



Agile Software Development



Dario Campagna

Manifesto for Agile Software Development

Utah, 2001. A group of 17 experts (with different backgrounds) met to discuss the growing field of what used to be called lightweight methods.

- Captures common ground about software development
- Sets out the values and principles of these lightweight methods
- Covers technical, organizational and people aspect of software development



We are uncovering better ways of developing software by doing it and helping others do it.





Agile, a bit of history

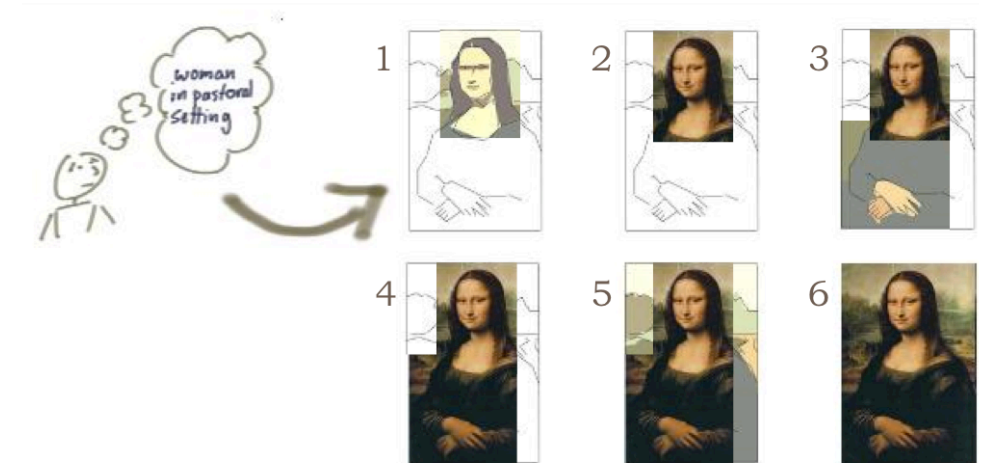
Iterative and Incremental Development



Iterate to evaluate and make changes to what you've already made

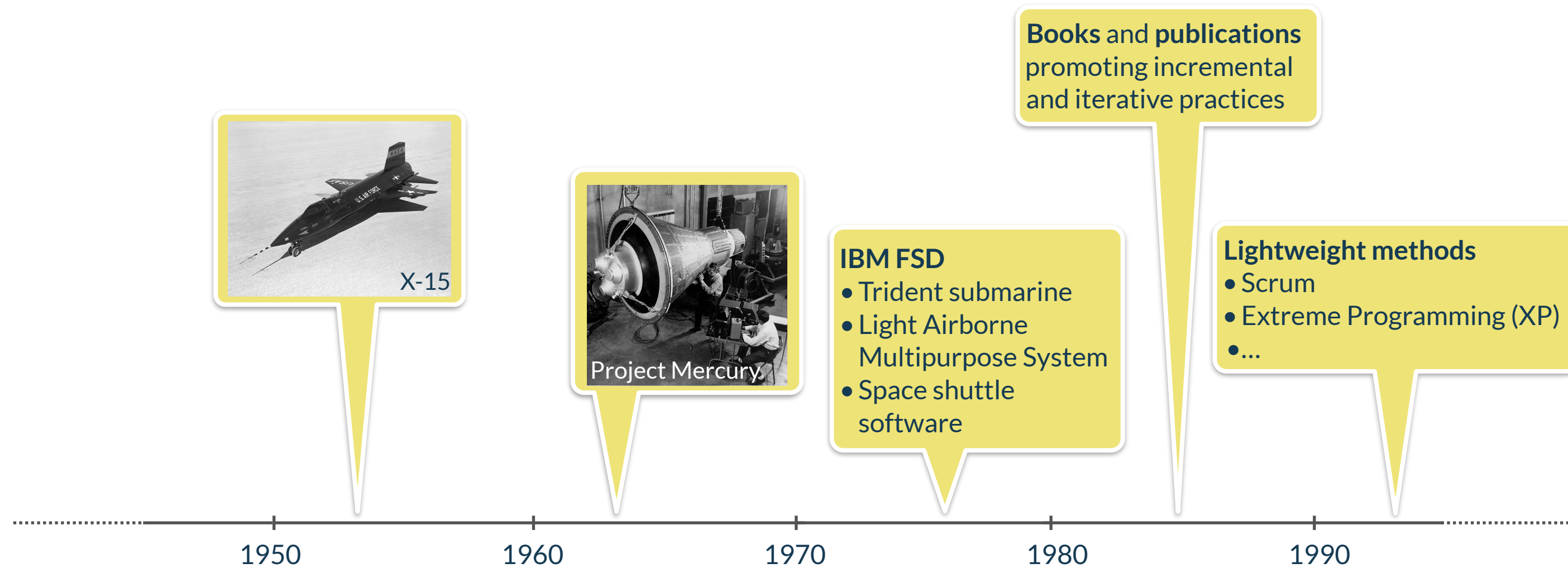


Increment to make additions

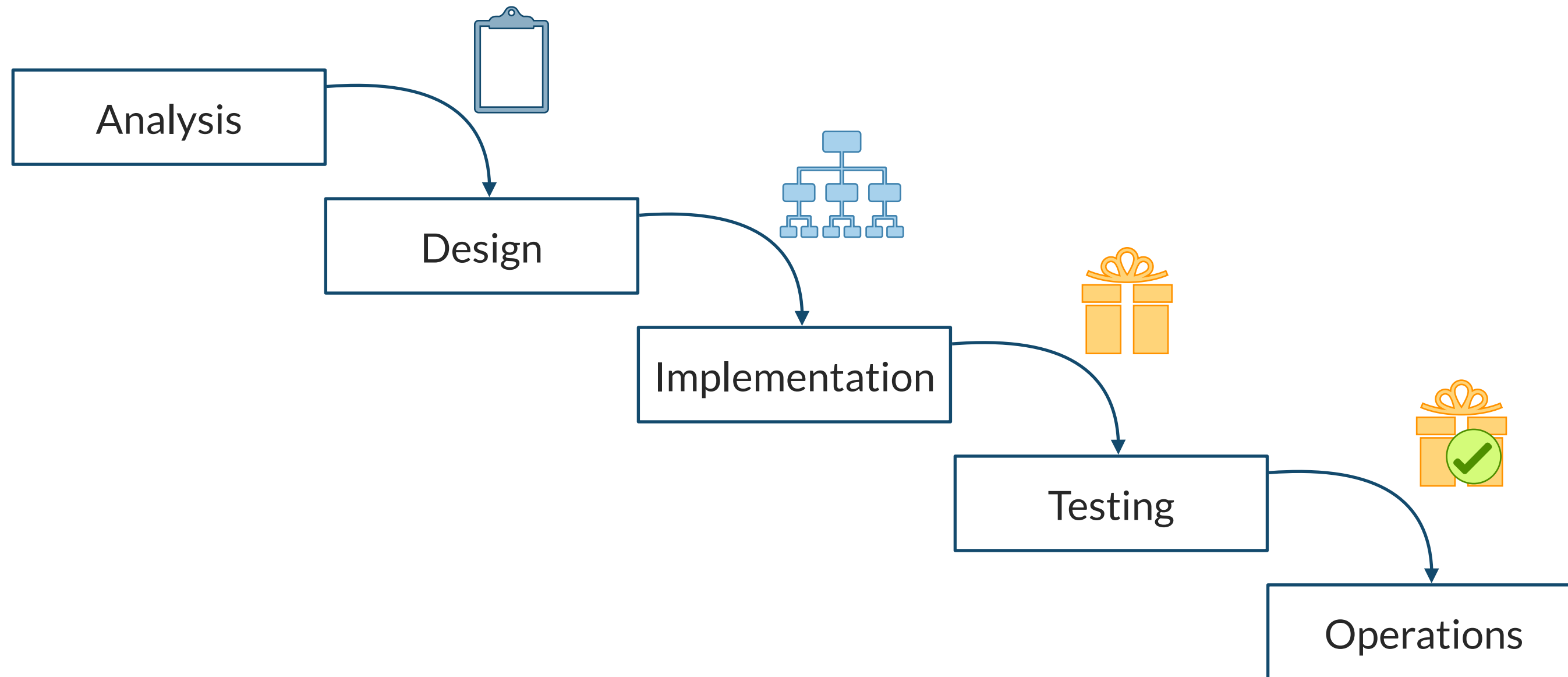


The two tactics can be **conjoined**

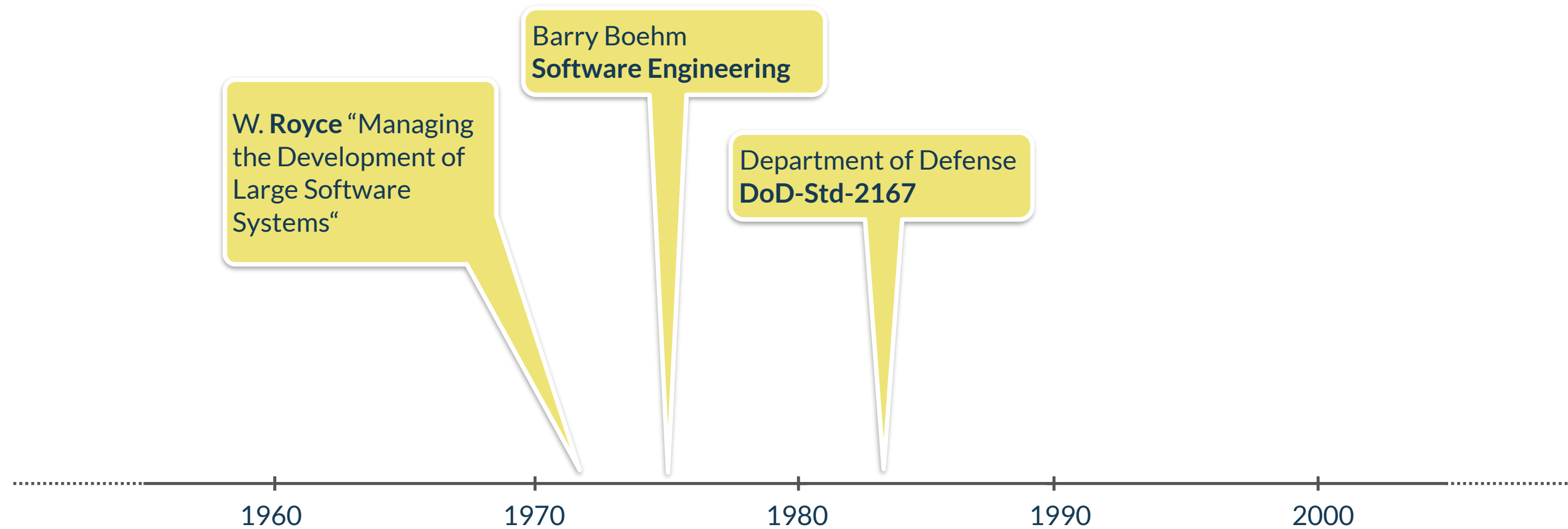
Iterative and Incremental Development



Specify than build approach (a.k.a, waterfall)



Specify than build approach (a.k.a, waterfall)



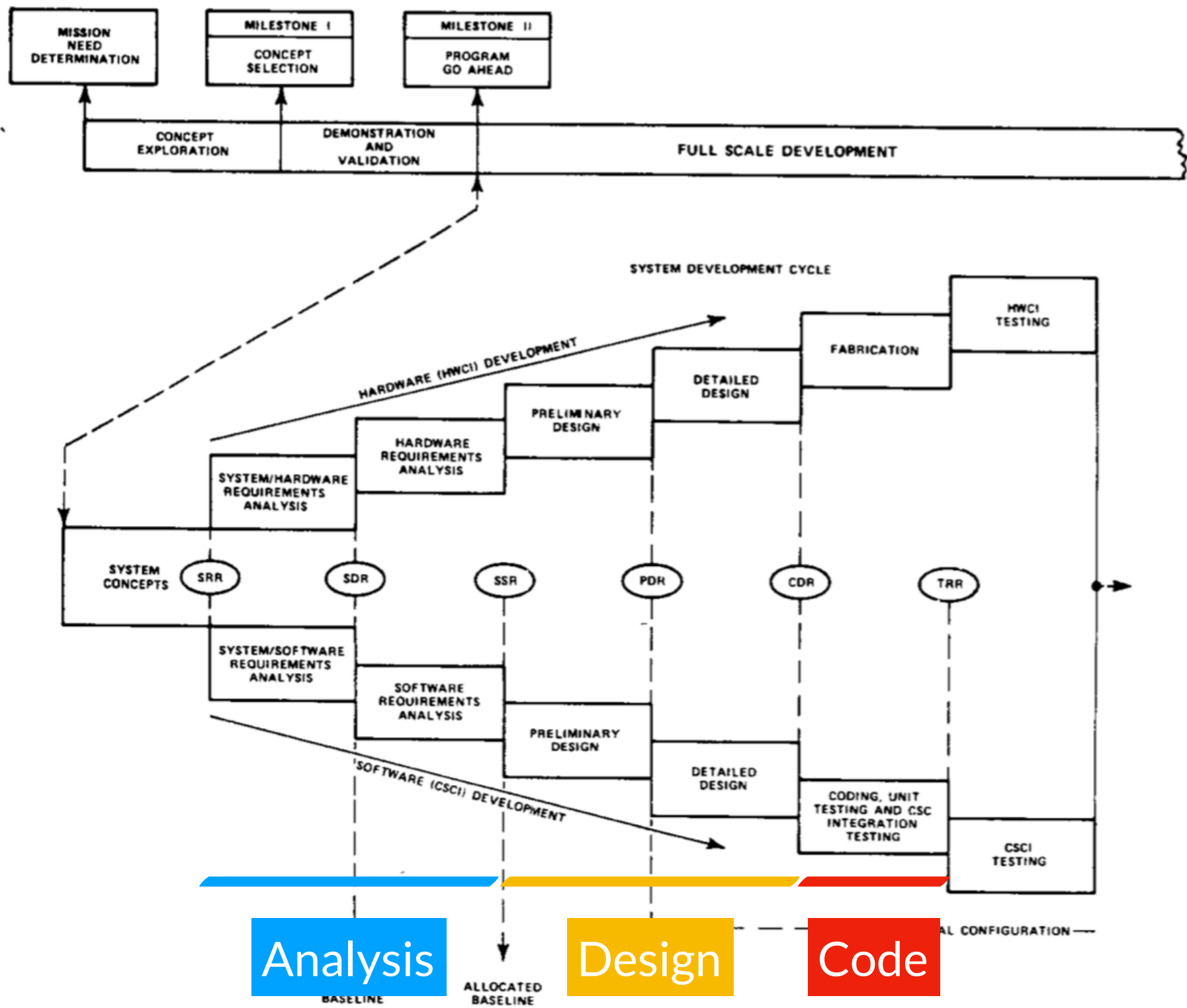


FIGURE 1. System development cycle within the system life cycle.

