Can Kanban be used with/instead of LeSS or Scrum?

WHO is asking and WHY?

Names That We Should Remember…

When we talk about…

KANBAN

Taiichi Ohno

Scrum

Jeff Sutherland Ken Schwaber

LeSS

Bas Vodde Craig Larman

Who Is Asking? Why?

- Real Teams?
- Enabling coaches and Scrum Masters?
- Product Owner? Users?
- Sponsors? Other Stakeholders?
- Executives?
- Line managers?
- Legal / Compliance?
- Methodology gurus? ;)
- Tooling experts? ;) ;) ;)

Questioning assumes a certain (minimum) degree of knowledge and awareness.

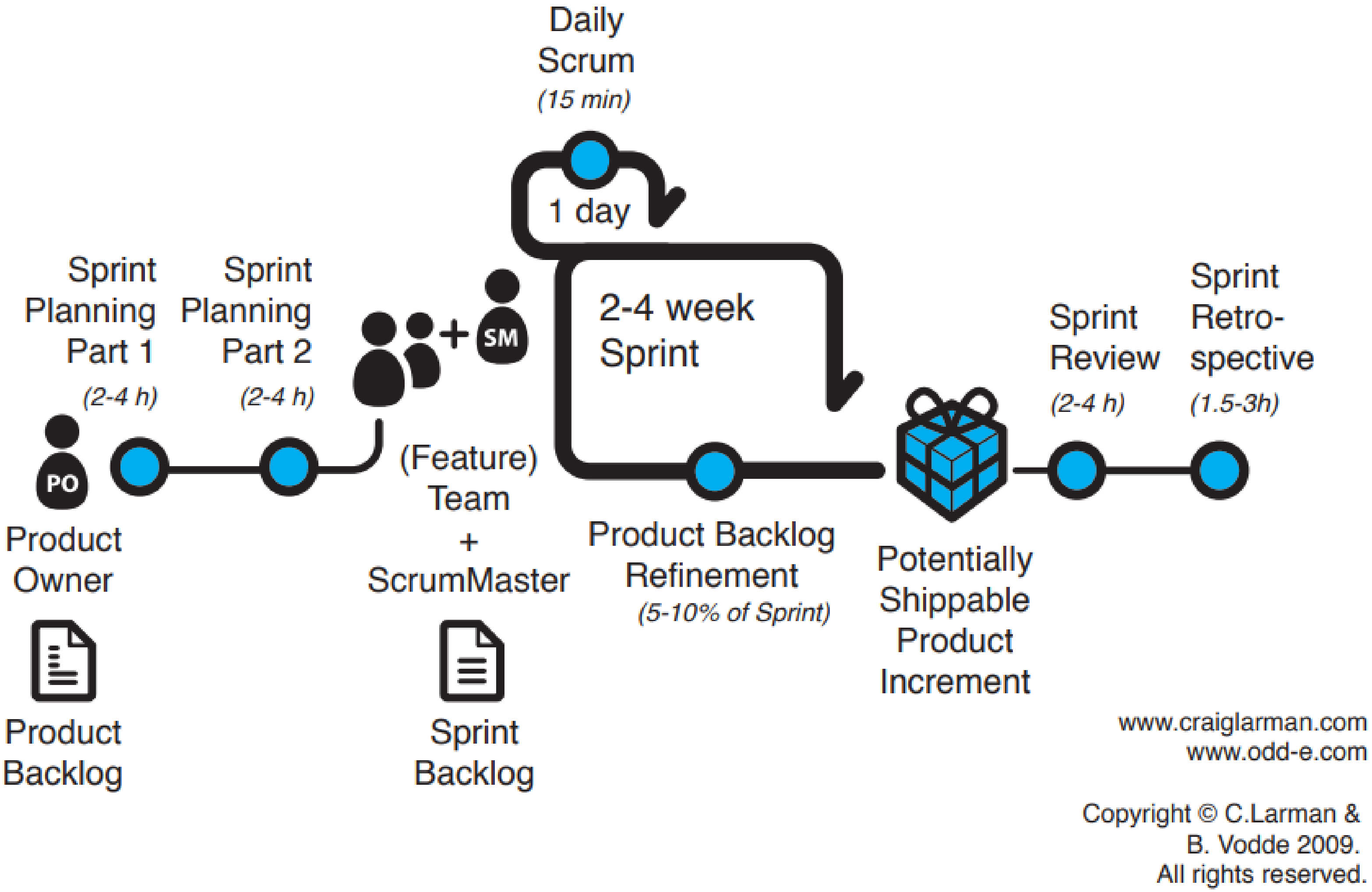
Scrum

“Scrum isn't designed for method tailoring. ***Ken Schwaber*** *notes that "****Scrum is not a methodology that needs enhancing. That is how we got into trouble in the first place, thinking that the problem was not having a perfect methodology.*** *Effort centers on the changes in the enterprise that is needed."[43]*

Bas Vodde *reinforces this statement, suggesting that Scrum isn't like traditional, large methodologies that require you to "pick and choose" elements. It is the basics on top of which you add additional elements to localize and contextualize its use.[44]”*

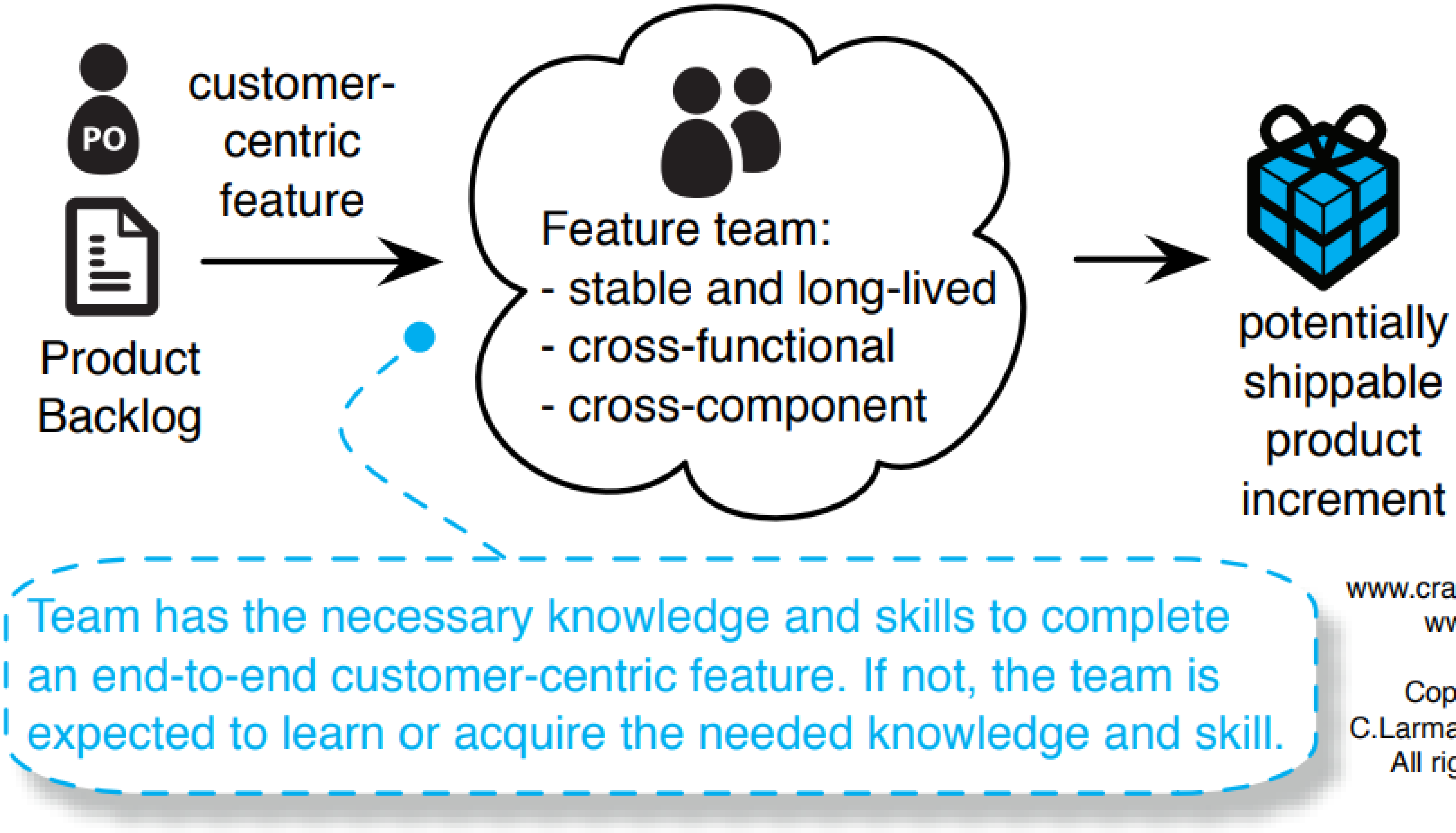
Source: https://en.wikipedia.org/wiki/Agile_software_development

