Life Cycle Objectives

CSE 403, Spring 2003

Software Engineering

http://www.cs.washington.edu/education/courses/403/03sp/

Readings and References

- *Rapid Development*, Steve McConnell
  - Chapter 10, Customer-Oriented Development
- *Implementing VisiCalc*, Bob Frankston
  - http://www.frankston.com/public/writing.asp?name=ImplementingVisiCalc
- *Painless Functional Specifications,* Joel Spolsky
  - http://www.joelonsoftware.com/articles/fog0000000036.html
- *Anchoring the Software Process*, Barry Boehm, USC
  - http://citeseer.nj.nec.com/boehm95anchoring.html
- *I Have Abandoned My Search for Truth, and Am Now Looking for a Good Fantasy*, Ashleigh Brilliant

Elements of Lifecycle Objectives (LCO)

- Operational Concepts — What is it?
- System Requirements — What does it do for us?
- System and software architecture — How?
- Lifecycle plan — Who wants it? Who'll support it?
- Feasibility Rationale — Is this really true?

Definition of Operational Concept

- Top-level system objectives and scope
  - » User community?
    - business, personal, demographic
  - » Environment this program works in?
    - device availability, networking fabric, ...
  - » Major benefits?
    - Given the above, will the potential user be interested?
  - » Establish what the system does and does not do
    - Realistic expectations now avoid disappointments later
    - "Warning: system will not make you young, sexy, rich."

Elevator Pitch

- "Okay, we're going to the 75th floor. You've got a minute and a half. What is this thing of yours is supposed to do anyway?"
- Don't sell yourself on something that isn't true
  - » You are **not** marketing something that's already been made
  - » You **are** trying to figure out if there is a need for this wonder-blob that is being proposed

CueCat 2000

BackRub 1996

VisCalc 1979

VisiCalc Design Principles

- VisiCalc was a product, not a program.
  - » Decisions were made with the product in mind and, to the extent possible the programming was towards this end.
- The goal was to give the user a conceptual model which was unsurprising -- it was called the principle of least surprise.

Implementing VisiCalc, Bob Frankston

There is one *today*, many *tomorrows*

- Early discussion of objectives and scope is great
  - » you can make radical changes now to improve capability for growth and change in the future
- Don't create a product that is static by design
  - » A "point-solution" solves a particular problem at a particular time for a particular user group
- Don't need to design the changes, just show that they can happen in various ways
  - » think abstraction and layers

Who will use it?

- Create typical scenarios for product usage
- Make up various example users
  - » if you can't think of one, what does this tell you?
- Be specific
  - » assign them names, job titles, working relationships
  - » dream up situations - typical, busy, breakdown, ...
- Talk to the customer about these scenarios
  - » you'll be amazed - they know what's hot or not

A scenario from WhatTimeIsIt.com

Cindy is a teenager in high school. She goes to a pretty pathetic public high school, and she's pretty smart, so when she gets home at 2:00 pm, it only takes her about 7 minutes (on average) to do her Algebra homework. None of her other teachers even bother to give her homework. Her baby brother (half brother) is vegged out in front of the only TV set watching Teletubbies, so she spends the afternoon (from 2:07 until about 6:30, when her *new* mommy serves dinner) surfing the net and chatting with her friends on AOL. She's always looking for exciting new web sites. As a result of typing "What Time Is It?" randomly into a search engine (by mistake, she meant to ask one of her friends using Instant Messenger) she gets to WhatTimeIsIt.com, and sets up a new account. She chooses a user name and "RyanPhillipe" as her password, selects her time zone, and *voila* -- finds out what time it is.

http://www.joelonsoftware.com/articles/WhatTimeIsIt.html

Other Usage Scenarios

- Who is responsible for on-going funding?
  - » Can you imagine the scenario in which the funding for maintenance is requested? Is any money forthcoming?
- Which group is doing sysadmin for this?
  - » Are they glad about it? Does it totally mess them up?
- Who will support it when it breaks?
  - » 1-hour on-site support? 3-month bug fix release cycle?
- Who will be responsible for new features?
  - » requirements? implementation?

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

Risk Reduction

- "Failing to write a spec is the *single biggest unnecessary risk* you take in a software project"
  - » Joel Spolsky
- The act of writing the spec -- describing how the program works [from user perspective] in minute detail -- will force you to actually design the program
  - » you get a chance to see the potholes before you fall in
  - » you get a chance to back up and change your mind before you've written thousands of lines of code

Once in motion, ideas stay in motion

- People get attached to their creations
  - » if it's just a paragraph or two, it's easy to change
  - » if it's pages and pages, it's hard to change
- Nobody wants to throw out hard work
  - » even if the problem it solves is now irrelevant!
  - » it feels like criticizing, instead of discussing
- Architects get blinders very quickly
  - » if we take that approach, then my group won't be needed at all on this project ∴ that's a bad approach

Specs as Communication Support

- It's always amazing how:
  - » people hear and remember some things
  - » people hear and don't remember other things
  - » two people hear exactly the same thing and remember something completely different
- Write stuff down, then point to it when needed
  - » single source of information
  - » fantasy reduction benefits are huge
    - but remember, specs are a tool, not a magic elixer

Face the problems early

- Writing an outline of program features makes you think about the high level areas of interest
  - » Can't overlook major functional areas
- Writing the details makes you think about how you are going to do these things
  - » Can't overlook major architectural defects
- While you've still got time, you can toss the early architecture and replace it completely

What's in the spec?

