Kanban vs Scrum
Making the most of both

JAOO, Aarhus
Oct 6, 2009

Henrik Kniberg
Agile/Lean coach @ Crisp, Stockholm

henrik.kniberg@crisp.se
+46 70 4925284
Purpose of this presentation

To clarify Kanban and Scrum by comparing them

...so you can figure out how these may come to use in your context.
Scrum in a nutshell

Split your organization

Split your product

Large group spending a long time building a huge thing
Small team spending a little time building a small thing
... but integrating regularly to see the whole

Split time

Optimize business value

Optimize process

January

April

Henrik Kniberg
Core Scrum
These are central to Scrum. Without these you probably shouldn’t call it Scrum.

- **Retrospective** happens after every sprint
  - Results in concrete improvement proposals
  - Some proposals actually get implemented
  - Whole team + PO participates

- **PO has a product backlog (PBL)**
  - Top items are prioritized by business value
  - Top items are estimated
  - Estimates written by the team
  - Top items in PBL small enough to fit in a sprint
  - PO understands purpose of all backlog items

- **Have sprint planning meetings**
  - PO participates
  - PO brings up-to-date PBL
  - Whole team participates
  - Results in a sprint plan
  - Whole team believes plan is achievable
  - PO satisfied with priorities

- **Timeboxed iterations**
  - Iteration length 4 weeks or less
  - Always end on time
  - Team not disrupted or controlled by outsiders
  - Team usually delivers what they committed to
  - Team members sit together

- **Max 9 people per team**

- **Team has a sprint backlog**
  - Highly visible
  - Updated daily
  - Owned exclusively by the team

- **Daily Scrum**
  - Whole team participates
  - Problems & impediments surfaced

- **Demo happens after every sprint**
  - Shows working, tested software
  - Feedback received from stakeholders & PO

- **Have Definition of Done (DoD)**
  - DoD achievable within each iteration
  - Team respects DoD

The bottom line
If you achieve these you can ignore the rest of the checklist. Your process is fine.

- Delivering working, tested software every 4 weeks or less
- Delivering what the business needs most
- Process is continuously improving

Core Scrum Checklist

Recommended but not always necessary
Most of these will usually be needed, but not always all of them. Experiment!

- Team has all skills needed to bring backlog items to Done
- Team members not locked into specific roles
- Iterations that are doomed to fail are terminated early
- PO has product vision that is in sync with PBL
- PBL and product vision is highly visible
- Everyone on the team participates in estimating
- PO available when team is estimating
- Estimate relative size (story points) rather than time
- Whole team knows top 1-3 impediments
- SM has strategy for how to fix top impediment
- SM focusing on removing impediments
- Escalated to management when team can’t solve
- Team has a Scrum Master (SM)
- SM sits with the team
- PO items are broken into tasks within a sprint
- Sprint tasks are estimated
- Estimates for ongoing tasks are updated daily
- Velocity is measured
- All items in sprint plan have an estimate
- PO uses velocity for release planning
- Velocity only includes items that are Done
- Team has a sprint burndown chart
- Highly visible
- Updated daily
- Daily Scrum is every day, same time & place
- Sprint tasks are estimated
- SM sits with the team
- Max 15 minutes
- Each team member knows what the others are doing

Scaling
These are pretty fundamental to any Scrum scaling effort.

- You have a Chief Product Owner (if many POs)
- Dependent teams do Scrum of Scrum
- Dependent teams integrate within each sprint

Positive indicators
Leading indicators of a good Scrum implementation.

- Having fun! High energy level.
- Overtime work is rare and happens voluntarily
- Discussing, criticizing, and experimenting with the process

PO = Product owner  SM = Scrum Master  PBL = Product Backlog  DoD = Definition of Done

Typical waterfall => Scrum evolution

1. Waterfall
   - Requirements
   - Design
   - Code
   - Test

2. "ScrumButt"
   - Requirements
   - Design & code
   - Test

3. Scrum
   - Feature team 1
   - Feature team 2

Henrik Kniberg
Kanban in a nutshell

- Visualize the workflow
- Limit WIP (work in progress)
- Measure & optimize flow

Useful starting point for more info:
http://www.limitedwipsociety.org
Roots of Kanban (Toyota)

The two pillars of the Toyota production system are just-in-time and automation with a human touch, or autononation. The tool used to operate the system is kanban.
Kanban in software development
Typical Scrum => Kanban evolution

