MASSEY UNIVERSITY
TE KUNENGA KI PŪREHUROA
UNIVERSITY OF NEW ZEALAND

Software Engineering Design and Construction
159.251

The *Agile* Thinking in Software Development

Amjed Tahir
a.tahir@massey.ac.nz

Overview

- A software development methodology is a *framework* that is used to *structure*, *plan*, *execute* and *control* the process of developing software systems.
- A set of defined processes and practices
  - Guide the project from ***requirements gathering*** to ***deployment*** and ***maintenance***.
- Covers all aspects of the development process

Traditional view of software development

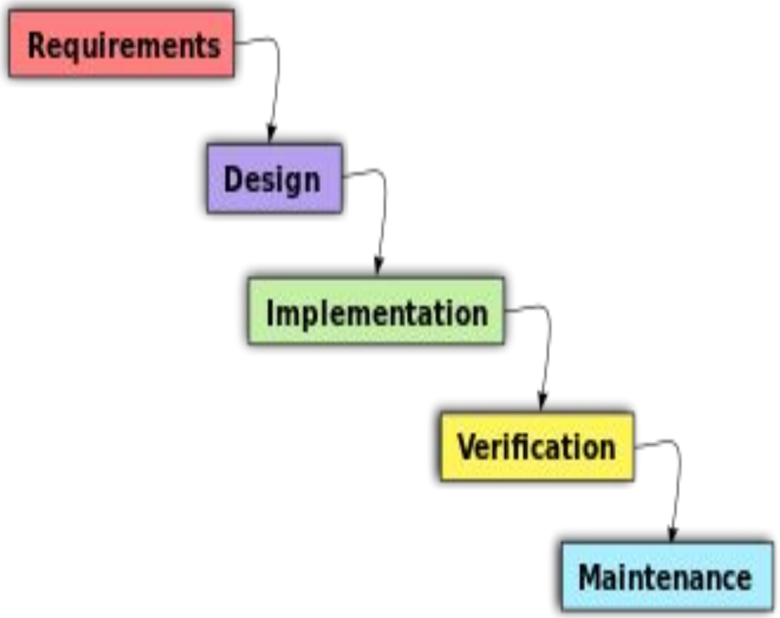
- Methods are well-defined, well-structured and well-organised ⇒ ***a good thing!***
- Pure engineering-oriented ⇒ similar to other engineering disciplines

  But:

  - Software development is different than other engineering discipline.
  - Many researchers and practitioners are questioning the 'engineeringness' of software development.

The Waterfall model

- is a sequential work process, which is flow looks like a waterfall (flowing steadily downwards).
- Traditionally, acknowledged as the de-facto model in software development.
- Well-structured .. a step-by-step model.



The problem

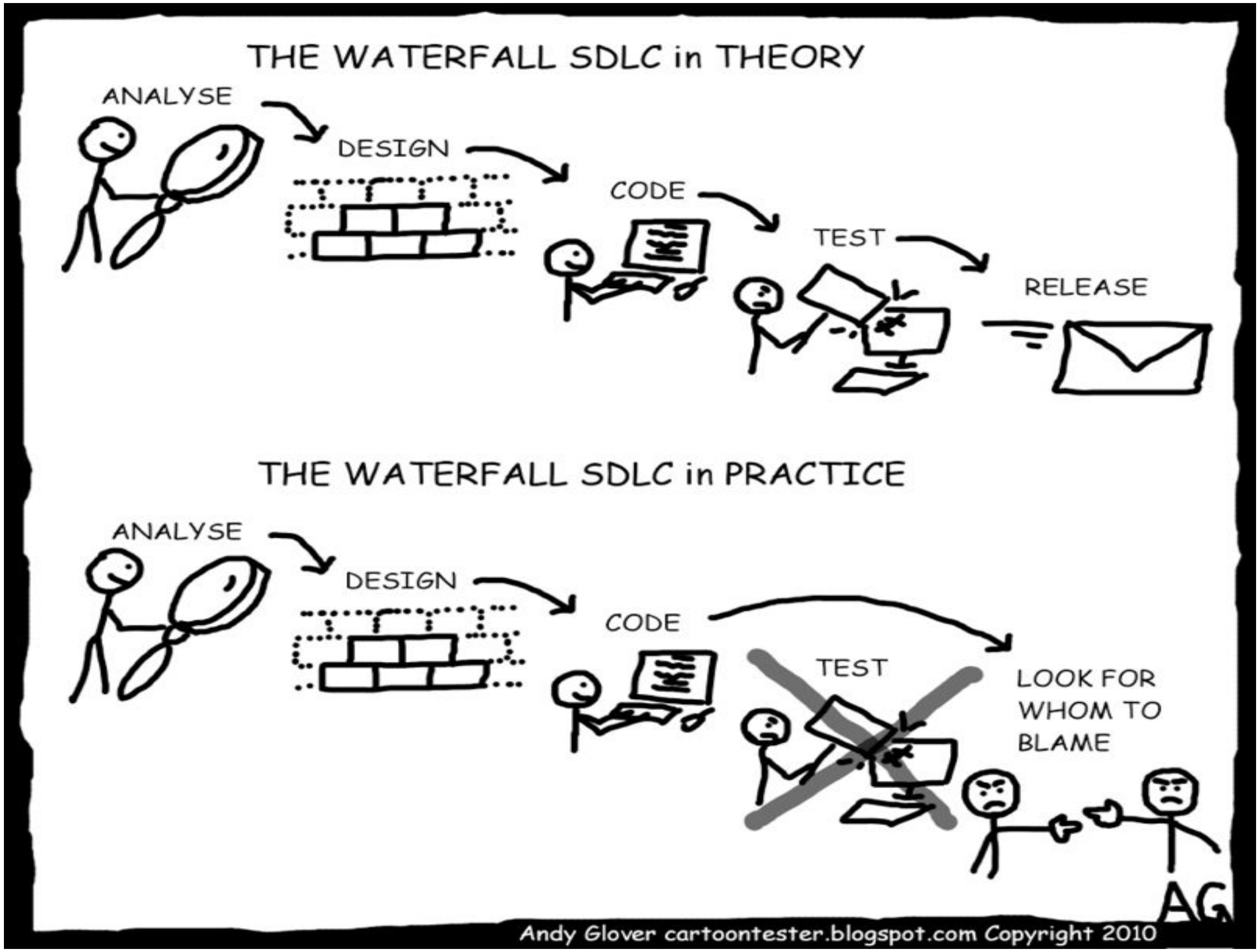
- Software development projects
- can be a real ‘roller coaster’!
  - ***Customers keep changing their minds***, so they change their requirements.
  - ***Technology*** keeps *changing*, and can be also ***inflexible…***
  - Traditional view of software development focus more on the technological aspects, and less on the human and management aspects of the development

Waterfall in a perfect world!

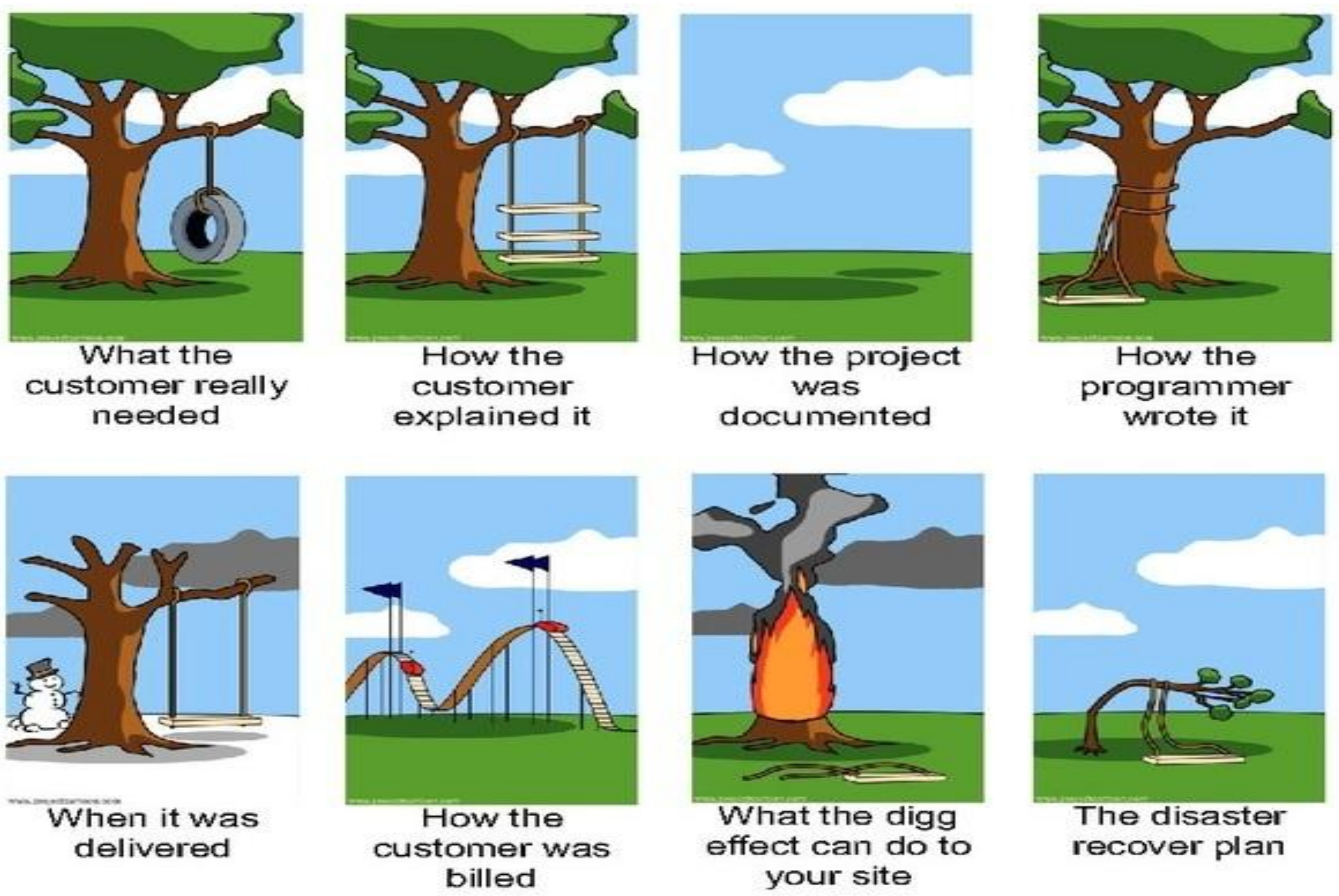
- **Requirements** are *simple* and *easy-to-understand*…
- **Customers** *do not change their requirements*.
- If a change is required, it will be informed at an early stage of the development.
- **Clients are engaged** with the development teams.

But in reality….

- Requirements are **quite complex**…
- Customers keep **changing their minds**
- Requirements change at **critical stages of development** (e.g., at time of deployment!)
- Clients are **not engaged** with the development teams.


THE WATERFALL SDLC in THEORY
ANALYSE
DESIGN
CODE
TEST
RELEASE
THE WATERFALL SDLC in PRACTICE
ANALYSE
DESIGN
CODE
TEST
LOOK FOR WHOM TO BLAME
AG
Andy Glover cartoontester.blogspot.com Copyright 2010

Real Life!