Scrum


Daily Scrum (15 min)
1 day
Sprint Planning Part 1 (2-4 h)
Sprint Planning Part 2 (2-4 h)
SM
PO
2-4 week Sprint
Sprint Review (2-4 h)
Sprint Retro-spective (1.5-3h)
Product Owner
(Feature) Team + ScrumMaster
Product Backlog Refinement (5-10% of Sprint)
Potentially Shippable Product Increment
Product Backlog
Sprint Backlog
www.craiglarman.com
www.odd-e.com
Copyright © C.Larman & B. Vodde 2009. All rights reserved.

https://less.works/scaling-book-images/scaling-agile-lean-development-thinking-tools/unpublished-one-team-scrum-en.pdf

Scrum



PO

Product Backlog

customer-centric feature

Feature team:

- stable and long-lived
- cross-functional
- cross-component

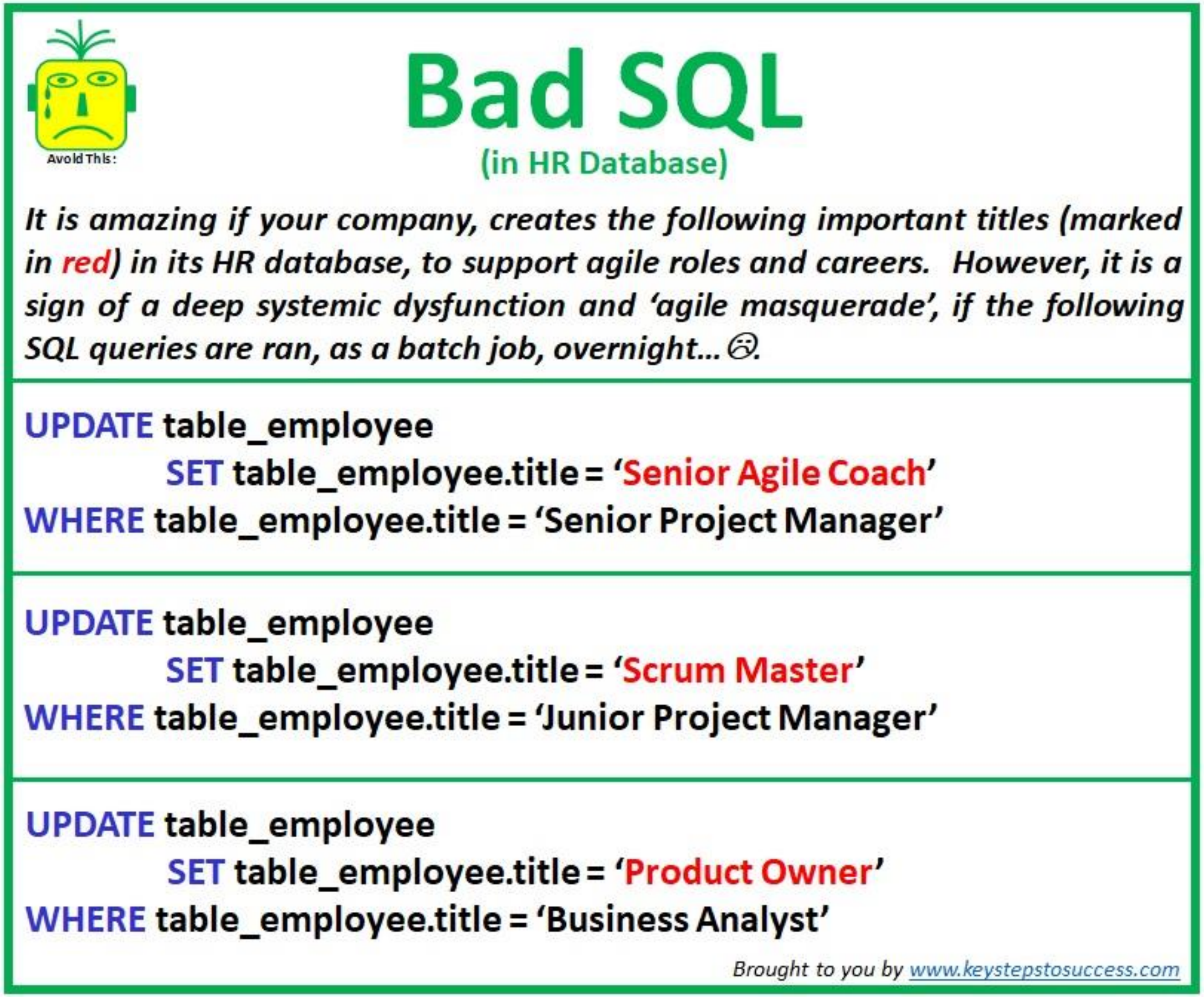
potentially shippable product increment

Team has the necessary knowledge and skills to complete an end-to-end customer-centric feature. If not, the team is expected to learn or acquire the needed knowledge and skill.

www.craiglarman.com
www.odd-e.com

https://less.works/scaling-book-images/practices-scaling-lean-agile-development/chapter-15-feature-team-en.pdf

Scrum Anti-patterns



Avoid This:

Bad SQL

(in HR Database)

It is amazing if your company, creates the following important titles (marked in red) in its HR database, to support agile roles and careers. However, it is a sign of a deep systemic dysfunction and 'agile masquerade', if the following SQL queries are ran, as a batch job, overnight…☹.

```
UPDATE table_employee
          SET table_employee.title = 'Senior Agile Coach'
WHERE table_employee.title = 'Senior Project Manager'
```

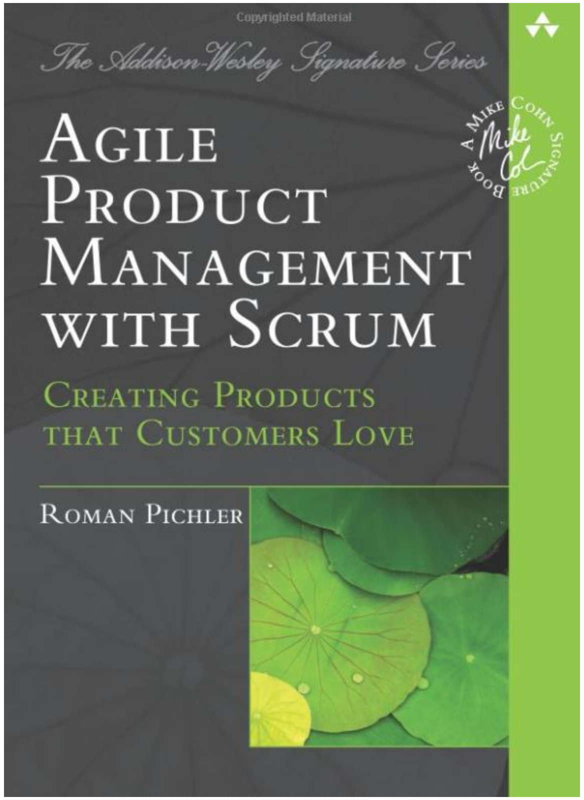
```
UPDATE table_employee
          SET table_employee.title = 'Scrum Master'
WHERE table_employee.title = 'Junior Project Manager'
```

