Object-Oriented Software

A model of the real world

Object-oriented

- Objects encapsulate data and functions.
- The world is modeled by the objects
  - each significant object in the world has an object in the software

Object/Class Model

<table>
<thead>
<tr>
<th>object name</th>
<th>type</th>
<th>dataname</th>
<th>ret-type</th>
<th>fname(param)</th>
</tr>
</thead>
</table>

Why is OO useful?

- Easier to understand
- Encourages re-use
- Easier to maintain
- Why? - because the objects in the software relate to objects in the real world and are thus easier to understand, have cleaner interfaces, are more stable and are more likely to be needed again

OO Requirements Analysis

- Identify the entities/objects in the real world that need to be modeled in the software.
- Many approaches
  - classic - draw object model
  - Ivar Jacobson - use case first
- I suggest starting with the Noun-in-Text approach

Identifying Entities

- Starting with a textual description (e.g. an SOW)
  - list nouns
- using domain knowledge
  - add other entities
- abstract activities
  - levels of abstraction
Active Task 1

Tom is starting a dental practice in a small town. He will have a dental assistant, a dental hygienist and a receptionist. He wants a system to manage the appointments.

When a patient calls for an appointment, the receptionist will check the calendar and will try to schedule the patient as early as possible to fill in vacancies. If the patient is happy with the proposed appointment, the receptionist will enter the appointment with patient name and purpose of appointment.

L2S7 - TTYP – hardcopy with your names – no grade

Active Task 2 – library – 8 min

- Work with one other person and develop the list of objects and the object model (attributes, functions, multiplicities) for the following task. Put your names on the sheet and turn in.
- The library has copies of books and patrons. Patrons can have up to 3 books borrowed at a time. Each book loan has a due date. Patrons can renew borrowed books once.

Use Case Diagram

- Draw important actors
- Add each action

Scenarios

- A description of a sequence of actions

Draw Interaction Diagrams

- Vertical lines for the classes
- Use scenario to put in function calls

Existence Dependency

- A class (parent) can be associated with a lower class (child) if
  - the lower (child) class only exists when the upper (parent) class exists
  - each instance of the lower (child) class is associated with exactly one instance of the upper (parent) class
Measurement is important

Improvement requires knowledge

Time Log - measuring what you do

- Fields
  - start (clock time - min)
  - stop (clock time - min)
  - interruptions (min)
  - categories (reading, studying, etc)
  - units (e.g. pages of text)
- See page in SOS

Example time log

<table>
<thead>
<tr>
<th>start</th>
<th>stop</th>
<th>int</th>
<th>delta</th>
<th>category</th>
<th>units</th>
</tr>
</thead>
<tbody>
<tr>
<td>8/21</td>
<td>16:40</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8/21</td>
<td>21:05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8/22</td>
<td>09:10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TL1 (time log) assignment

- Start today
- Turn in Thursday, Sep 5 at start of class
- Name and lab time must be on page
- Include date, hours, min start and stop, delta, interruptions, categories, units
- Determine your average pages/minute
- Pressman pp1-52, and any other course related reading