What the customer really needed
How the customer explained it
How the project was documented
How the programmer wrote it
When it was delivered
How the customer was billed
What the digg effect can do to your site
The disaster recover plan

What’s needed then?

- A ***flexible*** method to deal with unpredicted changes.
- Preferably an ***iterative*** and ***incremental*** methods.
- Focus more on the ***customers*** requests, by ***involving*** them from early stages.
- “Break communication barriers” between different team members by promoting a ***collaborative*** work environment.

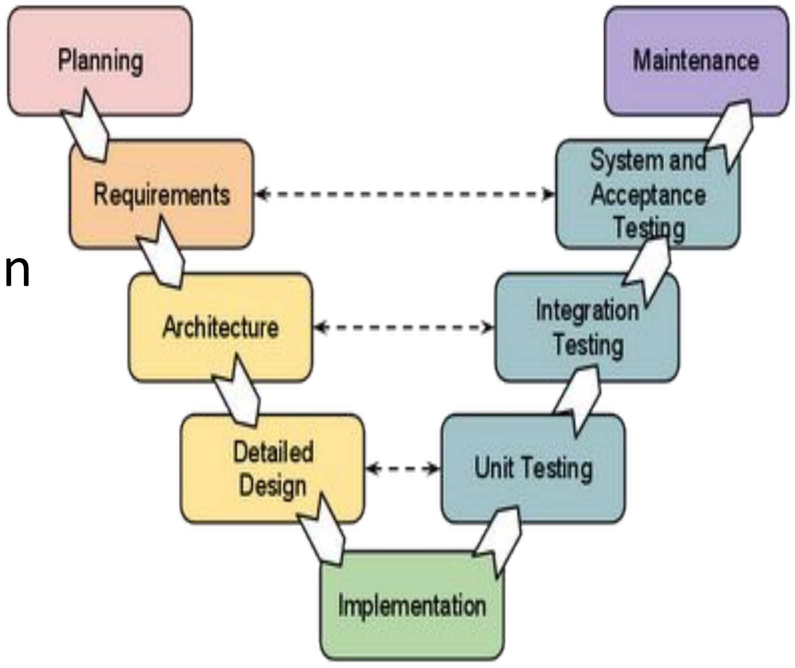
aka ⇒ the ***Agile*** way!

Before Agile….

- A number of methods were designed to overcome some of the Waterfall challenges.

- The ***V-model***

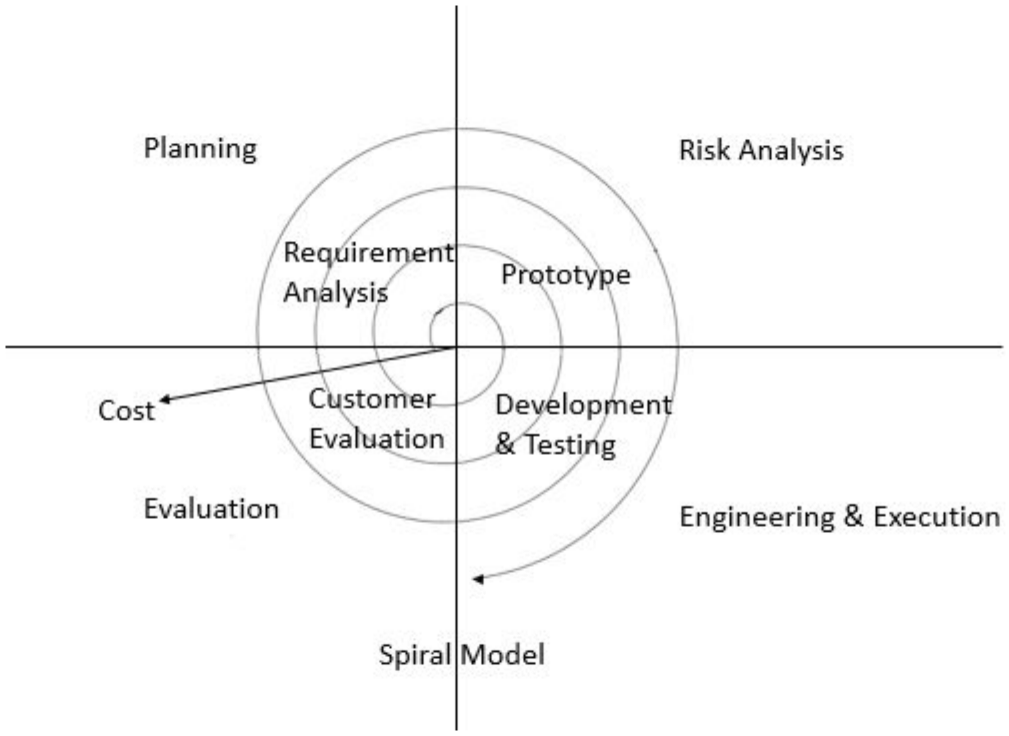
A waterfall model with verification and validation steps



Before Agile….

- Then was the ***Spiral Model***

One of the first models to implement the concept of ***incremental* and *evolutionary*** development.



Agile history

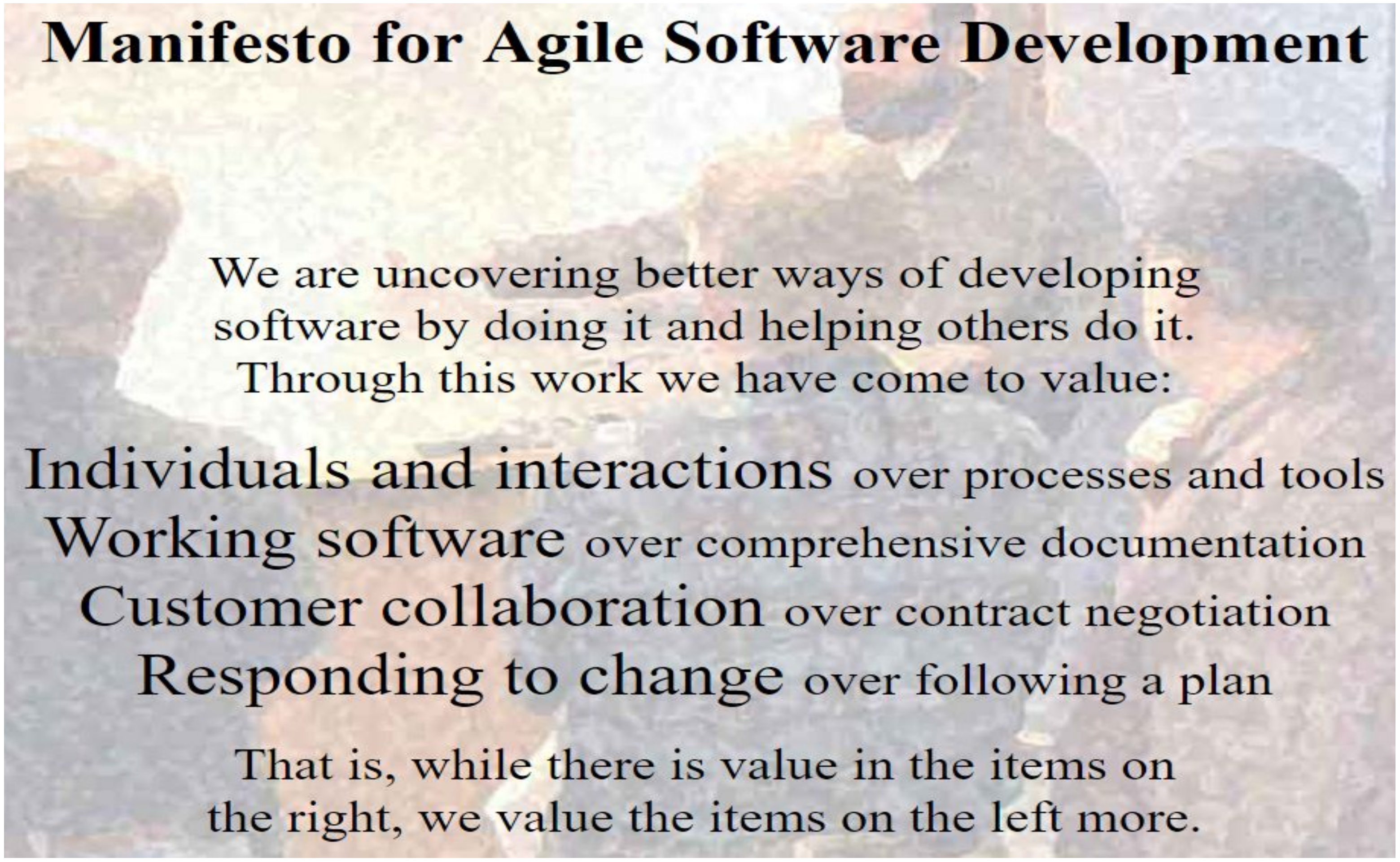
- All started in February 2001
- 17 professional software developers met to discuss issues with classical development approaches.. Including:

Kent Beck

Martin Fowler

- At the time, those developers (like others) thought that software development seems unproductive….

The Agile Manifesto



Manifesto for Agile Software Development

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

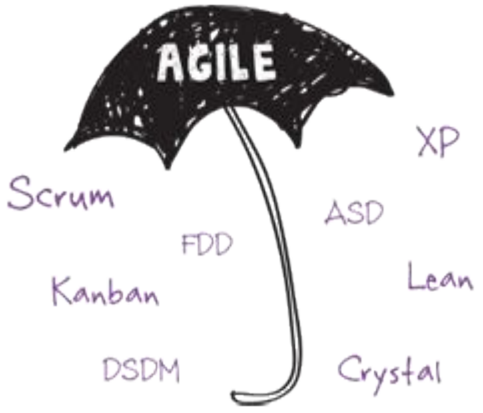
Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

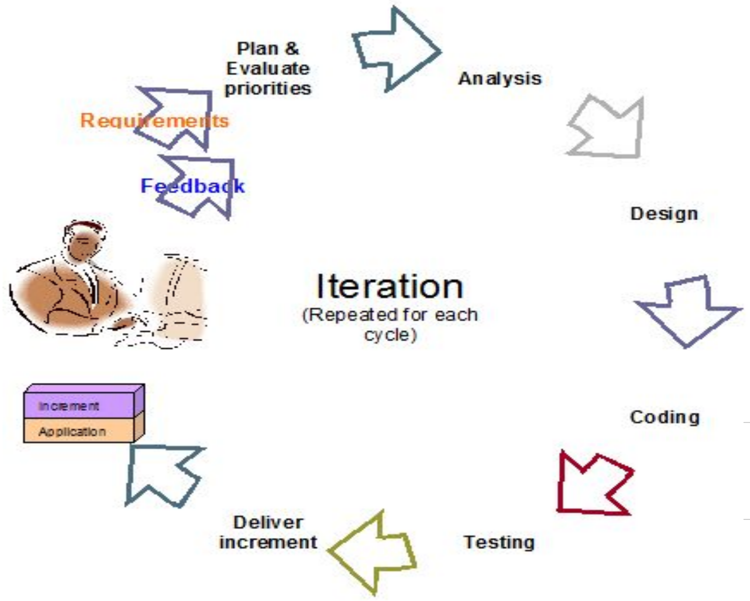
That is, while there is value in the items on
the right, we value the items on the left more.