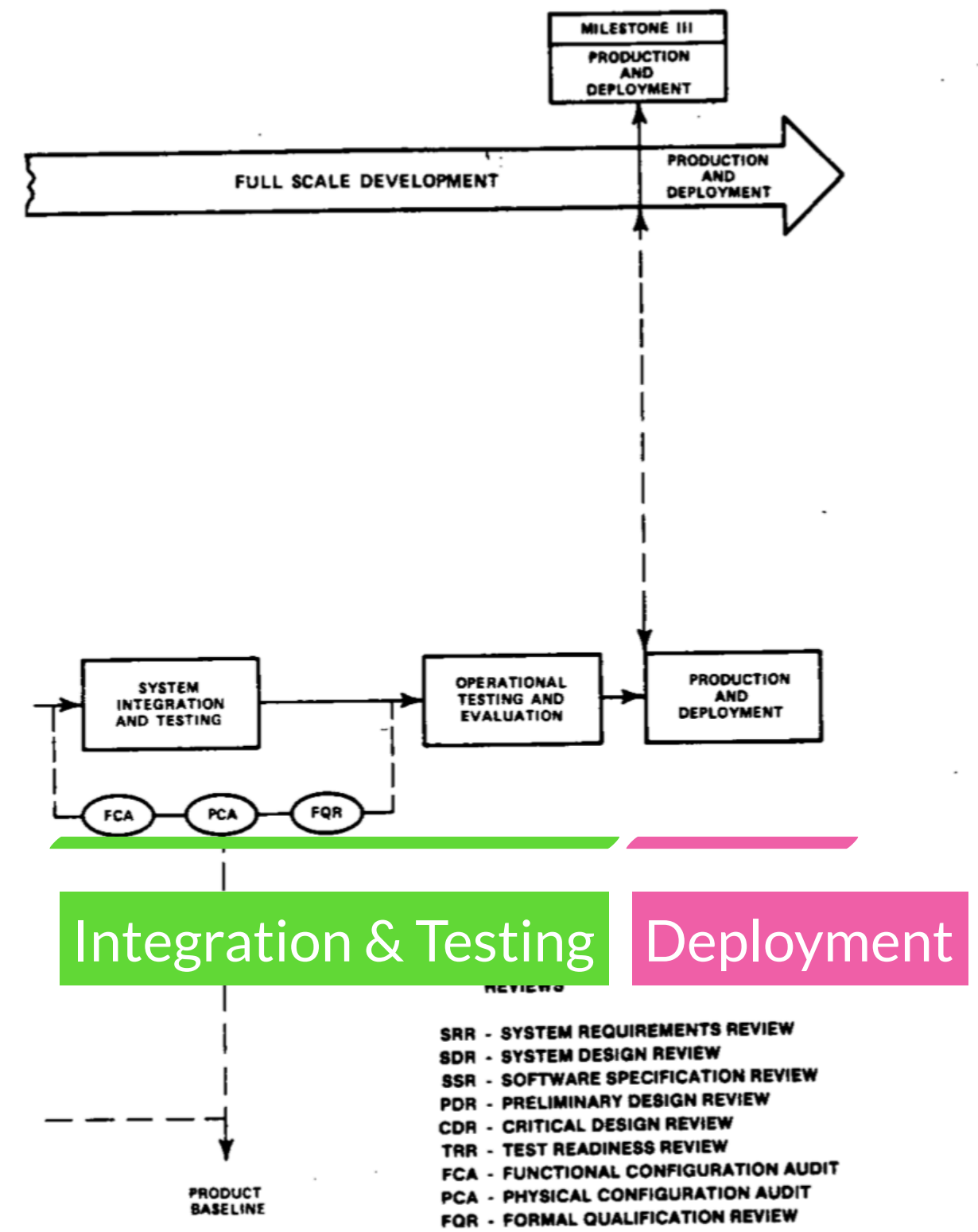
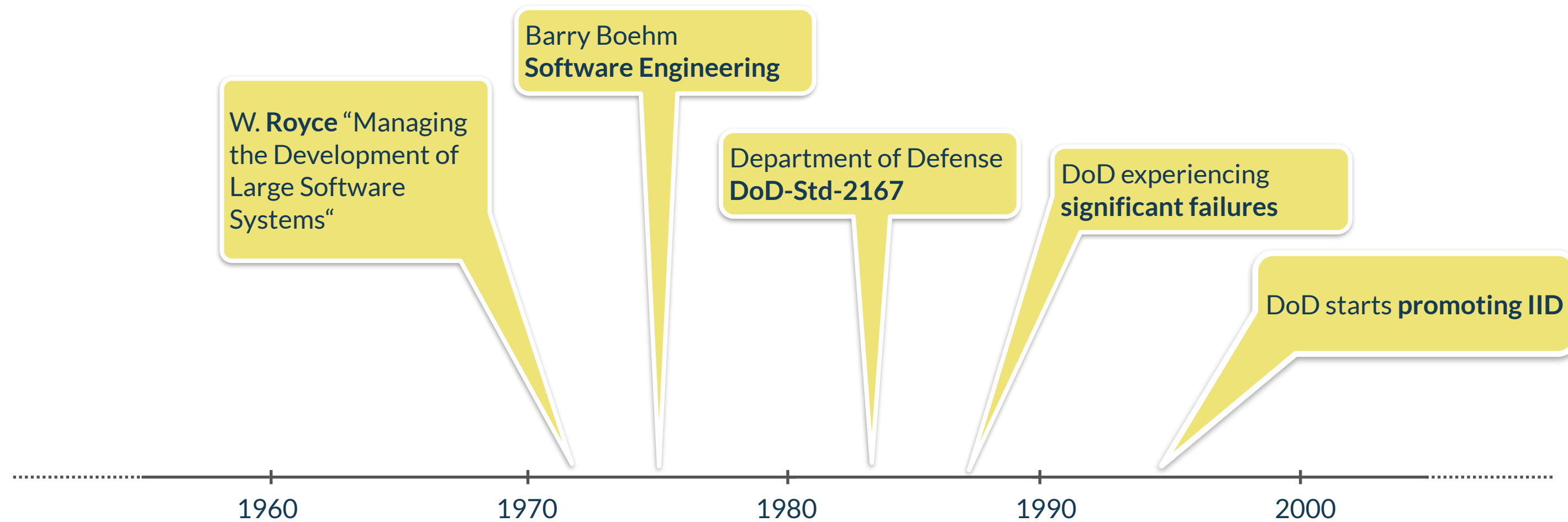


FIGURE 1. System development cycle within the system life cycle. (continued)



Specify than build approach (a.k.a, waterfall)



Why did projects fail?

1987 report of the Defense Science Board Task Force on Military Software.

Directive 5000.29 not only does not encourage this best modern practice, it essentially forbids it. We recommend that it be revised immediately to mandate and facilitate early prototyping before the baseline specifications are established (Rec. #23).

DoD-STD-2167 likewise needs a radical overhaul to reflect best modern practice. Draft DoD-STD-2167A is a step, but it does not go nearly far enough. As drafted, it continues to reinforce exactly the document-driven, specify-then-build approach that lies at the heart of so many DoD software problems.



Why did projects fail?

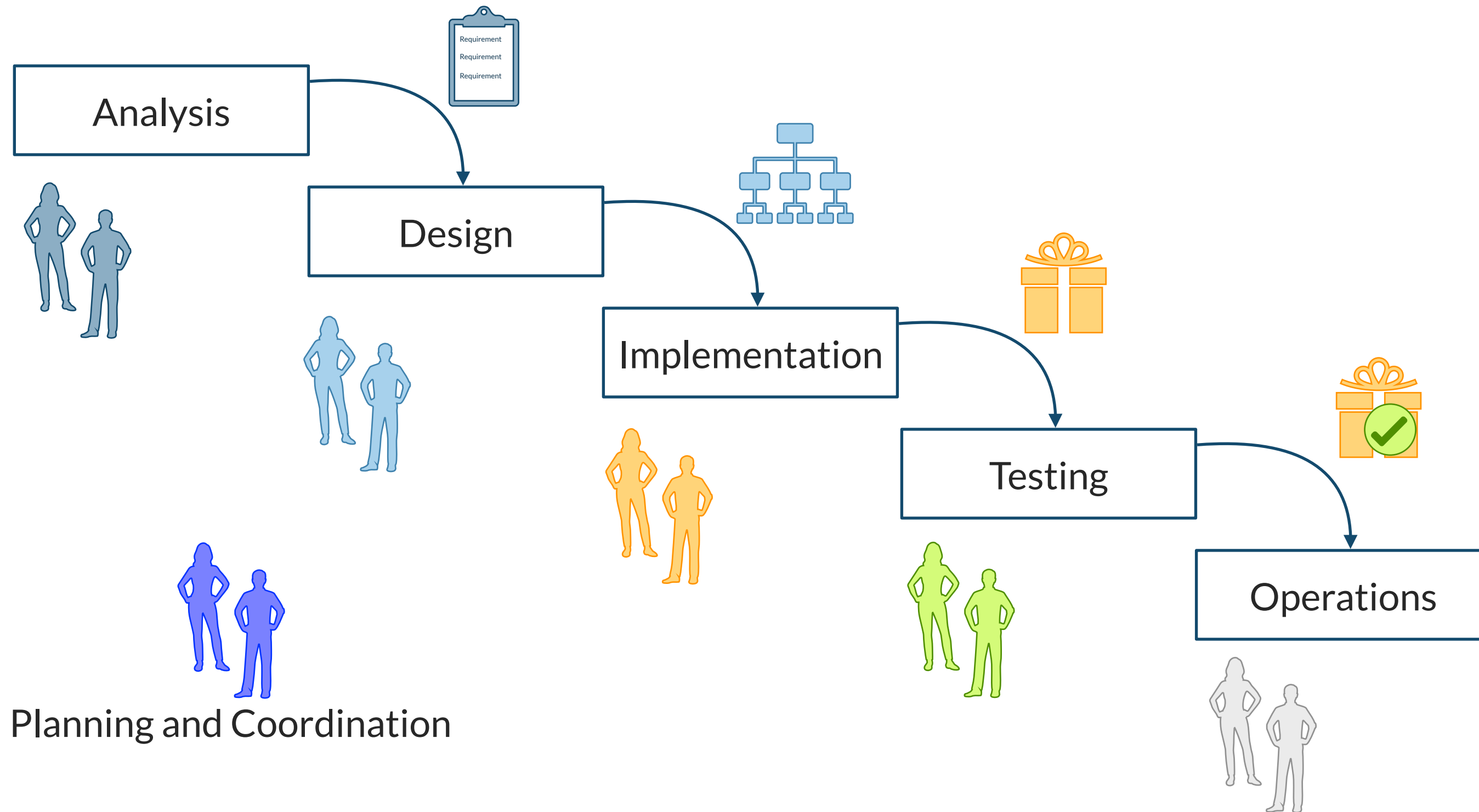
1994 report of the Defense Science Board Task Force on Acquiring Defense Software Commercially.

