

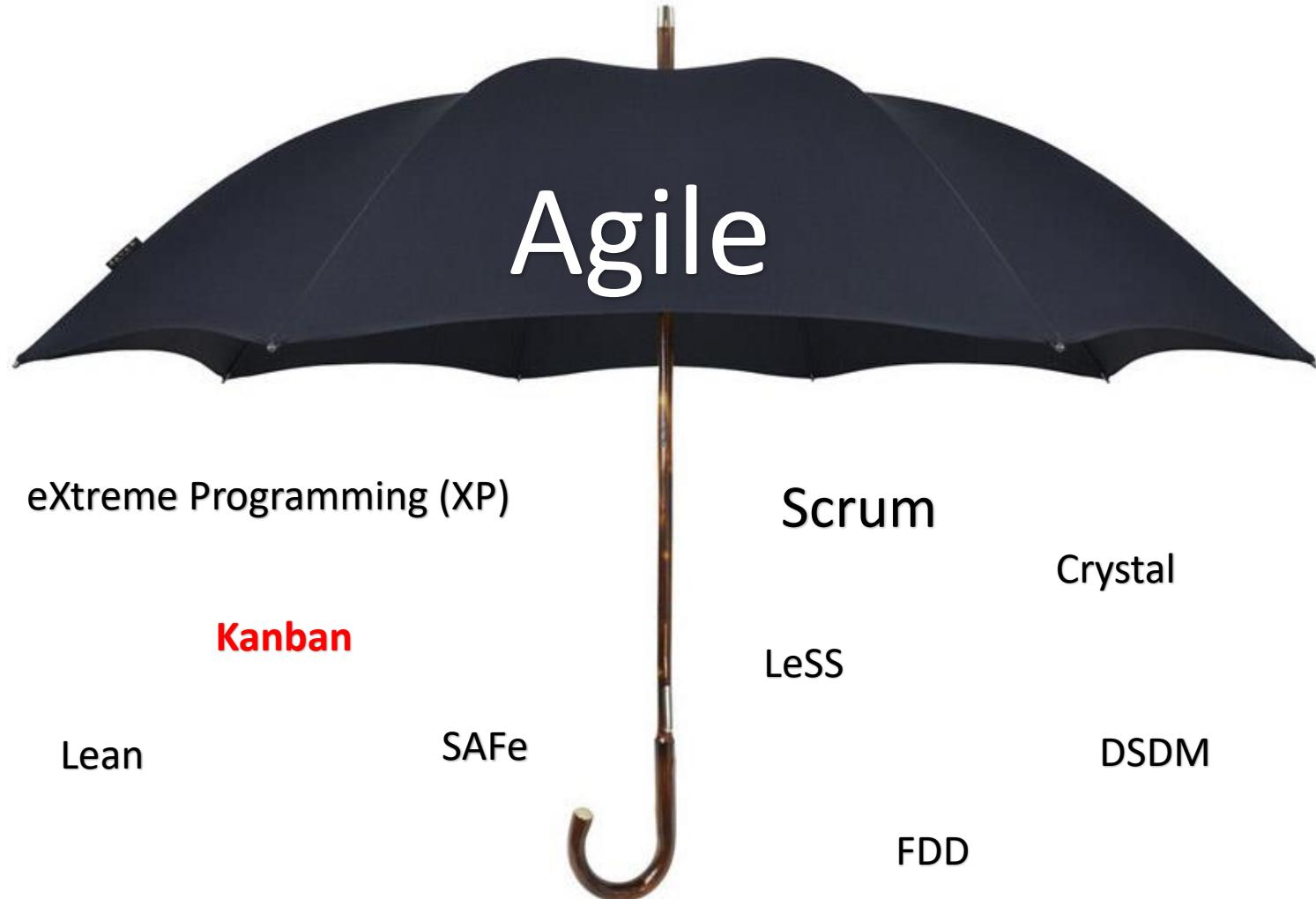
322 371 Software Engineering

Chapter 04: Agile Software Development - Kanban

By

Chitsutha Soomlek, Ph.D.

Department of Computer Science
Faculty of Science, KKU



Kanban

- かんばん (看板) = visual sign, card, or dashboard
- Kanban adopts **lean manufacturing** and **Toyota's "just-in-time" (JIT) production system**
- A software development process can be thought of as a pipeline with feature requests entering one end and improved software emerging from the other end
- Kanban limits the amount of **work-in-progress (WIP)** to prevent overproduction and bottleneck

Lean Thinking

- The ultimate goal is to provide **perfect value** to the customer through a **perfect value creation process that has zero waste.** ผลิตตามที่ลูกค้าต้องการ ไม่ต้องผลิตเพื่อ
- Maximize customer value while minimizing waste
- Create more value for customers with fewer resources
- Respond to changing customer desires with high variety, high quality, low cost, and **with very fast throughput times**
- Promote the culture of **continuous improvement**

Lean Thinking

ลดขั้นตอนที่ไม่จำเป็นลง เช่น กิจกรรมที่ต้องรอกัน ต้องลดการรอ

- Optimize the flow of products and services through entire value streams that flow horizontally across technologies, assets, and departments to customers
สิ่งที่เราจะได้คือ Process ที่ลดการใช้ทรัพยากรลง ลดจำนวนคนลง แต่ได้ประสิทธิภาพเท่าเดิม
- Eliminate waste along entire value streams
ทำลายของเสีย
- Creates processes that need less human effort, less space, less capital, and less time to make products and services at far less costs and with much fewer defects, compared with traditional business systems