- An author
  - » Take responsibility for your work
- Scenarios
  - » Let the customers see these ideas in action
- Non-goals
  - » Eliminate the "implied" goals
- Overview
  - » Elevator pitch with a drawing or two

What else is in the spec?

- Details of operation from user perspective
  - » what's it look like to the various users
  - » what happens during overload, weekends
  - » general performance parameters
  - » typical equipment requirements
- Open issues
  - » state them explicitly
- Side notes
  - » for different reader communities

Spolsky's Rules for Writing

- Be funny
  - » be specific, people love it and will discuss it
- Be understandable
  - » a customer who understands will help you succeed
- Write as simply as possible
- Review and reread
- Templates considered harmful
  - » an entry to fix every oversight in the last 5 years

System and Software Architecture

- Sufficient detail to support feasibility analysis
  - » multiple viable choices is great at this stage
  - » people lock on to a particular architecture *very* quickly and get attached to their perceived piece
- If you can't define an architecture that seems to make sense, don't ignore the problem
  - » Basic data flow or performance problems will kill a system, no matter how many features it has
  - » Rethink why and for whom you are doing this

What is the Life-Cycle plan?

- "The WWWWWHH principle"
  - » Why is the system being developed? Objectives
  - » What will be done When? Schedules
  - » Who will do it? Where are they? Responsibilities
  - » How will the job be done? Approach
  - » How much of each resource? Resources

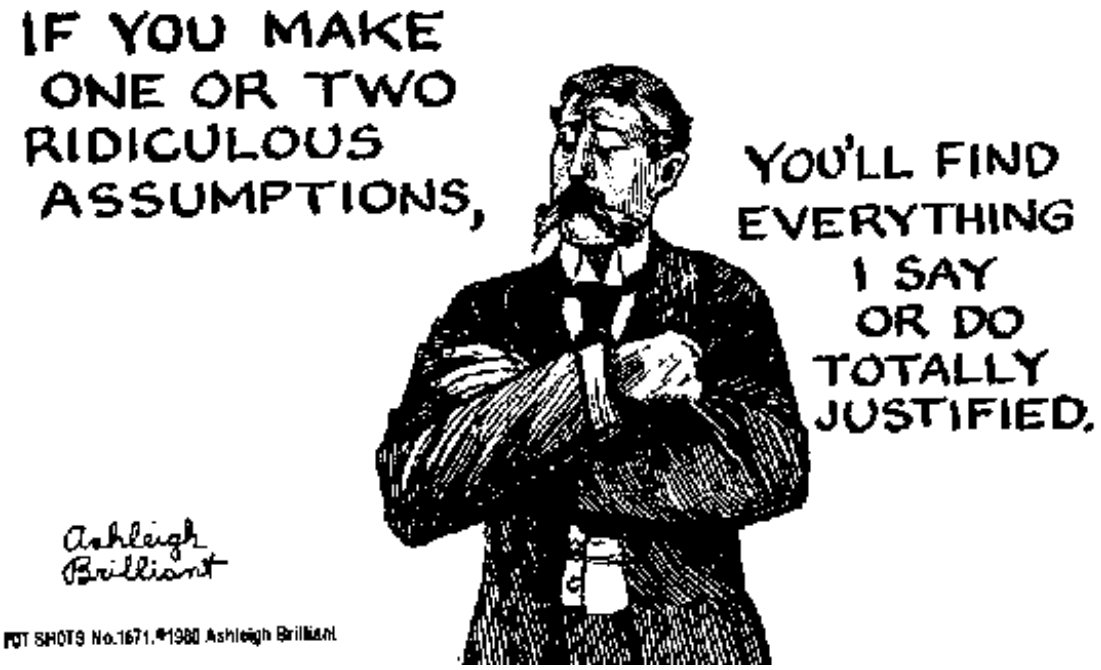
- This can be done in one or two slides early on in the project, more detail in later spins

Feasibility Rationale

Confirm the conceptual integrity and compatibility of the various components described above.

or

Take a reading on the plausibility meter.
Is it trying to tell you something?



Ashleigh Brilliant
Pot Shot 1671