Lean & Kanban 2009 Miami

May 6th-8th MANDARIN ORIENTAL

The Next Wave in Software Process

NetObjectives

VERSIONONE
Simplifying Software Delivery

Ultimate Software

InfoQ

Better Software Magazine

StickyMinds.com

David J Anderson & Associates
Management Consulting for Knowledge Workers

2008
Best Medium Company to Work for in America

Ranked #1
Best Medium Company to Work for in America
Scrumban: Lean Thinking for Agile Process Evolution

Corey Ladas
Modus Cooperandi
Lean Kanban 2009
**Kanban**

| For:       |  
|------------|--------------------------------------------------|
| Tool that infers reasonable estimates from historical data without nagging me for time stamps |  
| Due:       | 24 Mar                                           |
| Created:   | 1 Mar                                            |
| Started:   | 5 Mar                                            |
| Finished:  | 26 Mar                                           |

- **Return it to me when it’s done**
- **Halfway due**
- **Overdue!**
- **I need it by this date**

**Lead time**

**Cycle time**

**Make me a thing that does this**

**modus cooperandi**
Why pull? Why kanban?

- People with different skills have to work together to deliver product features.
- Don’t build features that nobody needs yet.
- Don’t test more code than you can deploy.
- Don’t write more code than you can test.
- Don’t write more specs than you can code.
Lean & kanban

• Watch out for cargo cult kanban!
• Kanban is a tool and a means to an end
• Evolutionary delivery and one-piece flow are far more interesting
• Kanban is interesting to the degree that it facilitates Evo & OPF
Lean thinking in knowledge work

- Analysis
- Design
- Testing
- Stabilization

Features

Lead time

Inventory
Life of a feature

1. Analyze
2. Design
3. Test
4. Build
5. Deploy

modus cooperandi
What is it that flows?

• Writing software does not resemble assembling a car
• If we want One-Piece Flow, what is the piece?
• Make work items value-oriented: features, use cases, scenarios
Evolutionary design

• The unit of production is an evolutionary enhancement
• Grow software, don’t build it
• Selection: parallel / spatial
• Iteration: serial / temporal
• We can do both!
Evolutionary design

Good evolutionary design is just good design!
Workcell organization

Analysis → Design → Features

Testing → Features

Features
Figure 1. Cleanroom Process Flow
Production leveling
Throughput management

- Throughput of what? *Business value*
- Manage latency thru standard work and SPC
- Manage bandwidth by policy!
- Kanban makes quantitative management easy
Throughput metrics

modus cooperandi
Why Scrumban?

• There is no “kanban process”
• Kanban needs process context for meaningful application
• Practitioners need examples of process transformation using kanban
• Scrumban is a simple story with a practical outcome that many people can relate to
• Scrumban is one of many possible stories about Lean transformation—we need more!
Is Kanban more Scrum than Scrum?

Have you read *The New New Product Development Game* lately?

---

**EXHIBIT 1**
Sequential (A) vs. overlapping (B and C) phases of development

<table>
<thead>
<tr>
<th>Phase</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Phase</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Phase</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Task cards

<table>
<thead>
<tr>
<th>to do</th>
<th>in process</th>
<th>done</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corey</td>
<td>David</td>
<td></td>
</tr>
<tr>
<td>Jim</td>
<td>Jim</td>
<td></td>
</tr>
<tr>
<td>David</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Corey</td>
<td></td>
</tr>
</tbody>
</table>
A problem with task cards
Multitasking limits and late binding

<table>
<thead>
<tr>
<th>to do</th>
<th>in process</th>
<th>done</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Jim</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Jim</td>
<td>David</td>
</tr>
<tr>
<td></td>
<td>David</td>
<td>Corey</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Kanban!

