Software Project Management - Laboratory

Lecture n° 1
A.Y. 2021-2022

Prof. Fabrizio Fornari

Who is Fabrizio Fornari?

2020 Postdoc in Computer Science at UNICAM

2018 PhD title in Computer Science at UNICAM. 3 months in Brisbane Queensland University of Technology (Australia)

2012-2013 Master's degree in Computer Science at UNICAM and University of Reykjavik (Iceland)

2010-2011 Bachelor degree in Computer Science at UNICAM

Collaboration

PROS Lab - PROcesses and Services Laboratory http://pros.unicam.it/

Past collaborations:

- National Research Center of Pisa (CNR)
- Apromore group https://apromore.org/ (University of Melbourne)
- Technical University of Denmark
- Sant’Anna School of Advanced Studies (Pisa)

My Research Topics

Business Process Management
Business Process Modeling and Verification
Business Process and IoT
Model-Driven Engineering for IoT

Developed Software Tools:

- BProVe, Business Process Verifier
- BEBoP, understandaBility vErifier for Business Process models
- RePROSitory, Repository of open PROcess models and logS

Teaching

2020-2021 & 2021-2022

- Software Project Management Laboratory at “University of Camerino”, Department of Computer Science (6CFU)

2019-2020

- Computer Science at “Università di Macerata”, Faculty of “Economia e diritto” (6CFU)

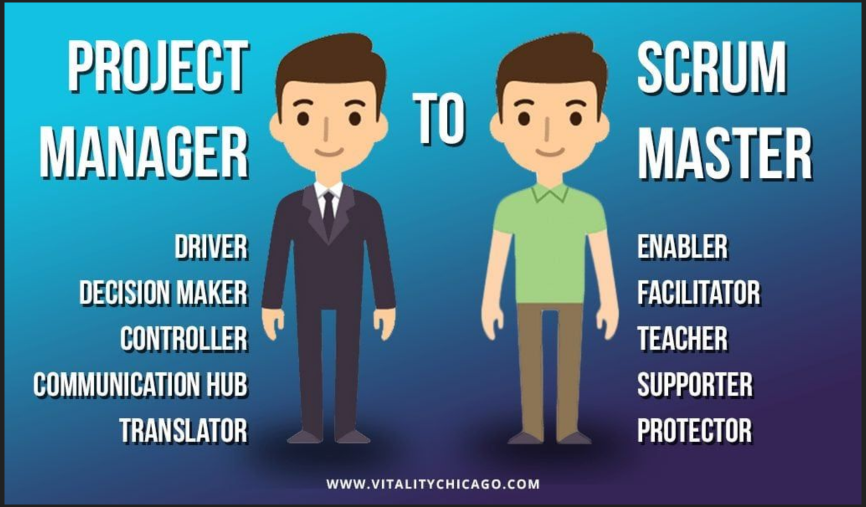
2018-2019

- Computer Science at “Università di Macerata”, Faculty of “Economia e diritto” (6CFU)
- Software Project Management Laboratory at “University of Camerino”, Department of Computer Science (3CFU)

Group Projects

I supervised/co-supervised a dozen of group projects and experimental thesis.

I try to apply together with the students the methodology that we will see during the course.



Course Overview

Course Objective

The course introduces the students to the basic knowledge of complex software system production following the **DevOps methodology**.

Prerequisite knowledge

- Basic Programming Experience
- Basic Software Engineering Methods and Techniques

Learning Outcome

The student will be able to manage the organization and the development of a software applying DevOps methodology.

Course Overview

Website:

- http://didattica.cs.unicam.it/doku.php?id=didattica:ay2122:spm:main

Teaching Hours:

- Thursday 9am-11am (Lab)
- Friday 11am - 1pm (Lab)

Students Meeting:

- After each lesson or,
- By requesting a meeting by sending an email to *fabrizio.fornari@unicam.it*
- My desk is in Building n°2 of the Computer Science Department

Note: only email coming from the @studenti.unicam.it domain will be processed.

Technical Requirements

- None!

But actually..

- Every Computer Science Background will help
- Your attitude is what matters the most

Exam

The exam involves the development of a software project by following the methodologies introduced during lecture hours.

Exam Evaluation:

- Git Usage
- Testing
- SCRUM application/Sprint management
- Presentation
- Overall Project

https://docs.google.com/spreadsheets/d/1Ysy8JWuopAN_Tt_PnjVwlaUK0K3rWcnyAh2K-mdoEMY/edit?usp=sharing

Projects

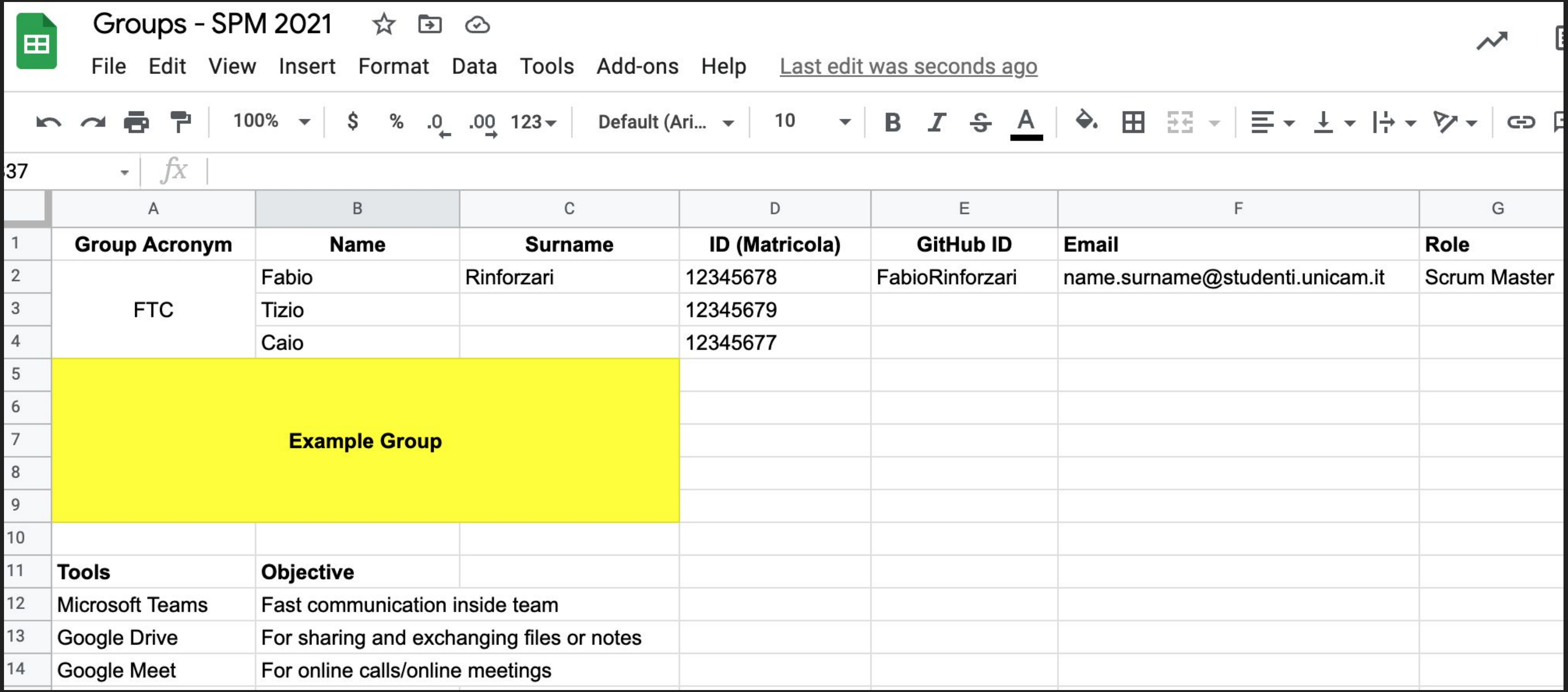
You will have to choose between 4 different projects.

Group Formation

- Group of 2 or 3 students have to be formed
- A Project will be assigned
- A GitHub account is needed for every student
- A GitHub repository will be assigned to each group

Define Groups

https://docs.google.com/spreadsheets/d/1tOdMoBKzBjcHGQI3ACmrkkcaFbc2TuPfgmFQM2jL9BU/edit?usp=sharing


Groups - SPM 2021
File Edit View Insert Format Data Tools Add-ons Help
Last edit was seconds ago
100%
Default (Ari...
10

	Group Acronym	Name	Surname	ID (Matricola)	GitHub ID	Email	Role
1	Group Acronym	Name	Surname	ID (Matricola)	GitHub ID	Email	Role
2	FTC	Fabio	Rinforzari	12345678	FabioRinforzari	name.surname@studenti.unicam.it	Scrum Master
3		Tizio		12345679			
4		Caio		12345677			
5–9	Example Group						
10							
11	Tools	Objective					
12	Microsoft Teams	Fast communication inside team					
13	Google Drive	For sharing and exchanging files or notes					
14	Google Meet	For online calls/online meetings					

Questionnaire

https://docs.google.com/forms/d/e/1FAIpQLSdsRrooafPFRHEmIx18r1JG78ECpiPNACMETE73Mez9_bRRSQ/viewform?usp=sf_link

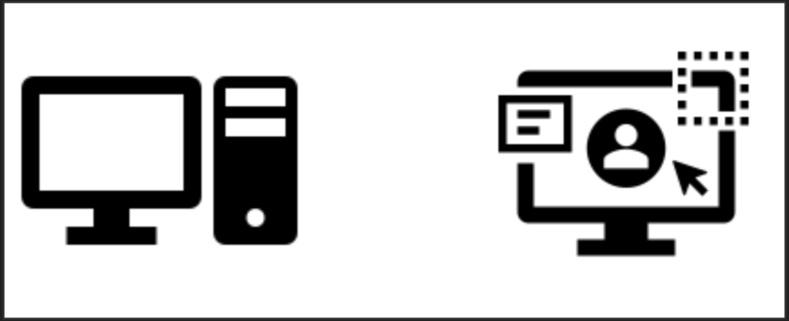
Course Introduction

Why do we study software project management?

Course Context

Hardware

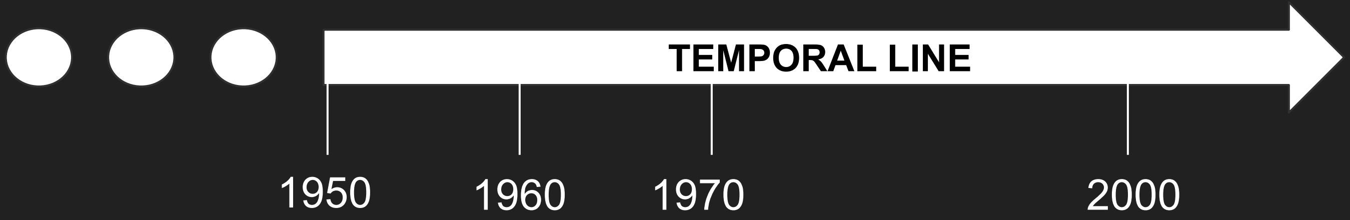
Software



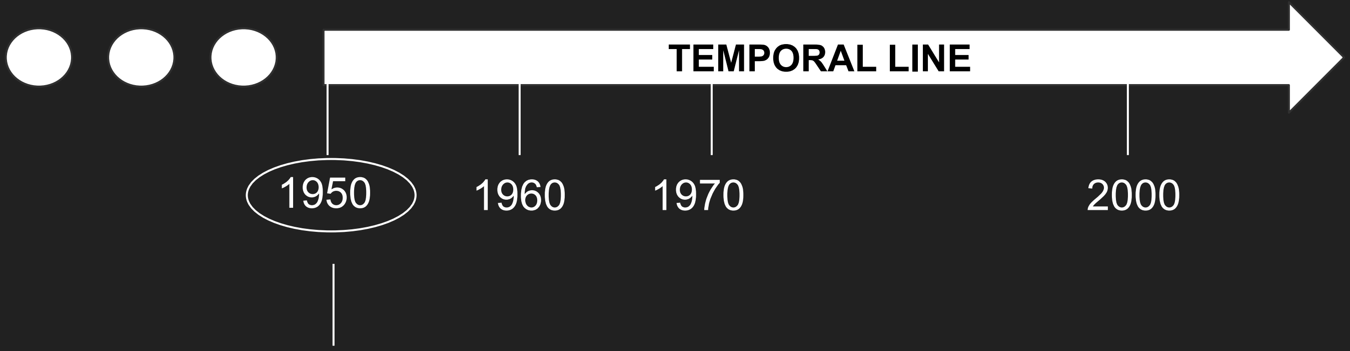
The physical part

The logical part

Long Story Short

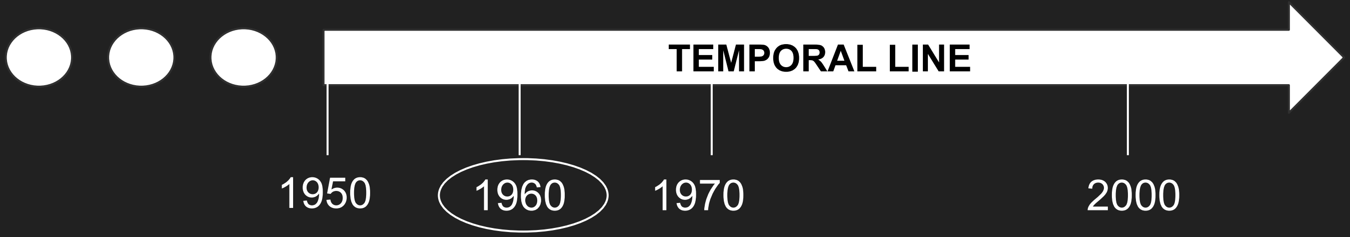


Long Story Short



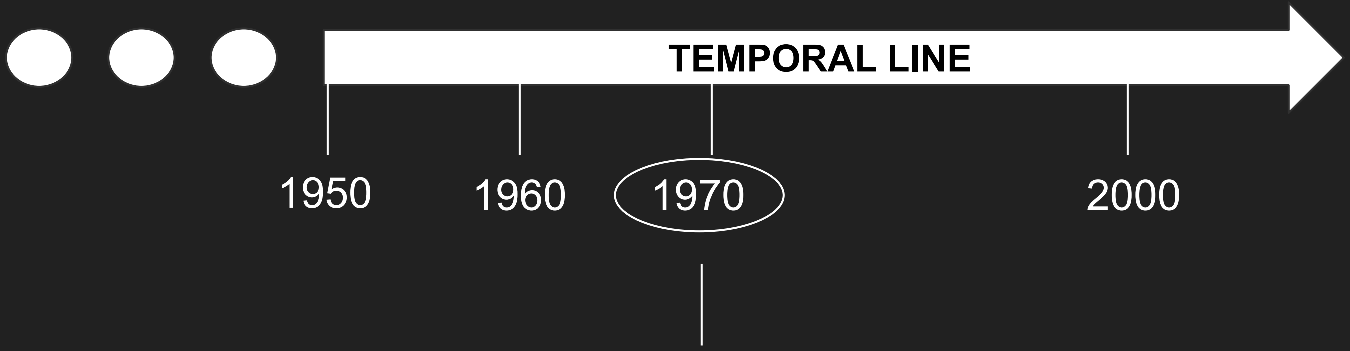
- Diffusion of the firsts computers
- Beginning of the “coding problem”
- Users and Hardware resellers are the main developers

