

Software Project Management – IS212

Software Development Processes

Software Development Processes

Objectives

On completing this module, you will be able to:

- Summarise the key **traditional** and **agile** software development processes
- Identify the **pros** and **cons** of each

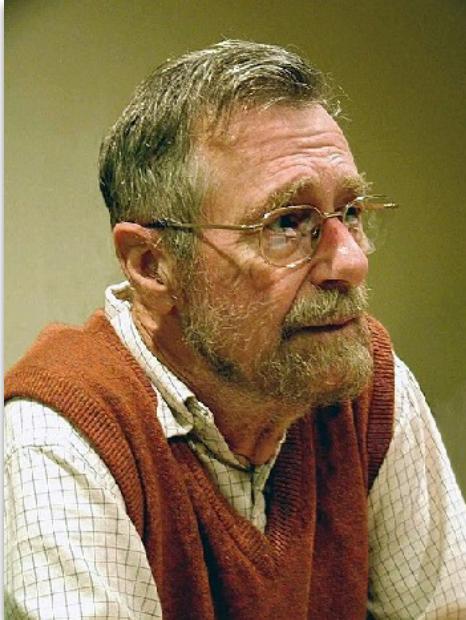
Traditional

- Waterfall
- RUP

Agile

- Extreme Programming
- Kanban
- Scrum

1960/70s – Software Crisis



The major cause of the software crisis is that the machines have become several orders of magnitude more powerful!

To put it quite bluntly: as long as there were no machines, programming was no problem at all; when we had a few weak computers, programming became a mild problem, and now we have gigantic computers, programming has become an equally gigantic problem.

– Edsger Dijkstra

over budget

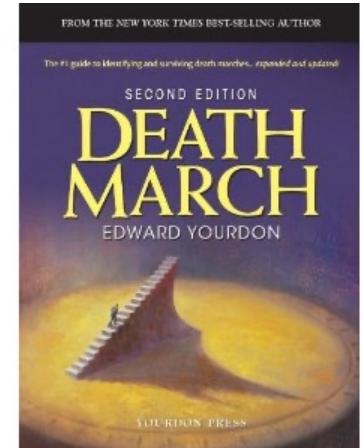
low quality

difficult to maintain

inefficient

Computing power was outpacing the ability of software developers to effectively use it

often didn't meet requirements





*Software requirements rarely
'frozen' before development*

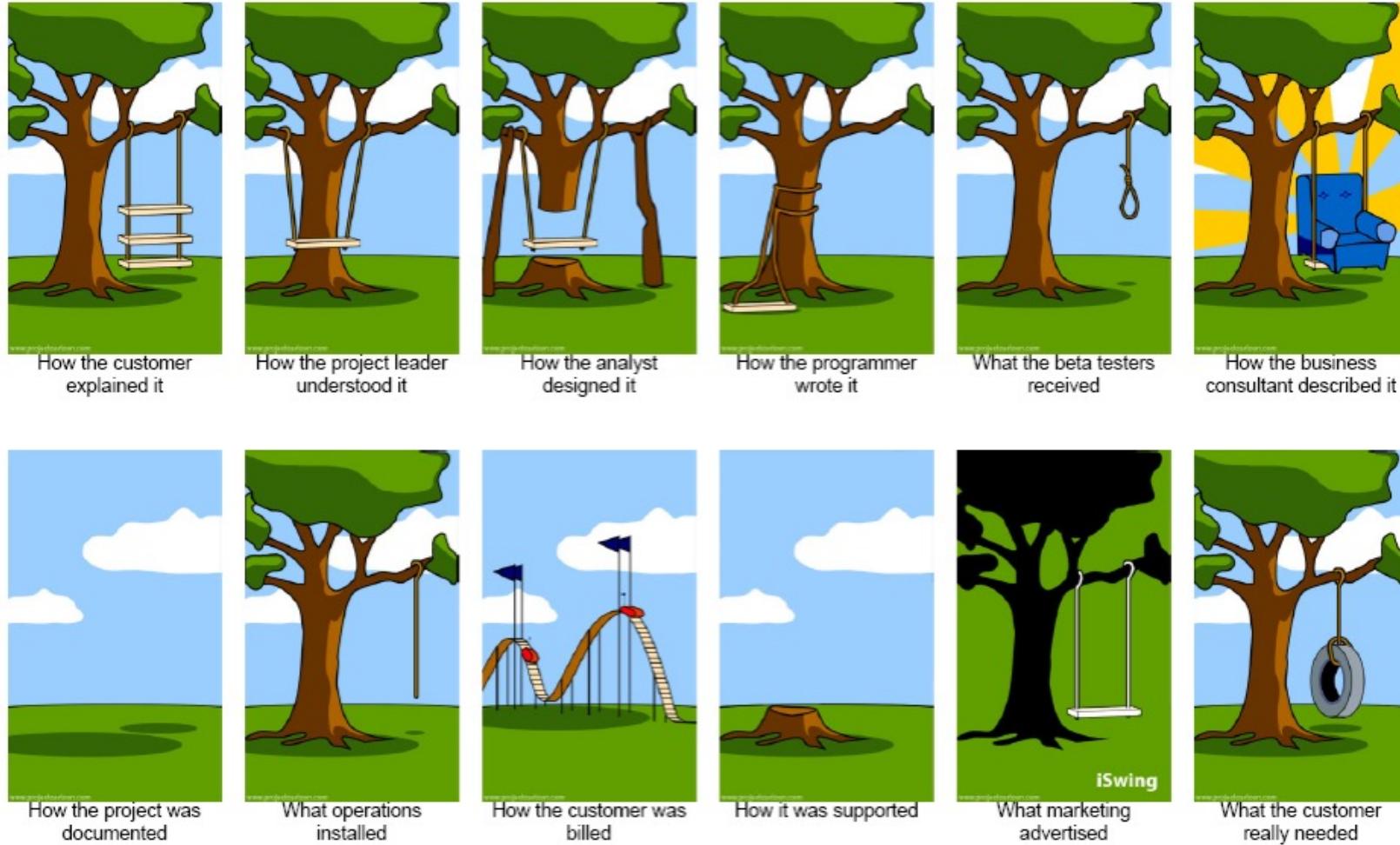
Software is subject to (frequent) change

Engineering **software** vs.
engineering, say, a **bridge**

Software and Work In Progress (WIP) are “invisible”

Maintenance & technical debt are major challenges

A classic comic...



Software Development Processes

Traditional

- Waterfall
- RUP

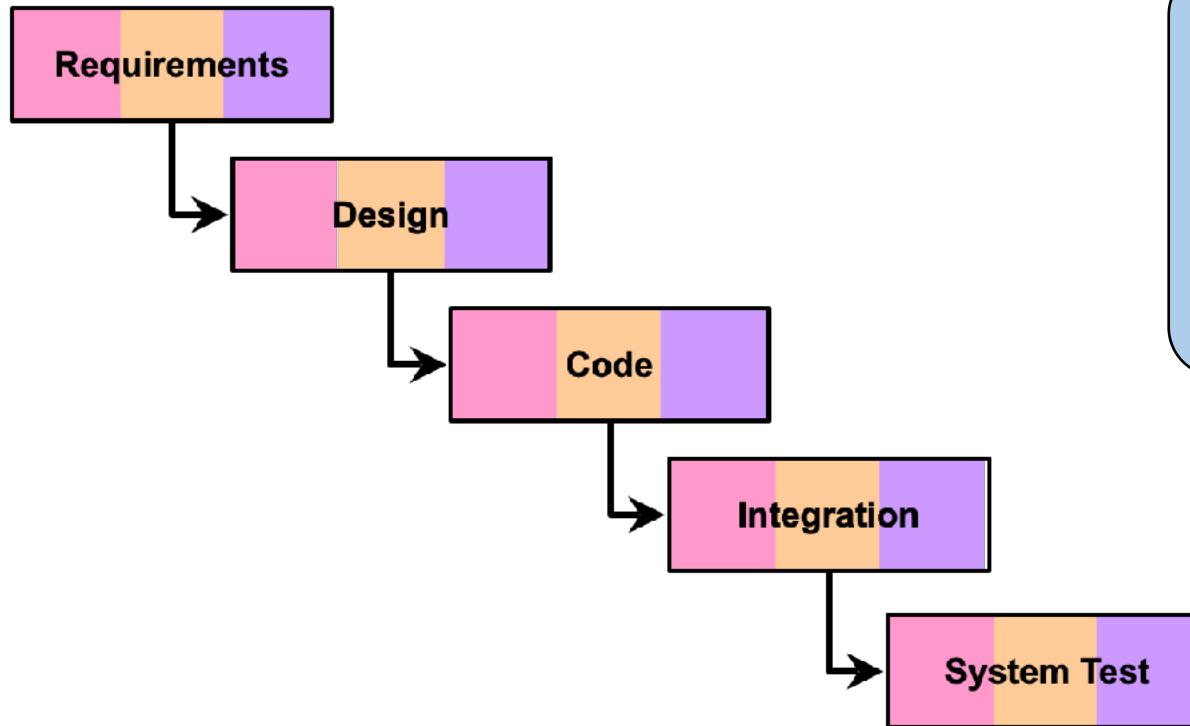
Agile

- Extreme Programming
- Kanban
- Scrum



Waterfall

Idea: break software development into linear, sequential phases



Padlet Exercise

What are some **pros** and **cons** of this process?

