CS 121
“Ordering Chaos”

Requirements

“Mike”
Michael A. Erlinger

Today:
Requirements – What is Desired
Administrivia

Know Issues???

portables
twiki
github
macs
teams
Review

Software development is hard

Managing complexity: abstraction, tools, software processes, principles & patterns

Waterfall vs. agile

Iteration 1: 1\textsuperscript{st} and 2\textsuperscript{nd} iOS tutorial

Lab 1 will start Sudoku implementation
Today

Requirements

Requirements modeling

Sudoku:
Why are requirements so important?

Obviously:

We need to know what we are supposed to build before we can build it.

Not So Obvious:

Failures to clearly specify at this stage are costly, more costly than at any other development phase.

Failures at this stage are common.
How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it

How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed
Requirements

What does the customer actually want?

Can requirements be created, analyzed, evaluated, etc??

Requirements describe a condition or capability to which system must conform.
Types of Requirements: FURPS+

Functional: features, capabilities

Usability: human factors, aesthetics, consistency, documentation

Reliability: frequency of failure, recoverability, predictability

Performance: response times, throughput, accuracy, availability, resource usage

Supportability: adaptability, maintainability, configurability

+: others

How might this fit into building a Sudoku Game??
Requirements in Software Development Paradigms

**Waterfall**
- Requirements up front
- Written documents are primary models

Stakeholders approve when SRS completed

**Agile**
Focus on most important behavior to add in the next iteration

Effective use of models including working software

Stakeholders drive priorities for next iteration
Requirements Errors COSTLY!!

<table>
<thead>
<tr>
<th>When problem is found</th>
<th>Requirements Errors</th>
<th>Design Errors</th>
<th>Construction Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td>1x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td>3x</td>
<td>1x</td>
<td></td>
</tr>
<tr>
<td>Construction</td>
<td>5-10x</td>
<td>10x</td>
<td>1x</td>
</tr>
<tr>
<td>System test</td>
<td>10x</td>
<td>15x</td>
<td>10x</td>
</tr>
<tr>
<td>Post ship</td>
<td>10-100x</td>
<td>25-100x</td>
<td>10-25x</td>
</tr>
</tbody>
</table>
Question

Most errors found by users in software are the result of:

A. coding errors
B. errors in requirements
C. system integration errors
D. design errors

Answer: B

Requirements Errors: Costly & Common

Case 1:

A friend was on a review (RED) team for a Norwegian air defense contract. The customer generated an ops (operations) concept document that was about 100 pages long from which a flat requirements document containing about 2000 requirements was generated. The problem with this contract was that the 2000 requirements were not definite enough, thus the result was a floating set of slightly different requirements every 6 months or so. The project was a 3 year project in it's 6th year with 100% overrun on a fixed price contract (reason for the red team review).
Why are requirements so difficult to specify?

Customers cannot tell you what they want
Customers cannot tell you what they do NOT want
Have conflicting needs
Requirements Example

Case2:

There were two ops concept documents, each about 2-4 pages long, one from developer view and one from a customer view.

This resulted in a 50 page top level system requirements document from which about 5 or 6 lower level requirements documents were written - e.g., spacecraft requirements document, network requirements document, set-top box requirements document; operations requirements document, air-link requirements document, etc.

A document was then written showing the linkage/relations between all the various requirements.
Growth in requirements

Why are requirements so difficult to specify?

Customers cannot tell you what they need
- they don’t know
- they don’t clearly articulate their needs- English sucks as a specification

Customers have conflicting needs

Their needs change

Technology changes, along with price
- Monitors – 10 years ago paid $2k for a 600x800, today $300 for a much better display + can run 2 displays, woops 4 displays.
Developing Requirements: Best Practices

Still a developing area; key observation is that customer interaction is a constant requirement

keep talking!!!
Developing Requirements

Inception: Elicitation

- define initial concept
- identify stakeholders
- gather background information: competitive analysis, technology review, etc
- Ask the CUSTOMER!!
- identify “perceived requirements“
- Use tools to capture all Requirements
Requirements

Inception

Elicitation: What does the customer say they want?

