NTT DATA
Trusted Global Innovator

INTRODUCTION TO AGILE METHODOLOGIES AND MINDSET

03/12/2019

Agenda

- **What is the Agile Methodology**
- The Agile manifesto and its principles
- Business Analysis Agile
- Scrum
  - Scrum Roles
  - Scrum Events
  - Scrum Artifacts
- Scaled Agile Framework

Success Projects With Agile Approach

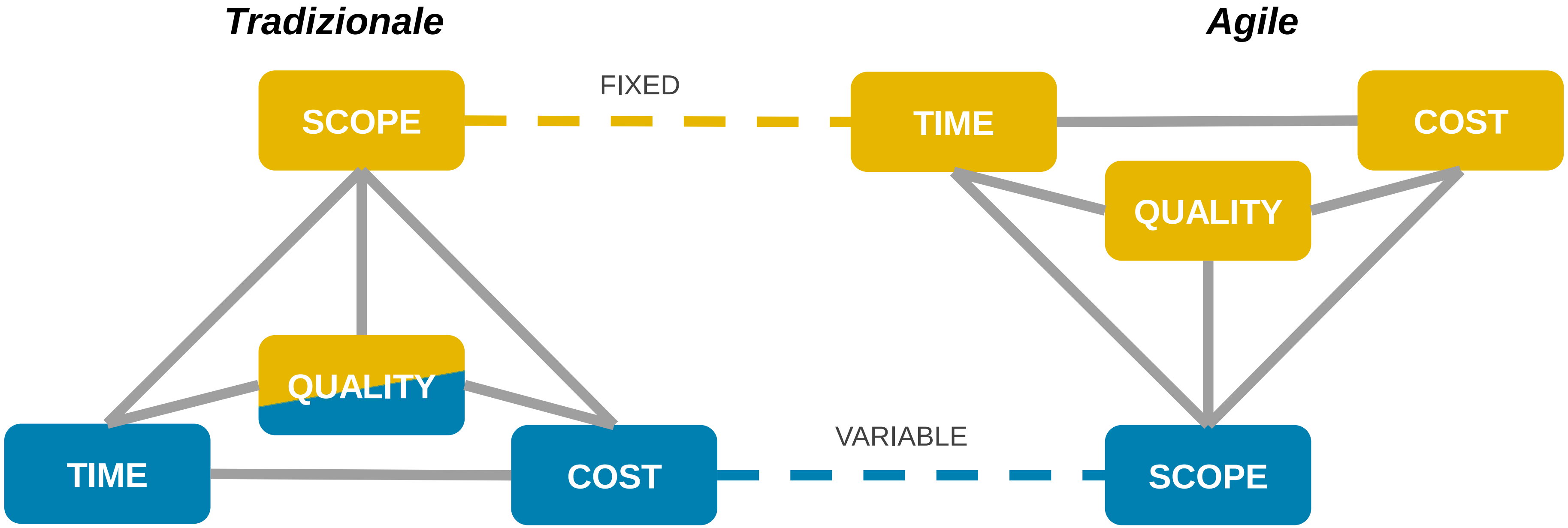
Agile allows to complete projects with success.

Dimensions	Metodo	Success	Partly of success	Failure
Projects	Agile	39%	52%	9%
	Waterfall	11%	60%	29%
Big projects	Agile	18%	59%	23%
	Waterfall	3%	55%	42%
Medium Projects	Agile	27%	62%	11%
	Waterfall	7%	68%	25%
Small Projects	Agile	58%	38%	4%
	Waterfall	44%	45%	11%

Data source: Chaos report 2015 on 10.000 projects

The Triangle Of Planning

Agile inverted the triangle: time and cost have been placed at forefront



Time and cost are fixed while the scope is variable

Agile Thinking

> Do the planning,
> But throw out the plans.

Mary Poppendieck

- Agile was created to **EVOLVE** and **ADAPT** immediately to dynamic market conditions
- Often traditional Project Management is **INEFFICIENT** in ensuring the **SUCCESS**
- Since 2001, Project Management **AGILE METHODOLOGIES** have been developed
- An agile transformation predicts an **APPROACH’S CHANGE** across enterprise
- Agile methodology entails significant changes in **COMPANIES’ CULTURE**
- Agile allows to **REDUCE COSTS** and **TIME** of software development, increasing the **QUALITY**

Agile Vs Traditional Project Management

TRADITIONAL PROJECT MANAGEMENT

- Often don’t respect time and costs
- Few projects end respecting the scope
- Over 60% IT projects develops functionalities not used frequently*

*Standish group analysis

About **40%** of IT **PROJECTS** managed with traditional methodologies (Waterfall) **DOES NOT** end with a **FULL SUCCESS***

AGILE

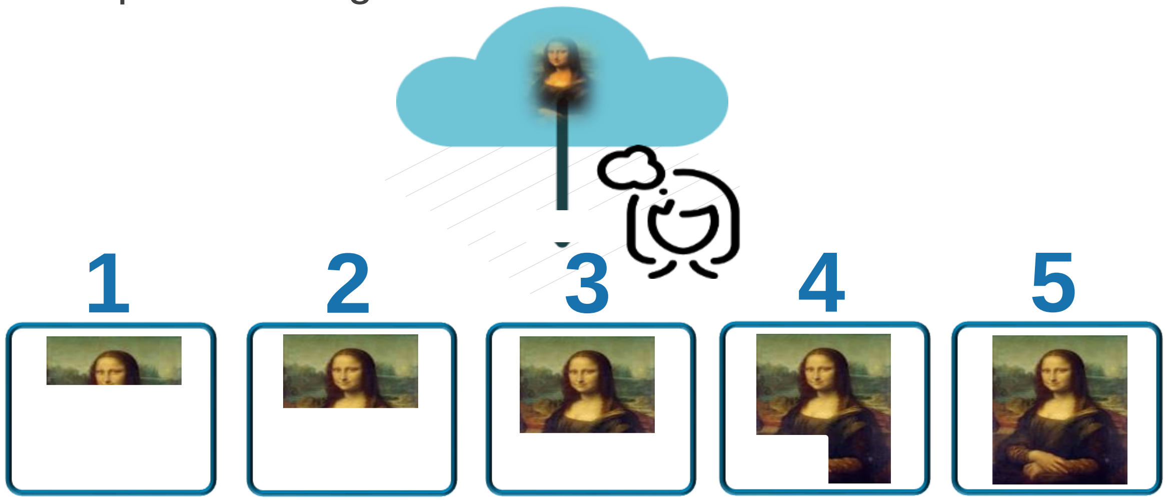
- Paradigm shift
- Time and costs are fixed, while the scope is variable
- The features wich give more value to business have the priority

Agile allows to manage **EFFICIENTLY** all **COMPANY PROCESSES**, adopting them quickly to the changes required by market

Incremental Approach Vs Iterative Approach

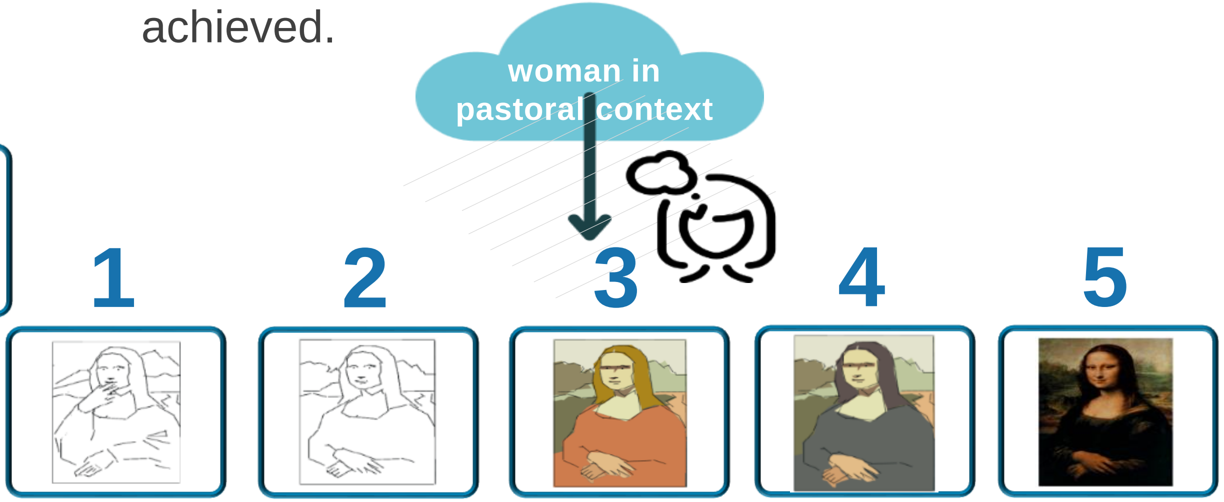
INCREMENTAL

The incremental approach requires a complete and fixed concept of the target to be achieved.



ITERATIVE-ADAPTATIVE

The iterative approach involves the "construction" through following versions, their validation up to the construction of the final result, with quality. It allows you to work around an idea that is not completely defined, gradually including changes until the desired result is achieved.



Agenda

- What is the Agile Methodology
- **The Agile manifesto and its principles**
- Business Analysis Agile
- Scrum
  - Scrum Roles
  - Scrum Events
  - Scrum Artifacts
- Scaled Agile Framework

The 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**

Traditional		**Agile**
Processes and Tools	**over**	People and Interactions
Comprehensive Documentation	**over**	Working Software
Contract Negotiation	**over**	Customer Collaboration
Following a Plan	**over**	Responding to Change

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

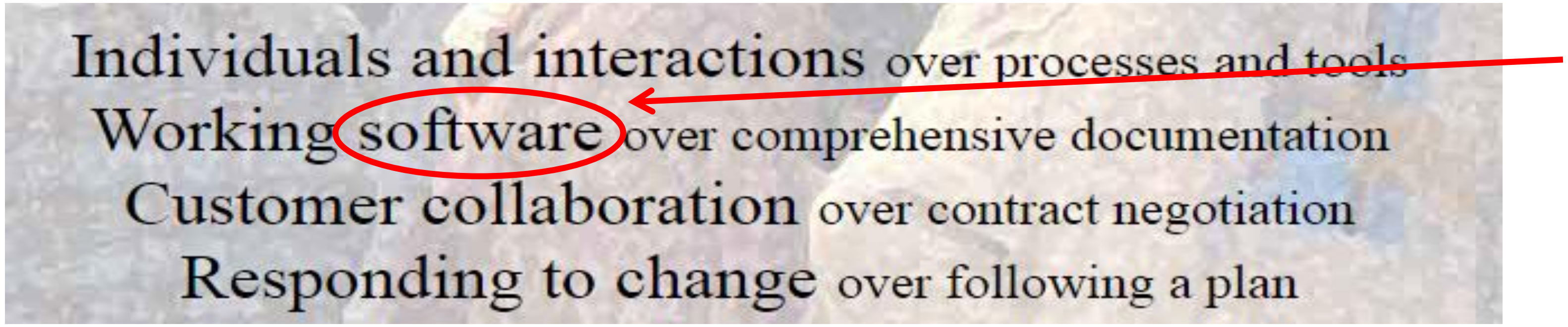
Agile Principles

1. Customer satisfaction by rapid delivery of useful software
2. Welcome changing requirements, even late in development
3. Working software is delivered frequently (weeks rather than months)
4. Close, daily cooperation between business people and developers
5. Projects are built around motivated individuals, who should be trusted
6. Face-to-face conversation is the best form of communication (co-location)
7. Working software is the principal measure of progress
8. Sustainable development, able to maintain a constant pace
9. Continuous attention to technical excellence and good design
10. Simplicity – the art of maximizing the amount of work not done – is essential
11. Self-organizing teams
12. Regular adaptation to changing circumstances

History The Agile Manifesto

Some authors have tried thereafter update the Agile Manifesto.

Agile Manifesto (2001)



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

The word that has limited the Agile approach is "Software".
Prince2® instead talks about ***working solutions***

Disciplined Agile Manifesto (2010)

Individuals and interactions over processes and tools
Working ***solutions*** over comprehensive documentation
Stakeholder collaboration over contract negotiation
Responding to change over following a plan

The Disciplined Agile Manifesto is included in the IBM Developer works documents

Beyond Agile Manifesto (2011)

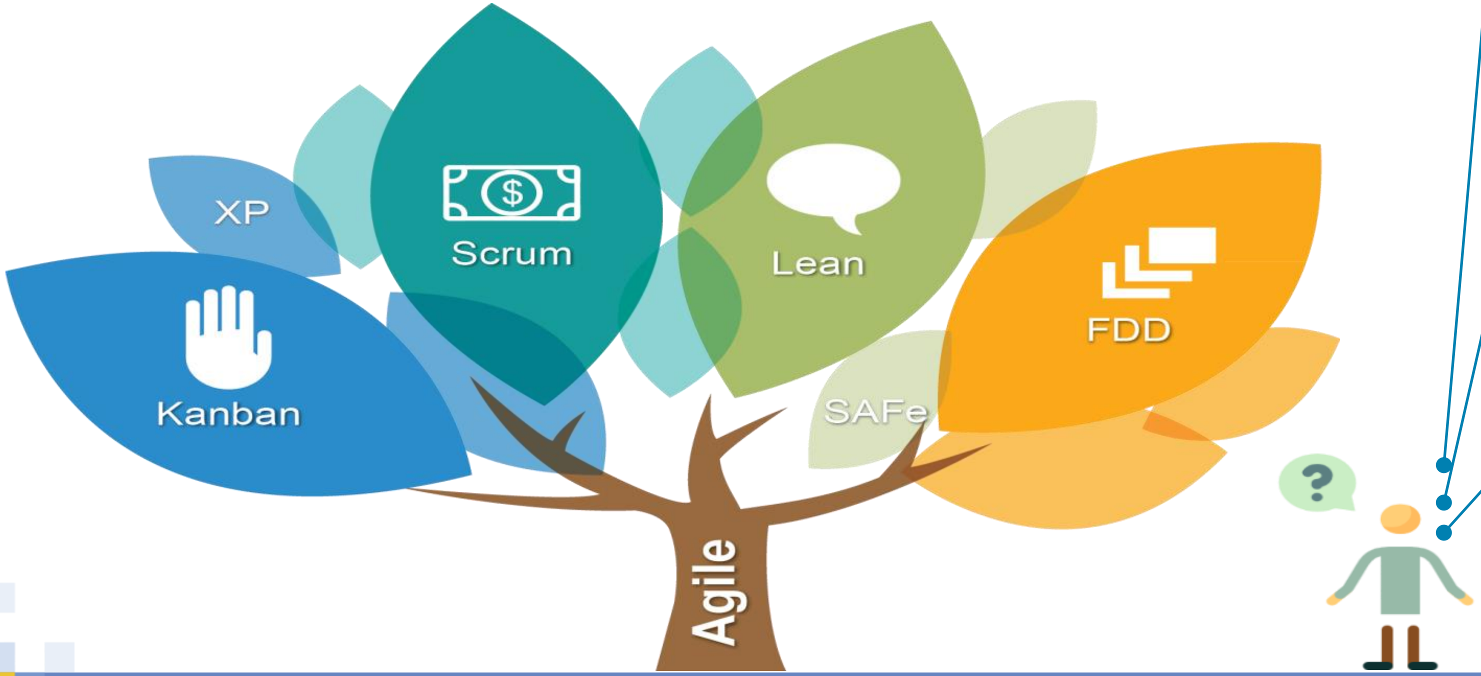
Team vision and discipline over individuals and interactions (over processes and tools)
Validated learning over working software (over comprehensive documentation)
Customer discovery over customer collaboration (over contract negotiation)
Initiating change over responding to change (over following a plan)

Created by ***Kent Beck***, one of the authors of the Agile Manifesto and creator of the XP methodology

Agile Environment

The ***"WORLD" AGILE*** is very complex in terms of frameworks, methods and practices.

The AGILE methodologies, mainly used by start-ups for a long time, are actually now also used by large companies.