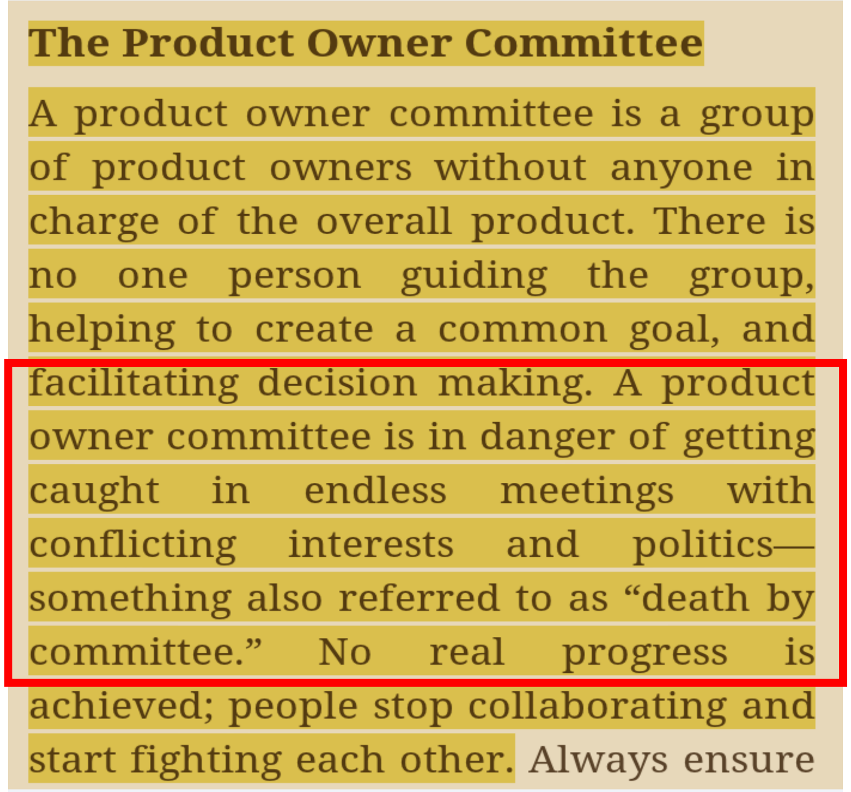
```
UPDATE table_employee
          SET table_employee.title = 'Product Owner'
WHERE table_employee.title = 'Business Analyst'
```

Brought to you by www.keystepstosuccess.com

https://www.keystepstosuccess.com/wp-content/uploads/2020/04/bad_agile_sql-1.jpg

Scrum Anti-patterns


Copyrighted Material
The Addison-Wesley Signature Series
Agile Product Management with Scrum
Creating Products that Customers Love
A Mike Cohn Signature Book
Roman Pichler

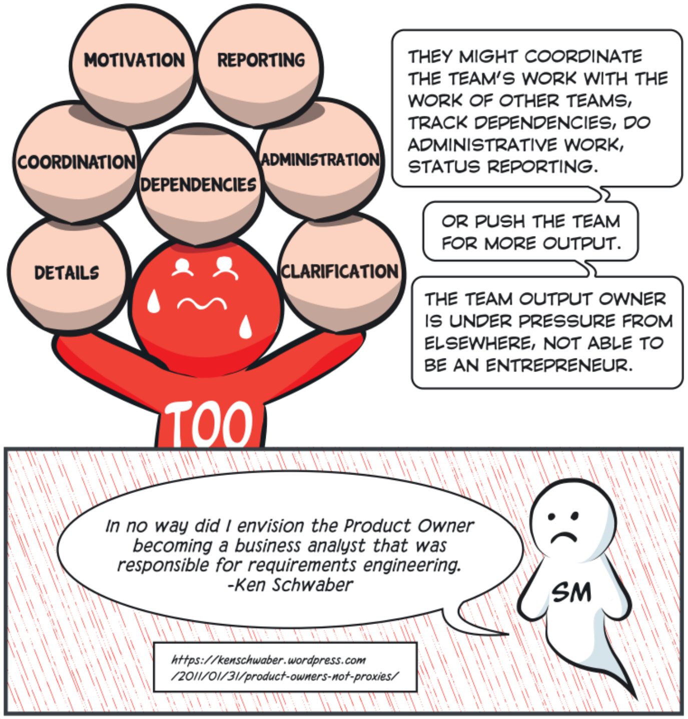


The Product Owner Committee

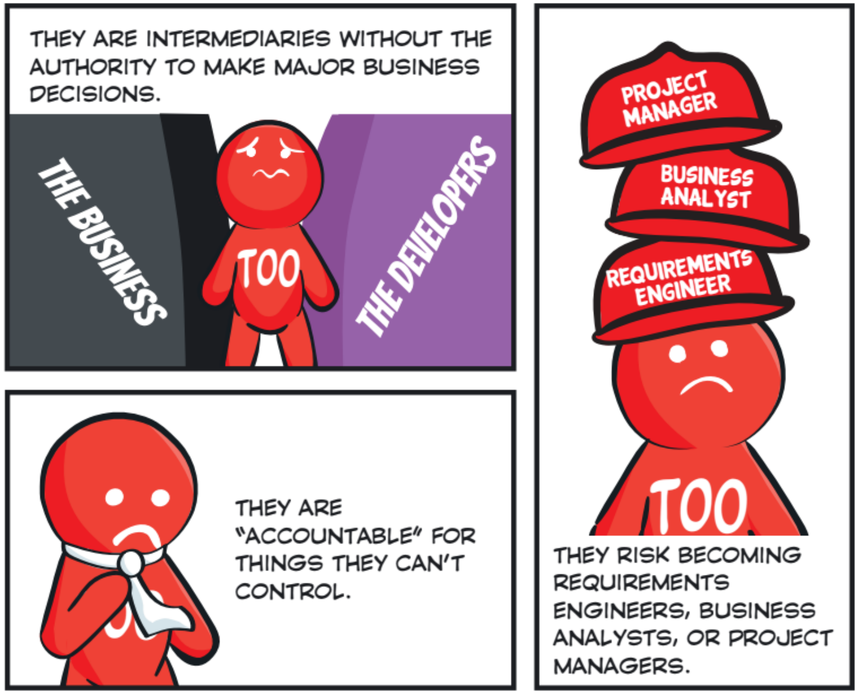
A product owner committee is a group of product owners without anyone in charge of the overall product. There is no one person guiding the group, helping to create a common goal, and facilitating decision making. A product owner committee is in danger of getting caught in endless meetings with conflicting interests and politics—something also referred to as “death by committee.” No real progress is achieved; people stop collaborating and start fighting each other. Always ensure

https://www.amazon.com/Agile-Product-Management-Scrum-Addison-Wesley/dp/0321605780/ref=sr_1_1?crid=9G3F88YRVFO4&dchild=1&keywords=agile+product+management+with+scrum&qid=1631966761&sprefix=agile+prod%2Caps%2C156&sr=8-1

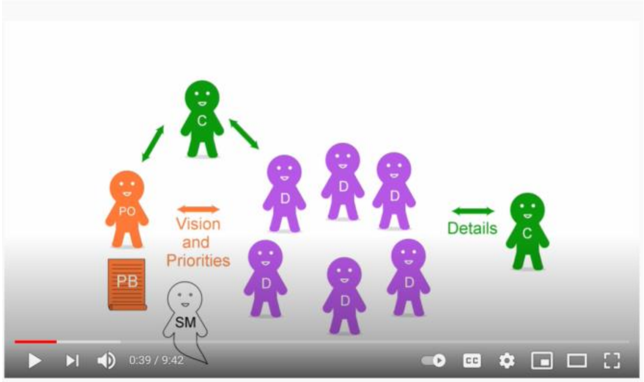
Scrum Anti-patterns


MOTIVATION
REPORTING
COORDINATION
ADMINISTRATION
DEPENDENCIES
DETAILS
CLARIFICATION
TOO
THEY MIGHT COORDINATE THE TEAM'S WORK WITH THE WORK OF OTHER TEAMS, TRACK DEPENDENCIES, DO ADMINISTRATIVE WORK, STATUS REPORTING.
OR PUSH THE TEAM FOR MORE OUTPUT.
THE TEAM OUTPUT OWNER IS UNDER PRESSURE FROM ELSEWHERE, NOT ABLE TO BE AN ENTREPRENEUR.
In no way did I envision the Product Owner becoming a business analyst that was responsible for requirements engineering. -Ken Schwaber
SM
https://kenschwaber.wordpress.com /2011/01/31/product-owners-not-proxies/

19 (C) 2019 Michael James (mj). All rights reserved.