<https://padlet.com/XXX/YYY>

Waterfall – Discussion



Many software projects are **iterative**



Software Development Methods

Traditional

iterative traditional
method →

- Waterfall
- RUP

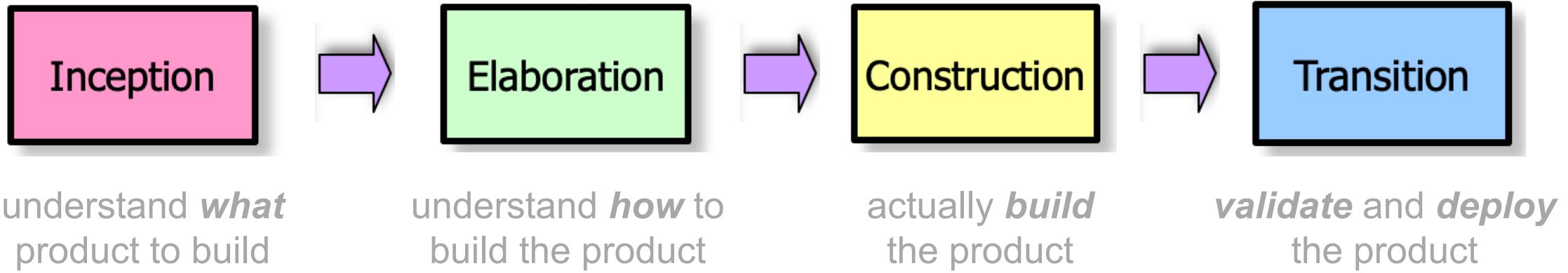
Agile

- Extreme Programming
- Kanban
- Scrum

Rational Unified Process (RUP)



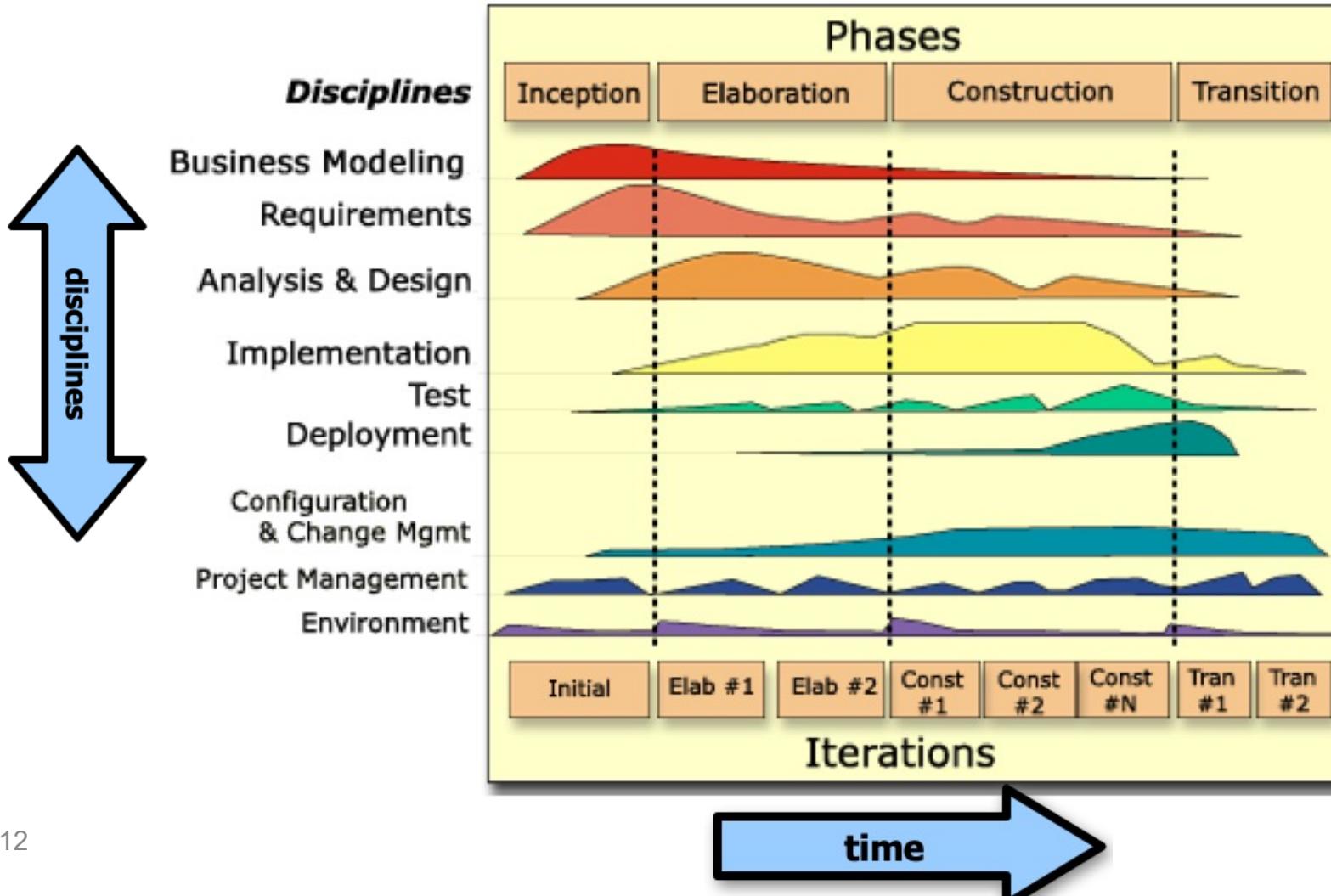
- The Rational Unified Process (RUP) consists of four phases
 - i.e. key 'milestones' for the project, similar to waterfall



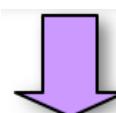
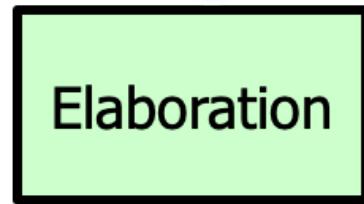
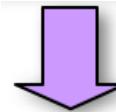
- Various 'engineering disciplines' then cut across all four phases
 - e.g. *business modelling, requirements, implementation, testing*

Rational Unified Process (RUP)

- Unlike waterfall – it **explicitly encourages iterations** within phases

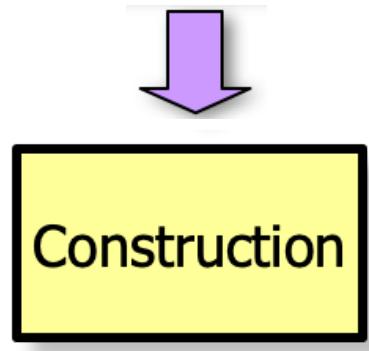


RUP phases – cross-discipline outcomes

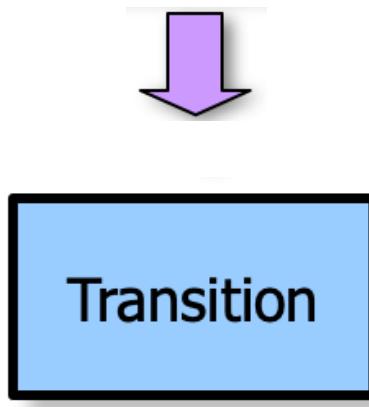


- Vision document / business case
 - Develop **high-level** requirements / **use cases**
 - Stakeholder concurrence; identify risks
-
- “80% complete” use case models
 - Prototypes for exploring key risks
 - Development plan; architecture description