Analysis: What does the customer really want? What can you realistically provide? Do you really understand?

SRS – Software Requirements Specification created based on Elicitation and Analysis
Specify **what not how** – hard to do

Having some sort of sorting game with pictures of chemical reactions and physical reactions

Chemical reactions: Baking soda/vinegar, baking, cooking, toasting, burning etc

Physical reactions: Cutting, ripping, mixing, melting

Another idea is a little more complex

Having students sort the chemical elements based on their properties into correct place on the periodic table.

My idea is that an alien periodic table has been found and the students need to place the alien elements into the correct spot based on the elements and properties.

The elements would be very similar to the earth periodic table. Students would need to know about the number of protons neutrons electrons

I attached an example of the alien periodic table activity

Need Unambiguous Specification … English is by nature ambiguous
Agile Requirements

At the start of each iteration:
  Incorporate new goals (often produced by last iteration)
  Remove items no longer needed
  Reprioritize
  Clarify requirements for goals at top of stack
  Plan iteration

Who prioritizes?
  Customer driven
  Risk driven

highest priority, best modeled goal

lowest priority, least modeled goal
Tools for modeling (primarily functional) requirements

- Storyboards
- Flow charts
- State diagrams
- Use cases
- User stories
- Prototypes
- Text documents
- etc.

There is no single tool that can give a comprehensive view
Storyboard for tic tac toe

1. **Start**
   - Player starts game
   - Grid appears
   - Player selects square
   - X appears

2. Computer selects square and O appears

3. Repeat until
   - 3 in a row
   - Player or computer wins
   - Stalemate

   OR
   - stalemate

   Congratulations!
   - You won!

   - Stalemate …
SRS vs Storyboard

Software specification documents (SRS)

- Precise
- Hard to produce
- Hard to assess

Storyboard

- Informal but conveys look and feel
- Easy/quick to produce
- Easy/quick to assess
- User Involvement
Flow chart
Tic Tac Toe

Player selects square

- Square empty?
  - Y: X appears
  - N: Etc.

X appears

- Three in a row?
  - Y: Player wins
  - N: Any empty squares?

- Y: O appears
  - Y: Etc.
  - N: Stalemate
- N: Etc.
Tic-tac-toe state diagram

- 3 X’s in a row
  - Player wins

- 3 O’s in a row
  - Computer wins

- No 3 in a row
  - Player’s turn
  - Grid has blank cell
    - Player’s turn
    - Grid has blank cell
      - Computer’s turn
      - Computer’s turn

- No 3 in a row
  - Grid full
    - Stalemate
    - Computer’s turn
    - Player wins
    - Computer wins

- No 3 in a row
  - Grid full
    - Stalemate

- No 3 in a row
  - Grid full
    - Stalemate
Use Case

Describe how a user interacts with the system to achieve a goal.

Tic tac toe use case: Play game
Tic tac toe “play game” use case

Play game:
1. Player starts game
2. Empty grid appears
3. Player selects blank cell
4. X appears in cell
9. Computer places 0 in a cell
10. Return to step 3

Augment with alternative paths
Tic tac toe “play game” Use Case

Play game:
1. Player start game
2. Empty grid appears
3. Player selects blank cell
   3a. Player selects non-blank cell, return to step 3
4. X appears in cell
   4a. X appears in cell, 3 X’s in a row, player wins
   4b. X appears in cell, grid full, stalemate
5. Computer places 0 in a cell
   5a. Computer places 0 in a cell, 3 0’s in a row, computer wins
   5b. Computer places 0 in a cell, grid full, stalemate
6. Return to step 3
Tic tac toe “play game” Use Case

We can embed other use cases

Play game:
1. Player start game
2. Empty grid appears
3. Player selects blank cell
   3a. Player selects non-blank cell, return to step 3
4. X appears in cell
   4a. X appears in cell, 3 X’s in a row, player wins use case
   4b. X appears in cell, grid full, stalemate use case
5. Computer places 0 in a cell
   5a. Computer places 0 in a cell, 3 0’s in a row, computer wins use case
   5b. Computer places 0 in a cell, grid full, stalemate use case
6. Return to step 3
# Tic Tac Toe Use Cases