Principal Reasons DoD Software Programs Get Into Trouble

- **Poor Requirements Definition**
 - Lack of User Involvement in Development Process
 - Inability of Users to Foresee Benefits of Automation Without Incremental Capability
- **Inadequate Software Process Management and Control by Contractor**
- **Lack of Integrated Product Teams**
 - Failure to Establish "Team" With Vendors and Users
 - Little Participation of Functional Area Experts
- **Ineffective Subcontractor Management**
- **Lack of Consistent Attention to Software Process**
- **Too Little Attention to Software Architecture**
- **Poorly Defined and Inadequately Controlled Interfaces Between Computer Hardware, Communications and Software**
- **Assumption That Software Upgrades Can "Fix" Hardware Deficiencies (Without Assessment of Cost and Schedule Risks)**
- **Focus on Innovation Rather than Cost and Risk**
- **Limited or No Tailoring of Military Specifications Based on Continuing Cost-Benefit Evaluations**



Specify-then-build approach and Organization



Management Theories

- Best practices should be pushed throughout the organization
- Planning and improvement work separated from normal work

Scientific Management

Frederick Taylor

Responsibilities of managers

- Planning
- Organizing
- Coordinating
- Commanding
- Controlling

General and Industrial Management

Henri Fayol



Charlie and Jane

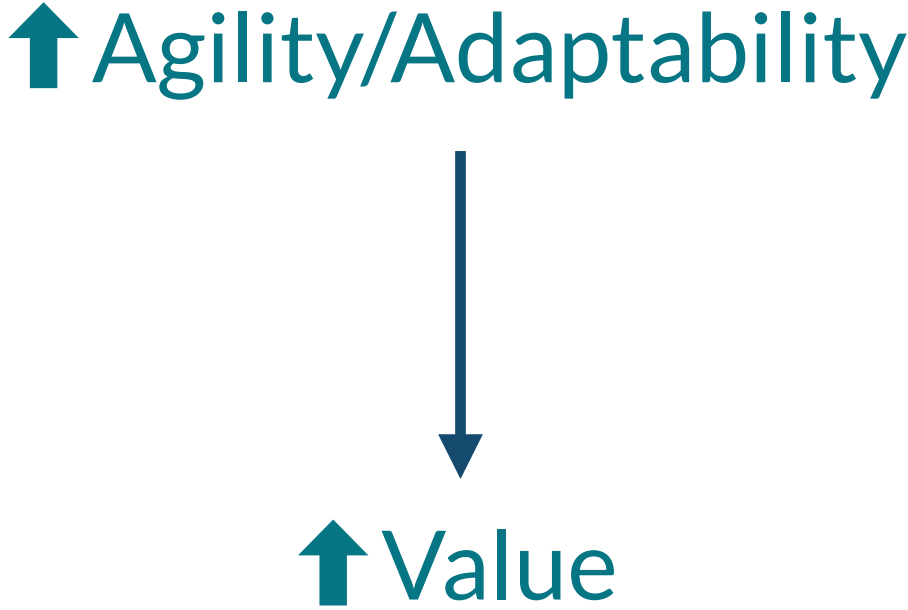




Characteristics of Agile



Adaptability



Agility ≠ Fast

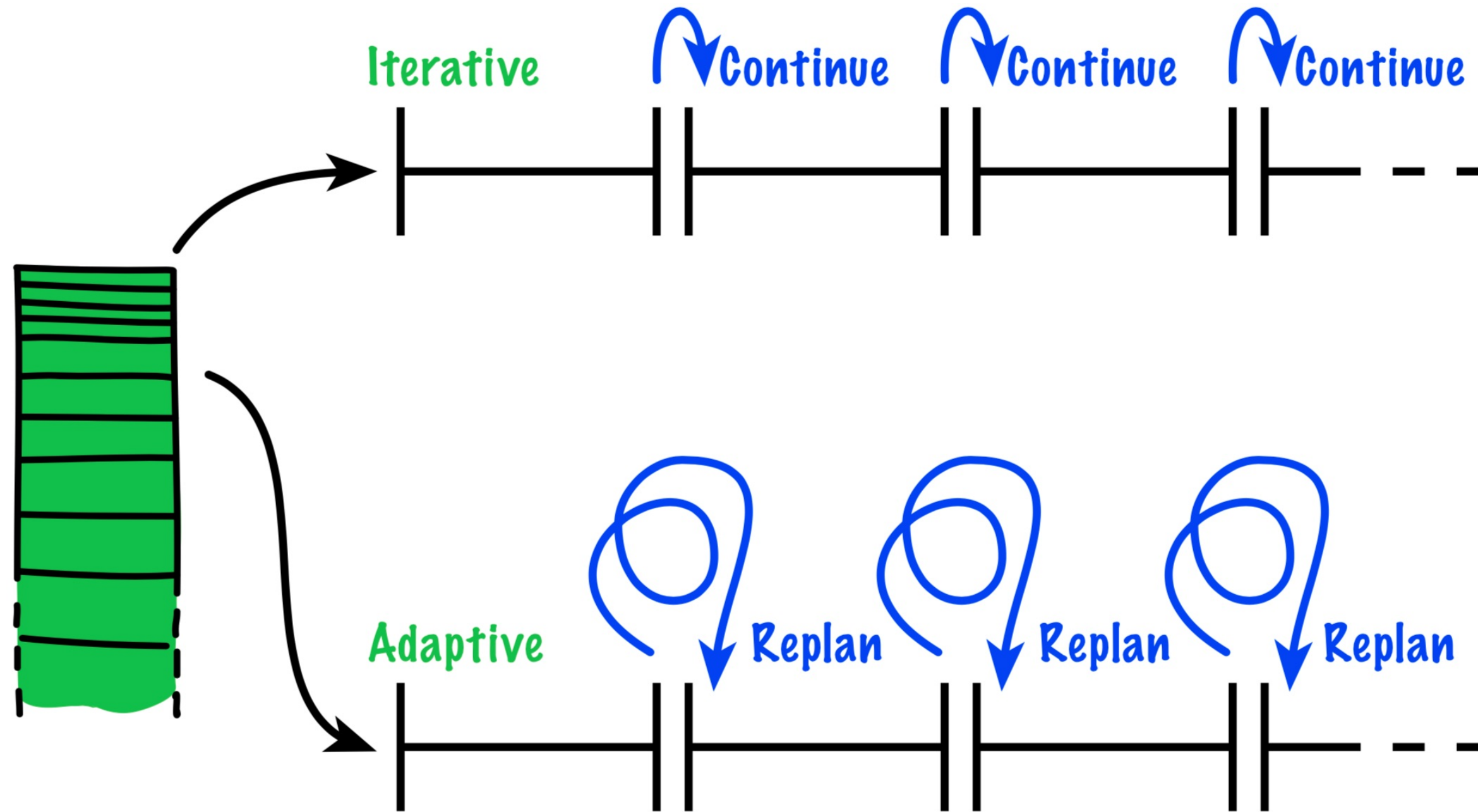
Agility ≠ Cheap

Adaptability as a driver for value.

Fast and cheap are not goals for Agile.