RUP phases – cross-discipline outcomes



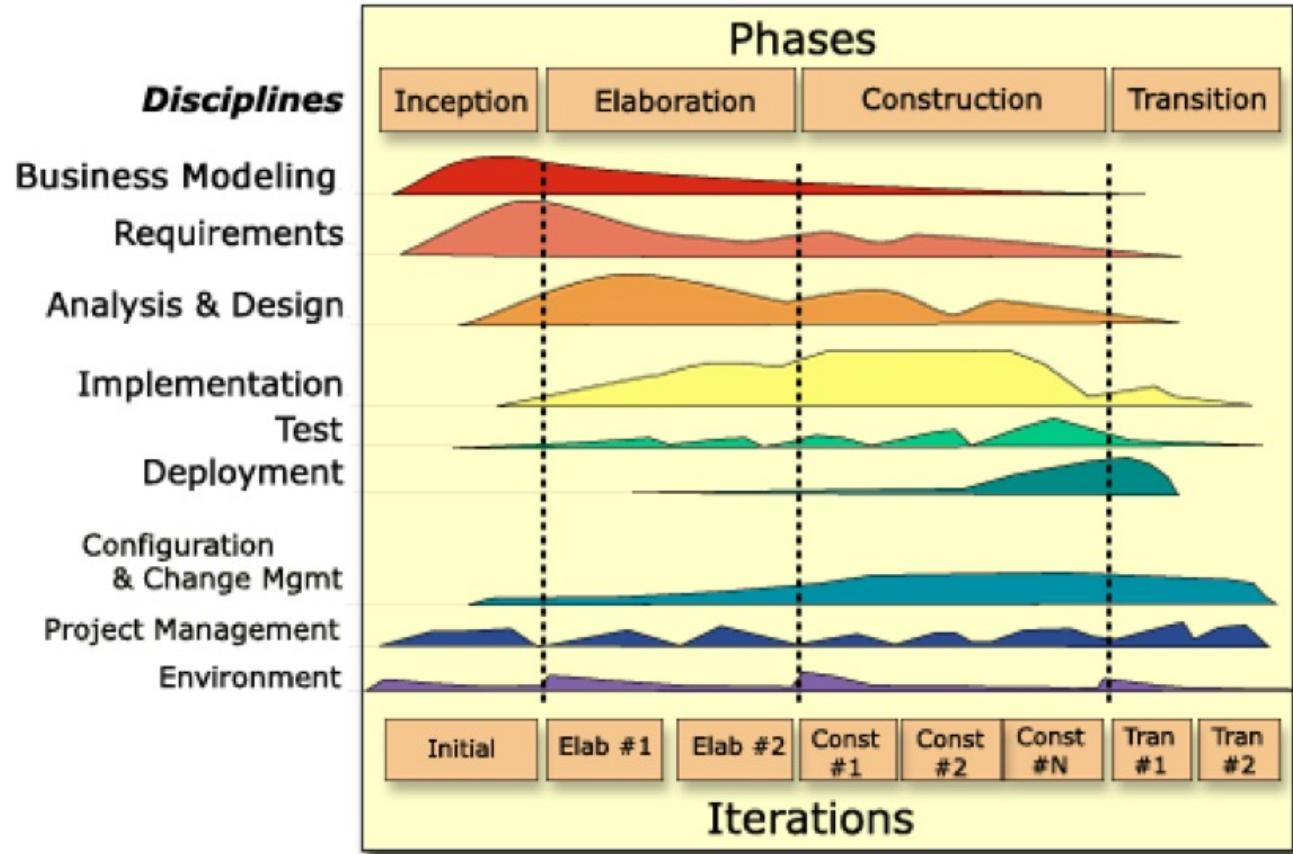
- Demonstrable prototypes
- Working software **components**
- Bulk of coding



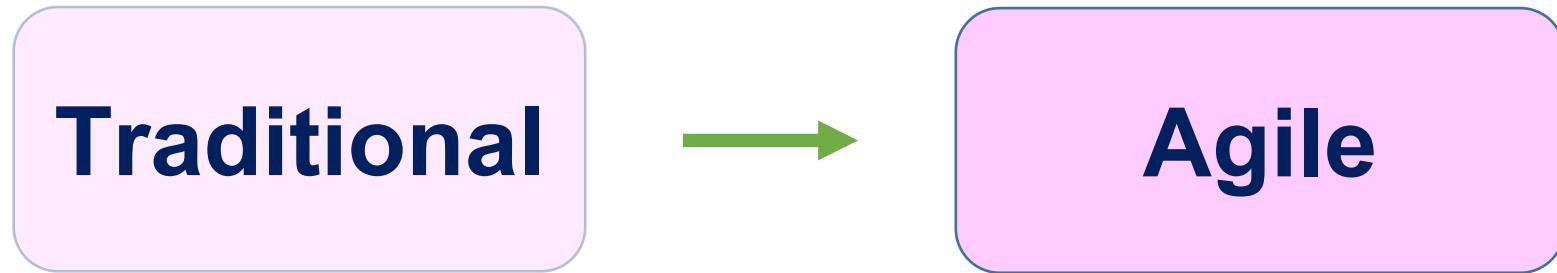
- Training end users / maintainers
- Evaluate against ‘Inception’ requirements
- ‘Post-mortem’ project analysis

Discussion: what's your assessment of the RUP method?

- advantages over waterfall?
- any possible drawbacks?