FRAMEWORK – LARGE SCALE METHODS

Enterprise Scrum
LeSS, Large Scale Scrum
Nexus, Scaled professional Scrum
SAFe, Scaled Agile Framework
Scrum @ Scale
Scrum of Scrum
Setchu, Scrum-based lightweight framework
Xscale

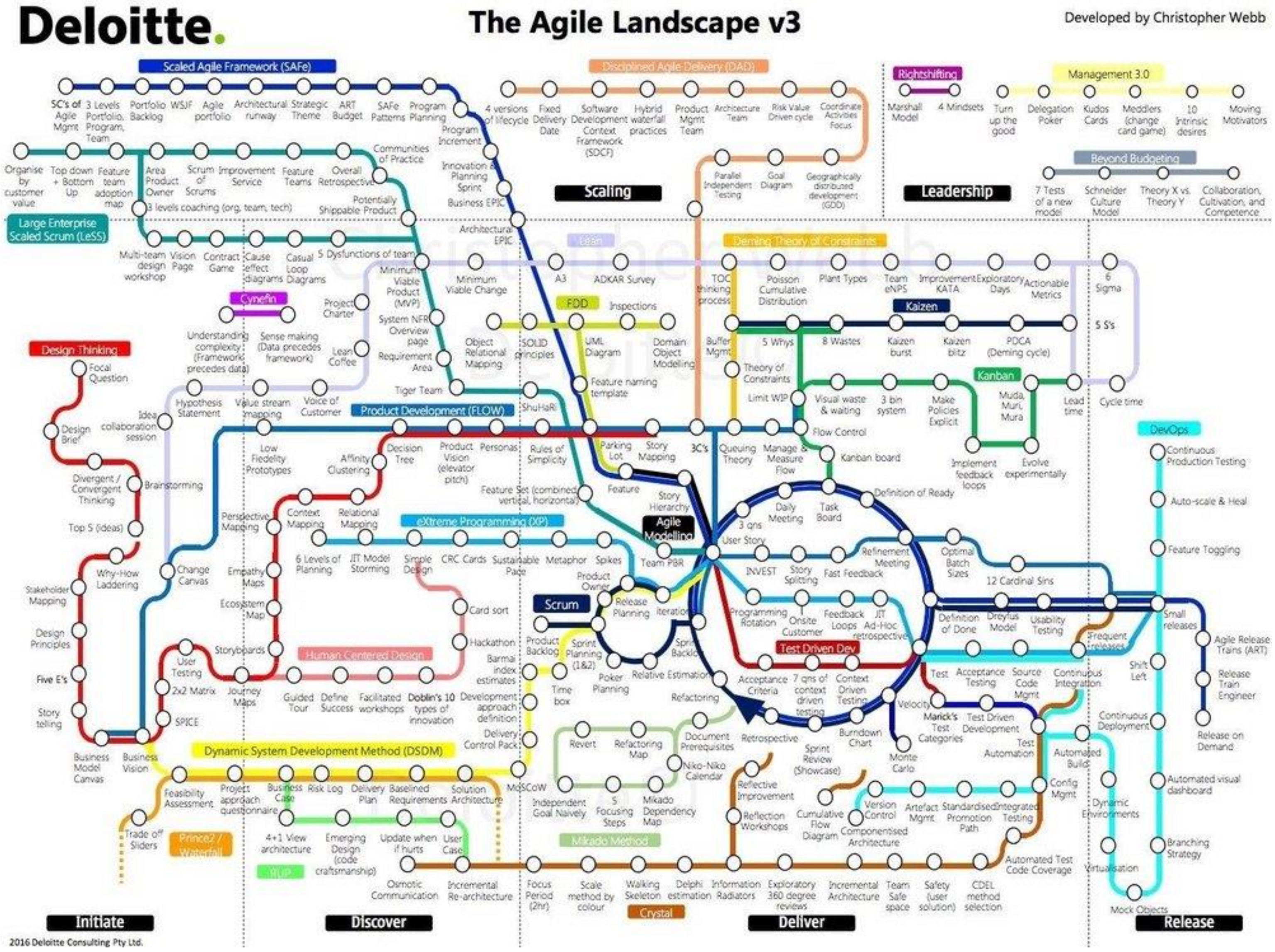
METHODS

ASD, Adaptive Software Development
Agile Modeling
AUP, Agile Unified Process
BADM, Business Analyst Designer Method
Crystal Clear Methods
DAD, Disciplined Agile Delivery
DSDM, Dynamic System Development Method
XP, eXtreme Programming
FDD, Feature Driven Development
Kanban
Scrum
Scrumban

PRACTICES

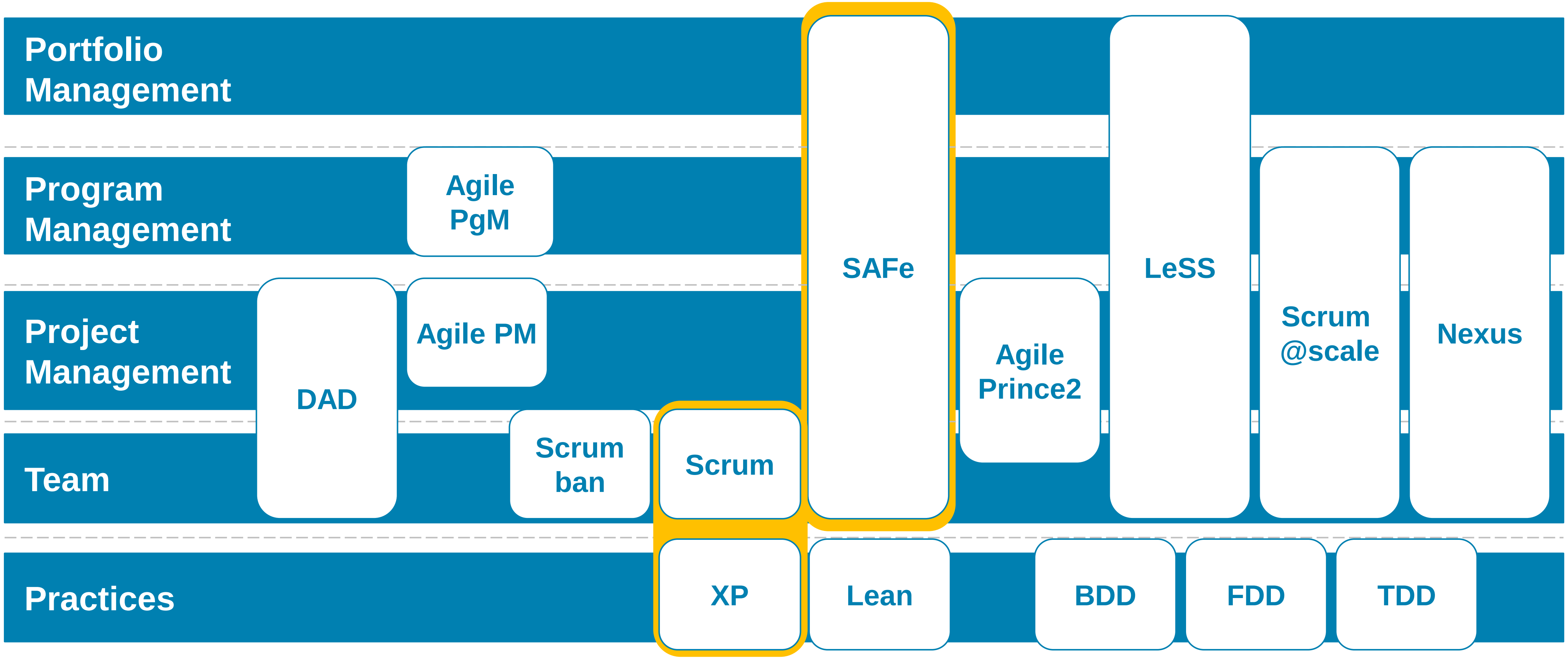
ATDD, Acceptance Test Driven Development
BDD, Behavior Driven Development
CD, Continuous Development
CI, Continuous Integration
CT, Continuous Testing
DDD, Domain Driven Development
IID, Iterative and Incremental Development
TDD, Test Driven Development

The agile landscape



Agile Blueprint

In the AGILE world different procedures, methods and framework can be adapted to different contexts.

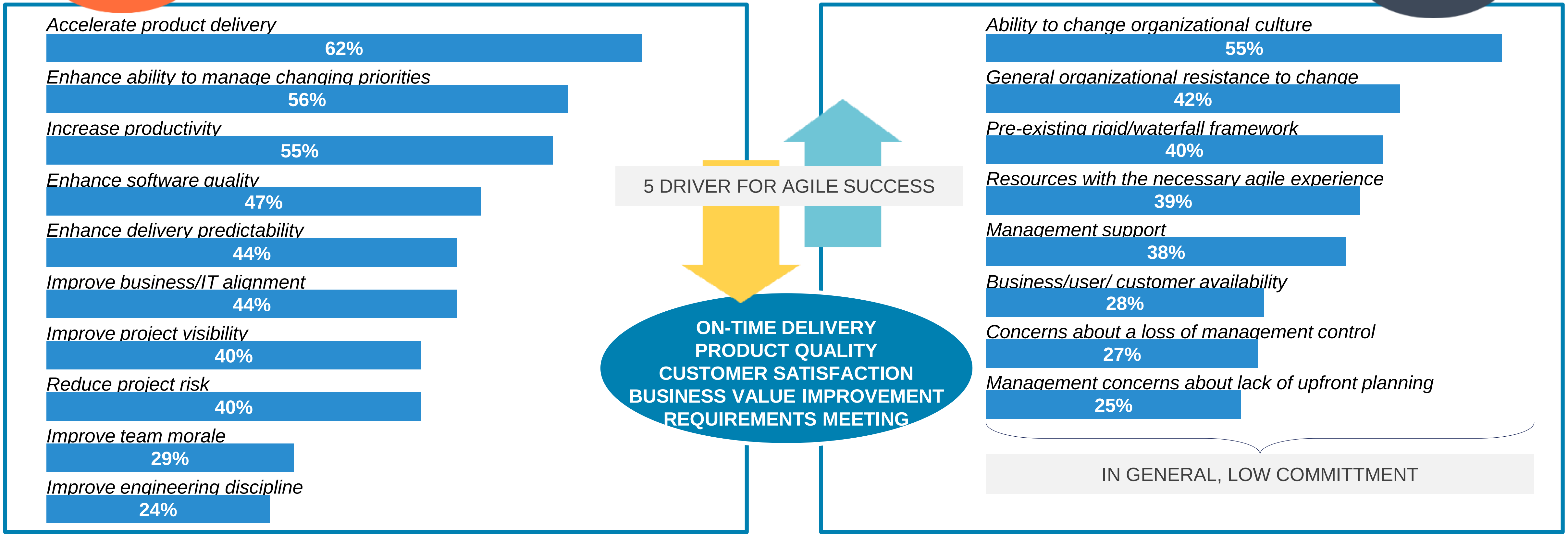


THE MOST USED METHODOLOGIES: in a recent survey it came to light that the most used methodologies are Scrum 58% and Scrum/XP hybrid for 10%.

Why Companies Choose Agile And Problems They Meet

THE FIRST 10 REASONS FOR WHICH COMPANIES DECIDE TO ADOPT AGILE

THE FIRST 10 REASONS FOR WHICH PROJECTS FAILED



5 DRIVER FOR AGILE SUCCESS

ON-TIME DELIVERY
PRODUCT QUALITY
CUSTOMER SATISFACTION
BUSINESS VALUE IMPROVEMENT
REQUIREMENTS MEETING

IN GENERAL, LOW COMMITTMENT

In Italy there are the same expections and problems for the use Agile methodology

Source: 10 Annual State of Agile Survey by VersionOne (2016)

Several Agile Approaches

Following there are some of the most used approaches like Agile:

	Context	Key values
SCRUM	**Information Technology** Introduced in 1990 by Schwaber & Sutherland	• Time and cost keeping • Focus on most important deliverables • Iterative process
LEAN	**Industrial Production** Originated from TPS (Toyota Production System) developed since 1948	• Continuos improvement • Just In Time response • Waste and waiting time reduction
eXtreme Programming	**Information Technology** Introduced in 1999 by Beck	• Short feedback • Face to Face daily conversation • Easiness of the realized deliverable
Features Driven Development	**Information Technology** Introduced in 1997 by De Luca	• Deliverable oriented • Iterative process • Features progressive integration
Test Driven Development	**Information Technology** Introduced in 2003 by Beck	• Test driven development • Deliverable quality oriented • Reduction of bugs

OVER 60% OF ICT AGILE PROJECTS ARE MANAGED WITH SCRUM

Scrum

SCRUM is an agile, lightweight process for managing and controlling software and product development in rapidly changing environments.

Features

Framework	• Sprint Retrospective, Daily scrum meeting, Sprint review meeting, Sprint planning meeting
Pillars	• Trasparency • Ispection • Adaptation
Team	• Development team • Product Owner • Scrum master
Artifacts	• Product backlog • Sprint backlog • Definition of done
Events	• Sprint • Sprint planning meeting • Daily scrum • Sprint review • Sprint Retrospective

Lean

The Lean methodology has origins from the TPS (Toyota Production System) developed since 1948 to improve industrial production. It was then differentiated into Lean production and Lean product development.

Features

Principles

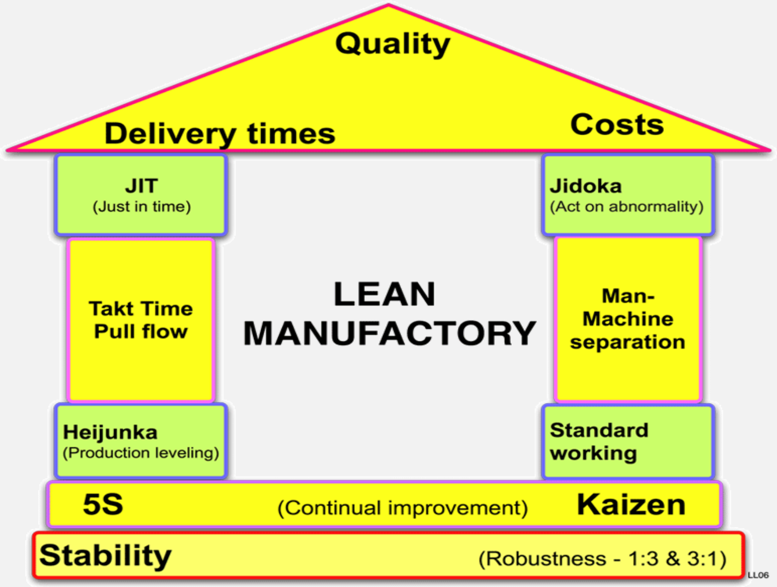
- Continuos improvement – kaizen
- Respect for people
- A long term vision
- The right process creates the right results
- Creating value through people development
- Respond immediately to problems

Pillars

- JIT - Just in time
- Automation

Scope

- Reduce waste
- Reduce waiting times
- Reduce overproduction
- Predict time boxing

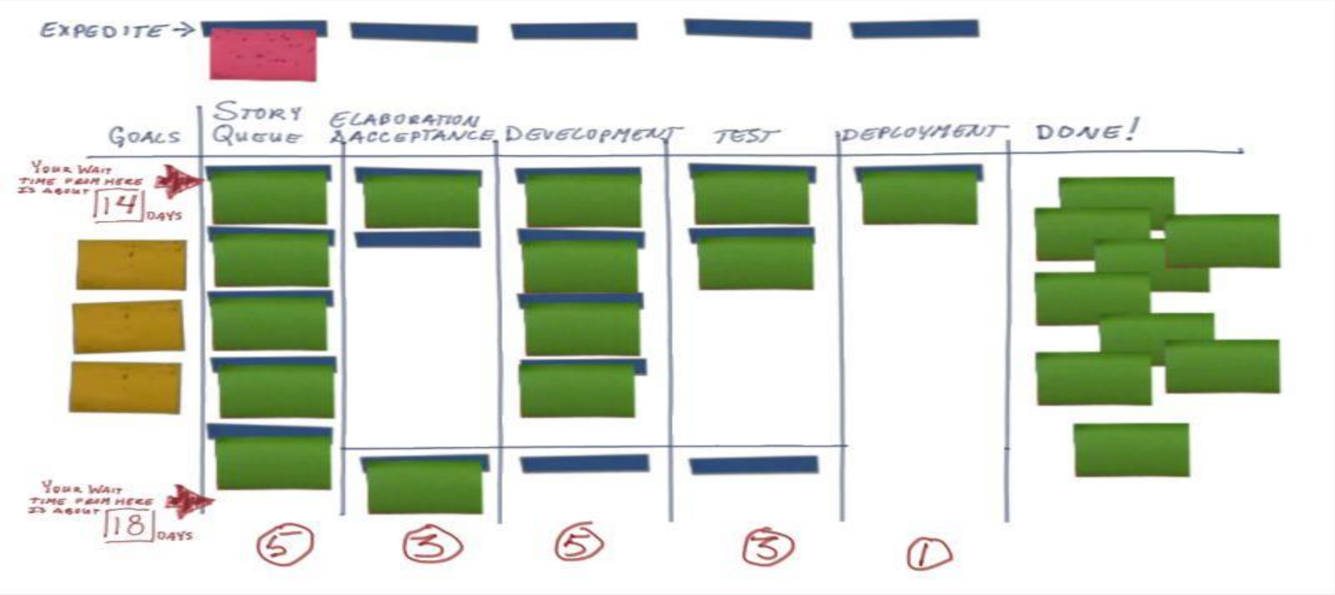


Kanban

Features

Scope

- View the stream, value stream mapping
- Restrict work in progress (WIP)
- Measure and optimize process



XP – Extreme Programming

The XP - eXtreme Programming methodology was introduced by Beck in 1999 in the IT area.

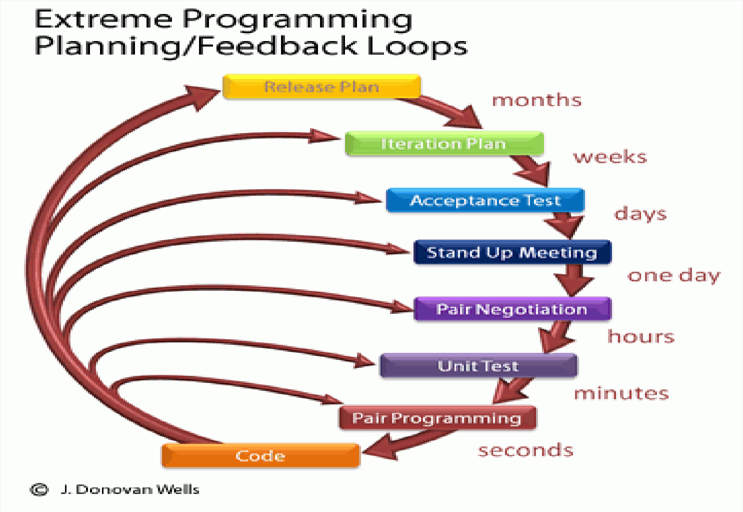
Features

Principles

- Simplicity, do the required thing in the easiest way
- Communication, daily face to face
- Feedback, continuous and transparent
- Respect, within the team and towards the other teams
- Courage, understood as openness, honesty and ethics

Best practices

- Short feedback
- Continuous process
- Shared understanding
- Team wellness



The Customer is actively involved, together with the team, in the process. ***Pair Programming*** *is a software development technique in which the two programmers work in pairs. The former writes the code and the second checks it in real time.*

FDD – Feature Driven Design

The FDD - Feature Driven Development methodology was introduced by De Luca in 1997 in the IT area.

Features

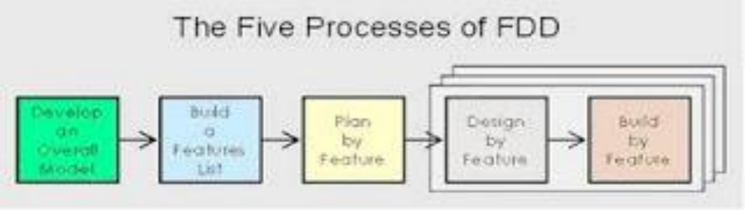
Processes

- Develop an overall model
- Build a features list
- Plan by feature
- Design by feature
- Build by feature

Iterations

The process is divided into iterations:

- Kick off Meeting del Package
- Design required classes/methods/documentations
- Design review
- Development and unit test
- Code review meeting
- Release meeting, starting integration process

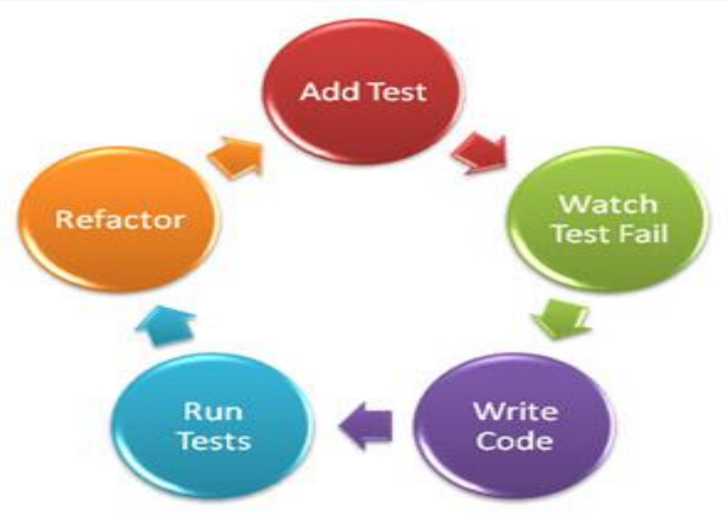


TDD – Test Driven Development

The TDD - Test Driven Development methodology was introduced for the first time by Beck in 2003 in the IT area.

