Lean Software Management: BBC Worldwide Case Study

Peter Middleton, Queen’s University Belfast
David Joyce, ThoughtWorks Australia
Lean Software Management: BBC Worldwide Case Study

Peter Middleton, Queen’s University Belfast & David Joyce, ThoughtWorks Australia

14 October 2011
European Lean IT Summit
Strategy & analysis

• **Customers**: Statistical Process Control
  – *Total end-to-end time to serve customer*

• **Targets** = System Conditions
  – *Why does system behave as it does?*
  – *Remove sub optimisation, then I.T.*

• **Purpose** of customer interactions
  – *Why did they contact us?*

• **Failure demand**: 30% - 70%
  – *Wrong information, delivery not made*
Analysis of why customers call

Failure demand: activity but waste

- Failure demand 37%
- Value demand 63%
Foundations

• **PRINCE2** unrealistic: ‘Ensuring that the information required for the project team is available’

• **Failure demand**: 30% - 70% of all demand is caused by failures in the system itself

• **Targets** cause massive waste due to sub optimisation. Measures drive behaviour.
BBC Worldwide
Digital Hub Software Team

- Media Village, White City, West London
- 9 staff: Analyst, Architect, QA, Developers
- Operating cost: £1.5m p.a.
- C#, .NET, MS SQL Server
- Created and maintained software
- 12 months data: Oct 2008 – Oct 2009
- Reported to Business & Project Boards
- Waterfall → Agile → Lean
Engineering Practices

• Test Driven Development (unit tests)
• Automated Acceptance Testing
• Source Control Software
• Bug tracking software
• Decoupling – improve legacy code
• Minimum Marketable Feature concept
• Daily Stand Up (15 minutes)
Japanese Manufacturing Techniques?

- Cars, Printers, Cameras
- Just In Time
- Lean Production
- Pull v. Push
- Kanban
Just-In-Time Principles

- Process Control
- Easy-To-See Quality
- Insistence on Compliance
- Line Stop
- Correcting One's Own Errors
- 100% Check
- Project-By-Project Improvement
Lean Software – key idea

• Reduce Work in Process:
  – Analysis
  – Specifications
  – Design
  – Untested code

• Benefits: (flow: concept to cash)
  – Visible management & less risk
  – Flexibility
  – Productivity
Fig. 1. The Software Pond. Source of errors masked by work-in-progress.

Fig. 2. The Software Lake - Drained by Lean Production. Source of errors exposed by reducing the work-in-progress.
# Kanban 101 (BNP Paribas)

<table>
<thead>
<tr>
<th>Work Items</th>
<th>Step 1</th>
<th>Step 2</th>
<th>...</th>
<th>Step n</th>
<th>Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queue</td>
<td>In Process</td>
<td>In Process</td>
<td>...</td>
<td>In Process</td>
<td></td>
</tr>
<tr>
<td>Queue</td>
<td>In Process</td>
<td>In Process</td>
<td>...</td>
<td>In Process</td>
<td></td>
</tr>
<tr>
<td>Queue</td>
<td>In Process</td>
<td>In Process</td>
<td>...</td>
<td>In Process</td>
<td></td>
</tr>
<tr>
<td>Queue</td>
<td>In Process</td>
<td>In Process</td>
<td>...</td>
<td>In Process</td>
<td></td>
</tr>
</tbody>
</table>

- **Work Items**: The items are divided into queues and then enter the process steps.
- **Step 1**: Items move from the queue to the in-process area.
- **Step 2**: Items continue through the process.
- **Step n**: The process continues until the item is done.
- **Done**: The item is completed and moved to the done area.
Lead time to customers -37%
Variance -47%
Development Time -73%; Variation -78%
Throughput: smaller, incremental deliveries
Fewer Bugs: -24%;
Variance: -33%
Continual Improvement
Days lost: -81% 26 to 5 days
Possible problems

1. Space needed for Kanban & info boards
2. Plan driven, document centric process
3. Poor fit with standardised reporting
4. Remit of IT – upstream & downstream
5. Command & control compliance model
6. Staff initiative and multi skilling

*Lean handles risk by low WIP, transparency, small units & frequent deliverables*
Lean & Kanban software process

- 37% reduction in lead time
- 47% reduction in lead time variation
- 73% reduction in development time
- 24% reduction in errors
- 33% reduction number of open errors
- 81% delays reduced continual imp.
- *8 increase in frequency of delivery

- Frequent small deliverables reduce risk
Value delivered

- The digital assets produced rose by hundreds of thousands of hours of content.

- 610% increase in valuable assets output by software products written by the team.
Differences Agile and Lean

- Batch / Push versus Pull
  - *Time-boxed iterations*
- Reliance on Data
  - *Focus on people*
- Continual Improvement
  - ‘*Velocity*, features, story points’
- Multiskilling
  - ‘*impediment list*’ / ‘*improvement backlog*’
- Evolution v. Revolution
Conclusion

- Lean applies from idea to release
- Iterates on continual customer feedback
- Software under quantitative control
- Pareto effect: 80 – 20 rule
- Frequent, small, high value deliverables
- Lean provides both discipline and agility

Follow up

• to the IEEE article

• p.middleton@qub.ac.uk
• dpjoyce@googlemail.com
• Twitter: @dpjoyce