<table>
<thead>
<tr>
<th>Goal setter</th>
<th>Goal</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Player</td>
<td>Play game</td>
<td>1</td>
</tr>
<tr>
<td>Player</td>
<td>Player wins</td>
<td>2</td>
</tr>
<tr>
<td>Player</td>
<td>Computer wins</td>
<td>2</td>
</tr>
<tr>
<td>Player</td>
<td>Stalemate</td>
<td>2</td>
</tr>
<tr>
<td>Player</td>
<td>Start new game</td>
<td>2</td>
</tr>
<tr>
<td>Player</td>
<td>Exit game</td>
<td>4</td>
</tr>
<tr>
<td>Player</td>
<td>Show stats</td>
<td>5</td>
</tr>
<tr>
<td>Player</td>
<td>Save game</td>
<td>6</td>
</tr>
<tr>
<td>Player</td>
<td>Load existing game</td>
<td>6</td>
</tr>
</tbody>
</table>
Fully dressed Use Case

Title
Primary actor
Goal in context
Scope
Level
Stakeholders and interests
Precondition
Minimal guarantees
Success guarantees
Trigger
Main Success Scenario
Extensions
Technology & data variations list
Related information
User story

As a player, I want to view the current configuration of the grid so I can figure out my next move.
Requirements Summary

Developer focused

Software specification documents (SRS)

- Precise
- Hard to produce
- Hard to assess

User focused- users can understand/produce, make clear priorities

Agile models

- Informal
- Easy to produce
- Easy to assess
User stories for Sudoku

You’ve launched Sudoku. What next?

As a player I want to

Do this ____________________

In order to achieve this goal _________________

Confer with your partner then offer up your suggestion.
Do we have consensus?
INVEST User Stories

• Independent  Should be independent of other stories
• Negotiable  Captures essence, but not details… not a formal contract
• Valuable  Add value to the project
• Estimable  We can estimate how long it will take
• Small  Can be implemented in one iteration
• Testable  We can write a test for it
Sudoku User Story

Feature: Display initial grid

User story: As a player I want to view the initial Sudoku grid in order to figure out my first move. The initial grid should have blanks and should have a unique solution.


Would showing a picture of Sudoku grid suffice?
Sudoku: User Stories

You’ve launched Sudoku. A valid grid with blanks appears. What is next?

As a player I want to

Do this _____________________

In order to achieve this goal ________________

Confer with a partner then offer up your suggestion.
User Story

Feature: Enter number in blank cell

User Story: As a player I want to enter a number in a blank cell in order to solve the puzzle.
Sudoku Rules

???????
Important features of Sodoku Game

First step of your team:

Write yours on the board on the continuum most important (left) to least important (right).
Requirement Quality, cont

Specify what **NOT** how

Unambiguous

Testable

Feasible

Consistent – analysis critical

Prioritized

Traceable – number of systems to build a table matching requirements (numbers) to specific code

Agreed upon by customer
Agile requirements

Project inception: move from requirements to goals
Identify high level scope
Key requirements driven by risk and customer
Initial “goal stack”

highest priority, best modeled goals

Well modeled means we understand what to do and how long it will take.

lowest priority, least modeled goals
Lecture Sample questions

Why are requirements so important?
Why are they so difficult to specify?
What are the various types of requirements?
What are the first steps of requirements gathering?
What is a customer elicitation?
What is involved in requirements analysis?
What is an SRS?
What constitutes quality in requirements?
How are requirements managed in Agile processes?
Summary

What are requirements?

What are the types of requirements?

Difficulties in getting ‘good’ requirements

Process of acquiring requirements

Cycle of elicitation - evaluate/analysis - refine
Lab 1: pair programming
Why are requirements so important?

We need to know what we are supposed to build before we can build it.

Failures at this stage are costly.

Failures at this stage are common.
Backlog

List of features to be implemented

Our current backlog:

- View initial grid

- Enter number in blank cell (This is the goal for our next iteration)
UI Design

We are going to need to decide on a UI design

We will look at that next week
The End