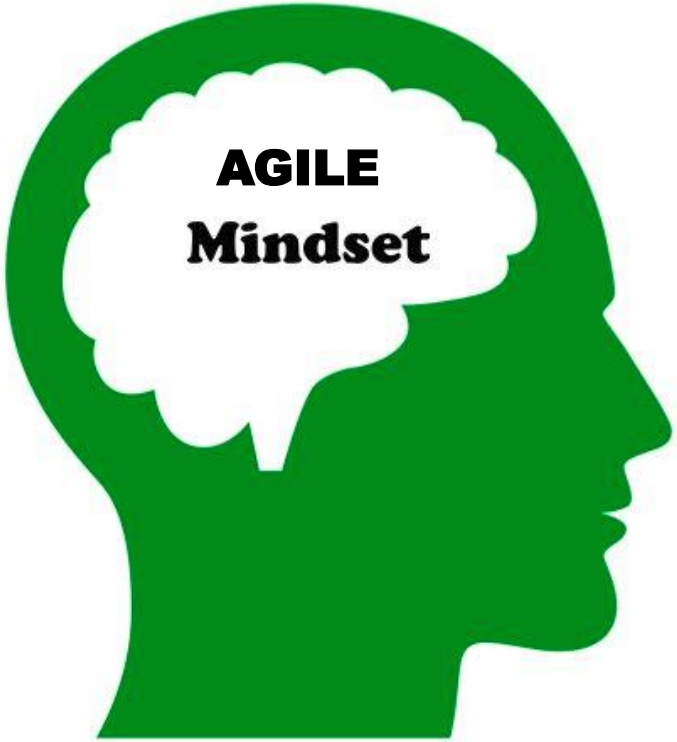
Information System Analysis and Modeling

Lecture 13: Agile methodologies

Mona Taghavi

LaSalle College
Montréal

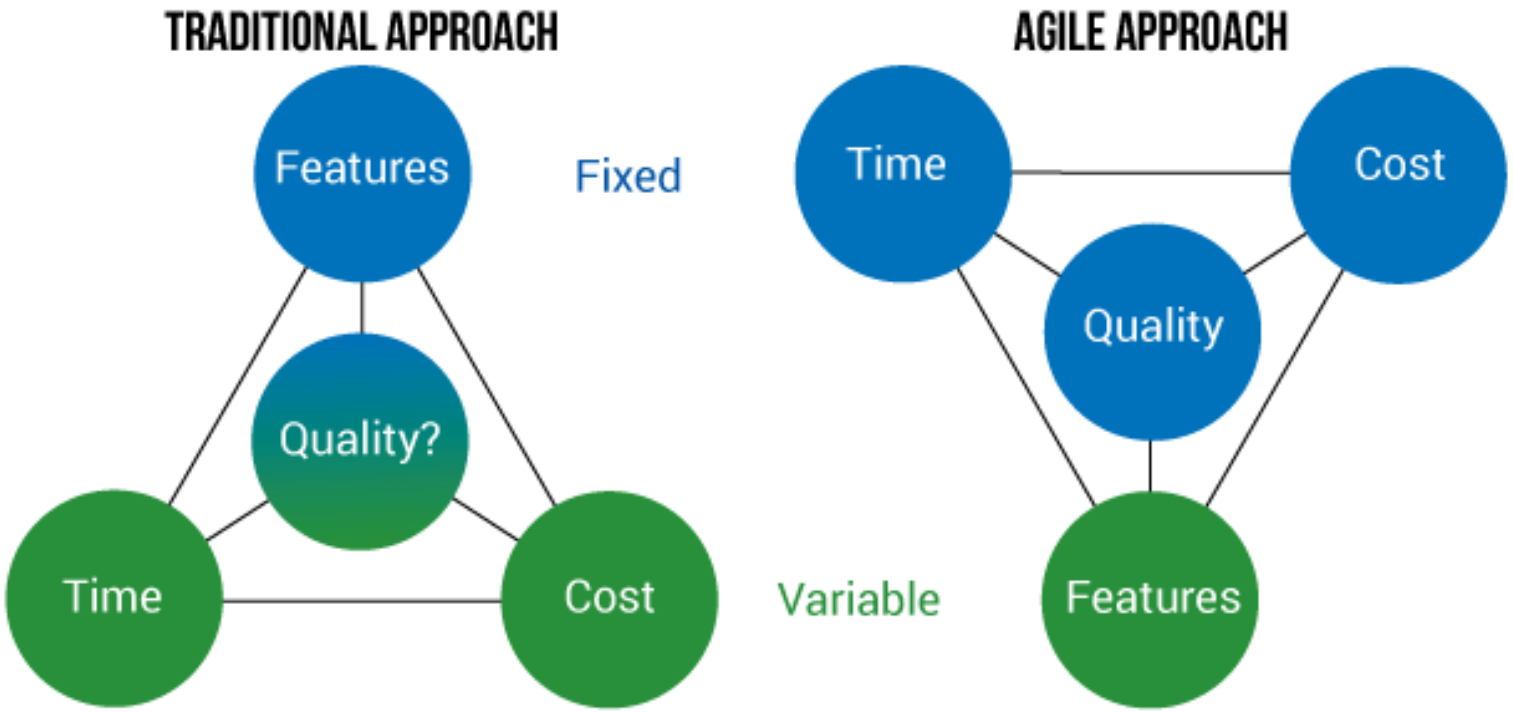
What Kind of Mindset Do You Have?



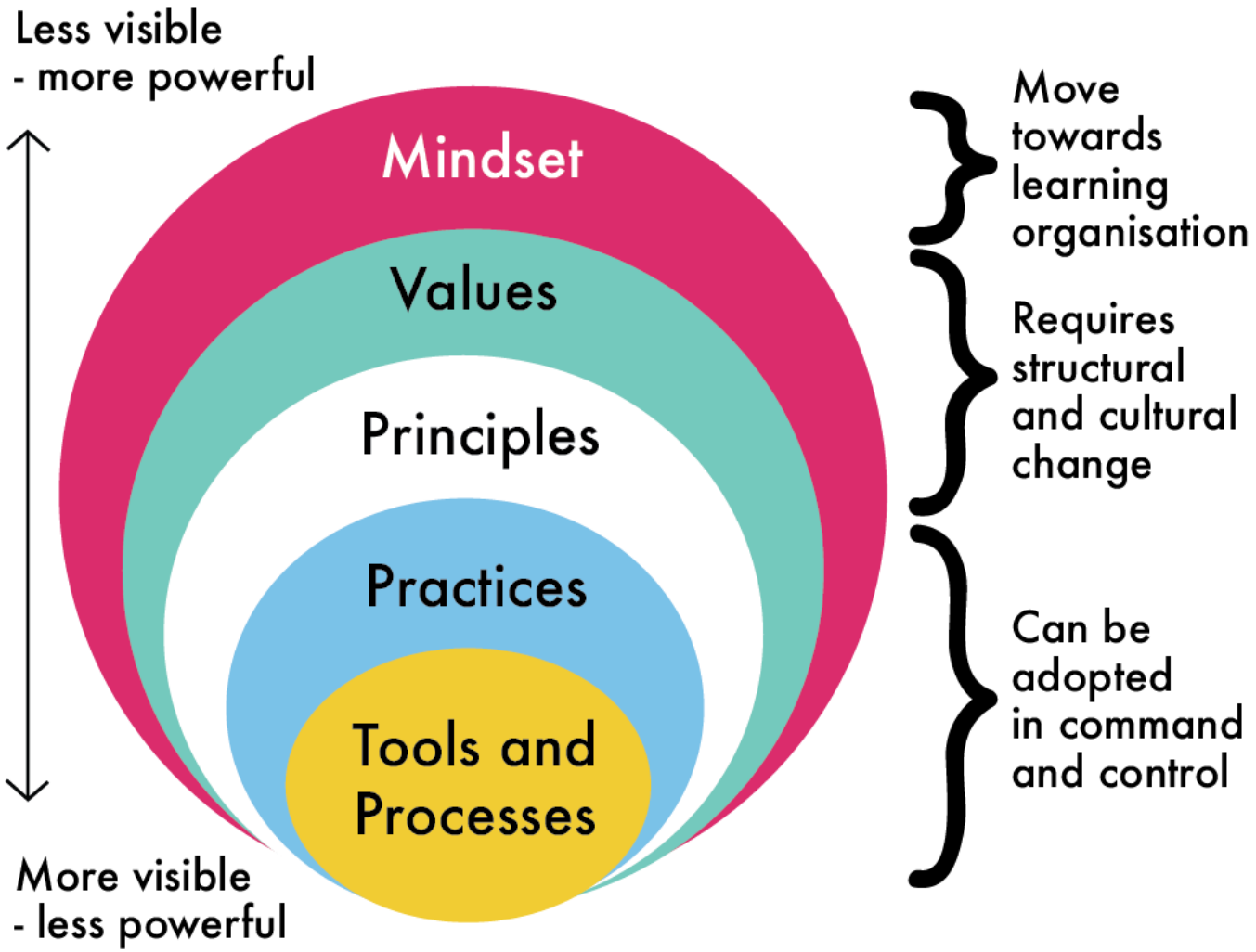
I can learn anything I want to.
When I'm frustrated, I persevere.
I want to challenge myself.
When I fail, I learn.
Tell me I try hard.
If you succeed, I'm inspired.
My effort and attitude determine everything.

