SWEN90016

Software Processes & Project Management

Marion Zalk
Department of Computing and Information Systems
The University of Melbourne
mzalk@unimelb.edu.au

2021 – Semester 1
Lecture 2

Lecture 1 – Recap

- ✓ Understand Assignments and our expectations
- ✓ Understand key elements of a Project and why organisations use them
- ✓ Understand the foundational components of Project Management
- ✓ Understand key skills, responsibilities & activities of a Project Manager
- ✓ Understand key elements of how to manage Projects
- ✓ Exposure to some Project Management Methodologies

Lecture 1 – Recap

- ✓ Explore key drivers in why projects fail / succeed
- ✓ Understand how organisations select the best / right projects
- ✓ Understand the Project Initialization process, Business Case structure and why organisations use them
- ✓ Explore various Investment techniques and financial models
- ✓ Understand responsibilities associated with building a Business Case and the accountable group / individual
- ✓ Understand what a Project Charter is and how it is used

L1 - Recap

Intended Learning Objectives

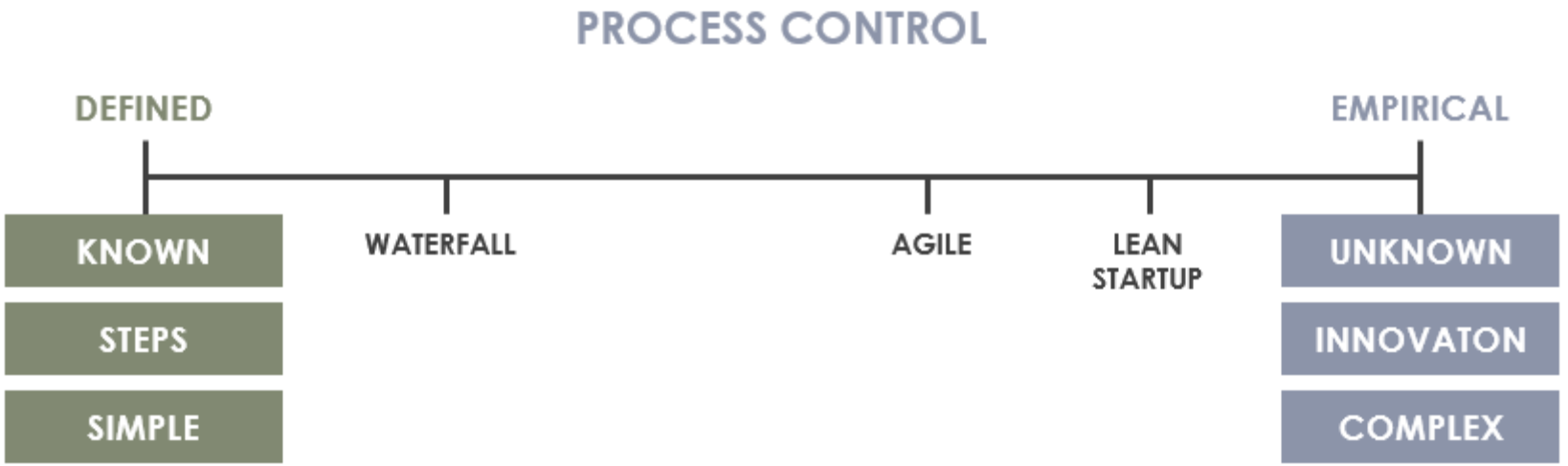
Module 4 – Process

Module 5 – Formal.

Module 6 – Agile

Module 4.1 – Empirical and Defined Process

Empirical process control expects the unexpected, while defined process control expect every piece of work to be completely understood in upfront.

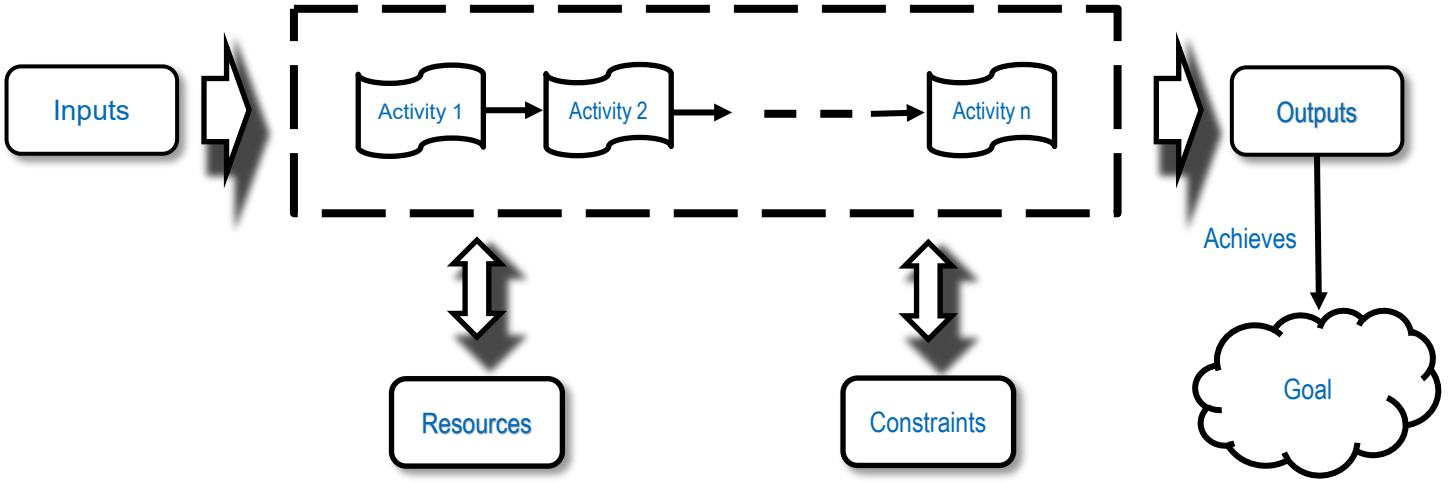


www.visual-paradigm.com/scrum/empirical-vs-defined-process-control

Module 4.1 – Defined Process Control

A process with a well-defined set of steps. Given the same inputs, a defined process should produce the same output every time.

Great when in an environment with relatively low volatility that can be easily predicted; given the same inputs, a defined process should produce the same output every time based on its repeatability and predictability nature.

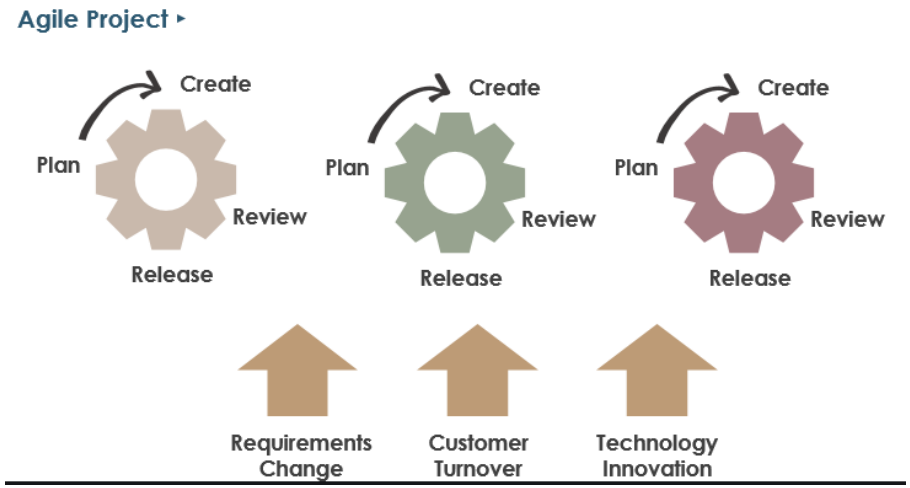


www.visual-paradigm.com/scrum/empirical-vs-defined-process-control

Module 4.1 – Empirical Process Control

In empirical process control, you expect the unexpected. Empirical process control has the following characteristics:

- Learn as we progress
- Expect and embrace change
- Inspect and adapt using short development cycles
- Estimates are indicative only and may not be accurate



www.visual-paradigm.com/scrum/empirical-vs-defined-process-control

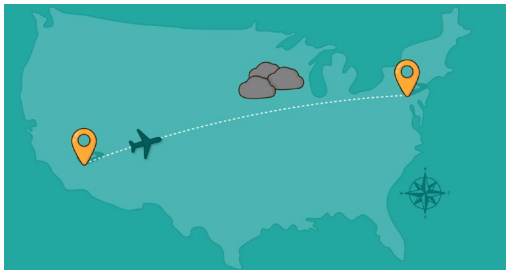
Module 4.1 – Defined v Empirical

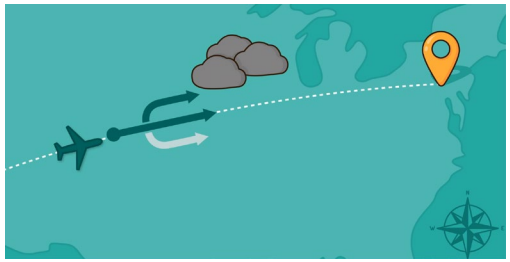
Defined / Repeatable Process

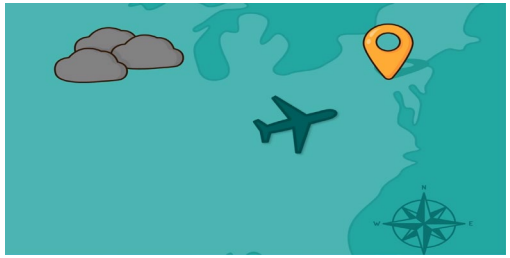
www.mountaingoatsoftware.com/exclusive/scrum-foundations