Features

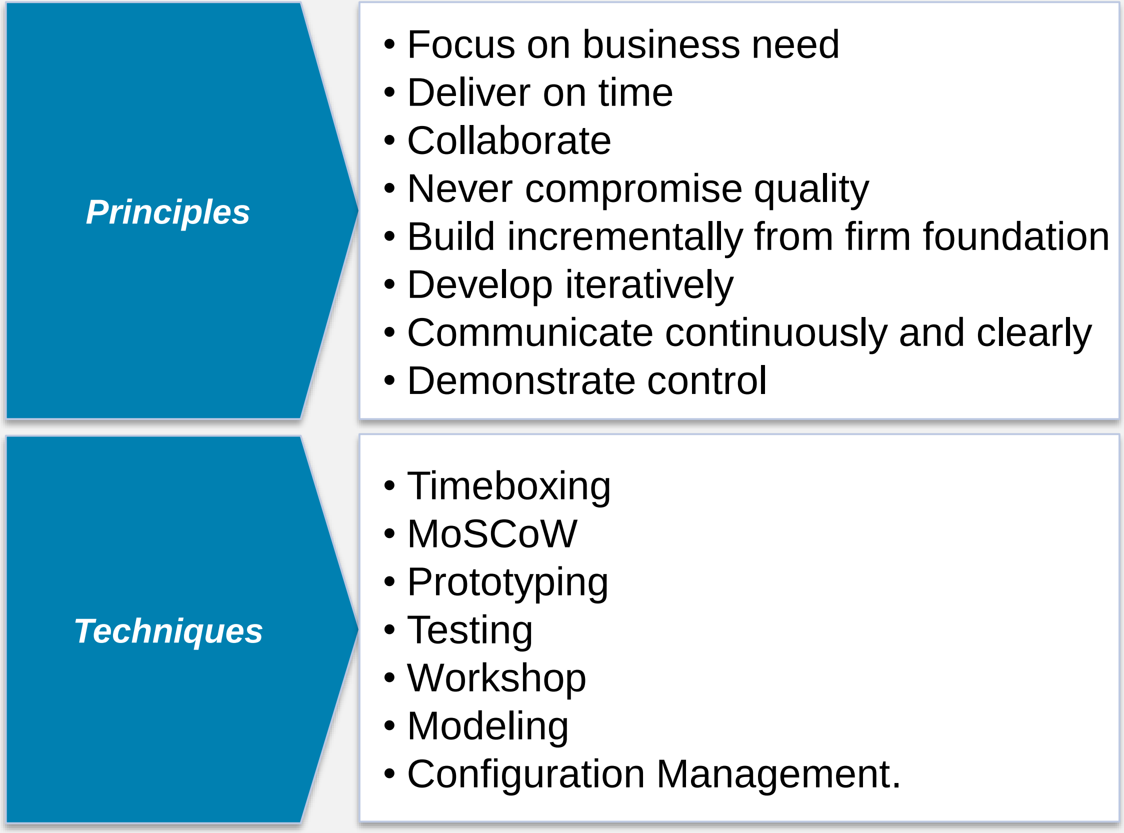
Approach	Unit test before starting to develop .
Step	• **Add test** • **Watch test fail** • **Write code** • **Run test** • **Refactor** • **Run test (optional)**
Outcomes	• Complete test of all the code • Significant reduction in the number of bugs in production • Increased quality of the final product



DSDM – Dynamic System Development Method

Agile methodology developed since 1994 and freely distributed by the DSDM Consortium.

Features



Principles

- Focus on business need
- Deliver on time
- Collaborate
- Never compromise quality
- Build incrementally from firm foundation
- Develop iteratively
- Communicate continuously and clearly
- Demonstrate control

Techniques

- Timeboxing
- MoSCoW
- Prototyping
- Testing
- Workshop
- Modeling
- Configuration Management.

DSDM®
www.dsdm.org

Crystal

Crystal is a family of Agile precise methodologies in the 90's by Alistair Cockburn.

Features

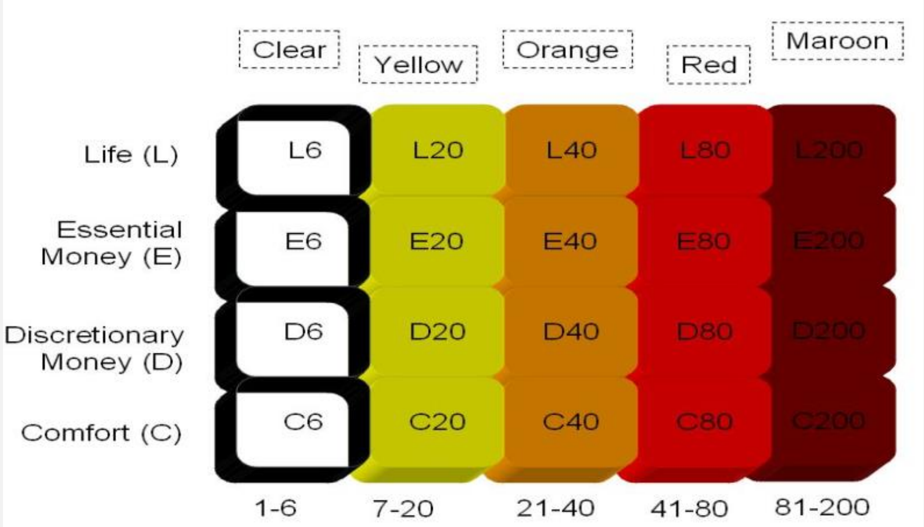
Families

Categorized through colors, according to the number of resources involved and the economic impact:

- Clear
- Yellow
- Orange
- Red
- Maroon

Principles

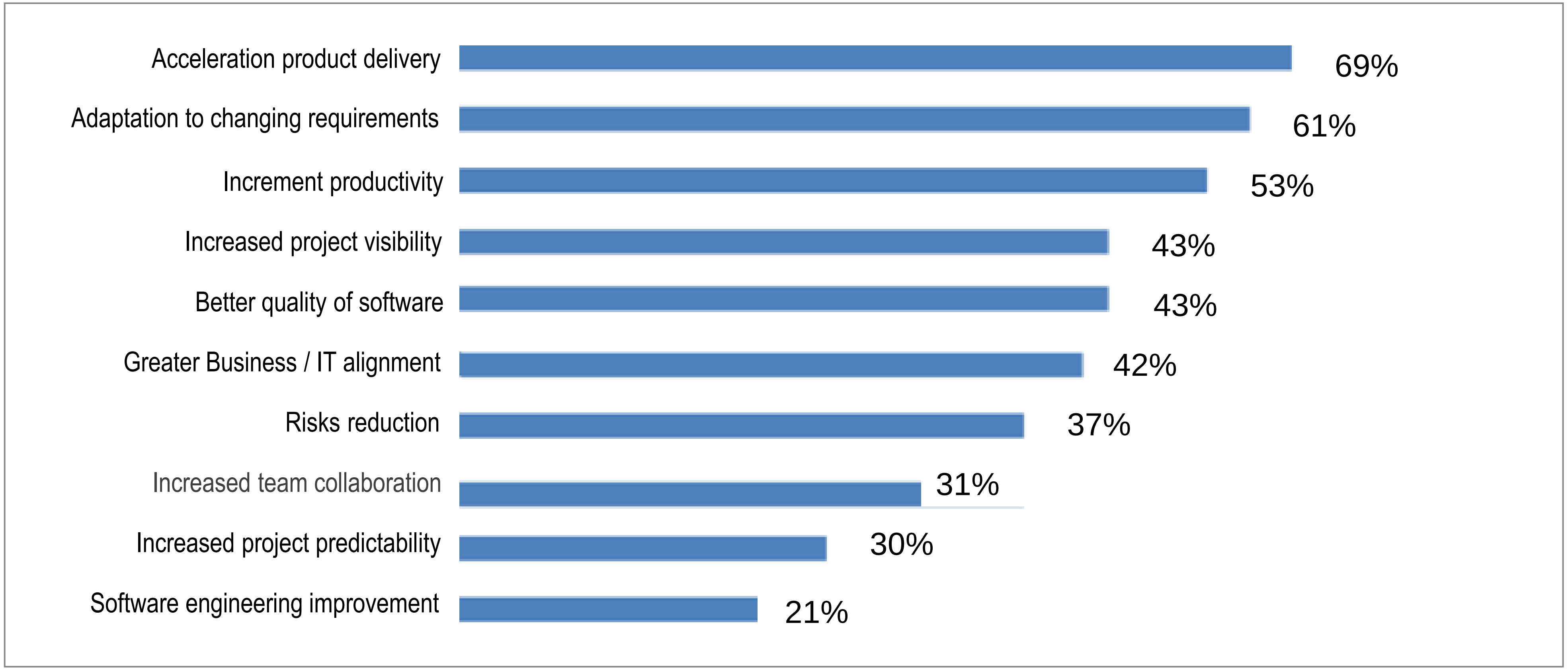
- Frequent delivery
- Reflective improvement
- Osmotic communication
- Personal safety
- Focus
- Easy access to expert users
- Technical environment



	Clear	Yellow	Orange	Red	Maroon
Life (L)	L6	L20	L40	L80	L200
Essential Money (E)	E6	E20	E40	E80	E200
Discretionary Money (D)	D6	D20	D40	D80	D200
Comfort (C)	C6	C20	C40	C80	C200
	1-6	7-20	21-40	41-80	81-200

Ten Reasons To Chose Agile

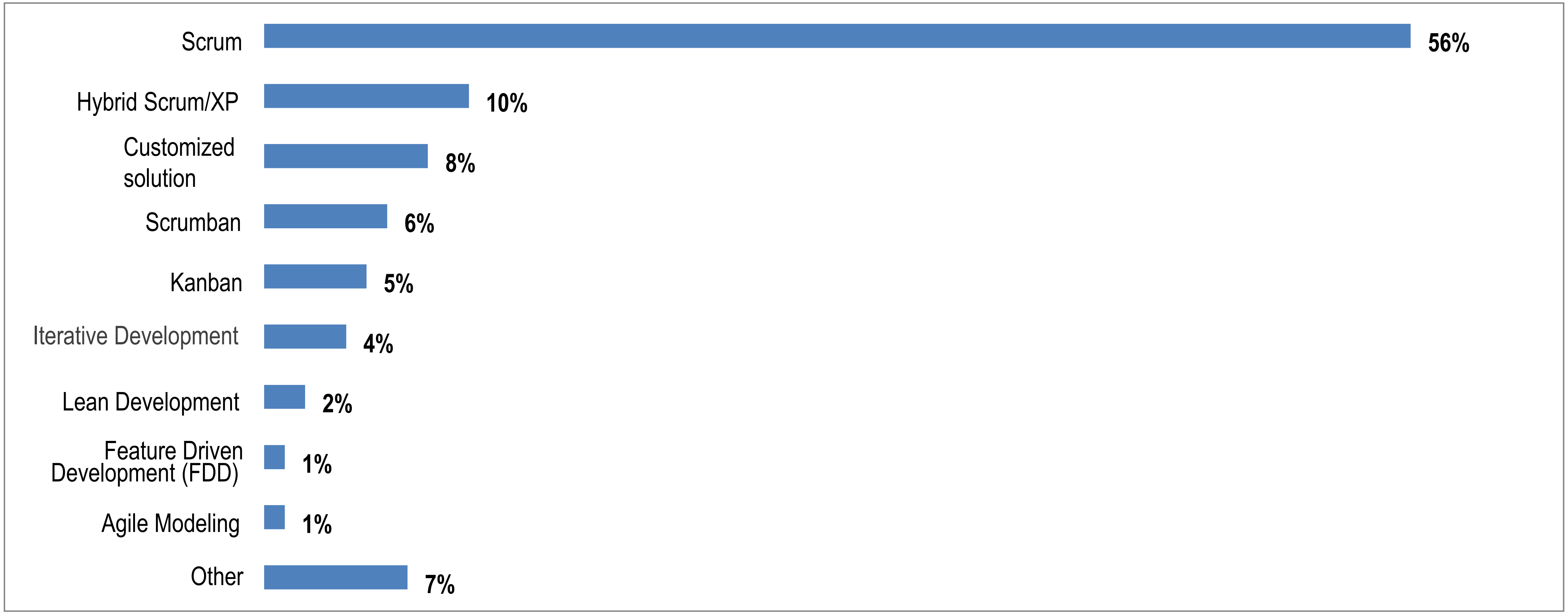
Following are the top reasons why companies decide to use an Agile approach in their projects.



Data source 11 Annual State of Agile Survey di VersionOne (2017)

Using Agile Methodologies

Scrum is confirmed as the most used methodology on the market. Many companies also tend to develop custom or hybrid solutions.



Data source 9 Annual State of Agile Survey di VersionOne (2015)

Agile Projects: Causes Of Failure

Causes of failure	%
Low Knowledge Agile approch	44
The company philosophy is in contrast with Agile	42
No support from management	38
External pressures for the use of traditional methodologies	37
Poor propensity for cultural change	36
Organizational or communication problems	33
Low team interst in adopting Agile	33
Insufficient training	30

Data source 9 Annual State of Agile Survey di VersionOne (2015)

Obstacles In Moving To Agile Software Development

Resistance to change	**%**
Ability to change organizational culture	44
Low availability of competent resources on Agile	35
Resistance to change	34
Rigidity of previous frameworks	32
Management support	29
Management concern for absence of plans	24
Resources availability	23
Concern over the absence of controls	22

Data source 9 Annual State of Agile Survey di VersionOne (2015)

Agile Success Factors

Success factors	%
On-Time delivery	58
Product quality	48
Customers/Users satisfaction	44
Increase business value	44
Compliance with the requirements	39
Project visibility	30
Productivity	29
Predictability	25
Process improvement	23

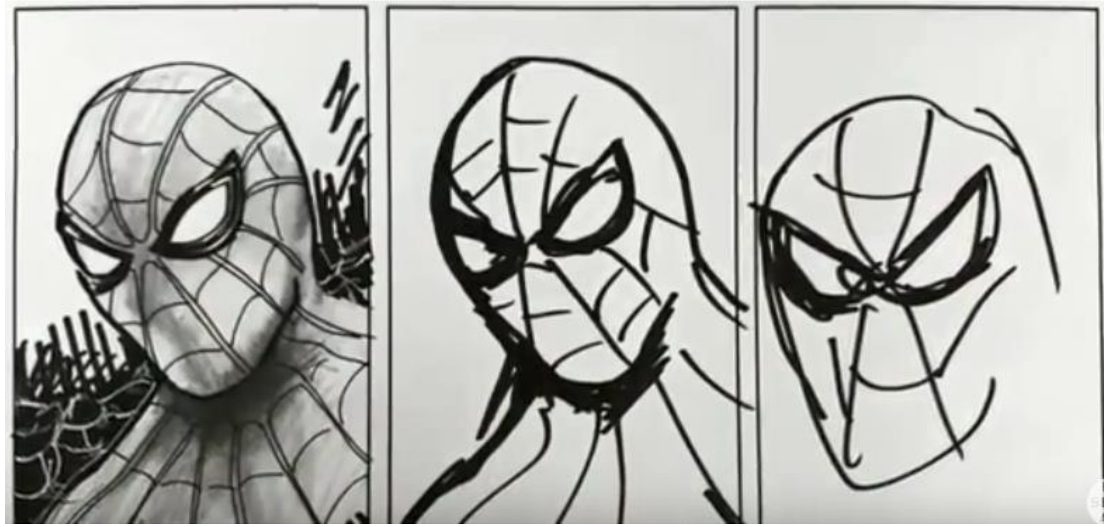
Data source 9 Annual State of Agile Survey di VersionOne (2015)

Agenda

- What is the Agile Methodology
- The Agile manifesto and its principles
- **Business Analysis Agile**
- Scrum
  - Scrum Roles
  - Scrum Events
  - Scrum Artifacts
- Scaled Agile Framework

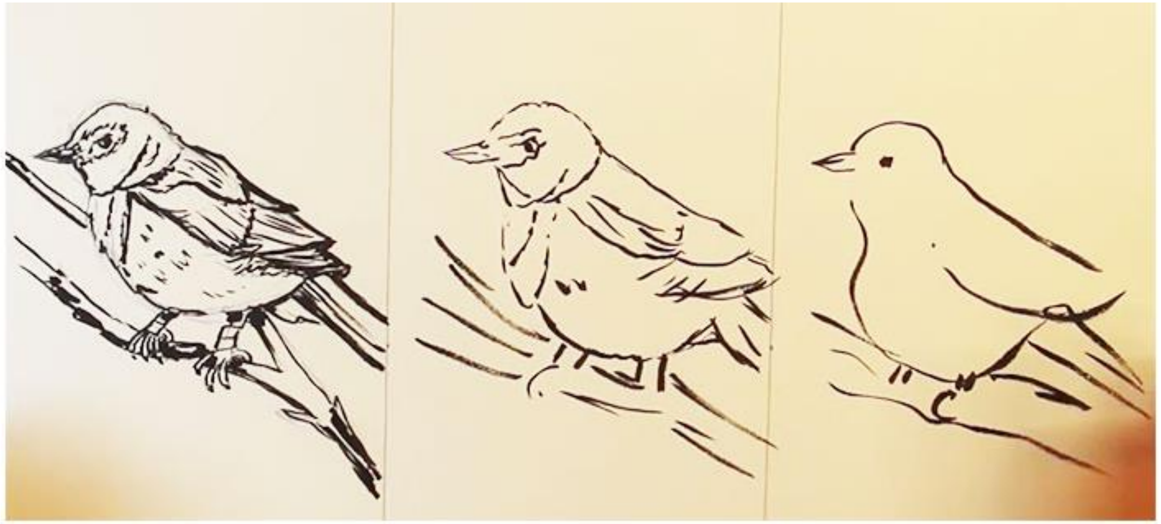
The Speed Drawing Example

How much time we need to develop values?