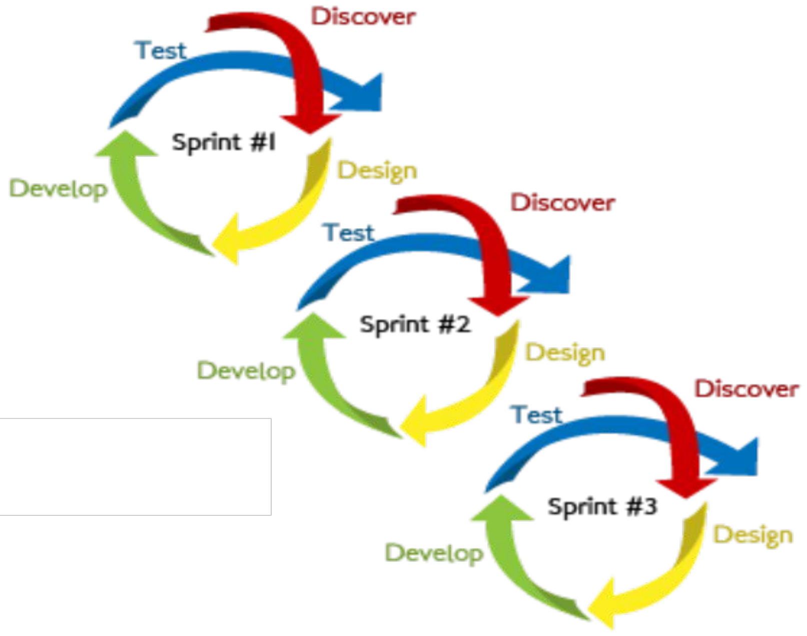
http://www.agilemanifesto.org/

What is Agile

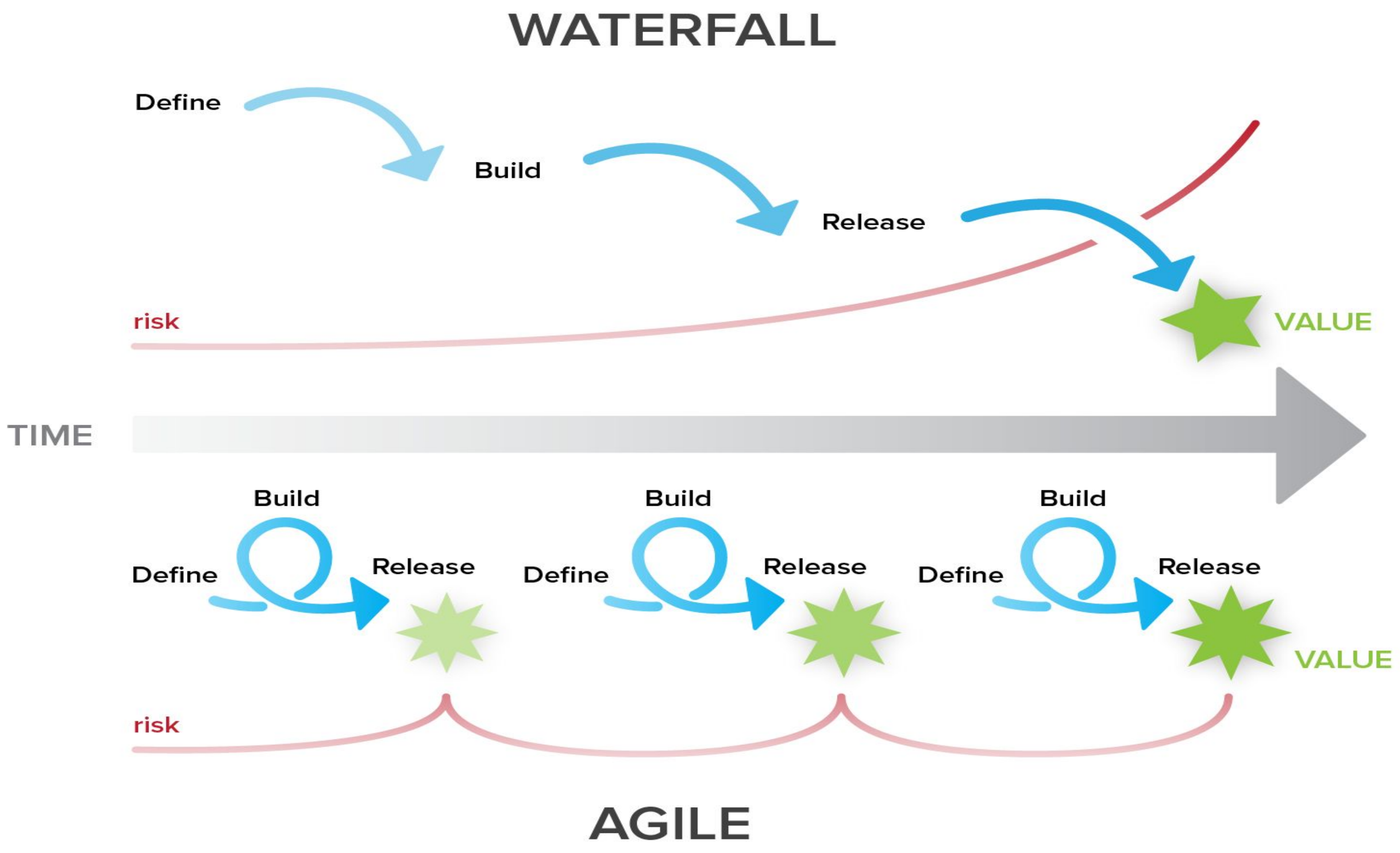
- Agile is an alternative to traditional software development methodologies.
- Focus on lean, collaborative and iterative development.
- It helps projects to respond to unpredictability (changes) through incremental and iterative development.
- Agile is not a methodology by itself..
    - Umbrella method
    - A group of principles and practices shared by different methods




Plan &
Evaluate
priorities
Analysis
Requirements
Feedback
Design
Iteration
(Repeated for each cycle)
Increment
Application
Coding
Testing
Deliver increment


Discover
Test
Sprint #1
Design
Develop
Discover
Test
Sprint #2
Design
Develop
Discover
Test
Sprint #3
Design
Develop

Agile vs Waterfall



Doing things differently

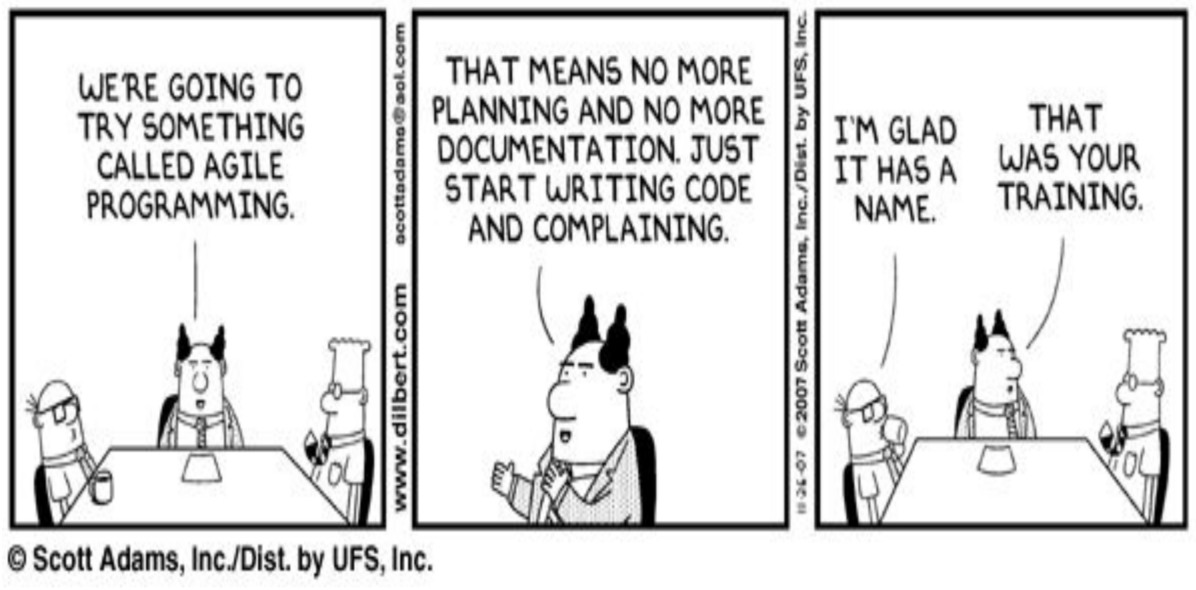
- Write less documentation
  - **but that *comes with a risk*!**
- Focus more on the final product (coding)!
- Get everyone involved… ***everyone***!!!
- Frequent delivery to customer .. ***continuously***!!
- ***People oriented*** methodology – great focus on individual ***(they write and use the software***!!)
- Use every opportunity to show the customer what you are doing .. And get feedback…
- Test you code, then retest it again.. ***Then do another round of testing***, and may QA will test it ***again***!!!
  –

But there are challenges….

- **Desire to collaborate**, and Sharing understanding
  - People might not be willing to collaborate and share!!
- **Ready to combine things together** (integrating different developers' work)
- **Keeping track** of the fast-changing progress
- **Maintaining** and **monitoring** schedules and **plans**

