System Requirements

CSE 403, Winter 2005
Software Engineering

http://www.cs.washington.edu/education/courses/403/05wi/

System Requirements

- Essential features of the system
  » defined at a level appropriate to the spin cycle
  » capabilities, interfaces, reliability levels, appearance
  » Easy to change early on, grows increasingly more difficult
- Customer’s involvement very important
  » they know the domain of interest far better than you do
  » what fits with their daily work and life patterns
  » what might the future bring
- Neither you nor the customer know everything
  » try to build joint ownership of the process
  » open communication can make change more acceptable

What does the customer want?

- “Better products for free”
  » Scott Adams
- Many customers exist for any single product
  » purchaser, user, user’s management, support, etc
- Write down attributes of expected user set
  » Who they are
  » What they need
  » What they think they need
  » What they want

References

- References
  » The Mythical Man-Month, Brooks
  » Chapter 7, Before the Project, The Pragmatic Programmer, Hunt & Thomas
  » Structuring Use Cases with Goals, A. Cockburn
    • http://alistair.cockburn.us/crystal/articles/alistairsarticles.htm
  » Use cases in theory and practice, A. Cockburn
    • http://alistair.cockburn.us/crystal/articles/alistairsarticles.htm
Attributes have a distribution

- Attributes of the user set are distributions
  - many possible values
  - each value with its own frequency
- The design will not meet all requirements of all members of the user set all the time
  - Postulate a complete set of attributes and frequencies
  - Develop complete, explicit, shared description of users
  - It is better to be explicit and wrong than to be vague

“Complete” Requirements

- You want to write down every requirement for every user of every aspect of the program
  - It’s not possible, there isn’t enough time or money
- You have to find a balance
  - comprehensible vs. detailed correctness
  - graphics vs. explicit wording and tables
  - short and timely vs. complete and late
- Different approaches for different parts are okay

Modularity, not a “pile of paragraphs”

- Split the information by point of view and adapt the documentation style as appropriate
  - Business functions
    - top level mission of application (text, graphics, Flash?)
    - specific functions that must be implemented (use case)
  - Context
    - drawings, text, references to interface standards
  - User Interface
    - text goals, sample layouts, some prototypes
  - Performance and Reliability
    - text goals, specific metrics for space, time, CPUs, ...

Concise is nice

- All the details are necessary at some point
  - but only some of the details are relevant right now
- Arrange the requirements so that the reader can drill down in areas of interest without having to pick out the details from chaos
  - Data flow graphics for top-level orientation
  - Tabular presentation of specific metrics
- The lower the level, the more structured
  - eg, Scenarios vs. Use Cases
Use Cases

• Use cases address “how to make functional requirements readable, reviewable”
  » As an expression “use case” is immediately attractive because the term implies “the ways in which a user uses a system”
• “I have personally encountered over 18 different definitions of use case”, A. Cockburn
• “True use cases are textual descriptions, with a hierarchy and cross-links.”, Hunt & Thomas

Use case dimensions

• Purpose
  » To gather user stories, or build requirements?
    • values are stories, or requirements
• Contents
  » Consistent, or can they be self-contradicting?
    • contradicting, consistent prose, formal content
• Plurality
  » Does a use case contain more than scenario?
    • 1 or multiple
• Structure
  » Informal structure or formal structure?
    • unstructured, semi-formal, formal structure

One choice

• Consistent, semi-formal documentation of requirements
  » Purpose = requirements
  » Contents = consistent prose
  » Plurality = multiple scenarios per use case
  » Structure = semi-formal

What is a use case?

• Sequence of interactions between the system under discussion and its external actors, related to a particular goal
  » An action connects one actor’s goal with another’s responsibility
  » An interaction is simple or compound
  » Scenarios and use cases go until goal success or abandonment
Overspecifying

- The simplest statement that accurately reflects the business need is best
  » Requirements are not architecture or design
  » Requirements are need
- Overspecified requirements are dangerous
  » they cannot be understood
  » they will not be read
  » they will rot
  » and they will be wrong
Requirements are fun!

• This is the time when you have the most leverage to create a successful project
  » you can change direction with the stroke of a pen
  » you can re-architect the moment you gain a deeper understanding of the true need
  » you can apply all the design tools and experience in your tool chest to finding ways to enable what is now only a dream for the customer

• Plus, you learn about a new knowledge domain!