THEY ARE INTERMEDIARIES WITHOUT THE AUTHORITY TO MAKE MAJOR BUSINESS DECISIONS.
THE BUSINESS
TOO
THE DEVELOPERS
PROJECT MANAGER
BUSINESS ANALYST
REQUIREMENTS ENGINEER
TOO
THEY ARE "ACCOUNTABLE" FOR THINGS THEY CAN'T CONTROL.
THEY RISK BECOMING REQUIREMENTS ENGINEERS, BUSINESS ANALYSTS, OR PROJECT MANAGERS.

(C) 2019 Michael James (mj). All rights reserved.



Why "Scrum" Isn't Making Your Organization Agile: Harmful Misconceptions About Product Owner Role

https://www.youtube.com/watch?v=cr2rjaGmUzo&t=22s

AMAZING SCRUM
Volume 1 Episode 1

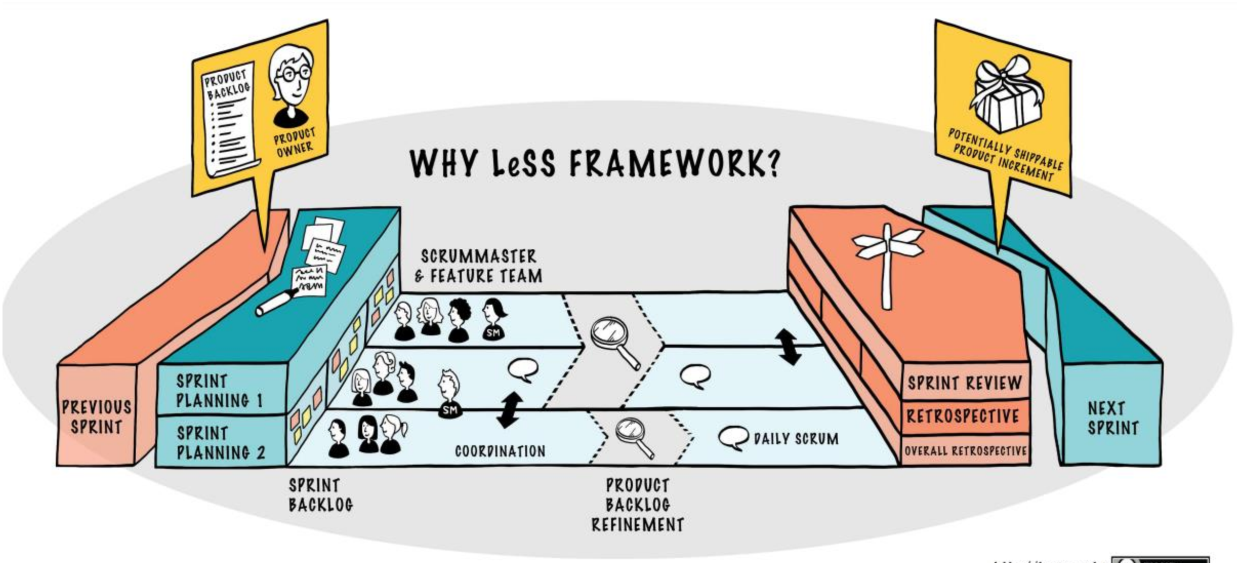
WHY "SCRUM" ISN'T MAKING YOUR COMPANY VERY AGILE

HOW MISCONCEPTIONS ABOUT THE PRODUCT OWNER ROLE HARM YOUR ORGANIZATION. AND WHAT TO DO ABOUT IT.

BY MJ

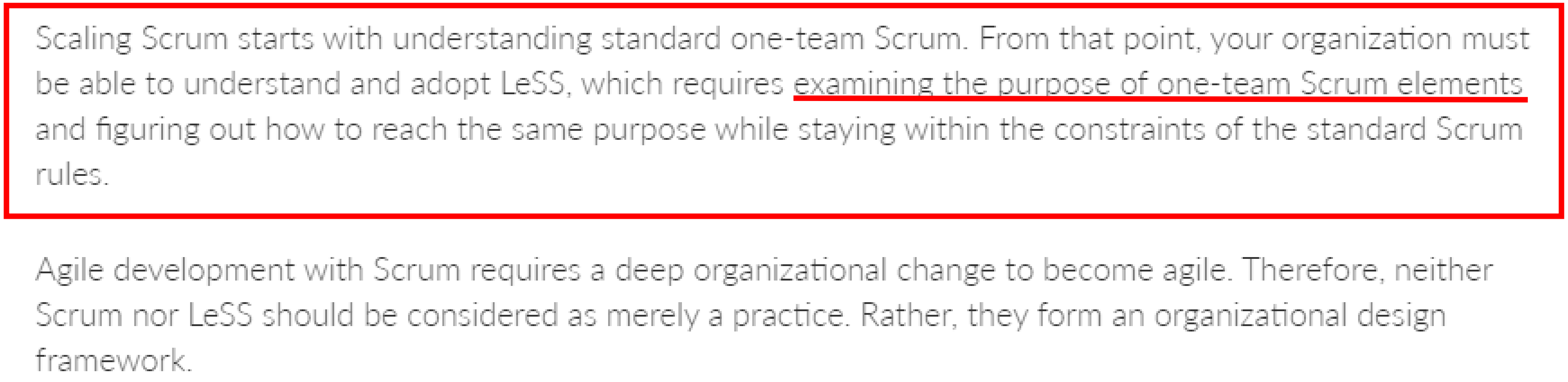
https://seattlescrum.com/Why-Scrum-Isnt-Making-Your-Company-Very-Agile/

Large Scale Scrum (LeSS)



http://less.works (cc) BY-ND

https://less.works/img/framework/why-less-framework.pdf



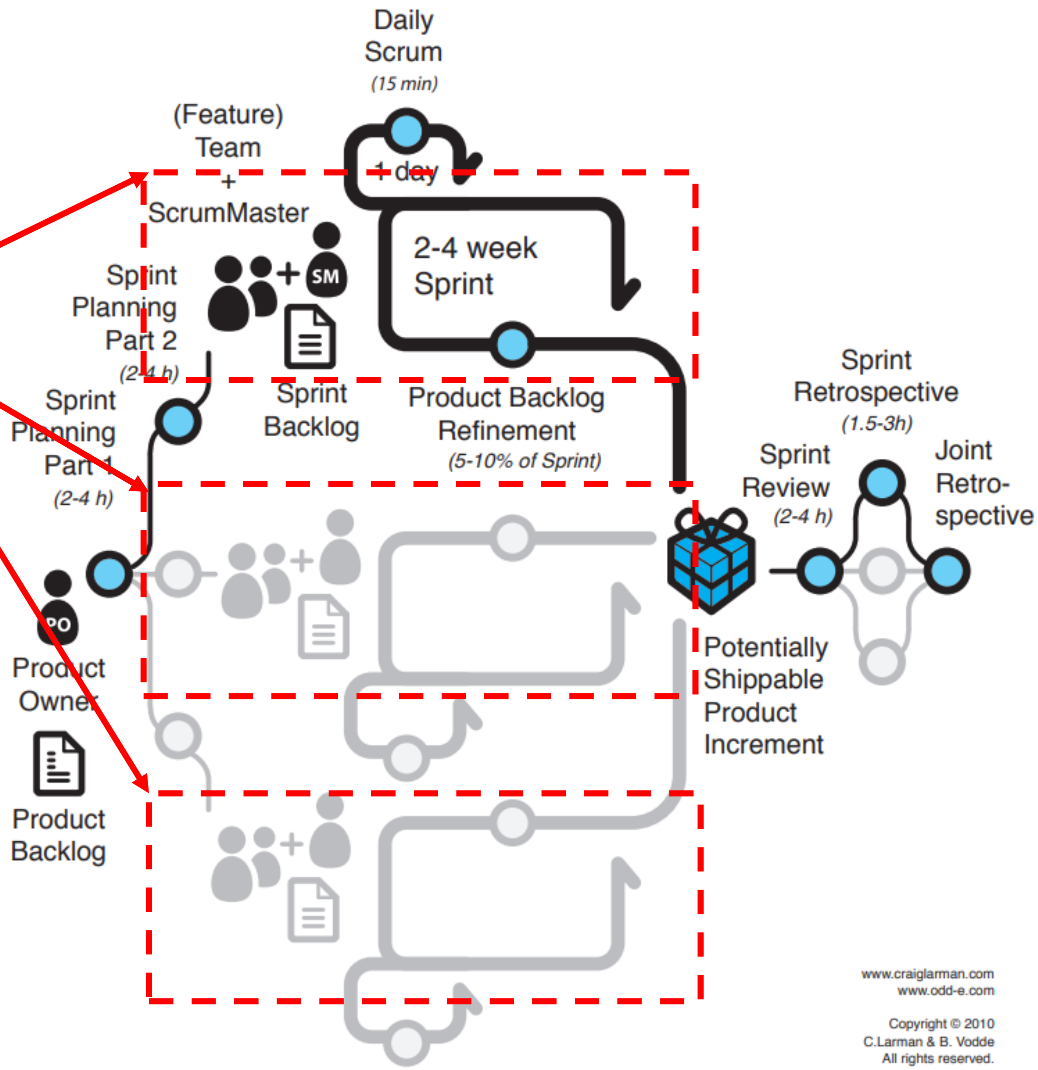
Scaling Scrum starts with understanding standard one-team Scrum. From that point, your organization must be able to understand and adopt LeSS, which requires examining the purpose of one-team Scrum elements and figuring out how to reach the same purpose while staying within the constraints of the standard Scrum rules.