I'm either good at it, or I'm not.
When I'm frustrated, I give up.
I don't like to be challenged.
When I fail, I'm no good.
Tell me I'm smart.
If you succeed, I feel threatened.
My abilities determine everything.

Mindset Approach

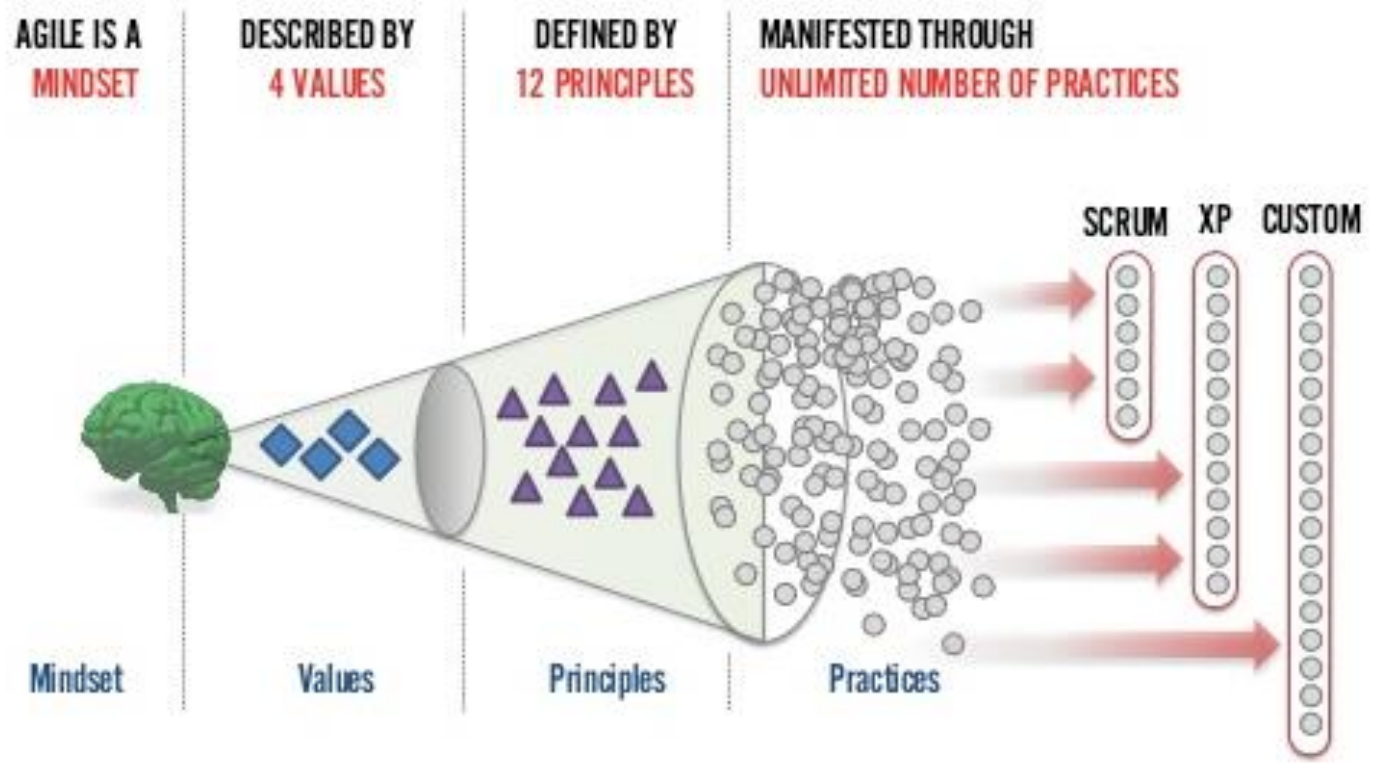


Agile

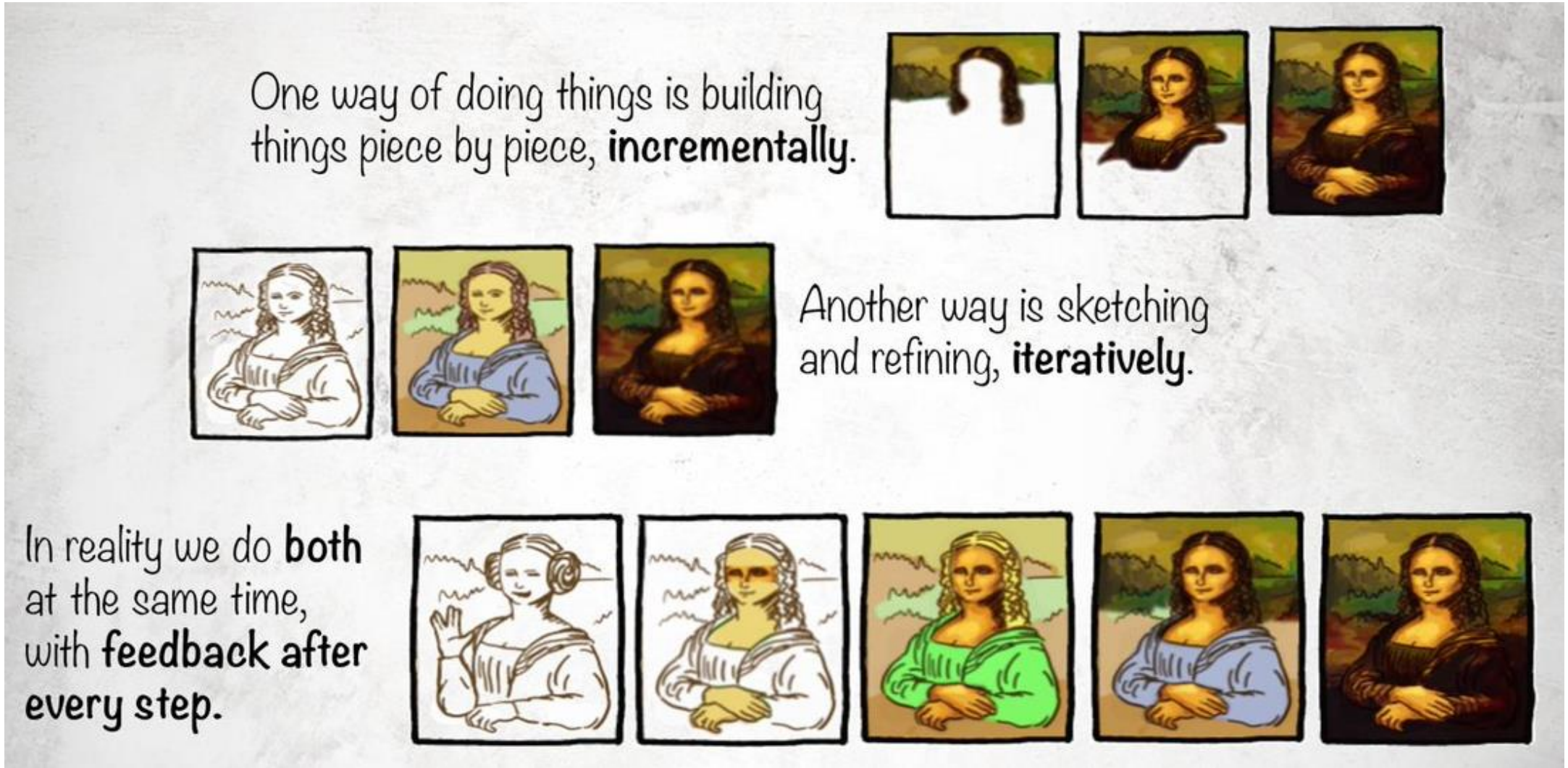


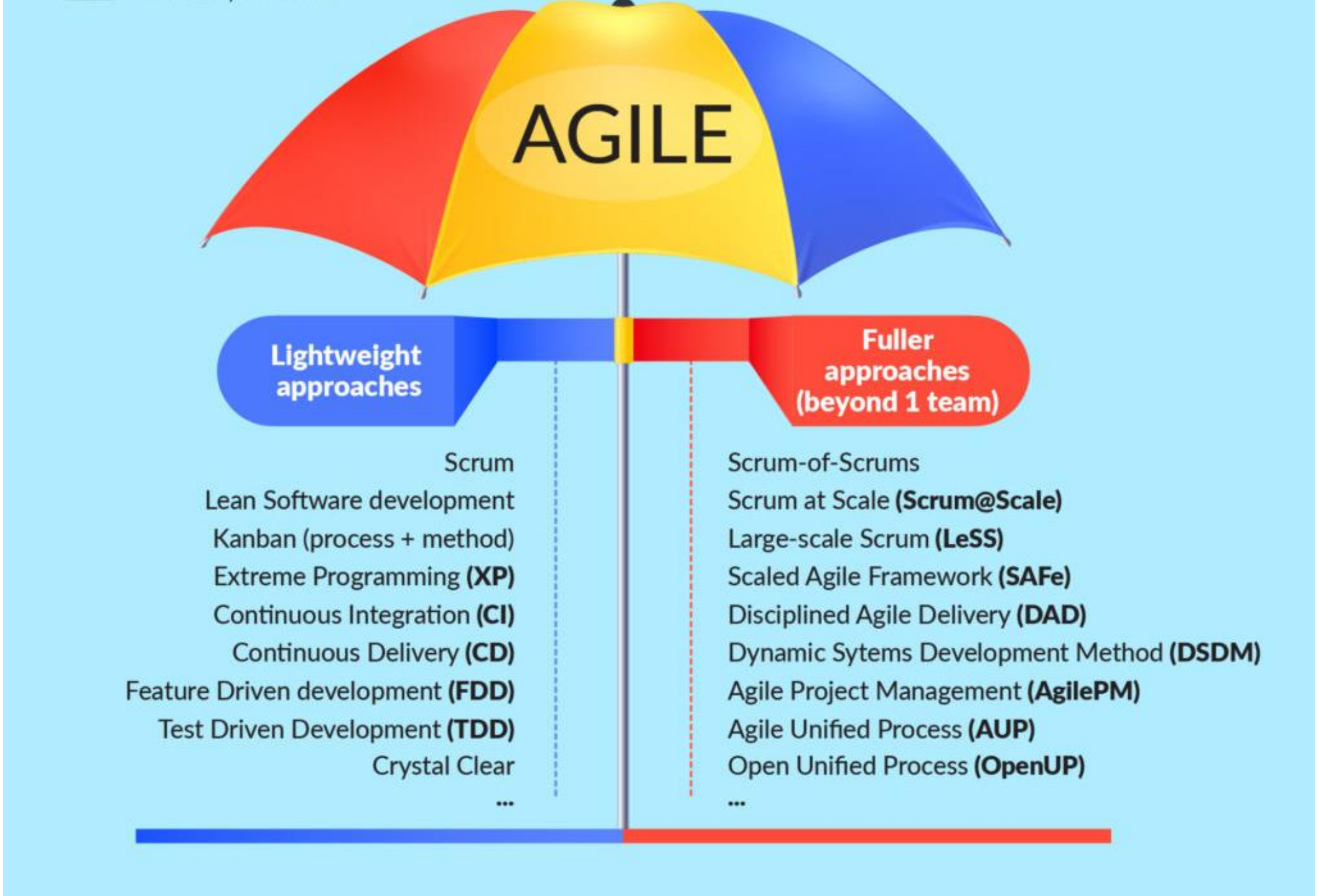
Agile Onion by AWA, Simon Powers

Mindset Manifestation



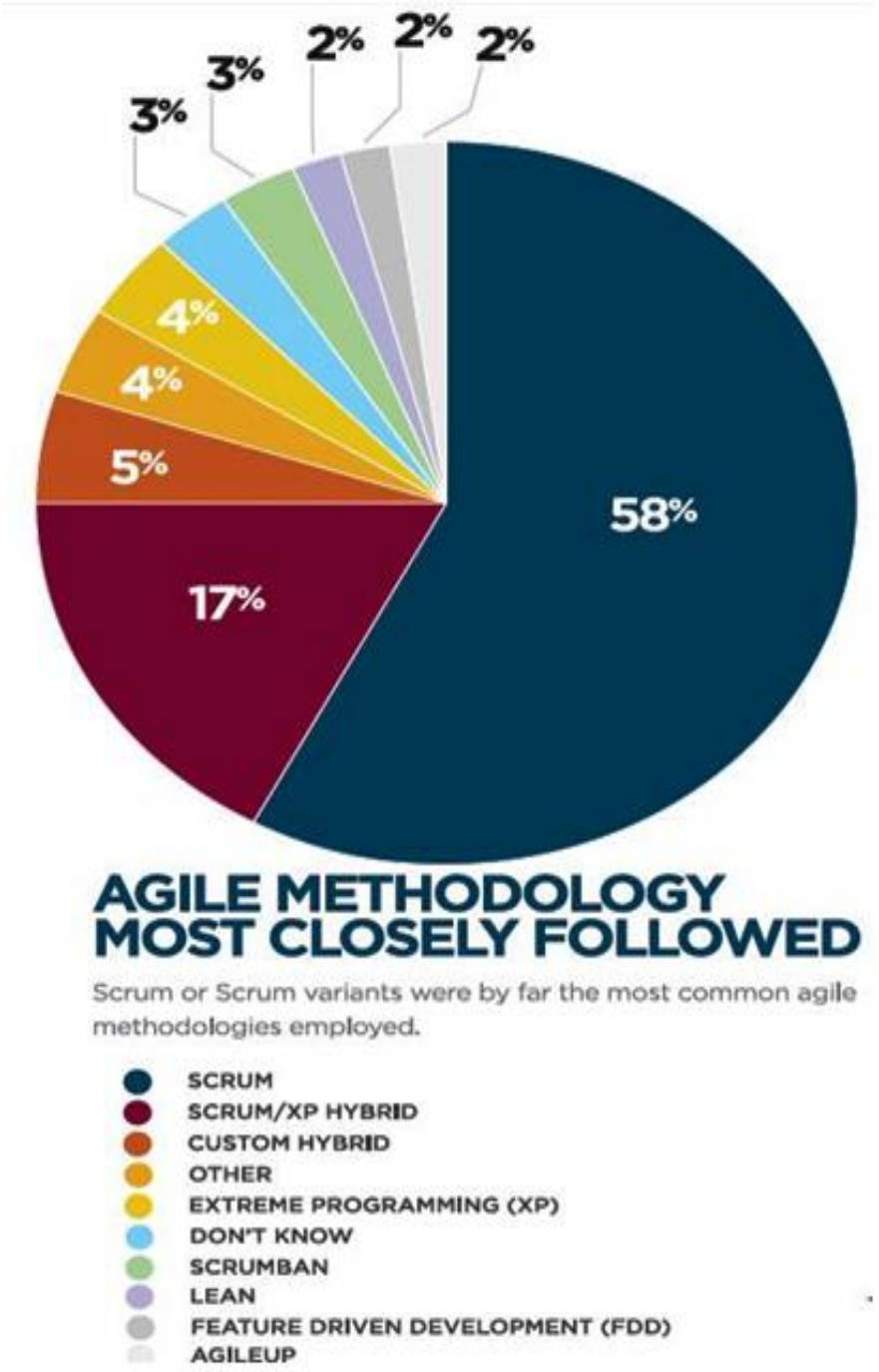
Agile: Iterative and incremental




AGILE
Lightweight approaches
Fuller approaches (beyond 1 team)
Scrum
Lean Software development
Kanban (process + method)
Extreme Programming (XP)
Continuous Integration (CI)
Continuous Delivery (CD)
Feature Driven development (FDD)
Test Driven Development (TDD)
Crystal Clear
...
Scrum-of-Scrums
Scrum at Scale (Scrum@Scale)
Large-scale Scrum (LeSS)
Scaled Agile Framework (SAFe)
Disciplined Agile Delivery (DAD)
Dynamic Sytems Development Method (DSDM)
Agile Project Management (AgilePM)
Agile Unified Process (AUP)
Open Unified Process (OpenUP)
...

The AGILE Methodologies

- AGILE is the blueprint that you will need to build the house – AGILE Methodologies is "**How**" you decide to build it.
- While SCRUM, at the present, is the AGILE Methodology most closely followed, other methodologies presented can be used as a "Hybrid" within SCRUM e.g. Kanban used within the phase of Development in SCRUM, or on their own.



