CSE 435
Software Engineering

Oct 12, 2015
Project

What are the project requirements?

In short, the Final Requirements:
- Software Requirements Specification (SRS) document
- demo-able prototype
- presentation
Project Requirements, in detail

Specific deliverables and due dates

- Intermediate Project Assignment - due Oct 28
- Customer meeting - tentatively scheduled for Oct 28
- Prototype version 1 - Nov 11
- SRS document: draft 1 - Nov 16
- SRS document: near final draft - Nov 25
- Prototype version 2, presentation and demo - Dec 7, 9
Intermediate Project Assignment

Due Wednesday, Oct 28 at 10:20 (hard copy!)

1. weekly meeting time and location
2. estimate of person-hours it will take to do this project
3. skeletal web site
4. initial enumerated list of requirements
   ○ hierarchical, as appropriate
5. first draft of use case diagram
   ○ including descriptions of each use case
6. first draft of conceptual domain model (UML entity-relationship diagrams)
7. additional questions for customer
Project Web Site

Skeletal website for each team by Oct 28

Top Level page

- brief project background and description
- team composition, with photos and roles, if appropriate
- links to team members’ personal pages
- link to cse435 course page
Project Web Site, cont.

Publicly accessible links

- original project description (including sample reports or other documents from customer, as needed)
- your SRS document
- your prototype
Project Web Site, cont.

Private information (accessible only to your team and the instructors)

- agendas and minutes of all meetings
- milestones with internal and required deadlines
- access to intermediate drafts of deliverables
- discussion forum between team members
- questions / answers between customers and other resources
more details of the where and how logistics of the websites will be forthcoming (Wednesday)
Object Oriented Modeling

● The OO model closely resembles the problem domain
  ▪ Base your model on the objects in the problem domain

● Iteratively refine the high-level model until you have an implementation
  ▪ Attempt to avoid big conceptual jumps during the development process
Attributes and Operations

Person objects

abstracts to

Person class
Attributes
- name
- age
- height
- weight

Operations
- move
- change-job
Attributes and Operations

Card objects

Card Class

Attributes
height
width
id-number

Operations
issue
change
Characteristics of Objects

- Identity
  - Discrete and distinguishable entities

- Classification
  - Abstract entities with the same structure (attributes) and behavior (operations) into classes

- Polymorphism
  - The same operation may behave differently on different classes

- Inheritance
  - Sharing of attributes and operations based on a hierarchical relationship
The Class Diagrams
Objects

- Something that makes sense in the application context (application domain)
  - J.Q. Public
  - Joe’s Homework Assignment 1
  - J. Q. Public’s driver’s license

- All objects have identity and are distinguishable

- NOT objects
  - Person
  - Drivers license
Classes

● Describes a group of objects with similar properties (attributes), common behavior (operations), common relationships to other classes, and common semantics

● Person
  ▪ J. Q. Public
  ▪ Joe Smith
  ▪ D. Q. Public

● Card
  ▪ Credit card
  ▪ Drivers license
  ▪ Rewards card
Class Diagrams

Class diagram

<table>
<thead>
<tr>
<th>Person</th>
</tr>
</thead>
<tbody>
<tr>
<td>age: integer</td>
</tr>
</tbody>
</table>

Instance diagram

<table>
<thead>
<tr>
<th>D. Q. Public: Person</th>
</tr>
</thead>
<tbody>
<tr>
<td>age= 32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>J. Q. Public: Person</th>
</tr>
</thead>
<tbody>
<tr>
<td>age= 35</td>
</tr>
</tbody>
</table>

Class with attributes

Objects with values

Objects have an identity

Do not explicitly list object identifiers

SSN OK!
### Examples

<table>
<thead>
<tr>
<th>Person</th>
<th>Card</th>
</tr>
</thead>
<tbody>
<tr>
<td>name: String</td>
<td>height: integer</td>
</tr>
<tr>
<td>age: integer</td>
<td>width: integer</td>
</tr>
<tr>
<td>height: integer</td>
<td>thickness: integer</td>
</tr>
<tr>
<td>weight: integer</td>
<td>id-number: integer</td>
</tr>
</tbody>
</table>
### Operations and Methods

- **Transformation that can be applied to or performed by an object**

<table>
<thead>
<tr>
<th>Card</th>
</tr>
</thead>
<tbody>
<tr>
<td>height: integer</td>
</tr>
<tr>
<td>width: integer</td>
</tr>
<tr>
<td>thickness: integer</td>
</tr>
<tr>
<td>id-number: integer</td>
</tr>
<tr>
<td>issue()</td>
</tr>
<tr>
<td>revoke()</td>
</tr>
</tbody>
</table>

- **May have arguments**

<table>
<thead>
<tr>
<th>Shape</th>
</tr>
</thead>
<tbody>
<tr>
<td>height: integer</td>
</tr>
<tr>
<td>width: integer</td>
</tr>
<tr>
<td>rotate(angle: integer)</td>
</tr>
<tr>
<td>move(x: integer, y: integer)</td>
</tr>
</tbody>
</table>
## Object Notation - Summary

<table>
<thead>
<tr>
<th>Class name</th>
</tr>
</thead>
<tbody>
<tr>
<td>attribute-1 : data-type-1 = default-value-1</td>
</tr>
<tr>
<td>attribute-2 : data-type-2 = default-value-2</td>
</tr>
<tr>
<td>attribute-3 : data-type-3 = default-value-3</td>
</tr>
<tr>
<td>operation-1(argument-list-1) : result-type-1</td>
</tr>
<tr>
<td>operation-2(argument-list-2) : result-type-2</td>
</tr>
<tr>
<td>operation-3(argument-list-3) : result-type-3</td>
</tr>
</tbody>
</table>
Associations

- Conceptual connection between classes
  - A credit card is issued-by a bank
  - A person works-for a company

Class diagrams

Instance diagram
Associations are Bi-directional

- There is no direction implied in an association

![Diagram showing associations between Country, City, Person, and Drivers-license]

Country name ➔ Has-capital ➔ City name

Person name ➔ Is-issued ➔ Drivers-license lic.-number: integer
Associations Have Direction

- Unified adds a direction indicator

```
<table>
<thead>
<tr>
<th>Country</th>
<th>Has-capital</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td></td>
<td>name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Person</th>
<th>Is-issued</th>
<th>Drivers-license</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td></td>
<td>lic.-number: integer</td>
</tr>
</tbody>
</table>
```

Inconsistently used!