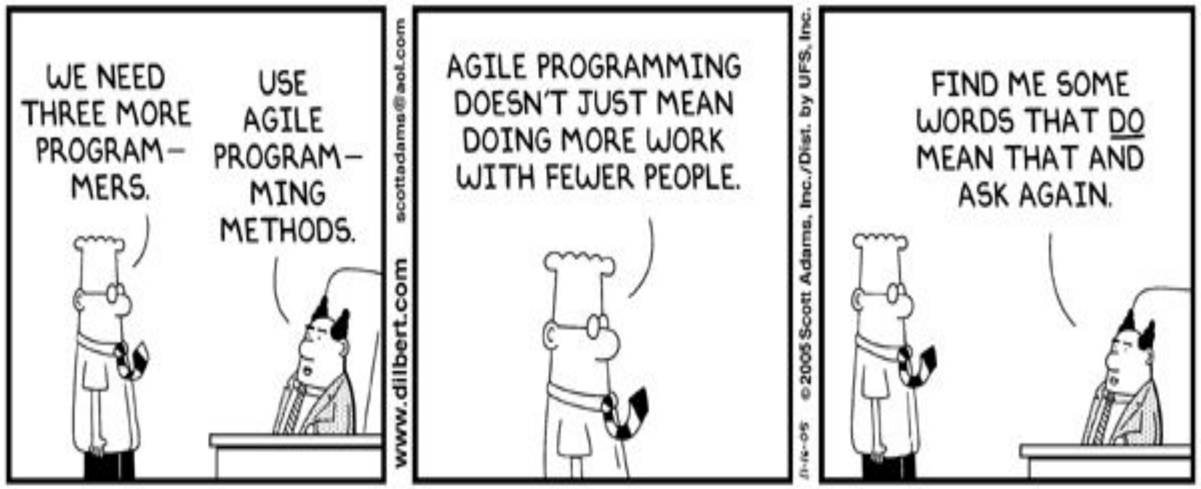
Misconceptions

Agile does not mean no planning or documentation

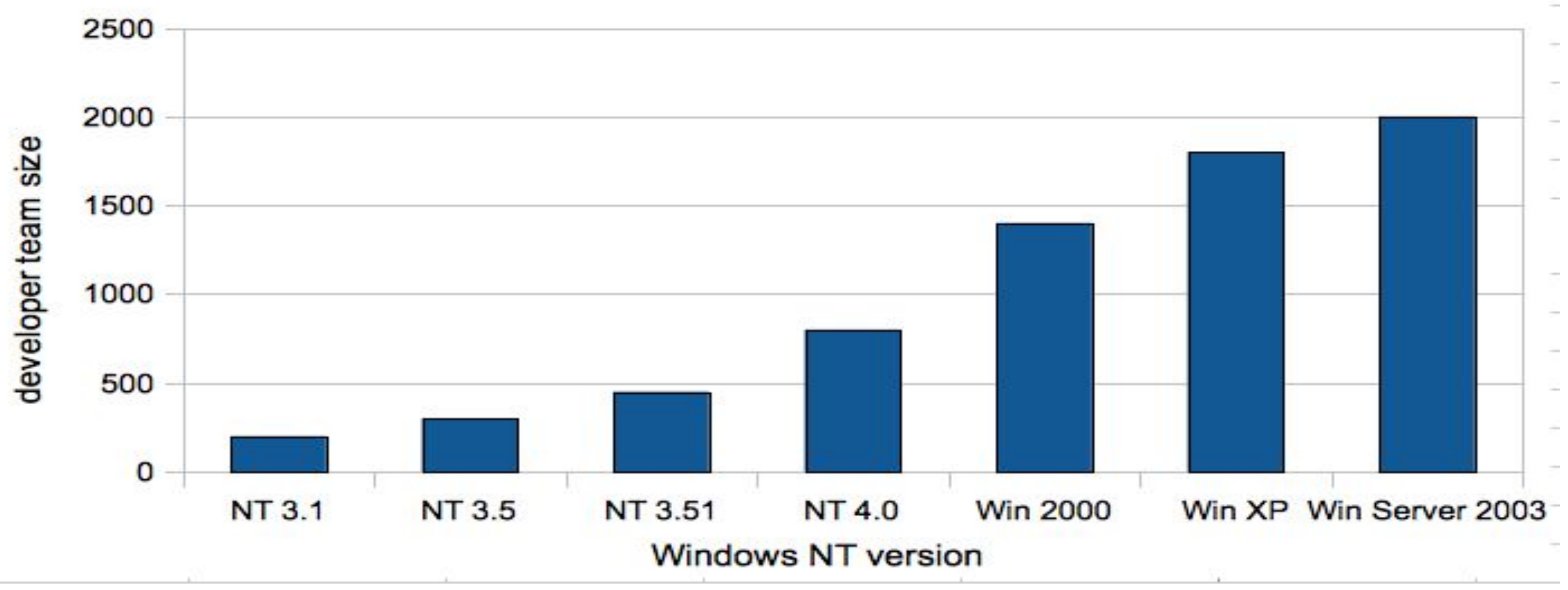

WE'RE GOING TO TRY SOMETHING CALLED AGILE PROGRAMMING.
THAT MEANS NO MORE PLANNING AND NO MORE DOCUMENTATION. JUST START WRITING CODE AND COMPLAINING.
I'M GLAD IT HAS A NAME.
THAT WAS YOUR TRAINING.
www.dilbert.com scottadams@aol.com
© 2007 Scott Adams, Inc./Dist. by UFS, Inc.
© Scott Adams, Inc./Dist. by UFS, Inc.

Misconceptions

and it does not mean less people to do the work


WE NEED THREE MORE PROGRAM-MERS.
USE AGILE PROGRAM-MING METHODS.
scottadams@aol.com
www.dilbert.com
AGILE PROGRAMMING DOESN'T JUST MEAN DOING MORE WORK WITH FEWER PEOPLE.
© 2005 Scott Adams, Inc./Dist. by UFS, Inc.
FIND ME SOME WORDS THAT DO MEAN THAT AND ASK AGAIN.

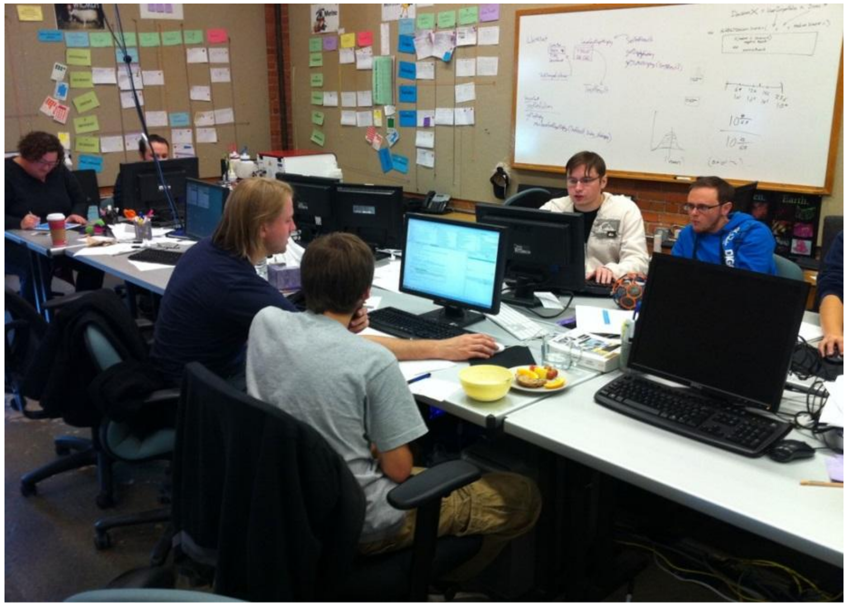
Increasing Team Sizes



- the size of Windows NT developer teams from 1993-2003
- source: V. Maraia: The Build Master: Microsoft's Software Configuration Management Best Practices. Addison-Wesley 2005.

Human aspects of software development

- Why is most software development collaborative in nature?
- What challenges does this add?



A typical Agile workspace

Source http://powersoftwo.agileinstitute.com/2012/08/a-recipe-for-agile-team-space.html

More of Agile ….

Agile examples….

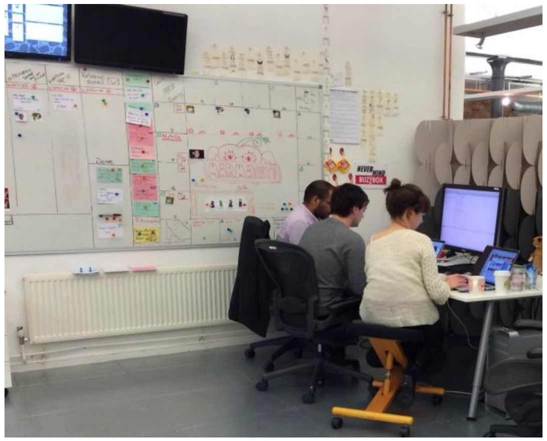
- **Several examples :**
  - ***Widely used***
    - eXtreme Programming (XP)
    - Scrum
    - Test Driven Development (TDD)
    - Lean and Kanban development…
  - ***Other methods***
    - Dynamic Systems Development Method (DSDM)
    - Feature-Driven Development (FDD)
    - Crystal Clear methodology

eXtreme Programming (XP)

- This method strongly believe that code is the most important part of the development.
- As the name implies, XP focus on *intensive* programming…
- Four main activities:
  - **Planning**: simple user stories, release schedule….
  - **Testing:** intensive unit tests, user acceptance test..
  - **Designing:** simple design, stand up meetings, move people around!
  - **Listening:** always listen to the client!

Unique features of XP

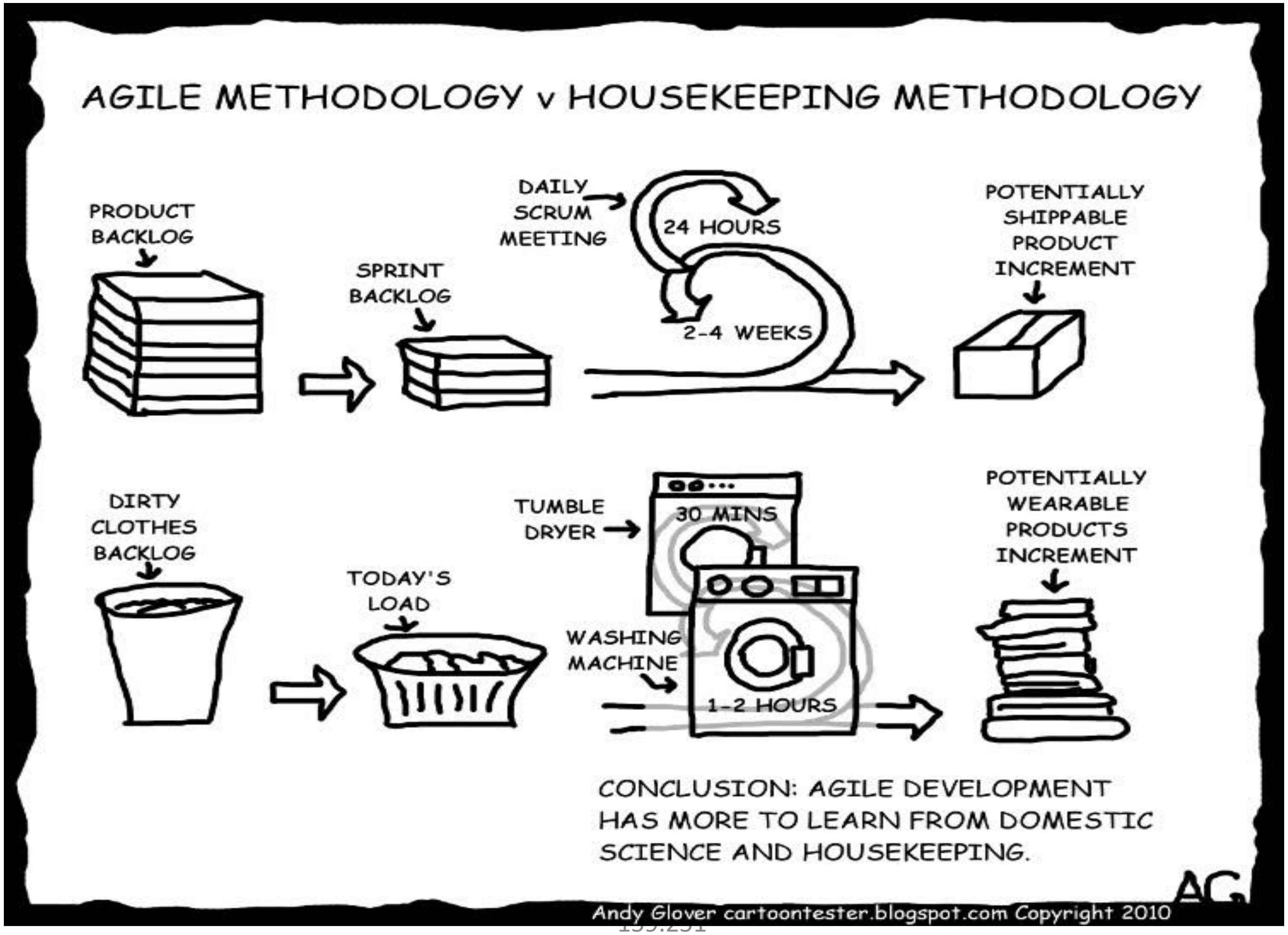
- Pair programming
- An open work place
- Stand up meetings
- Customers become part of the development team ⇒ sits together in the same room!
- Testing: pair testing and unit testing…
- Refactoring, whenever possible..



