Lean Software & Kanban

Software Engineering II

WS 2019/20

Enterprise Platform and Integration Concepts

Lean Software's Main Idea

Reduce Waste

Waste

- Anything not delivered to the customer
- Artifacts that do not deliver a business value

Lean software inspired by

- Toyota's "lean manufacturing" industrial production
- Just-in-time production

Lean Software's Principles

1. **Eliminate Waste**
   - Anything not delivering business value
   - E.g. requirements documents, partially done work, rarely used features (bloat), bugs, task switching, waiting
2. **Amplify learning**
   - "Try-it, test-it, fix-it" rather than "do it right the first time"
   - Short iteration cycles
3. **Decide as late as possible**
   - Avoid up front design decisions, make choices when information is available
4. **Deliver as fast as possible**
   - Working system at every iteration, fast feedback cycle

Lean Software's Principles

5. **Empower the team**
   - Motivate the team, self-organization
   - "Find good people and let them do their own job"

6. **Build integrity in**
   - Maintain the consistency of a system's design
   - E.g. through refactoring, automated tests, complete build system

7. **See the whole**
   - Focus on overall progress of the project
   - Strong common sense

Lean Software Summary

- **Idea:** Software development can benefit from industrial production recipes
- However, software has no production, only design
- “Lean” can be seen as more philosophy than method
- Reminder to look out for waste of any kind

Kanban

(看板)

Flow is constricted by bottlenecks
(at full capacity, there is little throughput)

Kanban's Main Idea

Minimize Work In Progress

- Kanban: literally "signboard" or "billboard" in Japanese
- Inspired by Toyota
- Visual process-management approach ("Kanban boards")

- "Stop Starting, Start Finishing"
- Ensure just-in-time production

Core Kanban Practices

1. **Limit work in progress (Kanban Limits)**
   - Limit amount of tickets per column
   - Focus on most productive task for the project
   - Pull work from previous columns
   - Reduce context switching (waste)

2. **Visualize**
   - Shared Kanban-Board with process steps as columns
   - Requirements (tasks, user stories,...) travel as notes from left to right.

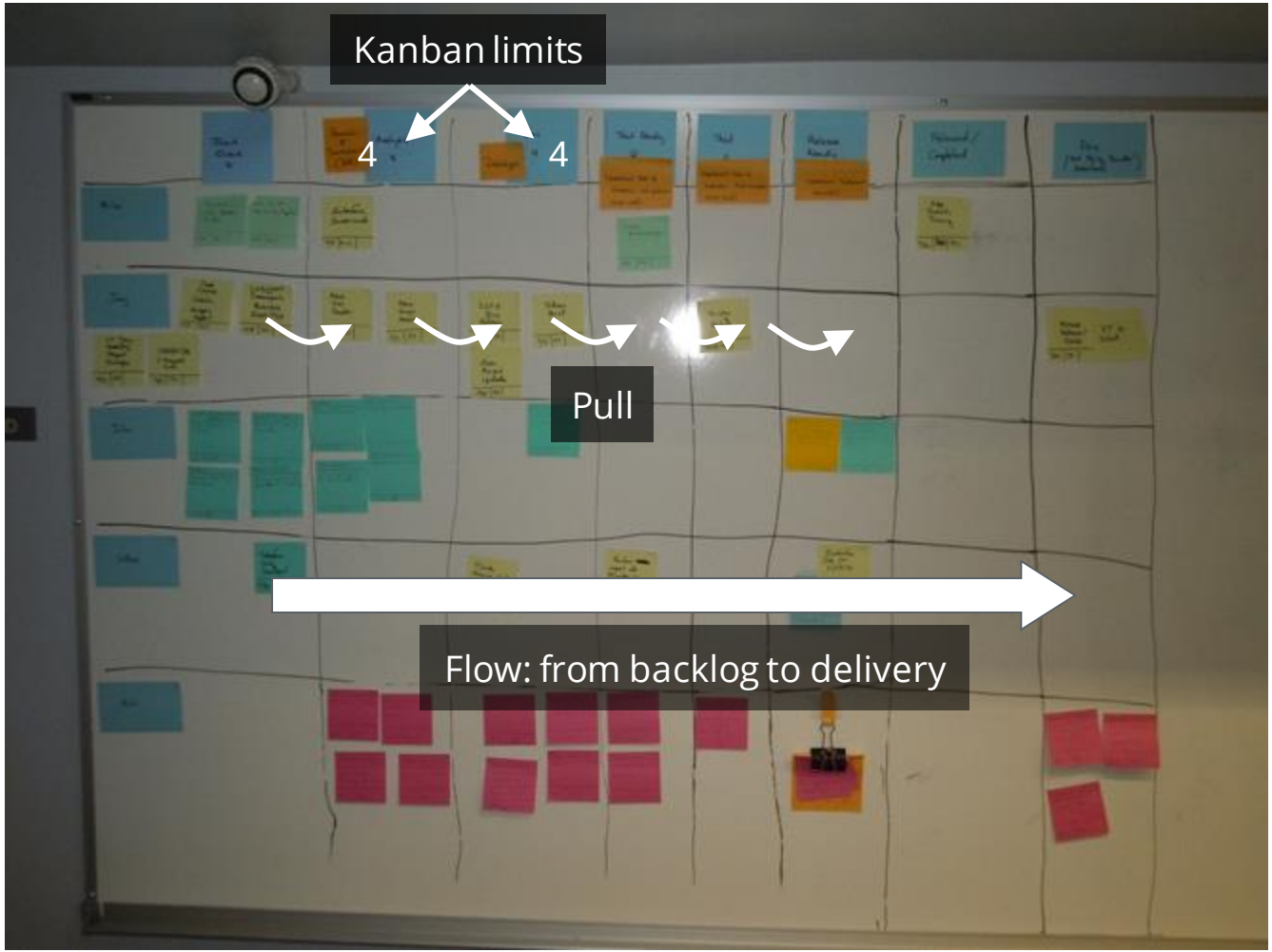
3. **Manage flow**
   - Measure length of queue, average cycle time and throughput
   - Identify bottlenecks and allow planning

