CSC 484 Lecture Notes Week 3, Part 2

Details of the ID Process

I. **Relevant reading.**

A. Textbook Chapters 10 and 11

B. Papers of the fortnight:

- *Storyboarding Best Practices*
- *Use Cases and Scenarios*

II. **Overview of Book Chs 10, 11.**

A. Coverage of familiar SE territory:

1. Requirements analysis.
2. User-level design.
3. Prototyping.

Overview of Chs 10, 11, cont'd

B. ID goals fully in line with SE.

1. Understand what users need.
2. Construct prototype to engage users.
3. Evolve prototype into design & imple'n.
4. Iterate as necessary.

Overview of Chs 10, 11, cont'd

C. See Figure 1.

III. Intro to Requirements Analysis (Sec 10.1)

A. Definition of user requirements.

B. Importance of gathering requirements.

C. Techniques to gather requirements.

D. Different requirements representations.

IV. What, How, and Why? (Sec 10.2)

A. *Precisely* the same goals as in SE:

1. Capture requirements sufficiently well to start design.

2. Don’t let fluctuating requirements slow down the process.

What, How, and Why?, cont'd

B. Goals achieved in two ways:

1. Focus on requirements first, postponing time-consuming design ***(traditional)***.

2. Focus on small requirements pieces, each with rapidly doable design ***(agile)***.

What, How, and Why?, cont'd

C. The ultimate in agility ...

1. Be agile enough to know when to go traditional.

2. When integrated design/imple'n impedes progress, focus on requirements alone.

V. What are requirements? (Sec 10.3)

A. More review of SE topics.

B. Definition of a requirement is a

statement of fact.

VI. **Different kinds of rqmts (Sec 10.3.1)**

A. ***Functional*** **--** what the artifact does.

B. ***Non-Functional*** **--** characteristics of the artifact, its development, its users.

Different kinds of requirements, cont'd

C. Book's treatment is comparable to SE treatment, *plus usability*.

D. I.e., add *usability goals* and *user experience goals* to non-functional requirements.

VII. Data gathering methods (Sec 10.4)

A. *Interviews.*

B. *Focus groups.*

C. *Questionnaires.*

D. *Direct and indirect observation.*

E. *Studying domain-specific documentation.*

F. *Researching similar products.*

VIII. Contextual Inquiry, Other Guidelines

A. Sec 10.4.1 not particularly useful.

B. Sec 10.4.2 repeats what's been said.

IX. Analysis, interpretation, presentation (Sec 10.5)

A. Very cursory treatment of SE topics.

B. Volere shell is yet another rqmts notation.

X. Task description, analysis (Sec 10.6, 10.7)

A. Again, all well-known SE techniques.

B. Important amendments to book's coverage:

Task description, analysis, cont'd

1. Storyboards can come first.
2. Scenarios have *both* pictures and prose.
3. Prototyping can commence *without* formal modeling, e.g., without UML use cases.

XI. **Design, prototyping, construction. (Ch 11).**

A. Again, very familiar SE territory.

B. *Conceptual design* is pretty fluffy.

C. Discussion of storyboarding, prototyping.

D. Punts on concrete design and construction.

XII. **Prototyping and construction (Sec 11.2).**

A. What is a prototype? (Sec 11.2.1)

1. Reduced-functionality version of product.

2. Allows user to interact.

Prototyping and construction, cont'd

B. Why Prototype? (Sec 11.2.2)

1. Helps stakeholders understand the product.

2. Helps achieve "full user engagement".

XIII. **Low-fidelity prototyping (Sec 11.2.3).**

A. Doesn’t look much like finished product.

B. Simple, cheap, quick.

Low-fidelity prototyping, cont'd

C. Storyboards may be low-fidelity prototype.

1. Don't afford significant interaction.

2. Interaction can be fundamental to prototype.

Low-fidelity prototyping, cont'd

D. Storyboarding activities include

1. *Sketching* -- embrace the wonders of clip art.

2. *Index cards* -- not my style, but if it works ...

3. *Wizards of Oz* -- humans sitting behind the scenes to simulate prototypical behavior.

XIV. **High-fidelity prototyping (Sec 11.2.4).**

A. Looks much like the finished product.

B. Recall balancing act --

1. Build prototype rapidly.
2. Include what user cares about.
3. Leave out time-consuming imple'n details.

XV. Compromises in prototyping (Sec 11.2.5).

A. *Horizontal prototyping*
-- lots of functions, little detail.

B. *Vertical prototyping*
-- much detail, few functions.

XVI. **Construction (Sec 11.2.6).**

A. One half page for many other courses.

B. Construction not focus of this text.

Construction, cont'd

C. Discussion of "Dilemma" pg 539.

1. Distinction between throw-away and evolutionary is important.

2. However, truly evolutionary prototypes make sense for software, not much else.

XVII. **Conceptual design (Sec 11.3).**

A. By the book's admission, there is no definitive characterization of a conceptual model.

B. Neither is there a single specific artifact.

Conceptual design, cont'd

C. Elements of conceptual model in:

1. *The general user requirements*
2. *Non-functional requirements*
3. *Scenarios, story boards, prototypes*
4. *Concrete design, implementation*

Conceptual design, cont'd

D. Conceptual design & metaphor summed up:

1. Present ideas understandable to users.

2. Explore alternative forms of interaction.

Conceptual design, cont'd

E. Other chapters present specific guidelines.

- in particular chapters 3 & 6
- we'll be getting there soon

XVIII. **Concrete design (Sec 11.4).**

A. As with construction, the book punts.

B. Earlier chapters provide some specifics.

C. Otherwise, the book defers to others.

XIX. **Scenarios in design (Sec 11.5).**

A. Well-established ideas in SE.

B. To reiterate earlier point,
scenarios include both pictures and prose.

XX. **Prototypes in design**

A. Again, well-established ideas in SE.

Prototypes in design, cont'd

B. ***Generating storyboards from scenarios (Sec 11.6.1).***

1. I think it's mostly the other way around.

2. But do whatever works for your team.

Prototypes in design, cont'd

C. ***Generating card-based prototypes (Sec 11.6.2).***

1. I think these are utterly anachronistic.

2. But again, do what works.

D. *Prototyping physical design (Sec 11.6.3).*

1. SEs do this plenty, particularly agile ones.

2. Evolve rough ideas into concrete ideas.

3. E.g., UI sketches -->
   UI screens -->
   working UI prototypes.

XXI. **Tool support (Sec 11.7).**

A. Plenty of code-level tools out there.

B. Also purely prototyping tools, e.g., Flash.

C. DENIM, others, have some interesting ideas.

Tool support, cont'd

D. Still no integrated tool for

- storyboarding
- prototyping
- design
- implementation