Software Development Methods

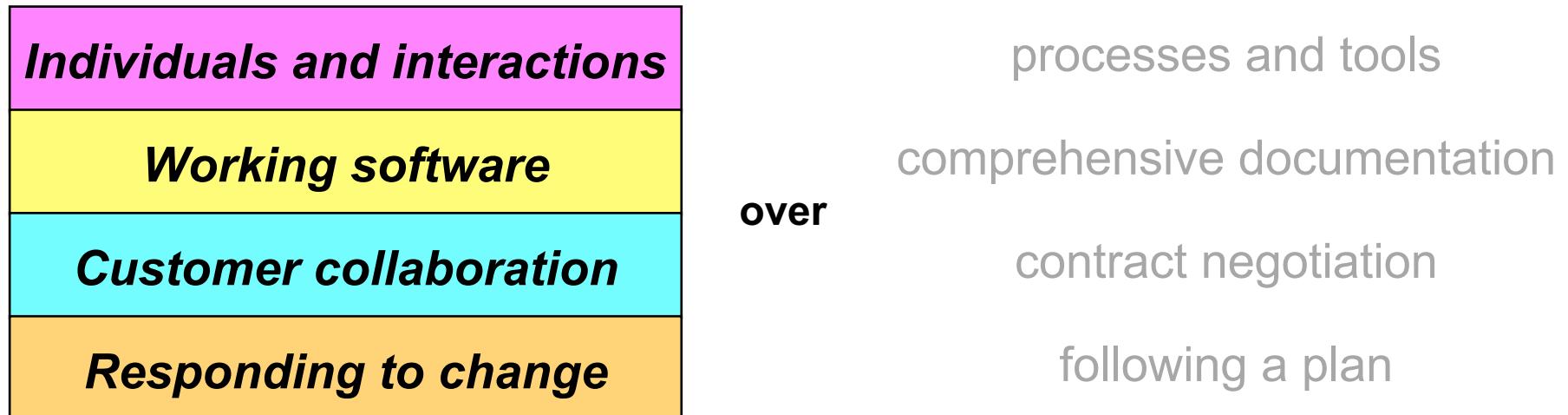


- Waterfall
- RUP

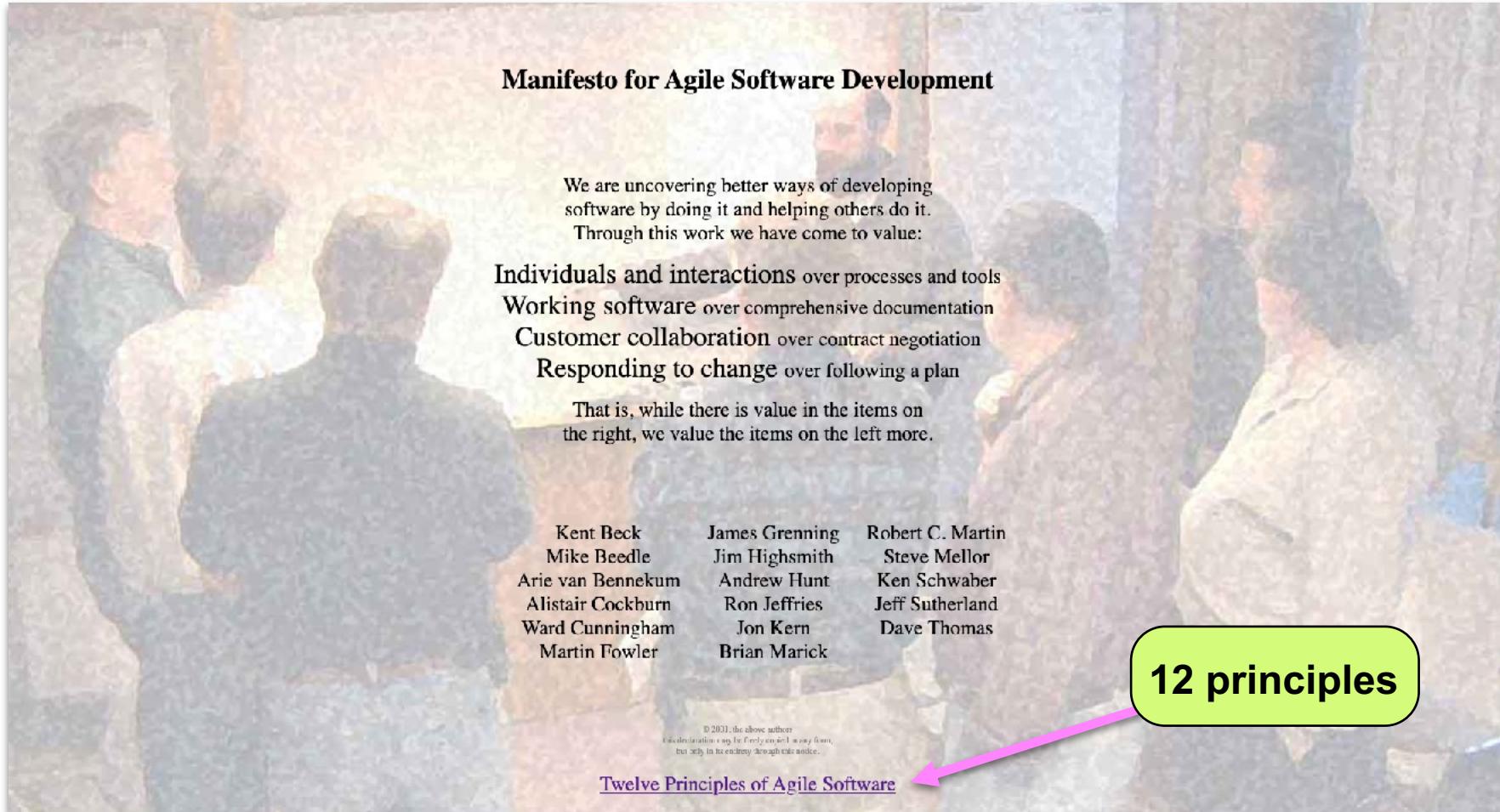
- Extreme Programming
- Kanban
- Scrum

Agile – Motivation

- Traditional heavyweight methods can be overly regulated, planned, and micromanaged, i.e. **bureaucratic**
- In reaction, a number of **lightweight methods** started evolving that collectively became known as “**agile**”, emphasising:



Agile Manifesto (2001)



<https://agilemanifesto.org/>

Agile Principles (*re-organised; Meyer '14*)

Agile principles

Organizational

- 1 Put the customer at the center.
- 2 Let the team self-organize.
- 3 Work at a sustainable pace.
- 4 Develop minimal software:
 - 4.1 Produce minimal functionality.
 - 4.2 Produce only the product requested.
 - 4.3 Develop only code and tests.
- 5 Accept change.

Technical

- 6 Develop iteratively:
 - 6.1 Produce frequent working iterations.
 - 6.2 Freeze requirements during iterations.
- 7 Treat tests as a key resource:
 - 7.1 Do not start any new development until all tests pass.
 - 7.2 Test first.
- 8 Express requirements through scenarios.

key insight



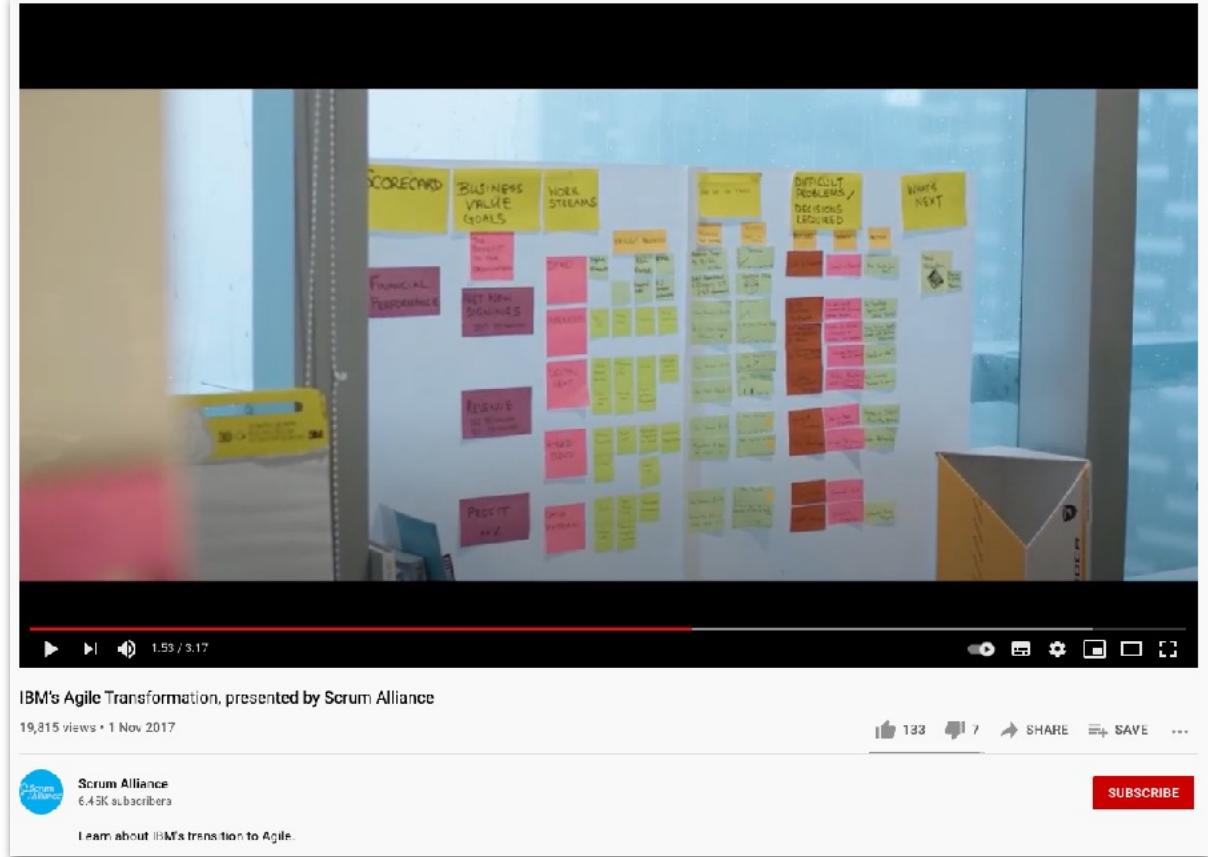
Bertrand Meyer

Agile – IBM Video

Padlet Exercise

What agile tools and methods do you see IBM using?