Scrum step 1

Feature team 1
Scrum

Feature team 2
Scrum

Feature team 2
Scrum

Scrum step 2

Feature team 1
Scrum

Feature team 2
Scrum

Feature team 2
Scrum

Scrum + Kanban

Feature team 1
Scrum

Feature team 2
Scrum

Feature team 2
Scrum

Operations / support team

Scrum

Operations / support team

Kanban

Henrik Kniberg
Can we compare Kanban and Scrum?
Should we?
**Tool**

“anything used as a means of accomplishing a task or purpose.”
- dictionary.com

**Physical tools**

**Thinking tools**
a.k.a. “mindsets” or “philosophies”
- Lean
- Agile
- Systems Thinking
- Theory of Constraints

**Process tools**
a.k.a. “organizational patterns”
- Pair programming
- Visualize the workflow

**Toolkits**
a.k.a. “frameworks”
- Scrum
- RUP
- XP
- Kanban

**Product Owner role**

- Henrik Kniberg

- **c**risp

- **Flow**

- **To do**: 5
- **Dev**: 3
- **Test**: 2
- **Release**: 3
- **Done**: 1

- **A**: H, I, J
- **B**: E, F
- **C**: D

- **FLOW**
Any tool can be misused

and will

The old tool was better!

Never blame the tool!
Compare for understanding, not judgement

More prescriptive

More adaptive

RUP (120+)

XP (13)

Scrum (9)

Kanban (3)

Do Whatever (0)

Do not develop an attachment to any one weapon or any one school of fighting

Miyamato Musashi
17th century samurai
Distinguish between the tool itself from specific usage techniques

Specific patterns, techniques, "best practices", etc
Scrum prescribes 3 roles

- **Product owner**
- **Team Scrum Master**
- **Team**
Scrum prescribes timeboxed iterations

Scrum team

Kanban team 1

Kanban team 2

Kanban team 3

Henrik Kniberg
Both limit WIP, but in different ways

Scrum board

Kanban board

**WIP limited per unit of time (iteration)**

**WIP limited per workflow state**
Both are empirical

<table>
<thead>
<tr>
<th>Capacity</th>
<th>Lead time</th>
<th>Quality</th>
<th>Predictability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Many small teams</td>
<td>Few large teams</td>
<td>Low WIP limits</td>
<td>High WIP limits</td>
</tr>
<tr>
<td>Few workflow states</td>
<td>Many workflow states</td>
<td>Short iterations</td>
<td>Long iterations</td>
</tr>
<tr>
<td>Little planning</td>
<td>Lots of planning</td>
<td>.... etc ...</td>
<td>.... etc ...</td>
</tr>
</tbody>
</table>

Kanban is more configurable

Great! More options!  Oh no, more decisions!
Scrum discourages change in mid-iteration

Scrum

<table>
<thead>
<tr>
<th>To do</th>
<th>Ongoing</th>
<th>Done :o)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Kanban

<table>
<thead>
<tr>
<th>To do</th>
<th>Ongoing</th>
<th>Done :o)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>D</td>
<td>2</td>
<td>B</td>
</tr>
</tbody>
</table>

- Wait until next sprint!
- I’d like to have E!
- Wait until a To Do slot becomes available! Or swap out C or D!

Policies
Scrum board is reset between each iteration

**Scrum**
- **First day of sprint**
  - Diagram showing tasks in progress
- **Mid-sprint**
  - Diagram showing tasks in progress
- **Last day of sprint**
  - Diagram showing tasks in progress

**Kanban**
- **Any day**
  - Diagram showing tasks in progress

Henrik Kniberg
Scrum prescribes cross-functional teams

Scrum team
Kanban team 1

Cross-functional team

Kanban team 2

Specialist
Cross-functional team
Specialist team
Scrum backlog items must fit in a sprint

Scrum

<table>
<thead>
<tr>
<th>Sprint 1</th>
<th>Sprint 2</th>
<th>Sprint 3</th>
<th>Sprint 4</th>
</tr>
</thead>
</table>

Kanban

WIP limit = 3

Long running task

Henrik Kniberg
Scrum prescribes estimation and velocity

Likely velocity: 8 per sprint (sustainable pace?)
Both allow working on multiple products simultaneously

Scrum example 1
Green Product
Green team
Yellow Product
Yellow team

Scrum example 2
All products
Cross-product team

Scrum example 3
All products
Cross-product team

Kanban example 1
Color-coded tasks

Kanban example 2
Color-coded swimlanes

Henrik Kniberg
Both are Lean and Agile

The Toyota Way
1. Base your management decisions on a Long-Term Philosophy, Even at the Expense of Short-Term Financial Goals
2. Create Continuous Process Flow to Bring Problems to the Surface
3. Use Pull Systems to Avoid Overproduction
4. Level Out the Workload (Heijunka)
5. Build a Culture of Stopping to Fix Problems, to Get Quality Right the First Time
6. Standardized Tasks are the Foundation for Continuous Improvement and Employee Empowerment
7. Use Visual Controls So No Problems are Hidden
8. Use Only Reliable, Thoroughly Tested Technology That Serves Your People and Processes
9. Grow Leaders Who Thoroughly Understand the Work, Live the Philosophy, and Teach It to Others
10. Develop Exceptional People and Teams Who Follow Your Company’s Philosophy
11. Respect Your Extended Network of Partners and Suppliers by Challenging Them and Helping Them Improve
12. Go and See for Yourself to Thoroughly Understand the Situation (Genchi Genbutsu)
13. Make Decisions Slowly by Concensus, Thoroughly Considering All Options; Implement Decisions Rapidly
14. Become a Learning Organization Through Relentless Reflection (Hansei) and Continuous Improvement (Kaizen)