Long Story Short



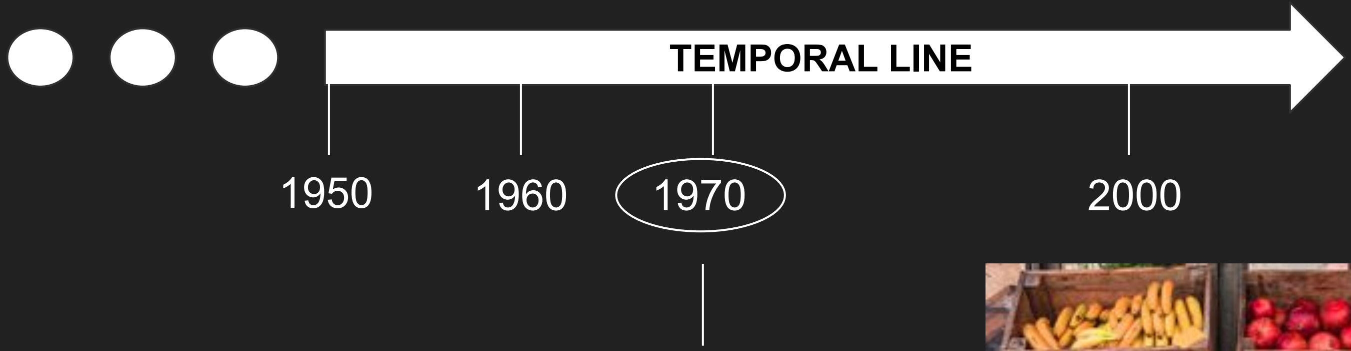
- Late ‘50s - early ‘60s
- Being a developer becomes a profession
- The activity to develop high-level languages begins
- Very few complex software projects are being developed

Long Story Short



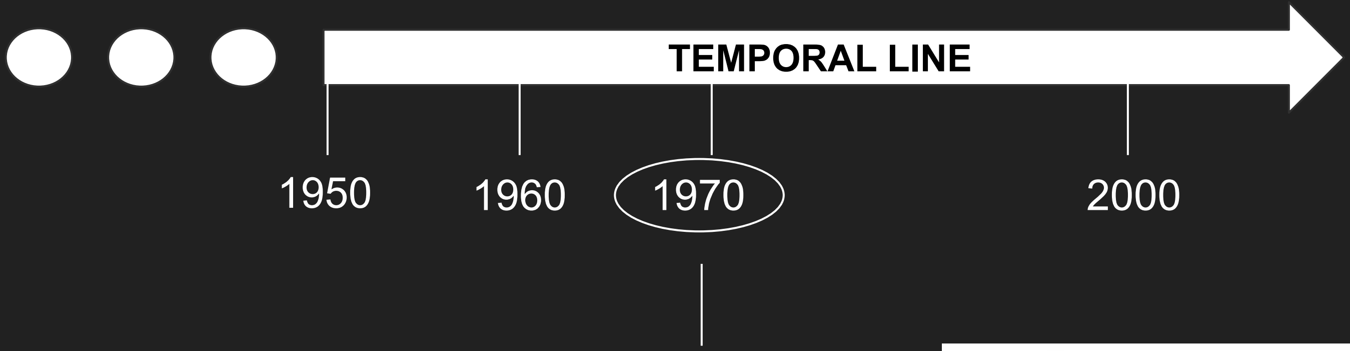
- Late ‘60s
- First attempt to develop complex software projects (e.g. operating systems such as the IBM 360)
- Increased complexity of systems

Long Story Short

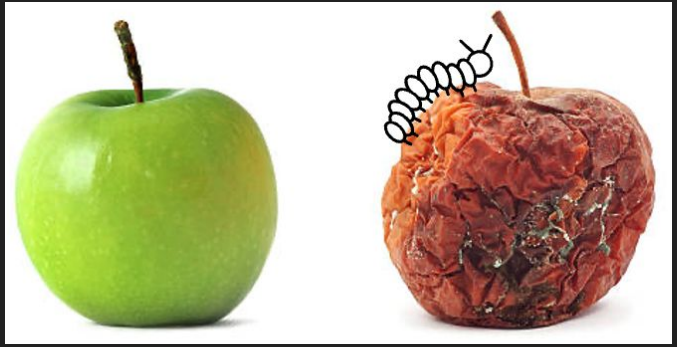


- Late ‘60s
- First attempt to develop complex software projects (e.g. operating systems such as the IBM 360)
- Increased complexity of systems
- Software Market Begins

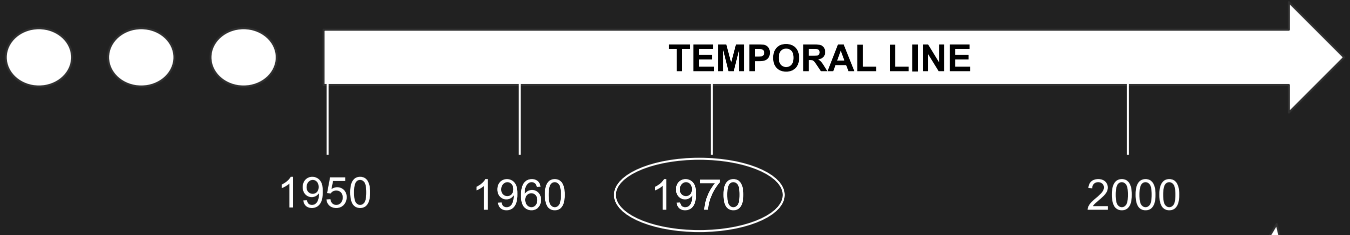
Long Story Short



- Late ‘60s
- First attempt to develop complex software projects (e.g. operating systems such as the IBM 360)
- Increased complexity of systems
- Software Market Begins
- Quality Requirements



Long Story Short



- Late ‘60s
- Projects running out of budget
- Projects running late
- Low Quality Software
- Software not compliant with the requirements
- Unmanageable projects, code too difficult to maint

Software Crisis

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

*..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 had become an equally **gigantic problem**.*

...To put it in another way: as the power of available machines grew by a factor of more than a thousand, society’s ambition to apply these machines grew in proportion, and it was the poor programmer who found his job in this exploded field of tension between ends and means. The increased power of the hardware, together with the perhaps even more dramatic increase in its reliability, made solutions feasible that the programmer had not dared to dream about a few years before. And now, a few years later, he had to dream about them and, even worse, he had to transform such dreams into reality!”

-Edsger Dijkstra

<u>Software Development was only about coding</u>

An answer to the Software Crisis

- Recognising that developing software is a complex process similar to those that generates engineering products (Software Development Process)
- The birth of Software Engineering

Requirements Definition

System Design

System Implementation

Testing

Deployment

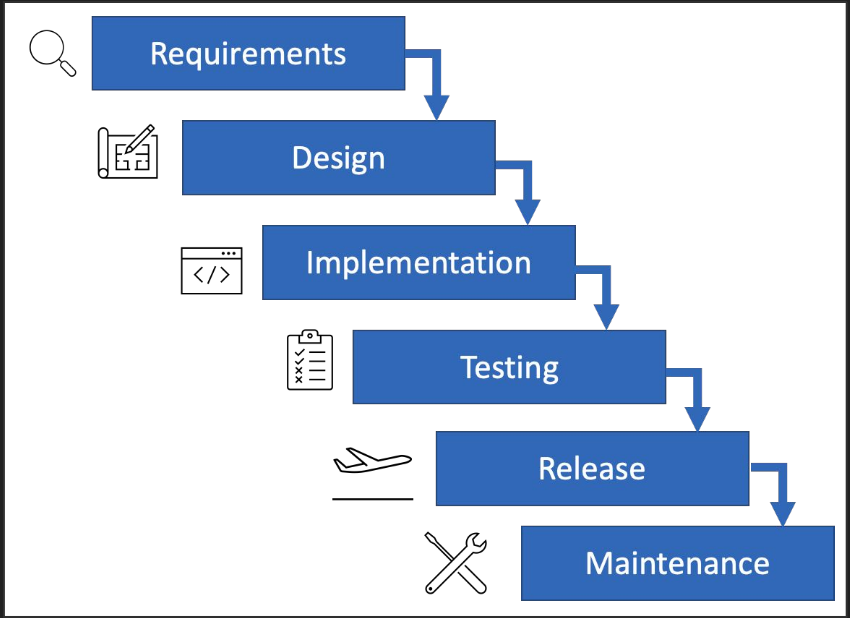
Maintenance

Bug Fixing

Software Development Process

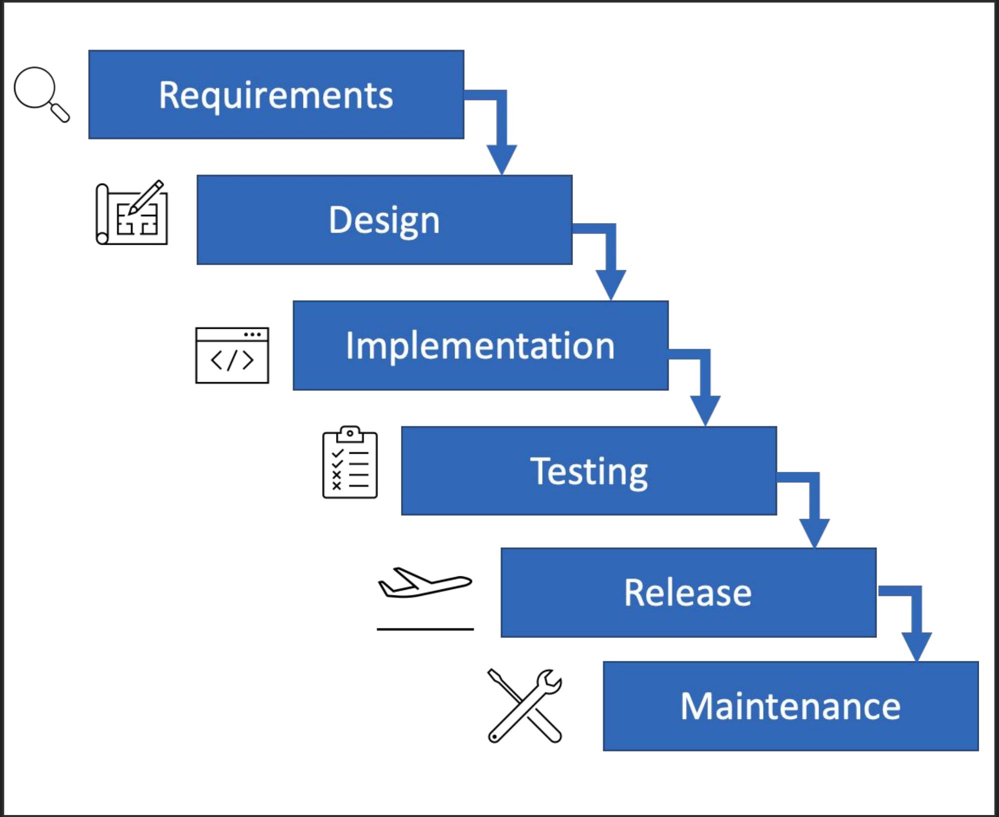
Software Development Process is the process of dividing software development work into distinct phases to improve design, product management, and project management. It is also known as a software development life cycle (SDLC)