Core Kanban Practices

4. **Make policies explicit**
   - Create explicit shared understanding of rules and assumptions
   - E.g. what columns mean, Definition of Done, which ticket to pull next

5. **Implement feedback loops**
   - Process of continuous improvement (“kaizen” in Japanese)
   - Don’t wait for feedback, build it into the process

6. **Improve collaboratively, evolve experimentally**
   - Try things out, evaluate



Push vs. Pull Systems

- How a team handles the inventory & scheduling of work items
- **Push production**: based on forecasted demand, schedule
- **Pull production**: based on consumed products, take only what is needed, process immediately

It can be argued that Scrum is a pull-based system as well, with work pulled in larger batches (the Sprint Backlog).

Scrum Sprint Planning: **Push**

- Forecasted demand (business needs)
- Estimated capacity of team

Kanban: **Pull**

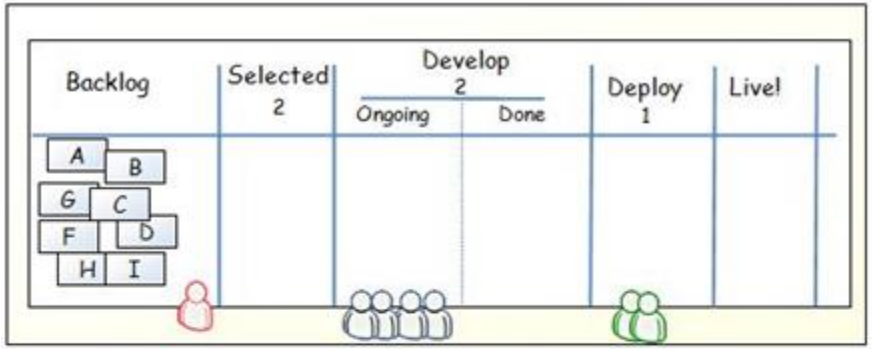
- No need for planning, no queue to push into
- Stories worked on based on actual demand and actual capacity

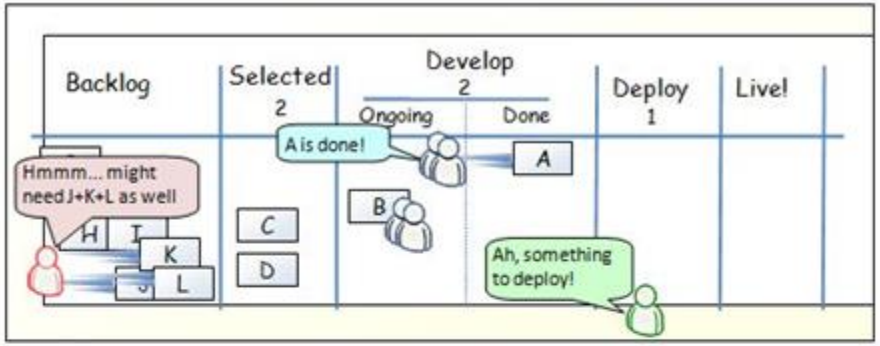
http://shanekercheval.me/blog/kanban-vs-scrum-pull-vs-push/