Module 4.1 – Defined v Empirical

Defined / Repeatable Process

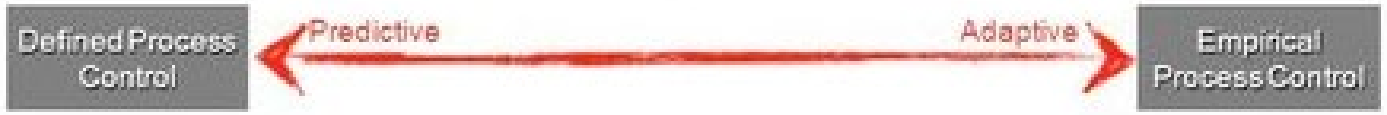






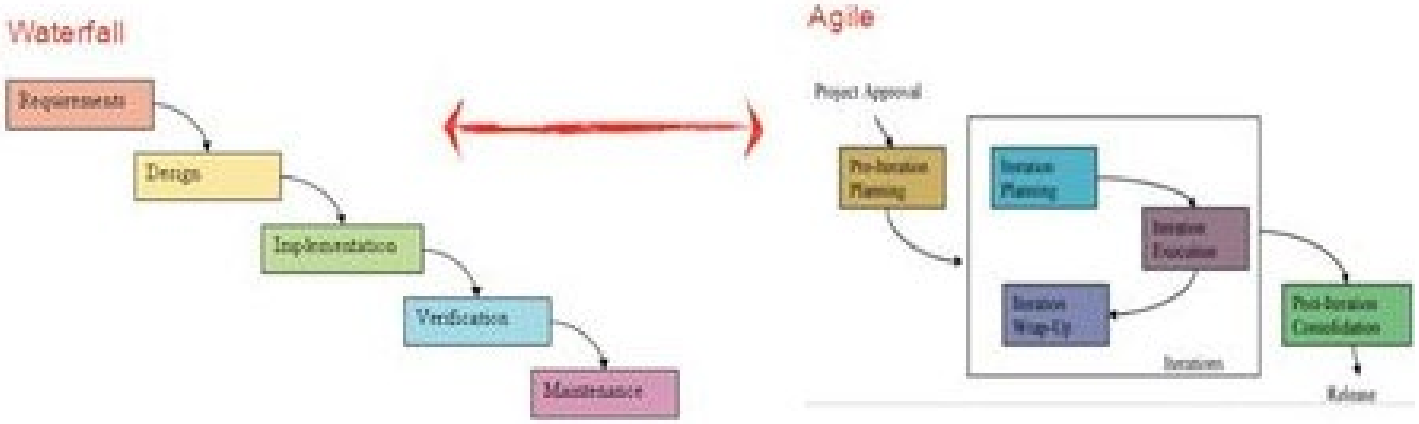
www.mountaingoatsoftware.com/exclusive/scrum-foundations

Module 4.1 – Defined v Empirical



*Requires that **every piece of work be completely understood**. Given a well-defined set of inputs, the same outputs are generated every time. A defined process can be started and allowed to run until completion, with the same results every time.*

*Provides and exercises control through **transparency**, frequent **inspection** and **adaptation** for processes that are imperfectly defined and generate unpredictable and unrepeatable outputs.*



www.visual-paradigm.com/scrum/empirical-vs-defined-process-control

Module 4.1 – What does a Process have to do with Project Management and Software Engineering?

1. Project Management is a process as it defines a series of tasks (Planning, Executing and Controlling) to deliver a specific / an agreed set of outcomes.

2. System Development Lifecycle (SDLC) is a term used in Software Engineering. It describes a process for planning, creating, testing, and deploying an information system. SDLC can be composed of hardware only, software only, or a combination of both.

Module 4.1 – What does a Process have to do with Project Management and Software Engineering?

1. Project Management is a process as it defines a series of tasks (Planning, Executing and Controlling) to deliver a specific / an agreed set of outcomes.

2. System Development Lifecycle (SDLC) is a term used in Software Engineering. It describes a process for planning, creating, testing, and deploying an information system. SDLC can be composed of hardware only, software only, or a combination of both.

SWEN90016

Software Processes & Project Management

Marion Zalk
Department of Computing and Information Systems
The University of Melbourne
mzalk@unimelb.edu.au

2021 – Semester 2

Lecture 2

Lecture 2 – Intended Learning Objectives

Module 5 – Software Development Lifecycles - Formal.

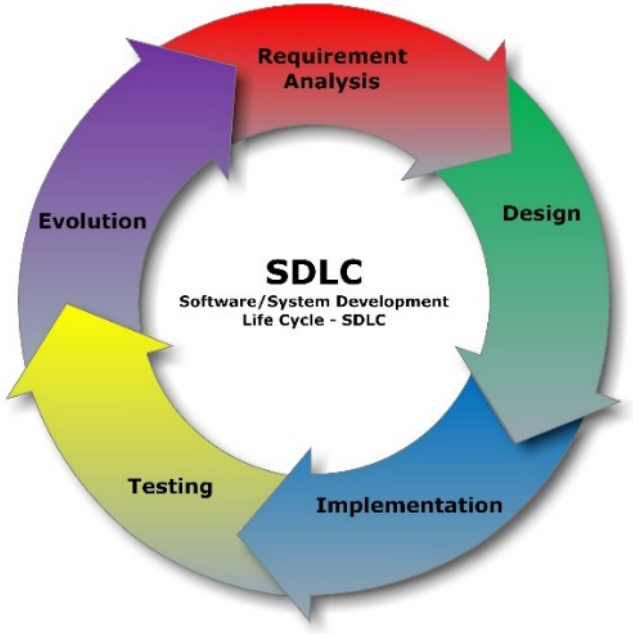
Module 6 – Software Development Lifecycles - Agile.

Software Development Life Cycle (SDLC)

The systems development life cycle (SDLC), also referred to as the application development life-cycle, is a term used in systems engineering, information systems and software engineering to describe a ***process*** for planning, creating, testing and deploying an information system.

Activities in SDLC:

- Requirements gathering
- Systems / Architectural Design
- Implementation / coding / Integration
- Testing
- Evolution:
  - Delivery and Release - Deployment
  - Maintenance

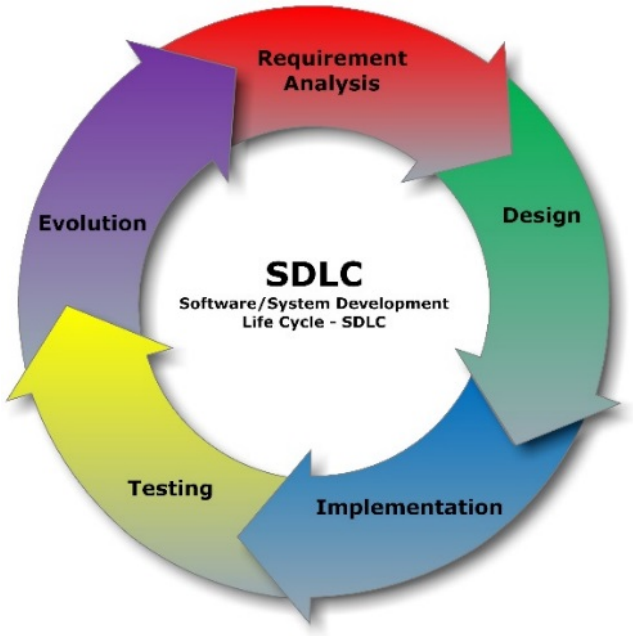


SDLCs

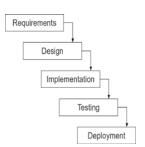
There are many SDLCs around with organisations typically favouring a blend of Formal and Agile approaches.

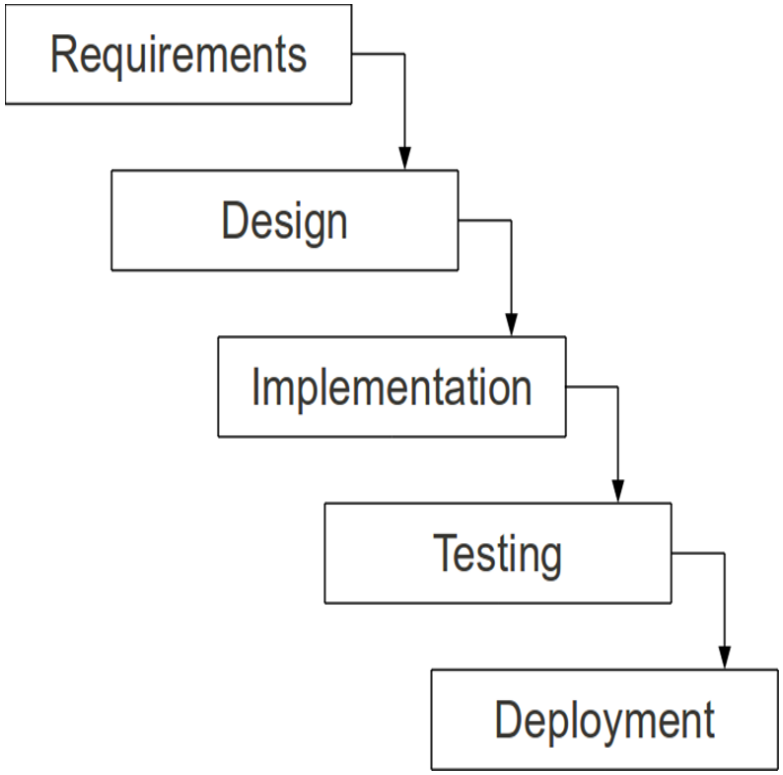
1. Formal
   - Waterfall
   - Incremental
   - V-Model

2. Agile
   - Scrum
   - Kanban
   - Extreme Programming



Waterfall





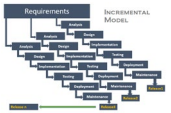
Advantages

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model
- Phases are processed and completed one at a time
- Documentation available at the end of each phase
- Works well for projects where requirements are very well understood and remain stable

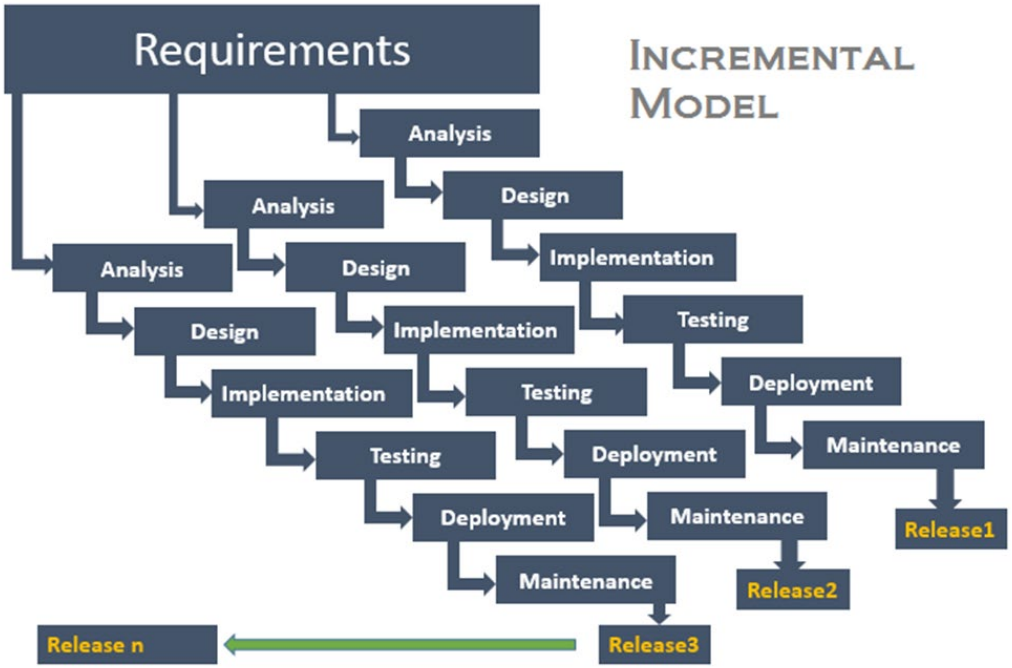
Disadvantages

- Difficult to accommodate change after the process in underway
- One phase must be completed before moving on to the next
- Unclear requirements lead to confusion
- Clients approval is in the final stage
- Difficult to integrate risk management due to uncertainty

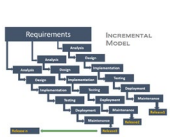
Incremental Model



In incremental model the ***whole requirement*** is divided into various releases. Multiple cycles take place, making the life cycle a ***multi-waterfall*** cycle. Cycles are divided up into smaller, more easily managed modules.