http://agilecoach.typepad.com/agile-coaching/2014/02/evolving-extreme-programming-practice.html

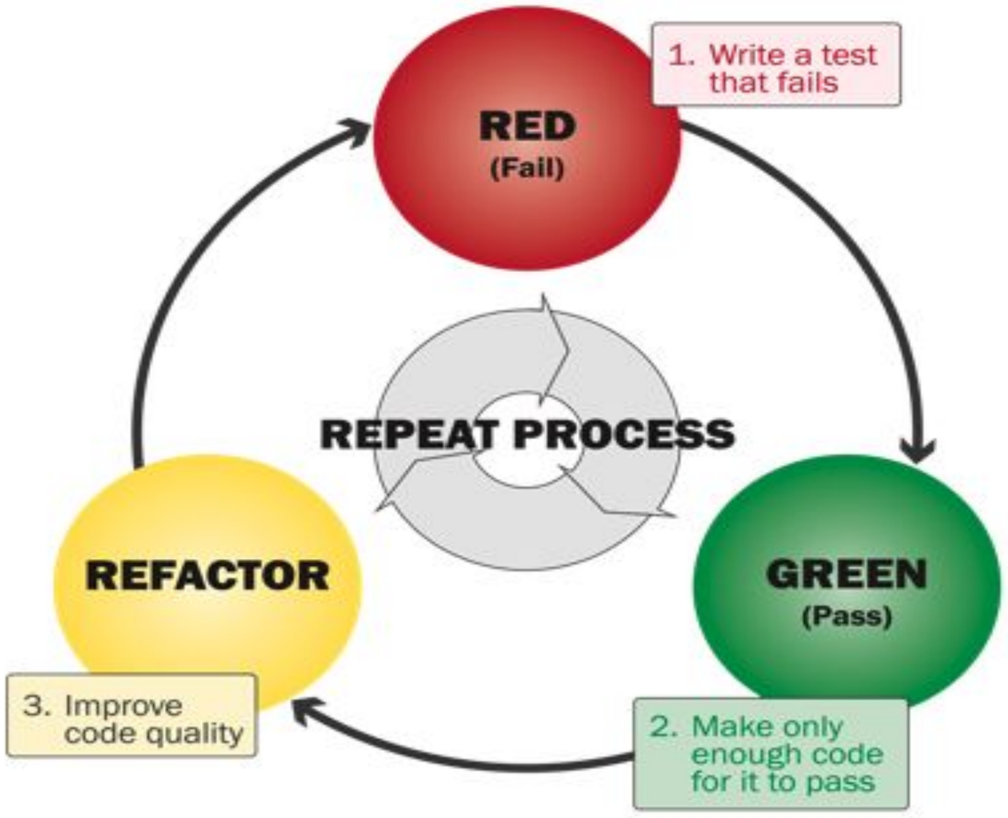
Scrum

- An iterative, human-oriented methodology.
- Promote communication between team members and customers.
- Responses to changes quickly…
- The development is led by a ***scrum master***, someone that can facilitate the workflow and remove impediments.
- Short iterative cycle (known as *Sprints*)


AGILE METHODOLOGY v HOUSEKEEPING METHODOLOGY
PRODUCT BACKLOG
SPRINT BACKLOG
DAILY SCRUM MEETING
24 HOURS
2-4 WEEKS
POTENTIALLY SHIPPABLE PRODUCT INCREMENT
DIRTY CLOTHES BACKLOG
TODAY'S LOAD
TUMBLE DRYER
30 MINS
WASHING MACHINE
1-2 HOURS
POTENTIALLY WEARABLE PRODUCTS INCREMENT
CONCLUSION: AGILE DEVELOPMENT HAS MORE TO LEARN FROM DOMESTIC SCIENCE AND HOUSEKEEPING.
AG
Andy Glover cartoontester.blogspot.com Copyright 2010

Test Driven Development (TDD)

- A testing-oriented methodology..
- The development start from testing, and centred around testing!
  - Other methods follow test-last approach
    - You first write your code and then you write ***tests*** to check your code (whether your code passes the test). If not then you go back and fix your code until it passes the test.
  - TDD follow a test-first approach
    - You write your ***tests*** first then you write code to pass the tests. The code should fail first (obviously!). Then try to write a code that passes the test – keep ***refactoring*** until the code passes the test.
- More on *Software Testing* and *Code Refactoring* later in this course (second part of the course!)



RED
(Fail)
1. Write a test that fails
GREEN
(Pass)
2. Make only enough code for it to pass
REFACTOR
3. Improve code quality
REPEAT PROCESS

https://github.com/mjhea0/flaskr-tdd

Lean development

- *In manufacturing:* **lean** is a systematic method for the ***elimination of waste*** within a manufacturing system.
- Originally created in Toyota Production System.
- Similar principles have been adopted in SE.
- Key principles include:
  - **Eliminate waste:** everything not adding value to the customer is waste – e.g., extra processes and extra features…
  - **Deliver Fast:** ship the product to the customer as soon as it is ready… just like in just-in-time (JIT) strategy.
  - **Empower the team:** "find good people and let them do their own job"
  - **See the whole:** everyone should fully understand the method
    - "Think big, act small, fail fast; learn rapidly"

Kanban

- **Kanban** is another engineering method for managing workflow with an emphasis on JIT delivery while not overloading the team members.
- Kanban is based on a very simple principle - agree on a limit to work-in-process (WIP)
- Pull new tasks from the queue only when something is finished.
- Kanban means signboard or billboard in Japanese.
- Also used widely in Toyota to manage their supply chain during production.
- Another form of lean software development.
- Follow the JIT philosophy…

Why Kanban?

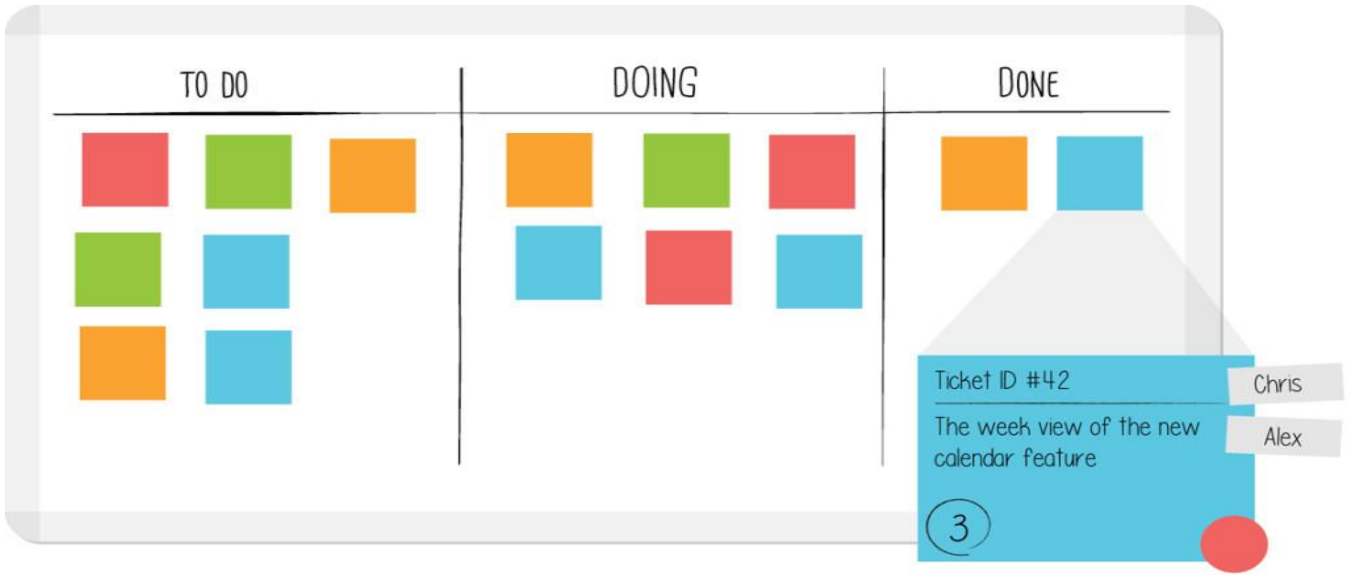
- Several issues with heavy-WIP.
- You may end up:
  - Building features are not needed right now! (e,g., Feature creep)
  - Writing more specifications than the team can code!
  - Writing more code than the team can tests!
  - Testing more code than the team can deploy!