7 Wastes (Muda)

- ไม่ต้องทำสำรอง หรือเพื่อไว้ทำแค่ลูกค้าสั่ง
- Over production – safety stock, work-in-process
- Waiting
- Transportation
- Non-value-added processing กระบวนการที่ตัดทิ้งได้
- Excess inventory การเตรียมวัตถุดิบในคลัง มากเกินความจำเป็น
- Defects สินค้าที่ผลิตแล้วมีปัญหา
- Excess motion การเคลื่อนไหวที่มากเกินความจำเป็น

Lean Principles

2/2560

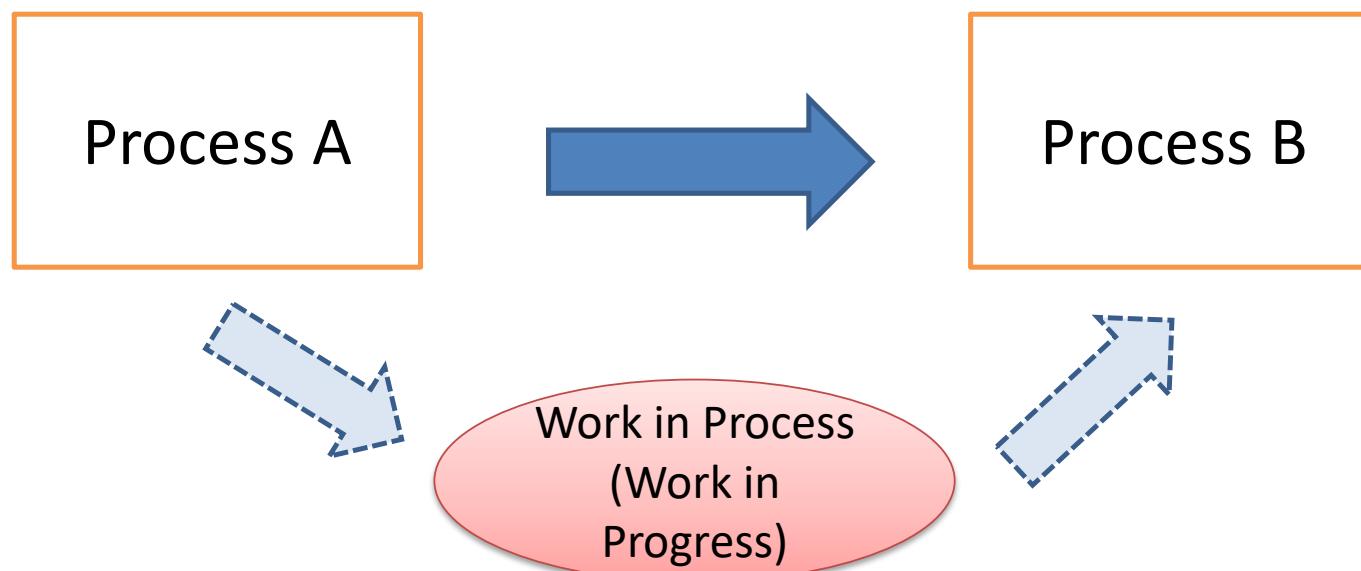


Figure from <https://www.lean.org/WhatsLean/Principles.cfm>

กระบวนการดัน

Pushing System

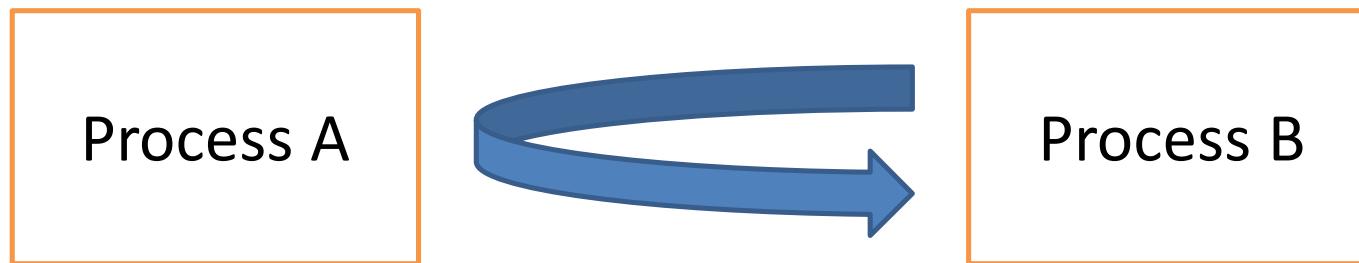
- Producer/Prior process pushes their finished product to the customer/next process.



Pulling System

- The customer/next process pulls what they need from the producer/prior process.