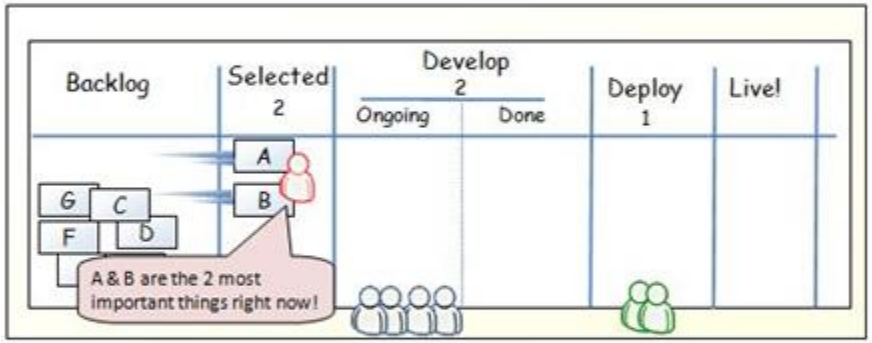
One day in Kanban Land

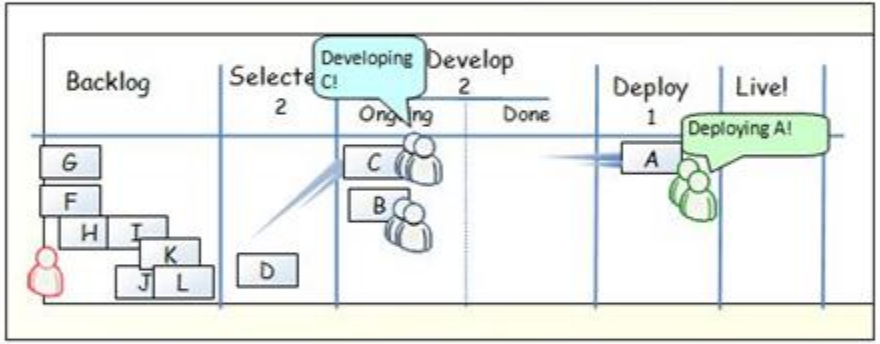
(A comic by Henrik Kniberg)

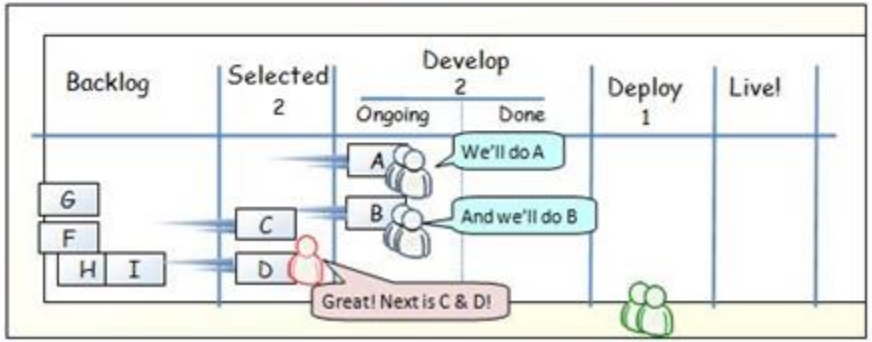
Henrik Kniberg http://blog.crisp.se/2009/06/26/henrikkniberg/1246053060000

