Lecture 6: Design

CS190: Software System Design

February 6, 2002
Steven P. Reiss

I. Today's Class

A. How to design a large software system

1. **Techniques and approaches**
2. **Appropriate guidelines**

B. This is much easier to do with an example

1. **As you start developing we will get back to this**

C. Again contrast standard SE versus XP

II. Design in the large

A. What is different about design in the large

1. **1 person vs 4 person vs 8 person**
2. **How do we change our emphasis**
3. **What is essential**

B. What is different about design for XP

1. **Partial design**
   a) Don't know what is in the future
   b) Don't have a full understanding
2. **Support for joint ownership**
   a) How to design to support this

C. Objectives

1. **Break the system into independent pieces**
   a) Separation of concerns
   b) Small number of them at the top level
   c) Each piece only needs to know the other top-level pieces
   d) Each piece should be as self-contained as possible
   e) Each piece should isolate future changes
   f) Each piece should isolate implementation dependencies

2. **Create well-defined INTERFACES**
    a) Interfaces are the key to a good design
    b) These determine how the pieces interact
    c) These provide the basis for designing implementations
    d) Interface should not reflect potential implementations
    e) Get these right and the rest of the system follows
3. **Design with the future in mind**
    a) Things are going to change (especially with XP)
    b) Try to encapsulate any implementation decisions
    c) Try to encapsulate anything that might change
    d) Design for extensibility and modifiability
4. **Keep things as SIMPLE as possible**
    a) You will want to change things later on
    b) Especially important with XP

III. Starting a Design

A. This is often the hardest part -- getting started

B. Start with the system model from specifications

1. **This is why you want to build one from your first set of stories in XP**
2. **Break the system into components/modules based on the top-level system model**
3. **Limit the number of modules to 5-10**

C. Go through specifications and create interfaces

1. **For each module determine what external operations are needed**
    a) External operations from the user
    b) External operations from other modules
2. **For each module determine what it needs from other modules**
    a) Go through its operations to determine this
3. **Note that this is an interative process**

D. XP vs. standard design

1. **In standard design you try to get the interfaces as complete as possible**
2. **In XP you try to develop an initial interface that is as simple as possible but can be extended later on**

IV. Design Techniques

A. Core + Extensions

1. **Try to develop a small core that people use**
2. **Everything else is done as extensions**
3. **Each extension only interacts with the core**
4. **Why is this good**
5. **How this fits into XP**

B. Risk-based Design

1. **Determine what are the riskiest parts of the system**
   a) Generally what you don't understand how or what
   b) Possibly performance, etc.
2. **Either design for or around these parts**
   a) Better to design around by encapsulation
   b) But you might have to redesign the system to accommodate if this is essential and can't be achieved otherwise
   c) Note that this is what the spiral method is all about
3. **For XP, it would probably good to identify risky stories early on**
   a) To be sure they can be implemented

C. Interface Design

1. **Concentrate on making the interfaces as simple as possible**
   a) Minimum number of methods
   b) Minimum number of arguments passed
   c) Java-style interfaces, abstract classes and factories
2. **Work in terms of interfaces until they are correct**

3. **Other components only call interfaces, never implementations**
4. **For XP, you probably want to maintain interfaces as well**
    a) Implement as needed

V. Design Guidelines

A. SIMPLIFY

B. Interfaces first and foremost

C. Continually improve your design

D. Encapsulate all important decisions

E. Minimize communications paths

F. Document all decisions

G. Keep organized and focused

H. Minimize risk