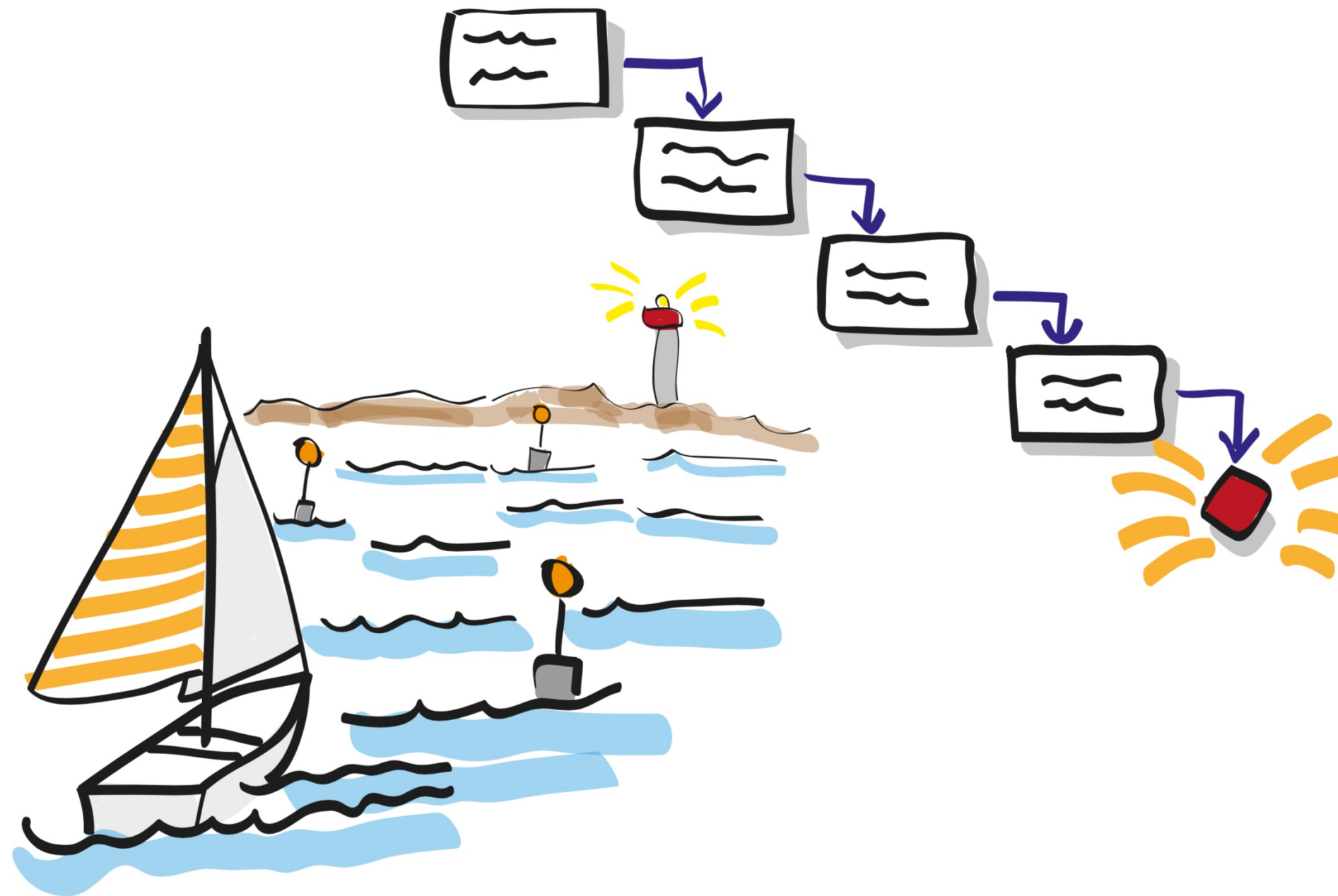


Adaptive Vs Iterative



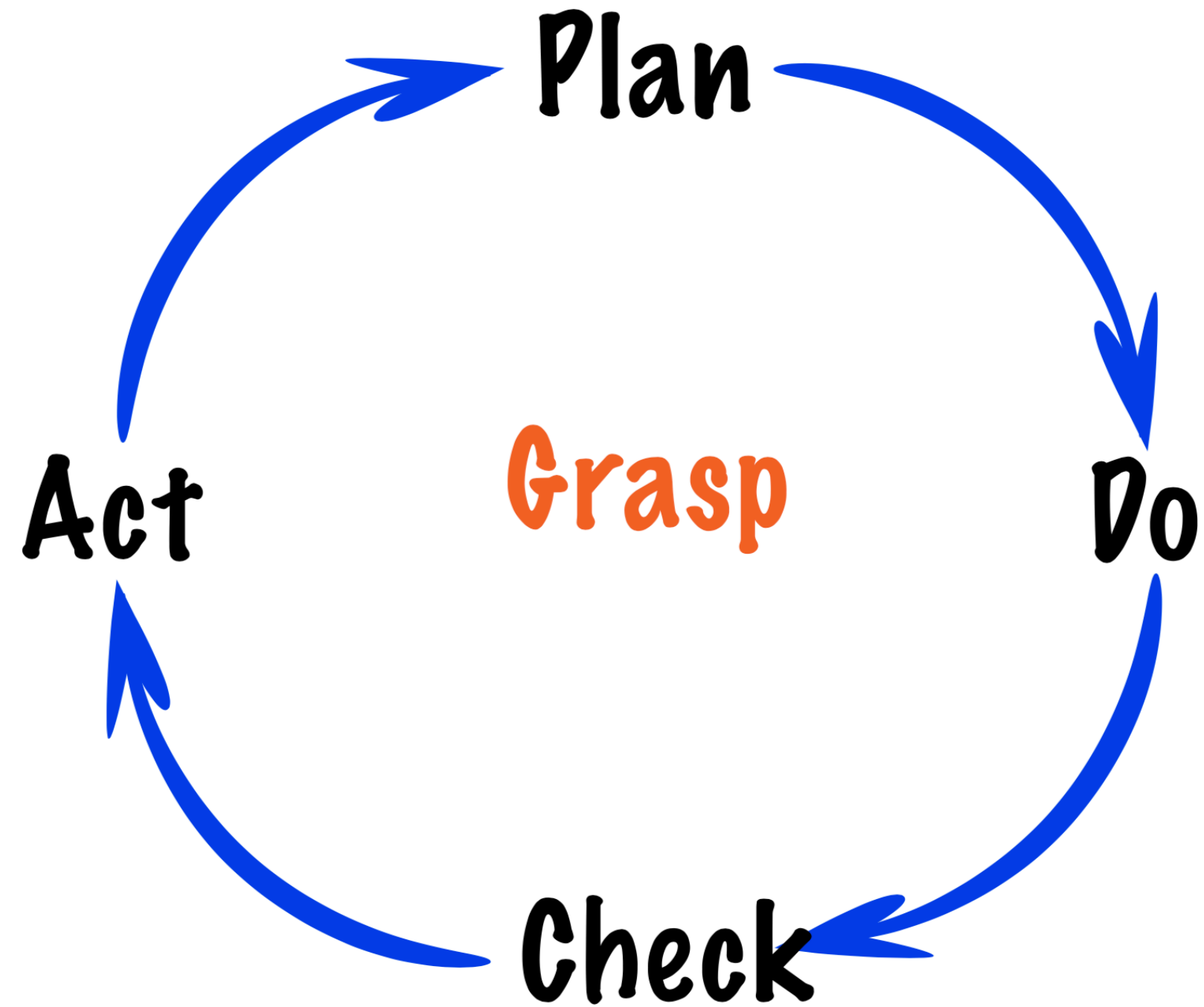
© Pierluigi Pugliese

Empirical Vs Defined Process



© Pierluigi Pugliese

Continuous Improvement



© Pierluigi Pugliese





Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it.

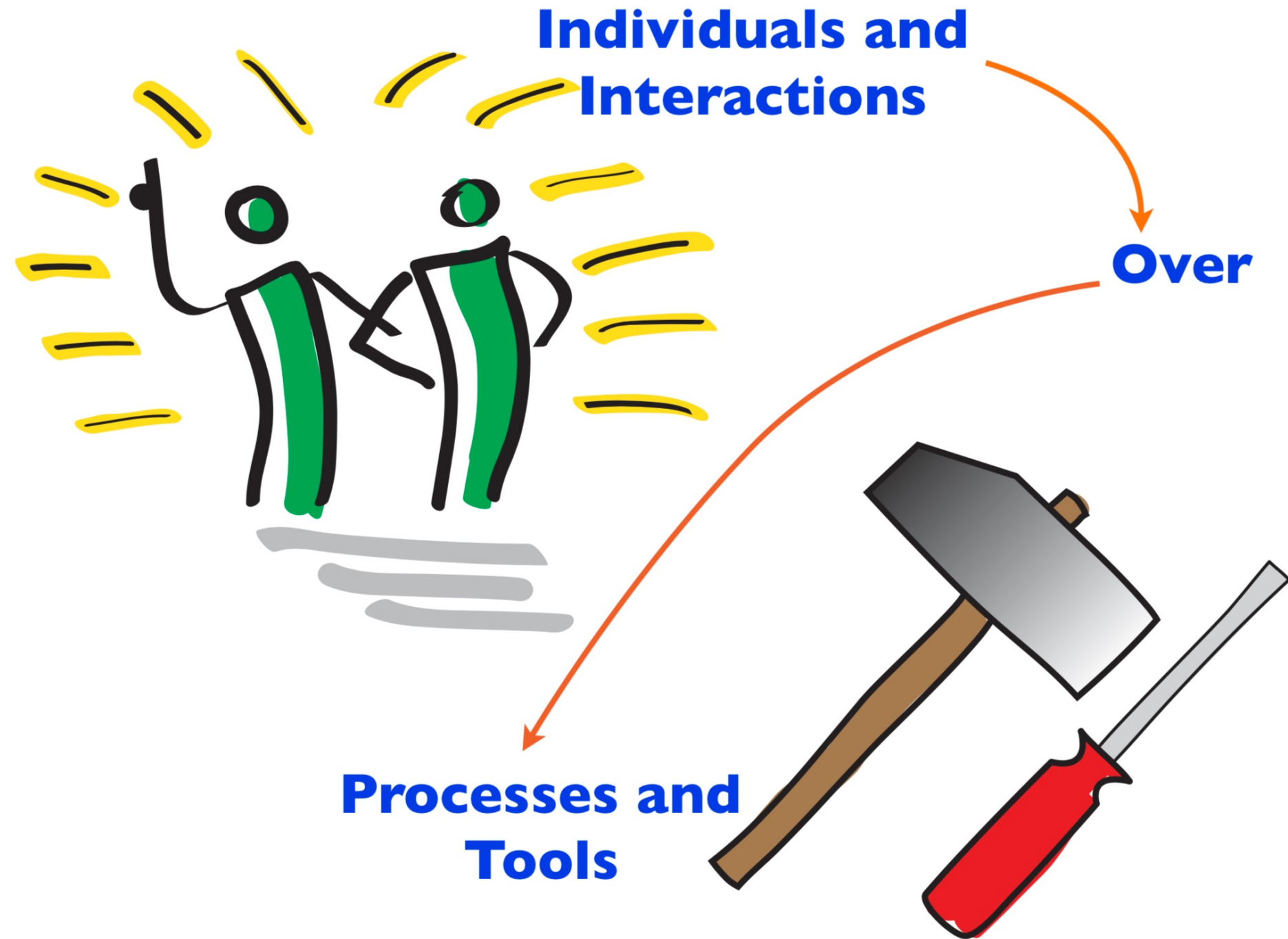
Through this work we have come to value:

