Software Engineering and OOP

Tuesday, 4 December 2018 9:02 AM

Software Engineering

The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software.

- Software is everywhere around us - powerful yet fragile
- Unrealistic industry expectations
- Dunning-Kruger: you don't know what you don't know
- Brook's list of 'Woes of the Craft'

Object-Oriented Programming (OOP)

- is a programming paradigm
- Groups operations and data into modular objects
- Network of interacting objects - state (data) and behaviour (operations on data)

Paradigm	Programming Languages
Procedural Programming paradigm	C
Functional Programming paradigm	F#, Haskel, Scala
Logic Programming paradigm	Prolog

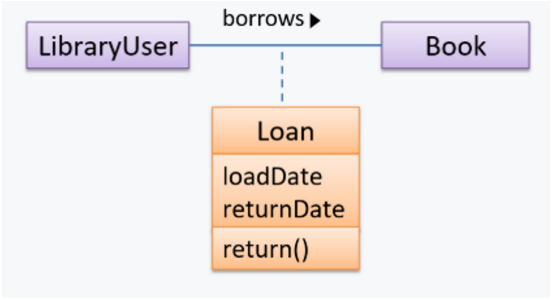
- Example: Java
  - Varargs: int... I
  - JAR: Java archive, java -jar

Objects

- Interface vs implementation
- Abstraction and encapsulation (packaging together, information hiding)
  - Encapsulation strengthens abstraction
- Instructions contained in CLASS
  - Class-level members: attributes and methods
- Enumeration = fixed set of constants

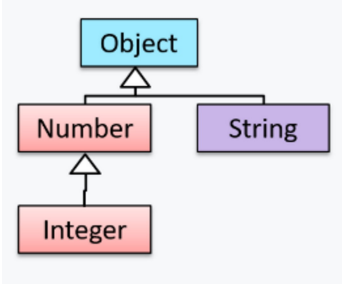
Associations

- Can change over time or be generalised in classes
- Multiplicity
  - Optional: 0..1
  - Compulsory: 1
- Dependency: other object not kept as parameter, but used in method
- Composition vs Aggregation
  - Composition = whole-part (e.g. Board-Square)
    - Part cannot exist without whole
    - Auto-deletion does not imply composition
  - Aggregation = container-contained (e.g. Club-Person)
- Association classes



Inheritance

- Superclass and subclass (sub-type is-a relationship)
- Inheritance hierarchies/ trees



- Overriding: changing behaviour from parent/ ancestor class
    - Overloading: different method signature - type sequence of params
- Interface = behaviour specification, can be implemented (is-a)
- Abstract classes - cannot be instantiated, can be subclassed
    - Abstract methods => abstract class
- Substitutability (every Boy is a Person)
- Dynamic and static binding
    - Overriden methods = dynamic binding at runtime
    - Overloaded methods = static binding at compile time

Polymorphism

- Different objects respond differently to identical messages
- Substitutability, operation overriding, dynamic binding