The AGILE Methodologies: XP – Extreme Programming

- A discipline of software development based on values of simplicity, communication & feedback. It works by bringing the whole team together in the presence of simple practices, with enough feedback.
- XP takes commonsense principles and practices to extreme levels, for example:
  - If code reviews are good, we'll review code all the time (pair programming), If testing is good, everybody will test all the time (unit testing), even the customers (functional testing)

XP – Core Values

XP (Extreme Programming) focuses on software development best practices and features 5 Core Values and 13 Core Practices:

XP 5 Core Values:

1. **Simplicity** – Build the simple solution first
2. **Communication** – all team members know what is expected of them
3. **Feedback** – get impressions of suitability early
4. **Courage** – allow work to be visible to others
5. **Respect** – people work together as a team and everyone is accountable

XP 13 Core Practices:

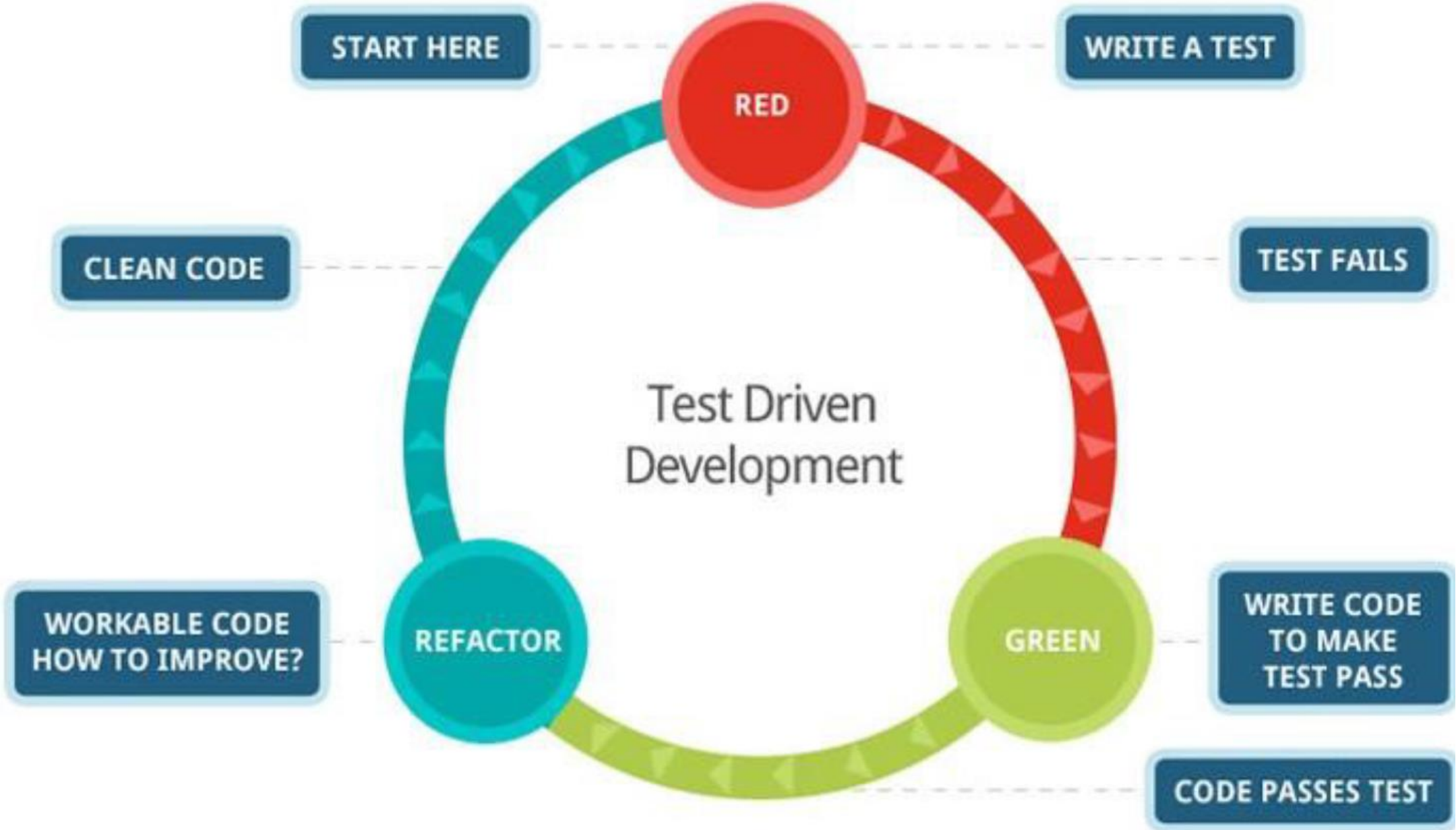
1. Whole Team
2. Planning Games
3. Small Releases
4. Customer Tests
5. Collective Code Ownership
6. Code Standards
7. Sustainable Pace
8. Metaphor
9. Continuous Integration
10. Test Driven Development
11. Refactoring
12. Simple Design
13. Pair Programming

Practices

- **The Planning Game** — Quickly determine the scope of the next release by combining business priorities and technical estimates. As reality overtakes the plan, update the plan.
- **Small releases** — Put a simple system into production quickly, then release new versions on a very short cycle.
- **Metaphor** — Guide all development with a simple shared story of how the whole system works and easy to understand naming.

Practices (Continue …)

- **Simple design** — The system should be designed as simply as possible at any given moment. Extra complexity is removed as soon as it is discovered.
- **Testing** — Programmers continually write unit tests, which must run flawlessly for development to continue. Customers write tests demonstrating that features are finished.
- **Refactoring** — Programmers restructure the system without changing its behavior to remove duplication, improve communication, simplify, or add flexibility.


START HERE
RED
WRITE A TEST
CLEAN CODE
TEST FAILS
Test Driven
Development
WORKABLE CODE
HOW TO IMPROVE?
REFACTOR
GREEN
WRITE CODE
TO MAKE
TEST PASS
CODE PASSES TEST

Practices (Continue …)

- **Pair programming** — All production code is written with two programmers at one machine.
- **Collective ownership** — Anyone can change any code anywhere in the system at any time.
- **Continuous integration** — Integrate and build the system many times a day, every time a task is completed

Practices (Continue …)

- **40 hour week** — Work no more than 40 hours a week as a rule. Never work overtime a second week in a row.
- **On-site customer** — Include a real, live user on the team, available full-time to answer questions.
- **Coding standards** — Programmers write all code in accordance with rules emphasizing communication through the code.

Toyota Production System

- TPS revolutionized the Japanese economy post WWII –delivery of innovative products faster and with less waste

Toyota Production System

- Toyota sent people on study trips to America to learn from the leaders of the automotive industry. The Americans were well known for their mass production technique which was popularized by Henry Ford in his Ford Motor Company.
- Toyota could not realistically copy the American way of working.
- Debut in the manufacturing industry mid-20th century as part of the Toyota Production System (TPS), i.e., “produce what you need, only as much as you need, when you need”.

Toyota Production Model

- Let production be dictated by demand. This principle is called pull production.
- Do not view production as optimal at any point, but see it at suboptimal at any point and try to improve it on a daily basis. This is the Kaizen principle.
- Focus on detecting and eliminating waste during the production. Please note that Toyota defined waste in a different way than we define it in our everyday language.

	Traditional Manufacturing	Lean Manufacturing
Scheduling system	Made to Stock (Push)	Made to Order (Pull)
Lead Time	Long lead time	Short lead time
Flexibility	Not flexible	Highly flexible
Demand/ Supply	Supply oriented	Demand oriented

McDonald's Revolution

Fast Food business was not fast.

Old Process

1. Waiter walks to car. Gives menu.
2. Waiter returns and takes order.
3. Waiter walks back to kitchen.
4. Food prepared.
5. Waiter walks to customer. Serves food.
6. Waiter returns to pick up plates to wash.

Old Process

1. • Waiter walks to car. Gives menu.
2. • Waiter returns and takes order.
3. • Waiter walks back to kitchen.
4. • Food prepared.
5. • Waiter walks to customer. Serves food.
6. • Waiter returns to pick up plates to wash.

New Process

1. • Customer places own order at restaurant.
2. • Food prepared and served in bag.
3. • Customer eats. Throws away trash.
4. • ....
5. • ....
6. • ...

(Non) Value Adding Activities

- Value Adding Activity:
  1. Transform form or function of service or product.
  2. Customer must be willing to pay for that transformation.
  3. Done correctly the first time.

- Non-Value Adding Activities (NVAA) do not meet one or several of these criteria.

Lean, Eliminating Waste, and Seeing the Whole

- *Lean is a mindset—a mental model of how the world works.*

- The core idea is to maximize **customer value** while minimizing waste.
  - **Non value adding activity = waste**