ลักษณะการผลิตที่ Process B จะดึงมาจาก Process A เอง ทำให้ของเสียน้อย
B ดึงมาจาก A แล้วของใหม่จะเติมเข้า A



Just-in-Time (JIT) Production System

- Produce just enough needed items when they are in need
- Pulling system
- Limit work in process

Kanban Principles

ทำ Board ในการแสดง flow การทำงาน

- Start with what you do know
- Agree to pursue evolutionary change
- Initially, respect existing roles, responsibilities and job titles ไม่ต้องปรับเปลี่ยนตำแหน่งการทำงาน (ไม่ต้องเหมือน Scrum)
- Encourage acts of leadership at all levels

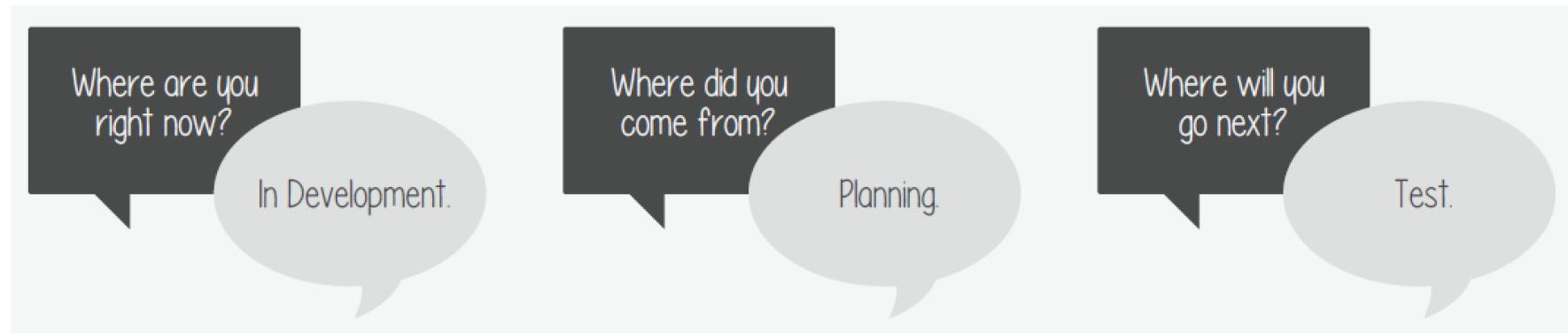
Kanban Practices

- Visualize work
- Focus on Flow
- Limit work-in-progress (WIP)
- Manage flow and make policies explicit
- Continuous Improvement

Visualize Work

- Create a visual model of your work and workflow
- Observe the flow of work moving through your Kanban system
- Make the work, blockers, bottlenecks and queues visible
- Increase communication and collaboration