Agile development with Scrum requires a deep organizational change to become agile. Therefore, neither Scrum nor LeSS should be considered as merely a practice. Rather, they form an organizational design framework.

Large Scale Scrum (LeSS)

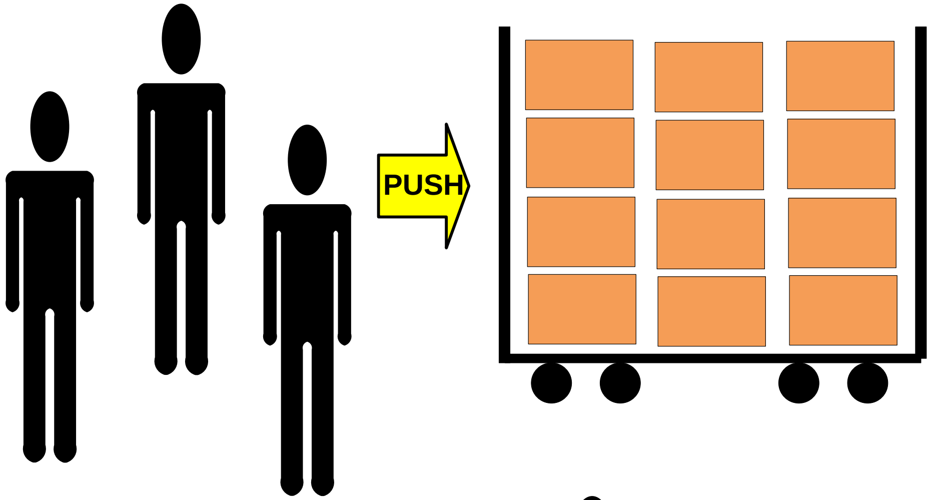
Team = organizational building block

LeSS Product Group = organization (parallel)

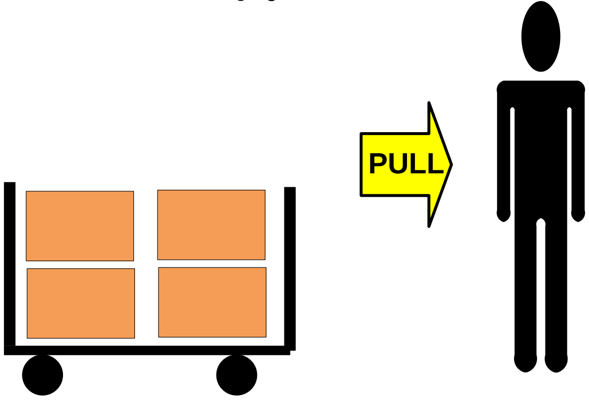


https://less.works/scaling-book-images/practices-scaling-lean-agile-development/chapter-2-large-scale-scrum-framework1-en.pdf

Kanban



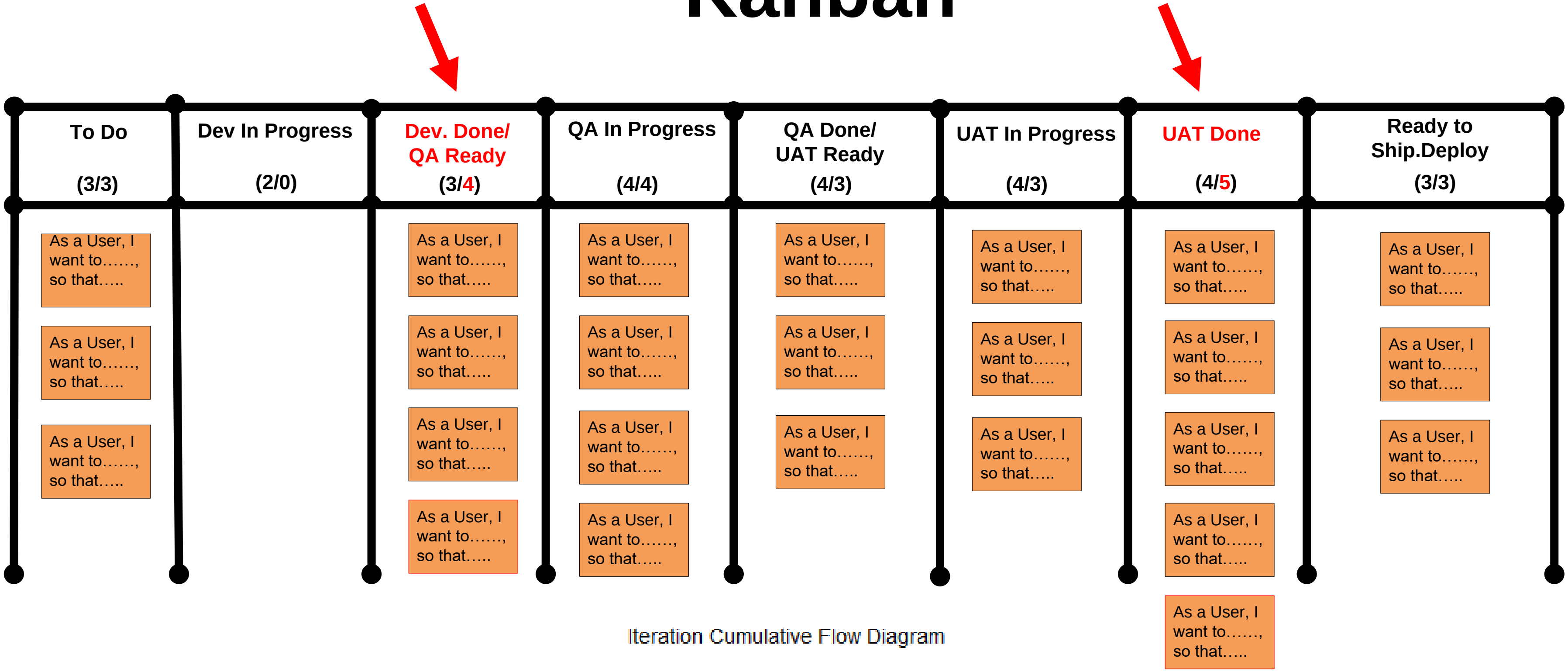
Anticipated Demand
Approximated Usage
High Quantities
Voluminous Storage/Inventory
High Waste of resources (time & money)
Heavy Weight Shipments
Longer turn around
Bottlenecks & Bubbles (Vacuum)