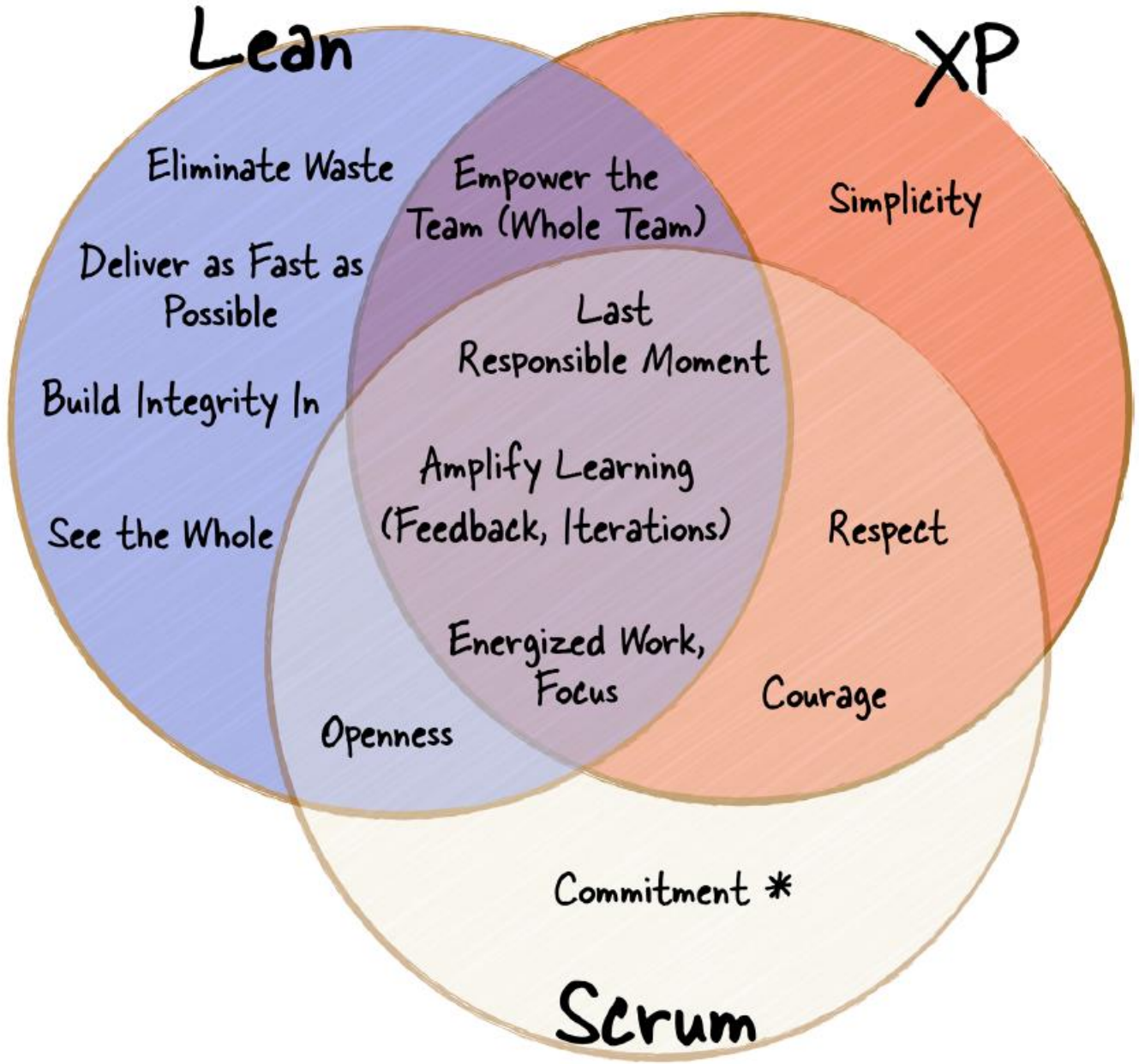
- Lean: faster, cheaper, without sacrifice of quality

LEAN Principles

LEAN Product Development

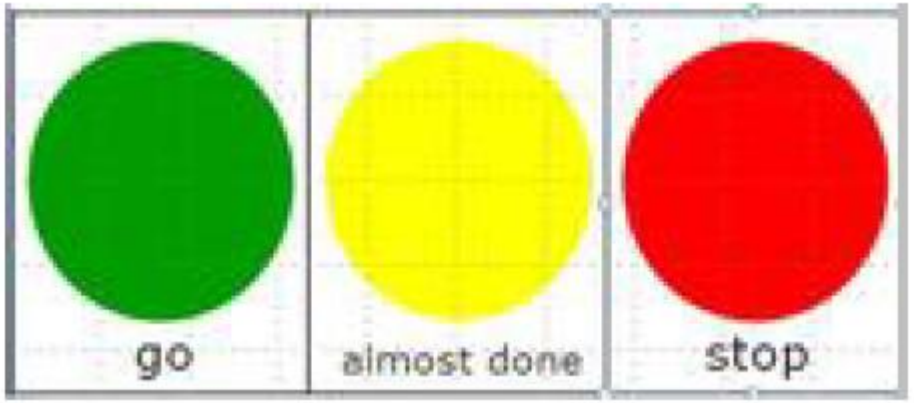
The Seven Wastes of Lean:

1. Partially Done Work – Work started but not yet complete
2. Extra Processes – Extra work that does not add value
3. Extra Features – Features that are not required
4. Task Switching – Multitasking between several different projects
5. Waiting – Delays waiting for reviews and approvals
6. Motion – The effort required to communicate or move information or deliverables from one group to another
7. Defects – Defective documents or software that needs correction


Lean
XP
Scrum
Eliminate Waste
Deliver as Fast as Possible
Build Integrity In
See the Whole
Empower the Team (Whole Team)
Last Responsible Moment
Amplify Learning (Feedback, Iterations)
Energized Work, Focus
Openness
Simplicity
Respect
Courage
Commitment *

KANBAN

- a Japanese manufacturing system in which the supply of components is regulated through the use of a card displaying a sequence of specifications and instructions, sent along the production line.
- Literally translates to “signal card”



- Core objective is to provide continuous delivery of value without overburdening the development team.

KANBAN

- As a software tool, Kanban is a method for managing the creation of products with an emphasis on continual delivery while not overburdening the development team. Like Scrum, Kanban is a process designed to help teams work together more effectively.

- Kanban is about *helping a team improve the way that they build software*. A team that uses Kanban has a clear picture of what actions they use to build software, how they interact with the rest of the company.

Lack of Roles is a Strength!

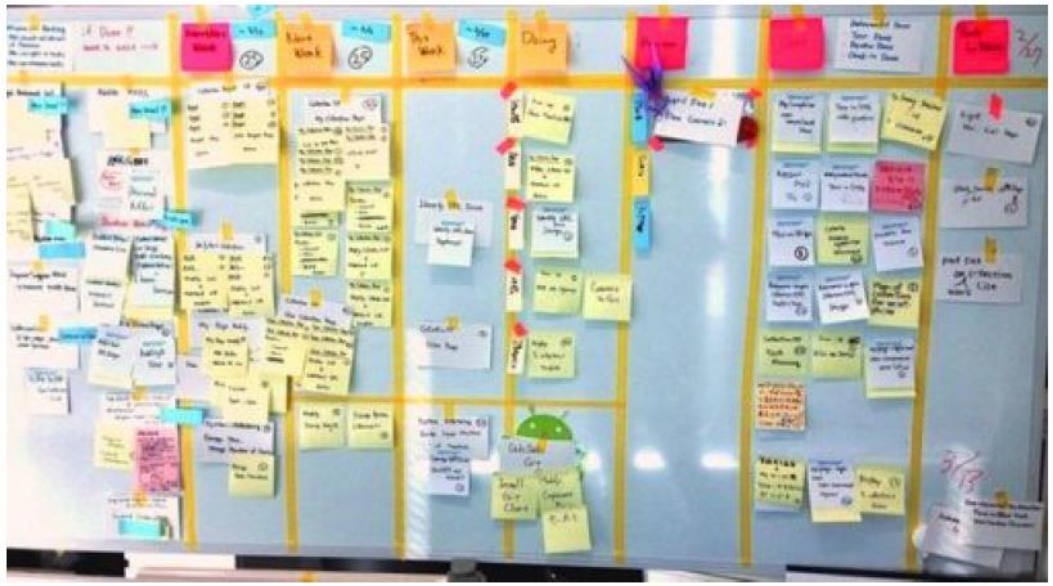
- No prescribed roles in Kanban!
- Roles remain same as today
- Build cross-functional skills
- Kanban Change Agent
  - Kanban Lead
  - Kanban Coach
  - Leads Kanban Initiative
  - Facilitates Kanban system design
  - Helps remove impediments
  - Servant Leader