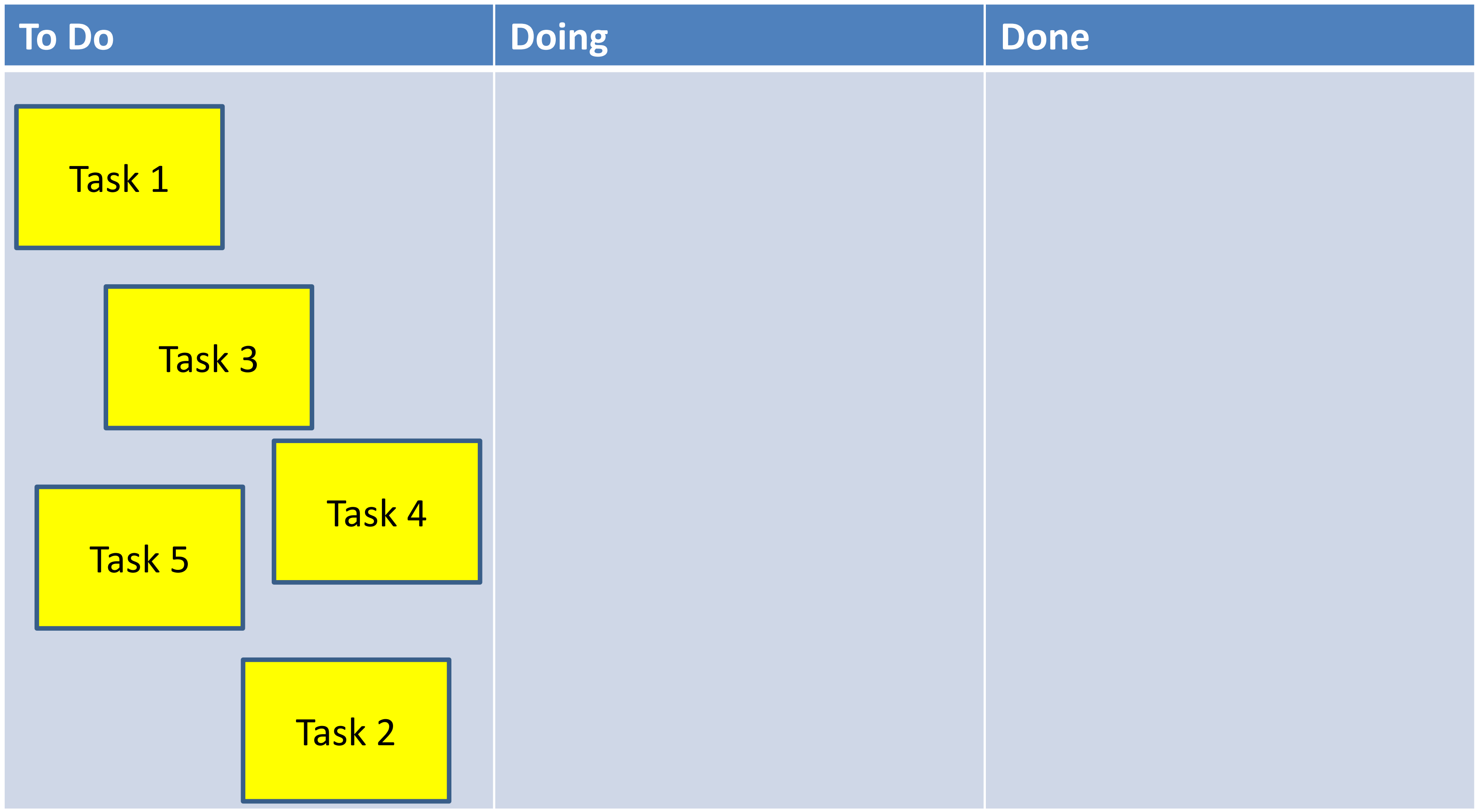
Kanban board

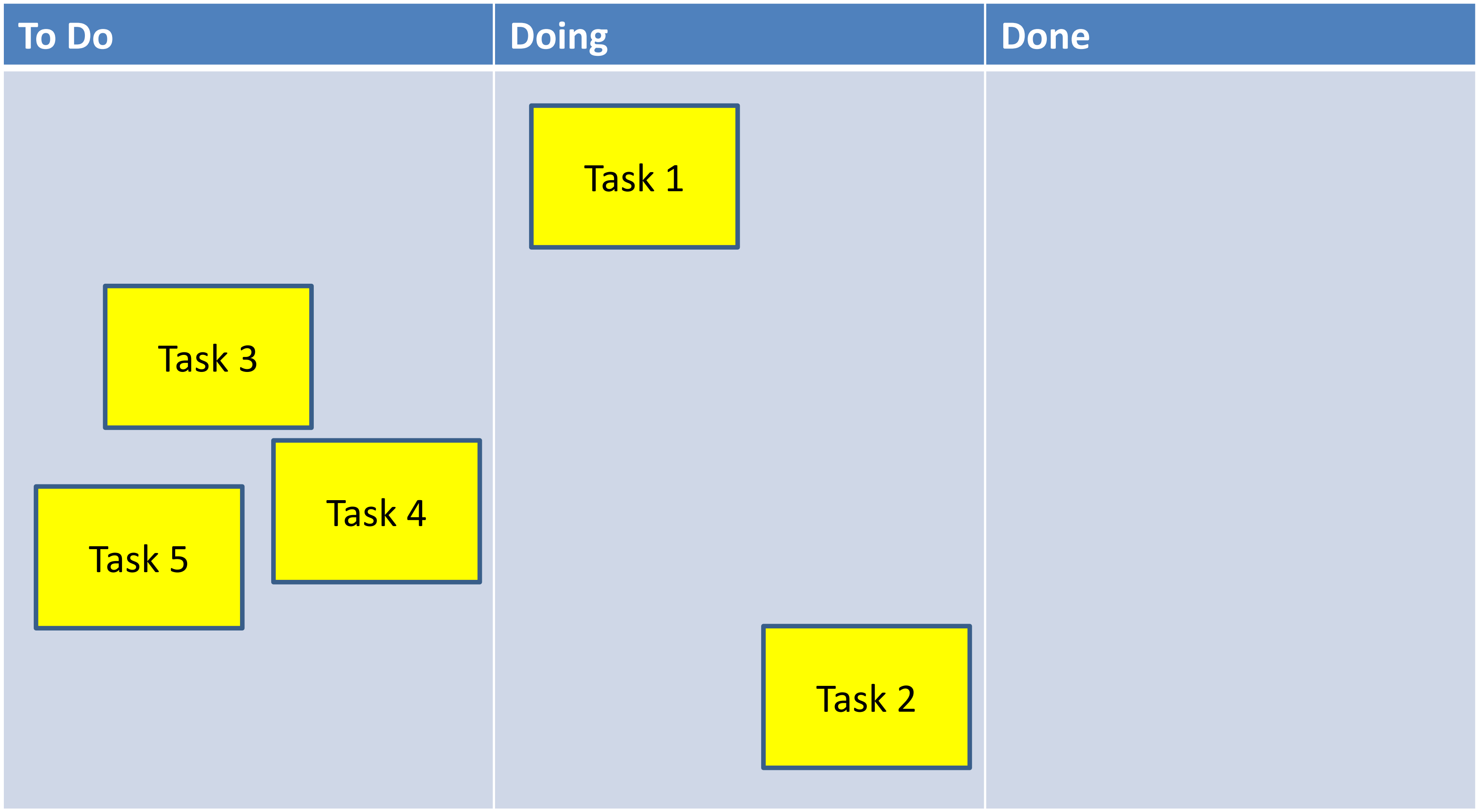
- A ***Kanban board*** is a work and workflow visualisation tool that enables you to optimize the flow of your work.
- Traditionally uses *a physical board* (e.g. whiteboard)
- *Virtual* (online) boards are widely used …
- Shows what a real-time workflow.
- Can be used to monitor daily, weekly or monthly progress…
- Multiple boards can be used as well…

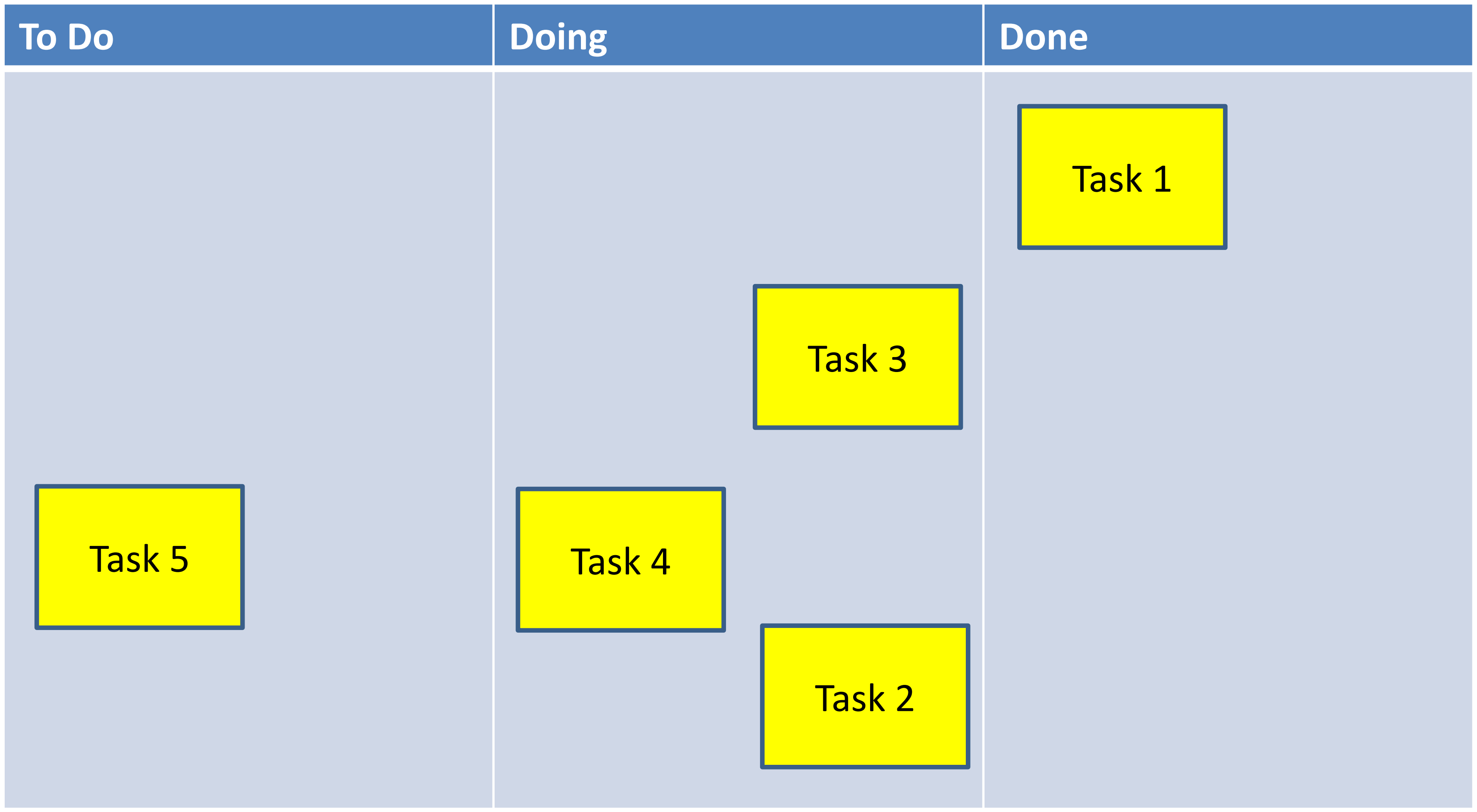
Kanban concept is quite simple….

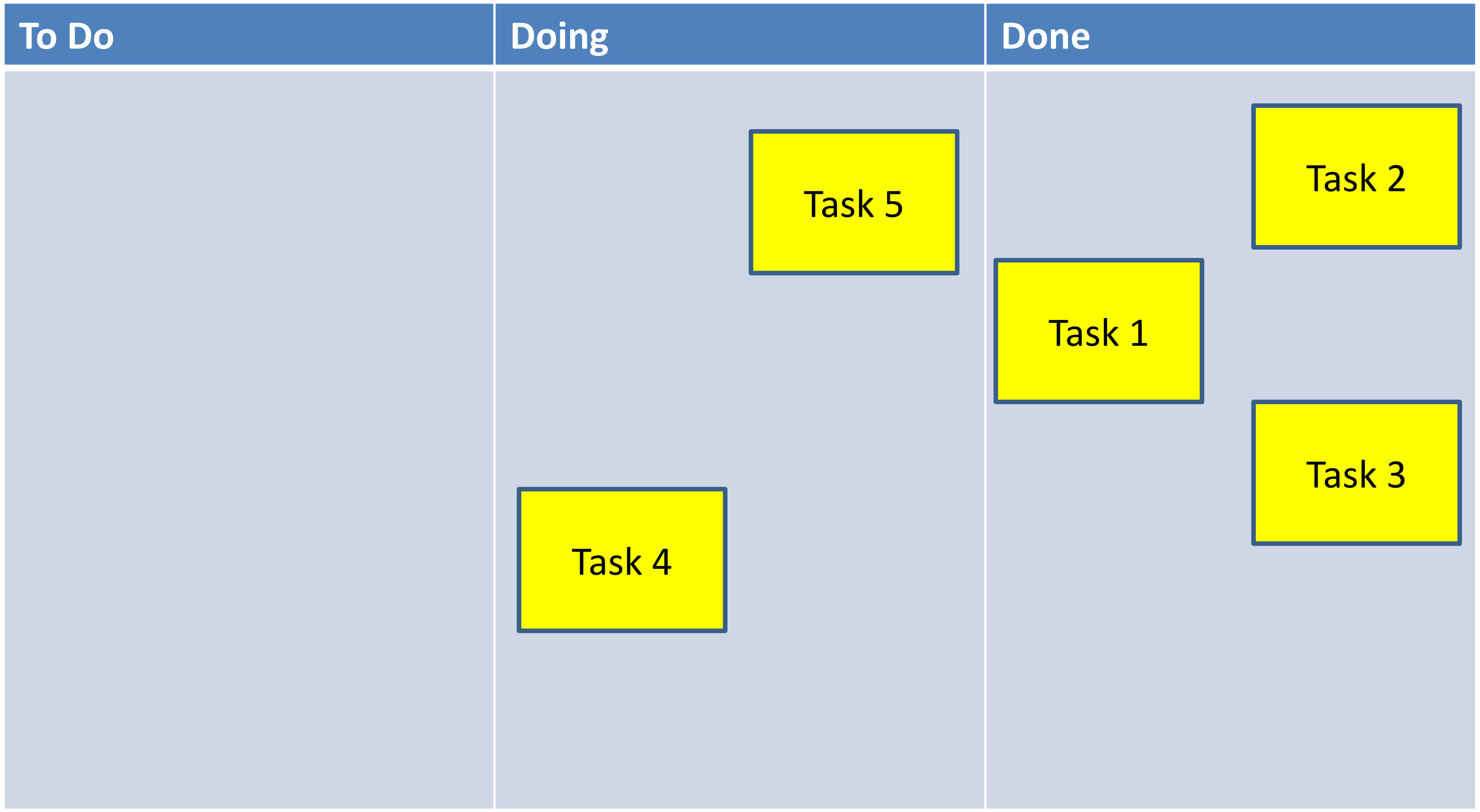
- Create a simple board with few columns (preferably 3).
- Write tasks on sticky notes and keep moving them as the work progresses.