Driven by precise Demand
Smaller, more digestible quantities
Low Waste of resources (time & money)
Light Weight Shipments
Cost effectiveness
Quicker turn around
Smoother Flow

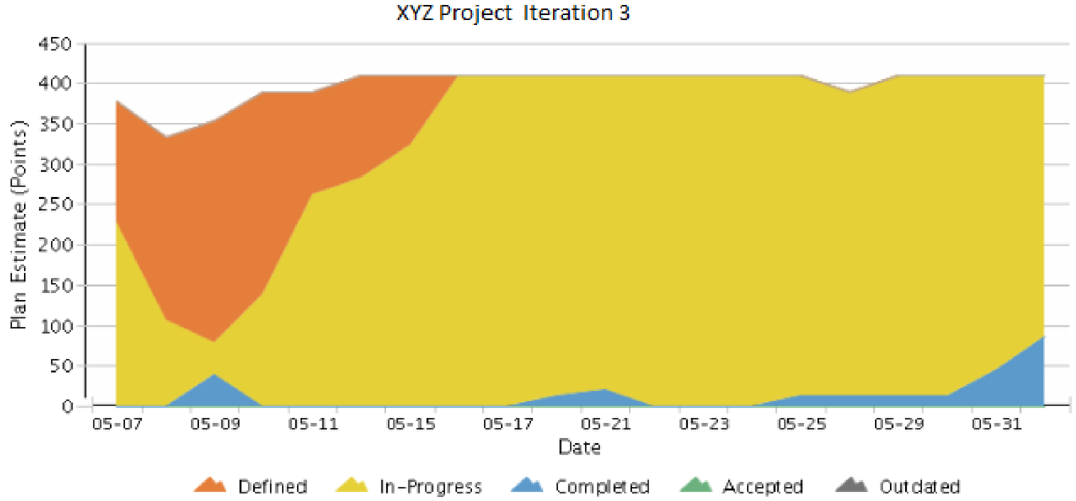
Source: Gene Gendel’s Kanban training materials

Kanban



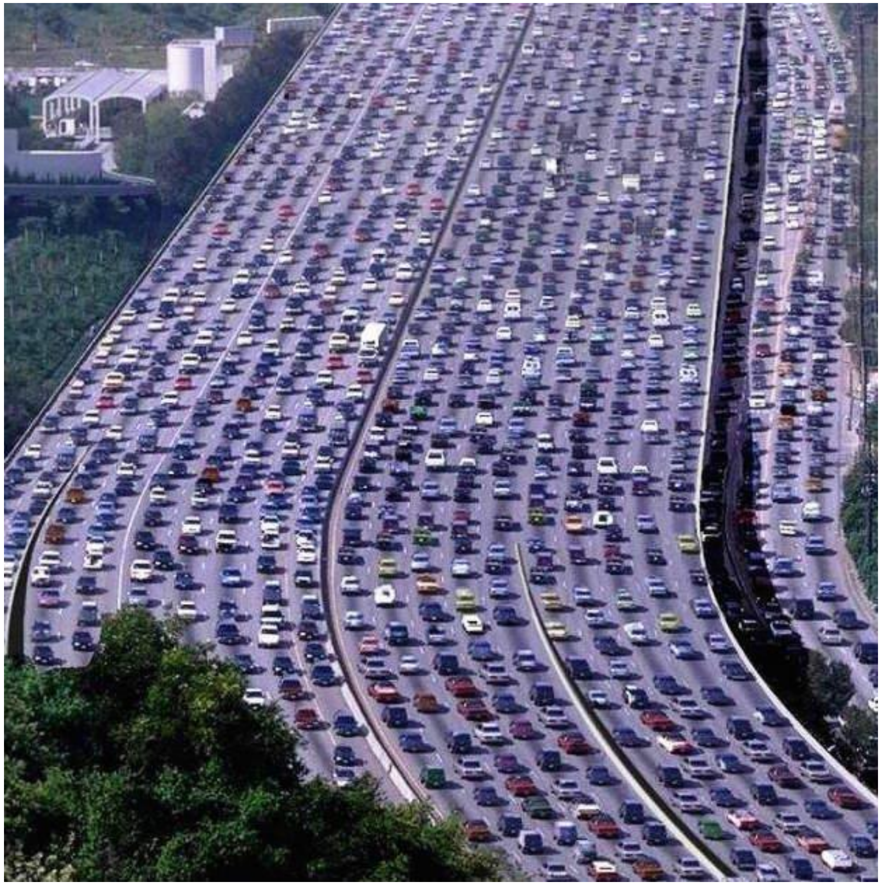
To Do (3/3)	Dev In Progress (2/0)	Dev. Done/ QA Ready (3/4)	QA In Progress (4/4)	QA Done/ UAT Ready (4/3)	UAT In Progress (4/3)	UAT Done (4/5)	Ready to Ship.Deploy (3/3)
As a User, I want to……, so that…..		As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..
As a User, I want to……, so that…..		As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..
As a User, I want to……, so that…..		As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..	As a User, I want to……, so that…..
		As a User, I want to……, so that…..	As a User, I want to……, so that…..			As a User, I want to……, so that…..	
						As a User, I want to……, so that…..	

