Pleasing Your Customer

Kenneth M. Anderson
University of Colorado, Boulder
CSCI 5828 — Lecture 3 — 01/20/2009

© University of Colorado, 2009
Goals

- Review material from Chapter 1 of Pilone & Miles

- Concepts include
  - Ultimate goal of software development
    - Hint: title of chapter gives it away
  - Importance of Iteration
  - Initial steps in planning a development activity
Pleasing Your Customer

- If your customer is unhappy, you’re unhappy
  - The ultimate goal of software development is
    - solving your customer’s problem by delivering
      - what is needed
      - on time
      - on budget
  - This is NOT easy
- Important: Need to have a clear idea of who your customer is!! Why?
Who is the customer?

- The person or persons playing the role of the customer can vary across development contexts.
  - In chapter 1, we have a situation in which we are being hired by Tom, a small business owner, to create a website.
    - Tom is clearly the customer:
      - He is providing requirements and paying for the work.
    - But when the website is deployed, who becomes the customer?
      - Tom’s customers!
Customer == User

- HCI and CSCW research shows that systems live or die by how happy the “end users” are with the system
  - Tom’s customers in this case are the end users
    - However, in this initial sequence of development, we only have Tom and we have to go by what he says
    - In the future, Tom will be hearing from his customers about the utility and usability of our website and he will convey that feedback to us
  - What’s the difference between utility and usability?
Other Customers

- You (!)
  - Often for only small scale software

- CTOs
  - Acquiring enterprise level systems for an organization
  - Who are the end users in this situation?

- New Application Development (be it desktop, web, mobile)
  - For version one: development team
    - How can you avoid this? Who are the end users?
Chapter Example (I)

- Tom’s Trails Website for existing small business
  - Find Trails
  - Buy hiking boots and other related items
  - Learn from Tom’s experience (FAQs, monthly column, etc.)
- Needs website created in three months time for upcoming industry trade show
- Two questions
  - How much will it cost?
  - How long will it take?
Chapter Example (II)

- Big Bang Approach to Software Development
  - The developer that Tom hires
    - takes his initial ideas
    - disappears for a month and codes like crazy
    - displays final product to Tom
  - What happens?
The developer delivered software that did not meet Tom’s expectations

And if the customer is unhappy...

Reasons

Did not fully understand Tom’s requirements

Tom’s view of what he wanted can shift over time

Did not check in with Tom during development

So any unclear requirements were resolved by developer making arbitrary decisions
Not Guesswork
Software is NOT Guesswork
Software is NOT Guesswork
Software is NOT Guesswork

You need to keep the customer in the loop to make sure you’re on the right path even when you’re SURE you know what the customer wants… why?
Great Software Delivers

What is needed

On Time

On Budget
What is needed
On Time
On Budget
Great Software Delivers

What is needed

On Time

On Budget

requirements

project planning

project management
So, how to get there?

- The developer in the example
  - did not apply basic project planning/management techniques
  - did not make use of iteration
  - did not acquire feedback
  - did not clarify requirements with the user

- The developer made use of a life cycle (barely worth the name) called **CODE & FIX**

  actually just “code” and now needs to do a LOT of fixing!
Iteration is important because requirements CHANGE.

The Goal

The Trainwreck

Start

The Goal
With iteration a project can make **course corrections** as requirements change so that **what’s delivered** matches **what’s needed**.
Iteration (I)

- An iteration produces working software
  - when you complete an iteration, you have something to show the customer
- Iterations should be relatively short with respect; where “short” is defined by the estimated length of the project
  - they can be anywhere from two weeks to six weeks
  - Our book picks 20 days, because that is roughly what you get if you start with a calendar month and subtract out the weekends
Iteration (II)

- Iterations are “fractal”
  - They decompose nicely into smaller iterations
    - A 20 day iteration can be split into two “internal” 10-day iterations
    - A work day can be split into roughly three 2-hour iterations
      - Here the customer is yourself: what can I get done in two hours
        - did the result match my expectations? should I show this to my co-worker for feedback? etc.