Waterfall Model



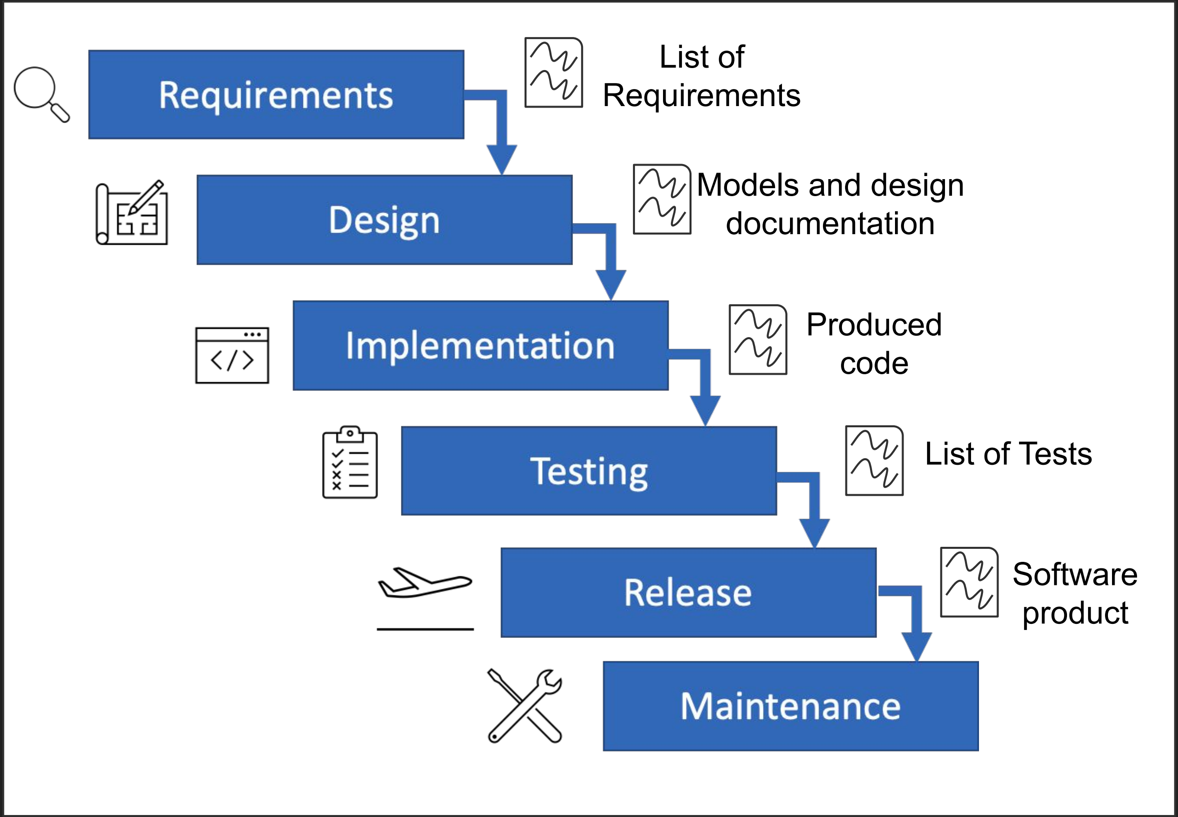
Waterfall Model

- Each phase is separated from the other
- Staff dedicated to each phase

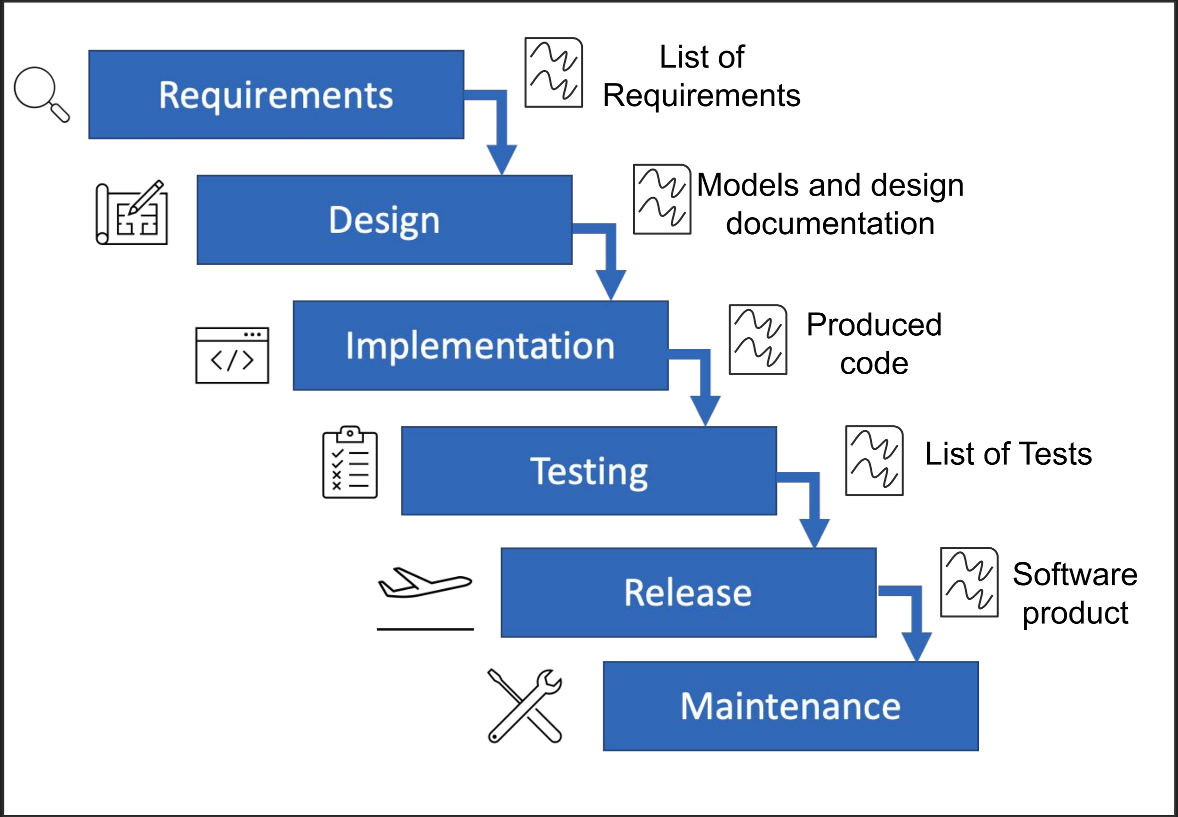


Waterfall Model

- Guided by the production of documents
- Progress measurable based on the amount of documentation produced
- Documents to support personnel changes

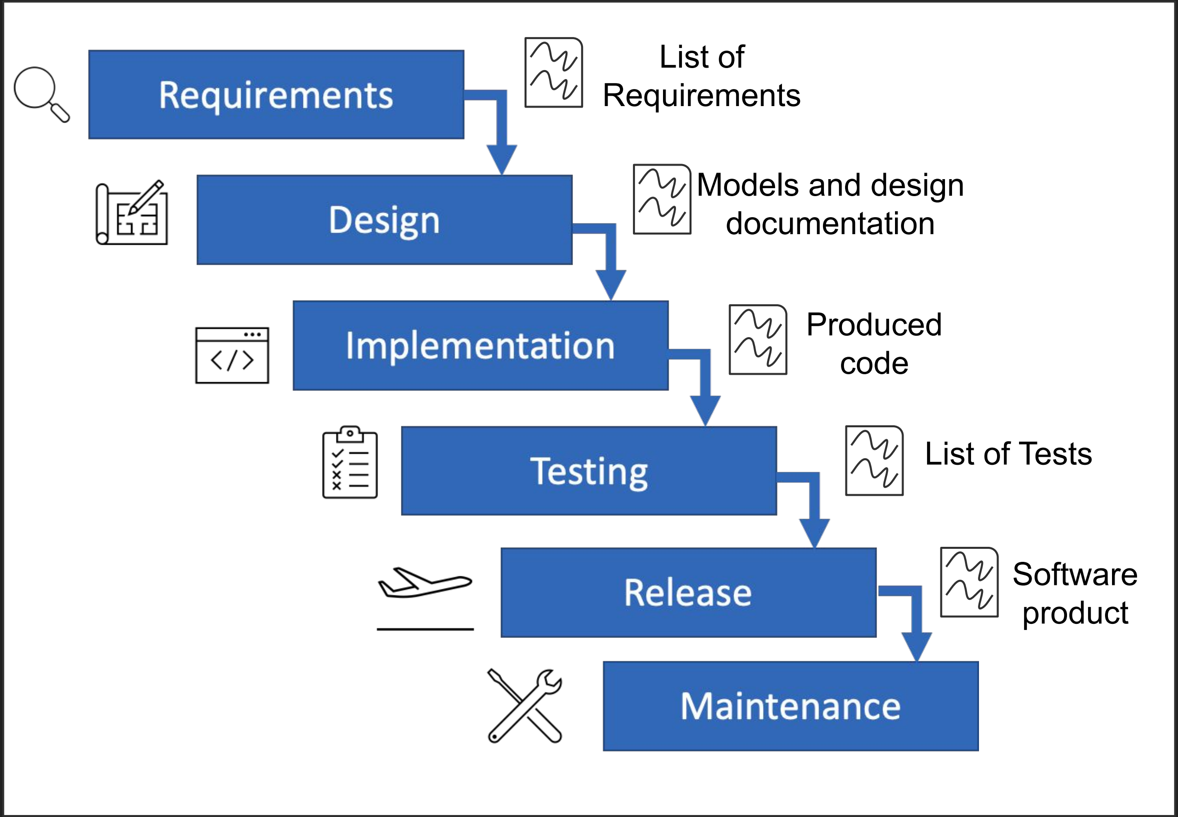


Waterfall Model - Negative Aspects ?

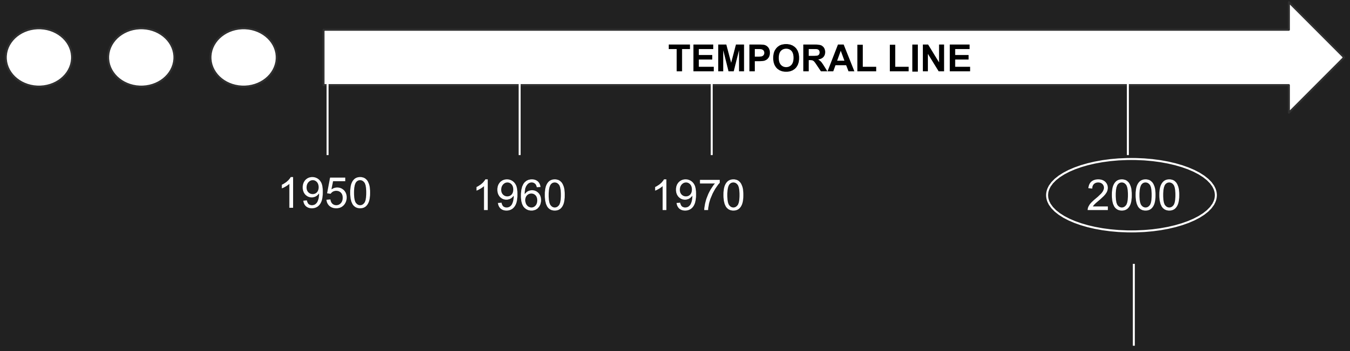


Waterfall Model - Negative Aspects

- Too much focused on the production of documents and less on the actual software product
- Software is released only at the end
- Customer involved only during the initial phase (requirements definition)
- Changings in the requirements are not possible after the requirements phase is over



Long Story Short



- The birth of the Agile paradigm
- Release of the Manifesto for Agile Software Development

The Agile Manifesto

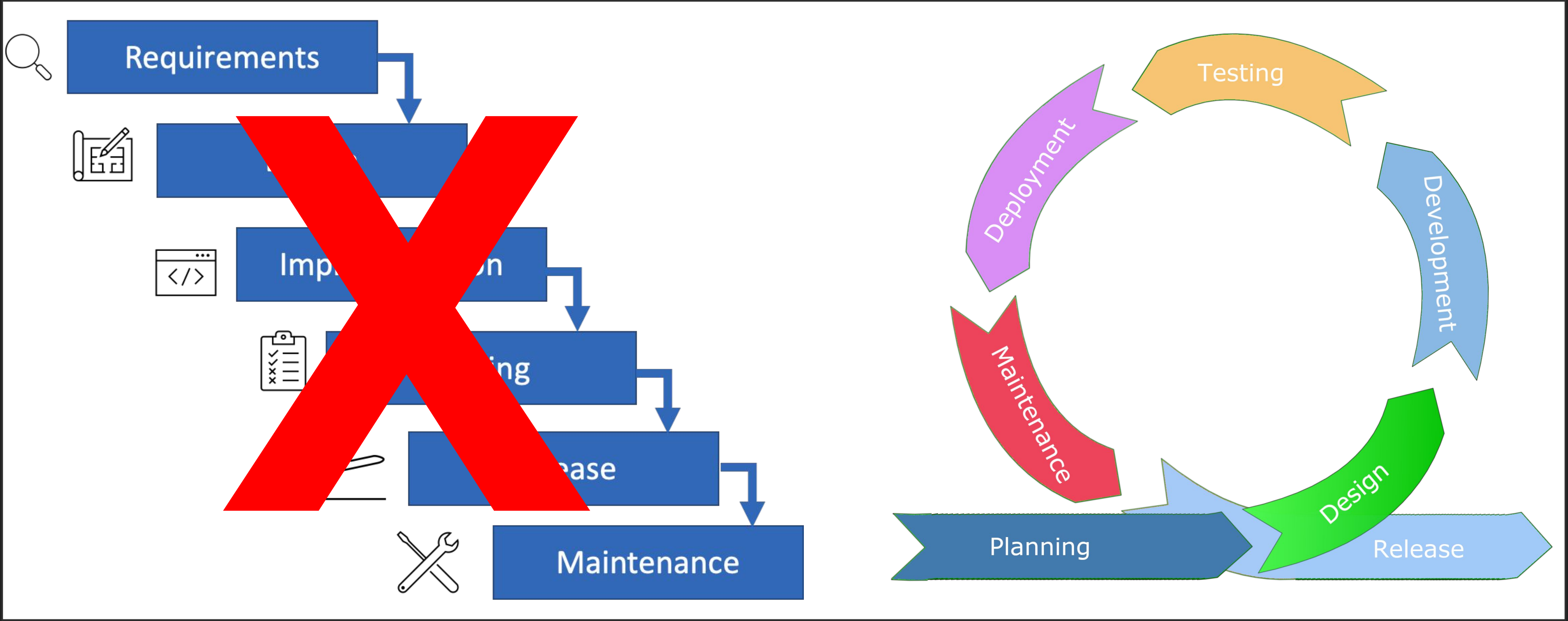
Individuals Interactions	>	Processes Tools
Working Software	>	Comprehensive Documentation
Customer Collaboration	>	Contract Negotiation
Responding to Change	>	Following a Plan

Manifesto: https://agilemanifesto.org/

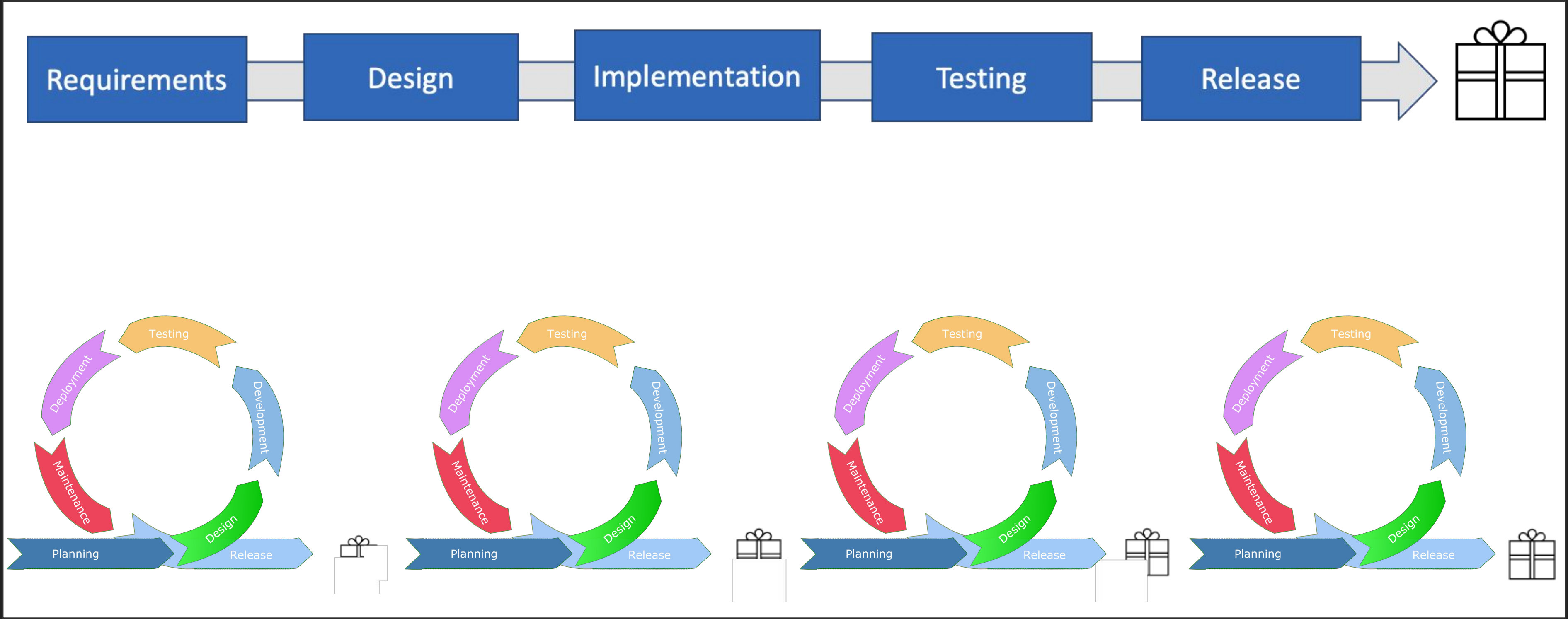
Agile Model

- Complex work divided into small parts
- Big companies divided into small teams
- Long projects divided into lists of tasks to be fulfilled in short amount of time
- Requirements can change in anytime