[X over Y]

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

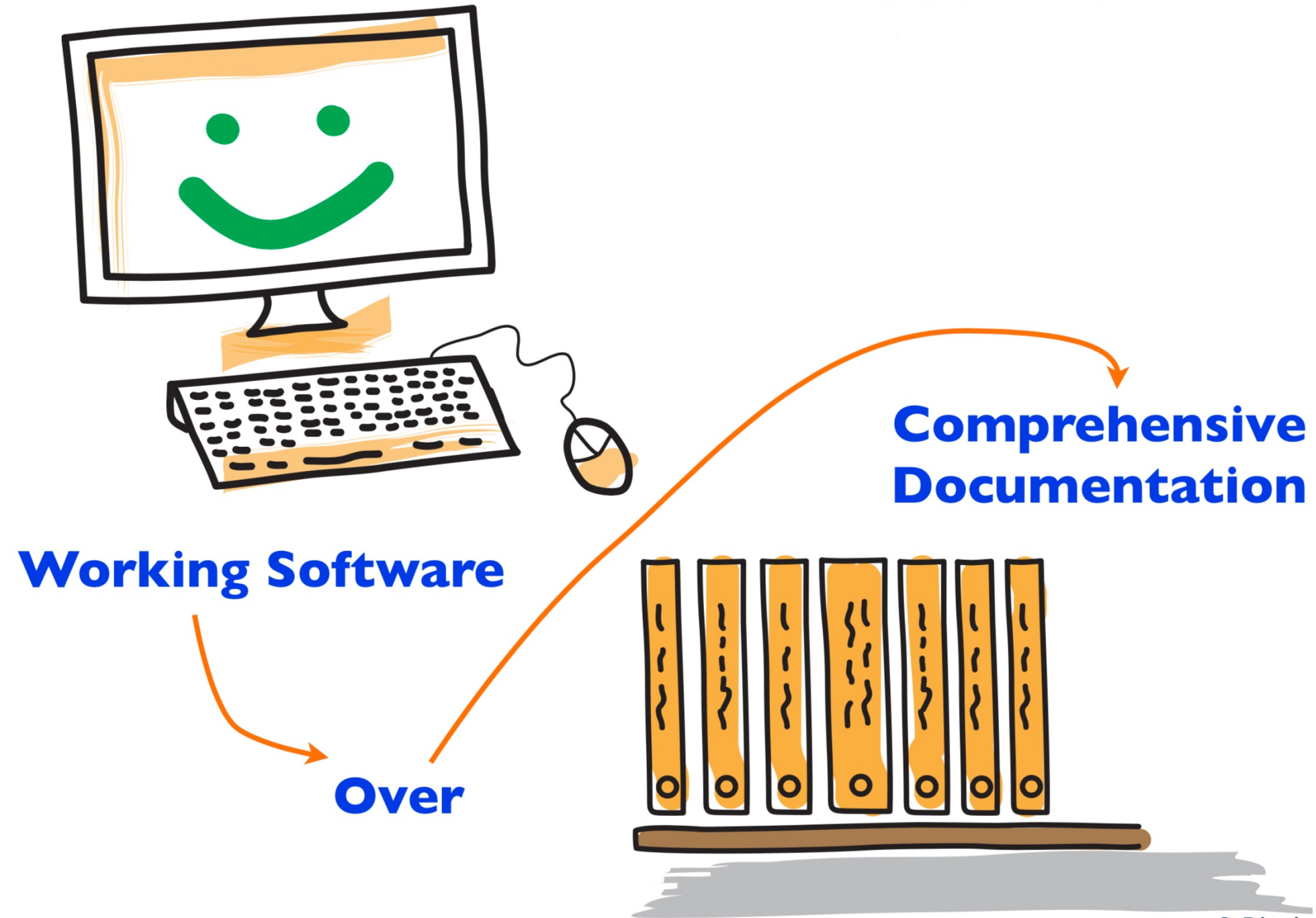


We value...



© Pierluigi Pugliese

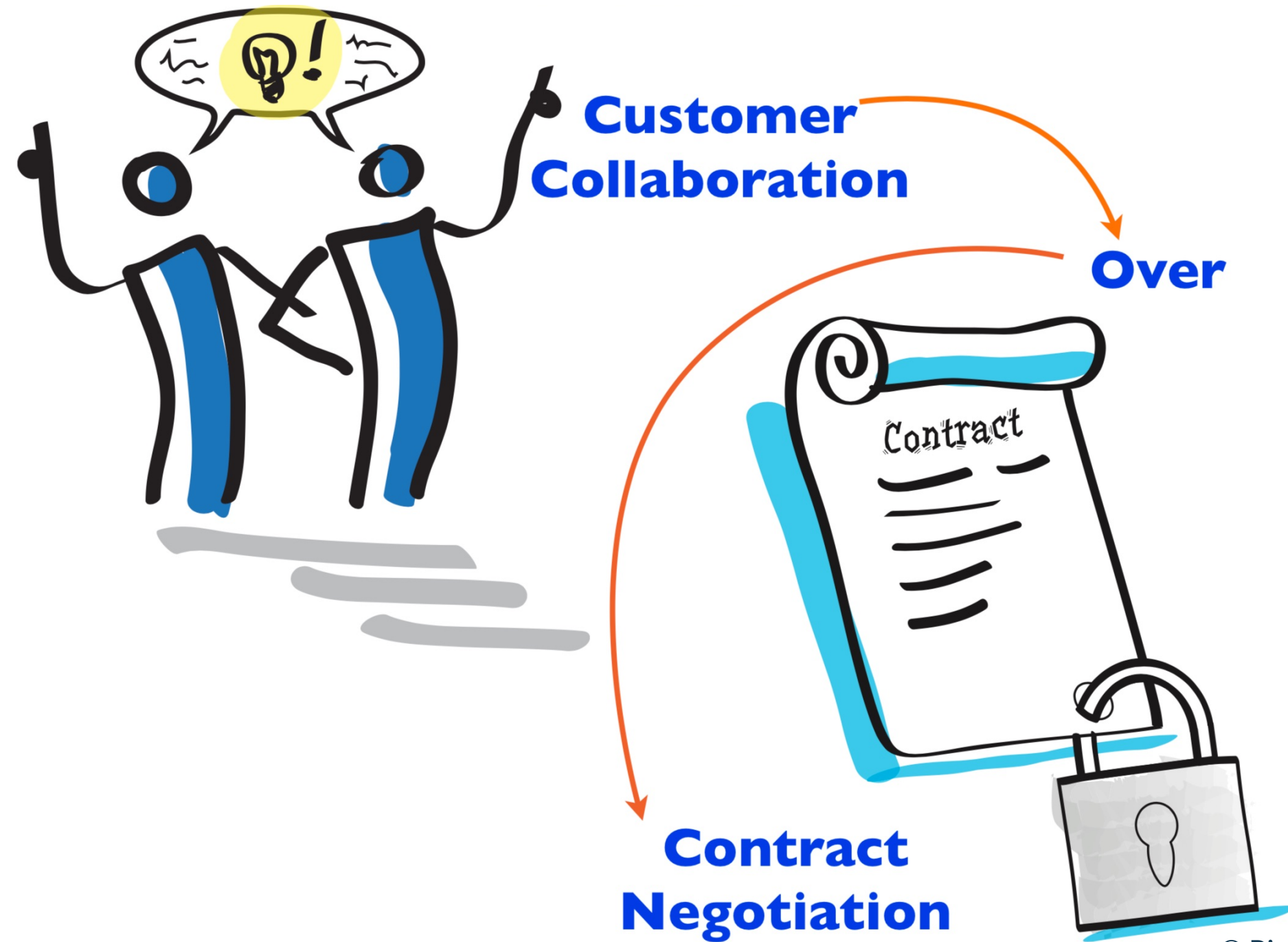
We value...