<https://padlet.com/XXX/YYY>



<https://www.youtube.com/watch?v=Xu0nxyebc6g>

Highly iterative and flexible
Working software delivered frequently

Working software (not “WIP”) is the measure of progress

Agile – Key Characteristics

Late changes in requirements – no problem

Close, daily co-operation

Visualise activities and “WIP” to manage team’s capacity

Software Development Methods

Traditional

- Waterfall
- RUP

Agile

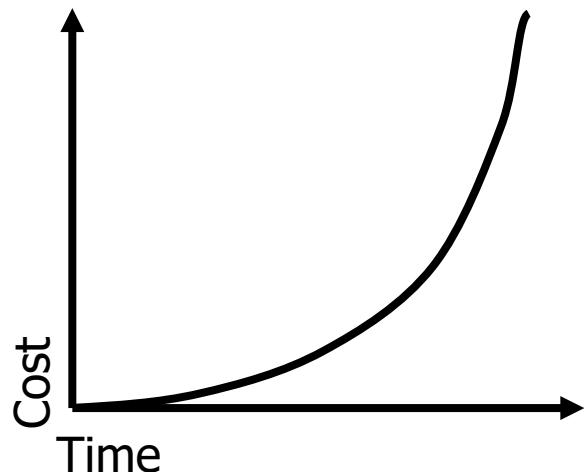
- Extreme Programming
- Kanban
- Scrum

Extreme Programming (XP)

- Premise of extreme programming (XP)

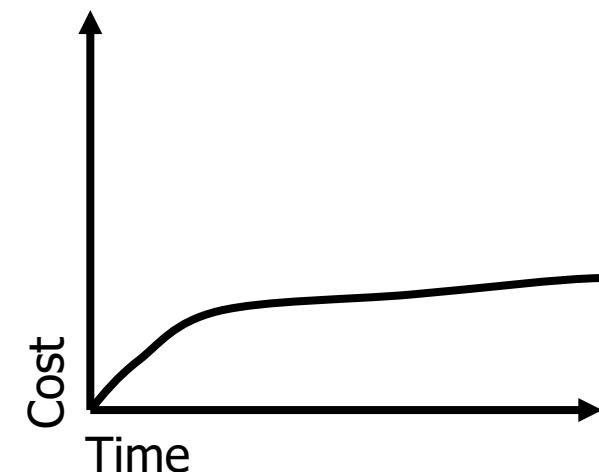
Traditional

- upfront design
- code late
- release when 'done'



XP

- code early
- release early
- continuous design



Extreme Programming (XP)

- Focus – customer satisfaction
- XP Principles –

Coding is the core activity

Code doesn't only deliver the solution
– use it to explore / explain problems too!

LOTS of testing during dev

Design unit tests or software contracts first
– *then code!* (aka **Test-Driven Development**)

Developers and customers communicate directly

Programmer must understand the business requirements to design a technical solution

Regular refactoring

Overall system design is considered during regular **refactoring** exercises

Fine-scale feedback

- Pair programming
- Planning game
- Test-Driven Development
- Whole team

Shared understanding

- Coding standard
- Collective code ownership
- Simple design
- System metaphor

12 XP Practices

Continuous process

- Continuous integration
- Design refactoring
- Small releases

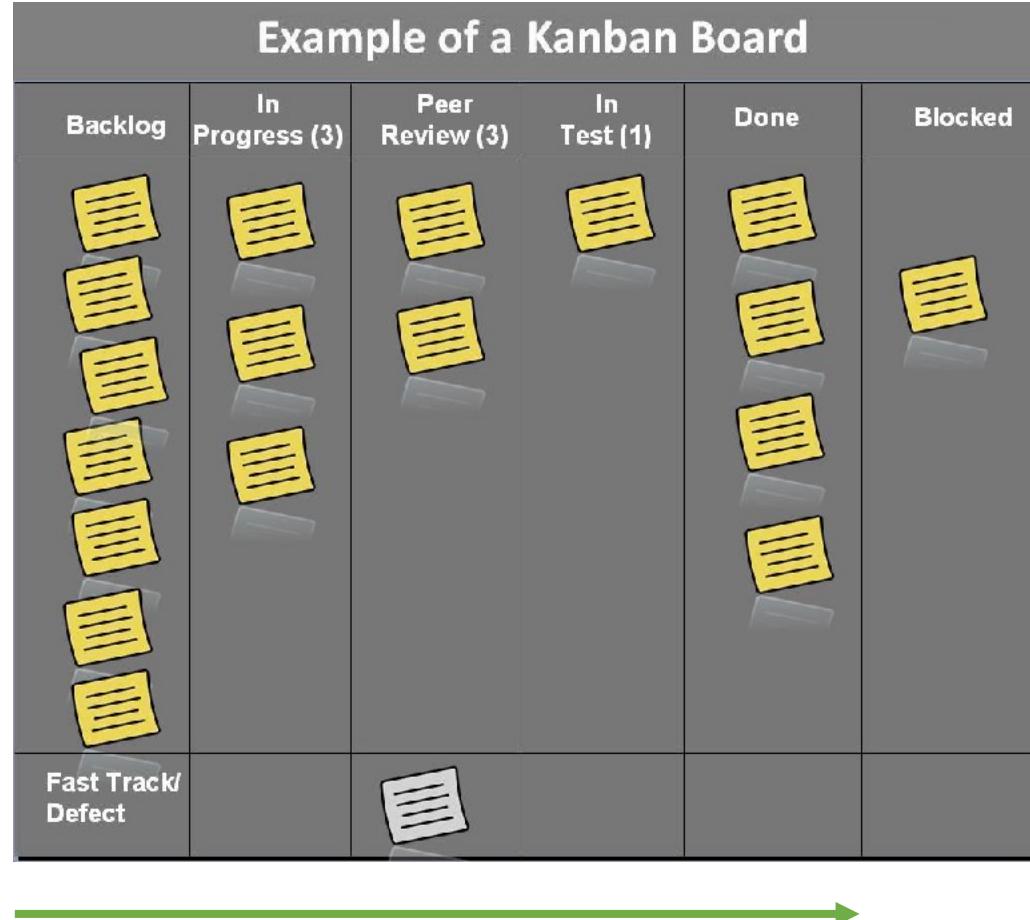
Programmer welfare

- Sustainable pace

XP – Impact

- XP provided a ‘jolt’ that brought attention to agile methods.
- Some see XP as ‘**dogmatic**'; others see it as a **consistent**, strong view of how programming should be practiced.
- Many of the **individual practices** promoted by XP have left their mark on industry, especially that:
 - Projects should **integrate code all the time**
 - Tests are a **key resource**, and should be run against code often