Iteration Cumulative Flow Diagram

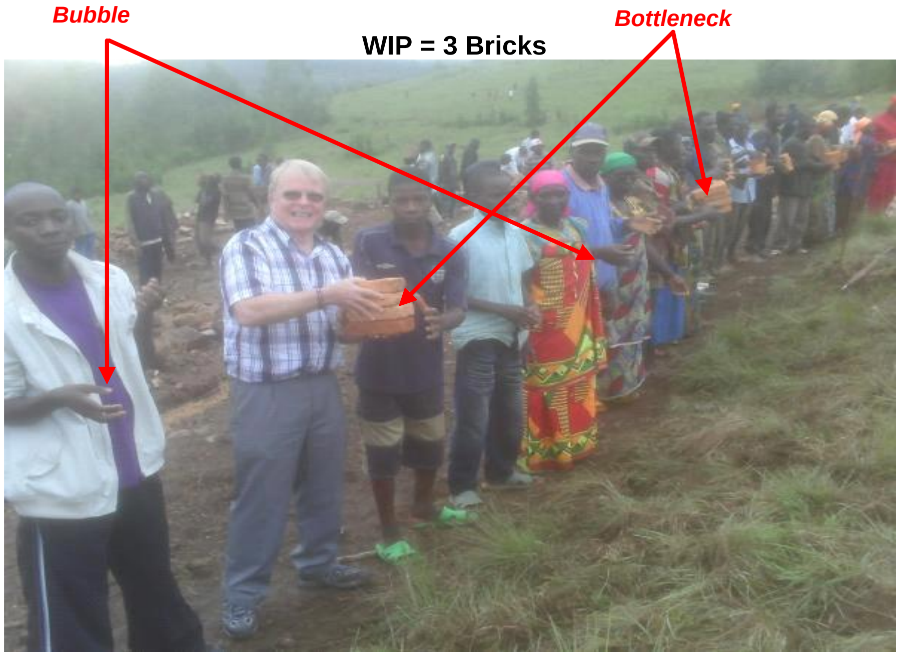


Source: Gene Gendel's Kanban training materials

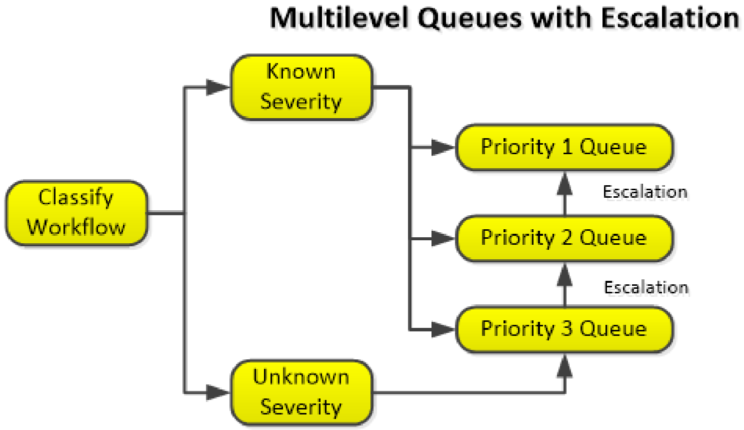
Kanban



https://less.works/img/queueing_theory/



Courtesy of: http://leecadden.wordpress.com/2010/06/page/2/



Multilevel Queues with Escalation

Classify Workflow

Known Severity

Unknown Severity

Priority 1 Queue

Priority 2 Queue

Priority 3 Queue

Escalation

Escalation

Source: Gene Gendel's Kanban training materials

Workflow Management

Lean Product Development

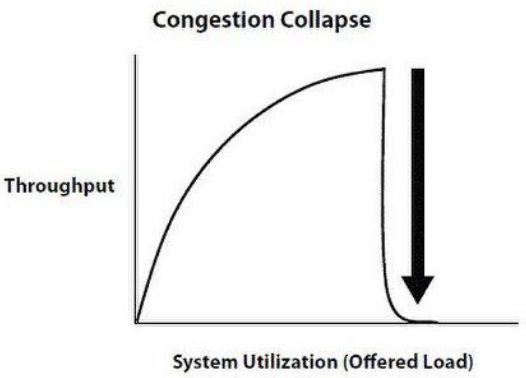
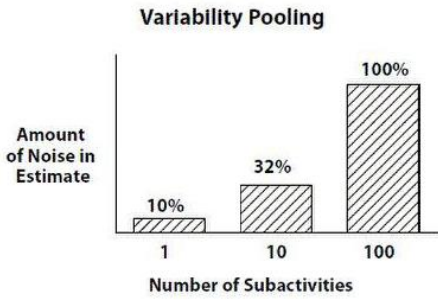
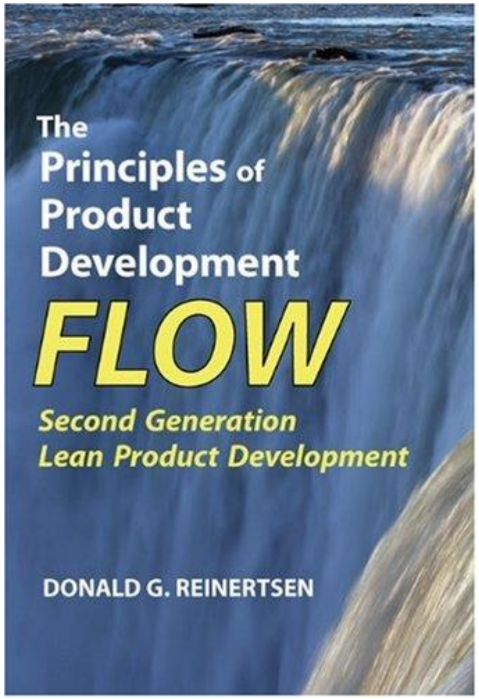


Figure 7-4 Congestion collapse is an abrupt drop in throughput at high levels of system utilization. It results in an undesirable state of high utilization and low throughput.



$$\sigma_T = \sqrt{\sigma_1^2 + \sigma_2^2 + \sigma_3^2 ..... + \sigma_n^2}$$

Figure 4-7 When we combine activities with uncorrelated variability, the total standard deviation grows as a square root function. The more we subdivide activities, the higher the coefficient of variation in each subactivity. We can reduce variability by pooling activities together.


The Principles of Product Development
FLOW
Second Generation Lean Product Development
DONALD G. REINERTSEN

https://www.amazon.com/Principles-Product-Development-Flow-Generation/dp/1935401009/ref=sr_1_1?dchild=1&keywords=the+principles+of+lean+product+development+donald&qid=1631967205&sr=8-1

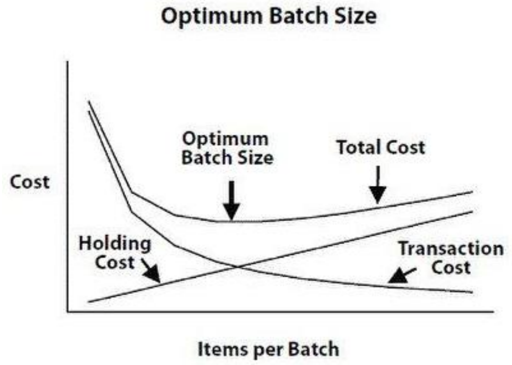


Figure 5-5 Optimum batch size is another U-curve optimization. Higher transaction costs shift optimum batch size higher. Higher holding costs shift it lower.

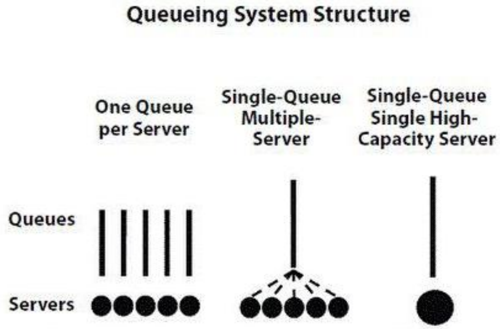
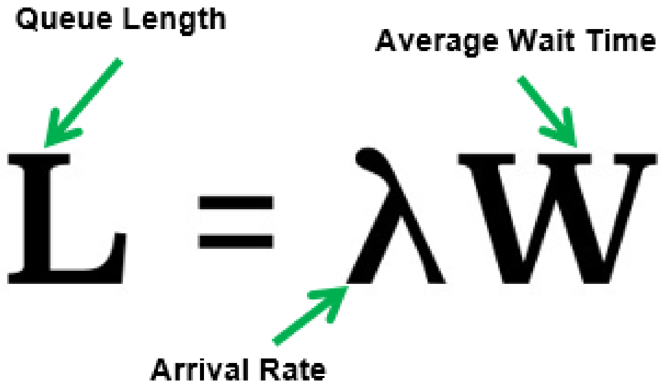
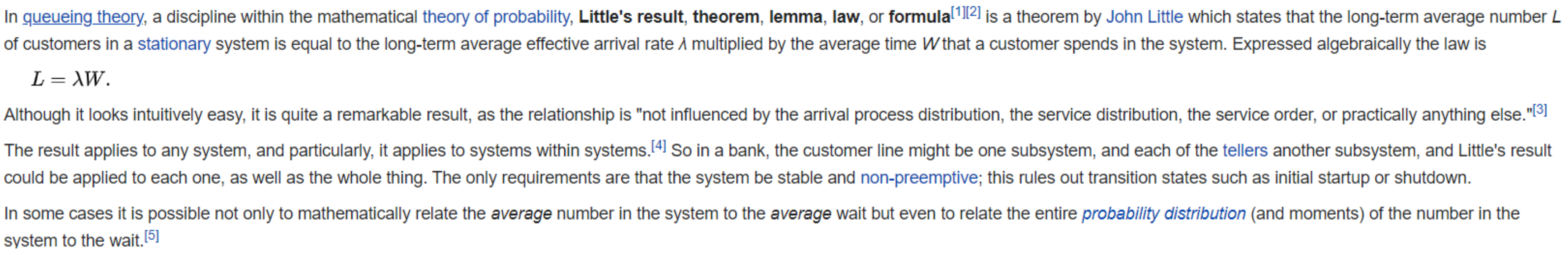

Queueing System Structure
One Queue per Server
Single-Queue Multiple-Server
Single-Queue Single High-Capacity Server
Queues
Servers

Figure 3-7 Single-queue multiple-server systems will have less variation in processing time than systems that have a dedicated server for each queue. The lowest variation occurs with a single high-capacity server, but only when it is very reliable.