Building a Kanban Board



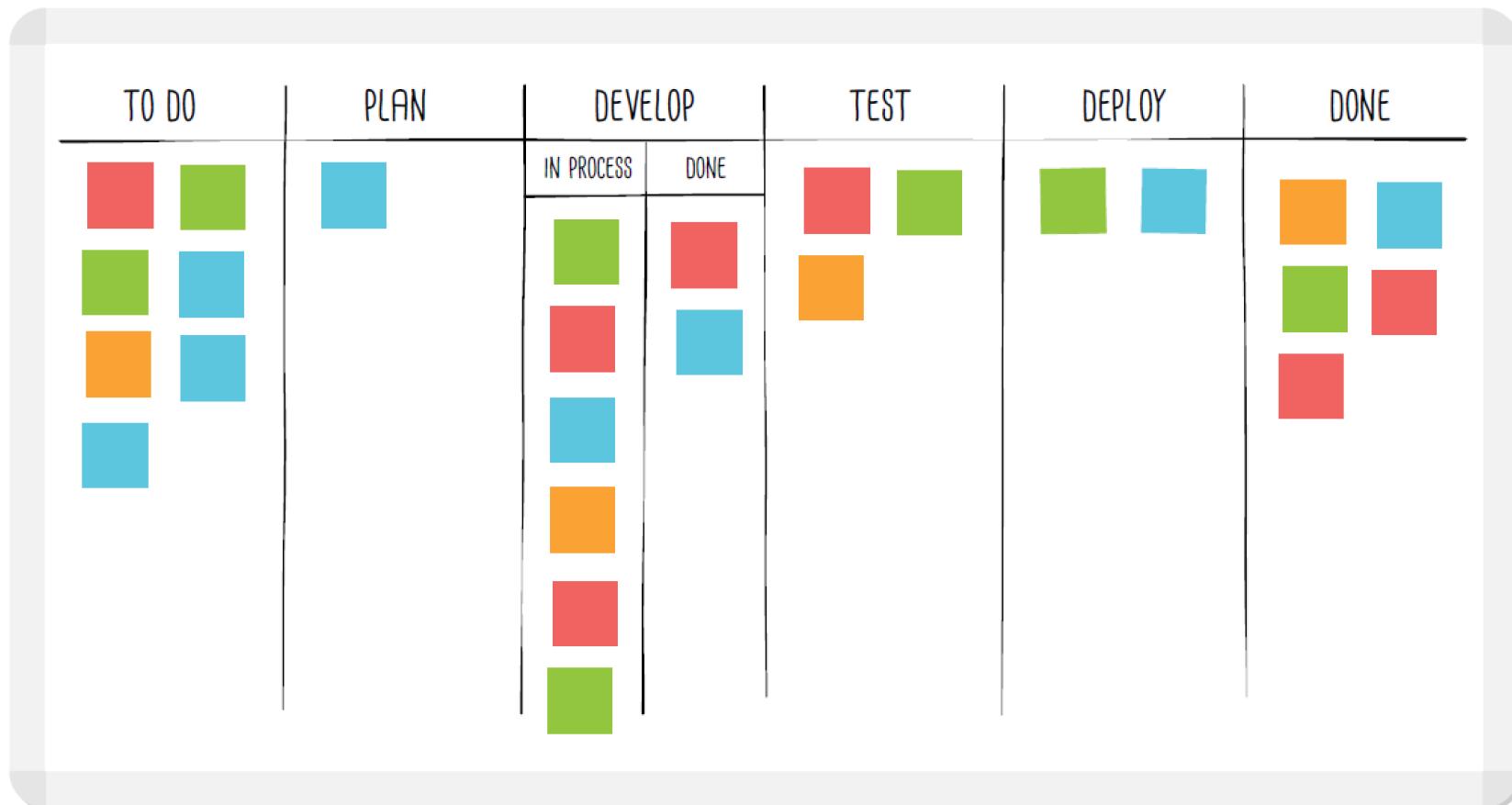
Planning > In Development > Test

Plan > Develop > Test > Deploy > Done

Example of Kanban Boards

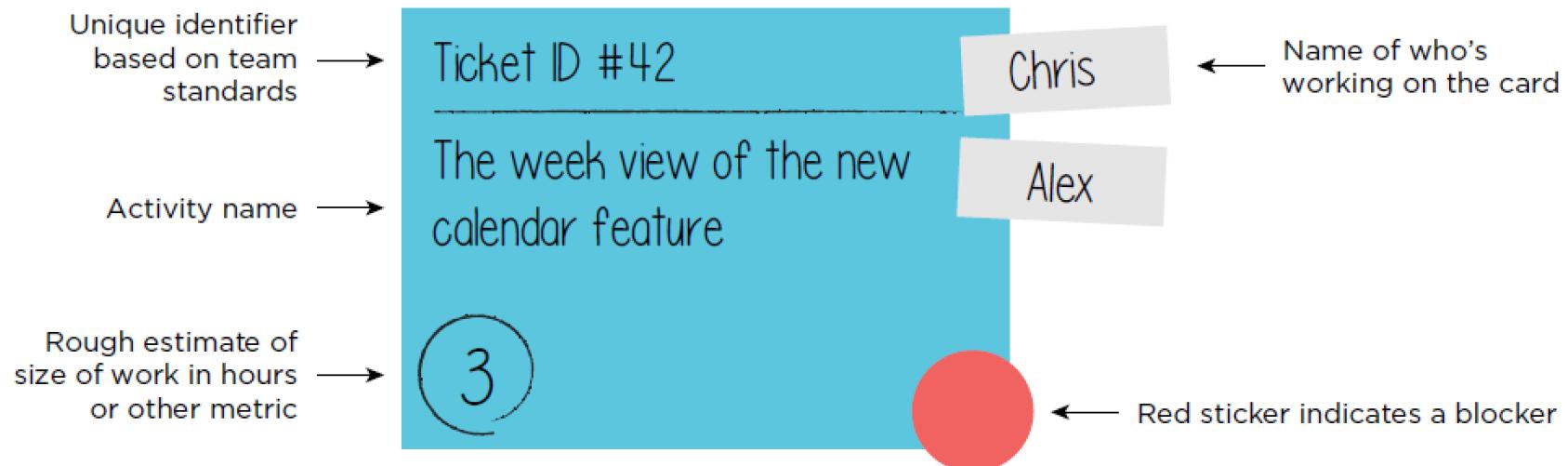


Example of Kanban Boards

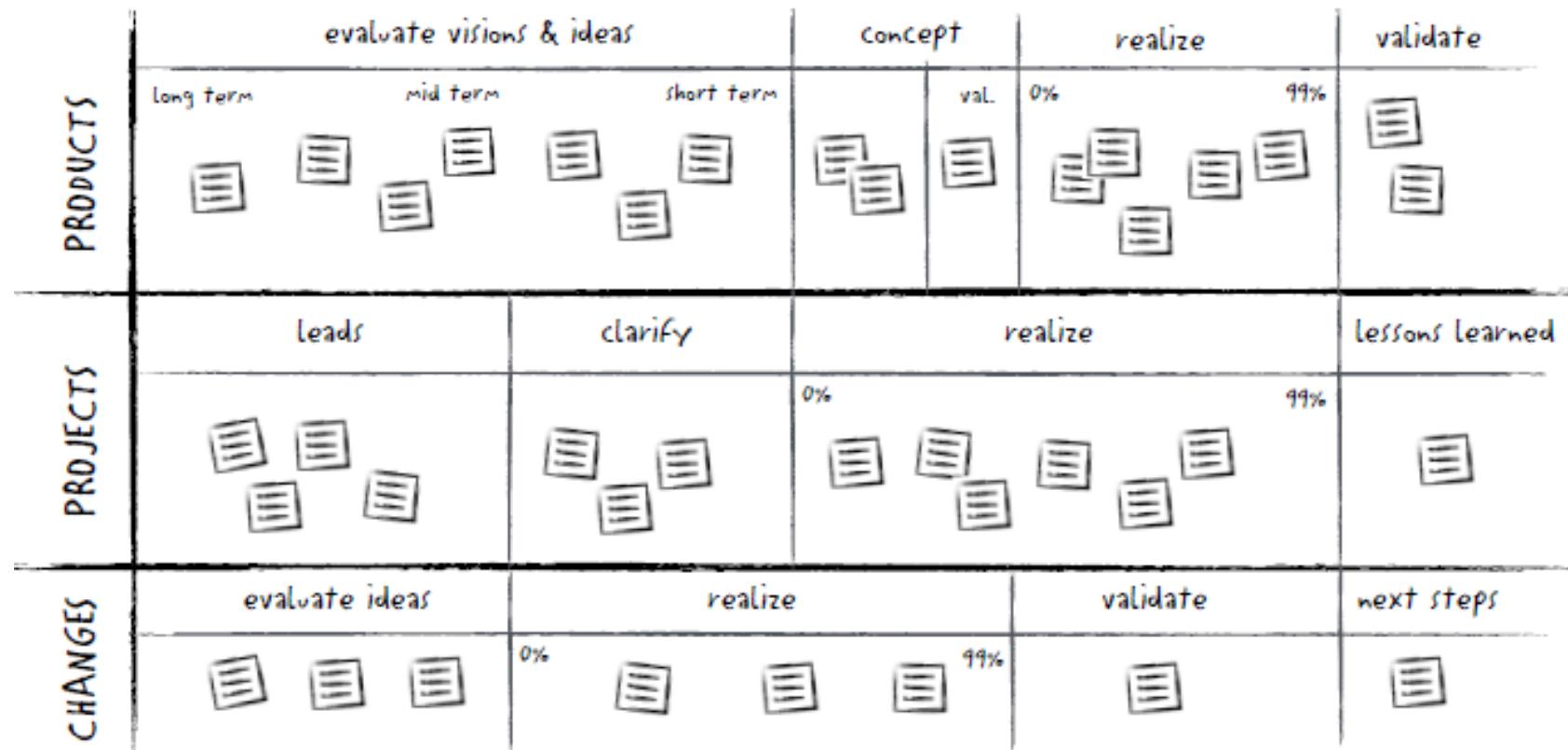


Example of Cards/Work items

ต้องมี ID เพื่อจะบอกว่าเป็นงานชิ้นไหน



Example of Kanban Boards



Focus on Flow

- Think about the entire board as a **single system**
- The board has a **purpose** and a **capacity**, as well as **capabilities** and **interactions**
- Focus on managing the flow of work through the system, rather than directing the work of each team member.
- Work items move across the board, from left to right, as progress occurs
- When a team member is ready to start work on something new, he or she **pulls a new work item into the appropriate lane** on the board.

Focus on Flow

- How does the work flow?
- Where does the work get stuck? ต้องตรวจสอบว่ามีใครที่ไม่ได้ทำงานหรือไม่เขียนขั้นตอนครบหรือไม่
- Is anyone working on anything that's not on the board?
- The best way to start observing the flow of work is with Standup meeting จะถามหรือพูดคุยกันเกี่ยวกับเรื่อง Flow ของงาน
 - routine meetings called **daily standups**
 - weekly **retrospectives**

Limit Work-in-Progress (WIP)