Kanban Core Practices

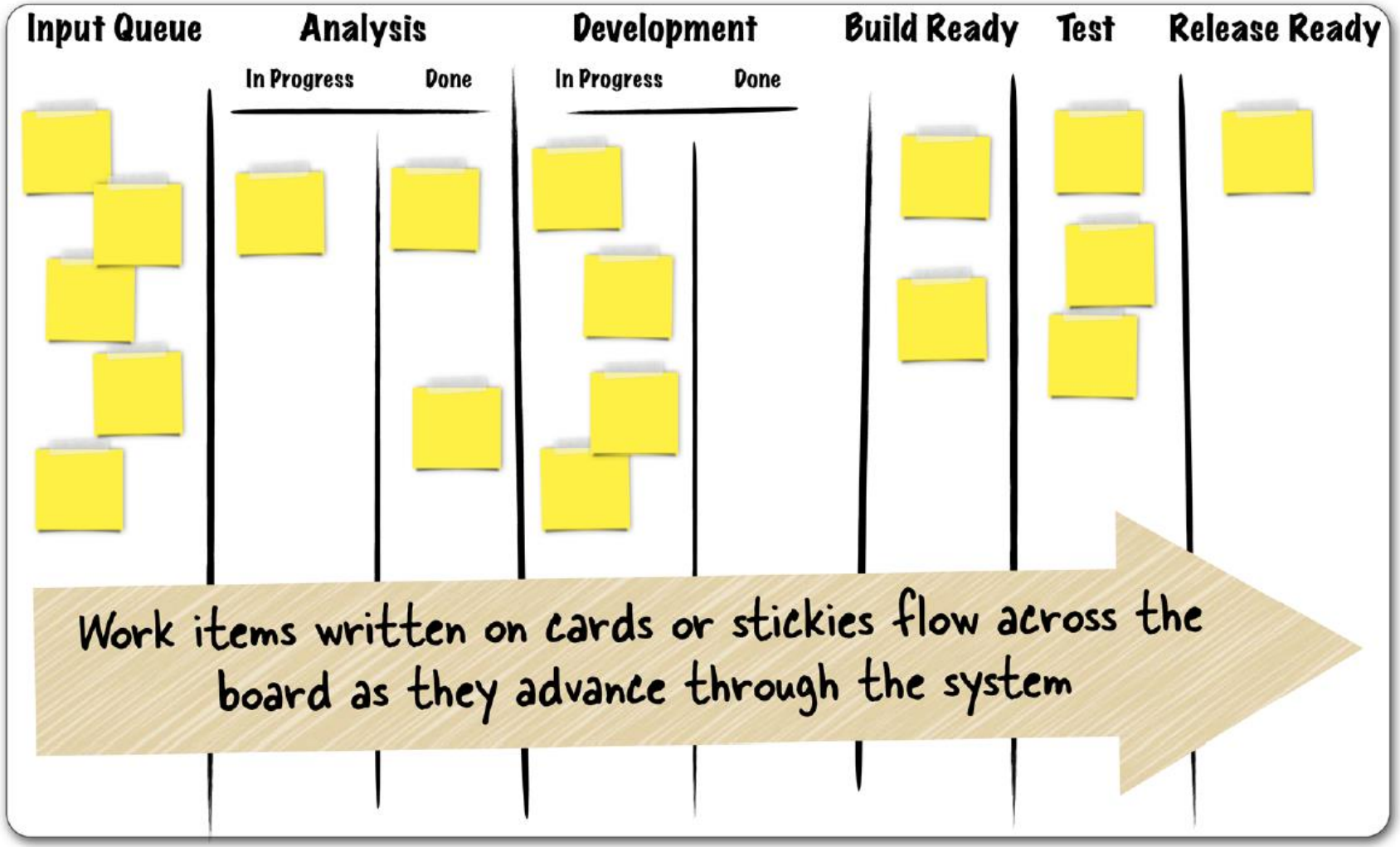
1. Visualize
2. Limit Work-in-Progress
3. Manage Flow
4. Make Policies Explicit
5. Implement Feedback Loops
6. Improve Collaboratively, Evolve Experimentally

1. Visualize

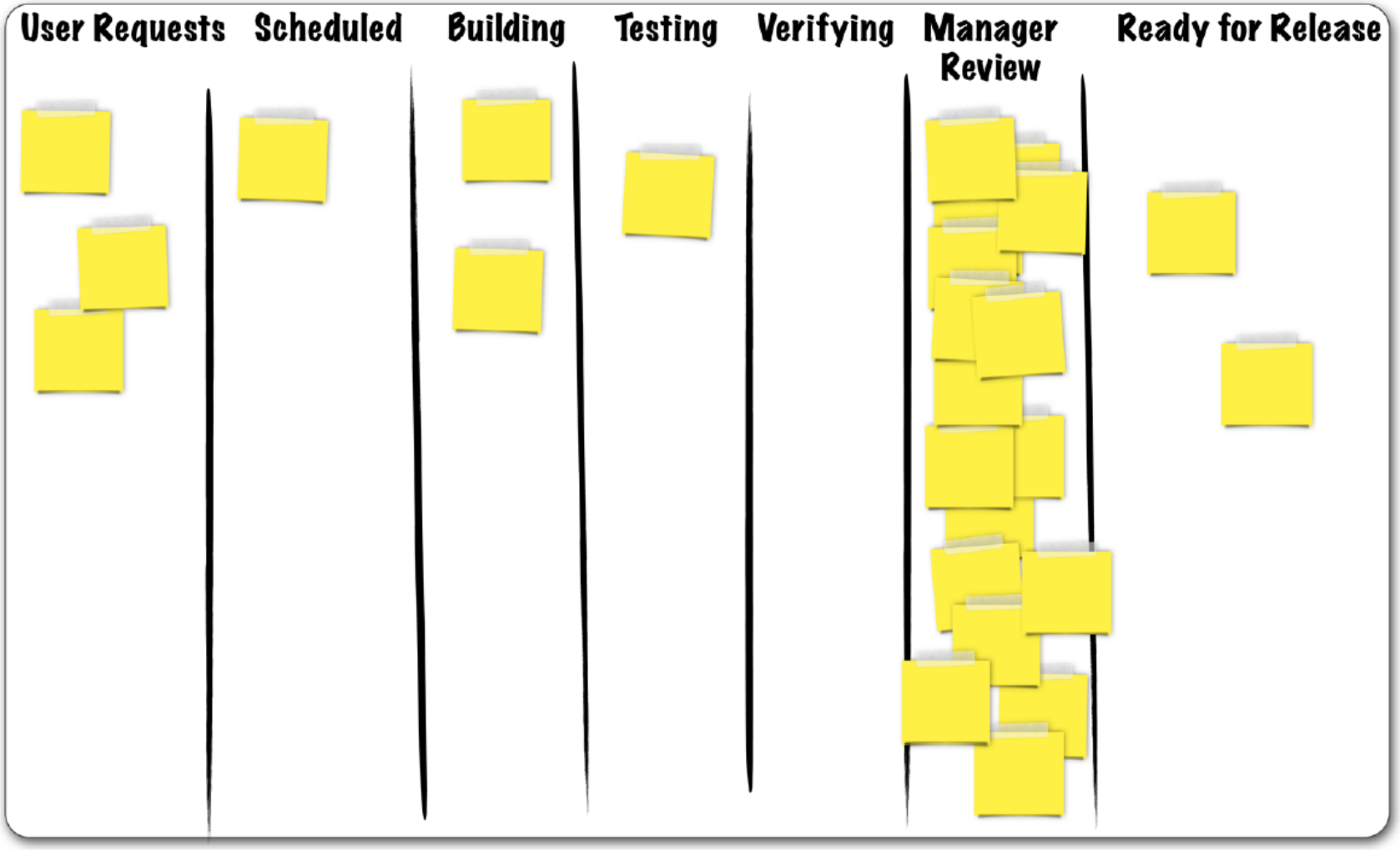
- Visualize Workflow
  - Make the invisible, visible
  - Mechanism
  - Interactions
  - Handoffs
  - Queues & Buffers
- Cards Walls
  - View of system
  - Visually track WIP
  - Self-organize, live collaboration
  - Near real-time project status