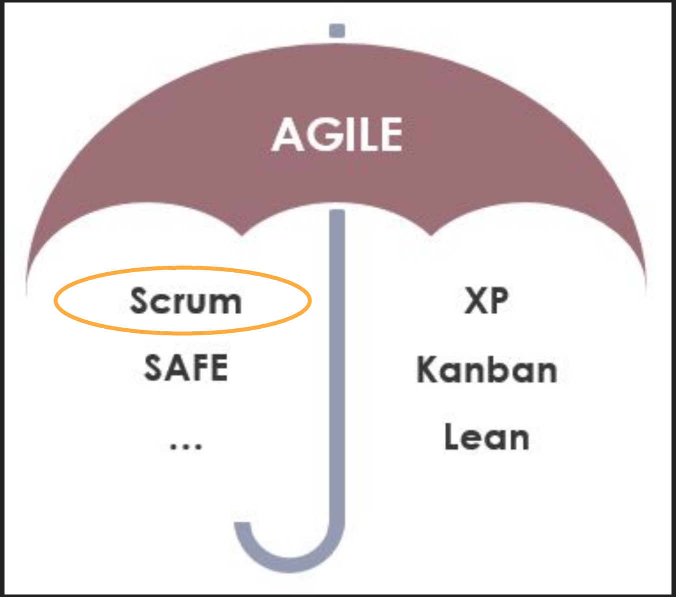
Software Development Process



Waterfall vs Agile

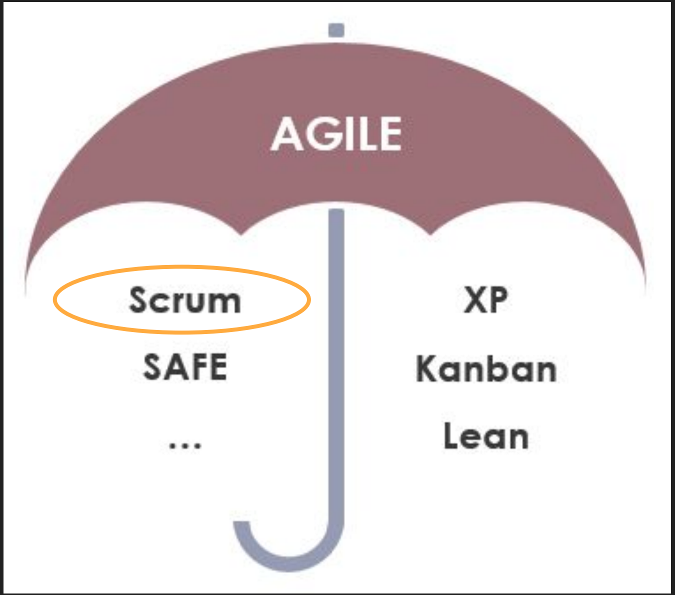


Under the term Agile



SCRUM

Scrum is an Agile framework for project management that emphasizes teamwork, accountability and iterative progress toward a well-defined goal.



Schwaber, K. (1997). Scrum development process. In *Business object design and implementation* (pp. 117-134). Springer, London.

SCRUM

Individual and Interactions

Working Software

Collaboration with Customer

Requirements are likely to change or not known at the start of the project.

Scrum is:

- Lightweight
- Simple to understand
- Difficult to master

SCRUM

Benefits:

- Better Quality Products
- Reduced time to market
- Increase Return on Investment
- Higher Team Morale
- Enhance Team Collaboration

SCRUM

The main components of the Scrum Framework are:

- **Roles**
- **Artifacts**
- **Events**
- **Sprint**

Roles

Product Owner - is responsible for working with the user group to determine what features will be in the product release. Some of the responsibilities:

- Develop the direction and strategy for the products and services, including the short and long-time goals;
- Provide or have access to knowledge about the product or the service;
- Understand and explain customer needs for the Development team;

Scrum Master - is the facilitator for an agile development team. Some of the responsibilities:

- Act as a coach, helping the team to follow scrum values and practices;
- Help to remove impediments and protect the team from external interferences;
- Promote a good cooperation between the team and stakeholders;

Scrum Team - is formed by 3 to 9 people who MUST fulfill all technical needs to deliver the product or the service. They will be guided directly by the Scrum Master, but they will not be directly managed. They must be self-organized, versatile, and responsible enough to complete all required tasks.

Artifacts

The SCRUM artifacts are used to help define the workload coming into the team and currently being worked upon the team.

The main artifacts:

- **Product backlog** - a collection of user stories which present functionalities required/wanted by the product team. Usually the product owner takes responsible for this list.

- **Sprint backlog** - a collection of stories which could be included in the current sprint.

User Stories

A User Story is a simple and quick description of a specific way that the user will use the software. Generally between one and four sentences long.

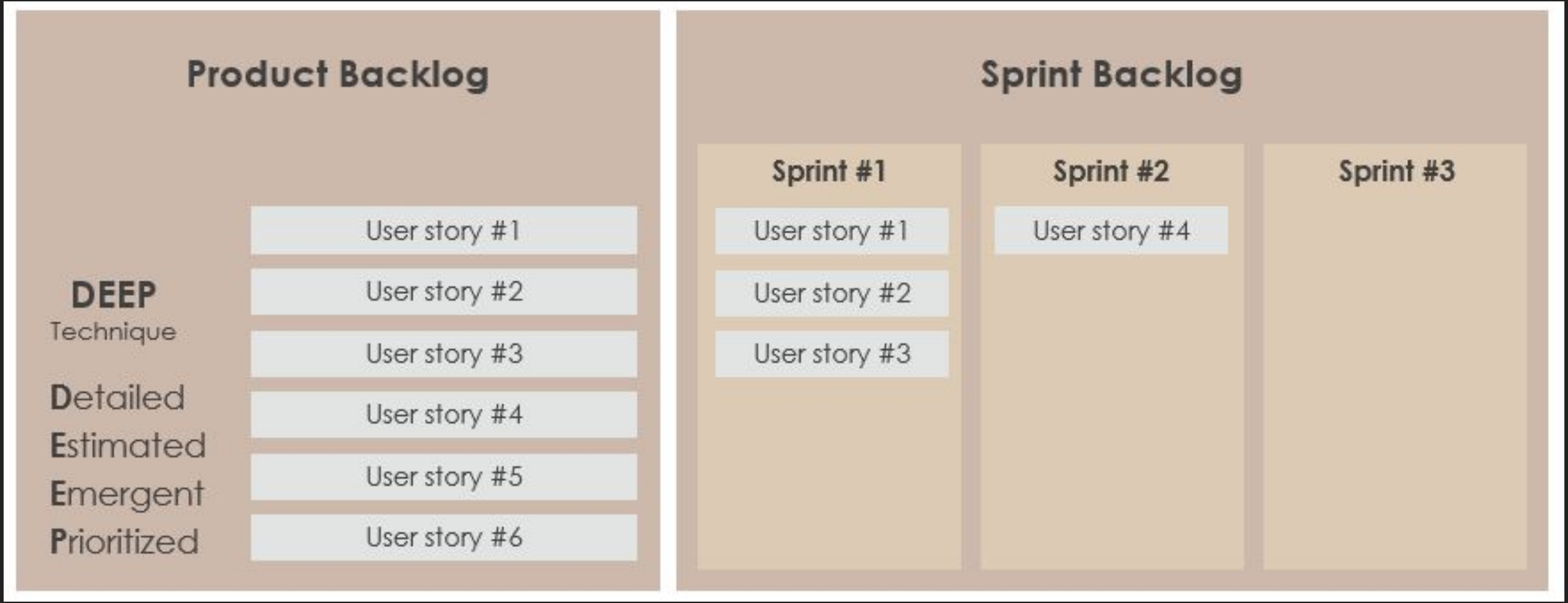
Can generally follow a template:

As a <*type of user*>,
I want to <*specific action I'm taking*>
so that <*what I want to happen as a result*>.

e.g. "As a customer, I want to be able to create an account so that I can see the purchases I made in the last year to help me budget for next year."

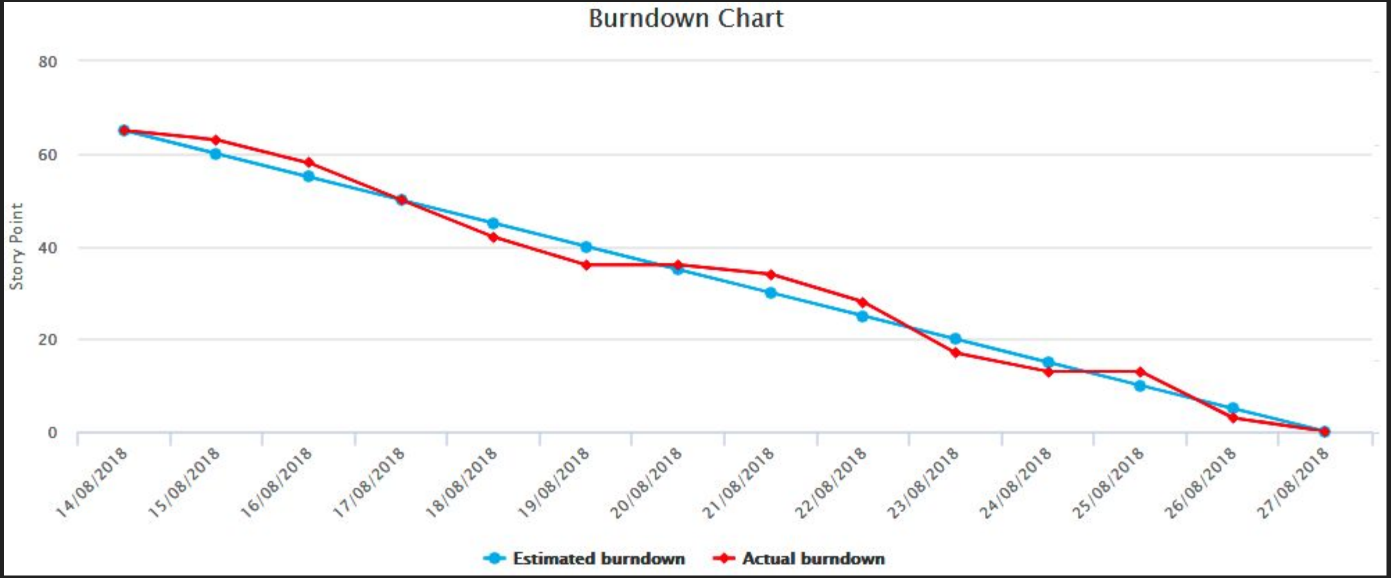
Assign a value to estimate the effort needed to elaborate a user story (e.g., 1 to 5).

Artifacts: Product Backlog and Sprint Backlog

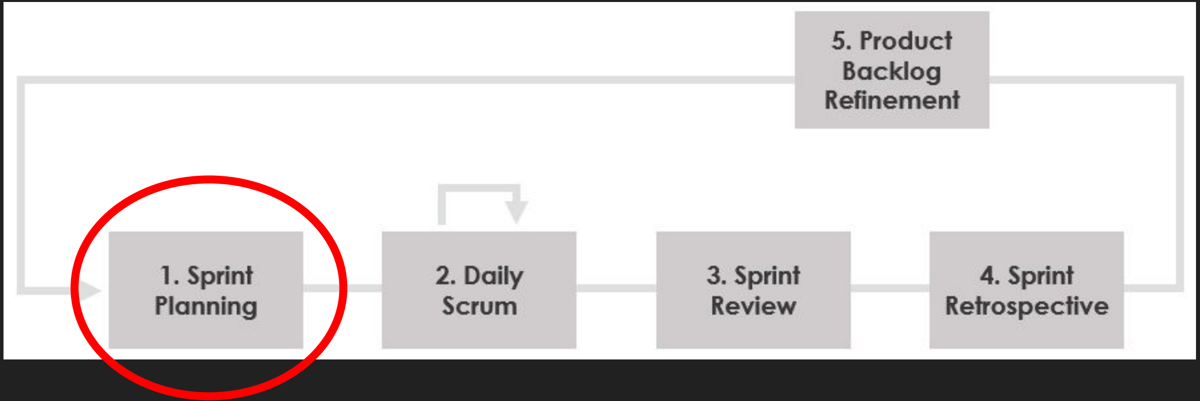


Artifacts: Burn-down chart

A burn-down chart is a graphical representation of work left to do versus time.
It is useful for predicting when all of the work will be completed.

