CS 535 Object-Oriented Programming & Design
Spring Semester, 2003
Doc 16 Some Heuristics

References

Object-Oriented Design Heuristics,Riel

Reading

Object-Oriented Design Heuristics, Chapter 2. Chapter 5 sections 1-4, 6-12.

Some Heuristics

Exceptions

What is the point of Exceptions?

When to use Inheritance?

When to use Composition?

5.1 Inheritance should be used only to model a specialization hierarchy

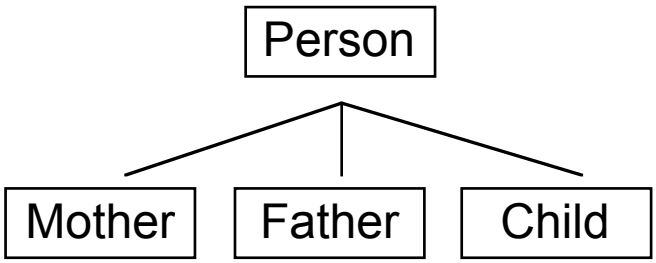
5.12 Explicit case analysis on the type of an object is usually an error. Use polymorphism in most of these cases

```
if x is type A do foo
else if x is type B do foo2
else if x is type C do foo3
else do foo4
```

verses

```
x doFoo
```

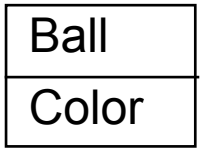
2.11 Be sure the abstractions you model are classes and not simply the roles objects play



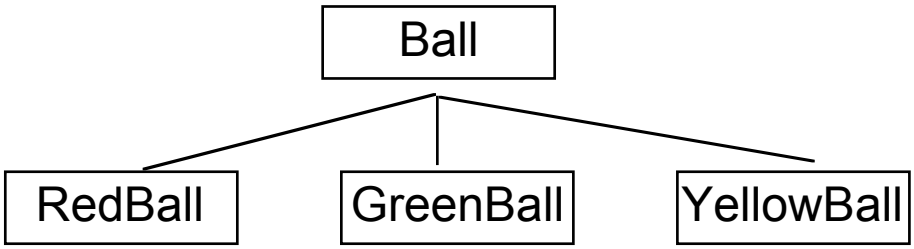

```
initialize
   mother := Mother new.
   father := Father new.
etc.
```


```
initialize
   mother := Person new.
   father := Person new.
etc.
```

5.13 Explicit case analysis on the value of an attribute is often an error. The class should be decomposed into an inheritance hierarchy, where each value of the attribute is transformed into a derived class



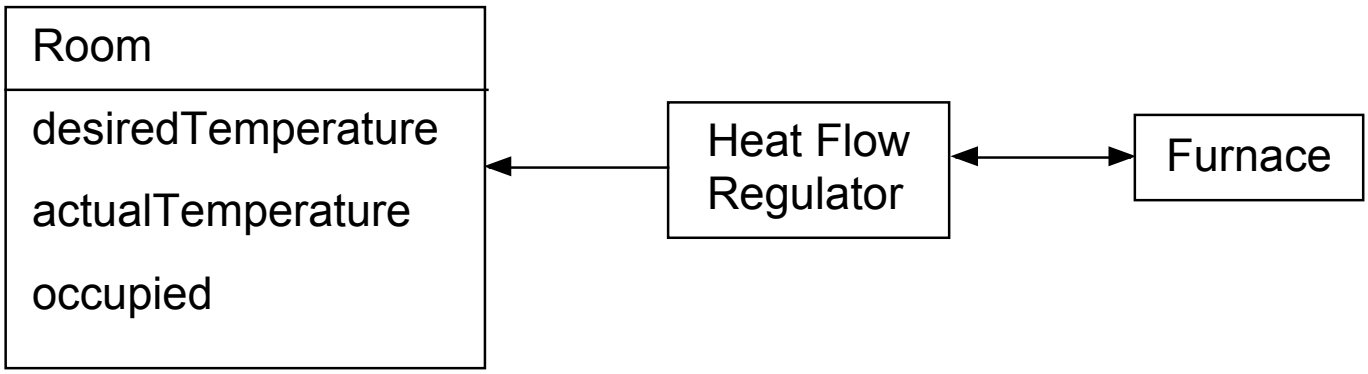
Verses



5.15 Do not turn object of a class into a subclass of the class. Be suspicious of any subclass for which there is only one instance.

3.1 Distribute system intelligence horizontally as uniformly as possible

Top-level classes should share the work uniformily




```
HeatFlowRegular>>someMethod
   room  isOccupied &
   (room desiredTemperature > room actualTemperature)
      ifTrue: [ blah]

HeatFlowRegular>>someMethod
   room  needsHeat ifTrue: [ blah]
```

3.2 Do not create god classes/objects in your system.

Be suspicious of a class whose name contains

- Driver
- Manager
- System
- Subsystem

3.9 Do not turn an operation into a class

5.2 Subclasses must have knowledge of there superclass by definition, but base classes should not know anything about their derived classes.

5.3 All data in a parent class should be private; do not use protected data.

5.4 In theory, inheritance hierarchies should be deep.

5.5 In practice, inheritance hierarchies should be no deeper than the average person can keep in their short-term memory.

A common value for this depth is six.

5.9 If two or more classes share only common data (no common behavior) then that common data should be placed in a class that will be contained by each sharing class.

5.10 If two or more classes have common data and behavior then those classes should each inherit from a common parent class that captures those data and methods.

5.11 If two or more classes share only a common interface, then they should inherit from a common parent class only if they will be used polymorphically