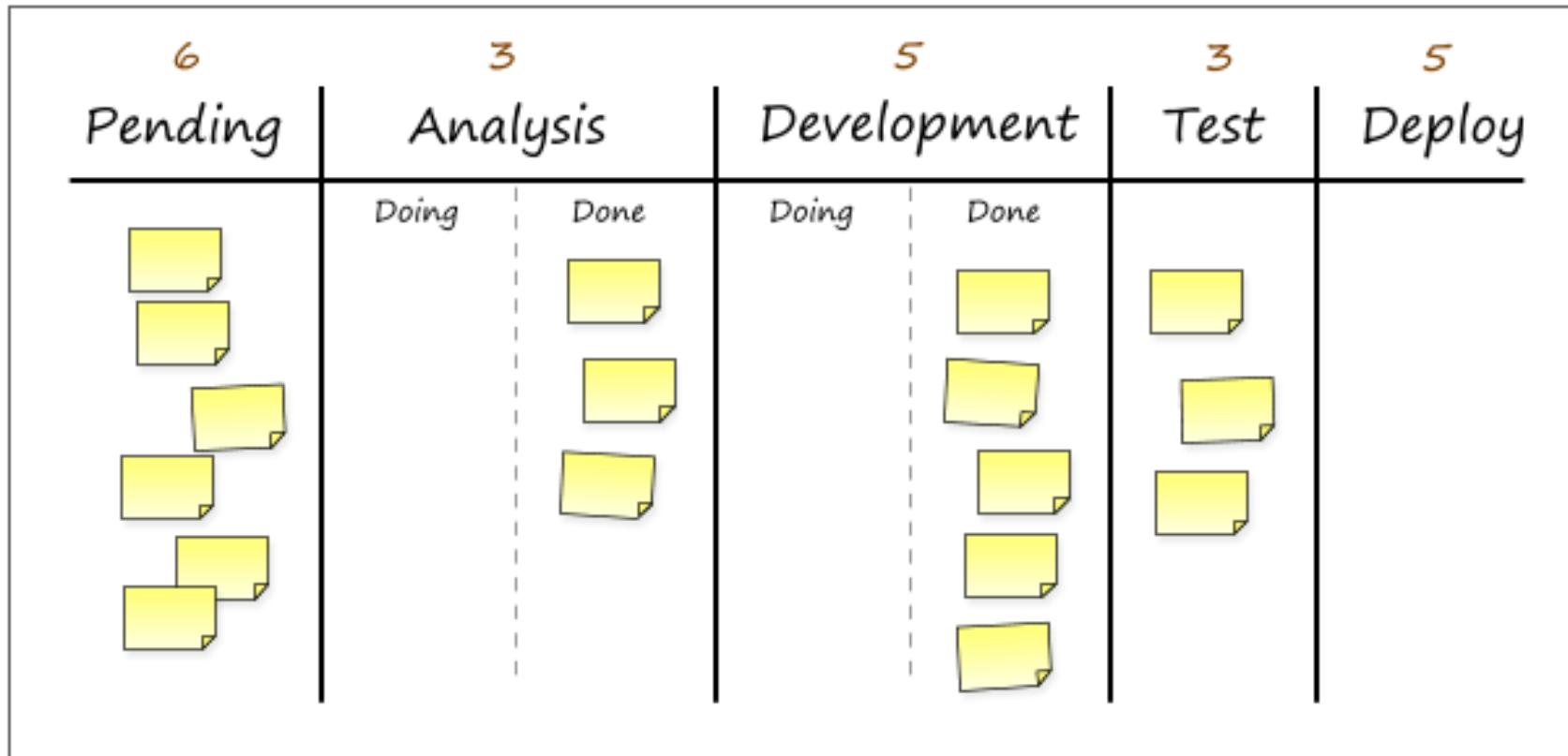
ต้องลิมิตงานที่กำลังทำ ไม่ให้มีงานมาแทรก

- Work in process
- WIP = All of the tasks you're working on right now
- Limiting how much unfinished work is in process
- Reduce the time it takes an item to travel through the Kanban system
- Avoid problems caused by task switching and reduce the need to constantly reprioritize items
- If a team member is consistently responsible for too many work items, experiment with personal WIP limits. *ถ้างานเลขบ่อยๆ ควรมองไปดูว่ามีคนที่ทำงานหนักเกินไหม*