10 Minutes 1 Minute 10 Seconds

10 Minutes 1 Minute 10 Seconds



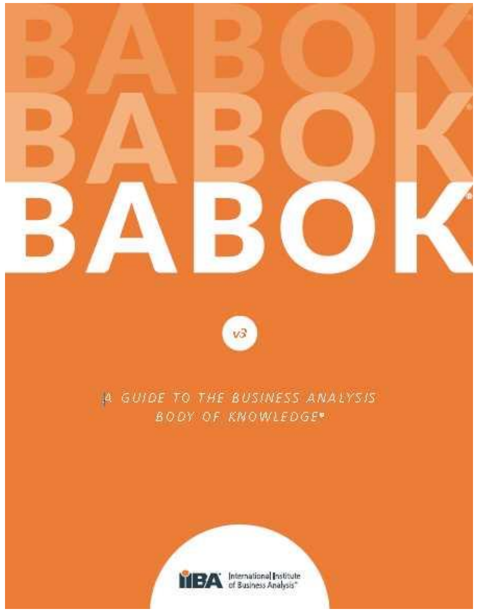
BABOK® Guide

A Guide to the Business Analysis Body of Knowledge (BABOK Guide)
The definitive community-based and consensus-driven Global Standard in business analysis.

BABOK® Guide expands the scope of business analysis, providing essential direction and support for practitioners in areas such as agile, business intelligence, information technology, business architecture and business process management.

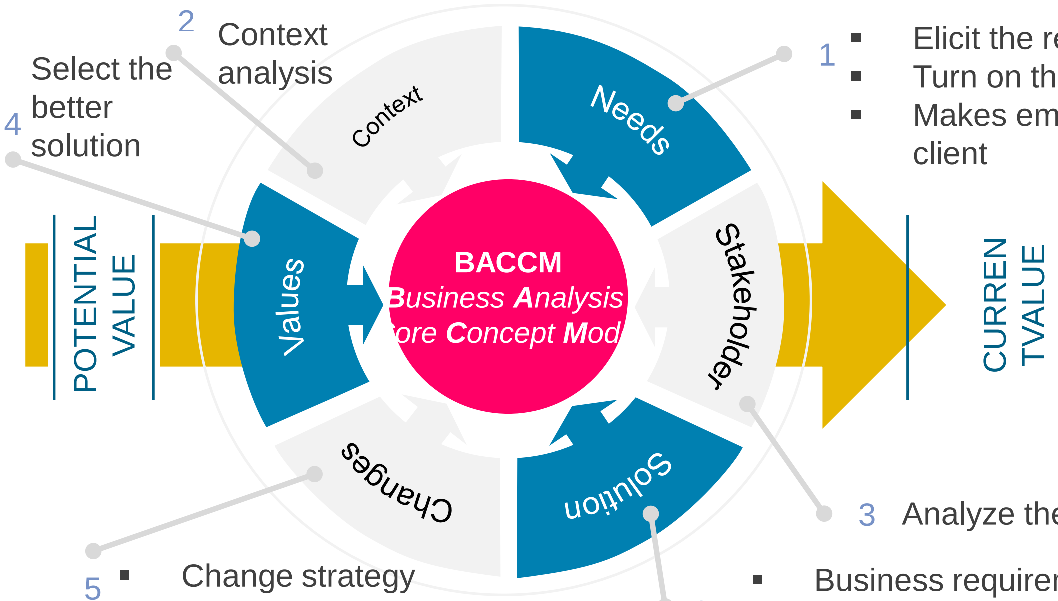
Developed with the contribution of 150 writers and researchers from 20 countries, *A Guide to the Business Analysis Body of Knowledge®* (*BABOK Guide®*) was reviewed by over 200 business analysis experts as well as 60 global thought leaders and more than 5500 insights and comments were received from global business analysis communities making *BABOK® Guide*, **the essential reference for better business outcomes since 2005.**

https://www.youtube.com/watch?v=tX2hjd-37pw#action=share


BABOK
BABOK
BABOK
v3
A GUIDE TO THE BUSINESS ANALYSIS BODY OF KNOWLEDGE®
IIBA International Institute of Business Analysis™

The Conceptual Framework

The Business Analysis Core Concept Model™ is a **CONCEPTUAL FRAMEWORK** for the business analysis and includes what is the business analysis e what it includes.



2 Context analysis

4 Select the better solution

1
- Elicit the real needs
- Turn on the idea
- Makes empathy with the client

3 Analyze the Stakeholders

5
- Change strategy
- Risk eveluation
- Perimeter of the solution

6
- Business requirements
- Requirements for stakeholders
- Solution requirements
- Evaluate alternative solutions

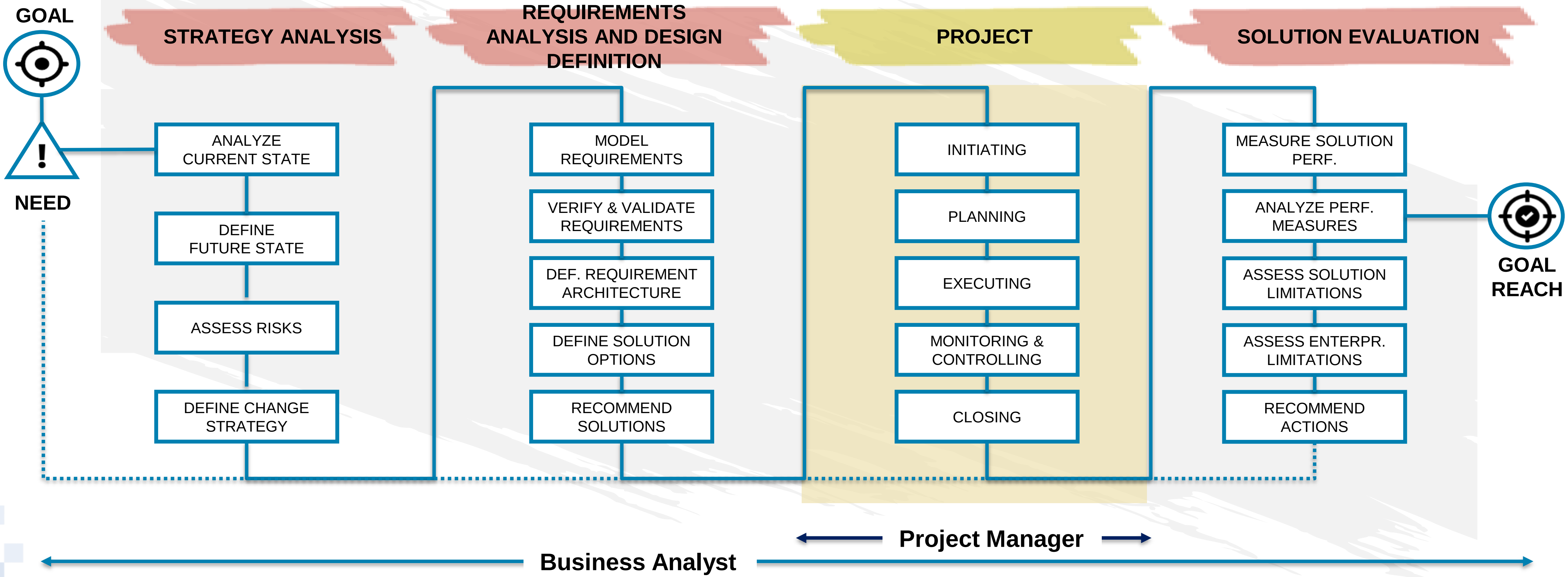
WHAT THE BUSINESS ANALYSIS DOES:

ADDRESS **CHANGES** THROUGH A GOOD UNDERSTANDING OF THE **NEEDS,** IDENTIFYING THE SOLUTION THAT MAXIMIZES **VALUE,** ANALYZING THE **CONTEXT** AND WHAT **STAKEHOLDERS** REALLY WANT.

Source BABOK® Guide – Version 3.0 – NTT Data elaboration

Business Analysis

- BUSINESS ANALYSIS PLANNING & MONITORING
- ELICITATION & COLLABORATION
- REQUIREMENTS LIFE CYCLE MANAGEMENT



Business Analysis

Business Analysis

What is

- The practice of **ENABLING CHANGE** in a company defines the requirements by recommending solutions that provide **VALUE** to the stakeholders

Why we use it

- It is used to understand the current state, to define the future state and to determine the activities required for moving from the **CURRENT** to the **FUTURE**

Where we use it

- It is applicable on a **VARIETY of INITIATIVES** within a company, in strategic, tactical or operational terms
- It can be used inside the **LIMITS** of the **PROJECT**
- It can be performed in different optics: agile, business intelligence, information technology, corporate architecture and business process management

How we use it

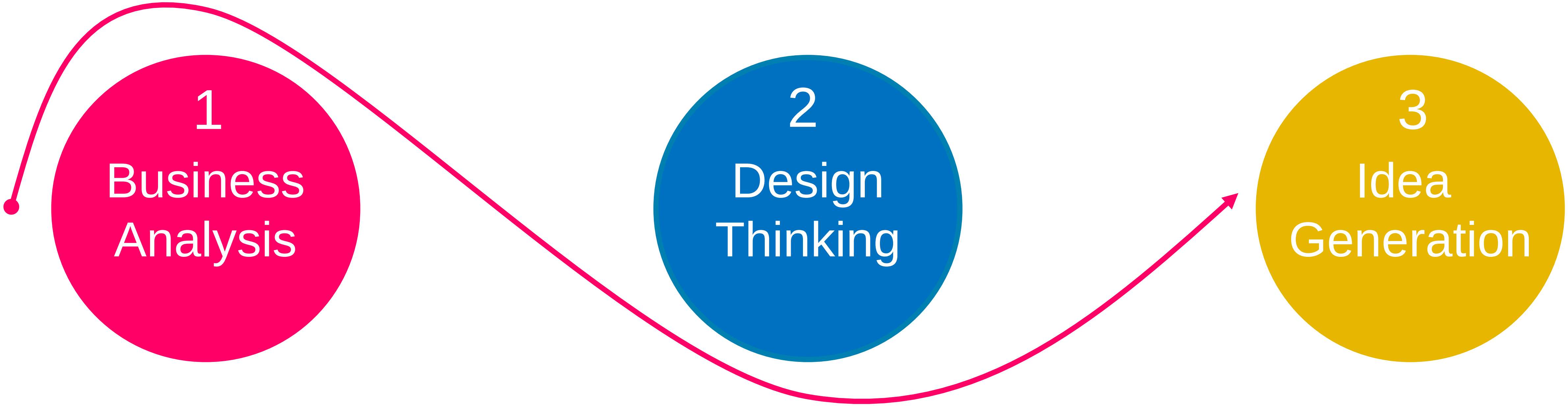
- Through the application by professionals of the BA of the ACCEPTED COMMON PRACTICES, competence, knowledge, terminology and attitudes

Who uses it

- Any **PERSON** who is **RESPONSIBLE** for discovering, synthesizing and analyzing information from a variety of sources in order to **DRIVE CHANGE** and **FACILITATE COLLABORATION OF INTERESTED PARTIES**

Source BABOK® Guide – Version 3.0 – NTT Data elaboration

From The Business Analysis To Generation Of An Idea



1
Business Analysis

2
Design Thinking

3
Idea Generation

It uses general approach

Explore the problem

Create your idea

Agile BA + Lean

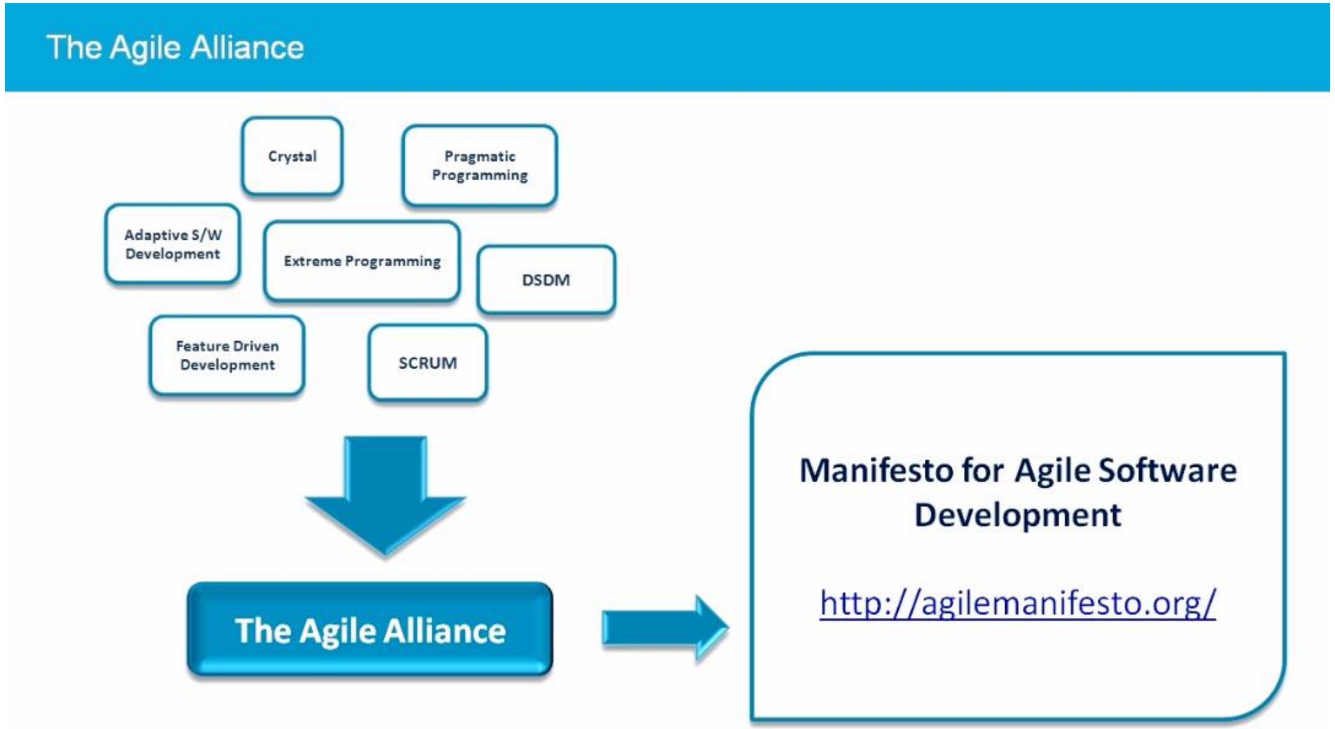
Visual and iterative techniques

Agenda

- What is the Agile Methodology
- The Agile manifesto and its principles
- Business Analysis Agile
- **Scrum**
  - Scrum Roles
  - Scrum Events
  - Scrum Artifacts
- Scaled Agile Framework

THE AGILE ALLIANCE

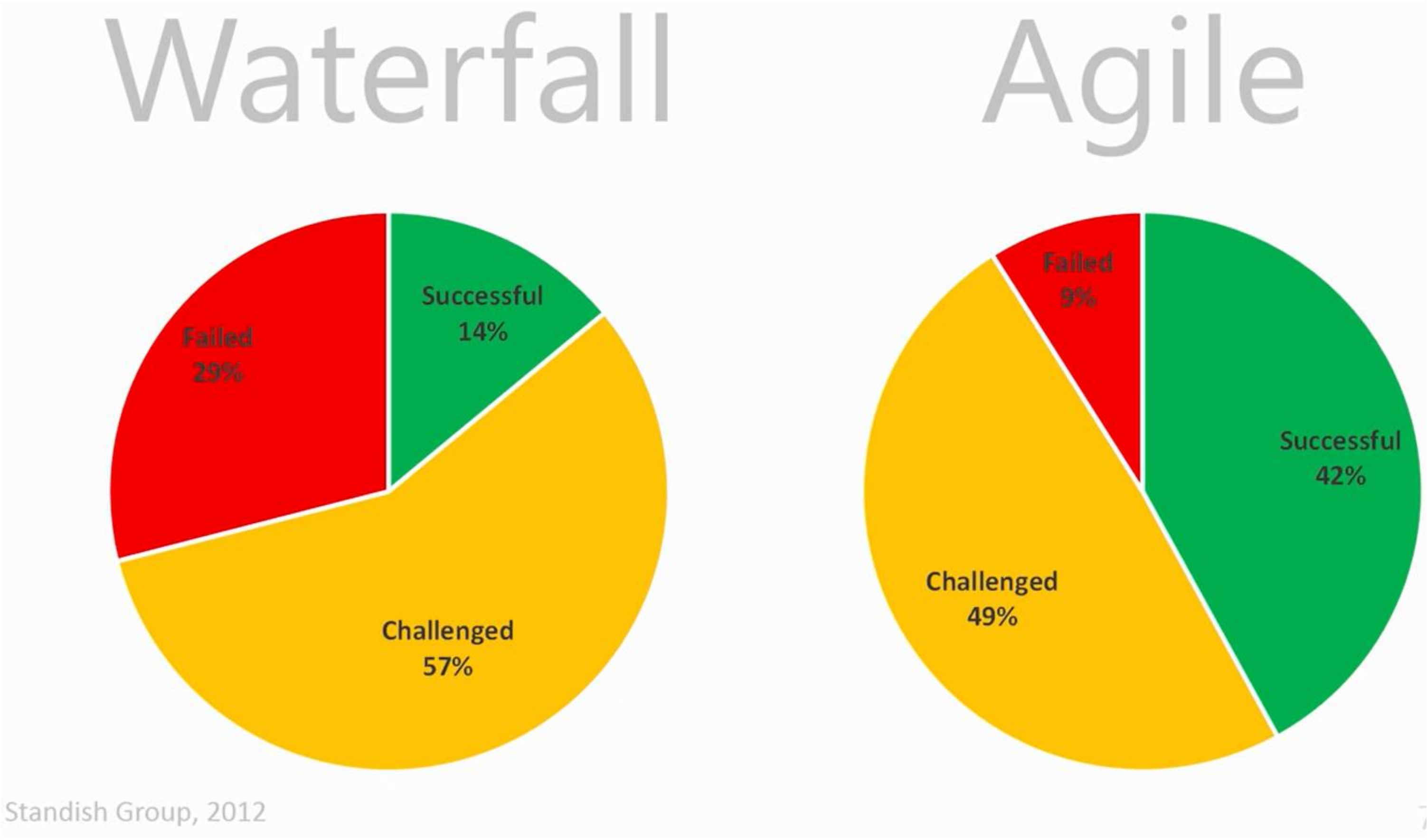
- In the **MID 1990s NEW SOFTWARE METHODOLOGIES** evolved as **ALTERNATIVES** to the heavyweight **WATERFALL** oriented processes. These new software methodologies advocated **PROCESSES** that were **MUCH LIGHTER WEIGHT**, resulting in a **MORE NIMBLE TEAM** which could **RESPOND QUICKLY** to **CHANGING REQUIREMENTS** in **DYNAMIC DEVELOPMENT ENVIRONMENTS**
- The term **AGILE** was used to describe how this class of software processes worked. The Agile movement was **NOT ANTIMETHODOLOGY**, instead it strove to **RESTORE BALANCE BETWEEN REQUIRED PROCESS** and the **NEED** to **GET THE WORK DONE** as **EFFICIENTLY** as possible
- In **FEBRUARY OF 2001**, leaders of these Agile software methodologies met in Utah to find common ground. This group named themselves the "**AGILE ALLIANCE**" and agreed on a set of common values, ideas, and themes that define the Agile software development approach