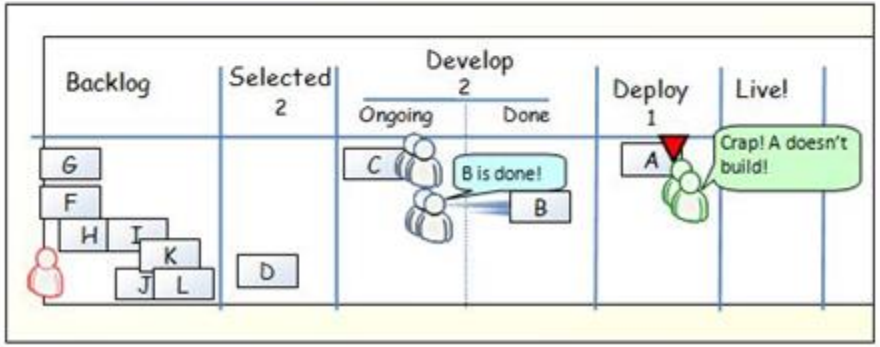

Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
A
B
G
C
F
D
H
I

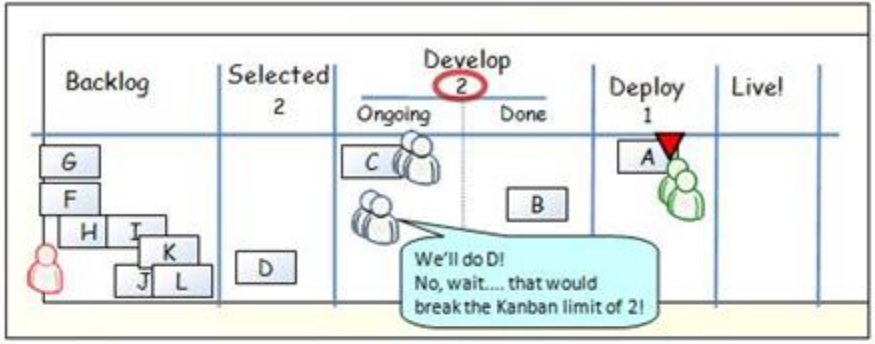

Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
A is done!
Hmmm... might need J+K+L as well
Ah, something to deploy!
A
B
C
D
H
I
K
L

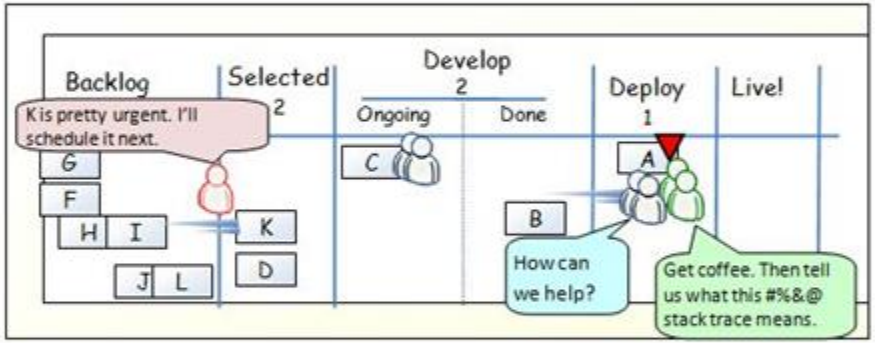

Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
A
B
G
C
F
D
A & B are the 2 most important things right now!

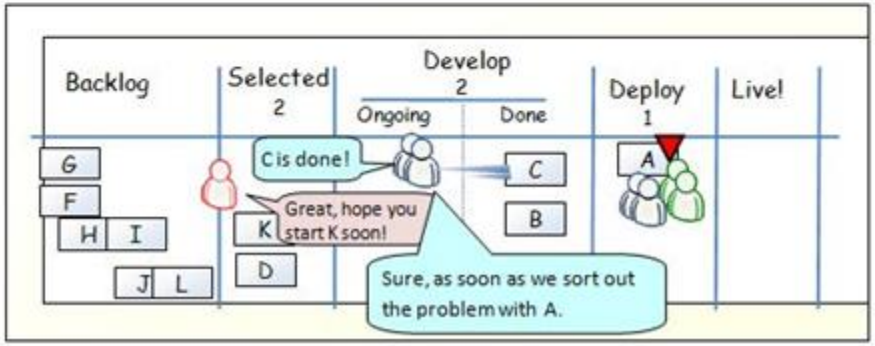

Backlog
Selected 2
Developing C!
Develop 2
Ongoing
Done
Deploy 1
Live!
Deploying A!
G
F
H
I
K
J
L
D
C
B
A