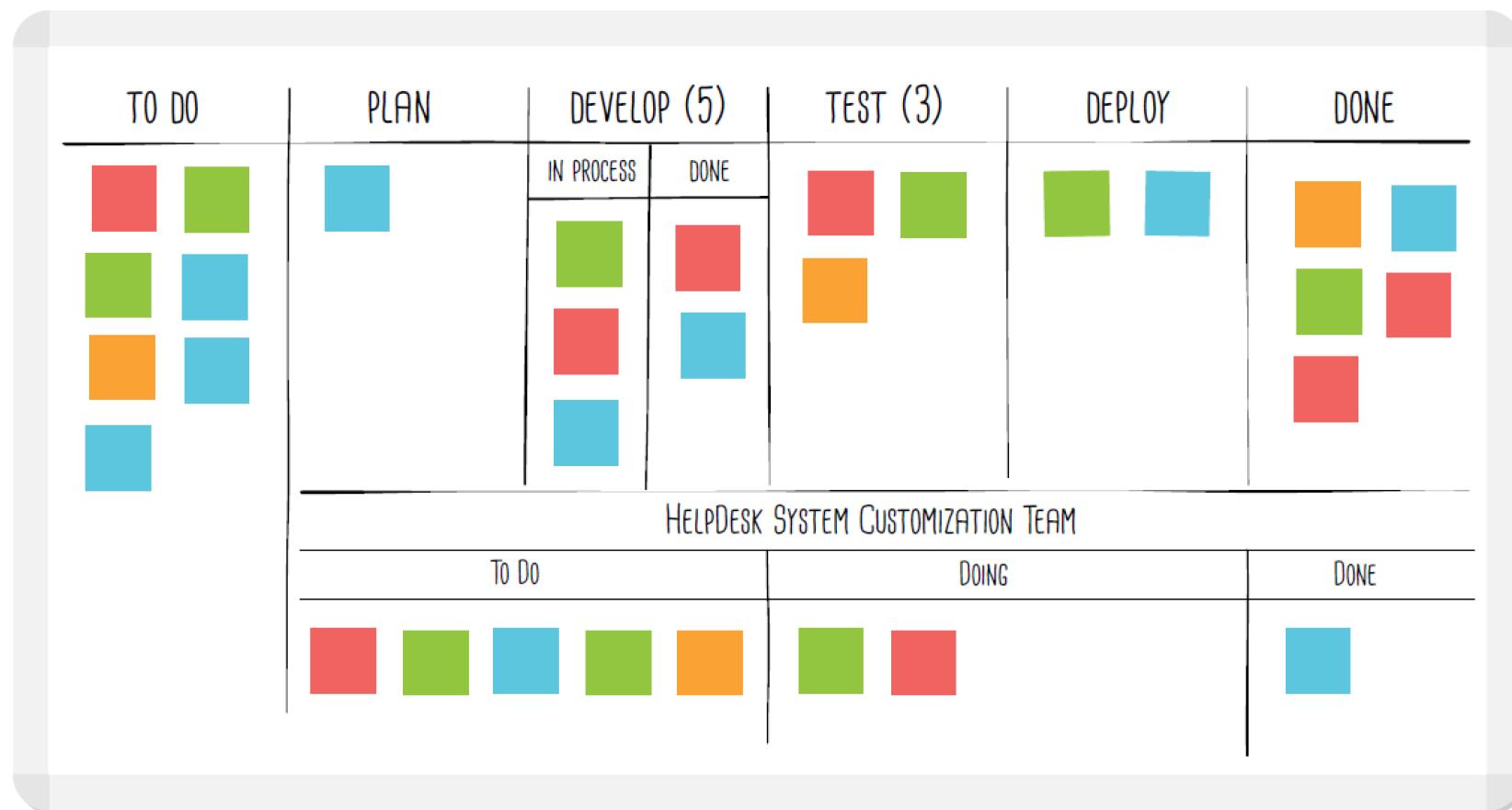
ต้องรอการ
Pull ไม่
สามารถ Push
ไปเองได้

WIP

เกิดการ Block ที่ Test



WIP



Manage Flow and Make Policies Explicit

- Using work-in-process limits and developing **team-driven policies** สร้างนโยบาย
- Optimize your Kanban system to improve the smooth flow of work
ทำการวิเคราะห์ว่า Flow มีประสิทธิภาพหรือไม่ // อาจจะใช้การดูเวลาที่ทำในแต่ละงาน
- Collect **metrics** to analyze flow
- Get leading indicators of future problems by analyzing the flow of work

Continuous Improvement

มีการเก็บค่าสถิติเพื่อที่จะบอกว่าเมื่อเรารับรู้แล้วมันได้ผลหรือไม่ได้ผล

- Implement feedback loop
- คุณภาพของสินค้าที่ออกแบบยังไง
 - Improve collaboratively, evolve experimentally (using models & the scientific method)
 - Teams measure their effectiveness by tracking flow, quality, throughput, lead times and more
 - Experiments and analysis can change the system to improve the team's effectiveness

Continuous Improvement

ปัจจัยที่ดูว่า Flow เริ่มต้นใหม่

- Tools for tracking/measuring flow:
 - Total WIP จำนวนงานค้าง
 - Blockers เรามีการบันลือกงานบ่อยแค่ไหน ที่จุดไหน บ่อยไหม
 - Throughput ผลลัพธ์ของการทำงานเป็นไปเร็ว ช้า
 - Lead time เมื่องานเข้ามาแล้วอยู่ไหนระบบนานแค่ไหน

Total WIP

งานที่ค้างอยู่บน KANBAN
BOARD

- Total WIP = All of the tasks currently on your Kanban board
- It's anything that's been started (by anyone) but not completely finished.

Average WIP per person = Total WIP/No. of team member

- If the result indicates too much work for one person, then reduce WIP.

Blockers

ทำให้เกิดดีเลย์

- A blocked item can't move to the next stage in your process because of an issue.
- Similar to a bottleneck, it creates delays
- A blocker typically signals an unfinished dependency, a defect or an unavailable skillset ปัญหาอาจเกิดจากการแต่งงานที่ไม่ดีพอ
ทำให้มีงานที่มีความสัมพันธ์กันอยู่
- Measuring blockers:

ดึกการบล็อกบ่อยแค่ไหน How often are items blocked?

บล็อกนานแค่ไหน – How long do they stay blocked?

เกิดในกระบวนการไหน – Where in the process do blockers happen?

อาจทำให้เกิดการรอคิวเกิดขึ้น

ผลลัพธ์ที่ออกมาก

Throughput

วัดโดย ทุกวันสุดท้ายของแต่ละสัปดาห์ มีงานที่ done กี่ชิ้น

- Throughput is the number of items completed per time period.
- At the end of each week, record how many items were completed (i.e., moved to “Done” and never moved backwards).
- Track this number from week to week to see how changes made in your Kanban system affect how much total work actually gets done.

Lead Time

มันคือจำนวนเวลาที่ต้องการ
จะทำการบันทึกวันแรกที่ถูกนำเข้าใน Board และบันทึกอีกครั้งในวันที่งาน Done

- Lead time is how long a card takes to travel across the entire board.
- The clock starts when a card is pulled onto the board and stops when it reaches the “Done” lane.
- On the back of each card, record the start date. Then, when the card reaches “Done,” record the date and calculate how many days (or working days) passed since its start date.