© Pierluigi Pugliese

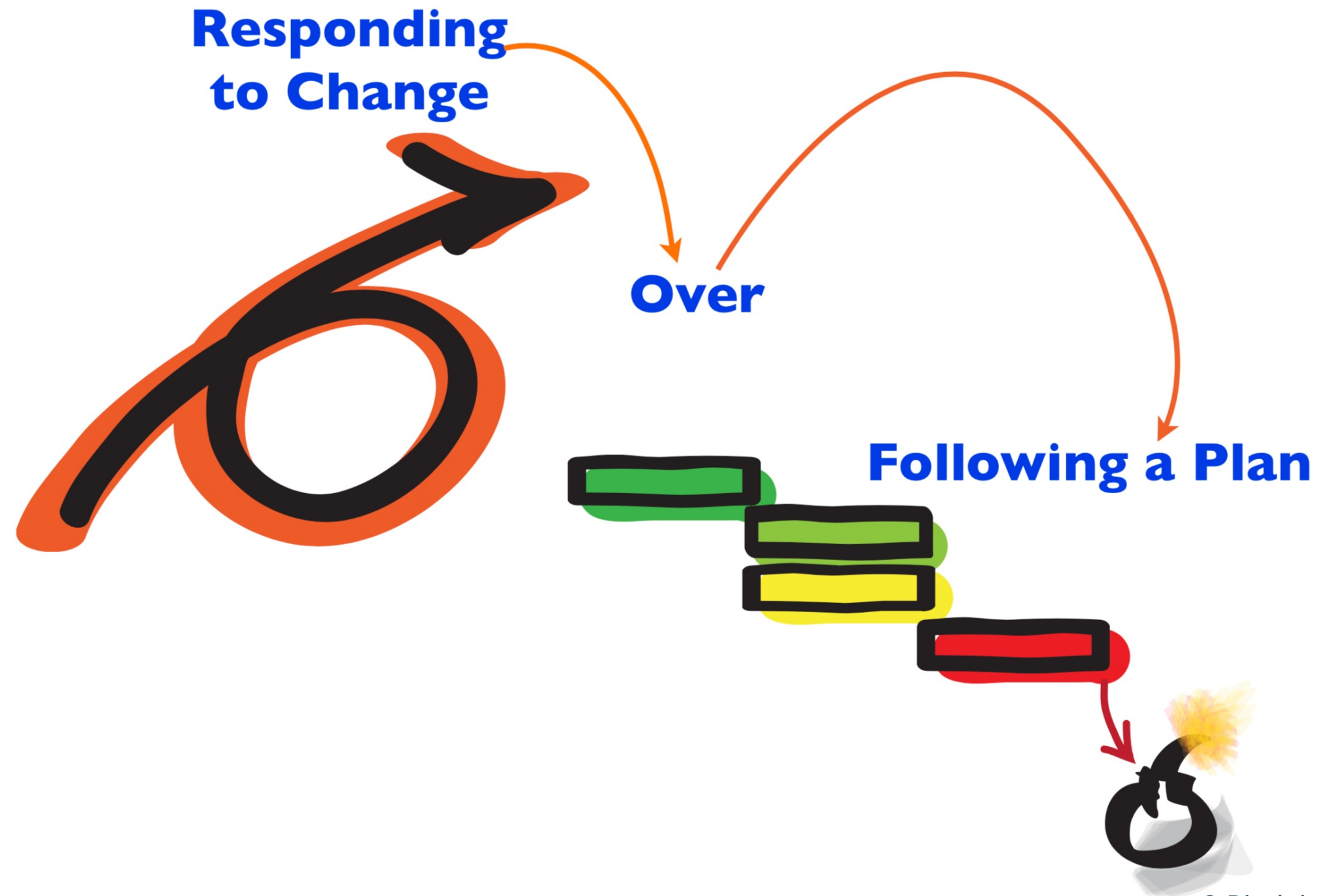


We value...



© Pierluigi Pugliese

We value...



© Pierluigi Pugliese



Agile Manifesto Principles

1	Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.	4	Business people and developers must work together daily throughout the project.
2	Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	5	Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
3	Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.	6	The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.



Agile Manifesto Principles

7	Working software is the primary measure of progress.	10	Simplicity – the art of maximizing the amount of work not done – is essential.
8	Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.	11	The best architectures, requirements, and designs emerge from self-organizing teams.
9	Continuous attention to technical excellence and good design enhances agility.	12	At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

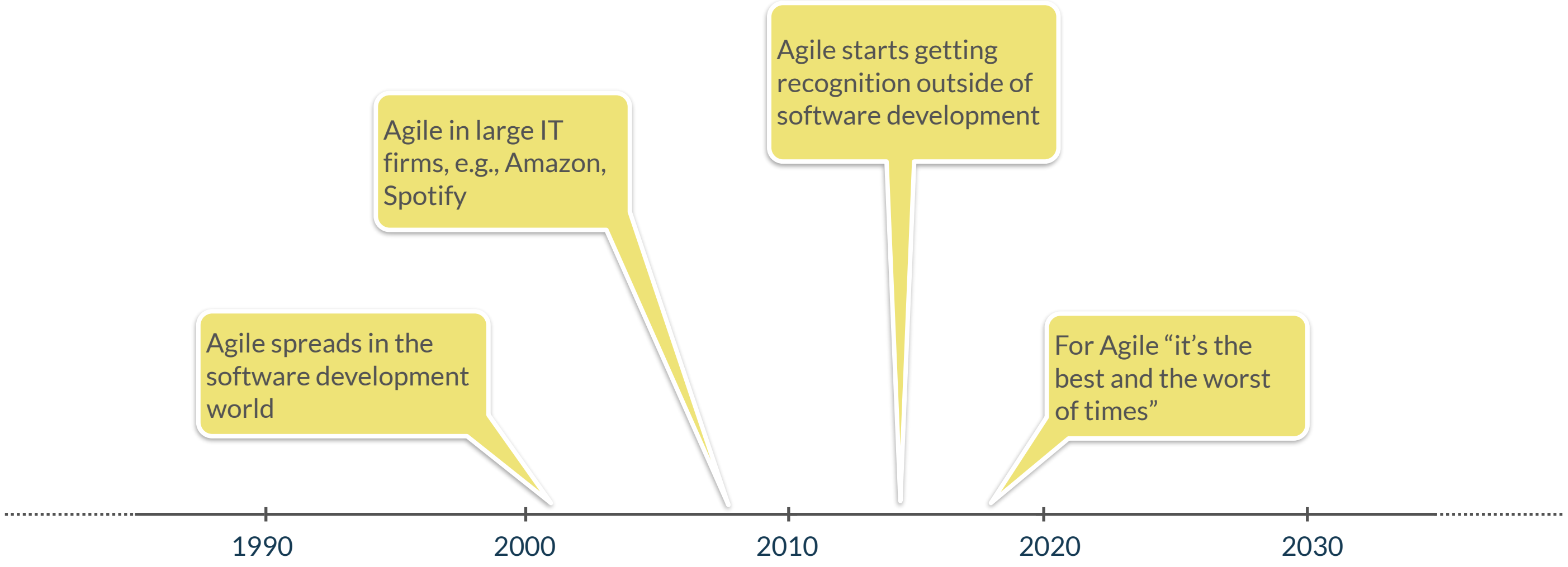




What happened after 2001?