- Iterations promote continuous building and testing
  - Otherwise, you are not delivering “working software”
Iteration (III)

- Each iteration is a mini life cycle
  - A typical life cycle consists of these major activities
    - Requirements; Design; Code; Test; Deploy
  - High-level: ideas come in and working software comes out
  - Iterations decompose this high-level process such that all of the major steps are performed for each iteration
    - By doing this, you increase your chances of producing high quality software that solves your customer’s problem
Indeed, the MIT Sloan Management Review published an analysis of software development practices in 2001. Strong correlation between quality of software system and the early delivery of a partially functioning system:

- the less functional the initial delivery, the higher the quality of the final delivery!

Strong correlation between final quality of software system and frequent deliveries of increasing functionality:

- the more frequent the deliveries, the higher the final quality!

These results suggest that iterations need to be short!
Iterations can help you plan ahead to achieve success

- **Input:**
  - List of features with estimates and priority
  - Customer supplies list and priority
    - Book recommends priority scheme based on multiples of 10
    - Developer supplies estimates (we’ll see how in lecture 5)

- **Output**
  - List of proposed iterations with features assigned
    - Highest priority features are tackled first
    - Features assigned have estimates less than iteration length
Can you do it?

Having an iteration plan lets you keep track of whether the project is feasible

\[(\text{Days of Work Left}) - (\text{Days left before deadline})\]

\[=\]

Can you do it?

You want this number to be NEGATIVE

15 - 30 == -15 (i.e. fifteen days ahead of schedule)
And things will change

- The book shows Tom introducing a new request near the end of the second iteration that leads to three new features, each with their own estimates and priorities.

- With an iteration plan, you can “rejigger” the features across iterations to come up with a new plan that shows how you’ll respond to the change request.

- It will most likely create one or two more iterations which can be stressful because, note, the deadline HASN’T changed.

- In lecture 7, we’ll discuss what to do if the new schedule slips past the deadline.
Working Code

- Note: Your software isn’t complete until it has been released.

- We measure progress via working software.
  - NOT via number of iterations complete.

- So, if we complete three out of 10 features in two iterations of a four iteration project.
  - We are 30% done NOT 50% done.
Please develop an iteration plan for the following:

- Iteration length is 10 days; Due date is 40 days away

<table>
<thead>
<tr>
<th>Feature</th>
<th>Estimate</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log In</td>
<td>2 days</td>
<td>30</td>
</tr>
<tr>
<td>Create Account</td>
<td>6 days</td>
<td>20</td>
</tr>
<tr>
<td>View Stories</td>
<td>2 days</td>
<td>10</td>
</tr>
<tr>
<td>Add Story</td>
<td>4 days</td>
<td>10</td>
</tr>
<tr>
<td>Rate Story</td>
<td>1 day</td>
<td>20</td>
</tr>
<tr>
<td>Find Similar Stories</td>
<td>5 days</td>
<td>40</td>
</tr>
<tr>
<td>Contextual Ads</td>
<td>7 days</td>
<td>10</td>
</tr>
</tbody>
</table>
At the end of day 8 in the first iteration, customer submits new idea that results in these features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Estimate</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add Comment</td>
<td>6 days</td>
<td>30</td>
</tr>
<tr>
<td>View Comments</td>
<td>4 days</td>
<td>20</td>
</tr>
<tr>
<td>Rate Comment</td>
<td>2 days</td>
<td>10</td>
</tr>
</tbody>
</table>

Is it still possible to meet the deadline?
Wrapping Up

The ultimate goal of software development is pleasing your customer; to do that you need to

- deliver (i.e. ship) a system that solves the customer’s problem
- on budget (under budget preferable)
- on time (ahead of schedule preferable)
  - (There is one more thing we should achieve; what?)

Iteration is the key technique to achieving this goal

- Provides ability to plan project, respond to change, and deliver what’s needed
Coming Up Next

- Lecture 4: Introduction to Concurrency
  - Chapter 1 of Magee and Kramer
- Lecture 5: Gathering Requirements
  - Chapter 2 of Pilone & Miles