ILL NEED TO KNOW YOUR REQUIREMENTS BEFORE I START TO DESIGN THE SOFTWARE.

FIRST OF ALL, WHAT ARE YOU TRYING TO ACCOMPLISH?

I'M TRYING TO MAKE YOU DESIGN MY SOFTWARE.

I MEAN WHAT ARE YOU TRYING TO ACCOMPLISH WITH THE SOFTWARE?

I WON'T KNOW WHAT I CAN ACCOMPLISH UNTIL YOU TELL ME WHAT THE SOFTWARE CAN DO.

TRY TO GET THIS CONCEPT THROUGH YOUR THICK SKULL: THE SOFTWARE CAN DO WHATEVER I DESIGN IT TO DO!

CAN YOU DESIGN IT TO TELL YOU MY REQUIREMENTS?

© Scott Adams, Inc./Dist. by UFS, Inc.
Question

What phase of development involves understanding what the system is going to do?

A. Requirements
B. Scenarios or use cases
C. Specifications
D. Human factors
E. Web site design
Motivation

• You have to know what to build before you build it
  • Understanding the user’s needs
  • Understanding what would meet those needs

• Requirements is the definition of the problem
  • Not a definition of the system
  • Or even a definition of what will be built
  • These come later
Importance of Requirements

• Many software projects fail
• Poor requirements cause over half of these failures
  • 13% fail due to incomplete requirements
  • 12% fail due to lack of user involvement
  • 10% fail due to unrealistic expectations
  • 9% fail due to changing requirements
  • 7% fail because the system is no longer needed
Requirements Process

• Requirements is an on-going process
  • You need a good first approximation
  • Think of it as a dialog with your stakeholders
  • Understand what they want
  • Their understanding what you can do

• Things are going to change
  • People change their minds
  • Markets change
  • New opportunities arise
Requirements are Difficult

• Communications difficulties, e.g. different vocabularies
• Can’t anticipate all questions
• You won’t understand the user’s needs and desires
• Building the system creates opportunities
• Users are fickle (don’t know what they really want)
• Users don’t understand what is possible/difficult
Requirements Goals

• Define the problem from the user's point of view
• Determine the outlines of the best solution
• Determine what is required and what is optional
• Determine what is feasible (resource limitations)
• Determine acceptance criteria
• DO NOT WORRY ABOUT IMPLEMENTATION
Creating Requirements

• Put the USER first
  • Don’t assume the final system
• Assessing the needs for the project
  • Get the data from the potential users
  • But what data is required?
  • But how to get the data?
Who is the User

Consider a shopping site

- Is it the person doing the shopping
- Is it the person who does the packing and shipping
- Is it the person who does the ordering of products
- Is it the person who sets up the catalog
- Is it the person owning the store
- Is it the person who is paying you to build the site
Sample Interviews
Talking to Your Client

• Understanding what they want
  • And why they think they want it
  • There should be a reason

• Understanding what they have done
  • Do they have a current system to interface with
  • Have they talked to users?
    • Do they know their potential users
    • You might want to do it for them
  • Do they have a current user base
    • Have they solicited feedback
    • Can you get access to the base for interviews, surveys, ...
Good and Bad Clients

• Not all clients are equal
• Good clients
  • Provide appropriate information
  • Keep in touch, answer questions, communicate
  • Provide appropriate feedback throughout the project
  • Provide direction
  • Have “done their homework”
• Bad Clients
What’s Worse than a Bad Client

• He who represents himself has a fool for a client
  • Holds in law
  • Holds in medicine
  • Holds in software development

• Clients emphasize the user and the user’s needs
  • Programmers emphasize the programmer
  • Easier to code doesn’t mean easier to use

• Those of you working on your own project
  • Do real user studies
  • Find a set of users outside your group and listen to them
Getting USER Information

• Talking to users
  • Interviews
  • Focus Groups
  • Questionnaires (Qualtrics)
  • Facebook, Mechanical Turk

• Observing users
Talking to Users

• Understanding what they want
  • And why they think they want it
• Understanding what is important
  • Features, performance, ...
• Vocabulary
• Who is in charge
Interviewing Potential Users

• Structured versus unstructured interviews
• Interviews require considerable preparation
  • Determine what information is needed
  • Find out about the interviewees
  • Decide on questions and organization
• Process
  • Move from general to specific to general (open) questions
  • Summarize the interview
    • In terms of stories, use-cases, scenarios
    • Examples of the use of the system
    • In terms you can feed back to the client
  • Follow up
Questionnaires and Surveys

