Information Systems Concepts

Agile Approaches

Roman Kontchakov

Birkbeck, University of London

Based on Chapter 3 of Bennett, McRobb and Farmer:
Outline

- Agile Approaches
  - Section 3.4.2 (pp. 79 – 80)
Agile Approaches

I'll need a project plan to justify the resources we need to change our software.

I can make those software changes in ten seconds.

Done.

Good work. Now all we need is that plan.
Yay, monkeys. This is not autobiographical, but I did indeed used to have a job writing software. VB! MS SQL! I affectionately referred to myself and my co-developers as code monkeys, especially when a client asked me a question that I didn’t want to answer (“What do I know? I’m just a code monkey.”).
Agile Approaches

- Reaction against heavyweight methodologies
  - The crushing weight of corporate *bureaucracy*
  - The rapid pace of information technology *change*
  - The *dehumanizing* of detailed plan-driven development

- Evolved from the mid 1990s
  - Originally called ‘lightweight’ methodologies
  - Sounds not serious/robust enough

- Agile Alliance
  - A weekend in February 2001 at Snowbird, Utah
Agile Approaches

- **eXtreme Programming (XP)**
- SCRUM
- Dynamic Systems Development Method (DSDM)
- Feature-Driven Development (FDD)
- Adaptive Software Development (ASD)
- Crystal Clear
- Agile Modeling
- ......

*RUP (and USDP) can also be used in an agile manner*
Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.
Agility

- In the software development sense:
  - The ability to respond quickly to change & environment
  - The adaptability to suit new or unexpected challenges

- Where does agility come from?
  - Agile methodologies derive much of their agility by relying on the tacit knowledge embodied in the team, rather than writing the knowledge down in plans.
Criticism of Agile Approaches

- Are we returning back to cowboy coding?
  - Not really – Agile Methodologies do have disciplines

- Criticism
  - Hacker interpretations
    - e.g., “responding to change over following a plan”
    - “Great! Now I have a reason to avoid planning and to just code up whatever comes next.”
<table>
<thead>
<tr>
<th>Home-ground area</th>
<th>Agile methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developers</td>
<td>Agile, knowledgeable, collocated, and collaborative</td>
</tr>
<tr>
<td>Customers</td>
<td>Dedicated, knowledgeable, collocated, collaborative, representative, and empowered</td>
</tr>
<tr>
<td>Requirements</td>
<td>Largely emergent; rapid change</td>
</tr>
<tr>
<td>Architecture</td>
<td>Designed for current requirements</td>
</tr>
<tr>
<td>Refactoring</td>
<td>Inexpensive</td>
</tr>
<tr>
<td>Size</td>
<td>Smaller teams and products</td>
</tr>
<tr>
<td>Primary objective</td>
<td>Rapid value</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Home-ground area</th>
<th>Plan-driven methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developers</td>
<td>Plan-oriented; adequate skills; access to external knowledge</td>
</tr>
<tr>
<td>Customers</td>
<td>Access to knowledgeable, collaborative, representative, and empowered customers</td>
</tr>
<tr>
<td>Requirements</td>
<td>Knowable early; largely stable</td>
</tr>
<tr>
<td>Architecture</td>
<td>Designed for current and foreseeable requirements</td>
</tr>
<tr>
<td>Refactoring</td>
<td>Expensive</td>
</tr>
<tr>
<td>Size</td>
<td>Larger teams and products</td>
</tr>
<tr>
<td>Primary objective</td>
<td>High assurance</td>
</tr>
</tbody>
</table>

Take Home Messages

- Manifesto for Agile Software Development
  - The four ‘over’s