Incremental Model



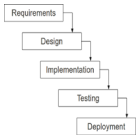
Advantages – compared to standard waterfall

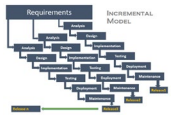
- Each release delivers an operational product
- Less costly to change the scope/requirements
- Customers can respond to each build
- Initial product delivery is faster
- Customers get important functionality early
- Easier to test and debug during smaller iterations

Disadvantages – compared to standard waterfall

- More resources may be required
- More management attention is required
- Defining / partitioning the increments is difficult and often not clear
- Each phase of an iteration is rigid with no overlaps
- Problems may occur at the time of final integration

Formal Models





Characteristics where "Formal" Models make sense:

- Projects where the customer has a very clear view of what they want
- Projects that will require little or no change to requirements
- Software requirements are clearly defined and documented
- Software development technologies and tools are well-known
- Large scale applications and systems developments

SWEN90016

Software Processes & Project Management

Marion Zalk
Department of Computing and Information Systems
The University of Melbourne
mzalk@unimelb.edu.au

2021 – Semester 2

Lecture 2

Lecture 2 – Intended Learning Objectives

~~Module 5 – Software Development Lifecycles - Formal.~~

Module 6 – Software Development Lifecycles - Agile.

Intended Learning Objectives

Software Development Lifecycles

Agile:

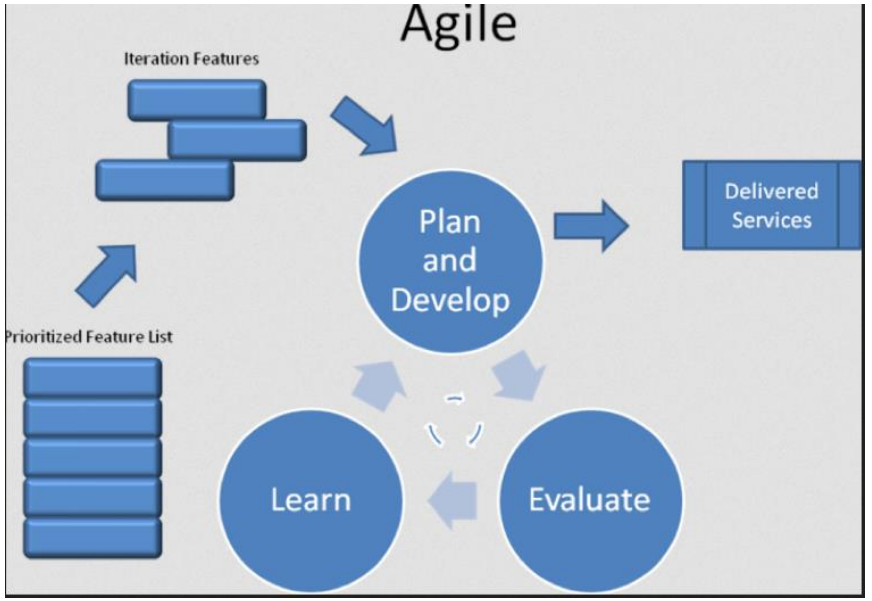
1. Understand what Agile is and its origins.
2. Understand the Agile framework.
3. Understand Scrum – Roles, Ceremonies and Artefacts.
4. Understand advantages / disadvantages of Agile.

SDLCs

There are many SDLCs around with organisations typically favouring a blend of Formal and Agile approaches.

1. Formal
   - Waterfall
   - Incremental
   - V-Model

2. Agile
   - Kanban
   - Scrum
   - Extreme Programming



Why is Agile attractive

- We are in an ever changing global world with the pace of change increasing
- Customer needs and demands are exponentially increasing – products must continually be delivered
- Low Technology cost, ease of use and the global market place has increased competition and reduced entry barriers
- The war for talent is over – and we have lost! Cross functional teams help minimise the potential loss
- Long development cycles are like long lunches – a thing of the past
- Quality is no longer something we do / check later – it must be part of everything we do
- Cross functional groups are more fun!

What is Agile

- A framework based on iterative development where requirements and solutions evolve through collaboration between self-organising cross-functional teams
- A disciplined process that encourages frequent inspection and adaptation
- A leadership philosophy that encourages teamwork, self-organisation and accountability
- A set of engineering best practices intended to allow for rapid delivery of high-quality software
- A business approach that aligns development with customer needs and company goals

What is Agile

- In software development, we think about methodologies, activities, interactions, results, work products, artefacts and processes to organise the work.

The main tasks of software development remain the same regardless of methodology used, however with Agile, the flow of activities, how they are undertaken and who is involved is extremely different.

What is Agile - Origins

- Changes initiated in large US corporates in 1990's
- Software engineers frustrated with
  - long lead times before products were delivered
  - Decisions made early in the project couldn't be changed later
- 17 software engineer thought leaders met first in 2000 to discuss software engineering and different approaches
- Famous meeting in 2001 at the Snowbird ski lodge in Utah where they met to change the way the industry designed, engineered and deployed software
- Brought together by the Agile Manifesto

https://techbeacon.com/agility-beyond-history%E2%80%94-legacy%E2%80%94-agile-development

Intended Learning Objectives

Module 8 - Software Development Lifecycles

Agile:

1. ~~Understand what Agile is and its origins.~~
2. Understand the Agile framework.
3. Understand Scrum – Roles, Ceremonies and Artefacts.
4. Understand advantages / disadvantages of Agile.

Agile Framework

Primary elements of the Agile framework include:

- Manifesto
- 12 Key Principles
- Kanban
- Scrum

Agile Framework - *Manifesto*

Agile Manifesto

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.

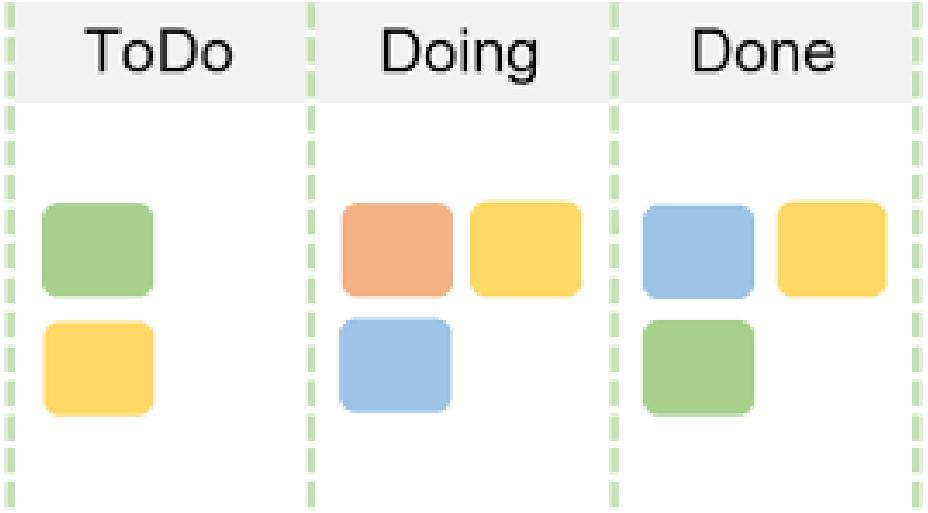
http://www.agilealliance.org

Agile Framework – *12 Key Principles*

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, shorter timeframes is the preference.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need and trust them.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity - the art of maximizing the amount of work not done - is essential.
11. The best architectures, requirements, and designs emerge from self-organising teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

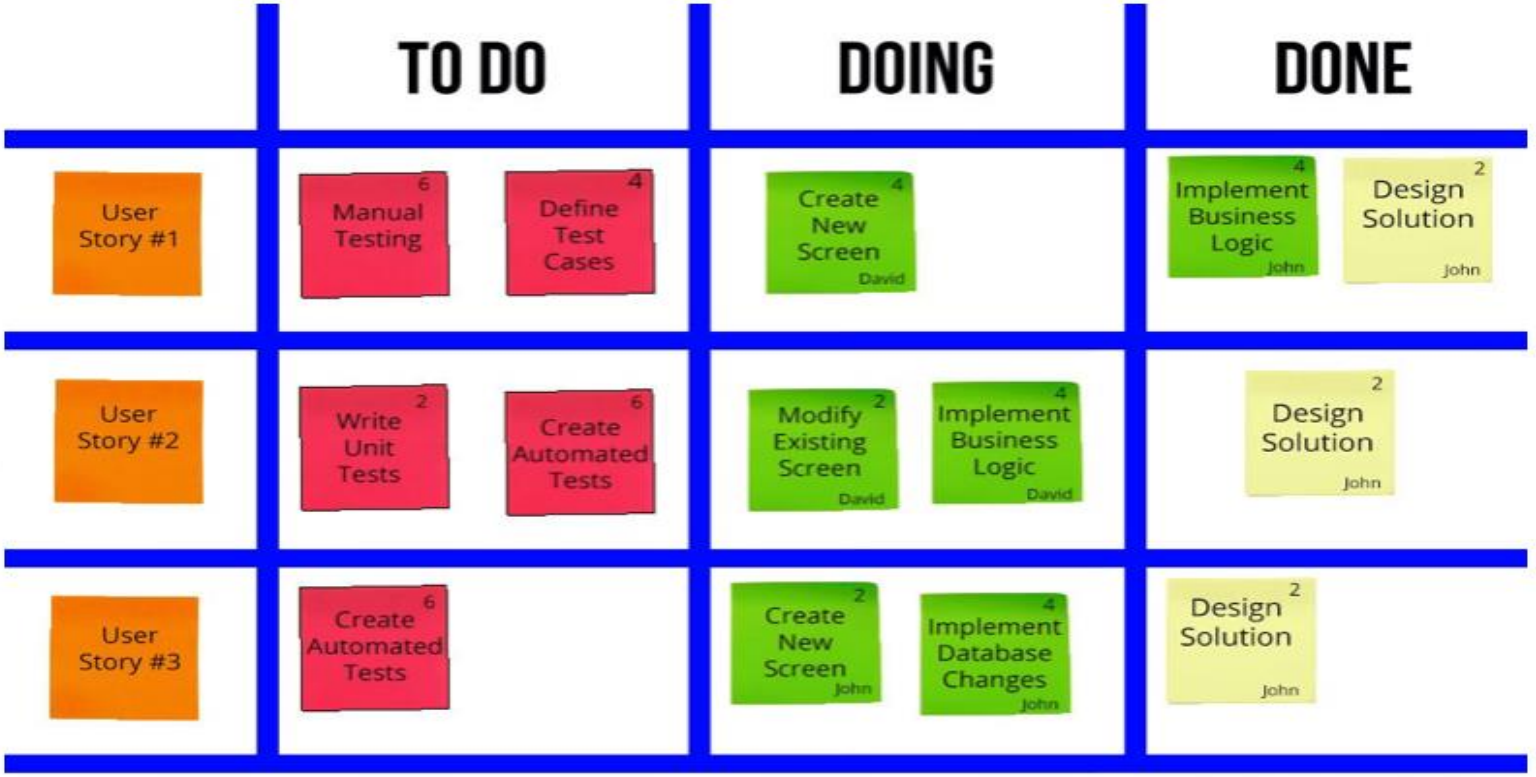
Agile Framework - *Kanban*

- ***Signboard / Billboard:*** Work items are visualised to provide participants a view of progress and process, from start to finish usually via a Kanban board



Agile Framework - *Kanban*

Visual progress gives transparency/accountability for self-organizing teams often referred to as ***SWIMLANE*** boards

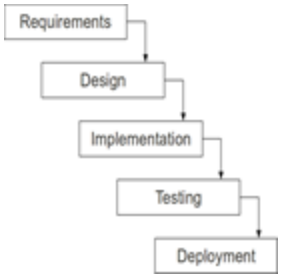


Agile Framework - *Scrum*

Scrum is an agile way to manage a project

"The… 'relay race' approach to product development…may conflict with the goals of maximum speed and flexibility. Instead a holistic or 'rugby' approach—where a team tries to go the distance as a unit, passing the ball back and forth—may better serve today's competitive requirements."