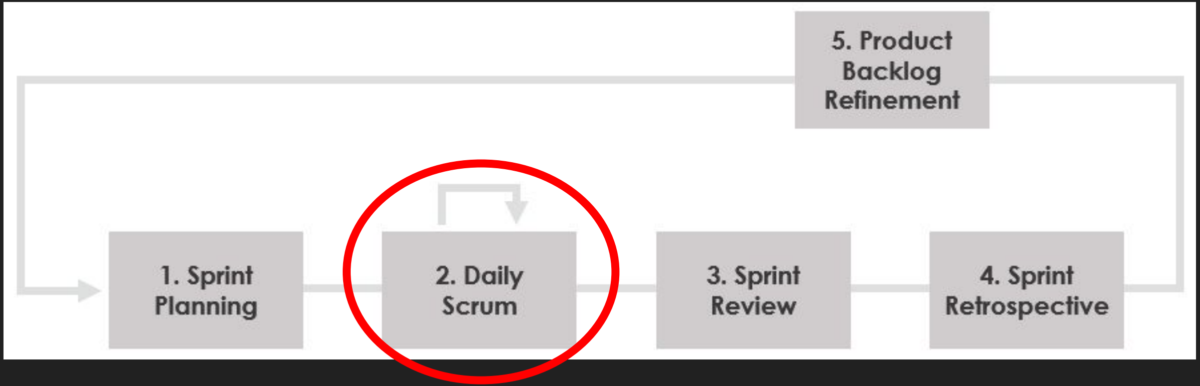


Events



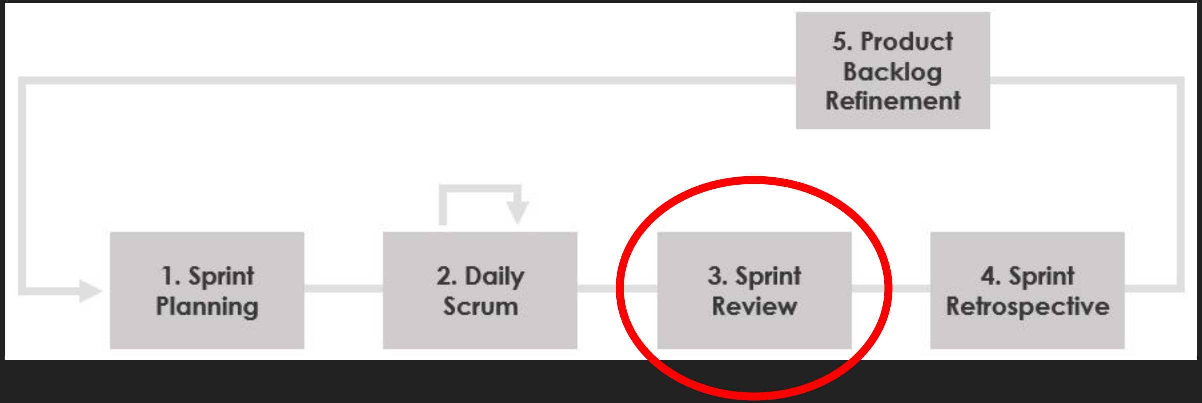
- All sprints begin with planning.
- The team needs to identify and commit to which items are going to be delivered as part of the sprint.
- Here the Scrum master has a main role

Events



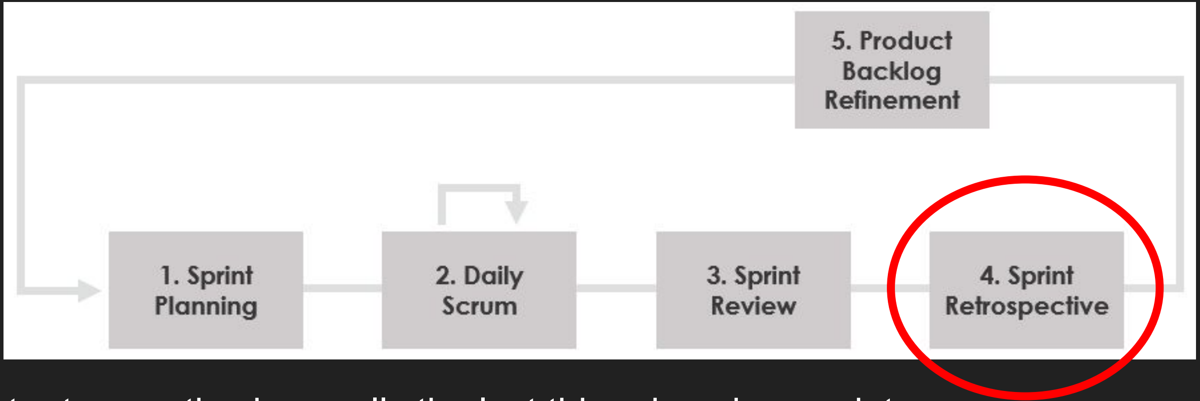
- The aim of this meeting is to ensure everyone within the team knows the status of the tasks accomplished (done) and of those in progress.
- The team has to answer the following questions:
  - What have we done until now?
  - What are we going to do today?
  - What are the impediments?
- No longer than 3 minutes per person
- The SCRUM master must where possible mitigate outside interruptions and distractions to the team

Events



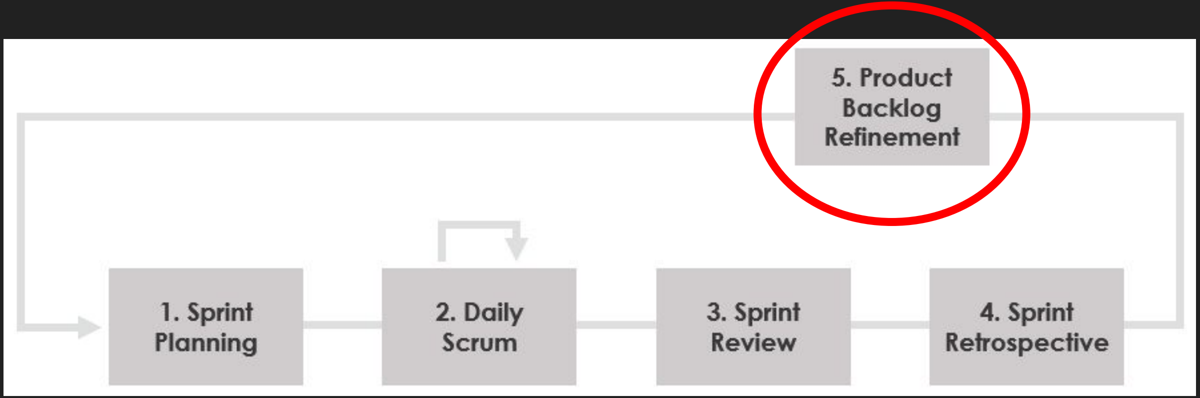
- A Sprint Review/Demo meeting is held at the end of the Sprint to inspect the Increment.
- The Team demonstrates the Increment with focus on the Sprint Goal according to the Definition of Done.
- The Product Owner reviews and accepts the delivered Increment.

Events



- The sprint retrospective is usually the last thing done in a sprint.
- You can schedule a scrum retrospective for up to an hour, which is usually quite sufficient.
- The retrospective gives the team the opportunity to identify 3 key aspects:
  - What should starting doing?
  - What did not go well (and stop doing again)?
  - What went well (and should keep doing)?
- Continually improve the team efficiency.

Events



- Think of the backlog as the roadmap of the project.
- As the team collaborates to create a list of everything that needs to be built or done for project completion, this list can be modified and added to throughout the project to ensure that all of the necessary needs of the project are met.
- It can be done anytime along the sprint period

Sprint

In the Scrum Framework all activities needed for the implementation of entries from the Scrum Product Backlog are performed within Sprints (also called ‘Iterations’). Sprints are always short: normally about 2-4 weeks.

Milestones > Epics > User Stories > Tasks

Groups of issues that Correspond to a project, feature, or time period

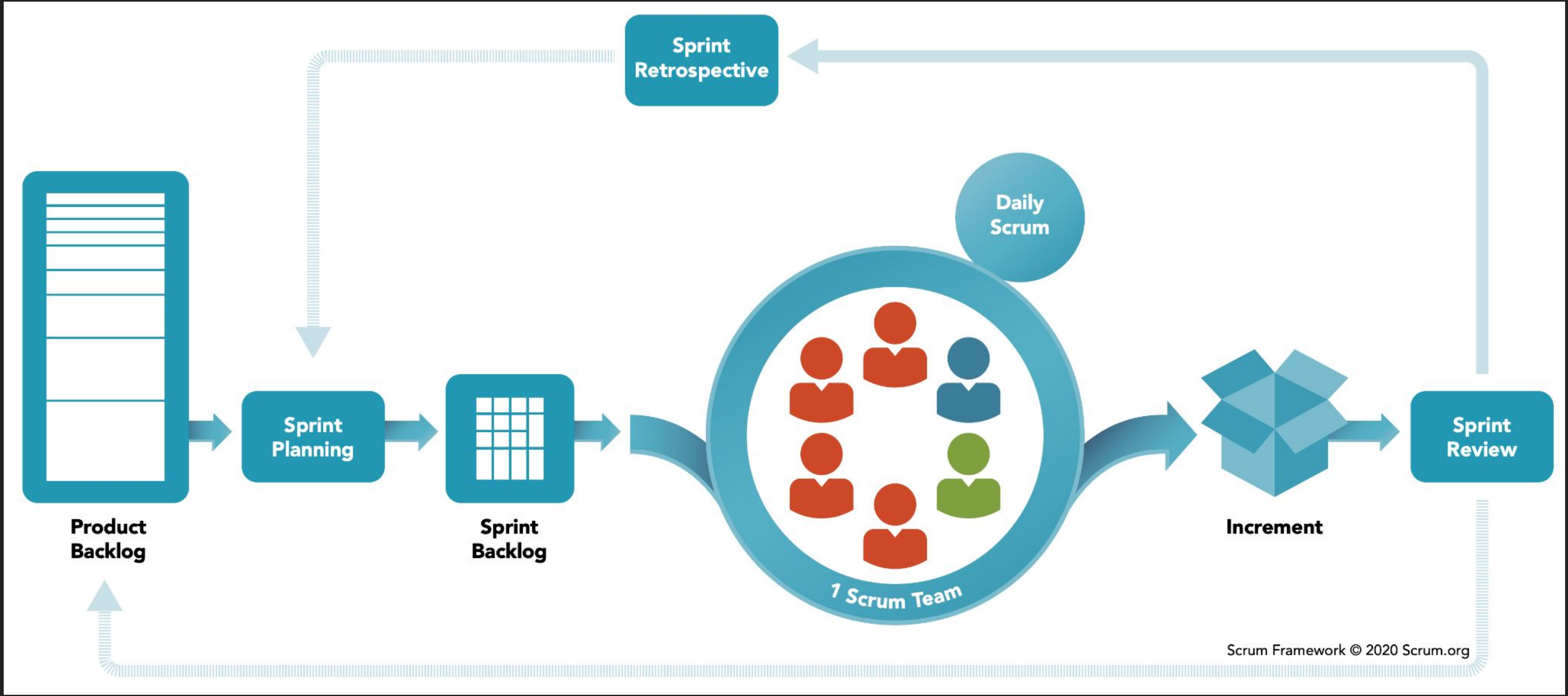
An epic captures a large body of work. It is essentially a “large user story” that can be broken down into a number of smaller stories.

A story is a brief statement of a product requirement or a business case.

A task is typically something like “code this”, “design that”, “create test data for such-and-such”, and so on. Tend to be things done by one person. A task is not written in the user story format. A task has more a technical nature.

Experienced Scrum Teams spend time and effort to break down complex and larger items (i.e user features or epics) into smaller user stories (or subsequently breaking down into tasks, or subtasks).

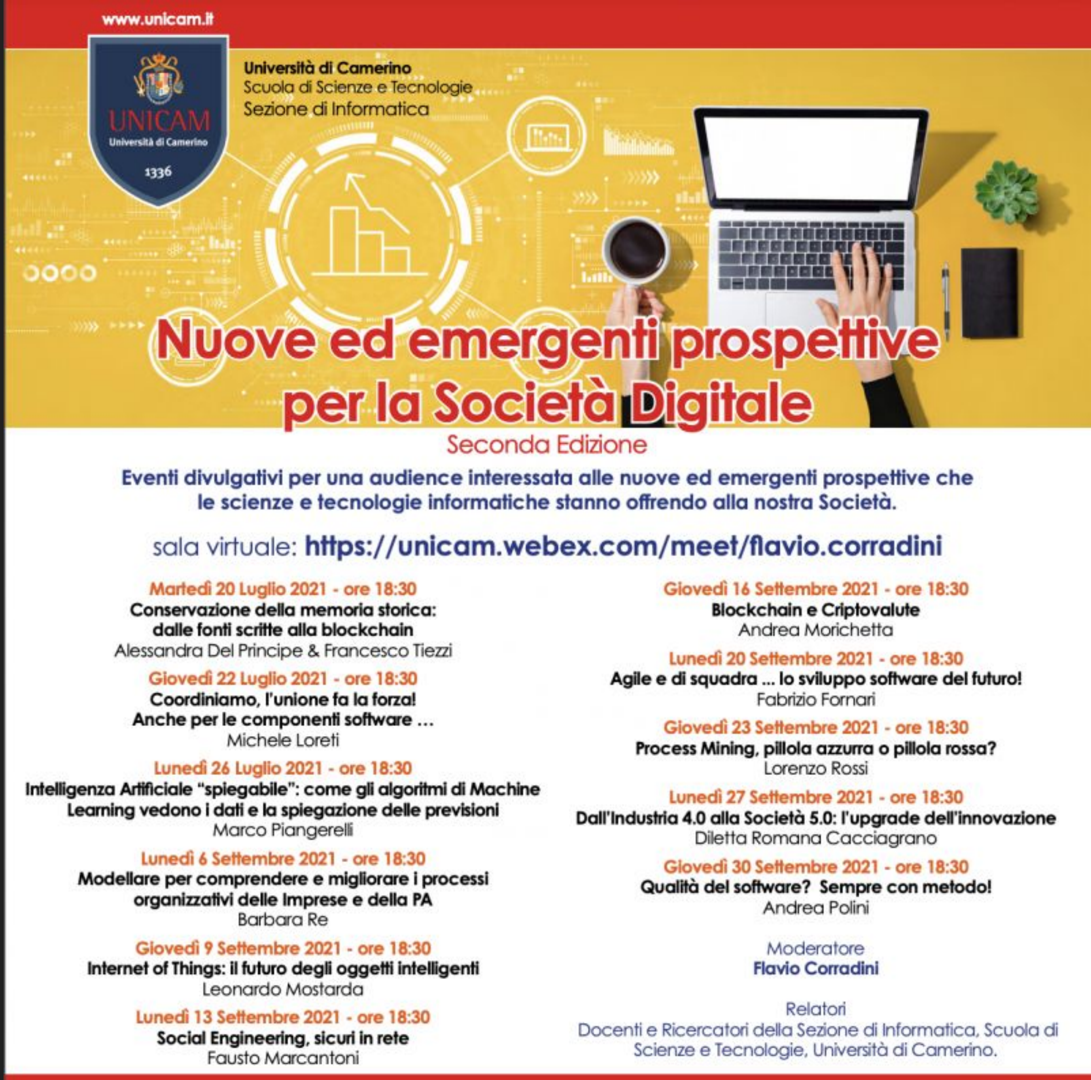
SCRUM - Framework



Italian Seminars

https://computerscience.unicam.it/nuove-ed-emergenti-prospettive-la-societa-digitale

My <u>edited</u> seminar on Agile and Scrum: https://youtu.be/G1IOB8CQll8 (Italian)



www.unicam.it

UNICAM Università di Camerino 1336

Università di Camerino
Scuola di Scienze e Tecnologie
Sezione di Informatica

Nuove ed emergenti prospettive per la Società Digitale

Seconda Edizione

Eventi divulgativi per una audience interessata alle nuove ed emergenti prospettive che le scienze e tecnologie informatiche stanno offrendo alla nostra Società.