Agile Manifesto
1. Individuals and Interactions over Processes and Tools
2. Working Software over Comprehensive Documentation
3. Customer Collaboration over Contract Negotiation
4. Responding to Change over Following a Plan

Henrik Kniberg
Minor difference:
Scrum prescribes a prioritized product backlog

Scrum:
- Product backlog must exist
- Changes to product backlog take effect next sprint (not current sprint)
- Product backlog must be sorted by “business value”

Kanban:
- Product backlog is optional
- Changes to product backlog take effect as soon as capacity becomes available
- Any prioritization scheme can be used. For example:
  - Take any item
  - Always take the top item
  - Always take the oldest item
  - 20% on maintainance items, 80% on new features
  - Split capacity evenly between product A and product B
  - Always take red items first
Minor difference:
Scrum prescribes daily meetings

... but many Kanban teams do that anyway.
Minor difference:
In Scrum, burndown charts are prescribed.

No specific types of diagrams prescribed in Kanban. Teams use whatever they need.
Example: Scrum board vs Kanban board

Scrum

Kanban

Henrik Kniberg
Evolve your own unique board!

Some of these photos courtesy of David Anderson, Mattias Skarin, and various other people.
Scenario 1 – one piece flow

<table>
<thead>
<tr>
<th>Backlog</th>
<th>Selected</th>
<th>Dev</th>
<th>In production :o)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>A B G C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F D H I J</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L K M</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Ongoing
- Done
Scenario 1 – one piece flow

<table>
<thead>
<tr>
<th>Backlog</th>
<th>Selected 2</th>
<th>Dev 3</th>
<th>In production :o)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A</td>
<td>Ongoing</td>
<td>Done</td>
</tr>
</tbody>
</table>
### Scenario 1 – one piece flow

![Diagram of one piece flow](image)

- **Backlog**
  - G
  - F
  - H
  - J
  - M

- **Selected**
  - C
  - D
  - I
  - L
  - K

- **Dev**
  - B
  - A
  - Ongoing
  - Done

- **In production :o)**
Scenario 1 – one piece flow

<table>
<thead>
<tr>
<th>Backlog</th>
<th>Selected 2</th>
<th>Dev 3</th>
<th>In production :o)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>J</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Henrik Kniberg
### Scenario 1 – one piece flow.

<table>
<thead>
<tr>
<th>Backlog</th>
<th>Selected</th>
<th>Dev</th>
<th>In production :o)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ongoing</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Done</td>
</tr>
<tr>
<td>G</td>
<td>D</td>
<td>C</td>
<td>A</td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>I</td>
<td></td>
<td></td>
</tr>
<tr>
<td>J</td>
<td>L</td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>K</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Henrik Kniberg
Scenario 2 – Deployment problem
Scenario 2 – Deployment problem

Backlog | Selected 2 | Dev 3 | In production :o)
---------|-----------|-------|-------------------
G        | A         |       |                   
C        | B         |       |                   
F        |           |       |                   
D        |           |       |                   
H        |           |       |                   
I        |           |       |                   
J        |           |       |                   
L        |           |       |                   
E        |           |       |                   
K        |           |       |                   
M        |           |       |                   

Henrik Kniberg
Scenario 2 – Deployment problem

Backlog

Selected

Dev

In production :o)

<table>
<thead>
<tr>
<th>G</th>
<th>C</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>D</td>
<td>B</td>
</tr>
</tbody>
</table>

Henrik Kniberg
Scenario 2 – Deployment problem

<table>
<thead>
<tr>
<th>Backlog</th>
<th>Selected</th>
<th>Dev</th>
<th>In production :o)</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>I</td>
<td></td>
<td></td>
</tr>
<tr>
<td>J</td>
<td>L</td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>K</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

POI | G

Ongoing: C, B
Done: A

In production :o)
### Scenario 2 – Deployment problem

#### Backlog

<table>
<thead>
<tr>
<th>G</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>I</td>
</tr>
<tr>
<td>J</td>
<td>L</td>
</tr>
<tr>
<td>M</td>
<td>K</td>
</tr>
</tbody>
</table>

#### Selected

2

#### Dev

<table>
<thead>
<tr>
<th>Ongoing</th>
<th>Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>A</td>
</tr>
<tr>
<td>!?</td>
<td>B</td>
</tr>
</tbody>
</table>

