The software development process
A personal view

Robert Brady

robert.brady@cl.cam.ac.uk

May 2017
<table>
<thead>
<tr>
<th>Approx Size</th>
<th>Description</th>
<th>Approx date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1kb</td>
<td>Punch-card program</td>
<td>1965</td>
</tr>
<tr>
<td>2kb-10kb</td>
<td>Computer Science project</td>
<td></td>
</tr>
<tr>
<td>16kb</td>
<td>Operating system of Sinclair Spectrum</td>
<td>1982</td>
</tr>
<tr>
<td>18 Mb</td>
<td>Human Genome – protein coding</td>
<td></td>
</tr>
<tr>
<td>20Mb</td>
<td>Our trading system</td>
<td>1996</td>
</tr>
<tr>
<td>500Mb</td>
<td>Windows (50 Mlines)</td>
<td>2015</td>
</tr>
<tr>
<td>20Gb</td>
<td>Google (2 Glines)</td>
<td>2015</td>
</tr>
<tr>
<td>300Gb</td>
<td>Storage on my laptop</td>
<td></td>
</tr>
</tbody>
</table>

Complex, vast number of ‘corner cases’
Recruiting great developers

‘Just recruit great developers’

- They are 10-50 times more productive than ordinary developers
- who are 10-50 times more productive than poor developers
- Self-starters: artists who don’t want or need managing
- Most of you

For a growing company, great developers...

- Do not exist in sufficient numbers
- Progress to more important things (sales, CEO)

So this is for the rest of us...
What management can do

Tell developers what you want them to do, measure their performance

- Ordinary software companies do this
- Turns great developers into ordinary ones
- Lose a factor of 10-50 productivity

Supply infrastructure for developers to succeed, and trust them

- Communicate the problem – eg we really want to land customer A
- Cut out the middle-man – eg onsite customer experience
- Autonomy – eg work on what you want 20% of your time
- Recognition – eg ‘brown bag’ talks
- Creates great developers

Not limited to developers

- Literature on "Theory X-Theory Y" management culture
Success rate of projects (Johnson 1998)

Break up large projects up into shorter ones (weeks not months)
Actual use of requested features (Johnson 2002)

Ship frequently. It is the only way to stop wasting effort.
Hard-to-find bugs really matter

<table>
<thead>
<tr>
<th>Period</th>
<th>average</th>
<th>bugs</th>
<th>mt tf</th>
</tr>
</thead>
<tbody>
<tr>
<td>10–20d</td>
<td>15d</td>
<td>1</td>
<td>15d</td>
</tr>
<tr>
<td>20–40d</td>
<td>30d</td>
<td>2</td>
<td>15d</td>
</tr>
<tr>
<td>40–80d</td>
<td>60d</td>
<td>4</td>
<td>15d</td>
</tr>
<tr>
<td>80–160d</td>
<td>120d</td>
<td>8</td>
<td>15d</td>
</tr>
</tbody>
</table>

...Etc.

Run automated tests every day
Short development cycles or ‘sprints’

Most people use Jira by Atlassian (equivalent of Salesforce)

- Everything you need
- Integrated customer site
- Statistics for team motivation and feedback
Hey Pig, I was thinkin' we should open a restaurant.

I don't know. What would we call it?

How about "Ham-n-Eggs."

No thanks, I'd be committed, but you'd only be involved!
How a sprint works

Team members (‘pigs’) commit at the beginning of a sprint
- Scheduling meeting with ‘chickens’ (sales, management ...)

Chickens refrain from interfering during sprint
- Unplanned changes reduce efficiency and motivation
- Customers don’t deploy fixes in less than 2 weeks anyway

Scrum master fixes problems
- My computer doesn’t work
- The customer didn’t answer my question

Team succeeds, not individuals
- eg, John hasn’t finished testing, Fred will help
- Fred gains cudos with peers
- Part 1B project is run this way
Daily scrum meeting

Old picture – now cluster around a Jira screen
Daily scrum meeting

- Daily
- 15-minutes
- Stand-up

Co-ordination, not problem solving

- Whole world is invited
- Only ‘pigs’ may talk
  - team members
  - Scrum master
  - product owner
- Helps avoid other unnecessary meetings
There is an ‘I’ in team

Developers at customer site
- Transforms understanding and motivation
- Fixes things at the airport on the way back

Developers have 20% own time
- Gives a ‘Brown bag’ talk on what he did
- Enables large organizations to innovate
Test-oriented development – example

Class MyClass

```c
int Square(int a) {
    return a**2
    (return a*a in v.2)
}
```

Test() {
    assert Square(0) = 0
    assert Square(-5) = 25
    assert Square(5) = 25
}

Some environments run the tests every time you check in

Test manager’s job – "prove to me that your change works"
Automated tests

Report a bug

- Write a test script
- Run it with each daily build (it fails!)
- Make it work and you are done
- Run the tests automatically with each daily build
  - Find out when someone else breaks your feature
  - May find regression problems unrelated to your feature
  - Refactor safely

New features

- User stories from the customer
- Write the tests based on the user stories (check with user!)
- Make it work and you are done
- Run the tests automatically with each daily build