During the last 18 years



Agile is enjoying both

The best of times

- Ways to deliver instant, intimate, incremental, risk-free value at scale
- Spreading from IT Department to all parts, and all kinds, of organizations
- Ridicule of Agile turned to envy

The worst of times

- Agile implemented as a superficial patch on traditional management
- Some consultant and coaches are selling “get Agile quick” schemes
- Huge amount of “fake Agile” going on
- Risk that Agile is being dumbed down as to become a shadow of the real thing





Quick introduction to Scrum



Takeuchi and Nonaka

“The New New Product Development Game”

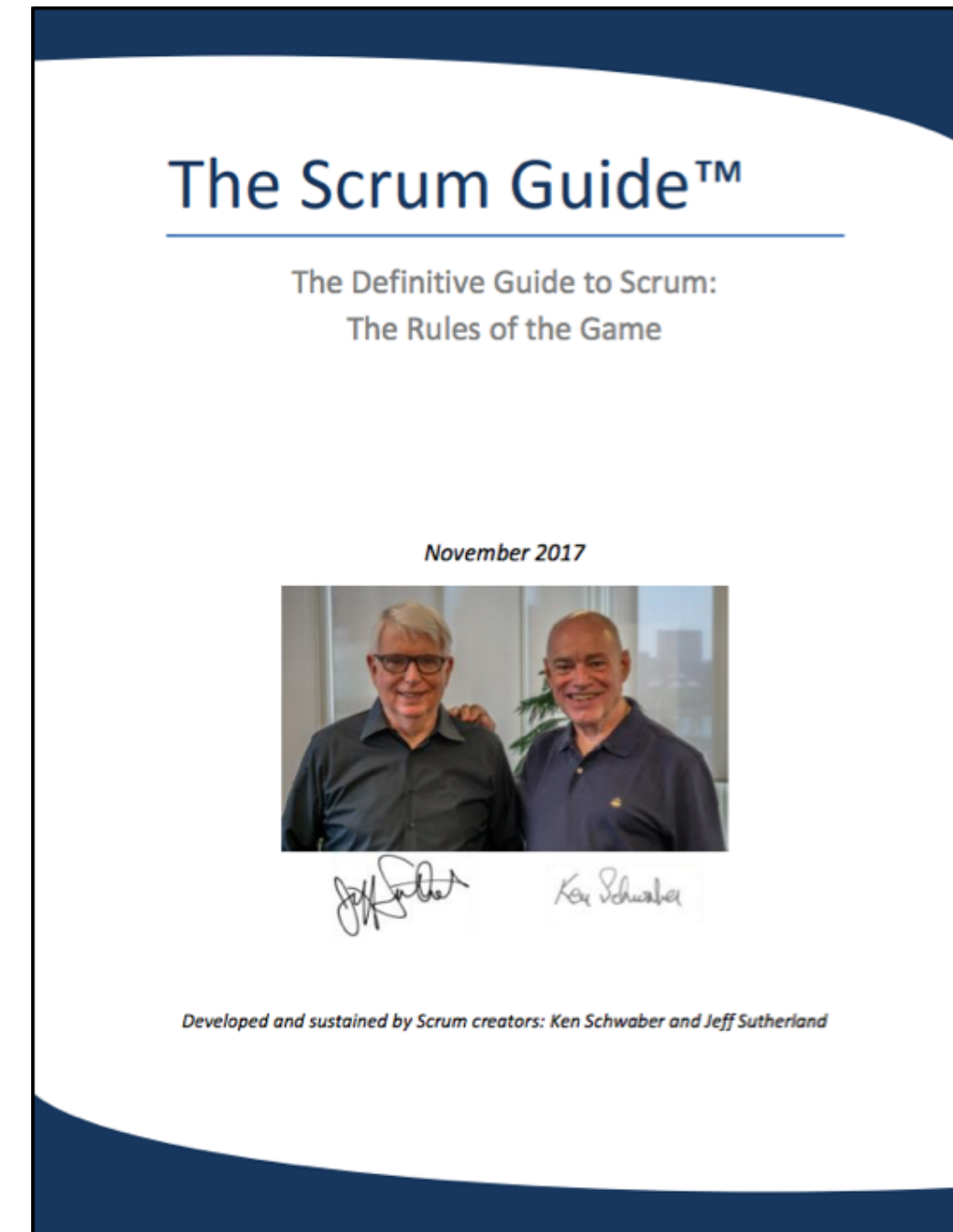
- Used the term Scrum
- Referred to the game of rugby to stress the importance of teams
- Their research showed that teams requires autonomy to achieve excellence



Scrum is...

A lightweight framework for project management.

- Few but clearly defined roles
- Self-organizing team
- Time boxed iterations
- Founded on empirical process control theory
- Simple to understand
- Difficult to master

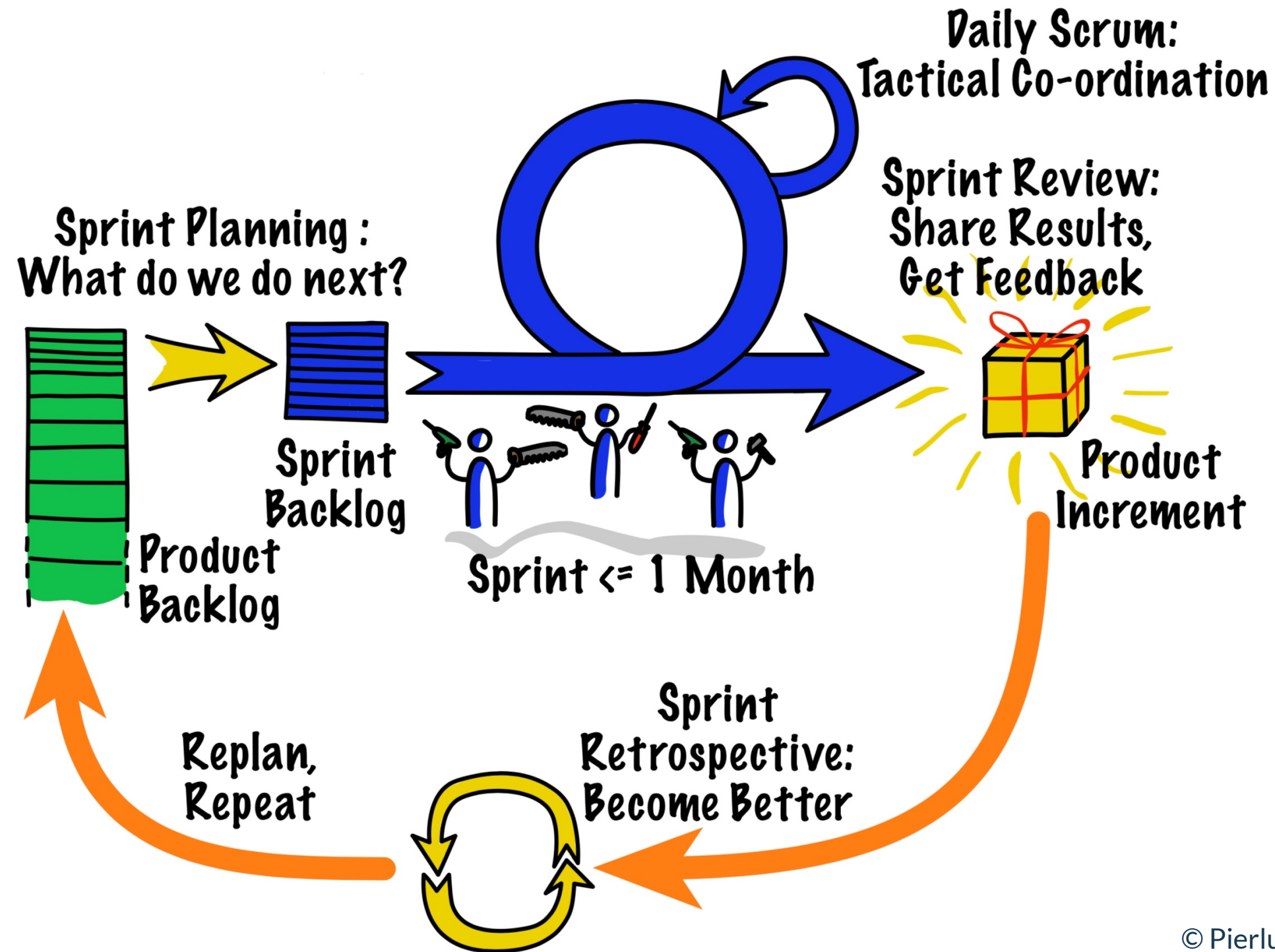


Scrum is not a process or a technique for building products; rather, it is a framework within which you can employ various processes and techniques. Scrum makes clear the relative efficacy of your product management and development practices so that you can improve.

Ken Schwaber and Jeff Sutherland - The Scrum Guide™



Process Overview



© Pierluigi Pugliese



References



Agile Manifesto

<https://agilemanifesto.org>

Scrum Guide

Jeff Sutherland, Ken Schwaber

<https://scrumguides.org>