Lecture 5: Specifications

CS190: Software System Design

February 1, 2002
Steven P. Reiss

I. Today's Class

A. Specifications in classical software engineering

B. Presentation guidelines for Monday

C. Any remaining time will be allocated to groups

II. Specifications

A. Purpose

1. Location in the software engineering framework

2. Purpose -- to decide exactly what will be built

3. Note difference from requirements

a) Requirements -- user's point of view

b) Requirements define the problem to be solved

c) Specifications -- programmer's point of view

d) Specifications -- define the solution

B. Goals

1. Precise definition of what will be done

a) Doesn't generally define how it will be done however

2. Defines the scope of the project

3. Provides a system model

4. Provides a basis for design

5. Provides a basis for cost/time estimation

6. Ensure that the system is buildable

7. Ensure that what will be built is what the user wants

a) Specifications given back to the user for verification

C. Contents

1. Specifications provide a model of the software

2. Model consists of interacting components

- a) Each component has a task (or set thereof)
- b) Set of commands
- c) Description of what it does
  - (1) Including both normal and error modes

3. Modeling a system

- a) Done using similar technologies to requirements
- b) Data flow diagrams, use case diagrams, state transition diagrams, natural language

D. Developing a specification

1. First enumerate all the inputs and outputs

- a) User, other systems, databases, ...

2. Next enumerate all the system functionality

- a) Describe each command of the system
  - (1) How it maps inputs to outputs
- b) Describe any VISIBLE system states
- c) Describe any IMPLIED system data structures

3. Then build a diagramatic model of the system

- a) Data flow is typically best
- b) Start with nodes for inputs, outputs, internal data
- c) Add action nodes to indicate appropriate transformations

4. Iterate building/modifying this model

- a) Ensure all listed actions/commands are covered
- b) Ensure error behavior is well defined in all cases

5. Annotate the diagram (fill in the details)

- a) Each data node represents a data object (class)
  - (1) List the implied methods and what they do (pseudo code or natural language)
- b) Each action node represents an control object (class)
  - (1) Each command yields a method
  - (2) Implicit methods for mappings
  - (3) List all methods and what they do
- c) Ensure each arc is meaningful

        (1) What methods use is

        (2) What information is passed

    d) Add additional annotations with any other requirements

        (1) Non-functional requirements go here

E. Specification vs. Design

1. Specification is not a system design

a) You're trying to describe what the system does, not how

2. Keep the specification at a high level

a) No more that 10 items in the first diagram

b) Use subdiagrams where necessary

c) Do not get into fine grain details, just provide enough

    (1) You would be confortable designing the component

    (2) You have confidence the component can be built

    (3) You have confidence the system will work

3. The specification won't be right the first time

a) Won't include everything

b) Adding things might yield a better approach

4. Changes here are simple and cheap

a) Much easier that in code or even detailed design

b) Save a lot of work by getting this right

5. Keep the specification SIMPLE

a) Ask if each object is really necessary to system understanding

b) Favor data objects over control objects

6. Non-functional specifications

a) Performance criteria

b) Reliability and recovery criteria

c) Ease of use

d) Portability

e) Manuals and other documentation

f) Interactions with other systems

g) Breakdown of effort, cost estimates

F. User Interface Specifications

1. Part of the specifications should be the UI

a) Should be fully specified

b) Should be able to write user manual at this point

2. Generally what you include is

a) Pictures of any visual interfaces (screen layout, menus, dialogs boxes, etc.)

b) Syntax for any command languages

c) Samples of expected outputs

d) Description of what each command does

e) Description of how errors are handled

III.Specifications and XP

A. XP does specifications via stories

1. The initial story is the user requirement

2. The conversation and description that goes with the story describes what the system does to meet this requirement

a) This is effectively the specification

3. Note that there is no overall formal system model

a) You will probably need one in any case

b) But note that it will be defined dynamically over time

4. You should have as many stories as possible before you start building the system

a) The overall model is a way of ensuring the answers and descriptions in the stories are CONSISTENT

B. XP does specifications via prototyping

1. Emphasis on building something to see how it works

a) Try out multiple user interfaces, multiple approaches to a problem

b) Choose the one that works best

2. Understand that what the system will do will change over time

IV. Requirements Presentations (Monday)

- A. Statement of problem to solve
  - 1. Provide a concise motivation for your project
  - 2. Who is the customer
  - 3. What is their need
- B. Provide a small sample of stories
  - 1. Give a feel for what the system will be used for
  - 2. Give a possible solution to that story in terms of the system
    - a) Explain how the system will address the story
    - b) Why does this meet the user’s requirments
- C. Act as a salesman
  - 1. You want to convince me that this is a worthwhile project
  - 2. You want to convince the class that they want to work on this project
    - a) Teams aren’t fixed at this point
    - b) You might be able to or want to attract others to work on your project
  - 3. If you want, a team can propose multiple projects at this point

V. Specifications Presentations (Following week)

- A. Refined requirements presentation
  - 1. Larger set of stories
- B. System model should be included
  - 1. System model as a basis for consistent answers to the stories
  - 2. Be able to describe how the story would be implemented in terms of the system model
  - 3. This should be done at a high level
- C. Think of selling the customer on your solution