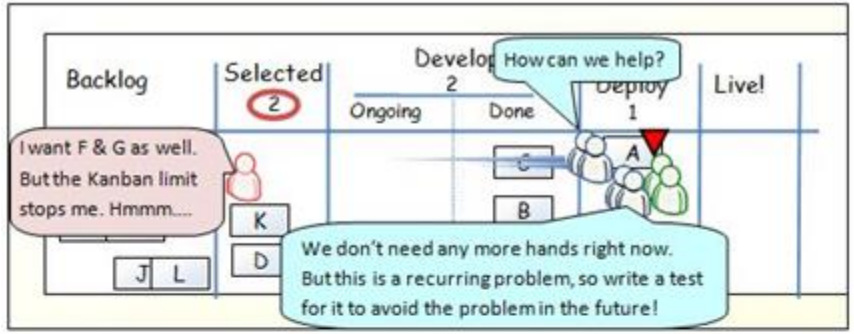

Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
We'll do A
And we'll do B
Great! Next is C & D!
G
F
H
I
A
B
C
D

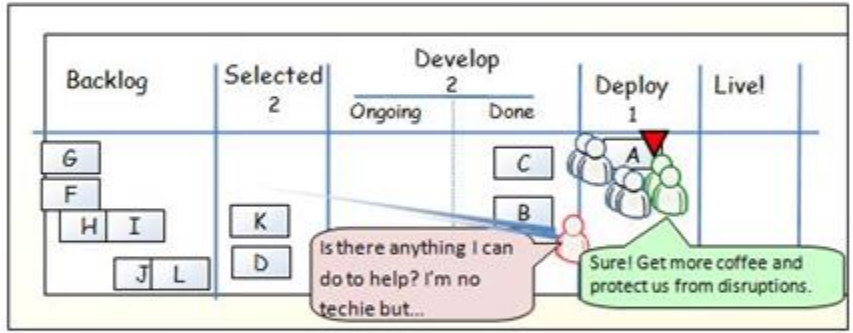

Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
B is done!
Crap! A doesn't build!
G
F
H
I
K
J
L
D
C
B
A

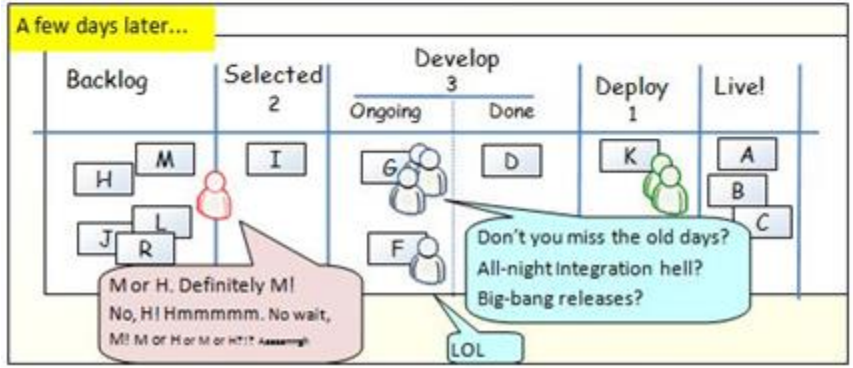

Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
G
F
H
I
K
J
L
D
C
B
A
We'll do D!
No, wait.... that would break the Kanban limit of 2!


Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
K is pretty urgent. I'll schedule it next.
How can we help?
Get coffee. Then tell us what this #%&@ stack trace means.


Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
C is done!
Great, hope you start K soon!
Sure, as soon as we sort out the problem with A.


Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
How can we help?
I want F & G as well. But the Kanban limit stops me. Hmmm.....
We don't need any more hands right now. But this is a recurring problem, so write a test for it to avoid the problem in the future!