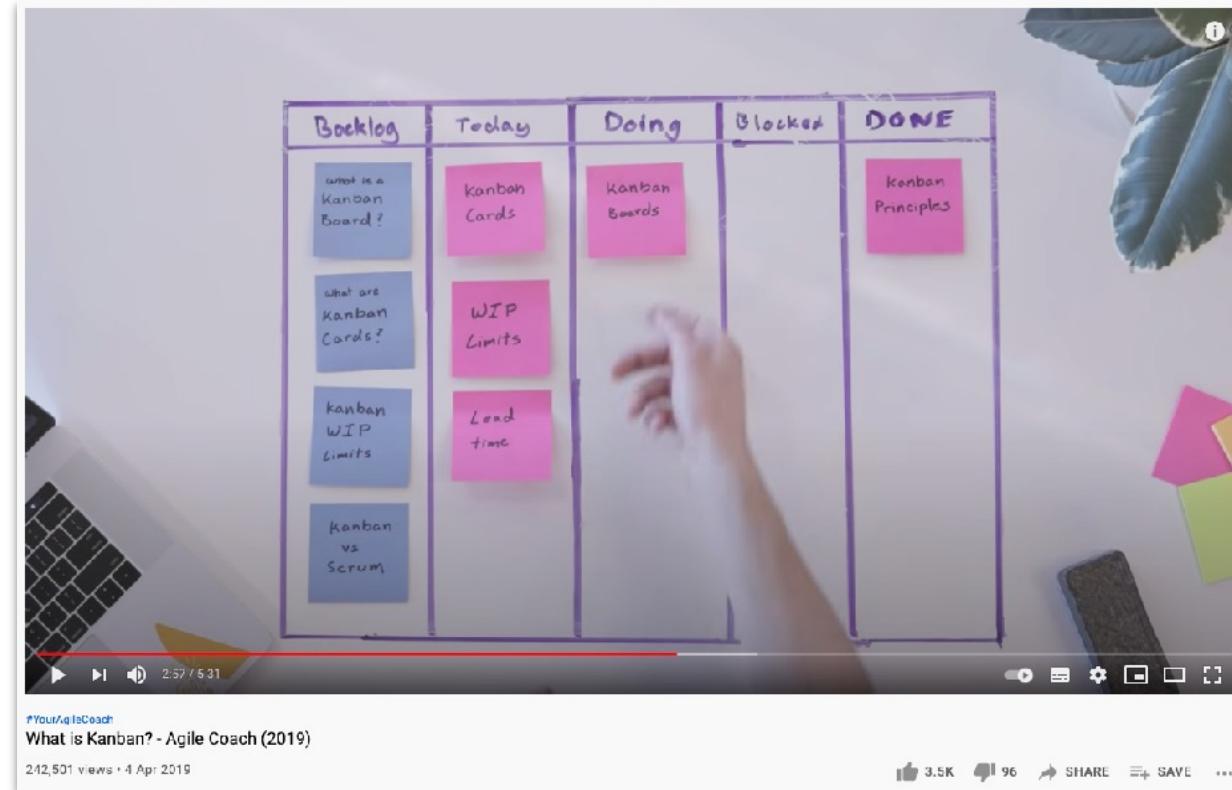
Kanban



“kanban”

- Japanese word meaning (roughly) “large visual board”
- *idea: visualising a team’s workflow helps identify potential waste / constraints*
- NB: a single ‘kanban board’ is used throughout the *whole project*

Kanban



<https://www.youtube.com/watch?v=iVaFVa7HYj4&t=116s>

Kanban

- Work-in-progress (WIP) doesn't deliver value until deployed
- Limiting / minimising WIP makes it easier to identify inefficiencies and constraints in a team's workflow
- Kanban: clearly visualise WIP and match to team's capacity
- Simple, but effective approach to help teams coordinate

Kanban – Virtual Boards

The Kanban Method

To Do

- Time out when accessing reports (Feb 26, 2020)
- Webpack update
- Analyze transactions performance
- Setup Staging 2 test environment

Development

- + Add a card

Code Review

- + Add a card

Testing

- + Add a card

Done

- + Add a card

<https://trello.com/en>

Kanban – Virtual Boards

The screenshot shows a Kanban-style virtual board interface with the following data:

Column	Item	Description	Assignee	Status
TO DO	When requesting user details the service should return prior trip info	SEESPACEEZ PLUS	TIS-37	Green up arrow
	Engage Jupiter Express for outer solar system travel	SPACE TRAVEL PARTNERS	TIS-25	Green up arrow
	Create 90 day plans for all departments in the Mars Office	LOCAL MARS OFFICE	TIS-12	Green up arrow
IN PROGRESS	Requesting available flights is now taking > 5 seconds	SEESPACEEZ PLUS	TIS-8	Green up arrow, Red down arrow
	Register with the Mars Ministry of Labor	LOCAL MARS OFFICE	TIS-11	Green up arrow, Red down arrow
	Engage Saturn Shuttle Lines for group tours	SPACE TRAVEL PARTNERS	TIS-20	Red background
IN REVIEW	Register with the Mars Ministry of Revenue	LOCAL MARS OFFICE	TIS-13	Green up arrow
	Draft network plan for Mars Office	LOCAL MARS OFFICE	TIS-49	Green checkmark, Red down arrow, Profile icon
DONE	Add pointer to main css file to instruct users to create child themes	LARGE TEAM SUPPORT	TIS-56	Green up arrow
	Homepage footer uses an inline style - should use a class	LARGE TEAM SUPPORT	TIS-68	Green up arrow, Red down arrow, Profile icon
DONE	Engage JetShuttle SpaceWays for travel	SPACE TRAVEL PARTNERS	TIS-23	Green up arrow, Red down arrow, Number 5, Profile icon

<https://www.atlassian.com/software/jira>

Scrum



“scrum”

- borrowed from rugby
(formation of players)

