Computer Science 2704

Object-Oriented Software Design and Construction with C++

William D McQuain
Fall 1999

Based on course slides of Todd Stevens, Spring 1999

---

Beginning

- Overview of the course
- Lab Accounts and Software
- Progression of Roles
- First Topics
  - abstraction
  - separation
The URL for the class web page

http://courses.cs.vt.edu/~cs2704/fal1999/

Lab accounts and software

- If you do not have an account for the 116/118 lab, fill out the form being distributed (you must have a university PID).

- The software for the course is installed on all of the lab machines and should be used for your course work. You may also set up the software on your own machine.
**Design Strategies in Object Oriented Programming**

- **Abstraction**
  - modeling essential properties
- **Separation**
  - treat what and how independently
- **Composition**
  - building complex structures from simpler ones
- **Generalization**
  - identifying common elements
**Connections among Strategies, Structures and Goals**

**Abstraction**

- Modeling entities in software
- Only essential aspects should be captured
  - attributes
  - behavior
Artistic Abstraction

Wassily Kandinsky, Cossacks, 1910-11

Practical Abstraction

Critique??
Abstraction

A named collection of attributes and behavior relevant to modeling a given entity for some particular purpose.

Car:

Properties of a Good Abstraction

- well named
- coherent
- accurate
- minimal
- complete
Good Abstractions

Mapping Abstraction to Software

<table>
<thead>
<tr>
<th>real-world</th>
<th>abstraction</th>
<th>software</th>
</tr>
</thead>
<tbody>
<tr>
<td>entity</td>
<td>attributes</td>
<td>{data, data,…}</td>
</tr>
<tr>
<td>behavior</td>
<td></td>
<td>{method, method,…}</td>
</tr>
</tbody>
</table>
In object-oriented programming, the independent specification of an interface and one or more implementations of that interface.

What is to be done

vs

How it is to be done
Separation of Interface from Implementation

Interchangeability of Implementations
Mapping Abstraction to Software

<table>
<thead>
<tr>
<th>real-world</th>
<th>abstraction</th>
<th>software</th>
</tr>
</thead>
<tbody>
<tr>
<td>entity</td>
<td>attributes</td>
<td>{data, data,…}</td>
</tr>
<tr>
<td>behavior</td>
<td></td>
<td>{method, method,…}</td>
</tr>
</tbody>
</table>

General Structure of a Class

className

public

{data, data, ….}

private

{method,method, …}
General Structure of an Object

Multiple Instances of a Class
Software Engineering Goals

Object-Oriented Software Structures
- objects
- classes
- inheritance
- templates
- design patterns

Software Engineering Goals
- reusability
- extensibility
- flexibility