<table>
<thead>
<tr>
<th>to do</th>
<th>ready</th>
<th>2</th>
<th>in process</th>
<th>5</th>
<th>done</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- David
- Corey
- Jim
- Corey
Pull
<table>
<thead>
<tr>
<th>to do</th>
<th>ready</th>
<th>in process</th>
<th>done</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>5</td>
</tr>
</tbody>
</table>

- **Ready (2):**
  - David
  - Corey

- **In process (5):**
  - Jim
  - Corey
Pull
Workflow
Specialization
**Fixed-size backlog**

<table>
<thead>
<tr>
<th>backlog</th>
<th>ready</th>
<th>specify</th>
<th>complete</th>
<th>execute</th>
<th>done</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>Jim</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>David</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Corey</td>
<td></td>
</tr>
</tbody>
</table>

*Backlog items:*
- Item 1
- Item 2
- Item 3
- Item 4
- Item 5
- Item 6
- Item 7
- Item 8

*Ready items:*
- Item 1
- Item 2

*Specify items:*
- Corey

*Execute items:*
- Jim
- David
- Corey

*Done items:*
- Item 1
- Item 2
- Item 3
Order point
Business value

• ...is not the same thing as customer value
• ...or user utility
Minimum Marketable Feature

An approach to requirements definition and scheduling that aims to strike a good balance between keeping work requests small versus delivering value to the business.

http://www.SoftwareByNumbers.org
Minimum Marketable Feature

deluxe feature A → deluxe feature B

economy feature A → economy feature B → deluxe enhancement A1 → deluxe enhancement B → deluxe enhancement A2

modus cooperandi
Planning-on-demand

- The ideal work planning process should always provide the development team with best thing to work on next, no more and no less.

- Further planning beyond this does not add value and is therefore waste.
Lean planning games

• Pull scheduling
• Minimize NVA overhead
• Production leveling
• Balance stakeholder voices
Lean planning games

• Option thinking
• Decision markets
• Make options compete for inclusion in a planning portfolio
Lean planning games

Game theory:
• Mechanism design
• Auction systems
• Voting systems
Rolling Wave Planning

Different planning buckets for different time horizons:

• 6 week bucket: well-defined MMFs
• 3 month bucket: loosely-defined features
• 6 month bucket: broad feature areas
• 1 year bucket: strategies, goals, market forces
Rolling Wave Planning

MMFs only need to “gel” when we have to pick the next one for production

Update ALL of the buckets once every 6 weeks OR every time an MMF completes.
Real options

- **Stage** - A project can be divided into distinct stages where pursuit of each stage is contingent on a reassessment of costs and benefits at the time the preceding stage is completed.

- **Abandon** - A project can be terminated midstream and remaining project resources relatively easily redeployed.

- **Defer** - A decision on whether to invest can be deferred for some period without imperilling the potential benefits.

- **Strategic growth** - An initial baseline investment opens the door to pursue a variety of potential follow-on opportunities.

- **Change scale** - Resources allocated to a project can be contracted or expanded, or the operational system enabled by a project can be scaled up or down more easily.

- **Switch** - An asset developed for one purpose can be redeployed to serve another purpose (switch use). A key foundation technology supporting a project can be swapped out for another (switch inputs).
Incremental Funding Model

- “ship early, ship often”
- MMFs enable real options
- MMFs enable 80/20 rule
- Reduce risk and increase profitability by making project self-funding at earliest date
Feature crews

- Large-scale pull scheduling system used by Microsoft Office
- Cross-functional workcells for product features
- Internally self-organizing
Feature crews

- Branch-by-feature
- Quality gates
Inside the Feature Crew...

• You could use a traditional SDLC
• You could use Extreme Programming
• You could use Cleanroom
• You could use Scrumban

Whatever gets you through the gates!
Large-scale kanban systems

• Feature Crews + MMFs + Rolling Wave
• Architectural layering
• Software supply chains: Toyota is not vertically integrated!
• Matrix organization that works
Large-scale kanban systems
Thank you!

http://www.LeanSoftwareEngineering.com

http://www.ModusCooperandi.com