Source: «The Agile Manifesto», «Software Practices (SCRUM): SCRUM Roles», Skillsoft Course, 2016

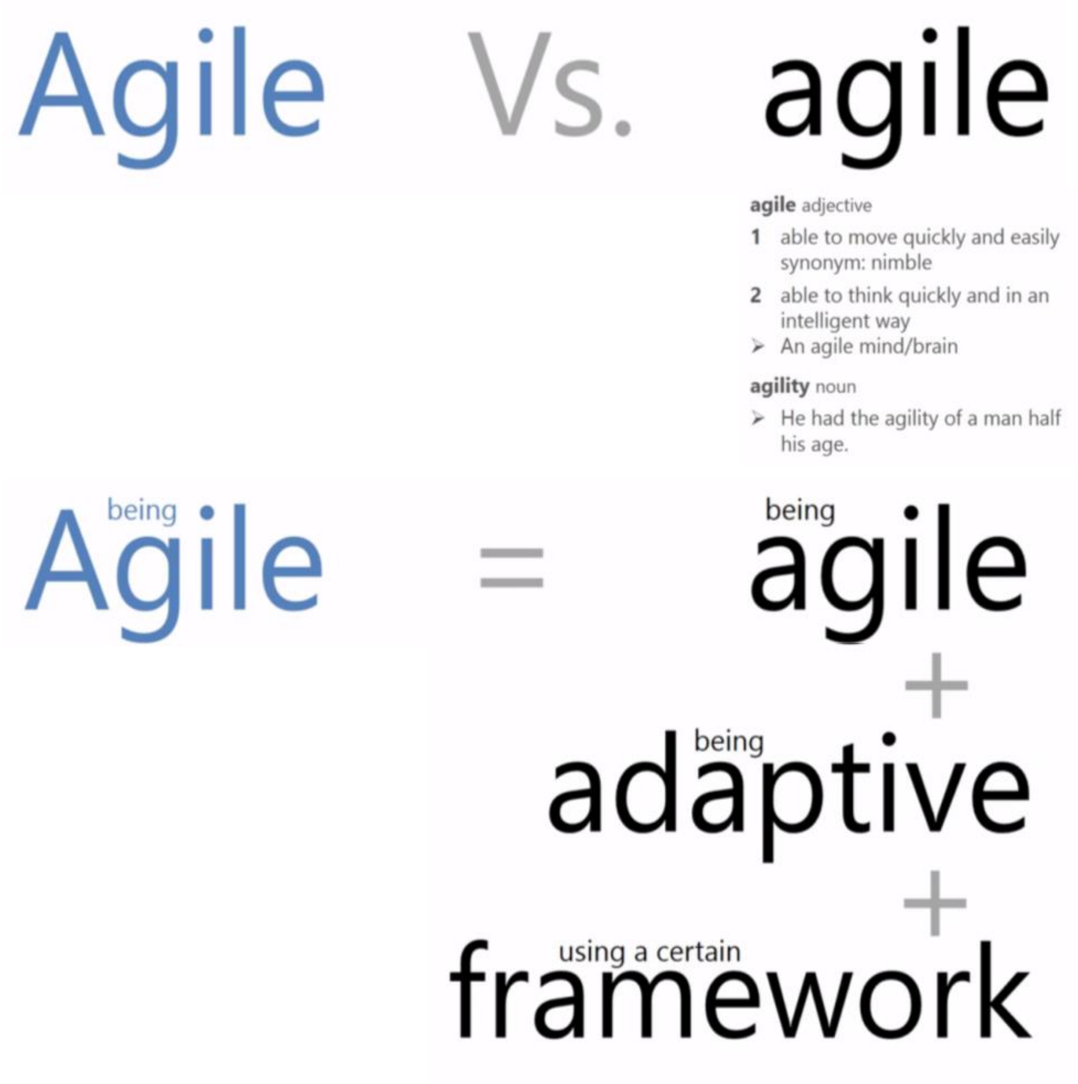
Agile success

the amount of Success in Agile environments is three times more than that of traditional projects



Standish Group, 2012

Using Agile terminology doesn't make you Agile…



Agile Vs. agile

agile adjective

1 able to move quickly and easily synonym: nimble

2 able to think quickly and in an intelligent way

➢ An agile mind/brain

agility noun

➢ He had the agility of a man half his age.

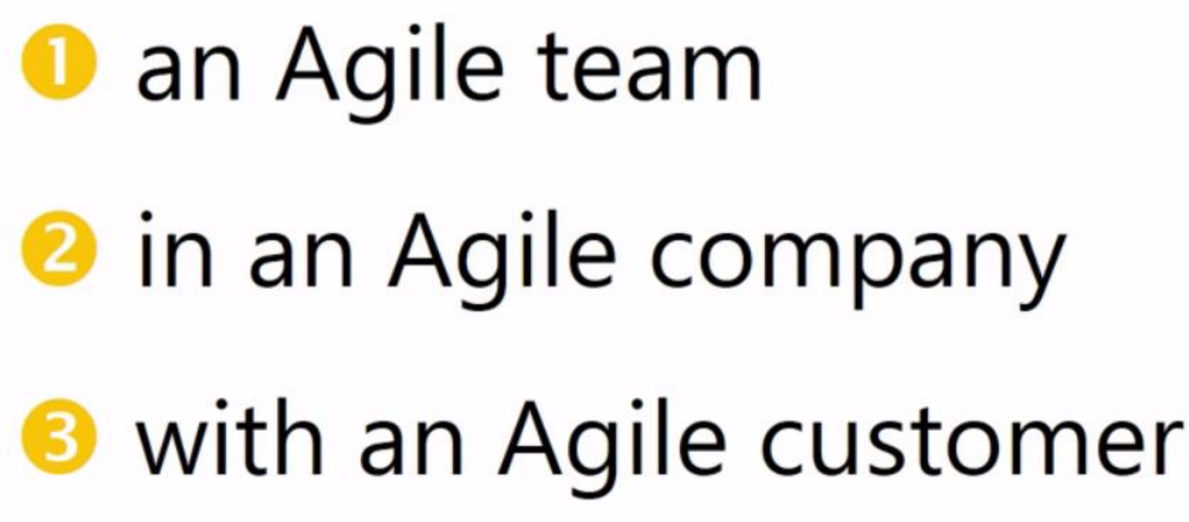
being Agile = being agile + being adaptive + using a certain framework

- Being **ADAPTIVE** is in the **CORE** of **EVERY AGILE FRAMEWORK**
- Use of an Agile Framework gives lots of **OPPORTUNITIES** but only if you **FOLLOW** and **ACCEPT** the **FRAMEWORK**

Agile Environment

Agile needs some requirements for its implementation

Agile needs



1. an Agile team
2. in an Agile company
3. with an Agile customer

…people able to **WORK** in an **AGILE FRAMEWORK**

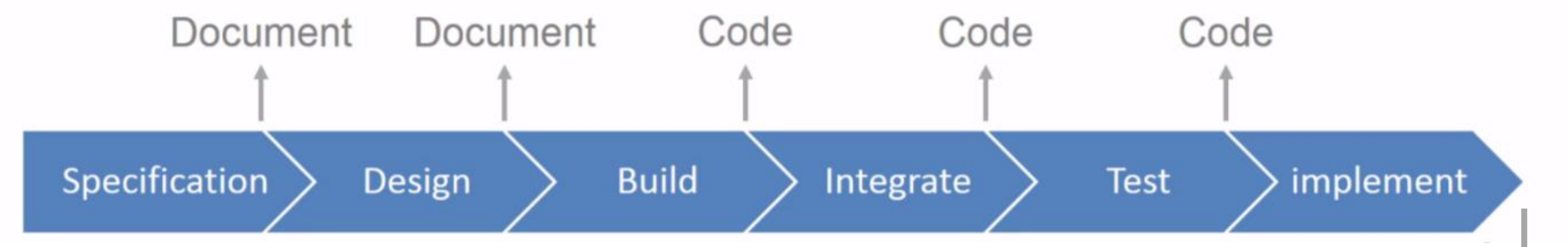
…which **ACCEPTS** the **CONSEQUENCES** of an Agile Frameworks (e.g. to empower developers (*))

… able to **COLLABORATE** and **UNDERSTAND** the **DIFFERENCES** of an Agile project

*(*) The word “**DEVELOPER**” refers to **ANYONE** who’s **INVOLVED** in the **PRODUCTION** of the **PROJECT OUTPUT**, including analysts, architects, programmers, testers, and UI designers*

When do we need agile…

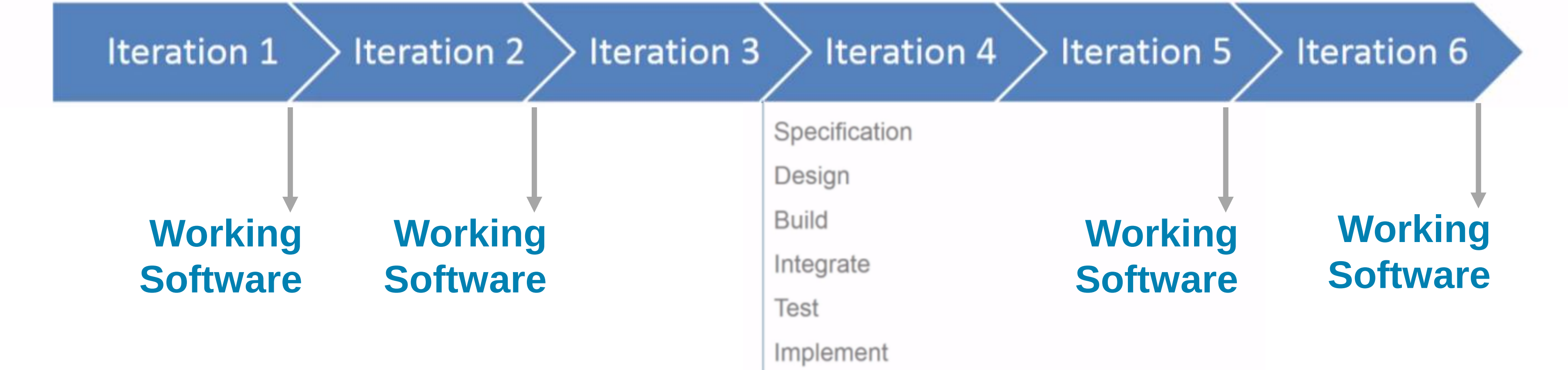
PREDICTIVE LIFECYCLES (traditional) run the **DEVELOPMENT PROCESSES ONE AFTER ANOTHER** (usually only once) and the **PRODUCT** is **CREATED** only at the **END** of the **PROCESS**



…but the customer doesn't know what they want, until they see the working software

Working Software

ADAPTIVE LIFECYCLES (Agile) have **MULTIPLE TIME-BOXED ITERATIONS** and **REPEAT** the **DEVELOPMENT PROCESSES** for each of them, creating **PIECES** of the **PRODUCT** at **EACH ITERATION**



Working Software

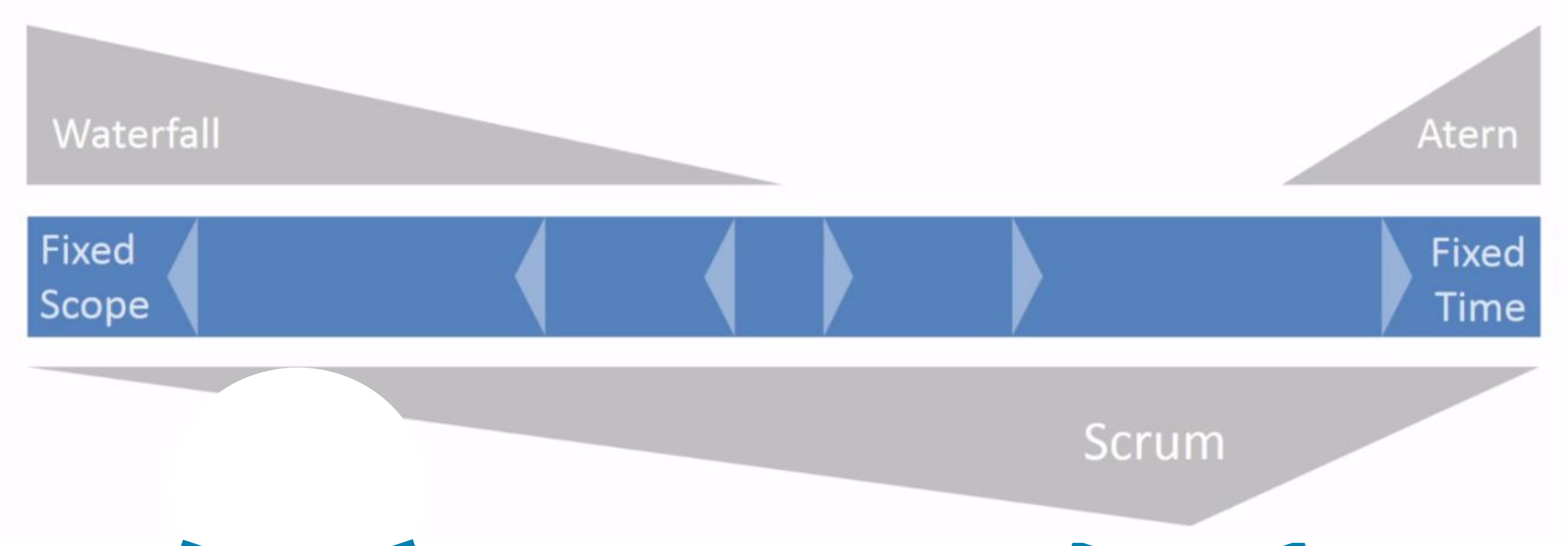
Working Software

Working Software

Working Software

…customer will be able to see the product, understand what they want and give feedbacks, to "adapt" the product in the next iteration

The development spectrum



NO ONE of the **AGILE APPROACHES** use the **FIXED SCOPE**

SCRUM is **IN** the **MIDDLE** between Fixed Scope and Time approaches but **CLOSER TO FIXED TIME** approach

Source: «The 45% Disaster», «Agile and Scrum Course», The Management Plaza, 2016

Scrum’s History

Since its first publication in 1995 until now, Scrum has been adopted by a large number of software development companies. Today, it is recognized as the most applied framework for agile development software

1995

- Jeff Sutherland and Ken Schwaber **CODIFIED** Scrum

1996

- Introduction of Scrum at **OOPSLA CONFERENCE** and published a paper “Scrum Software Development Process”

2001

- Jeff and Ken were amongst the 17 software development leaders creating the **MANIFESTO FOR AGILE SOFTWARE DEVELOPMENT**. Shortly after publication “**AGILE SOFTWARE DEVELOPMENT WITH SCRUM**” by K. Schwaber & M. Beedle