sala virtuale: **https://unicam.webex.com/meet/flavio.corradini**

Martedì 20 Luglio 2021 - ore 18:30
Conservazione della memoria storica: dalle fonti scritte alla blockchain
Alessandra Del Principe & Francesco Tiezzi

Giovedì 22 Luglio 2021 - ore 18:30
Coordiniamo, l'unione fa la forza! Anche per le componenti software …
Michele Loreti

Lunedì 26 Luglio 2021 - ore 18:30
Intelligenza Artificiale "spiegabile": come gli algoritmi di Machine Learning vedono i dati e la spiegazione delle previsioni
Marco Piangerelli

Lunedì 6 Settembre 2021 - ore 18:30
Modellare per comprendere e migliorare i processi organizzativi delle Imprese e della PA
Barbara Re

Giovedì 9 Settembre 2021 - ore 18:30
Internet of Things: il futuro degli oggetti intelligenti
Leonardo Mostarda

Lunedì 13 Settembre 2021 - ore 18:30
Social Engineering, sicuri in rete
Fausto Marcantoni

Giovedì 16 Settembre 2021 - ore 18:30
Blockchain e Criptovalute
Andrea Morichetta

Lunedì 20 Settembre 2021 - ore 18:30
Agile e di squadra … lo sviluppo software del futuro!
Fabrizio Fornari

Giovedì 23 Settembre 2021 - ore 18:30
Process Mining, pillola azzurra o pillola rossa?
Lorenzo Rossi

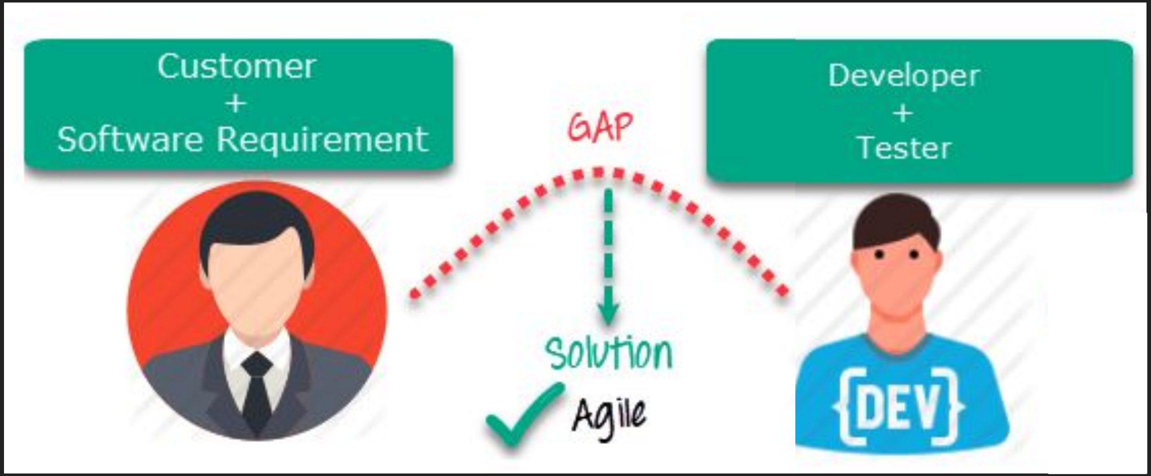
Lunedì 27 Settembre 2021 - ore 18:30
Dall'Industria 4.0 alla Società 5.0: l'upgrade dell'innovazione
Diletta Romana Cacciagrano

Giovedì 30 Settembre 2021 - ore 18:30
Qualità del software? Sempre con metodo!
Andrea Polini

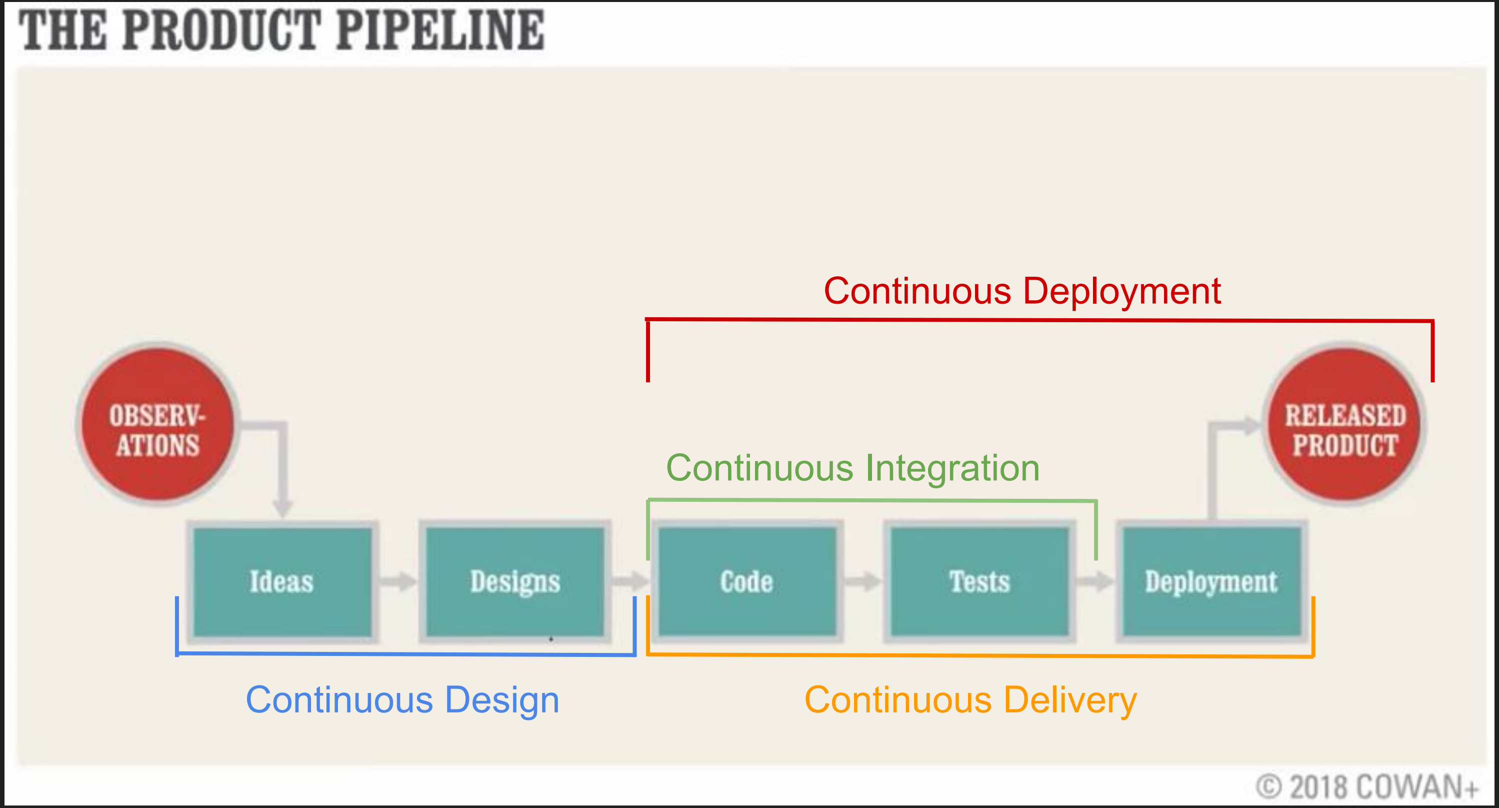
Moderatore
Flavio Corradini

Relatori
Docenti e Ricercatori della Sezione di Informatica, Scuola di Scienze e Tecnologie, Università di Camerino.

Focus of Agile paradigm



The Product Pipeline



Roles & Interfaces

DEVELOPER

Inputs: User stories

Outputs: Software design & Implementation

TESTER

Inputs: Working software, notes on target behavior

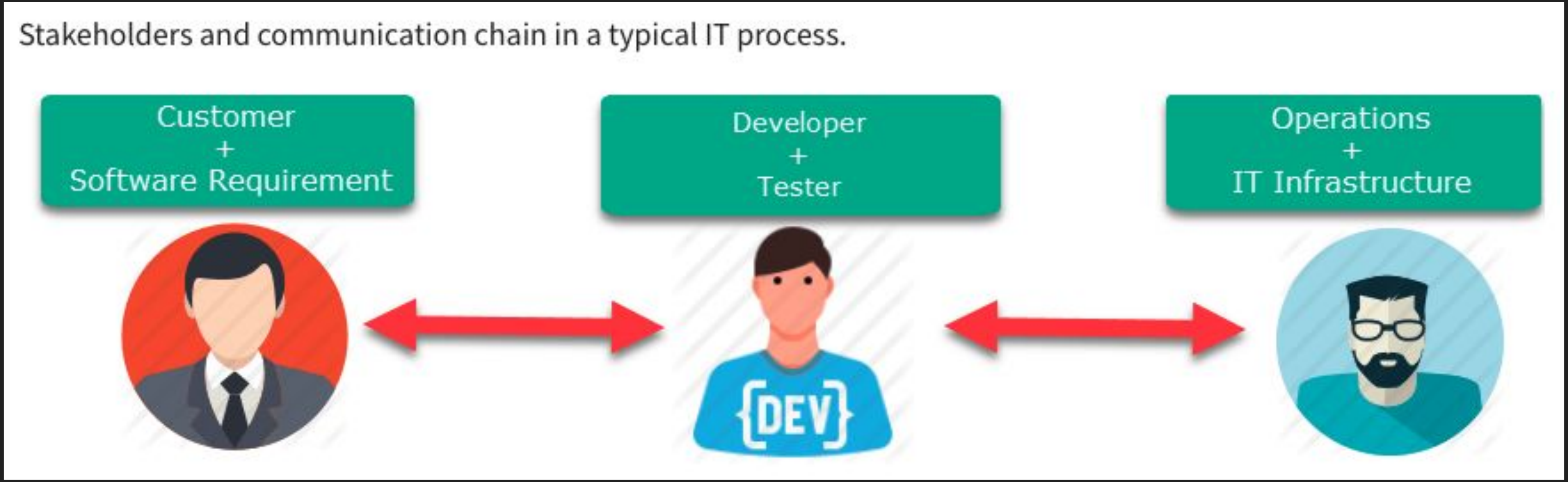
Outputs: Validated software

OPERATORS

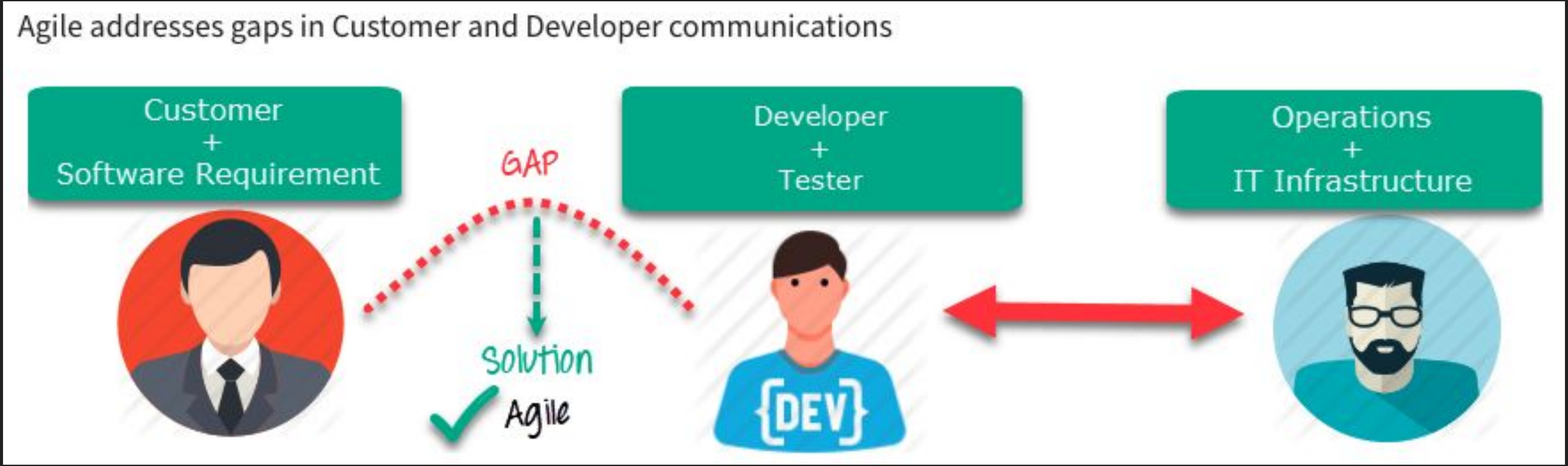
Inputs: Validated software deployment notes

Outputs: Working systems, monitoring & analytics thereof   evaluates what the software is doing and if it behaves as expected

Classic (and Old) Process


Stakeholders and communication chain in a typical IT process.
Customer + Software Requirement
Developer + Tester
Operations + IT Infrastructure
{DEV}

Agile


Agile addresses gaps in Customer and Developer communications
Customer + Software Requirement
GAP
Solution
Agile
Developer + Tester
{DEV}
Operations + IT Infrastructure

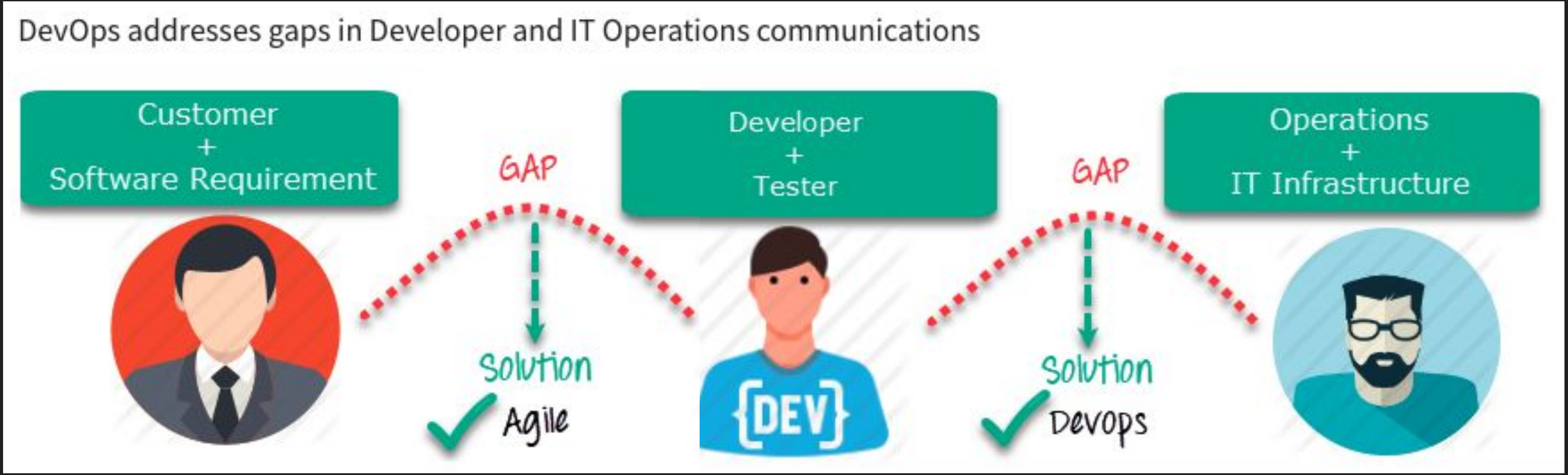
DevOps

“A compound of development (Dev) and operations (Ops), DevOps is the union of people, process, and technology to continually provide value to customers.