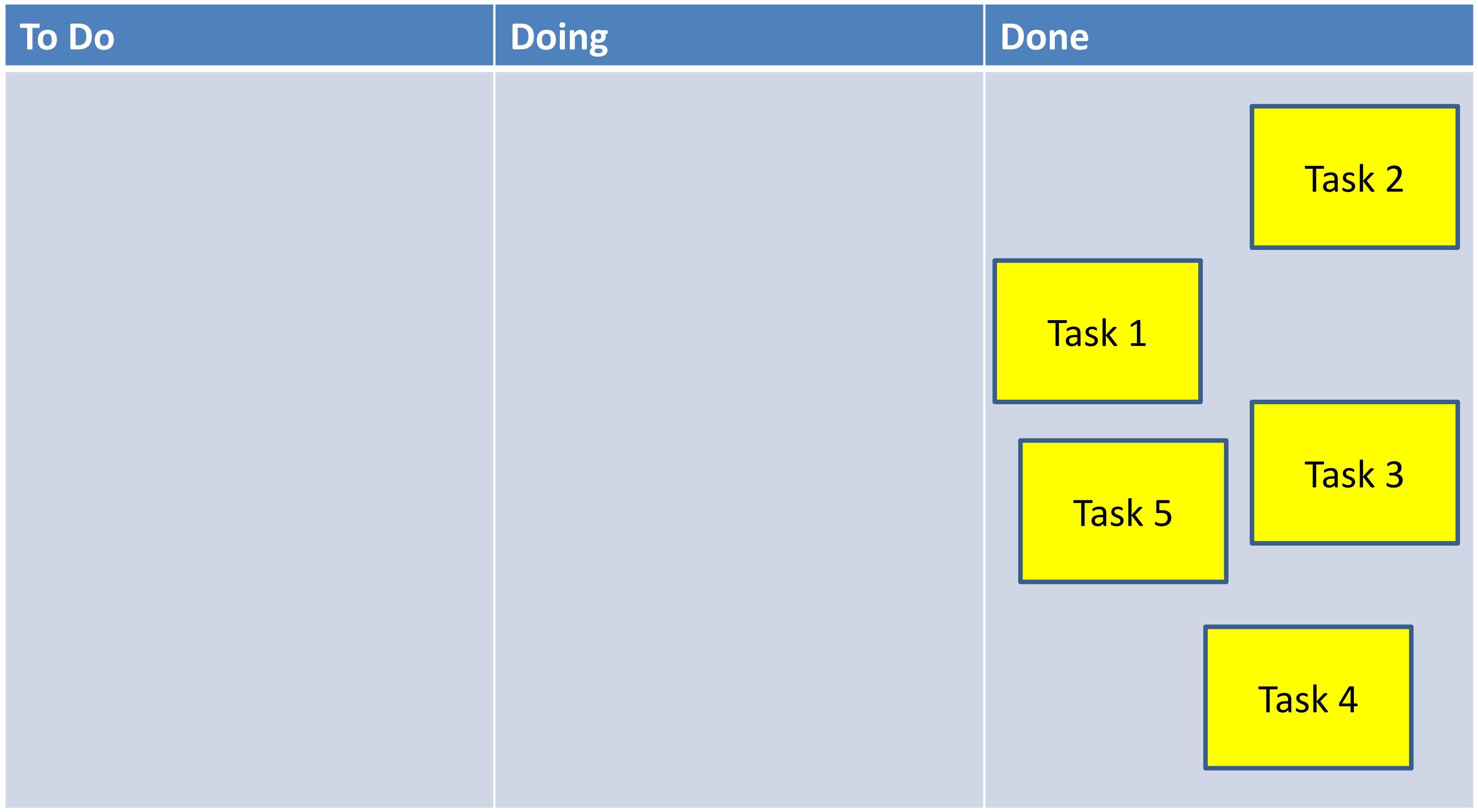



To Do
Doing
Done
Task 1
Task 3
Task 4
Task 5
Task 2

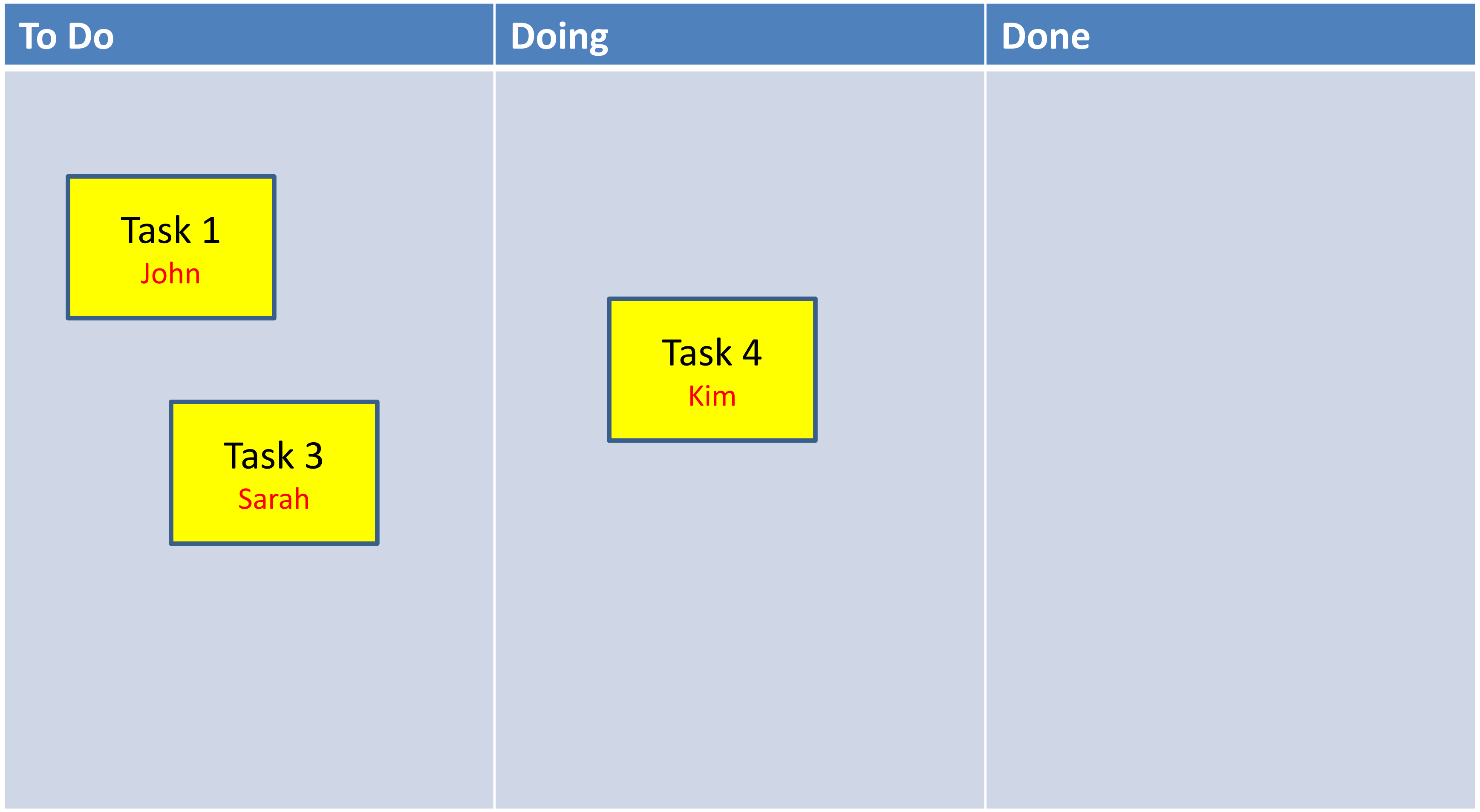

To Do
Doing
Done
Task 1
Task 3
Task 4
Task 5
Task 2


To Do
Doing
Done
Task 1
Task 3
Task 5
Task 4
Task 2

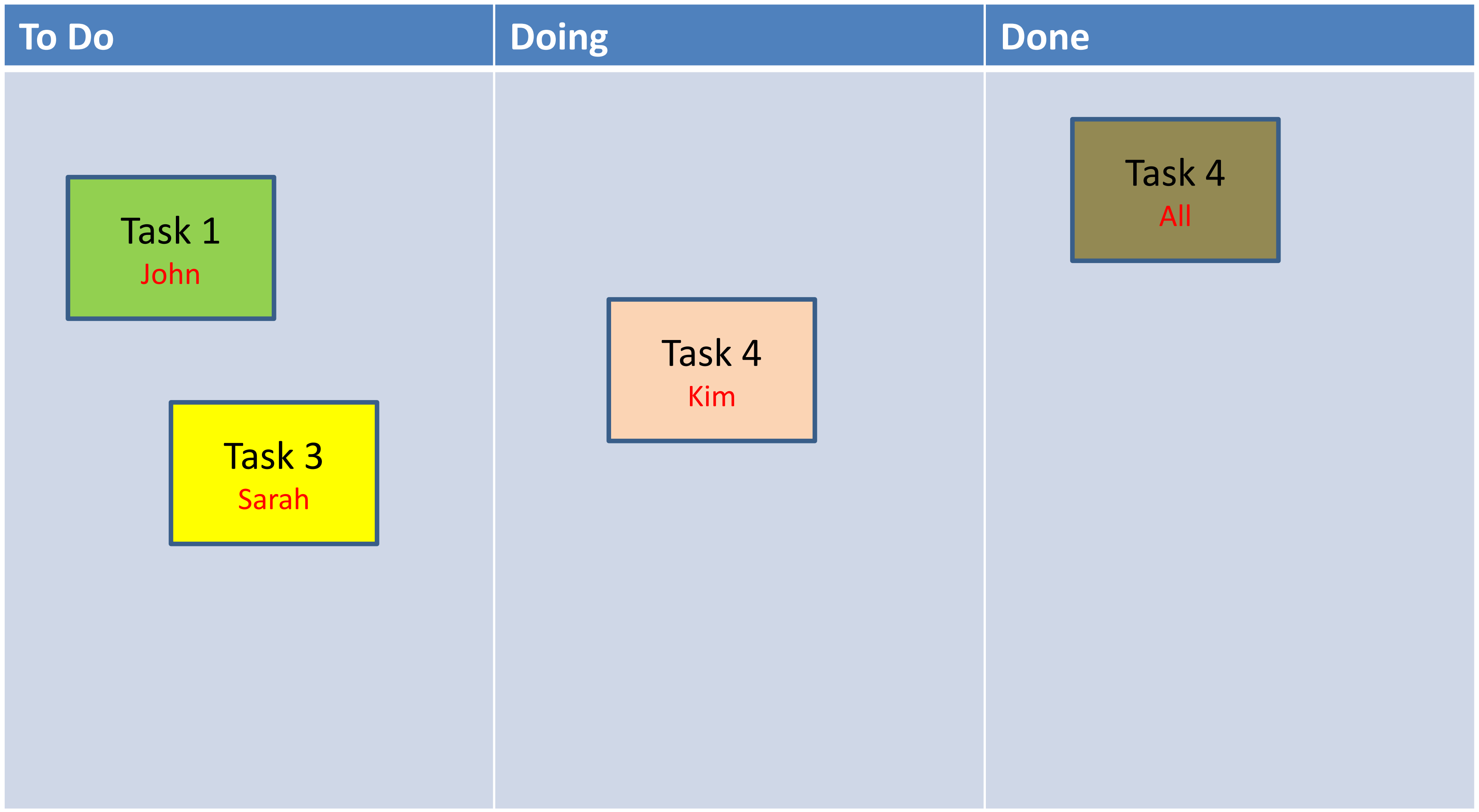

To Do
Doing
Done
Task 5
Task 2
Task 1
Task 3
Task 4


To Do
Doing
Done
Task 2
Task 1
Task 5
Task 3
Task 4

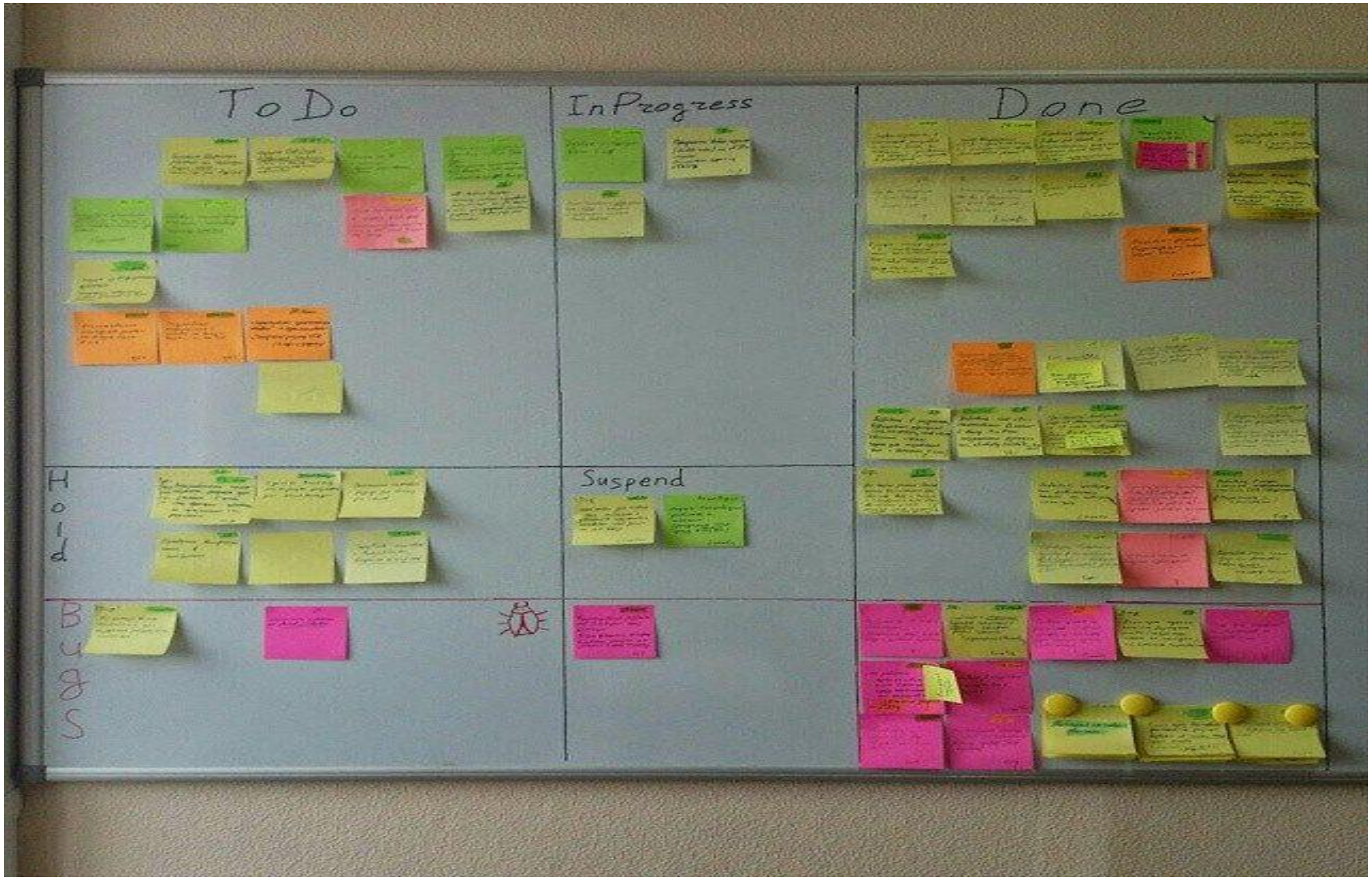
You can also assign names



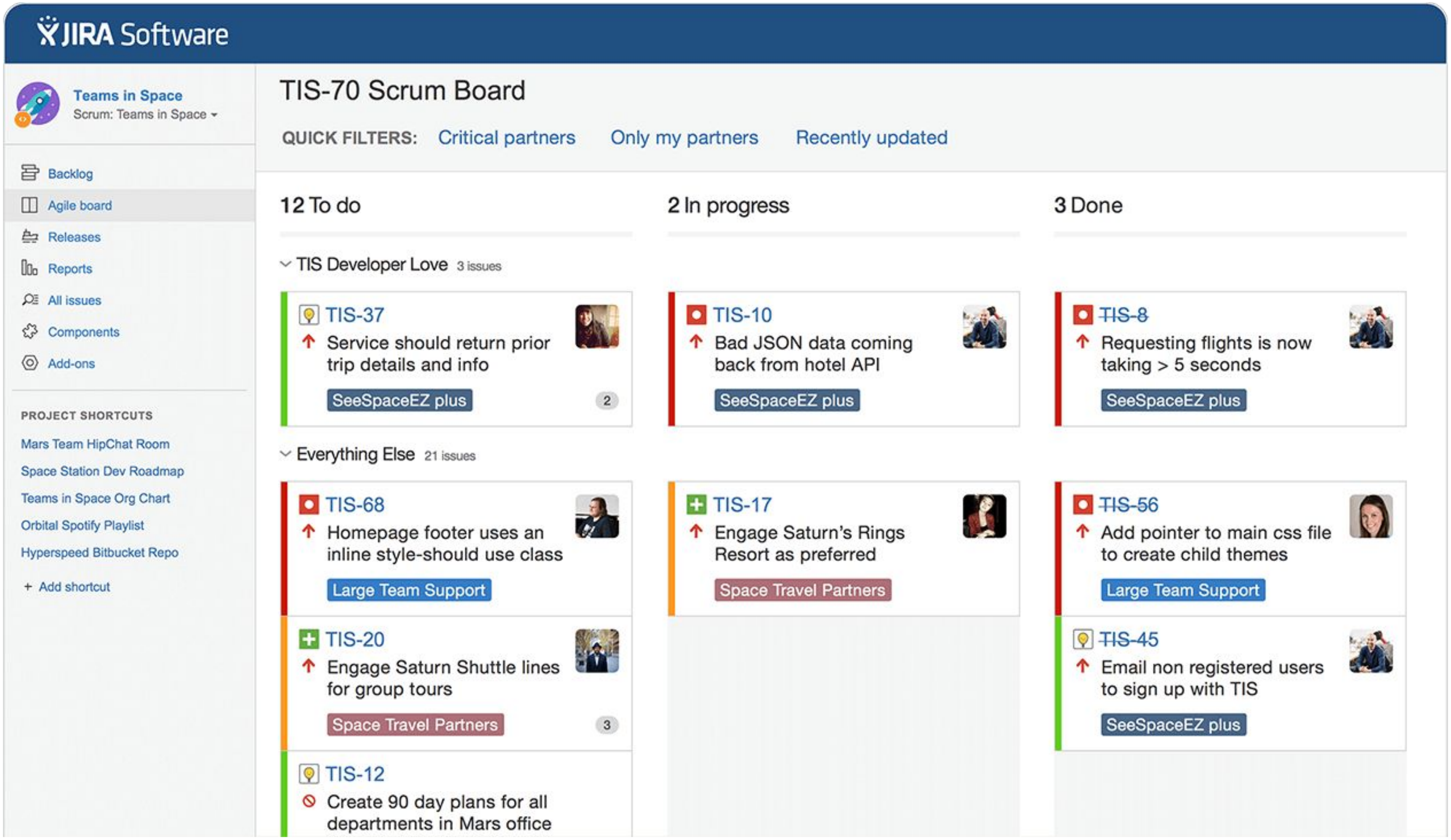
And colours



A typical Kanban board at a workplace



Kanban board on JIRA



JIRA Software

Teams in Space
Scrum: Teams in Space

- Backlog
- Agile board
- Releases
- Reports
- All issues
- Components
- Add-ons

PROJECT SHORTCUTS

- Mars Team HipChat Room
- Space Station Dev Roadmap
- Teams in Space Org Chart
- Orbital Spotify Playlist
- Hyperspeed Bitbucket Repo
- + Add shortcut

TIS-70 Scrum Board

QUICK FILTERS: Critical partners | Only my partners | Recently updated

12 To do	2 In progress	3 Done
TIS Developer Love 3 issues		
TIS-37 Service should return prior trip details and info — SeeSpaceEZ plus — 2	TIS-10 Bad JSON data coming back from hotel API — SeeSpaceEZ plus	~~TIS-8~~ Requesting flights is now taking > 5 seconds — SeeSpaceEZ plus
Everything Else 21 issues		
TIS-68 Homepage footer uses an inline style-should use class — Large Team Support	TIS-17 Engage Saturn's Rings Resort as preferred — Space Travel Partners	~~TIS-56~~ Add pointer to main css file to create child themes — Large Team Support
TIS-20 Engage Saturn Shuttle lines for group tours — Space Travel Partners — 3		~~TIS-45~~ Email non registered users to sign up with TIS — SeeSpaceEZ plus
TIS-12 Create 90 day plans for all departments in Mars office		

What is the difference between those Agile methods?

- Because it’s flexible, it allows you to do things in your own way…
- You can also create your method, but it has to be *Agile* and follow *Agile* principle.
- Combine techniques that works best for your project, team and organization

Now, let’s think ***Agile***!