطه اصلی</th>
<th>نقطه نهایی</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. این مورد کاربر با درخواست برای ثبت اختراع توسط کاربر آغاز می‌شود.</td>
<td></td>
</tr>
<tr>
<td>2. سامانه درخواست ورود اطلاعات مربوط به ثبت اختراع می‌کند.</td>
<td></td>
</tr>
<tr>
<td>3. کاربر اطلاعات مربوط به ثبت اختراع (شامل: مشخصات و چکیده، شرح و سابقه ایده، ادعا نامه، شرح کامل اختراع، فاصله پیوست و مخترعان (حداکثر 3 مخترع)) را وارد سامانه می‌کند.</td>
<td></td>
</tr>
<tr>
<td>4. کاربر تعیین می‌کند که این اختراع در ادامه اختراع دیگر، مرتبط با آن، اصلاح و بهبود آن یا با بخشی از اختراع قبلی است.</td>
<td></td>
</tr>
<tr>
<td>5. کاربر تعیین می‌کند که همه مشارکت هر کدام از مخترعان چه میزان است.</td>
<td></td>
</tr>
</tbody>
</table>

تشکیل پرونده اختراع کاربر در سامانه ذخیره شده است و مخترعان مربوطه می‌توانند آن را ویرایش کنند و همینطور آن را برای کارشناسی ارسال کنند.

نادار

روند جایگذین
مورد کاربرد نمونه

<table>
<thead>
<tr>
<th>تعمیم تشکیل پرونده اختراع: درخواست ایجاد حوزه جدید</th>
<th>مورد کاربرد</th>
</tr>
</thead>
<tbody>
<tr>
<td>شماره 3.1</td>
<td></td>
</tr>
<tr>
<td>کاربر با توجه به اختراع خود و این که حوزه‌ی مورد نظر آن در سامانه نیست درخواست ایجاد حوزه جدید می‌دهد.</td>
<td></td>
</tr>
<tr>
<td>توصیف اجمالی</td>
<td></td>
</tr>
<tr>
<td>کاربر عادی</td>
<td></td>
</tr>
<tr>
<td>عامل اصلی</td>
<td>عامل فرعی</td>
</tr>
<tr>
<td>ندارد</td>
<td></td>
</tr>
<tr>
<td>شرایط اولیه</td>
<td>قطعه اول</td>
</tr>
<tr>
<td>حوزه‌ی مورد نظر کاربر در حوزه‌های سامانه نیست.</td>
<td></td>
</tr>
</tbody>
</table>

روند اصلی

1. کاربر درخواست ایجاد حوزه‌ی جدید می‌کند.
2. سامانه درخواست اطلاعات مربوط به ایجاد حوزه‌ی جدید می‌کند.
3. کاربر اطلاعات مربوط به ایجاد حوزه‌ی جدید را وارد سامانه می‌کند.
4. اطلاعات مربوط به حوزه‌ی جدید روی پرونده‌ای که در حال تشکیل است قرار می‌گیرد.

شرایط نهایی

5. درخواست ایجاد حوزه جدید روی این پرونده قرار می‌گیرد.
Reference