Kanban – Rediscovering the Agile vision?

Jesper Boeg, Trifork Software Pilot
jbo@trifork.com

October 4, 2010
In general

- Trifork A/S
  - Development
  - Training and conferences

- Let me know if:
  - You have questions (The most important thing is not covering every single slide)
  - What I’m saying does not make any sense at all

- My power point skills leave a lot to be desired
  - So please bear with my far from impressive slide designs
Agenda

- Kanban origins
- What is software Kanban?
- How is software Kanban different from other agile methods?
- Disadvantages
- Notes on plan driven iterations
KANBAN IN MANUFACTURING
A simple example of a Kanban pull system

- New paper is ordered when the limit prescribed by the Kanban is reached
- When paper arrives the Kanban is returned along with the paper
KANBAN IN SOFTWARE
Software Kanban is based on Lean Value Sets

- Limit work in progress.
  - Focus on flow not utilization
  - Deliver often
Lean Value Sets

- Stop the line mentality
Lean Value Sets

- Part of the culture and a state of mind
Lean Value Sets

- Balance demand and throughput
  - Sustainable pace – no “cell” should work at more than 80-85 percent capacity
  - Having free time on your hands
  - Optimizing the whole
Lean Value Sets

- Prioritize
  - Focus on business value and minimal marketable feature set
To achieve this

- Start by mapping the value stream and track work on a white board
Define WIP limits for each stage

PO is falling behind. Maybe I can help out when this story is finished

<table>
<thead>
<tr>
<th>PO Inbox</th>
<th>PO specification</th>
<th>Brakedown</th>
<th>Development</th>
<th>Code review</th>
<th>Test locally with PO and tester</th>
<th>Test on DT (Every morning)</th>
<th>Release (Every Tuesday)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Plan pairing
  - Planned
  - In progress
  - Done
  - Remember: Unit test
  - Int. test
  - Coverage
  - Dept. issue

- Tester and PO need 10 min. preparation

- Only testers should move to "release"
Pick the low hanging fruits

- You will be surprised how much you can achieve by
  - Mapping the value stream
  - Limiting work in progress.
  - Optimizing the whole
How does that fit with current Agile best practices?

- You can do fixed iterations or not
  - As long as you deliver often
- You can estimate or not
  - As long as you are able to do the necessary planning
- You can leave out iteration retrospectives
  - If you replace them with spontaneous quality circles or a better way to continuously improve
It does not mean:

- It is illegal to do iterations
- It is illegal to estimate
- It is not possible to do release planning
- You are not focusing on improving the way you work
Focusing on value sets instead of practices

- Using Kanban focus is no longer on specific practices
  - Choose practices that will help you use resources at hand most effectively in your context

- You might end up doing Scrum 😊
  - If Scrum practices are the perfect way to limit WIP, build quality in, level throughput and demand and prioritize according to business value in your context
But that is not my practice!!

David Anderson:  
“I don’t care about your practices”

- Keep your eyes on the ball  
  – We are hopefully using best practices because they deliver value
Cargo Cult

- Once practices become faith based and cargo cult we risk losing sight of the goal
Loosing control?

- Kanban is NOT a “looser” way of doing Scrum
  - Metrics are just different
Typical metrics

- Cycle/Lead time
- Quality
  - Time spend bugfixing per iteration
- WIP
  - Average number of “stories” in progress (queues)
- Throughput
SO HOW DOES THIS MAKE A DIFFERENCE?
Traditional agile methods have challenges

- Items small enough to fit a 2 week iteration are often too small to deliver real business value
  - Test becomes waste
  - Retrospectives become waste
  - Feedback becomes waste
Traditional agile methods have challenges

- Fixed iteration goals stress the entire system:
  - Product owners rush to prepare for upcoming cycles
  - Testers race to complete work late in the development time-box
  - Developers prioritize finishing a set of features over refactoring, TDD and pair programming
We need to allow more than one cadence

David Anderson: “Concept that input cadence, output cadence and cycle time should be synchronous e.g. 2 week iteration, will be seen as edge case 5 years from now”

- Seems reasonable to decouple prioritization, delivery and cycle time to vary naturally according to context and transaction costs
  - Actually one of the main reasons Kanbans are used in manufacturing
Keeping a sustainable pace

- Sustainable pace is a core value in agile – tech wise and people wise
  - But many “agile” projects exhibit anything but sustainable pace
  - Both in terms of stressed out people and a low quality code base

Accept that most traditional agile methods are feature driven and therefore require more counter measures “working software” to keep a sustainable code base
Sure we are doing better than Waterfall

- But why not question:
  - Stopping the development team for 1-2 days to do sprint planning and review?
  - Low quality feedback because functionality is too small to provide business value?
  - Stressing the real bottleneck/constraint by protecting the development team from external interruptions?
  - Planning “inventory” around development to avoid adjustments during the iteration?
Some of the potential benefits

- Better functional quality
- More/earlier refactoring
- Focus on the “real” bottleneck
- Faster feedback
- Lead time
- Lower inventory
- Level flow
Rediscovering the Agile vision?

- Why we use Agile methods:
  - Flow
  - Feedback
  - Quality built in
  - Close communication and collaboration across the entire value chain
  - Continuous improvement

- Valuing people over processes and tools
  - Should that not count for Agile processes and tools as well??
BUT THERE ARE NO FREE MEALS
Difficulties

- People react very differently to the new structure
  - Some find it very hard to stay focused while others take on more responsibility and become true craftsmen
Difficulties

- Takes more effort to stay focused on releases
Difficulties

- Stronger need for overall plans and long term goals
  - Since people are no longer as focused on the short term goal
Difficulties

- Controlling continuous integration
  - When features are increasingly branched and merged to trunk to allow for fixed release dates
Difficulties

- Wrong perception of Lean

And what do you think about lean?
Difficulties

- Many more will probably come since we have yet to see the long term effect
NOTES ON PLAN DRIVEN ITERATIONS
Plan driven iterations

- We are responsible for teaching our customers and ourselves
  - We will deliver exactly what we planned
  - The world is “Frozen” during the iteration
  - Business value should always fit a “2 week iteration”
Plan driven iterations

- From a Lean perspective iteration planning, test, deployment, equals - Batch production
Plan driven iterations

- Batch optimization is built on the faulty belief that processing big batches we can make the individual machine/fase go faster
  - Restricting flow
  - Increasing inventory
  - Reducing quality
Plan driven iterations

- “We can’t do 2 week iterations because of iteration review/planning overhead”
  - Shows you are still living in the old world of “Batch production” optimization
  - Instead focus on reducing transaction costs
Kanban is “Leaner” than traditional Agile Methods

- But remember to distinguish between Lean manufacturing and Lean Product Development
  - You cannot eliminate variability without eliminating value added in LPD
  - Cost of delay in manufacturing is often the same
Why I like fixed iteration length

- Lowers transaction costs
- Makes planning easier
- Facilitates continuous improvement
QUESTIONS?
Kontaktinfo

- Jesper Boeg
- Mail: jbo@trifork.com
- Mobile: +45 51 54 28 20
- Twitter: J_Boeg