Workflow Management

Little’s Law





In queueing theory, a discipline within the mathematical theory of probability, **Little's result**, **theorem**, **lemma**, **law**, or **formula**[1][2] is a theorem by John Little which states that the long-term average number *L* of customers in a stationary system is equal to the long-term average effective arrival rate *λ* multiplied by the average time *W* that a customer spends in the system. Expressed algebraically the law is

$$L = \lambda W.$$

Although it looks intuitively easy, it is quite a remarkable result, as the relationship is "not influenced by the arrival process distribution, the service distribution, the service order, or practically anything else."[3]

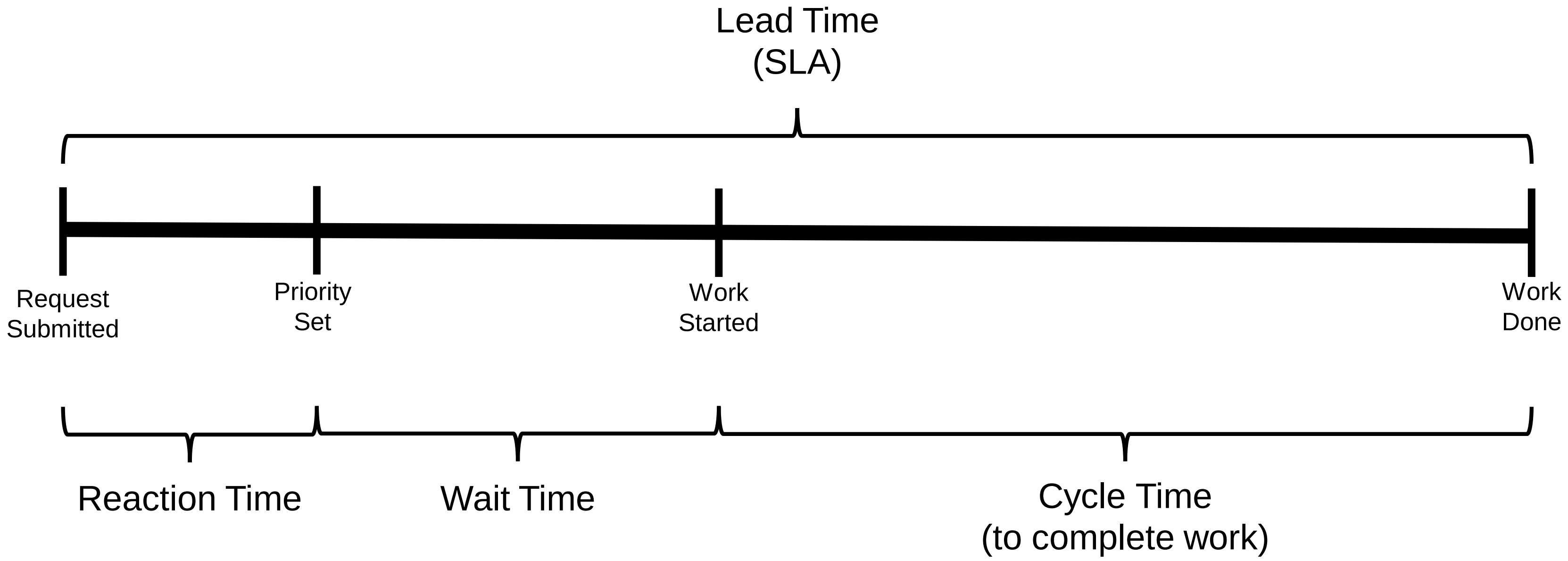
The result applies to any system, and particularly, it applies to systems within systems.[4] So in a bank, the customer line might be one subsystem, and each of the tellers another subsystem, and Little's result could be applied to each one, as well as the whole thing. The only requirements are that the system be stable and non-preemptive; this rules out transition states such as initial startup or shutdown.

In some cases it is possible not only to mathematically relate the *average* number in the system to the *average* wait but even to relate the entire *probability distribution* (and moments) of the number in the system to the wait.[5]

https://en.wikipedia.org/wiki/Little%27s_law

Workflow Management

Time Management



Avoiding The Uneducated Conclusions

> Scrum is better than LeSS

> LeSS is better than Scrum

> Scrum is better than Kanban

> LeSS is better than Kanban

> Kanban is better then Scrum

> Kanban is better than LeSS

> Scrum + Kanban is better than LeSS

> Less + Kanban is better than Scrum

> Scrum + Kanban is better than either one alone

> LeSS + Kanban is better than either one alone

10 Not-So-Good Reasons To Use Kanban

1. *“Jira mechanics to manage a scrum sprint (open/close), are too confusing. Kanban seems to be easier.”*
2. *“Our enterprise Jira set up for Scrum (e.g. issue schema) is centrally controlled and cannot be simplified by teams. Therefore, we will just use Kanban, instead.”*
3. *“Our Product Owner does not attend Scrum events. Therefore, we cannot officially call this Scrum. So, lets do Kanban, as it does not require having a PO”*
4. *‘The same reasoning, as above, but with respect to: Scrum Master”*
5. *“We cannot protect our sprint scope. We cannot deliver PSPI by sprint-end: nothing shippable to review. In order not to look bad, as Scrum, let’s use Kanban”*
6. *“Our so-called product backlog is full of technical items. It is really a technical backlog. We will tell business people that we will be switching to Kanban, because our work is too technical, so we don’t have to deliver PSPI to them every sprint”*
7. *“Our estimations are not reliable. We cannot maintain a steady Velocity. Lets use Kanban. There is Throughput, not Velocity - and nobody understands it ☺.”*
8. *“Kanban is purely an IT thing☹. Since this is how our agile transformation is handled (IT thing-only) anyway, lets use Kanban, instead of Scrum”*
9. *”Kanban does not explicitly require full-stack developers, so we don’t need to worry about nudging/encouraging/incentivizing developers to learn complimentary skills”*
10. *”Kanban is not focused on, or challenge organizational design, like Scrum (or LeSS) do, so lets use Kanban, since our managers are reluctant to support real changes”*

10 Better Reasons To Use Kanban

1. *”Our work is mainly customer service and back-end production support, with little dependency on prioritization, coming directly from Product Owner”*
2. *”We don’t need dedicated events, for clarifications. Therefore, we can schedule refinement/discussion events on-demand/short notice (no need for an official event)”*
3. *”Our Product Owner/users/customers continuously review and accept our work, as it gradually flows from TO-DO to Done. Sprint reviews will be, somewhere, redundant”*
4. *”Our average SLA is too short, to deal with overhead of sprint events (even small)”*
5. *”We are able to deliver to production frequently (e.g. full adoption of CI/CD), response cycle time is too short. Thus, waiting for a review is less critical for us”*
6. *”Our work items are so consistent in size/complexity that using velocity to relatively estimate them does not add much value (e.g. all items are 5 story points). Therefore, we use Kanban’s Throughput, instead of Scrum’s Velocity”*
7. *”The intake of new work items is so steady/gradual and their priority is so consistent, that there is no backlog build-up. Pulling of work from TO-DO queue is very natural, and is in balance with influx of work from upstream domains, into TO-DO queue”*
8. *”We have multiple streams of work, constantly converging and diverging, with multiple queues and WIP limits. Kanban - is a better way to manage our workflow”*
9. *”Our work is tightly coupled to third-party work and they already use Kanban efficiently, so we want to be consistent with their Takt time. Thus, we use Kanban.”*
10. *”Full spectrum of work is too wide. Building C/F teams is too challenging for now. Having ‘queue-specific’ workers with upstream & downstream skills is more realistic”*

10 Ways To Use Kanban With Scrum or LeSS

1. *Ensuring that a PULL (not PUSH) system is used by Scrum or LeSS team(s)*
2. *Putting WIP limits on Sprint queues, AND calling out violations*
3. *Ensuring one-piece workflow (high degree of swarming) is maintained*
4. *Moving away from story points b/c estimation is consistent (size/complexity)*
5. *Measuring Reaction Time, for each PBI that is entered in BL, before priority is set*
6. *Measuring Wait Time, for each prioritized PBI, before it gets planned into a sprint*
7. *Measuring Cycle Time for each PBI, while it is in WIP (inside a sprint)*
8. *Measuring Lead Time for each PBI (from time PBI gets created to time it is fully Done, including UNDONE work!!!)*
9. *Ensuring that Batch Size is optimized, assuming Transaction Costs, Holding Costs, automation capabilities (as they decrease Transaction Costs)*
10. *Measuring flow of work through queues of different teams (in LeSS), to identify/improve unevenness of flow and opportunities for improvements (e.g. mentoring, developers-travelers, dojos)*

- Be a smart cook: add ingredients to your meal thoughtfully
- Don’t’ overmix: not all ingredients mix well together
- Don’t over-spice: you will kill the original taste

….for using small fonts on the last three pages of this deck…

Q & A