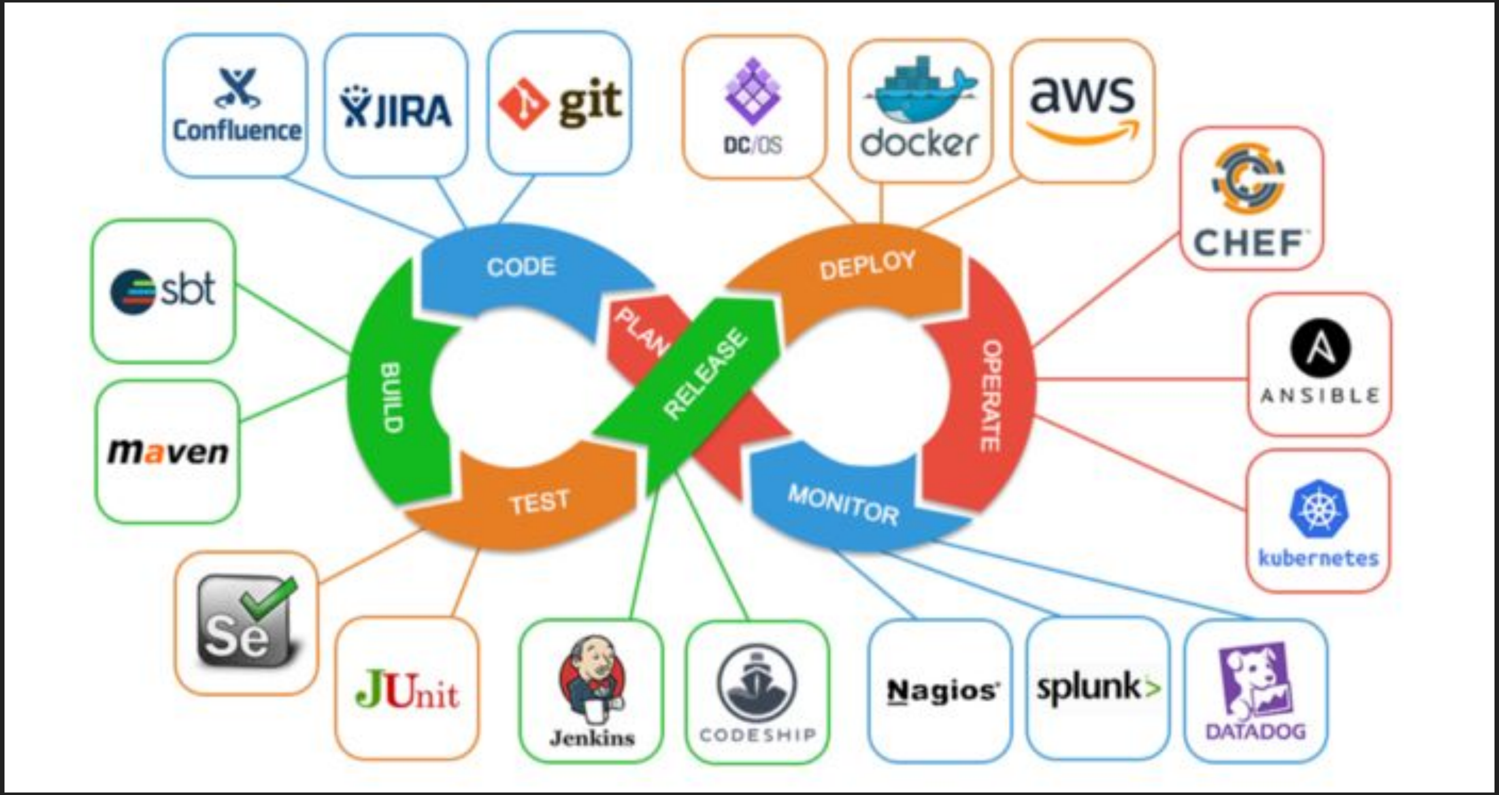
What does DevOps mean for teams? DevOps enables formerly siloed roles — development, IT operations, quality engineering, and security — to coordinate and collaborate to produce better, more reliable products. By adopting a DevOps culture along with DevOps practices and tools, teams gain the ability to better respond to customer needs, increase confidence in the applications they build, and achieve business goals faster.”

— Azure.microsoft.com

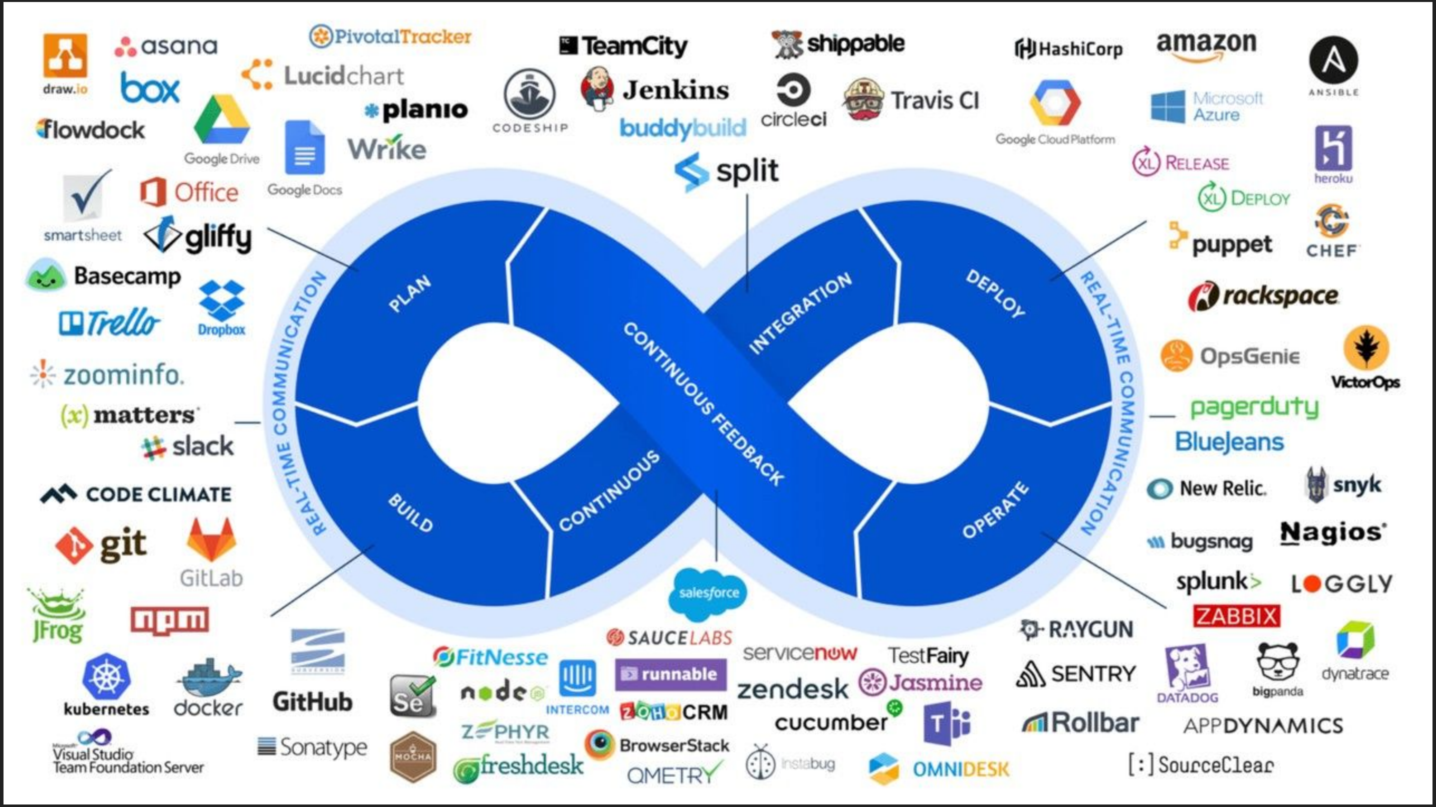
DevOps


DevOps addresses gaps in Developer and IT Operations communications
Customer + Software Requirement
GAP
Solution
Agile
Developer + Tester
{DEV}
GAP
Solution
Devops
Operations + IT Infrastructure

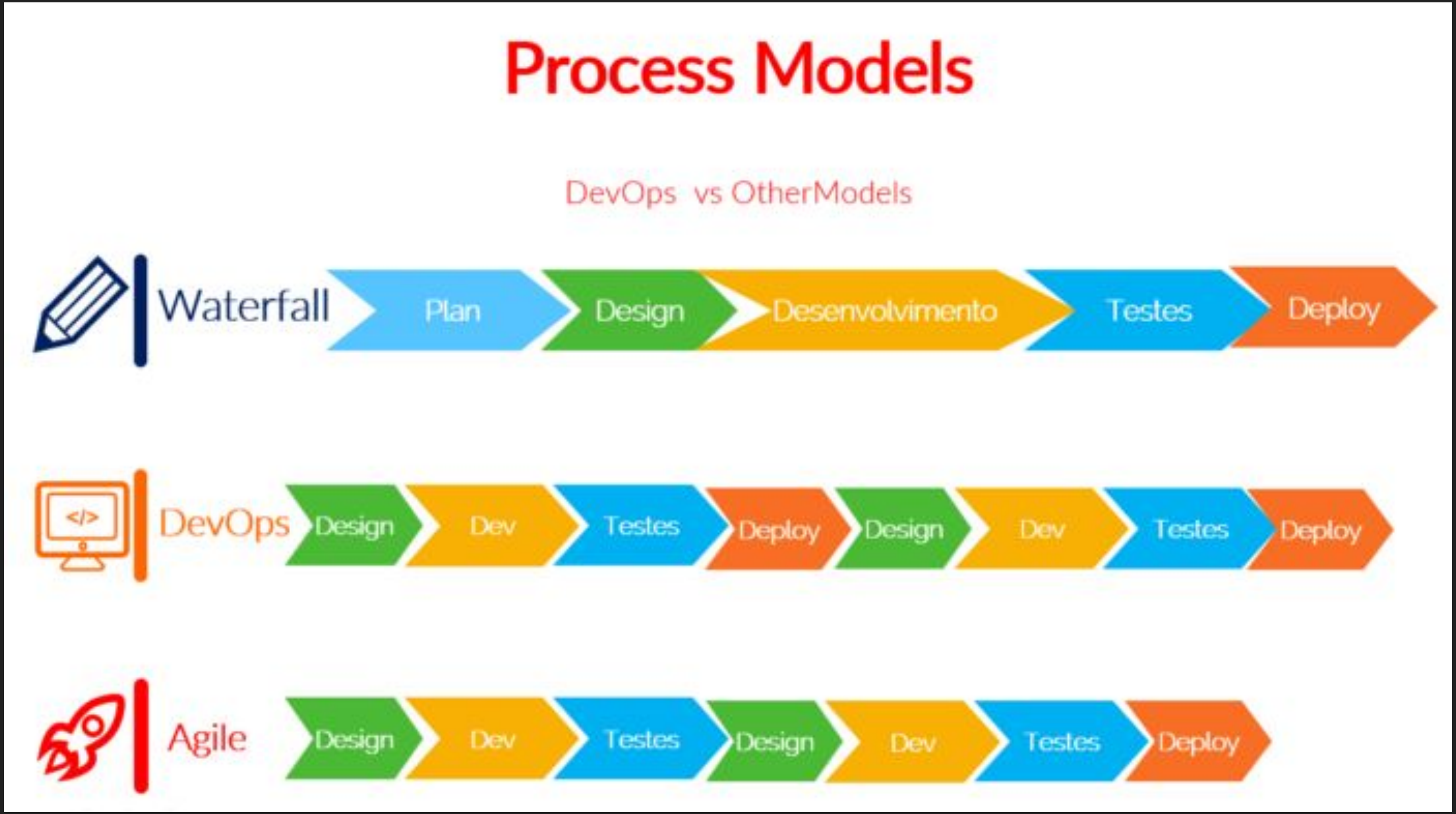
DevOps Technologies



DevOps Technologies

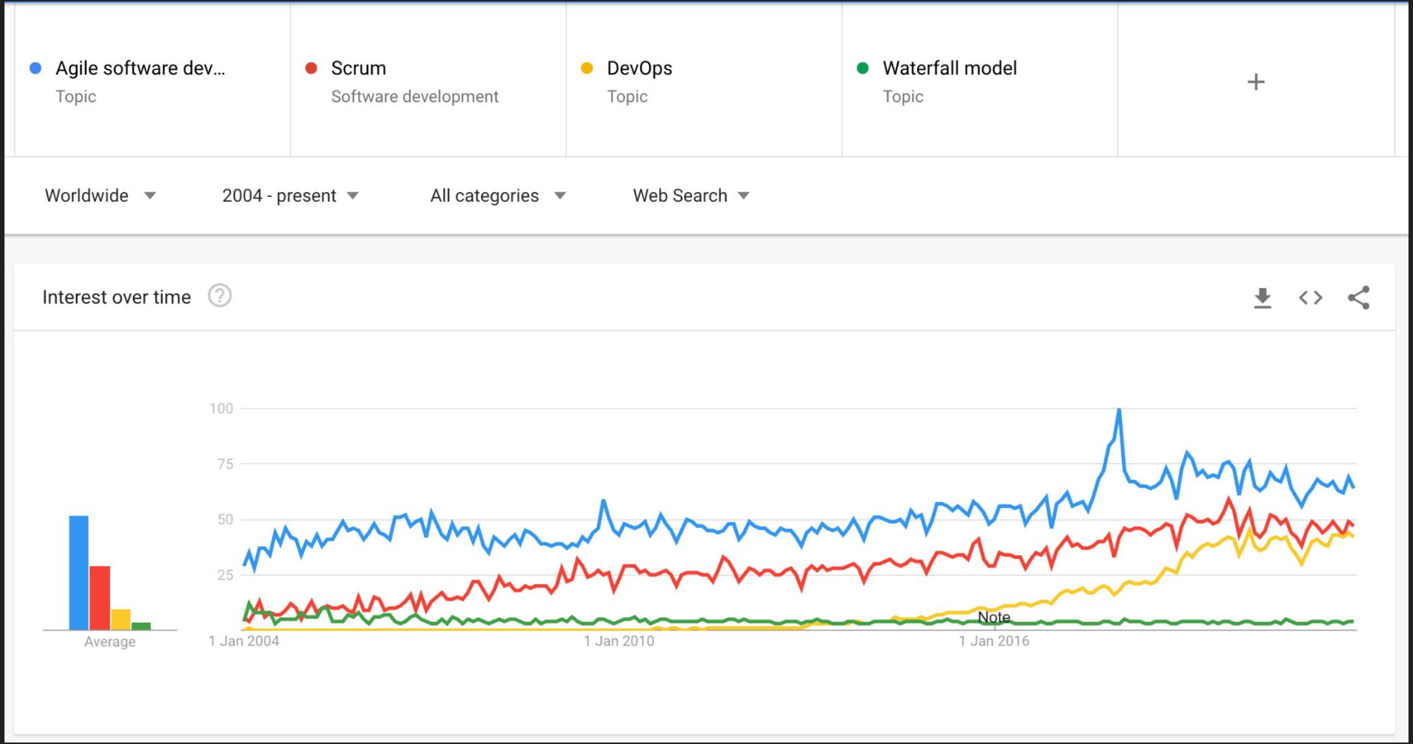


Recap


Process Models
DevOps vs OtherModels
Waterfall
Plan
Design
Desenvolvimento
Testes
Deploy
DevOps
Design
Dev
Testes
Deploy
Design
Dev
Testes
Deploy
Agile
Design
Dev
Testes
Design
Dev
Testes
Deploy