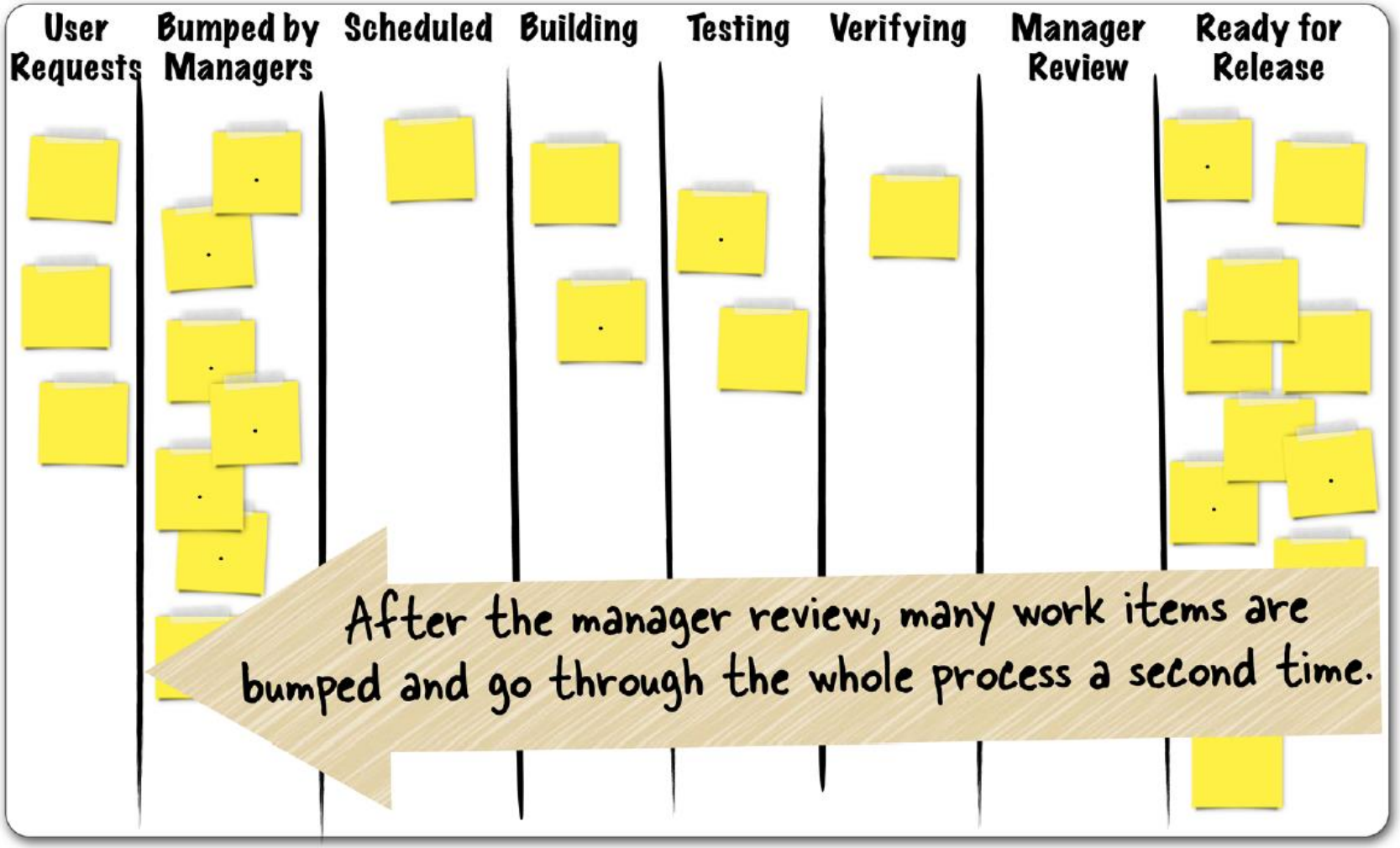
Kanban board



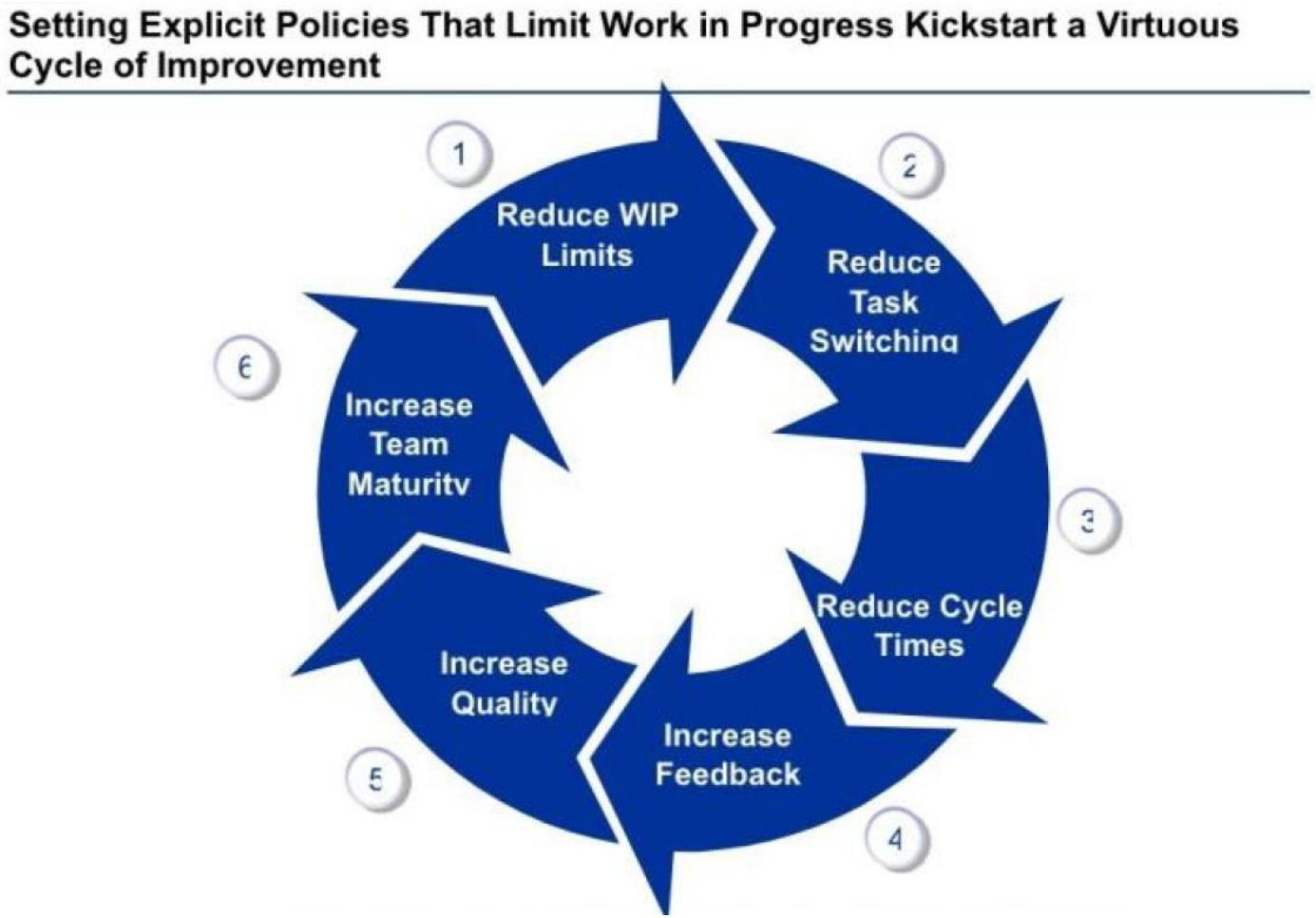
When you use a kanban board to visualize the workflow, problems caused by unevenness become easier to spot



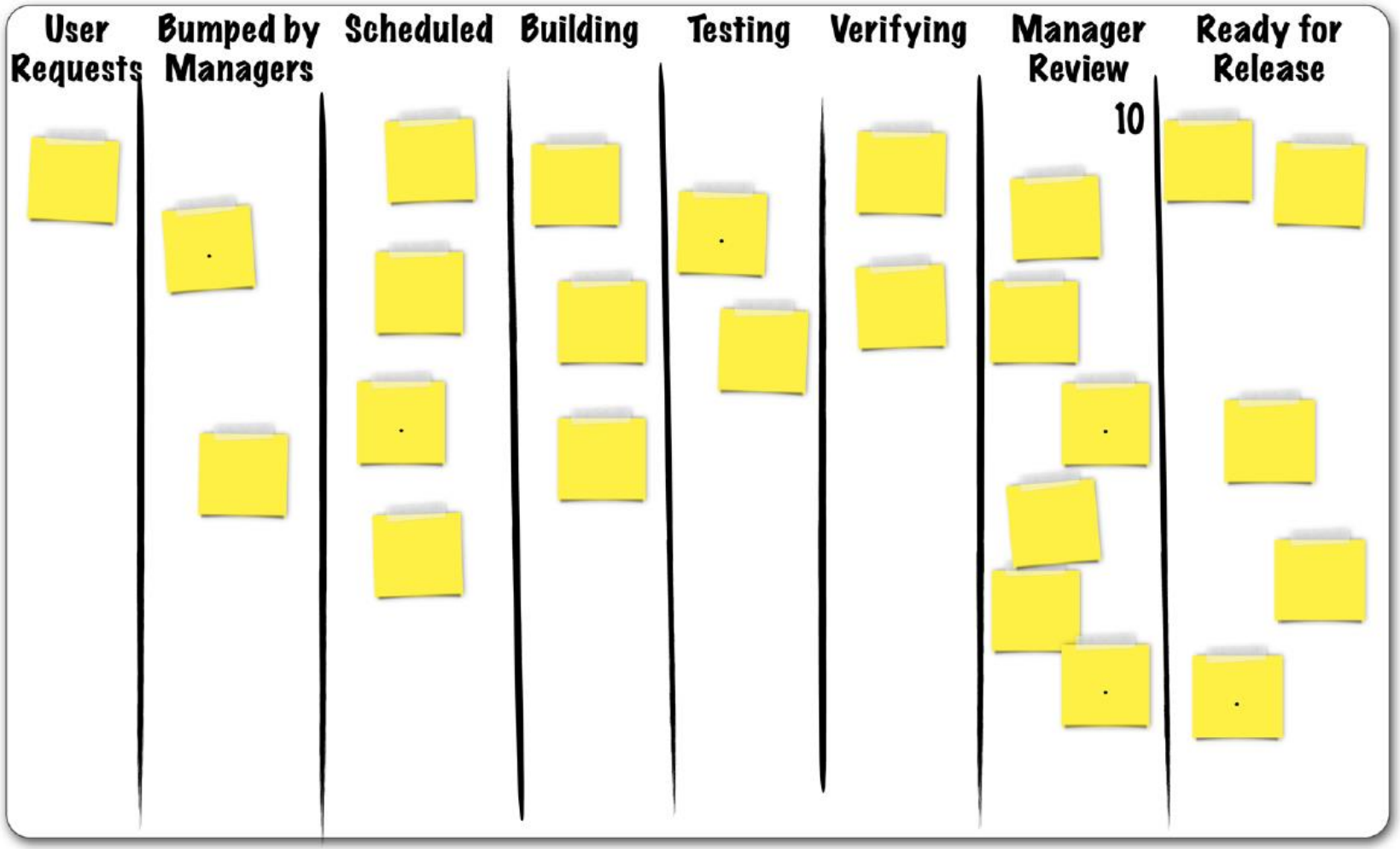
The kanban board makes the waste more obvious when you can see stickies go through the workflow more than once.