• What is your target audience (stakeholders)
  • Particular users
  • General public
  • Others that may be affected by the system
  • Avoiding bias in audience selection

• Needs a lot of thought
  • What information is needed
  • Phrasing questions so as not to bias the outcomes
  • Determining how to interpret the results

• **Likert scales**
  • Range of 1-7 (or 1-5) where ... rate X
  • Provides a means of compiling results

• **End with open questions** (won’t be answered by everyone)
Stories and Scenarios

- Used in requirements and specifications
  - Also called use cases
  - Also used in design, development, testing
- Descriptions of common use cases
  - Think about how your system will be used
  - A story describes a common use
    - From start to end
    - Should be as specific as possible (realistic)
Stories and Scenarios

- Detailed description of task
  - Use actual names
  - Be specific
  - It is a story

- Pictures of your application for each step
  - Can be crude
  - Should give sense of what is going on
Stories and Scenarios

• Form a basis for getting user feedback
  • Give the user a sense of what your app will do
  • Give the user a context for providing input
  • Give the user a chance to change the story

• Ensure your stories are consistent
Stories and Scenarios

• Should cover the principle tasks
• Should cover a diverse set of tasks
• Should give you a sense
  • Of what you will need to do
  • Of how the application will be used
Create A Scenario

- For either of the previous labs
- For your final project
- For using banner
Other Requirements

• Security constraints
• Privacy constraints
• Anticipated average and maximum loads
  • What should happen if these are exceeded
• Internationalization and accessibility
• Validity constraints
  • A product should only be ordered if the user charge is valid
  • Users should only be charged when the product is shipped
  • Avoid duplicate orders; avoid dropped orders
  • Users must have valid email addresses
  • Users must be previous registered
Other Requirements

- Legal issues (responsibility for money, privacy, children, ...)
- Reliability (up time, data safety)
- Performance
- Time and cost limits
- Adoption strategy
  - Meeting all stakeholders needs
Completeness

• The more you can state, the better off you’ll be
• The requirements are going to change
  • Clients will want different things as the project proceeds
  • New options and possibilities will arise
  • Some things might be too difficult
  • Your designs might not appeal to users
    • Even if they appeal to the client
• Keep these up to date
Specifications

• Next step after understanding what the user wants
• Also known as **Requirement Specifications**
• Different from requirements
  • Requirements look at things from USER’s point of view
  • Specifications look at things from PROGRAMMER’s
• Detail what will be built (WHAT not HOW)
  • Essentially a contract
  • We’ll go over these in more detail next week
Question

Specifications do not include:
A. Analysis of user surveys or interviews
B. Diagrams describing the user interface
C. Descriptions of user commands and what they do
D. Description of how system features relate to user requirements
E. Descriptions of existing systems the new system will have to interact with
Project Deadlines

• **By 2/17 you should have talked to your clients**
  • Have a good understanding of the project
  • Think of several stories you can work with
  • Provide a short report detailing this (meeting notes)

• **By 2/24 you should have**
  • Interacted with potential users for defining requirements
  • Shown client several potential designs/solutions
  • Gotten client and user feedback

• **By 2/27 you should have**
  • Handed in an initial specification
Next Time

• LAB on Monday
  • Do one of the available pre-lab technology modules
Creating Specifications

- How might you do this?
- What would you do with twitter data application
Major Causes of Software Failure

- Poor User Input
- Stakeholder Conflicts
- Vague Requirements
- Poor Cost and Schedule Estimation
- Skills that do not Match The Job
- Hidden Costs of Going “Lean and Mean”
- Failure to Plan
- Communication Breakdowns
- Poor Architecture
- Management Problems
Obtaining Requirements

- How might you do this?
- Example:
  - CD Query Data (from previous lab)
  - How would you find out what is wanted?
  - Who would you ask?
  - What questions would you ask?
  - What do you really want to know?
What’s Worse than a Bad Client

• He who represents himself has a fool for a client
  • Holds in law
  • Holds in medicine
  • Holds in software development

• Clients emphasize the user and the user’s needs
  • Programmers emphasize the programmer
  • Easier to code doesn’t mean easier to use

• Those of you working on your own project
  • Do real user studies
  • Find a set of users outside your group and listen to them
Designing the User Experience

• Should be done interactively with your clients
  • And with perspective users
• Provide them with 3 (or more) alternative designs
  • To show the range of available options
  • To avoid commitment on your part
• Be prepared for multiple iterations