2002

- Ken Schwaber founded the **SCRUM ALLIANCE** with Mike Cohn and Esther Derby

2006

- J. Sutherland created his own **COMPANY**, Scrum.inc

2009

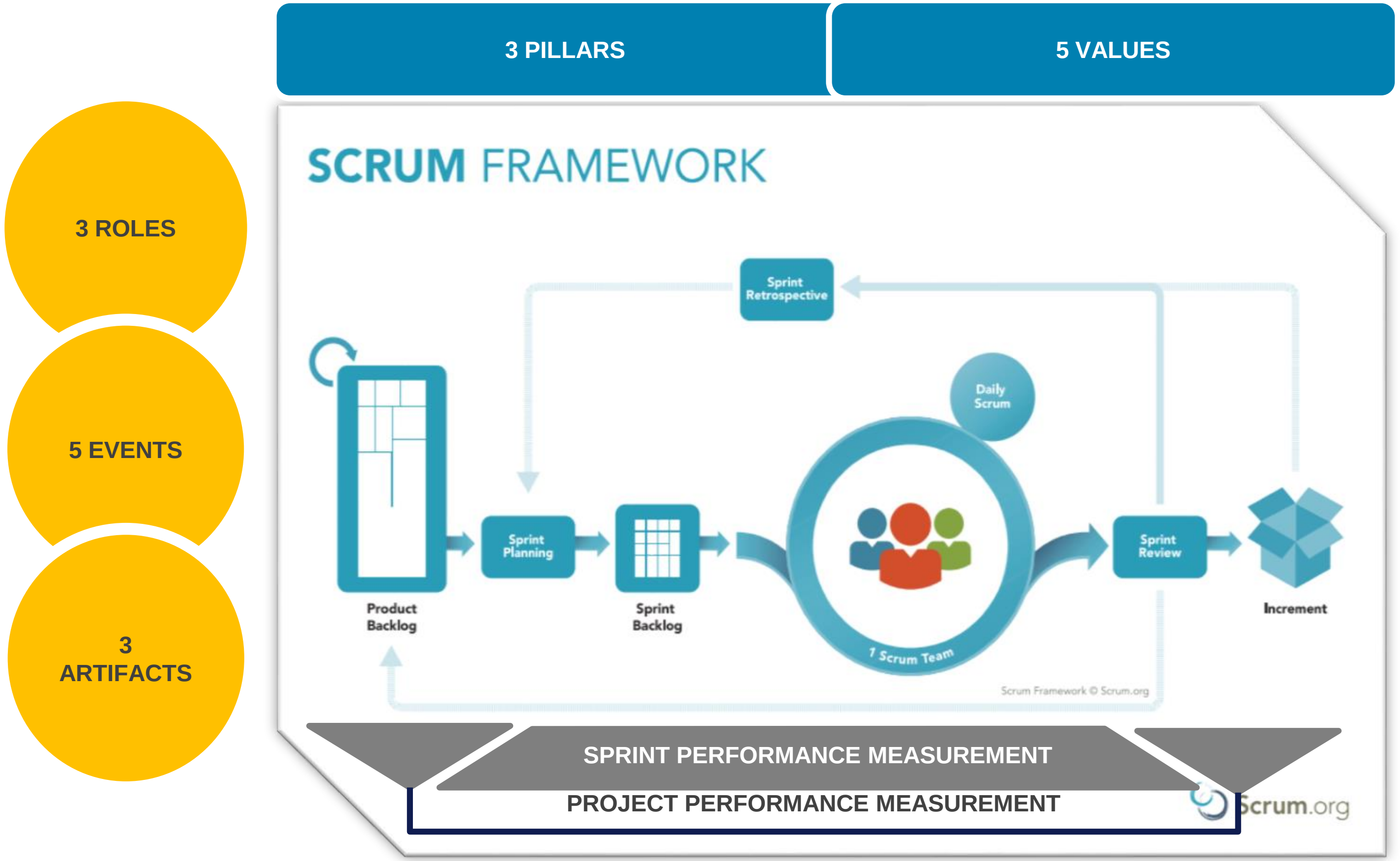
- K. Schwaber founded the **SCRUM.ORG**

2010

- Publication of the Scrum Guide in 2010 (updates in 2017)

Introduction to Scrum

SCRUM is an **AGILE FRAMEWORK**, not a methodology or body of knowledge, for **DEVELOPING** and sustaining **COMPLEX ADAPTIVE PRODUCTS** at the **HIGHEST VALUE**. Scrum is more about project delivery rather than project management. The Scrum Framework should be used entirely (we do not tailor it) and consists of **ROLES**, **EVENTS**, **ARTIFACTS**, and the **RULES** that bind them together. Scum is **LIGHTWEIGHT**, **SIMPLE TO UNDERSTAND**, **DIFFICULT TO MASTER**



3 PILLARS

5 VALUES

3 ROLES

5 EVENTS

3 ARTIFACTS

SPRINT PERFORMANCE MEASUREMENT

PROJECT PERFORMANCE MEASUREMENT

Source: «*The Scrum Guide™*», *Scrum.org, July 2016*

Guidelines

Adopting SCRUM involves an effort by all stakeholders in terms of::

- Acquisition of the new operational approach
- Acquisition of new knowledge and skills

SCRUM still follows some basic rules that are easy to understand.

For this reason, in order not to weigh down the various stakeholders with the study of formal rules, the idea is to define an operative model inspired by the Agile but lightened by some components.

SCRUM follows these basic rules(Pillars):

1. **TRASPARENCY**
2. **INSPECTION**
3. **ADAPTATION**

3 Scrum Pillars

TRANSPERENCY to share a vision and create visibility

- Aspects of the process must be **visible** to those responsible for the outcome
- Transparency requires those aspects be defined by a **common standard** so observers share a common understanding of what is being seen
  - A **common language** referring to the process must be shared by all participants
  - Those performing the work and those accepting the work product must share a common definition of “**DONE**”

INSPECTION to react rapidly

Scrum users must frequently inspect artifacts and progress toward a Sprint Goal to detect undesirable variances. Their inspection should not be so frequent that inspection gets in the way of the work

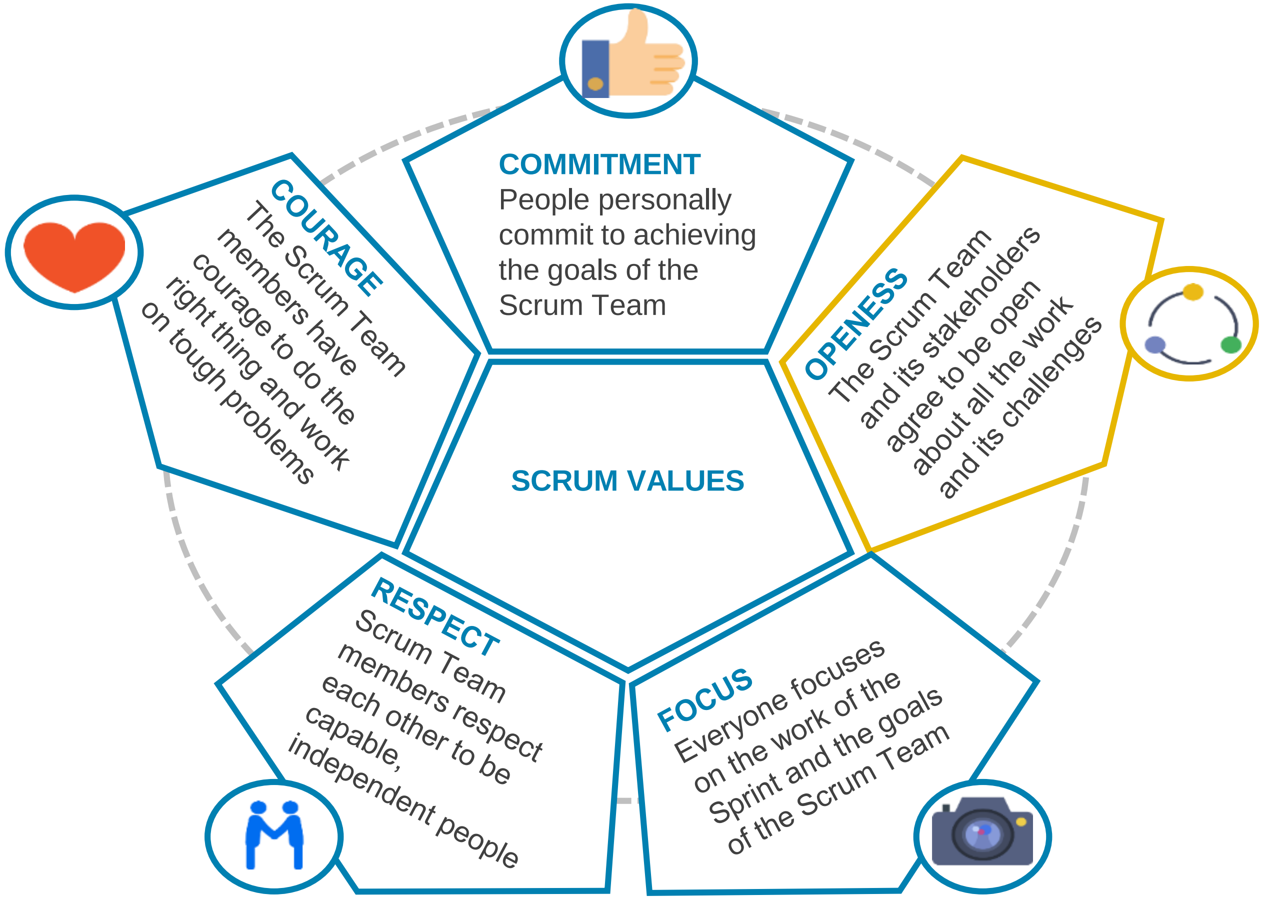
ADAPTATION to respond accurately to the needs

If the inspector determines that some aspects of the process deviate outside acceptable limits, and that the resulting product will be unacceptable, the process or the material being processed must be adjusted to minimize deviation

Four events for I&A: Sprint planning, Daily Scrum, Sprint Review, Sprint Retrospective

Scrum Values

When the **VALUES** of **COMMITMENT**, **COURAGE**, **FOCUS**, **OPENNESS** and **RESPECT** are **EMBODIED** and lived by the **SCRUM TEAM**, the **SCRUM PILLARS** of transparency, inspection, and adaptation **COME TO LIFE** and build trust for everyone. Successful use of Scrum depends on people becoming more proficient in living these five values



SCRUM VALUES
COMMITMENT
People personally commit to achieving the goals of the Scrum Team
OPENESS
The Scrum Team and its stakeholders agree to be open about all the work and its challenges
FOCUS
Everyone focuses on the work of the Sprint and the goals of the Scrum Team
RESPECT
Scrum Team members respect each other to be capable, independent people
COURAGE
The Scrum Team members have courage to do the right thing and work on tough problems

Visibility, Adaptibility, More Business Value, Less Risks

SCRUM is founded on **EMPIRICAL PROCESS CONTROL THEORY**, or empiricism. Empiricism asserts that knowledge comes from experience *and* making decisions based on what is known. Scrum employs an **ITERATIVE**, incremental **APPROACH** to **OPTIMIZE PREDICTABILITY** and **CONTROL RISK**. **THREE PILLARS** uphold every implementation of empirical process control: **TRANSPARENCY**, **INSPECTION**, and **ADAPTATION.**

VISIBILITY’

Visibilty is constant during project’ life cycle, thanks to feedback, transparency, review, retrospectives and frequent releases

ADAPTIBILITY

Frequent Feedback and iterative cycles facilitate adaptability and rapid response to changes, opposed to traditional approaches, really reactive only at the beginning

BUSINESS VALUE

Collaboration between development team and customer, iterative ad incremental releases of Business Value to the market, fasten ROI achievement

RISKS

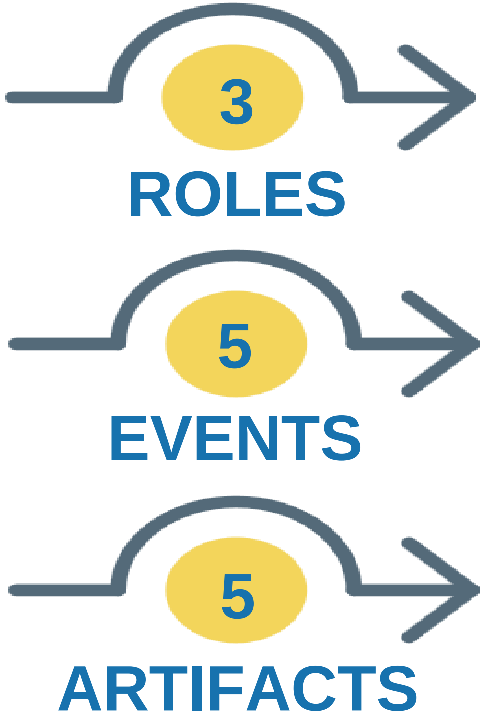
Active involvement of stakeholders, visibility increase**, transparency, quality** and **project monitoring sensibly mitigate risks’ potential impacts**

Scrum

SCRUM is an **AGILE FRAMEWORK**, not a methodology or body of knowledge, for developing and sustaining **COMPLEX ADAPTIVE PRODUCTS** at the **HIGHEST VALUE**. Scrum is more about project delivery rather than project management. The Scrum Framework should be used entirely (we do not tailor it) and consists of **ROLES**, **EVENTS**, **ARTIFACTS**, and the **RULES** that bind them together. Scum is **LIGHTWEIGHT**, **SIMPLE TO UNDERSTAND**, **DIFFICULT TO MASTER.**

3 PILLARS

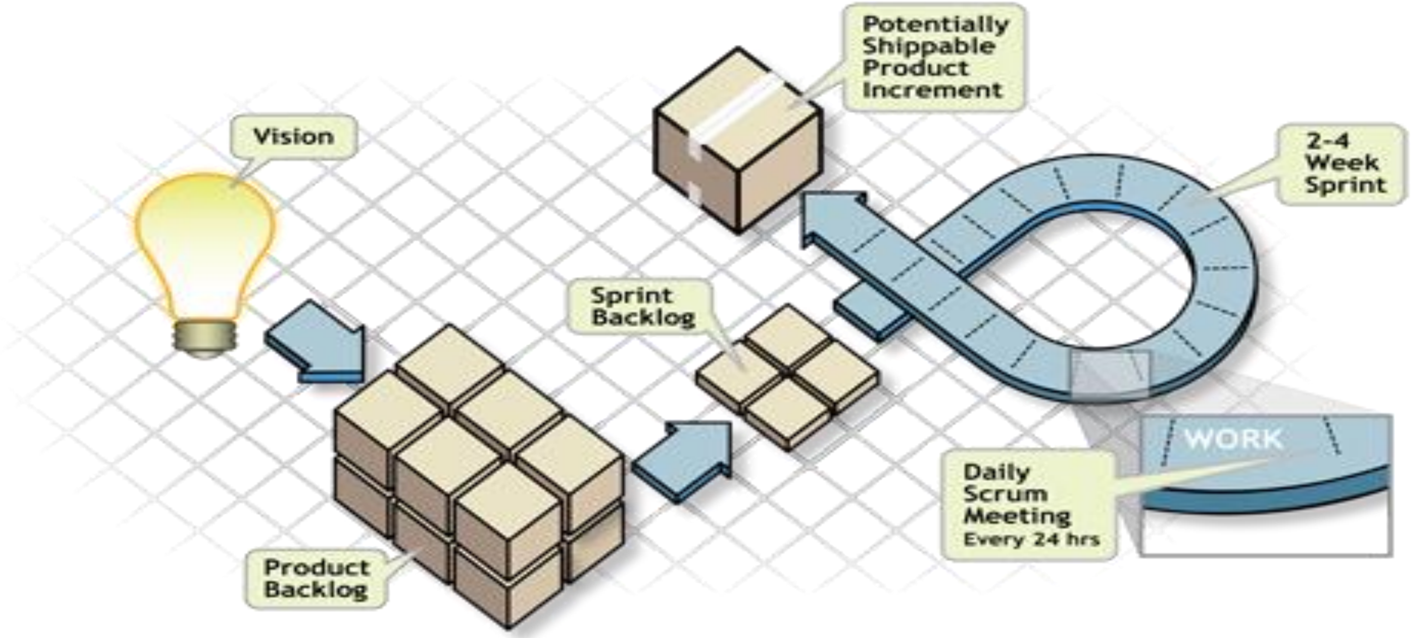
5 VALUE



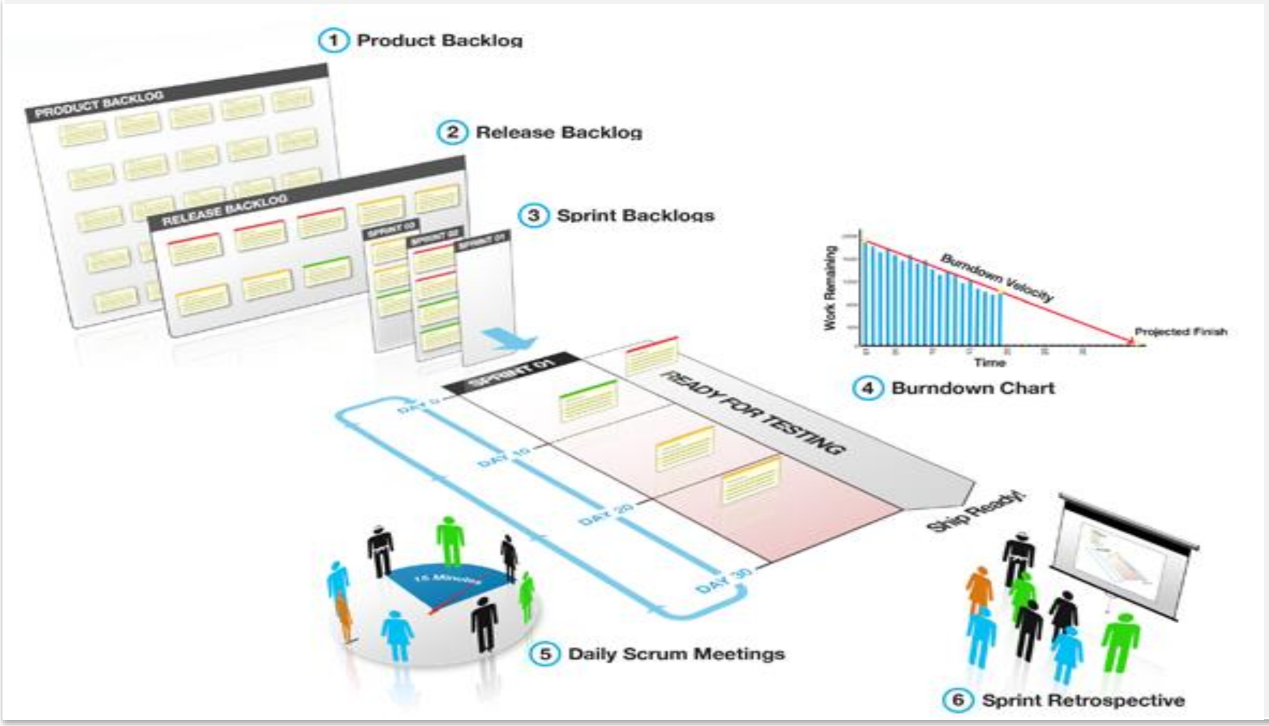
3 ROLES

5 EVENTS

5 ARTIFACTS


Potentially Shippable Product Increment
Vision
2-4 Week Sprint
Sprint Backlog
WORK
Daily Scrum Meeting Every 24 hrs
Product Backlog

The Model



«Scrum makes clear the effectiveness of your management and development practices used to improve them.»

Jeff Sutherland

1 **PRODUCT BACKLOG**: It is an ordered list of the macro requirements shared with the **PRODUCT OWNER**.

2 **RELEASE BACKLOG**: it is a list of all requirements inserted in the Product Backlog, and it represents the functionality implemented on the release (optional).

3 **SPRINT BACKLOG**: It is the set of Product Backlog items selected for the Sprint, plus a plan for delivering the product increment and realizing the Sprint Goal

4 **BURNDOWN CHART**: it is daily updated and it contains the activities tracking .