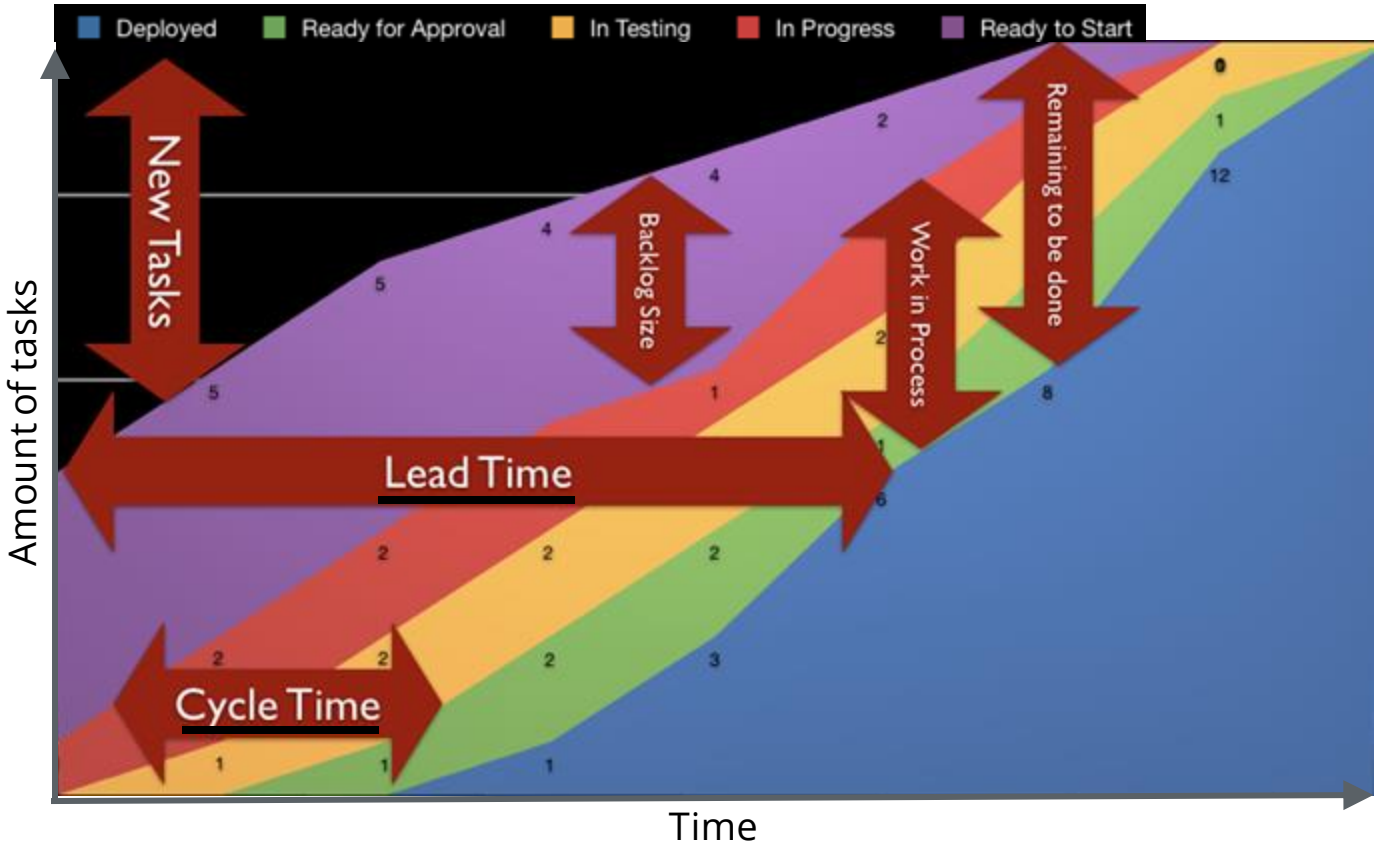

Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
Is there anything I can do to help? I'm no techie but...
Sure! Get more coffee and protect us from disruptions.


A few days later...
Backlog
Selected 2
Develop 3
Ongoing
Done
Deploy 1
Live!
M or H. Definitely M! No, H! Hmmmmm. No wait, M! M or H or M or H?!?
Don't you miss the old days? All-night integration hell? Big-bang releases?
LOL

Metrics — Cumulative Flow Diagram

Lead Time:
Time from ticket being placed on board to ticket delivered

Cycle Time:
Time from starting work on ticket to finishing



Discussion

- Main differences between Scrum and Kanban?
- What could be the biggest challenges when employing Kanban?
- What would be a "Kanban Master's" tasks?
  Is this a necessary role?
- What domains/contexts is...
  - Kanban better suited for?
  - Scrum better suited for?

References

- *"Agile!: The Good, the Hype and the Ugly"*. Bertrand Meyer. Springer Publishing Company, Incorporated, 2014. ISBN 9783319051543.
- *"Stop Starting, Start Finishing!"*. Arne Roock. Illustrated Edition. Blue Hole Press, 2012. ISBN 9780985305161.
- *"Kanban: Successful Evolutionary Change for Your Technology Business"*. David J. Anderson. Blue Hole Press, 2010. ISBN 0984521402.
- *"Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise"*. Mark Lines, Scott W. Ambler. IBM Press, 2012. ISBN 978-0-13-281013-5.