Hirotaka Takeuchi and Ikujiro Nonaka, "The New New Product Development Game", *Harvard Business Review*, January 1986.



Intended Learning Objectives

Module 8 - Software Development Lifecycles

Agile:

1. ~~Understand what Agile is and its origins.~~
2. ~~Understand the Agile framework.~~
3. Understand Scrum – Roles, Ceremonies and Artefacts.
4. Understand advantages / disadvantages of Agile.

Scrum

Scrum in 100 words

- Scrum is an agile process that allows us to focus on delivering the highest business value in the shortest time.
- It allows us to rapidly and repeatedly inspect actual working software (every two to four weeks).
- The business sets the priorities. Teams self-organise to determine the best way to deliver the highest priority features.
- Every two to four weeks, you can see real working software and decide to release it as is or continue to enhance it for another sprint.

Scrum is used by many organisations

- Microsoft
- Yahoo
- Google
- Electronic Arts
- High Moon Studios
- Lockheed Martin
- Philips
- Siemens
- Nokia
- Capital One
- BBC
- Intuit
- Intuit
- Nielsen Media
- First American Real Estate
- BMC Software
- Ipswitch
- John Deere
- Lexis Nexis
- Sabre
- Salesforce.com
- Time Warner
- Turner Broadcasting
- Oce

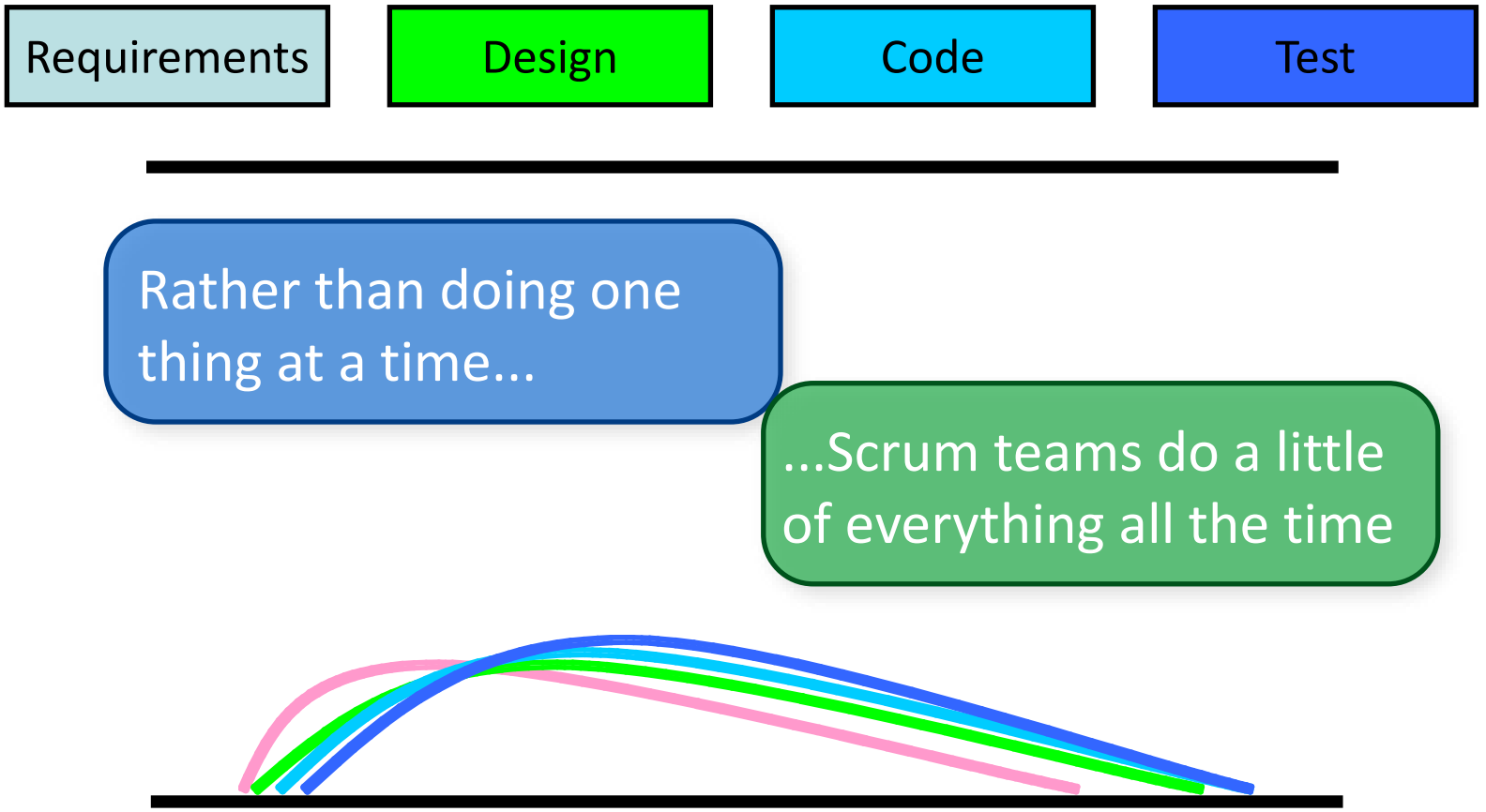
Scrum is used for all types of projects

- Commercial software
- In-house development
- Contract development
- Fixed-price projects
- Financial applications
- ISO 9001-certified applications
- Embedded systems
- 24x7 systems with 99.999% uptime requirements
- the Joint Strike Fighter
- Video game development
- FDA-approved, life-critical systems
- Satellite-control software
- Websites
- Handheld software
- Mobile phones
- Network switching applications
- ISV applications
- Some of the largest applications in use

Scrum Key Characteristics

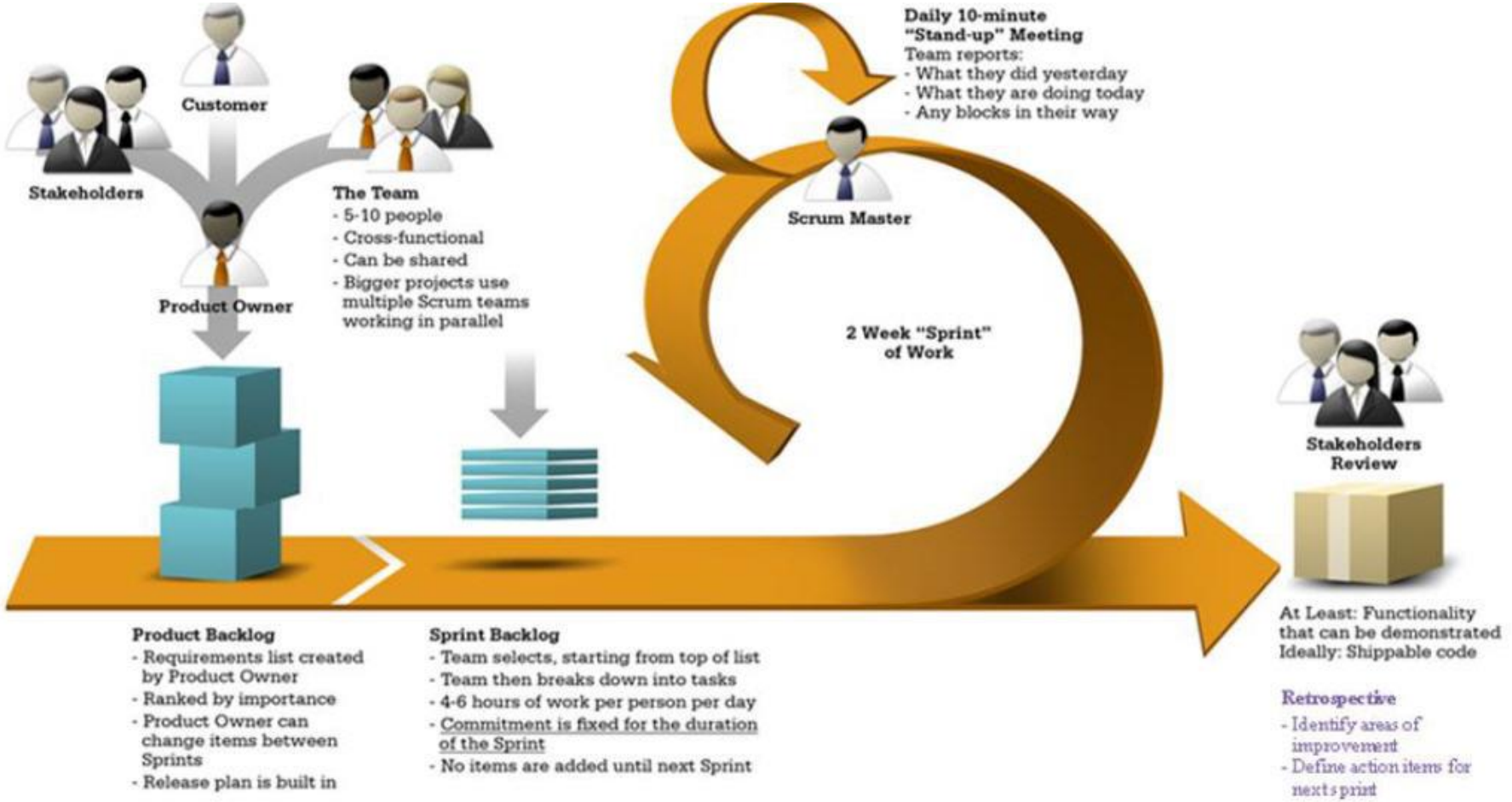
- Self-organising teams
- Product progresses in a series of focused sprints
- Requirements are captured as items in a list of product backlog
- Scrum is one of the agile processes – the one most widely used, discussed and debated
- Time frame is contained to a manageable size (weeks or months)