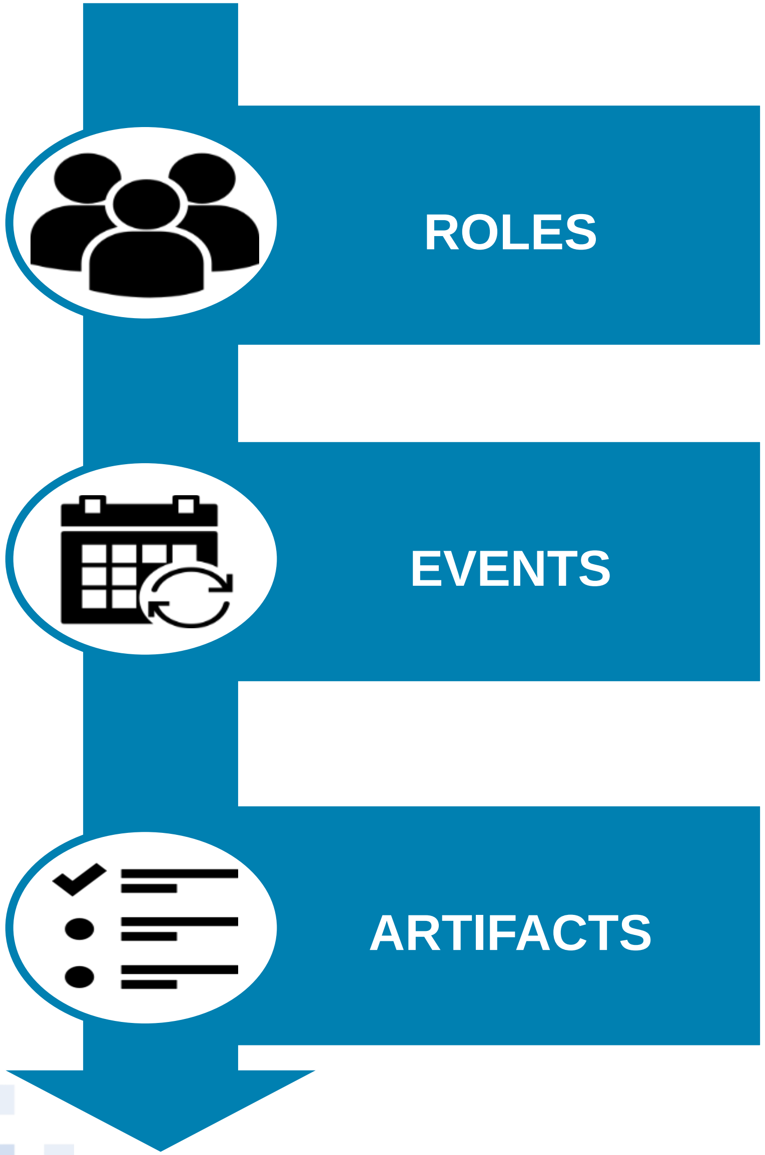
5 **DAILY SCRUM**: It's a 15-minute event for Development Team. This optimizes team collaboration and performance by inspecting the work since the last Daily Scrum and forecasting upcoming Sprint work.

6 **SPRINT RETROSPECTIVE**: It is an opportunity for the Scrum Team to inspect itself and create a plan for improvements to be enacted during the next Sprint.

Agenda

- What is the Agile Methodology
- The Agile manifesto and its principles
- Business Analysis Agile
- **Scrum**
  - **Scrum Roles**
  - Scrum Events
  - Scrum Artifacts
- Scaled Agile Framework

Scrum Framework



ROLES

- Product Owner
- Scrum Master
- Team

EVENTS

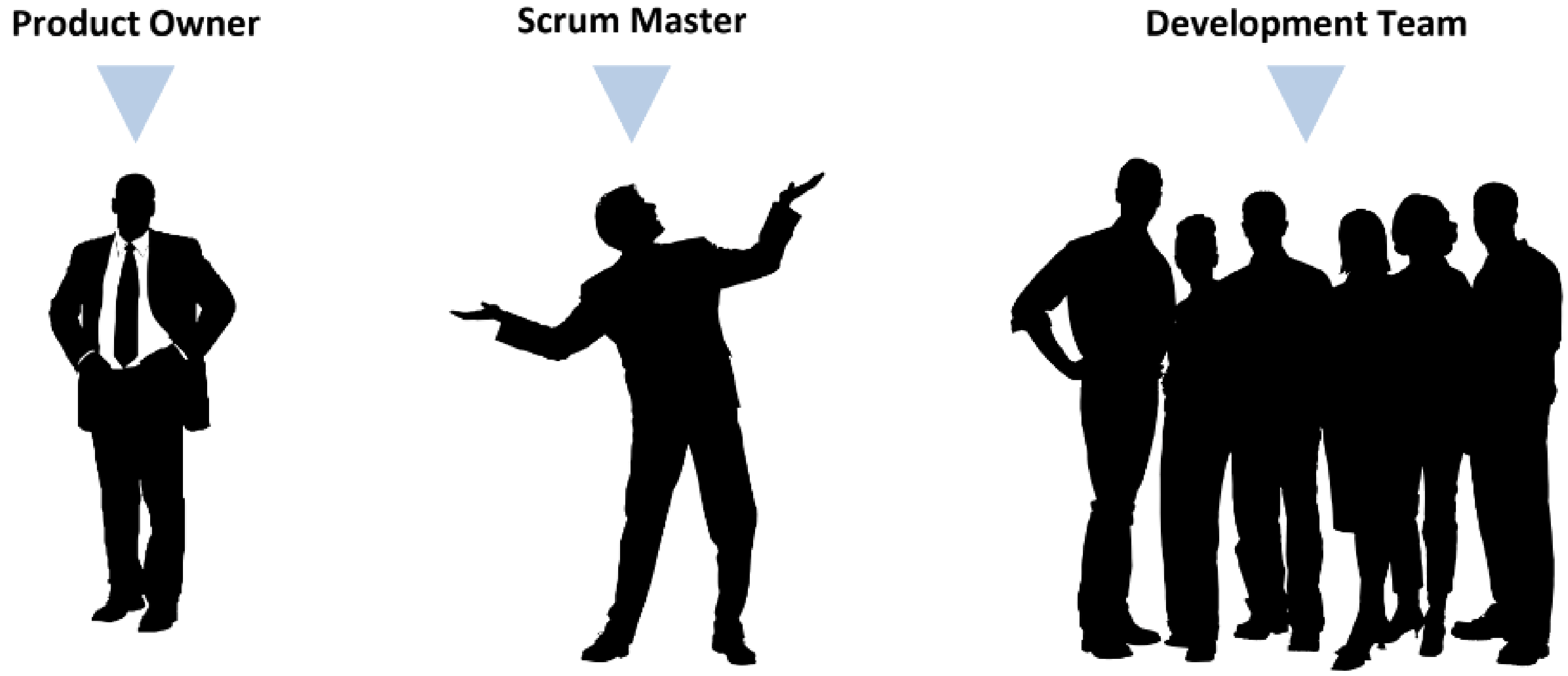
- Sprint planning
- Sprint review
- Sprint retrospective
- Daily stand up meeting

ARTIFACTS

- Product backlog
- Sprint backlog
- Burndown charts

Scrum Team

Product Owner, Scrum Master e Development Team



Product Owner

1 person
Full-time or part-time
Business oriented

Scrum Master

1 person
Full-time or part-time
Scrum coach and facilitator

Development Team

3 to 9 people
Full-time (recommended)
Specialist

The term "Scrum Team" is referred to all the members of the project.

The Scrum Team consists of a Product Owner, the Development Team, and a Scrum Master.

Scrum Teams are self-organizing and cross-functional

Self Organizing & Cross Functional

Product Owner, Scrum Master e Development Team

THEY ARE COHESIVE

Agile teams put emphasis on collaboration. They are constantly learning from each other and taking advantage of their shared skills to work together in harmony. Furthermore, their working relationships are clearly established so that they know that they can work towards a common goal.

THEY CAN FUNCTION WITHOUT OUTSIDE HELP

In general, an agile team has sufficient skills within the team so that it is not necessary to outsource help from an external group. Agile members have the advantage of being versatile enough to specialize in one thing but also to dabble in other fields. This not only helps save time, but also allows members to expand their experience

THEY ARE STABLE

With Agile Methodology. all members are trained to collaborate and deliver results in a timely manner.

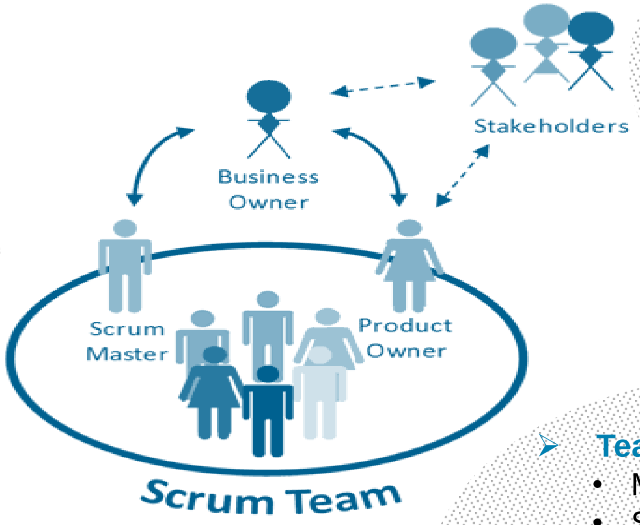
The Agile method emphasizes the **COLLABORATIVE EFFORT BETWEEN TEAM MEMBERS** and is therefore a great way to ensure that they deliver timely and relevant results. **THE ROLES AND RELATIONSHIP BETWEEN THE MEMBERS ARE ALSO WELL DEFINED** so there is no confusion about who does what.

Roles – Tasks And Characteristics

➢ **Scrum Master**

- Supervising of the project
- Removing the impediments
- Protecting the team

- Team building and Coaching
- Empathy



➢ **Product Owner**

- Representing the client
- Managing of the product backlog
- Defining ther elease planning

- Leadership
- Communication

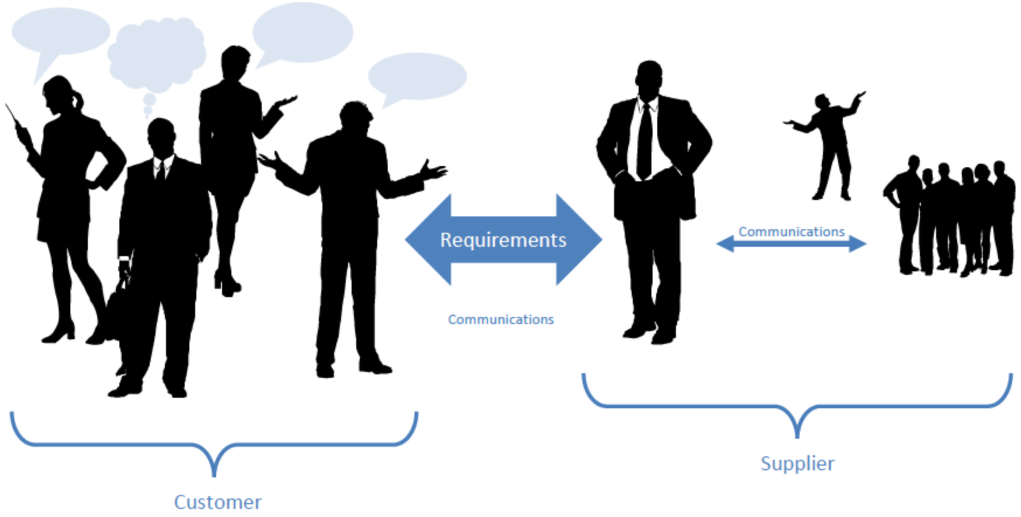
➢ **Team**

- Making estimates
- Selecting the job
- Performing the developments

- Proactivity
- Curiosity

The scrum team: roles & responsibilities

- The Product Owner is the **KEY STAKEHOLDER** and **HOLDS** the **VISION** of the **FINAL PRODUCT**, from the perspective of both the user's needs and the business needs, making decisions to achieve a good **RETURN ON INVESTMENT**
- The **MAIN RESPONSIBILITY** of the Product Owner is **MAXIMIZING** the **VALUE** of the **PRODUCT** and the **WORK** of the **DEVELOPMENT TEAM**
- Product Owner is the **SOLE PERSON RESPONSIBLE** for **MANAGING** the **PRODUCT BACKLOG**, by him/herself or having the Development Team do it, remaining **ACCOUNTABLE:**
  - ✓ **CREATING** and **ORDERING** the **ITEMS** based on the **BUSINESS VALUE** *(the way of measuring the business value is all up to the Product Owner)*;
  - ✓ making **ITEMS VISIBLE, CLEAR** and **UNDERSTOOD**
- The Product Owner must make the **HARD CHOICES** on the **PROJECT**, constantly **BALANCING COMPETING INTERESTS**
- Product Owner acts as a **CONDUIT** between the **EXTERNAL STAKEHOLDERS** and the **SCRUM TEAM**



The scrum team: roles & responsibilities

- Product Owner should be **RESPECTED BY** the **WHOLE ORGANIZATION:** no one can override Product Owner’s decisions
- Product Owner is responsible of **MEASURING** the **PROJECT PERFORMANCE** and **REPORTING** to the **STAKEHOLDERS**
- Product Owner is **ONE PERSON**, not a committee. The Product Owner may represent the desires of a committee, but those wanting to change a Product Backlog must address the Product Owner.

When making the transition from traditional waterfall to agile, job description often dictate who fills the PO role. This often leads to the **WRONG** person being selected for the **PRODUCT OWNER** role. This is a common mistake that must be avoided

- When selecting the Product Owner, he/she should be the **“REAL” SUBJECT MATTER EXPERT** in the product domain, able to communicate with both the technical and business audiences, and respected by the Scrum Team and stakeholders regarding their decisions

Sometimes the **PRODUCT OWNER** is **NOT AVAILABLE** for many reasons. The most common are: too busy, don't see the value of Scrum, don't understand the role of the product owner. A PO’s lack of participation contributes to the Product Backlog viewed as “incomplete” and not fully communicated

- The first thing is to educate on the importance of the product owner to ensure their engagement in the project. In the absence of the product owner, oftentimes what occurs is the identification of a **PROXY** to serve in the **full capacity** in terms of **decision making**, **power** and **authority**. When this occurs, the responsibilities of the product owner must transfer to the proxy in their entirety

The scrum team: roles & responsibilities

- The **MAIN RESPONSIBILITY** of the Scrum Master is **ENSURING SCRUM** is **UNDERSTOOD** and **ENACTED**. Scrum Masters do this by **TRAINING** people, **COACHING** and **ENSURING** that the **SCRUM TEAM ADHERES** to **SCRUM THEORY**, **PRACTICES**, and **RULES**.
- The Scrum Master is considered a **MANAGEMENT ROLE**: the Scrum Master does not manage people, but, **MANAGES THE PROCESS**
- The Scrum Master is a **SERVANT-LEADER** for the Scrum Team, **REMOVING IMPEDIMENTS** to the Development Team's progress and **FACILITATES SCRUM EVENTS AS REQUESTED** or **NEEDED**
- The Scrum Master **HELPS THOSE OUTSIDE** the Scrum Team **UNDERSTAND** which of their **INTERACTIONS** are **HELPFUL** to maximize the created value
- The Scrum Master **HELPS ORGANIZATIONS** in its **SCRUM ADOPTION**, planning Scrum implementations and helping employees and stakeholders understand and enact Scrum and empirical product development;

The scrum team: roles & responsibilities

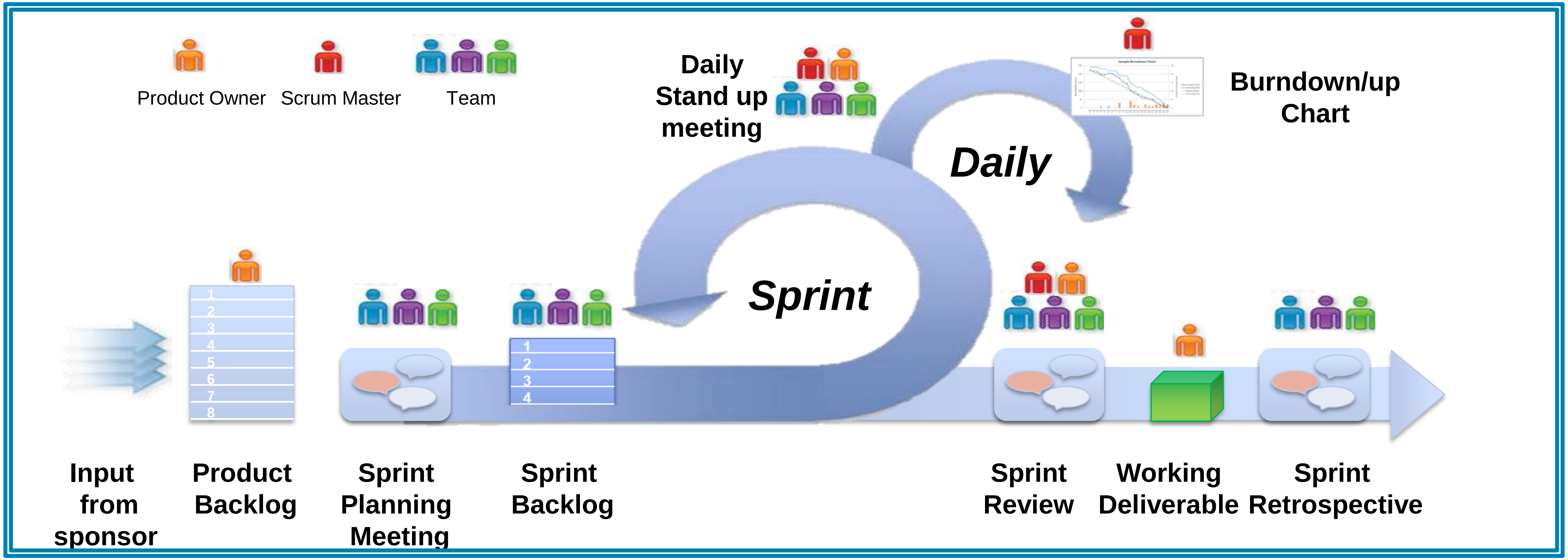
- The Development Team consists of **PROFESSIONALS** who do the work of **DELIVERING** a **POTENTIALLY RELEASABLE INCREMENT** of **“DONE” PRODUCT** at the end of each Sprint. Only members of the Development Team create the Increment
- The Development Team is **SELF-ORGANIZING**. No one tells the Development Team how to turn Product Backlog into Increments of potentially releasable functionality;
- The Development Team is **CROSS-FUNCTIONAL**. **INDIVIDUAL MEMBERS** may have **SPECIALIZED SKILLS** and areas of focus, but **ACCOUNTABILITY** belongs to the Development Team as a whole.
- The Development Team has **NO TITLES** other than “**DEVELOPER**”. Scrum recognizes **NO SUB-TEAMS** in the Development Team, regardless of particular domains (e.g. testing)
- The Development Team is responsible of **MEASURING** the **SPRINT PERFORMANCE**

The Development Team is **SMALL ENOUGH** to remain nimble and **LARGE ENOUGH** to complete significant work within a Sprint. **FEWER THAN THREE MEMBERS** decrease interaction, results in **SMALLER PRODUCTIVITY** gains and may encounter **SKILL CONSTRAINTS** during the Sprint. **MORE THAN NINE MEMBERS** requires too much coordination and generate **TOO MUCH COMPLEXITY** for an empirical process to manage.

