Software Design

Software design and complexity:

- scale
- environment (I/O, Network)
- infrastructure (libraries, frameworks)
- evolution (design for change)
- correctness (testing, analysis tools, automation)

software qualities:

- sufficiency/func correctness
- robustness
- flexibility
- reusability
- efficiency
- scalability
- security

A simple process:

- Discuss the software that needs to be written
- Write some code
- Test the code to identify the defects
- Debug to find causes of defects
- Fix the defects
- If not done, return to first step

design tips:

- Think before coding
- Consider quality attributes (maintainability,extensibility, performance)
- Consider alternatives and make conscious design decisions

Preview: The design process:

- ObjectOriented Analysis
  - Understand the problem
  - Identify the key concepts and their relationships
  - Build a (visual) vocabulary
  - Create a domain model (aka conceptual model)
- ObjectOriented Design
  - Identify software classes and their relationships with class diagrams
  - Assign responsibilities (attributes, methods)
  - Explore behavior with interaction diagrams
  - Explore design alternatives
  - Create an object model (aka design model and design class diagram) and interaction models
- Implementation
  - Map designs to code, implementing classes and methods

Objects:

- A package of state (data) and behavior (actions)
- Can interact with objects by sending messages
  - perform an action (e.g., move)
  - request some information (e.g., getSize)
- Possible messages described through an interface

---

sources:

- `http://www.cs.cmu.edu/~charlie/courses/15-214/2015-fall/index.html#schedule`