CS2704: Object-Oriented Software Design

Topic 2: Abstraction and Separation
Dr. Ben Keller
Fall 1999

Outline
• Abstraction
• Separation
• Identifying objects and classes

Abstraction
• General concept: describing an entity in terms of its aspects (qualities or features)
• Modeling entities in software
• Only essential aspects should be captured
  – Attributes
  – Behavior
• What is essential depends on situation

Example Abstraction

MEN WOMEN

A “Passenger” Abstraction
• Abstraction of a passenger for a flight reservation system
Attributes: Behaviors:
Properties of a Good Abstraction

- well named – clearly identifies abstraction
- coherent – sensible description
- accurate – only attributes of entity
- minimal – no irrelevant attributes
- complete – everything needed

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>methods</td>
<td>{method, method,…}</td>
</tr>
</tbody>
</table>

Mapping Abstraction to a Class

```
className
```

```
public
{data, data, …}
```

```
private
{method,method, …}
```

Separation

- Separation of what a component does from how it does it (ex. a procedure)
- Define classes by independently specifying the interface for objects in that class, and the implementations of that interface

Separation of Interface and Implementation

```
Interface
```

```
Implementation
```

Interchangeability of Implementations

```
implementation 1
implementation 2
```
Interchangeability of Implementations

General Structure of a Class

General Structure of an Object

Multiple Instances of a Class

Identifying Objects and Classes

- Study features of system
- Look for nouns (people, places, things)
- Example features:
  - “add course grade to student record”
  - “enter rental equipment description”
  - “add frequent flyer miles to customer record”

Good Classes

- Class should represent a set of objects although sometimes only use one
- Behaviors (methods) of class should be meaningful
  - Should a chess piece move itself?
  - What behaviors does a piece of “Data” have?