2. Limit Work-in-Progress

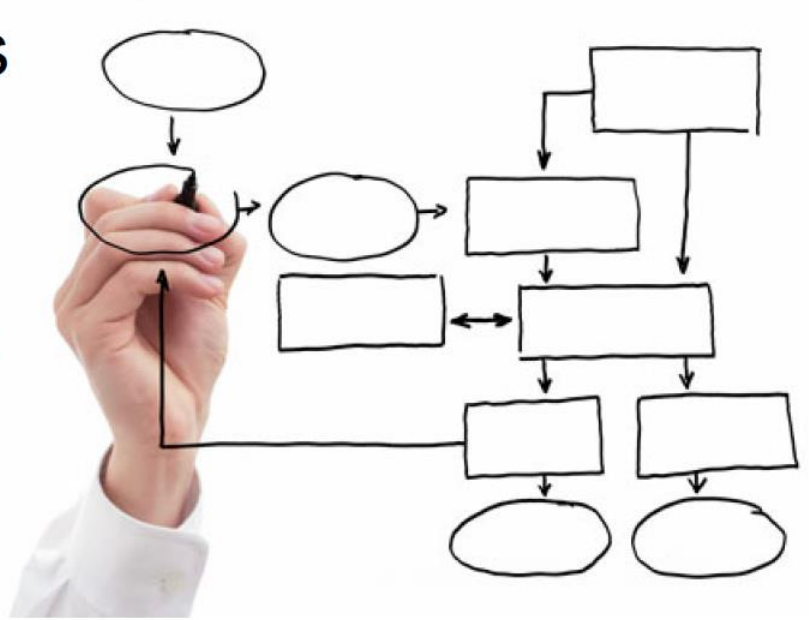


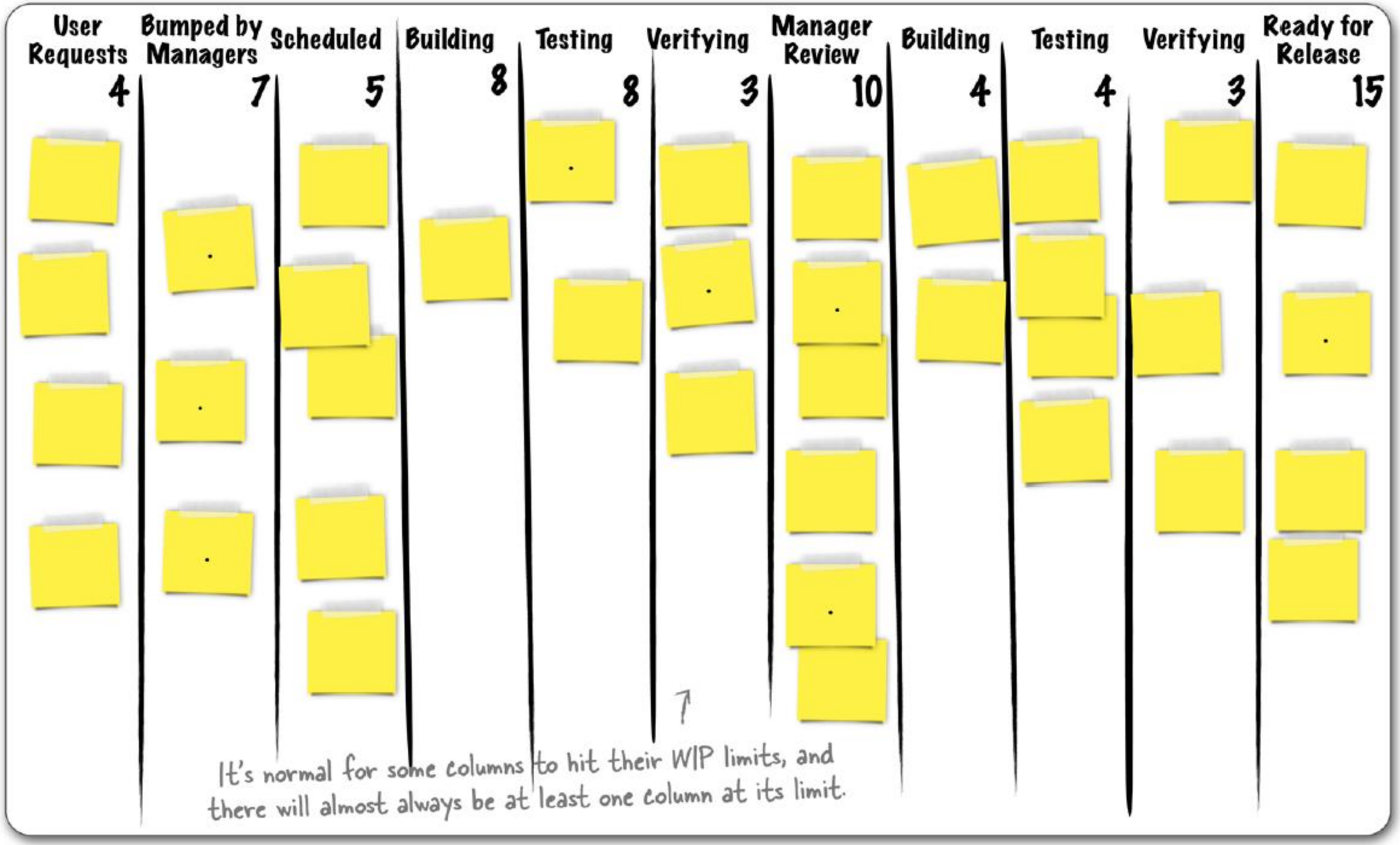
The number 10 in the Manager Review column is its WIP limit



3. Manage Flow

- Start with Existing Processes
- Seek smoothness, timeliness, good economic outcomes
- Drives Improvement
- Focus on Flow vs. Waste
- Measure Work
- Work-in-Progress (WIP)



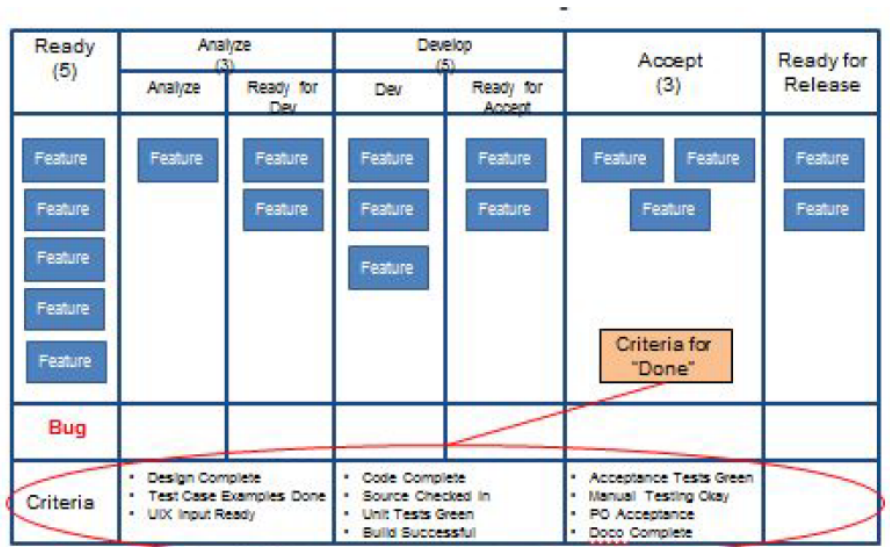

User Requests
4
Bumped by Managers
7
Scheduled
5
Building
8
Testing
8
Verifying
3
Manager Review
10
Building
4
Testing
4
Verifying
3
Ready for Release
15
It's normal for some columns to hit their WIP limits, and there will almost always be at least one column at its limit.

4. Make Policies Explicit

- Process Policies
- Explicit and Visible
- Build trust in the system
- Helps everyone understand what's expected
- Enables team members to make decisions
- Let's team decide how to keep work flowing
- Changes as teams evolves

Policy Examples

- Actions to be taken when finished coding
- Impediment/Blocker actions to be taken
- Production Bugs priority over QA Bugs, both priority over new development
- Work requested through different channel
- Who adds work to boards
- Ideas for design changes
- WIP Limits
- Definition of Done
- < 5 days dev



Implement Feedback Loops

- Purpose:
  - Compare expected outcomes to actual outcomes
  - Make process and policy adjustments

- Feedback Loops:
  - Standup Meeting
  - Service Delivery Review
  - Operations Review
  - Risk Review

When is Kanban a Good Fit?

- Uneven flow of work
  - Large batch transfers
  - Unplanned, speculative, disruptive requests
  - Blocking issues
- Deferred commitment is desirable
  - Priorities change frequently
  - Constant re-planning
  - High abandonment, discard, abort rates
  - Delivered work, never used
- System or workers are overburdened
  - Too much work-in-progress
  - Stressed workers
  - Poor quality
  - Long/unpredictable lead times, Long wait queues

When to Consider Other Options?

- Highly mature organization
  - Demand never exceeds capacity, flow is smooth and never interrupted, no overburden
- Facing extinction
  - No time to let Kanban work its magic, need revolution vs. evolution
- Boss lacks patience for incremental improvement to take effect