Lead Time



Start Date: June 8
Blocked Days: ✗ 2
Blocked location: Testing
Completed Date: June 18
Lead Time: 10 days

- Record the average lead time of every card that was finished that week (i.e., the cards you counted to measure throughput)

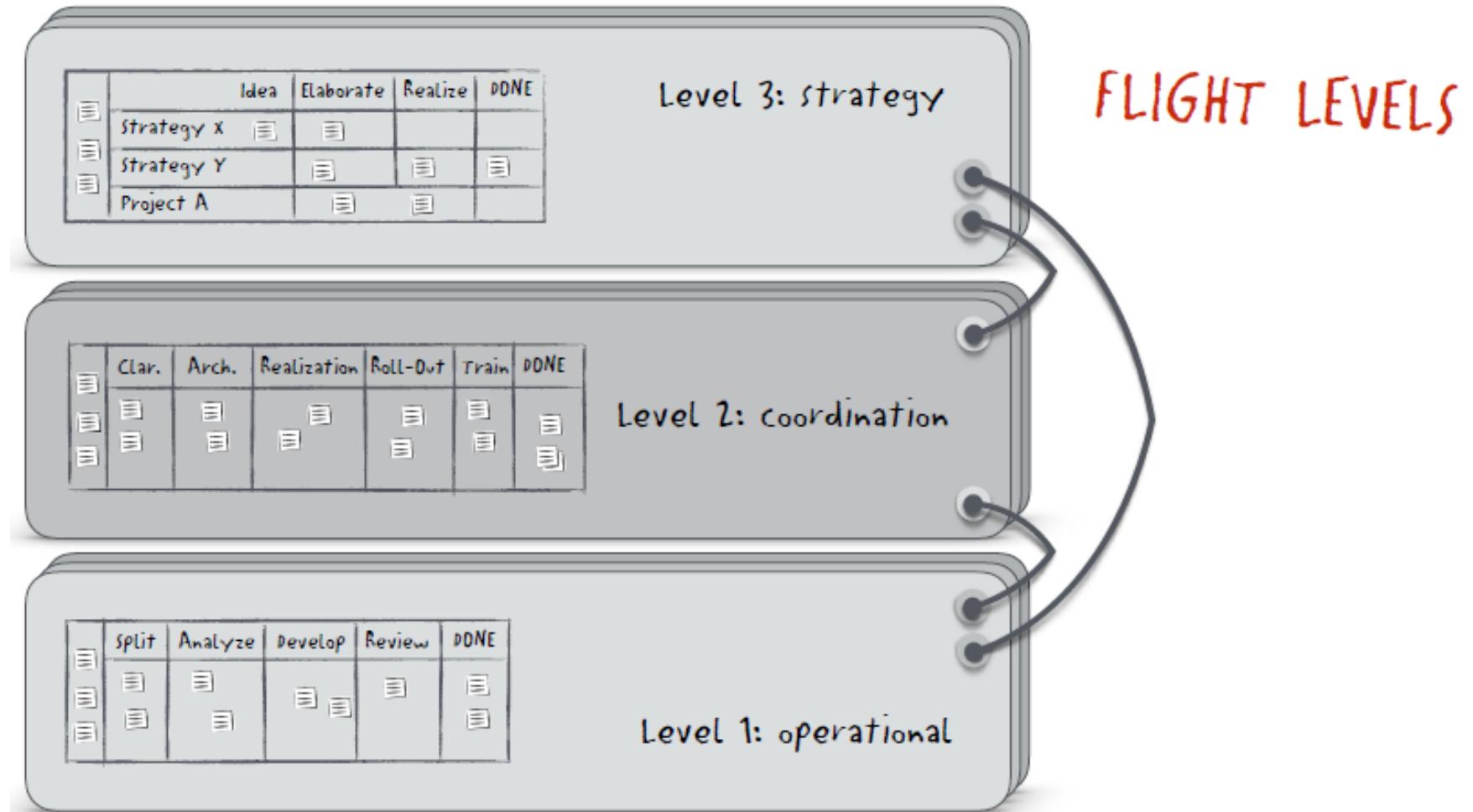
What's next?

กำจัด Blocker เพื่อให้ Lead time ลดลง

- Compare these four metrics to see what effect they have on each other.
- As you reduce the size of queues in your system, it should start to reduce your average lead time.
- As you experiment with ways to manage blockers more effectively, you should also see lead time reduced.

กำหนดขนาด WIP และ เรากำหนด throughput ที่เพิ่มขึ้น

- As you continue to limit your WIP, and work together as a team to complete work collaboratively, you should see throughput improve.



Scaling Agile Methods

- Agile methods have proved to be successful for small and medium sized projects that can be developed by a small co-located team.
- It is sometimes argued that the success of these methods comes because of improved communications which is possible when everyone is working together.
- Scaling up agile methods involves changing these to cope with larger, longer projects where there are multiple development teams, perhaps working in different locations.

Scaling up and Scaling out

- ‘Scaling up’ is concerned with using agile methods for developing large software systems that cannot be developed by a small team.
- ‘Scaling out’ is concerned with how agile methods can be introduced across a large organization with many years of software development experience.
- When scaling agile methods it is essential to maintain agile fundamentals
 - Flexible planning, frequent system releases, continuous integration, test-driven development and good team communications.

Example of Scaling up Agile Methods

- Large Scale Scrum (LeSS)
 - One product backlog
 - Multiple teams (up to about eight teams)
 - One product owner
 - One dedicated Scrum master for 1-2 teams
 - One sprint backlog per team
 - Team representatives work with PO
 - Same sprint kickoff and length
 - Continuous coordination and integration
- See <http://less.works/> for more info.

Scaling out

- Phrases in changing:
 1. **Fight phrase** – Organization will not accept and push back
 2. **Boxing phrase** – People in the organization should move to agile software development but not me
 3. **Changing**