Scrum Framework - *Sprints*

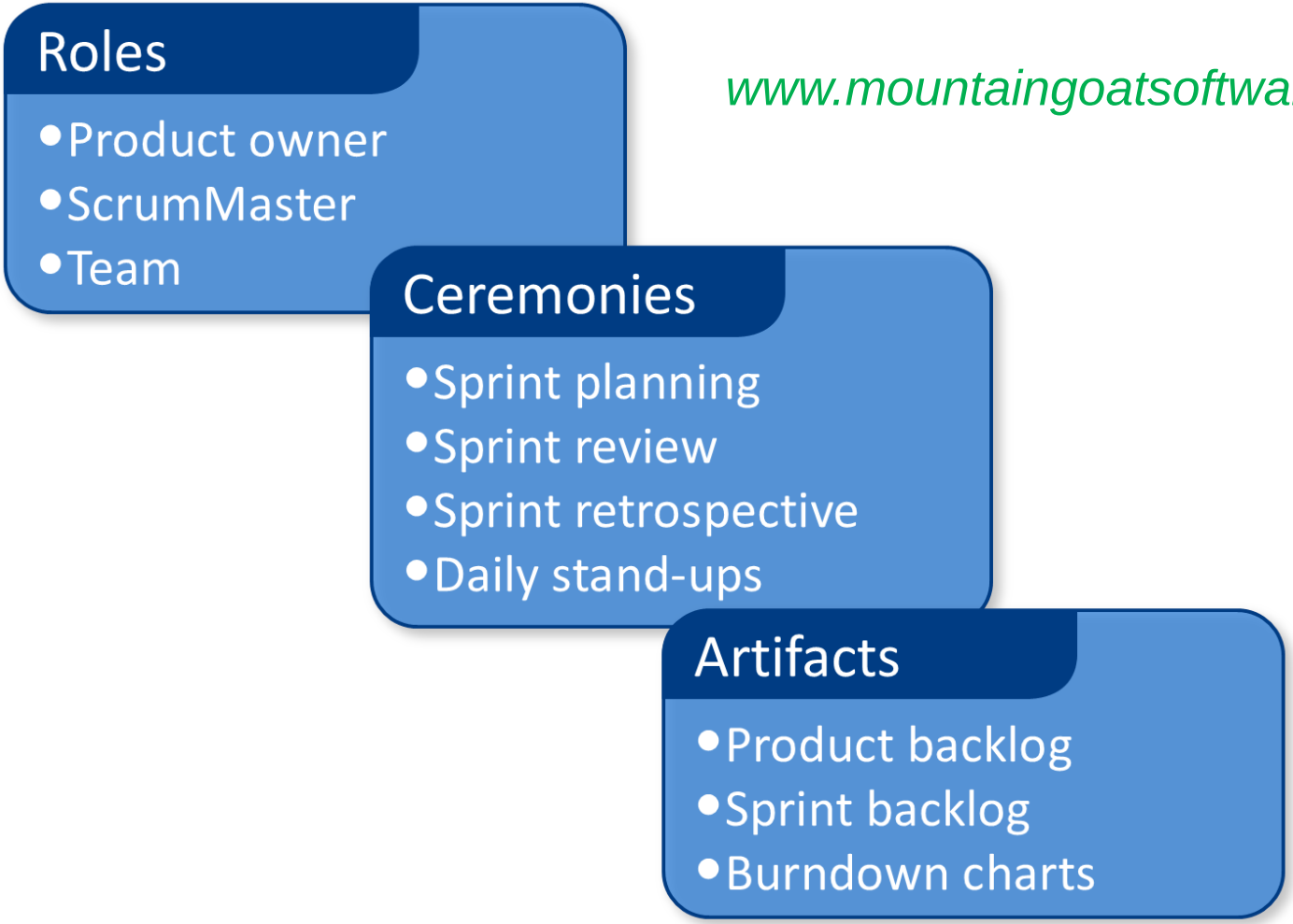


Source: "The New New Product Development Game" by Takeuchi and Nonaka. Harvard Business Review, January 1986.

Scrum Framework



Scrum Framework



www.mountaingoatsoftware.com

Scrum Roles

Roles

- Product owner
- ScrumMaster
- Team

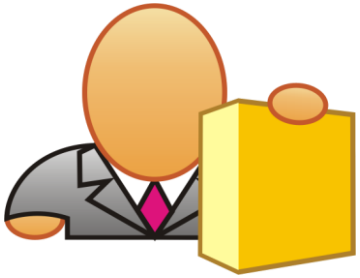
Ceremonies

- Sprint planning
- Sprint review
- Sprint retrospective
- Daily stand-ups

Artifacts

- Product backlog
- Sprint backlog
- Burndown charts

Scrum Roles – *Product Owner*



- Defines the features of the product
- Decides on release date and content
- Is responsible for the Benefits / Profitability of the product (ROI)
- Prioritises features according to market value
- Adjusts features and priority every iteration, as needed
- Accepts or reject work results

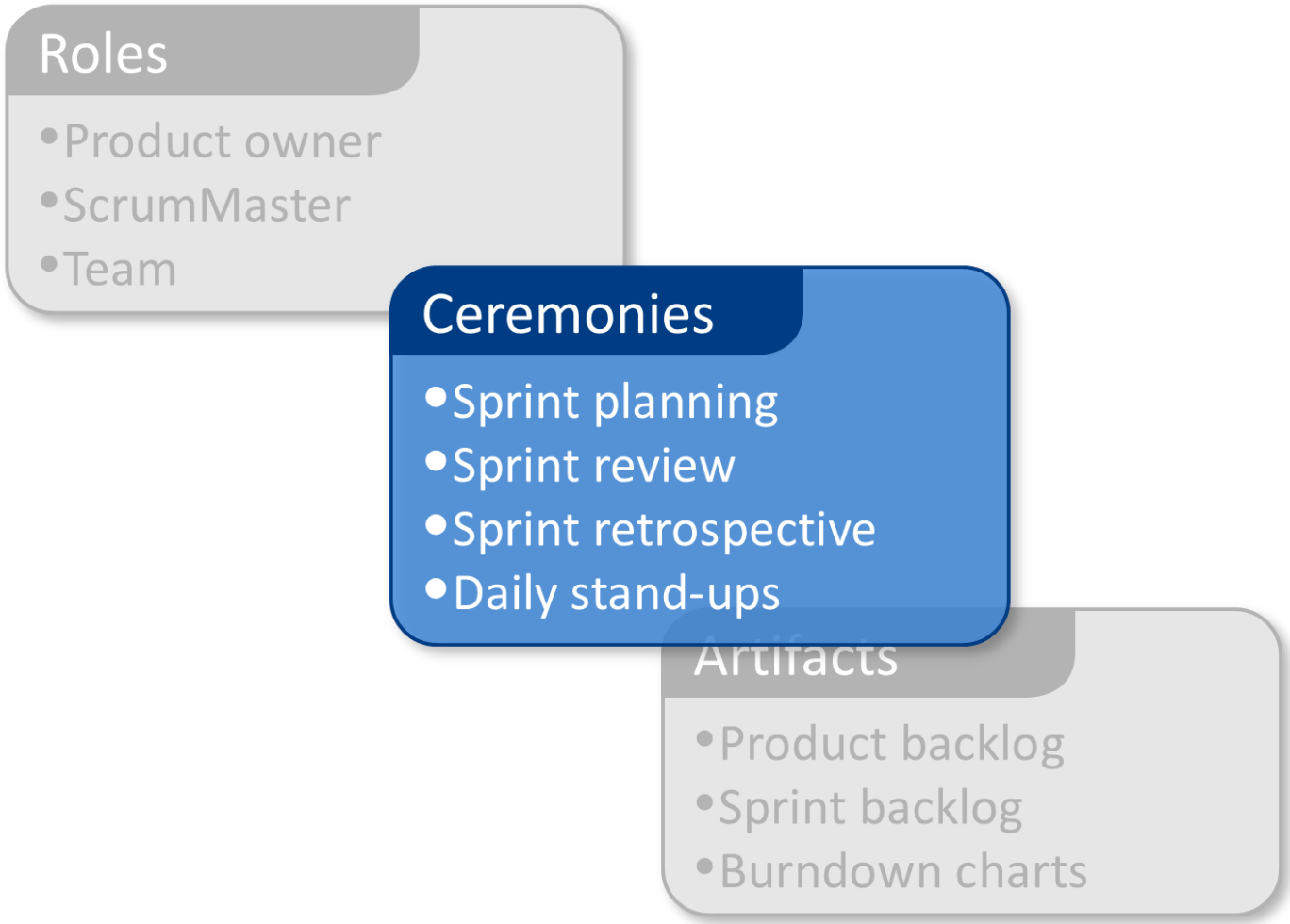
Scrum Roles – *Scrum Master*

- Represents management to the project
- Responsible for enacting Scrum values and practices
- Removes impediments / road blocks
- Ensures that the team is fully functional and productive
- Enables close cooperation across all roles
- Shields the team from external interferences
- Is a member & active participant of the Scrum Team

Scrum Roles – *The Team*

- Typically 6 - 9 people
- Cross-functional:
  - Programmers, testers, user experience designers, business representatives etc.
- Members should be full-time – some exceptions
- Co-located (physically or virtually)

Scrum Ceremonies / Meetings



Scrum Ceremonies / Meetings

Sprint Planning

- Defines how to achieve sprint goal (design)
- Create sprint backlog (User Stories) from product backlog
- Estimate sprint backlog in team velocity and Story Points
- Product Owner priority guides the work
- Release Plan is created
- High-level design is considered

Scrum Ceremonies / Meetings

Sprint Planning Meeting

1st Half of the meeting

- Team defines ***what*** can be done in this sprint
- Starts by writing down the Sprint goal
- Identify the items from the backlog that can achieve this

2nd Half of the meeting

- Team figures out ***how*** the work will get done
- Break down each (feature) user story/large user story in the sprint backlog to capture all the work required (story points)
- The user stories form the basis of the sprint plan that is used to track, cost and manage progress

Source: Head First Agile – A Brain-Friendly Guide to Agile Principles, Ideas, and Real-World Practices By Andrew Stellman, Jennifer Greene

Scrum Ceremonies / Meetings

Daily Stand-up

- Parameters
  - Daily
  - 15-minutes and no more than 30 mins
  - Stand-up
- Not for problem solving / Not a status meeting
  - Whole world is invited
  - Only team members, ScrumMaster, Product owner can **clarify** any questions with user stories
- Helps avoid other unnecessary meetings
- 3 key questions asked:
  1. What did I do yesterday.
  2. What will I do today.
  3. What is in my way to get my work completed.