Waterfall vs Agile vs Scrum vs DevOps

Google Trends indication



● Agile software dev... Topic

● Scrum Software development

● DevOps Topic

● Waterfall model Topic

Worldwide ▾ 2004 - present ▾ All categories ▾ Web Search ▾

Interest over time

https://trends.google.com/trends/explore?date=all&q=%2Fm%2F02zhbn,%2Fm%2F0ck_p8,%2Fm%2F0c3tq11,%2Fm%2F0867l

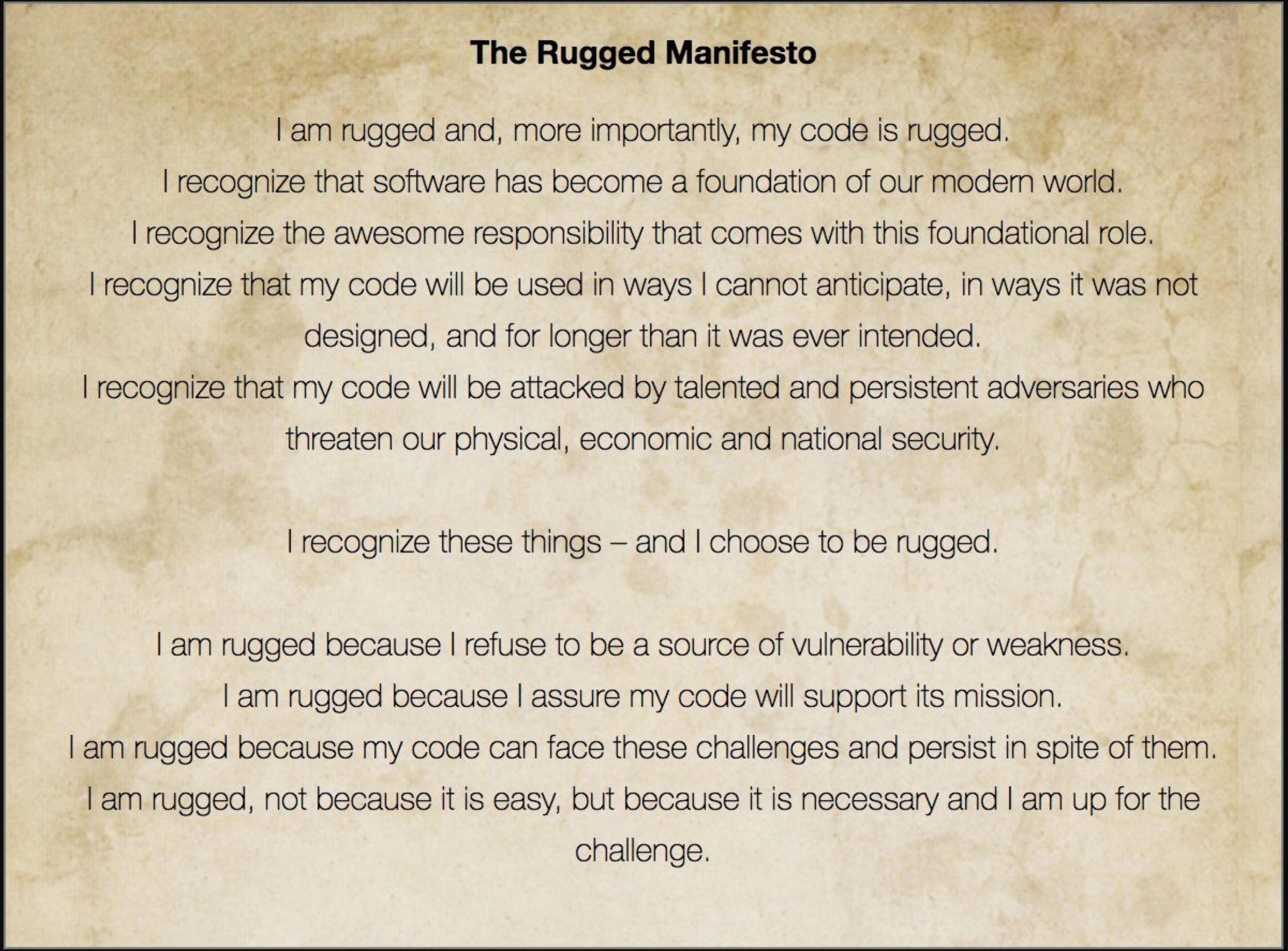
A Curiosity

DevOpsSec

Rugged DevOps

Incorporate security

If you describe someone's character as rugged, you mean that **they are strong and determined, and have the ability to cope with difficult situations**.



The Rugged Manifesto

I am rugged and, more importantly, my code is rugged.

I recognize that software has become a foundation of our modern world.

I recognize the awesome responsibility that comes with this foundational role.

I recognize that my code will be used in ways I cannot anticipate, in ways it was not designed, and for longer than it was ever intended.

I recognize that my code will be attacked by talented and persistent adversaries who threaten our physical, economic and national security.

I recognize these things – and I choose to be rugged.

I am rugged because I refuse to be a source of vulnerability or weakness.

I am rugged because I assure my code will support its mission.

I am rugged because my code can face these challenges and persist in spite of them.

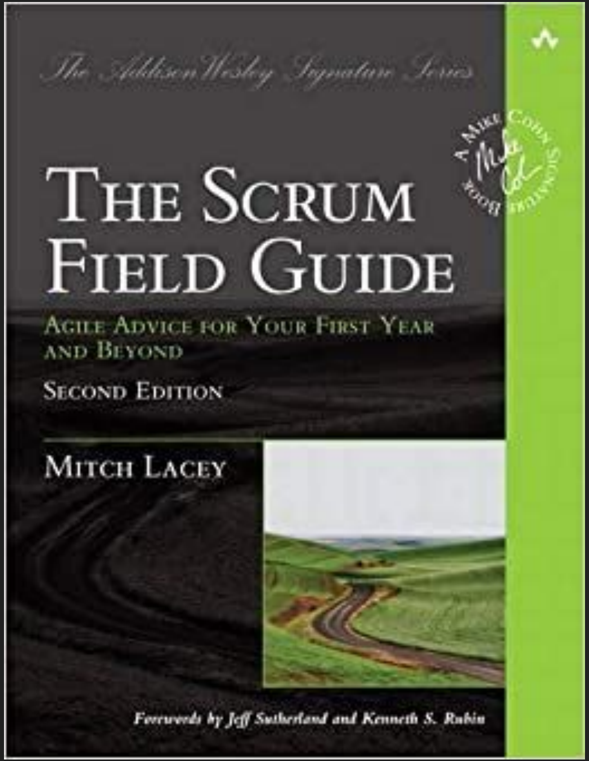
I am rugged, not because it is easy, but because it is necessary and I am up for the challenge.

Additional Materials

Scrum Field Guide,

The: Agile Advice for Your First Year and Beyond (Addison-Wesley Signature Series (Cohn)) 2nd Edition

By Mitch Lacey



The Addison Wesley Signature Series

A Mike Cohn Signature Book

THE SCRUM FIELD GUIDE

AGILE ADVICE FOR YOUR FIRST YEAR AND BEYOND

SECOND EDITION

MITCH LACEY

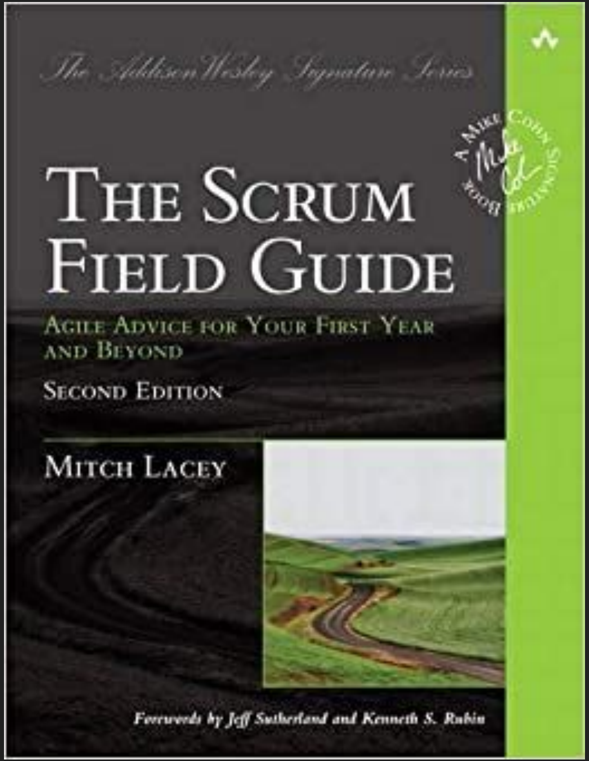
Forewords by Jeff Sutherland and Kenneth S. Rubin

Additional Materials

Scrum Field Guide,

The: Agile Advice for Your First Year and Beyond (Addison-Wesley Signature Series (Cohn)) 2nd Edition

By Mitch Lacey


The Addison Wesley Signature Series
A Mike Cohn Signature Book
The Scrum Field Guide
Agile Advice for Your First Year and Beyond
Second Edition
Mitch Lacey
Forewords by Jeff Sutherland and Kenneth S. Rubin

https://www.mountaingoatsoftware.com/

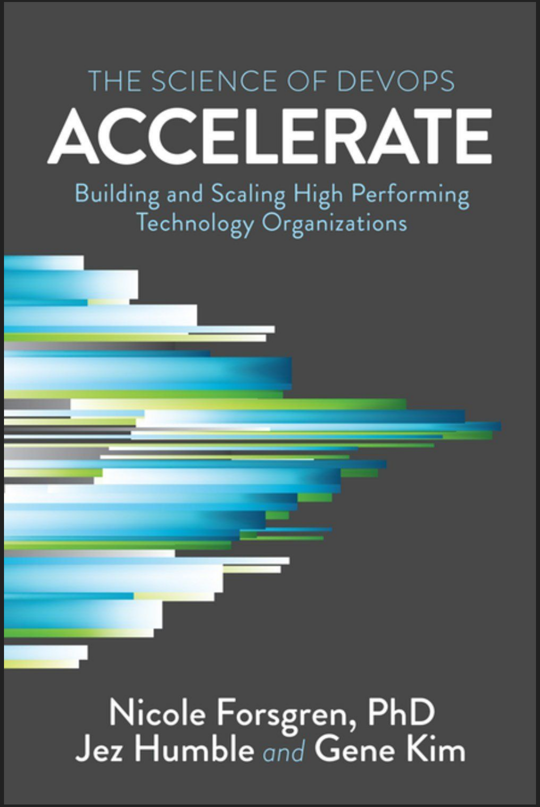
By Mike Cohn

Additional Materials

Accelerate

Building and Scaling High Performing Technology Organizations

By Nicole Forsgren, PhD Jez Humble and Gene Kim

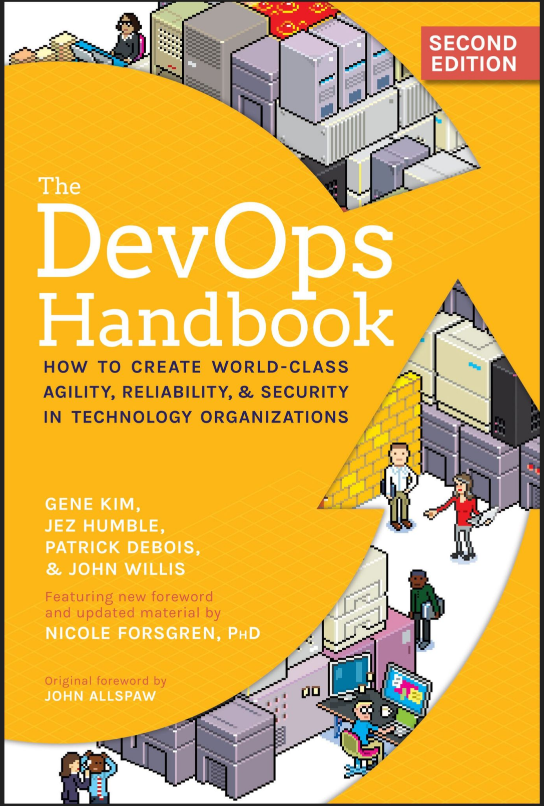


Additional Materials

The DevOps Handbook

How to create world-class agility, reliability, & security in technology organizations

By Gene Kim, Jez Humble, Patrick Debois, & John Wills


SECOND EDITION
The DevOps Handbook
HOW TO CREATE WORLD-CLASS AGILITY, RELIABILITY, & SECURITY IN TECHNOLOGY ORGANIZATIONS
GENE KIM, JEZ HUMBLE, PATRICK DEBOIS, & JOHN WILLIS
Featuring new foreword and updated material by NICOLE FORSGREN, PHD
Original foreword by JOHN ALLSPAW

Any Question?