- term introduced to
emphasise teamwork

We will follow the **scrum process**
for the IS212 project

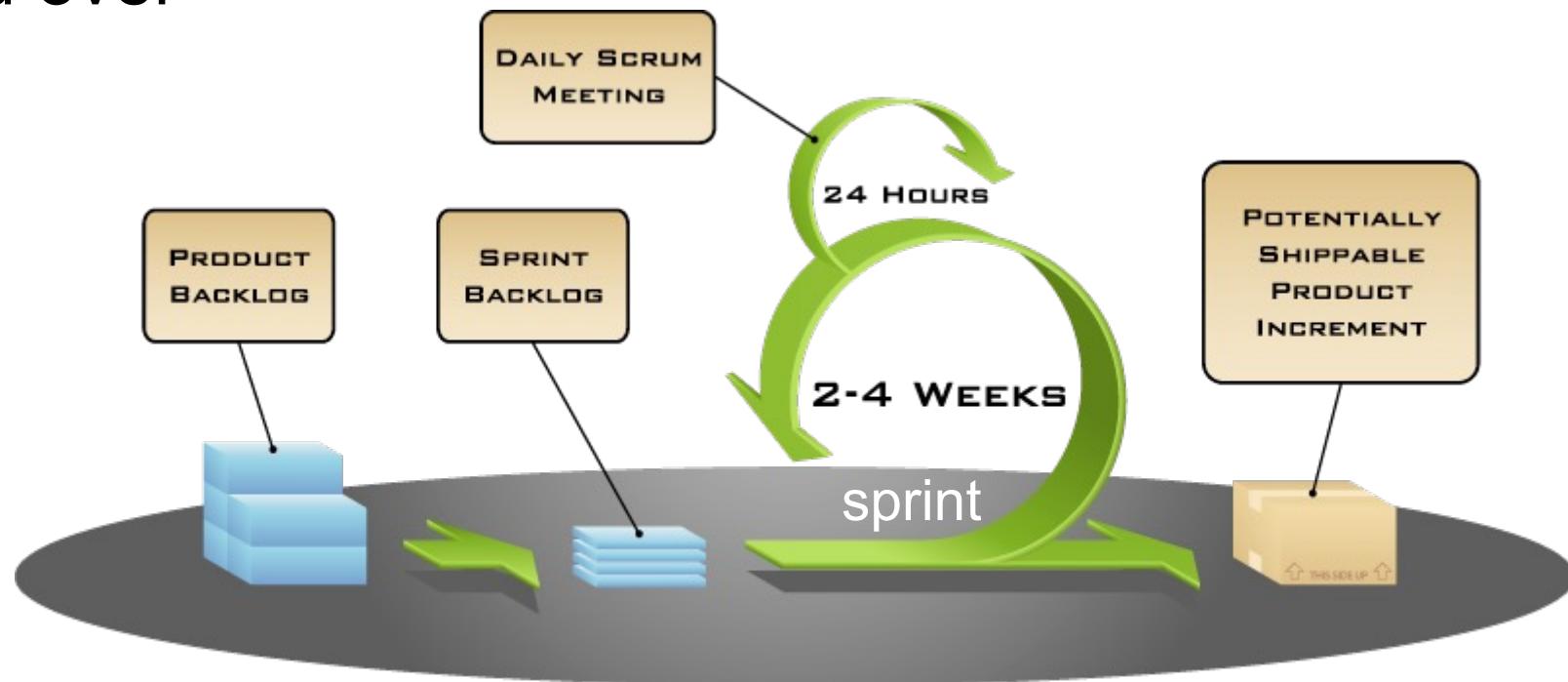
Scrum



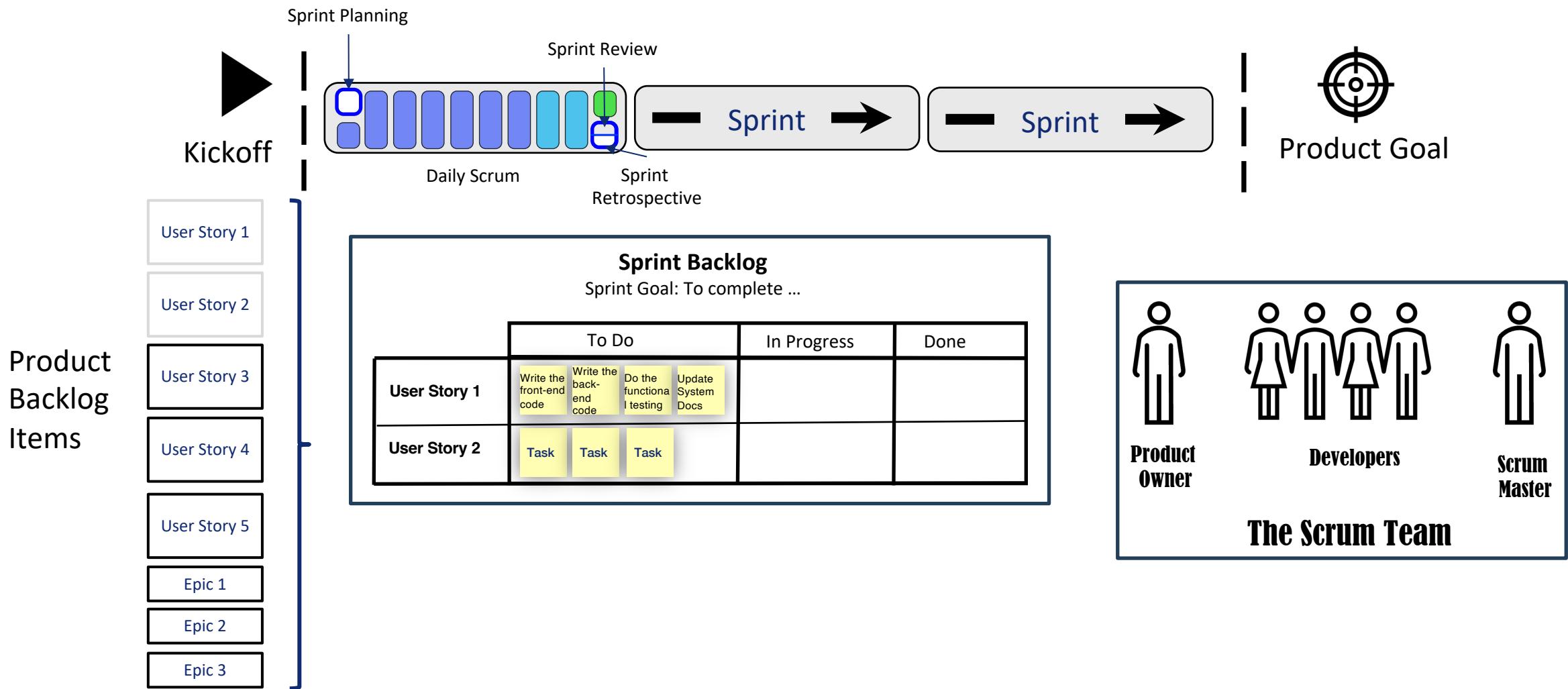
<https://www.youtube.com/watch?v=TRcReyRYIMg>

Scrum

- Scrum is a **lightweight, iterative method** for managing software development in a complex and changing environment
- It has a **cyclical nature** – the “**sprint**”, and processes within it, repeat over and over



Scrum – A Closer Look (More in Weeks 2–4)



Scrum – Accountabilities

Scrum Master

- Facilitates scrum events (standups, reviews, retros)
- Removes impediments
- Facilitates mentorship and conflict resolution
- Ensures that team embraces agile values to achieve the business goals



Scrum Teams: 5 to 8 developers

Developers

- Design, Code, Test and Integrate the software product
- Deliver High-Quality Shippable Product
- Are Self Organised and Cross functional
- Pace the development in a Sustainable manner



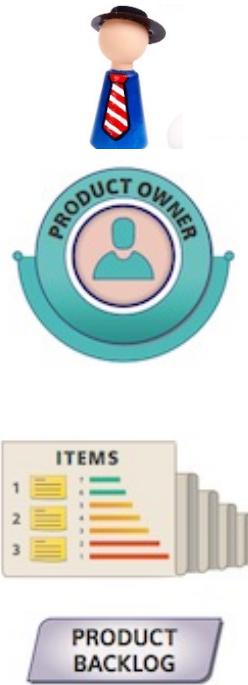
Product Owner

- Identifies and prioritises/re-prioritises product features
- Writes user stories and grooms the product backlog
- Collaborates with stakeholders (e.g. the customer)
- Validates product increments after each sprint



What is a Product Backlog? (More In Week 2)

- It is a **prioritised list** of features that
 - *Will deliver an overall **Product Goal***
 - *Adds **Value** to the customer*
 - *Is **Estimated** (i.e., amount of effort required)*
- A Product Backlog principally contains items that bring direct value to customers
 - *Features expressed as **User Stories** (see Weeks 2–3)*
- It can also contain items that bring indirect value to customers
 - *Enhancements and Fixes*
 - *Infrastructure set up (e.g. **Git repository**, **CI/CD pipeline**)*
 - *Exploratory research work*
- It is created and groomed by the **Product Owner** as the Product evolves
- It is a **living document** that continuously evolves as the Sprints are executed
- It does **not** contain **low-level tasks**
 - e.g. *sending emails* ✗, *implement PersonDAO class* ✗



Product Backlog Example (More in Week 2)

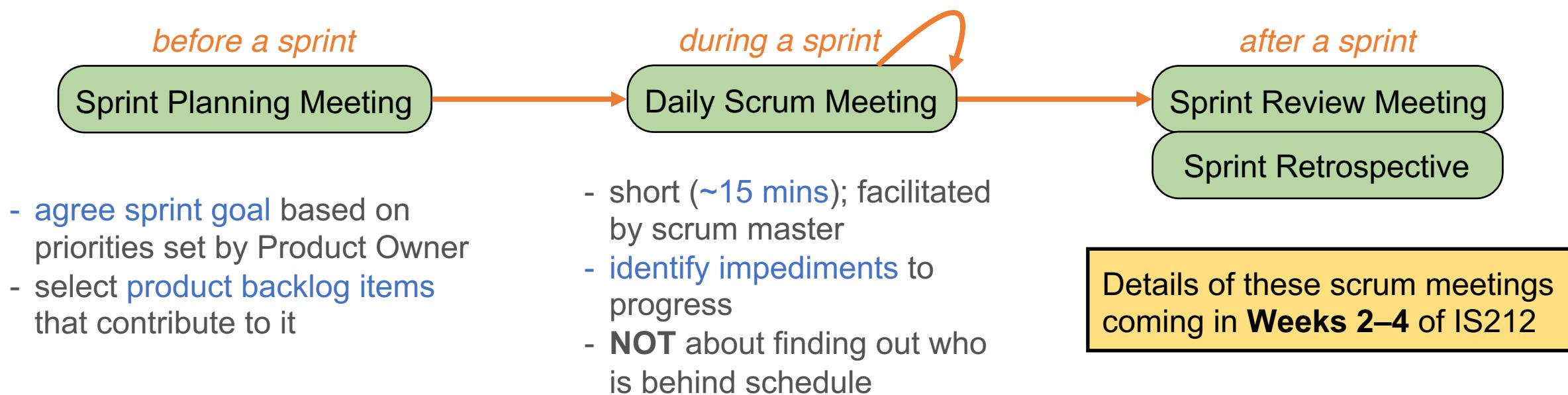
Priority	User Story	Estimate	More Details (Wiki Page)	Status
1	As a shopper, I want to browse the catalog of products, so that I know what to buy	3	http://wiki.shopping.com/US010	Done Sprint 1
2	As a shopper I want to add a product to my cart, so that I can buy the product	8	http://wiki.shopping.com/US011	Done Sprint 1
3	As a shopper, I want to remove a product from my cart so that I don't buy the product	5	http://wiki.shopping.com/US012	In-Prog Sprint 2
4	As a shopper, I want to checkout my cart so that I complete my shopping	13	http://wiki.shopping.com/US013	In-Prog Sprint 2
7	As a shopper I want to choose payment on delivery, so that can pay after receiving my product	1	http://wiki.shopping.com/US014	Sprint 3