Scrum Ceremonies / Meetings

Sprint Reviews - Showcase

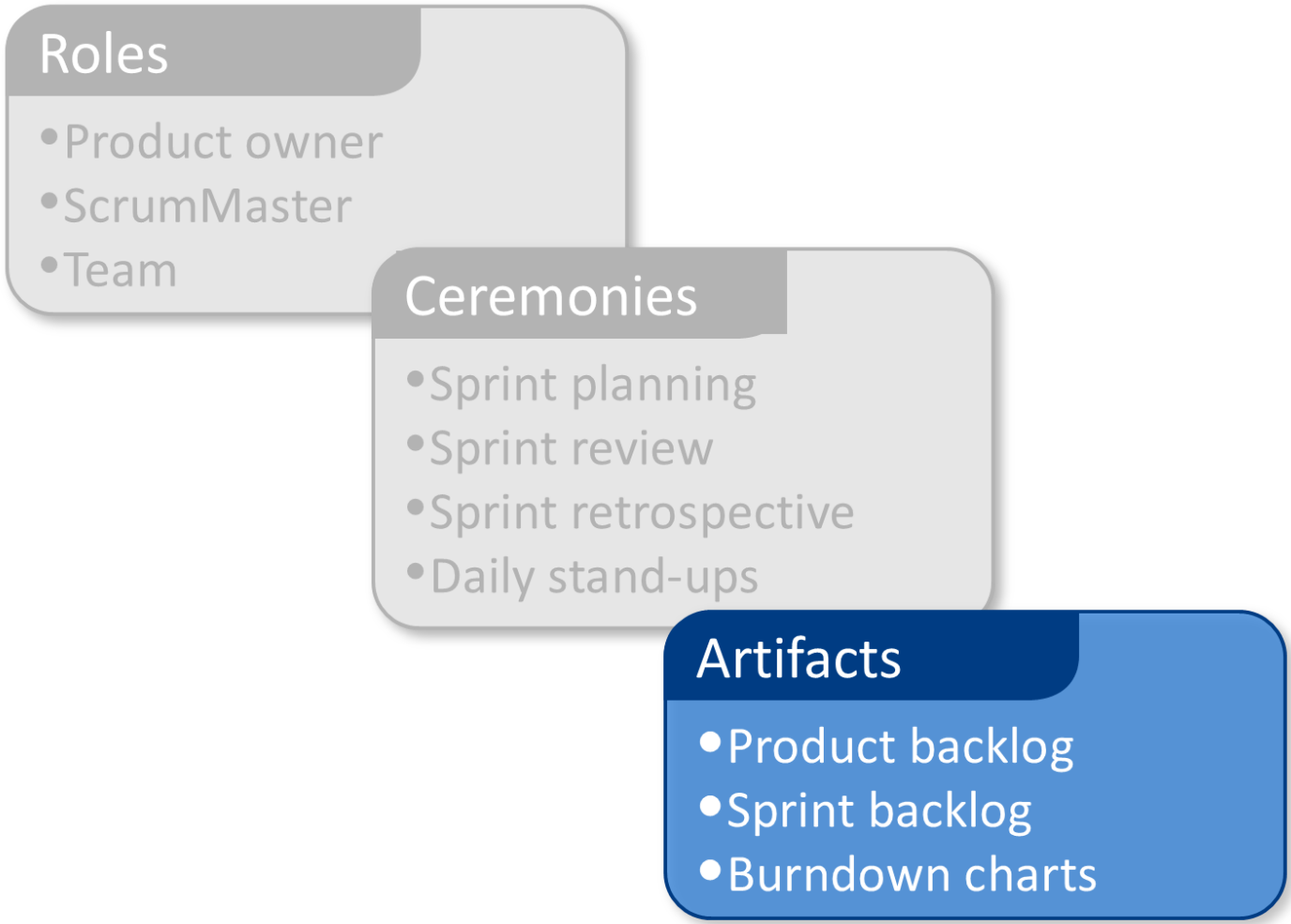
- Team presents what it accomplished during the sprint
- Typically takes the form of a demo of new features or underlying architecture
- Informal
- 2-hour prep time rule
- No slides
- Whole team participates
- Invite the world

Scrum Ceremonies / Meetings

Sprint Retrospective

- Periodically look at what is and isn't working
- Typically 30 minutes
- Done after every sprint
- Whole team participates:
  - ScrumMaster and Team
- Possibly Product Owner, customers and others (But generally NOT)
- Discuss what to:
  - Start Doing, Stop Doing and Continue Doing

Scrum Artefacts



Scrum Artefacts – Product Backlog

User Stories

- A User Story is a requirement expressed from the perspective of an end-user / customer of the system
- User stories shift the focus from writing about requirements to talking about them
- User stories are short, simple descriptions of a feature told from the perspective of the customer who wants the new capability of the system. They follow a simple template:
  - *As a < type of user >, I want < some goal > so that < some reason >*

Scrum Artefacts – Product Backlog

User Stories

- User stories are written at varying levels of detail.
- They can cover a large amounts of functionality such as this example from a Professional Membership Website:

  - As a site visitor, I want to get all information associated with my professional membership so that I have access to all information centrally.

- Because this level of detail is too large for an agile team to complete in one iteration, it is sometimes split into smaller user stories before it is worked on
- (Feature or epic user story)

Scrum Artefacts – Product Backlog

Story Points

- Story points are a unit of measure for expressing an estimate of the overall effort that will be required to fully implement a product backlog item or any other piece of work
- Story points help estimate how much work can be done in a sprint
- When estimating with story points, a value is assigned to each item. The raw values are unimportant, what matters are the relative values
- A story that is assigned a 2 should be twice as much as a story that is assigned a 1. It should also be two-thirds that is estimated as a 3 story point.
- Instead of assigning 1, 2 and 3, that team could assign 100, 200 and 300. Or 1 million, 2 million and 3 million. It is the ratios that matter, not the actual numbers

Scrum Artefacts – Product Backlog

Product Backlog

- **User Story 1**
- **User Story 2**
- **User Story 3**
- **User Story 4**
- **User Story 5**
- **User Story nn**

- The requirements
- A list of all desired work on the project
- Ideally expressed such that each item has value to the users or customers of the product
- Product Backlog User Stories are selected for a Sprint by Product Owner
- Reprioritised at the start of each sprint

Scrum Artefacts – Product Backlog Example - Professional Body Website

Product Backlog

- User Story 1
- User Story 2
- User Story 3
- User Story 4
- User Story 5
- User Story 6

News Section – Sprint 1

- User Story 1 - As a site visitor, I can read current news on the home page so that I stay current on key professional items
- User Story 2 - As a site visitor, I can email news items to the editor so that they can be considered for publication
- User Story 3 - As a site member, I can subscribe to an RSS feed of news so that I can stay current on the news that is of interest to me

Courses and Events – Sprint 2

- User Story 4 - As a site visitor, I can see a sorted list by date of all upcoming “Certification Courses” so that I can choose the best course for me
- User Story 5 - As a site visitor, I can see a list of all upcoming “Other Courses” (non-certification courses) so that I can choose the best course for me
- User Story 6 - As a site visitor, I can see a list of all upcoming “Social/Networking Events.” so that I can select ones I am able to attend

www.mountaingoatsoftware.com/agile/scrum/scrum-tools/product-backlog/example

Scrum Artefacts – Product Backlog Example - Professional Body Website

Product Backlog

- User Story 1
- User Story 2
- User Story 3
- User Story 4
- User Story 5
- User Story 6

News Section **- Total of 6 Story Points to complete Sprint 1**

- User Story 1 - As a site visitor, I can read current news on the home page so that I stay current on key professional items – *1 Story Points*
- User Story 2 - As a site visitor, I can email news items to the editor so that they can be considered for publication – *2 Story Points*
- User Story 3 - As a site member, I can subscribe to an RSS feed of news so that I can stay current on the news that is of interest to me – *3 Story Points*

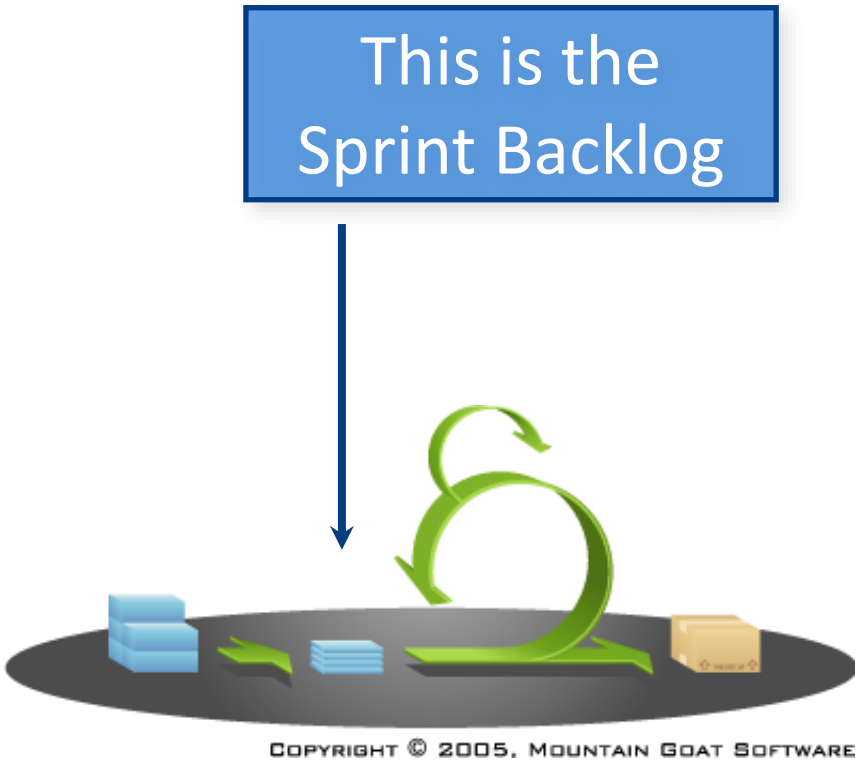
Courses and Events **- Total of 8 Story Points to complete Sprint 2**

- User Story 4 - As a site visitor, I can see a sorted list by date of all upcoming "Certification Courses" so that I can choose the best course for me – *2 Story Points*
- User Story 5 - As a site visitor, I can see a list of all upcoming "Other Courses" (non-certification courses) so that I can choose the best course for me – *5 Story Points*
- User Story 6 - As a site visitor, I can see a list of all upcoming "Social/Networking Events." so that I can select ones I am able to attend – *1 Story Points*

To complete this total Product Backlog would take 14 Story Points

www.mountaingoatsoftware.com/agile/scrum/scrum-tools/product-backlog/example

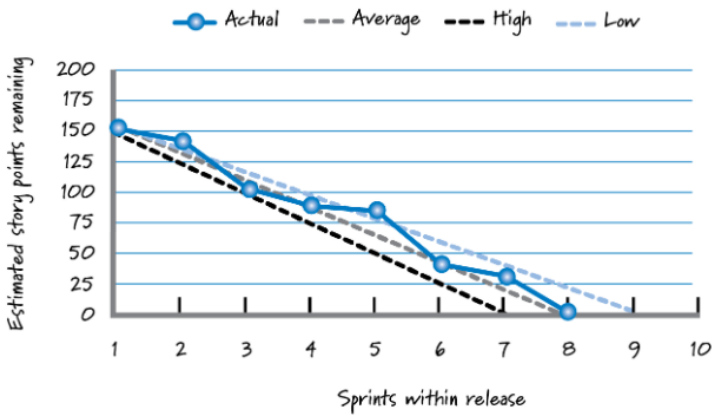
Scrum Artefacts – Sprint Backlog / User Story



- Scrum team decompose User Stories to Low level User Stories during Sprint Planning
- The User Stories are used for a conversation between the SME and developer. Developer updates the User Stories with the tasks and hours estimates, "Just-In-Time“
- Remaining estimated items are updated daily
- Sprint Backlog is seldom altered
- User stories in the sprint are either completed 100% or not done

Scrum Artefacts – Burn Down Chart

- A burn down chart is a graphical representation of work left to do versus time.
- The outstanding work (or backlog of user stories) is often on the vertical axis, with time along the horizontal.
- It is used to predict when all of the work will be completed.