#### In production

:o)
Scenario 2 – Deployment problem

Backlog | Selected 2 | Dev 3 | In production :o)

| G | D | !? | A |
| F | E |   |   |
| H |   |   | B |
| I |   |   | C |
| J |   |   |   |
| L |   |   |   |
| M |   |   |   |
| K |   |   |   |
Scenario 2 – Deployment problem

Backlog | Selected | Dev | In production :o)
|----------|----------|-----|------------------
|          | 2        | 3   |                  
| G        | D        | A   |                  
| F        | E        | B   |                  
| H        |          | C   |                  
| I        |          |     |                  
| J        |          |     |                  
| L        |          |     |                  
| M        |          |     |                  
| K        |          |     |                  

Henrik Kniberg
Scenario 2 – Deployment problem

Backlog | Selected 2 | Dev 3 | In production :o)

| G | D | A |
| F | E | B |
| H |   |   |
| I |   |   |
| J |   |   |
| L |   |   |
| M |   |   |
| K |   |   |

POI
## Scenario 2 – Deployment problem

### Diagram Description

<table>
<thead>
<tr>
<th>Backlog</th>
<th>Selected 2</th>
<th>Dev 3</th>
<th>In production :o)</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>J</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Ongoing**: D
- **Done**: E

### Notes

- In production :o) - A, B
"One day in Kanban land"

http://blog.crisp.se/henrikkniberg/tags/kanban/
## Kanban vs Scrum

### Summary

**Similarities**
- Both are Lean and Agile
- Both based on pull scheduling
- Both limit WIP
- Both use transparency to drive process improvement
- Both focus on delivering releasable software early and often
- Both are based on self-organizing teams
- Both require breaking the work into pieces
- In both cases the release plan is continuously optimized based on empirical data (velocity / lead time)

**Differences**

<table>
<thead>
<tr>
<th>Scrum</th>
<th>Kanban</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeboxed iterations prescribed.</td>
<td>Timeboxed iterations optional.</td>
</tr>
<tr>
<td>Team commits to a specific amount of work for this iteration.</td>
<td>Commitment optional.</td>
</tr>
<tr>
<td>Uses <strong>Velocity</strong> as default metric for planning and process improvement.</td>
<td>Uses <strong>Lead time</strong> as default metric for planning and process improvement.</td>
</tr>
<tr>
<td><strong>Cross-functional teams</strong> prescribed.</td>
<td><strong>Cross-functional teams</strong> optional. <strong>Specialist teams allowed.</strong></td>
</tr>
<tr>
<td><strong>Items broken down</strong> so they can be completed within 1 sprint.</td>
<td>No particular item size is prescribed.</td>
</tr>
<tr>
<td>Burndown chart prescribed</td>
<td>No particular type of diagram is prescribed.</td>
</tr>
<tr>
<td>WIP limited indirectly (per sprint)</td>
<td><strong>WIP limited directly</strong> (per workflow state)</td>
</tr>
<tr>
<td>Estimation prescribed</td>
<td><strong>Estimation optional</strong></td>
</tr>
<tr>
<td>Cannot add items to ongoing iteration.</td>
<td>Can add new items whenever capacity is available</td>
</tr>
<tr>
<td>A sprint backlog is owned by one specific team</td>
<td>A kanban board may be shared by multiple teams or individuals</td>
</tr>
<tr>
<td>Prescribes 3 roles (PO/SM/Team)</td>
<td>Doesn’t prescribe any roles</td>
</tr>
<tr>
<td>A Scrum board is reset between each sprint</td>
<td>A kanban board is persistent</td>
</tr>
<tr>
<td>Prescribes a prioritized product backlog</td>
<td>Prioritization is optional.</td>
</tr>
</tbody>
</table>

[Henrik Kniberg](https://www.crisp.se/henrik.kniberg/kanban-vs-scrum.pdf)
Don’t be dogmatic

Go away! Don’t talk to us! We’re in a Sprint.

Come back in 3 weeks.

Though Shalt Limit WIP

Beware of Dogma
Essential skills needed both Kanban and Scrum

Splitting the system into deliverable increments

Software craftsmanship

Retrospectives

Root-cause analysis

Henrik Kniberg

Take-away points

1. **Know your goal**
   - Hint: Agile/Lean/Kanban/Scrum isn’t it.

2. **Never blame the tool**
   - Tools don’t fail or succeed. People do.
   - There is no such thing as a good or bad tool. Only good or bad decisions about when, where, how, and why to use which tool.

3. **Don’t limit yourself to one tool**
   - Learn as many as possible.
   - Compare for understanding, not judgement.

4. **Experiment & enjoy the ride**
   - Don’t worry about getting it right from start.
   - The only real failure is the *failure to learn* from failure.