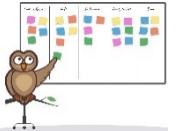
Product Goal: To provide a convenient way for Shoppers to shop online

Scrum – Sprints

- Sprints are fixed-length (e.g. 2, 3, or 4 week) iterations that aim to achieve some agreed-upon outcome, e.g. some tested/integrated software increment
 - For the IS212 project, we recommend a sprint length of **1 or 2 weeks**
 - Requirements are frozen for the length of a sprint
- Sprints are lightly managed using "time-boxed" meetings



The Scrum Taskboard



SCRUM
BOARD

- Information Radiation
- Each user story associated with one or more implementation tasks
- Monitored by the Scrum master
 - Movement of tasks
 - Stagnant WIP is a warning!
- Updated by Team during standup meetings



Sprint Backlog				
	To Do		In Progress	Done
User Story 1	Write the front-end code	Write the back-end code	Do the functional testing	Update System Docs
User Story 2	Task	Task	Task	

How to get Scrum right

- You will use the scrum process for the IS212 project
- You are encouraged to [learn by doing](#) – get your hands dirty, make mistakes in sprint #1, reflect, and then improve
- But we also provide a [scrum guide](#) and [primer](#) as required reading (see eLearn)
- Read them as we ‘deep dive’ into user stories, test cases, and agile estimation in **Weeks 2-4**

Ken Schwaber & Jeff Sutherland

The Scrum Guide

[The Definitive Guide to Scrum: The Rules of the Game](#)



Scrum – Certification

- The Scrum Alliance provides a number of certification routes for those keen to develop their agile credentials

Certifications by Scrum Team Role

SCRUM MASTER TRACK	PRODUCT OWNER TRACK	DEVELOPER TRACK
Certified ScrumMaster®  Intro course for those wishing to fill the role of Scrum Master or Scrum team member. Prerequisite: none Learn more → Find a course	Certified Scrum Product Owner®  Intro course for those who are closest to the "business side" of the project. Prerequisite: none Learn more → Find a course	Certified Scrum Developer®  Intro course focused on collaborative product development for Scrum team members. Prerequisite: none Learn more → Find a course
Advanced Certified ScrumMaster®  Advanced course for Scrum Masters who have one or more years of work experience in that role. Prerequisite: CSM Learn more → Find a course	Advanced Certified Scrum Product Owner®  Advanced course for Product Owners who already have one year of experience on a Scrum team. Prerequisite: CSPO Learn more → Find a course	Advanced Certified Scrum Developer®  Advanced course for Scrum product development team members. Prerequisite: CSD Learn more → Find a course
Certified Scrum Professional® ScrumMaster  Pinnacle course for experts wishing to develop mastery of the Scrum Master track.	Certified Scrum Professional® Product Owner  Pinnacle course for experts wishing to master the Product Owner track.	Certified Scrum Professional®  Pinnacle course for experts wishing to master Scrum product development. Prerequisite: active CSD

<https://www.scrumalliance.org/>

Scrum – Impact

- Scrum has turned the general idea of iterative development into a precise discipline
 - Codified goals, duration, and management of iterations (sprints)
- Savvy marketing operation – certifications turn scrum learners into scrum supporters
- Scrum / sprints have quickly become the industry standard iteration model – even beyond software development

Scrum vs. Kanban

- Scrum is a different process to Kanban, but it shares some similar foundational principles (e.g. minimising WIP)

	Scrum	Kanban
Cadence	Fixed-length sprints	Continuous flow
Release methodology	End of each sprint	Continuous delivery
Accountabilities / Roles	Product owner, scrum master, scrum team	No required roles
Key metrics	Velocity	Lead time, cycle time, WIP
Change philosophy	Requirements frozen during sprints	Requirements can change at any point
Visualisation	Scrum board cleared after every sprint	Kanban board used throughout project

Group Exercise

Google Drive Exercise

In breakout rooms, copy and complete the template at:

<http://smu.sg/XXX>

Be ready to share and discuss with us!

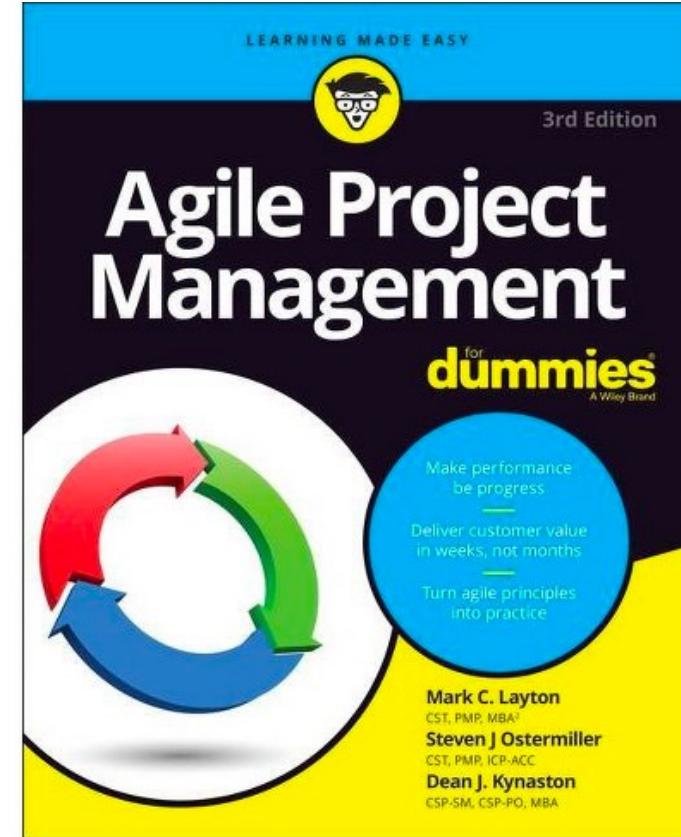
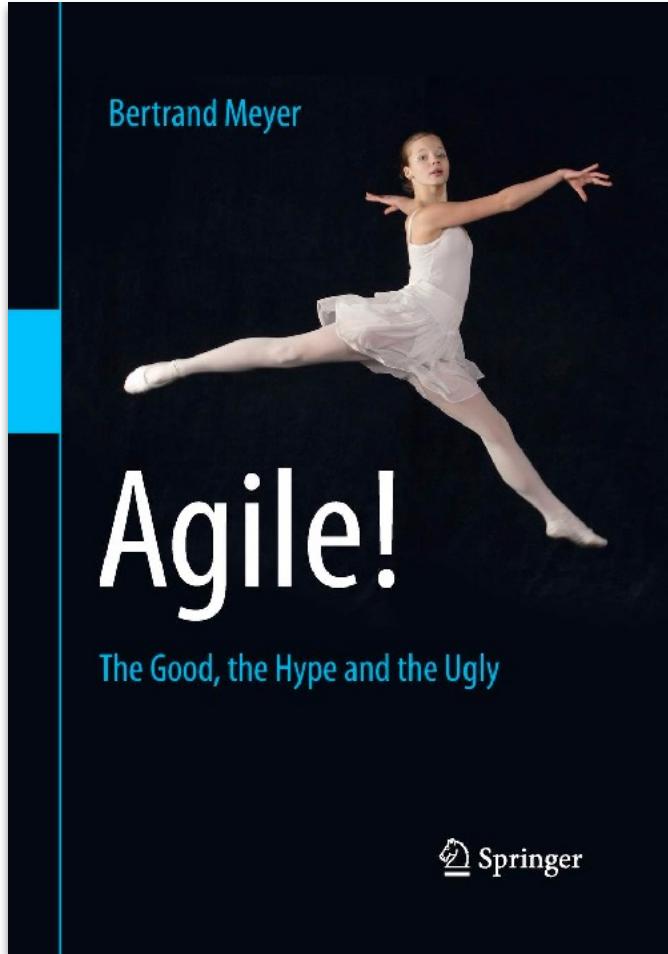
Traditional

- Waterfall
- RUP

Agile

- Extreme Programming
- Kanban
- Scrum

Additional Reading (Optional)



(ignore the title!)