Scrum Summary

- Another look at ***SCRUM***

https://www.youtube.com/watch?v=9TycLR0TqFA

Intended Learning Objectives

Software Development Lifecycles

Agile:

1. ~~Understand what Agile is and its origins.~~
2. ~~Understand the Agile framework.~~
3. ~~Understand Scrum – Roles, Ceremonies and Artefacts.~~
4. Understand advantages / disadvantages of Agile.

Agile – Advantages & Disadvantages

Advantages

- Customer satisfaction by rapid, continuous delivery of usable software
- People and interactions are emphasised rather than process and tools
- Continuous attention to technical excellence, good design and quality
- Regular adaptation to changing circumstances

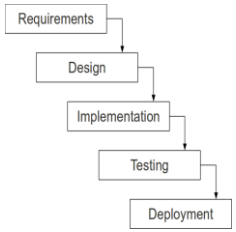
Disadvantages

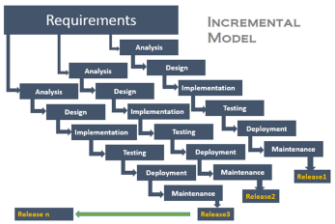
- Difficult to assess the effort required at the beginning
- Can be very demanding (from traditional approaches) on users time
- Harder for new starters to integrate into the team
- Agile is a very different approach – It can be intense for the team
- Requires experienced resources (which are limited in today's market)

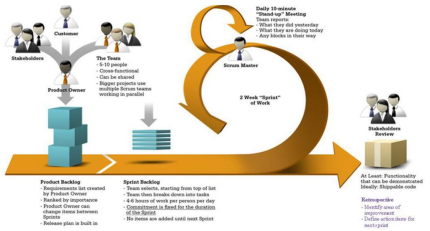
Formal or Agile which one Should I use???

There is no one right answer. The following questions can assist deciding:

- How Stable Are the Requirements?
- Do the end users need to collaborate?
- Is the Time Line Aggressive or Conservative
- What Is the Size of the Project
- Where Are the Project Teams Located
- What Are the Critical Resources?







Agile Where to find out more information

- *www.agilealliance.org*
- *www.mountaingoatsoftware.com/scrum*
- *www.scrumalliance.org*
- *www.controlchaos.com*

THE UNIVERSITY OF MELBOURNE

SWEN90016
Software Processes & Project Management

SDLCs Case Study

Marion Zalk
Department of Computing and Information Systems
The University of Melbourne
mzalk@unimelb.edu.au

2021 – Semester 2
Lecture 2

Background

THEN

Early ’90s, sending and receiving payments online between individuals was not an easy process.

People didn’t even imagine the possibility of such transactions, even though banks and other financial institutions had the means of conducting them.

NOW

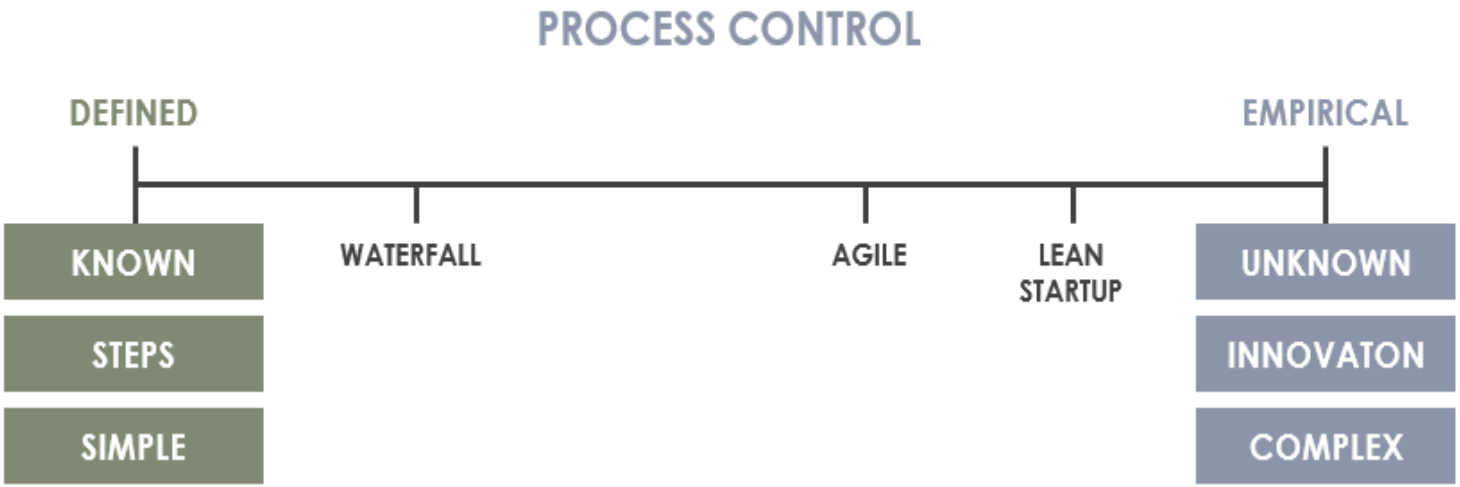
Used in 200+ countries in the world.

50+ curriences

Empirical and Defined Process

A process is a series of progressive and interdependent steps by which an end is attained.

Empirical process control expects the unexpected, while defined process control expect every piece of work to be completely understood in upfront.



www.visual-paradigm.com/scrum/empirical-vs-defined-process-control

PayPal Case Study

How it started……

1998: Initial product created to encrypt word documents

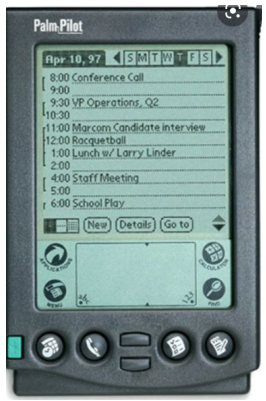
But no one wanted to keep word documents secret

Encrypted money transfer between PalmPilots

The **first tool** to transfer money electronically

Built web interface as a sales demo tool **ACCIDENT**

PalmPilot device not popular, so demo-ed the tech on the web



PayPal Case Study

A success story

1999: Encrypted money transfer **service** component

2000: Merge with an online bank (Elon Musk- X.com)

PayPal Case Study

A success story

2001: Changed name to PayPal (Confinity + X.com)

2002 Listed on NASDAQ; Acquired by Ebay (1.5bn)

Slow down

2002 -

Company was small, they could rapidly build solutions which put them at the leader

Innovation began to decrease, release cycles increased from weeks to months and many months

2008 - 6 key drivers for the slow down

1. Annual Planning –defined and documented in a Product Requirements Document (PRD) which could take months and were often out of date before coding began.

[1]

Slow down

2. Complex quarterly planning
New process for planning to allow for more flexibility, added complexity

3. Domain bottlenecks
Divided the environment into 85 domains, unintended consequence was a bottleneck at certain key areas (necessary for every project). These included risk, compliance, payments.

4. Developer context switching
To enable more projects to kick-off, developer were assigned to multiple projects during planning. This context switching reduced productivity of the individual and team.
[1]

Slow down

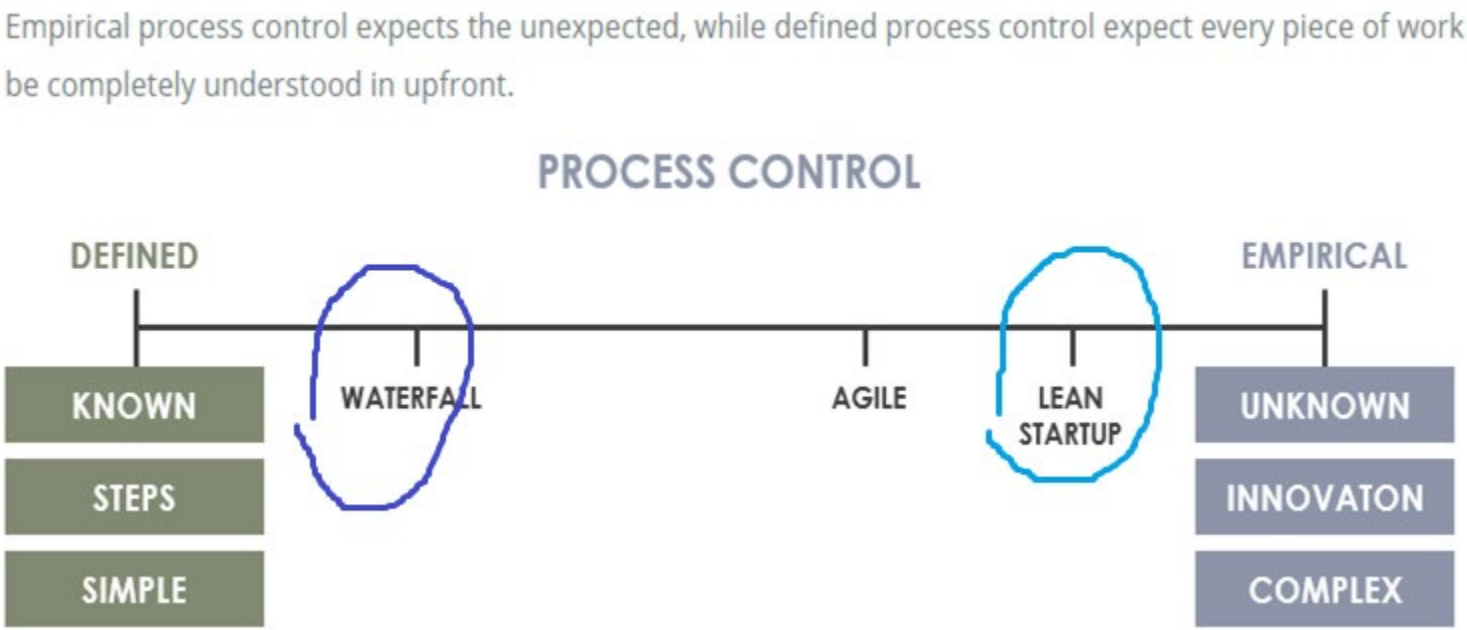
5. Traditional waterfall development
To facilitate and coordinate efforts across the organisation, they used waterfall. Those teams that had developed using grass root agile efforts were forced to follow the strict process of Waterfall.

6. Integration testing cycles
To ensure quality of each release- rigorous testing required, code branches grew and integration and testing cycles increased even for the smallest change- 6 weeks of integration testing cycle to ensure it didn't break the site.

[1]

Empirical and Defined Process

A process is a series of progressive and interdependent steps by which an end is attained.



www.visual-paradigm.com/scrum/empirical-vs-defined-process-control

Results

People (developers) were frustrated.

Exec management was frustrated.

Customers were frustrated

What happened?

[1]

Results

2012

PayPal promoted David Marcus- President
Focus was on the customer

Kirsten Wolberg joined as Lead Technology Business Operations - *Agile*
(had experience with Agile as CIO at salesforce.com)

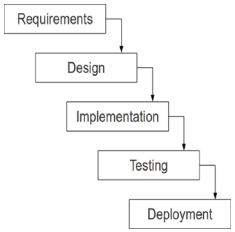
Alignment executive sponsorship and leadership with the employees and this enabled PayPal to embark on a *radical transformation.*

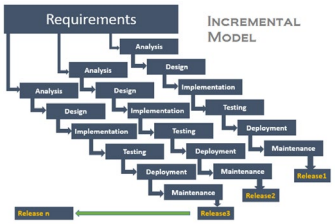
[1]

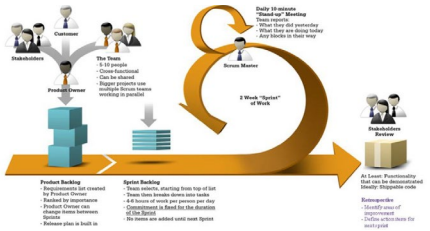
Formal or Agile which one Should I use???

There is no one right answer. The following questions can assist deciding:

- How Stable Are the Requirements?
- Do the end users need to collaborate?
- Is the Time-Line Aggressive or Conservative
- What Is the Size of the Project
- Where Are the Project Teams Located
- What Are the Critical Resources?



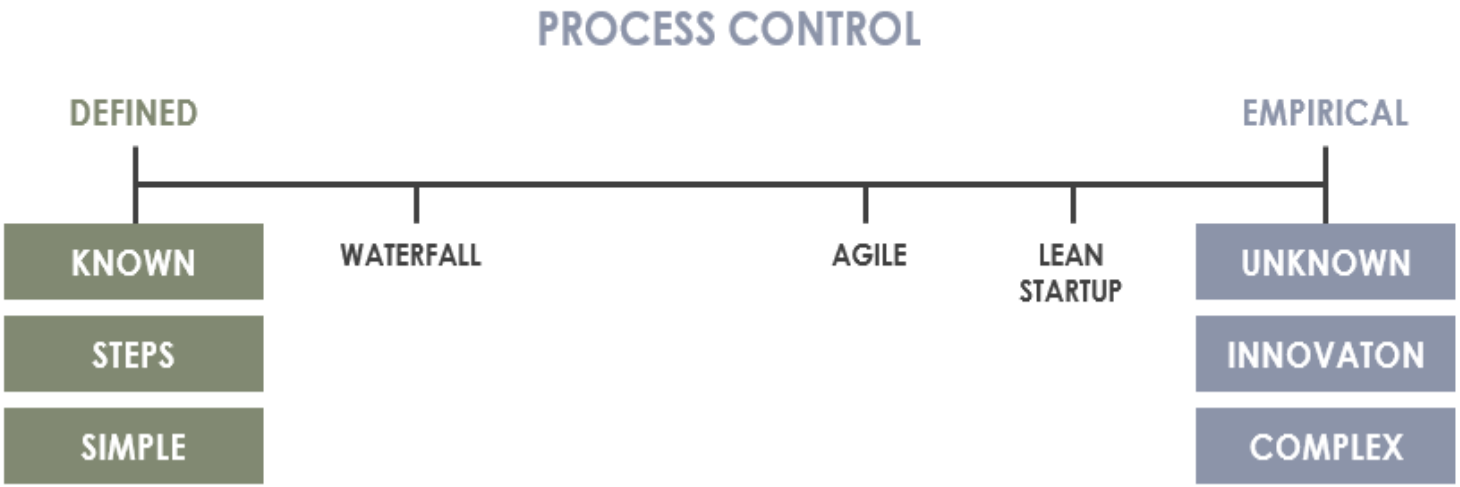




Open discussion

Let's think of some projects…..

Empirical process control expects the unexpected, while defined process control expect every piece of work to be completely understood in upfront.



www.visual-paradigm.com/scrum/empirical-vs-defined-process-control

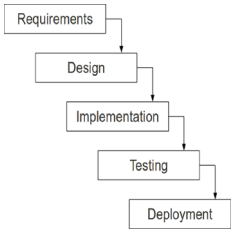
References

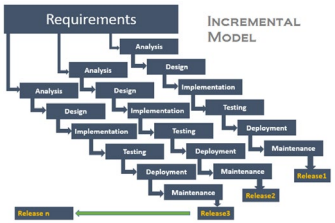
[1] (10 March 2021) PayPal Enterprise Transformation
Available:
https://www.paypalobjects.com/webstatic/en_US/mktg/pages/stories/pdf/paypal_transformation_whitepaper_sept_18_2015.pdf

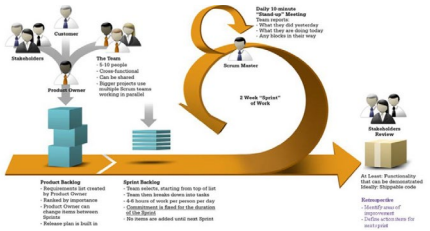
Wagile

- Waterfall- Agile (Hybrid) methodology
- Agile and waterfall are two distinctive **methodologies of processes to complete projects or work items**.
- May adopt a few Agile practices on top of a waterfall model, without significantly altering the waterfall model.
- Agile is an iterative methodology that incorporates a cyclic and collaborative process.
- Waterfall is a sequential methodology that can also be collaborative, but tasks are generally handled in a more linear process.

https://www.thoughtworks.com/en-au/insights/blog/good-and-bad-wagile







PRINCE2 and PMBOK

- PRINCE2 (Projects in a Controlled Environment) is a defined methodology with roles and responsibilities and deliverables (IT government contracts)
- PMBOK Guide is essentially a collection of good PM principles, techniques and guidelines that help you manage projects.
- PMBOK also provides guidelines on procurement (contract management, scope management).
- PMBOK, the project manager can seemingly become the primary decision maker, planner, problem solver, human resource manager and so on. Focus on soft skills.
- PRINCE2 aiding the project manager to oversee projects on behalf of an organisation's senior management (shares more of the functional and financial authority with senior management, not just the project manager).

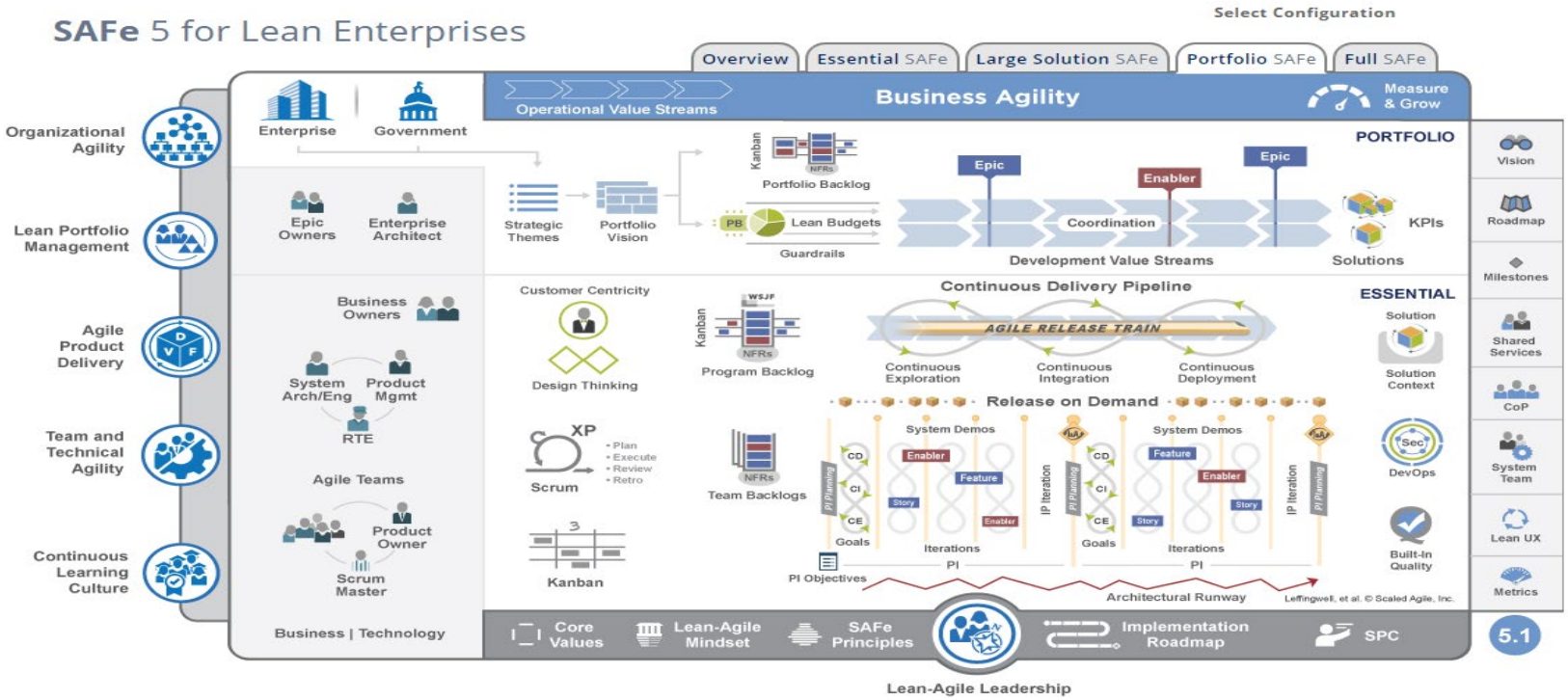
https://www.researchgate.net/profile/Joao-Miguel-Cotrim-2/publication/272148384_PRINCE2_Vs_PMBOK/links/54db6bf40cf233119bc62270/PRINCE2-Vs-PMBOK.pdf

SAFagile

- The **Scaled agile framework** (**SAFe**) is a set of organization and workflow patterns
- Guide enterprises in scaling lean and agile practices
- SAFe is one of a growing number of frameworks that seek to address the problems encountered when scaling beyond a single team.

https://en.wikipedia.org/wiki/Scaled_agile_framework

SAFagile



https://www.scaledagileframework.com/