Adaptive Planning

After completing the initial activities of creating the product backlog, the activities become iterative through an adaptive planning.

During any sprint the team takes on a new set of activities called Sprint Backlog, part of the entire Product Backlog.



Agenda

- What is the Agile Methodology
- The Agile manifesto and its principles
- Business Analysis Agile
- **Scrum**
  - Scrum Roles
  - **Scrum Events**
  - Scrum Artifacts
- Scaled Agile Framework

The sprint

The **heart of Scrum** is a Sprint, a time-box of one month or less during which a “Done”, useable, and potentially releasable product Increment is created. Sprints have consistent durations throughout a development effort. A new Sprint starts immediately after the conclusion of the previous Sprint.
Sprints contain and consist of the:

- Sprint Planning,
- Daily Scrums,
- the development work,
- the Sprint Review,
- and the Sprint Retrospective.

During the Sprint:

- No changes are made that would endanger the Sprint Goal;
- Quality goals do not decrease; and,
- Scope may be clarified and re-negotiated between the Product Owner and Development Team as more is learned.

The sprint

Each Sprint may be considered a project with no more than a one-month horizon. Like projects, Sprints are used to accomplish something. Each Sprint has a goal of what is to be built, a design and flexible plan that will guide building it, the work, and the resultant product increment.

Sprints are limited to one calendar month. When a Sprint’s horizon is too long the definition of what is being built may change, complexity may rise, and risk may increase. Sprints enable predictability by ensuring inspection and adaptation of progress toward a Sprint Goal at least every calendar month. Sprints also limit risk to one calendar month of cost.

Cancelling a sprint

A Sprint can be cancelled before the Sprint time-box is over. Only the Product Owner has the authority to cancel the Sprint, although he or she may do so under influence from the stakeholders, the Development Team, or the Scrum Master.

A Sprint would be cancelled if the Sprint Goal becomes obsolete. This might occur if the company changes direction or if market or technology conditions change. In general, a Sprint should be cancelled if it no longer makes sense given the circumstances. But, due to the short duration of Sprints, cancellation rarely makes sense.

When a Sprint is cancelled, any completed and “Done” Product Backlog items are reviewed. If part of the work is potentially releasable, the Product Owner typically accepts it. All incomplete Product Backlog Items are re-estimated and put back on the Product Backlog. The work done on them depreciates quickly and must be frequently re-estimated.

Sprint cancellations consume resources, since everyone regroups in another Sprint Planning to start another Sprint. Sprint cancellations are often traumatic to the Scrum Team, and are very uncommon.

The Events 1 Of 3

SPRINT PLANNING MEETING

It is the start-up meeting of the iteration, divided into two parts of 2 hours each

Part One

- Product Owner describes ***user stories*** in order of priority
- Based on previous sprints, the development team selects the stories to be implemented by creating the **backlog sprint.**

Part Two

(For the development team only)

- The team **breaks down** the stories into technical tasks to which it assigns an estimate in hours.
- At the end the team populates the taskboard
- The Product Owner remains available for any clarifications.

User Story

As a connection

I want to buy a parking ticket

So that I can use my car to go to my office

→

- Code the middle tier (8 hours)
- Code the user interface (4)
- Write test case(4)
- Update performance tests (4)

Technical task with estimation

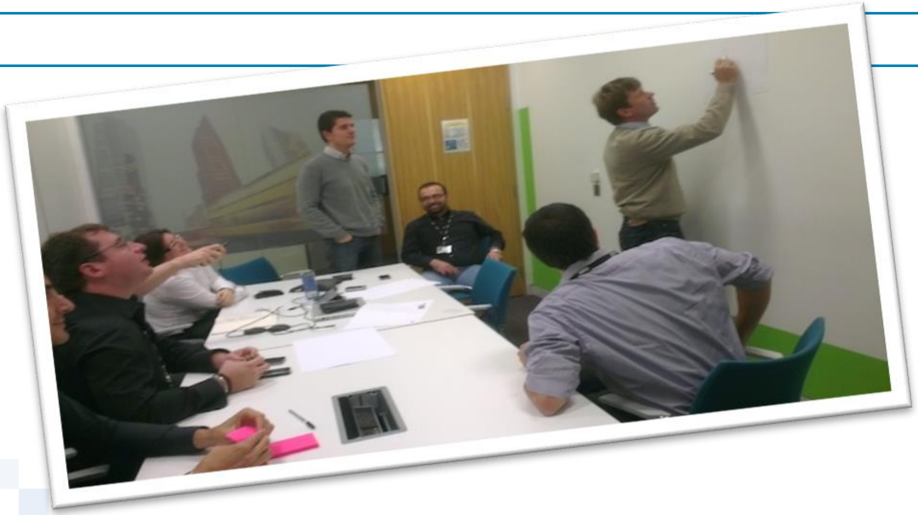
The Events 2 Of 3

SPRINT REVIEW MEETING

At the end of each iteration the Team shows to the Product Owner, to the Scrum Master and to all the stakeholders involved, in 4 hours, the activities developed. Generally, a demo of what is developed and feedback is collected.

At the end of the demo meeting sprint, data and metrics are shared (Burndown chart, Speed of development, possible bugs, ...) and the problems encountered.

SPRINT RETROSPECTIVE



At the end of the sprint review, the entire team meets for about 3 hours to analyze the progress of the sprint ended: what went well and what went wrong.

This event is an integral part of the "inspect and adapt" process and is necessary to improve team performance.

The Events 3 Of 3

DAILY STAND UP MEETING

The Daily Scrum is a 15-minute (maximum 20 minutes) time-boxed event for the Development Team. The Daily Scrum is held every day of the Sprint at the same time and place in order to reduce complexity.

Development Teams will use questions, some will be more discussion based. Here is an example of what might be used:

- What did I do yesterday that helped the Development Team meet the Sprint Goal?
- What will I do today to help the Development Team meet the Sprint Goal?
- Do I see any impediment that prevents me or the Development Team from meeting the Sprint Goal?

During the meeting the problems are not resolved but they are directed towards the scrum master.

The Stand Up Meeting helps to avoid any other unnecessary meeting!

Continuous Improvement

The process of continuous improvement as the base of the new methodological approach we should adopt, foresees the adoption of some basic rules:

- Any change to the planning must be agreed by the development team with the client to share elapsed, impacts, risks…
- The software development and release cycle must be as much as possible defined in a fixed time window (minimum one week) and must follow the correct procedure of: development, testing and production.
- Each new functional, technical or architectural requirement must be identified, described and classified within a tracking system before being taken over by the development team. The working group undertakes to use a formalism able to identify:
  - Type of requirement:
    - **F**unctional,
    - **A**rchitectural
    - **T**echnical
  - User Story
  - Punctual requirement associated with the User Story

  Each anomaly identified must be identified, described and classified within a tracking system before being taken over by the development team. The working group undertakes to use a formalism able to univocally identify the anomaly within the system.
- Each release must be identified by an X.Y.Z version and accompanied by the relevant release note.

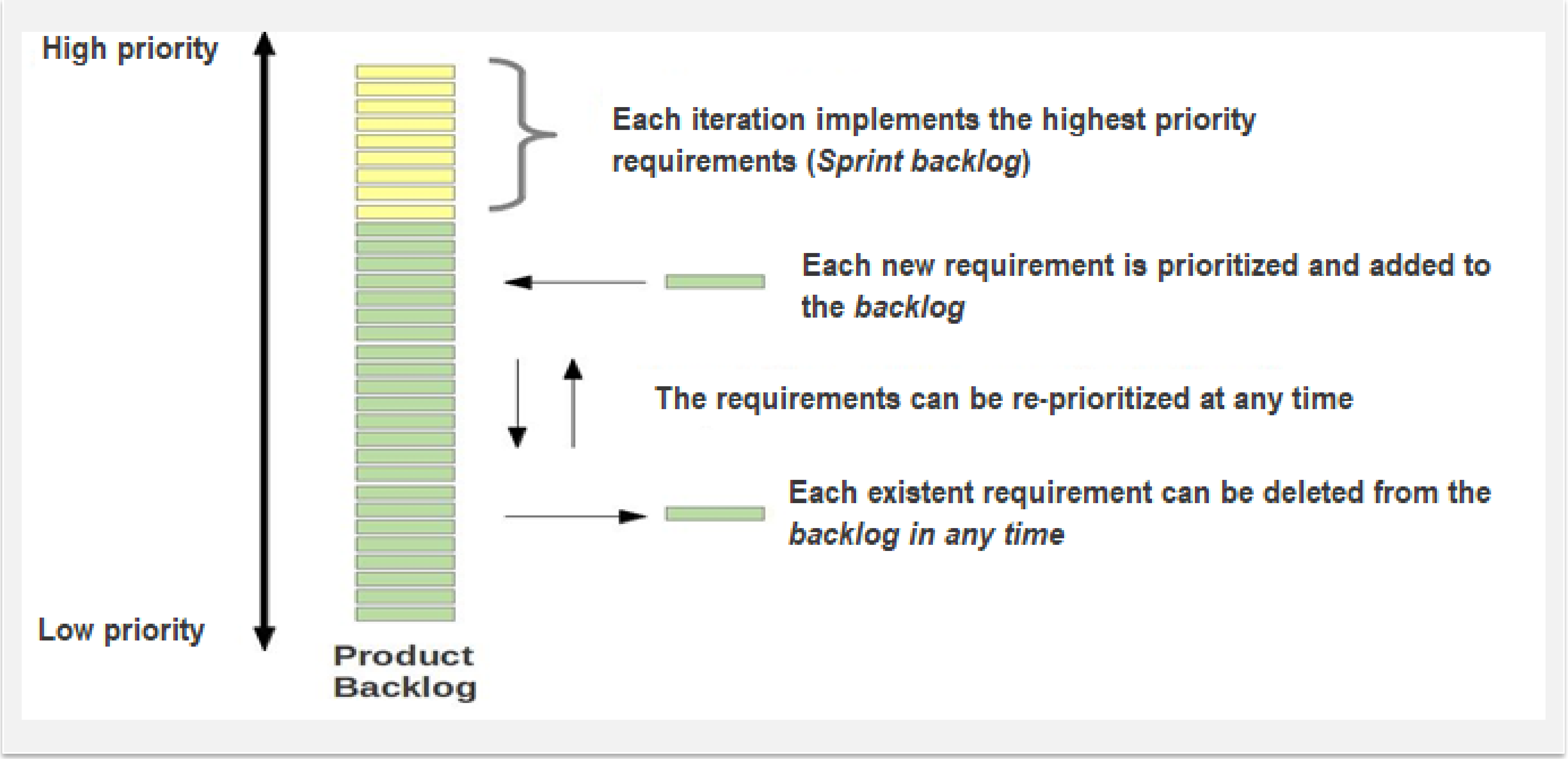
Agenda

- What is the Agile Methodology
- The Agile manifesto and its principles
- Business Analysis Agile
- **Scrum**
  - Scrum Roles
  - Scrum Events
  - **Scrum Artifacts**
- Scaled Agile Framework

Product Backlog - Value Driven

The criteria of priority is given by the "value" that the requirement, the action of contrast to the risk or the change, bring to the business.

THE SET OF PRIORITIES ORDERED IS THE PRODUCT BACKLOG.



High priority
Low priority
Product Backlog
Each iteration implements the highest priority requirements (*Sprint backlog*)
Each new requirement is prioritized and added to the *backlog*
The requirements can be re-prioritized at any time
Each existent requirement can be deleted from the *backlog in any time*

The Product Backlog Characteristics

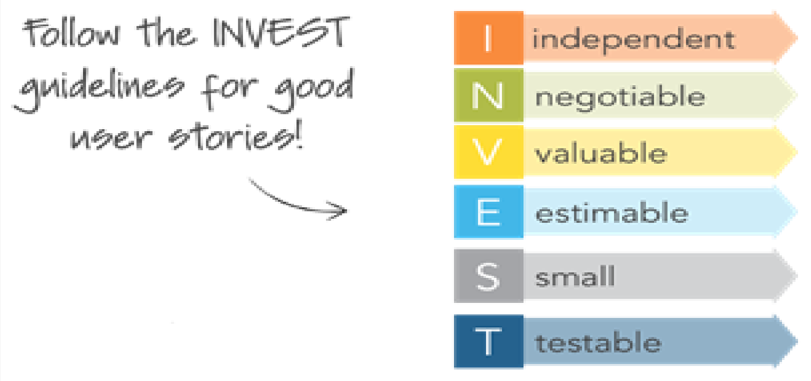
- List of project requirements
- List of all the work necessary to achieve the project objective
- Each element must be important for the customer
- Items are ordered by importance from the product owner
- The order is re-evaluated at each sprint

GOAL SETTING

Specific
Measurable
Achievable
Realistic
Timely

It is developed by the ***USER STORIES*** that **DESCRIBE THE FUNCTIONAL NEEDS**

- **As** a <*role*>,
- **I want** <*functionality*>,
- **so that** <*business benefit*>



Follow the INVEST guidelines for good user stories!

I independent
N negotiable
V valuable
E estimable
S small
T testable

User story are harvested in a hierchical way creating the ***backlog*** (e.g. Epic→ Feature → User story → Task)

Product Backlog – Prioritization And Estimation

After defining the User Stories, the client proceeds with their prioritization.

The a prioritization is essential because of the Client believes that everything is priority and urgent.

Different methods are available, including MoSCoW:

- **M**ust have
- **S**hould have
- **C**ould have
- **W**ant to have

After stories definition and their prioritization by the Client, the Team makes the estimation of the stories.

One of the most famous method used for this purpose, is the **FIBONACCI SERIES** (0, 1, 1, 2, 3, 5, 8, 13, …) made using the planning poker game.

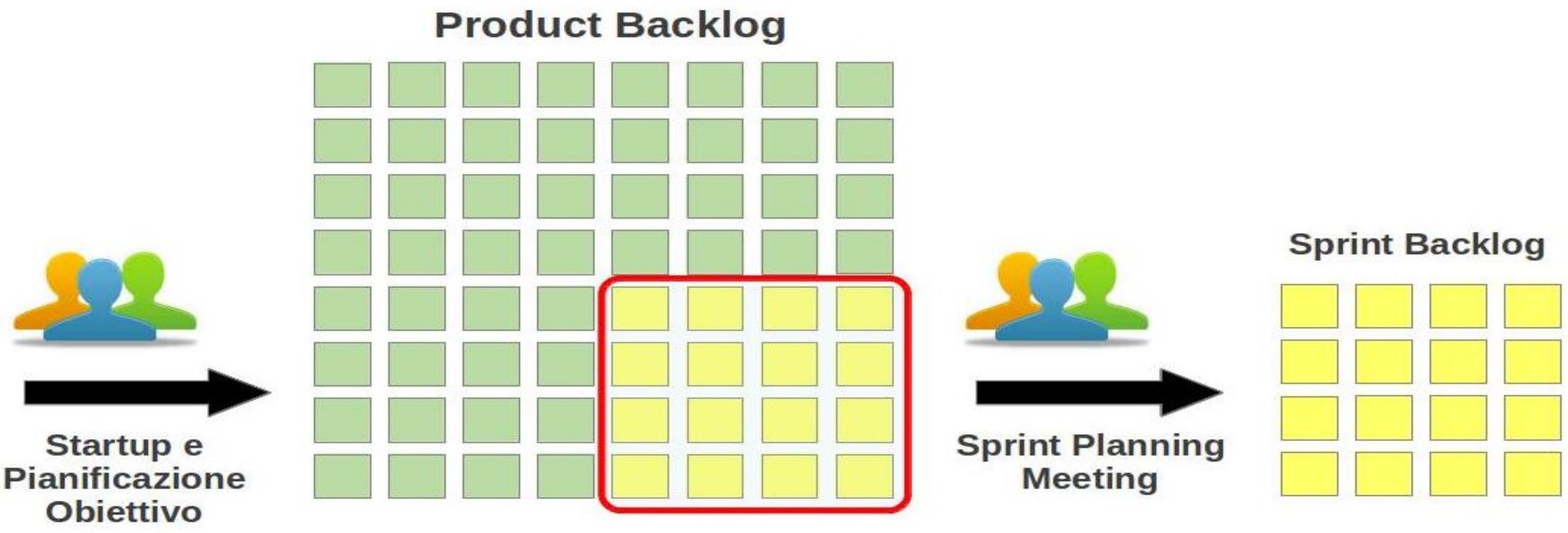
The estimate is performed by the team that has to agree on the evaluations in terms of:

- **ESTIMATION IN STORY POINTS**: the absolute value of the estimate that is said relative to the other stories is not important. It will be the speed in the sprints to key how many will be done;
- **ESTIMATION IN IDEAL DAYS:** value of the ideal day dedicated solely to the realization of the story, without the consideration the interruptions (e.g. the meetings).

From The Product Backlog To Sprint Backlog

For each iteration, there is a planning of the business requirements to implement, of the actions to contrast the risk to undertake and of the changes to implement (principle of "late decision" ...)

In order to define the tasks that we have to insert in the Sprint backlog, usually we create a **RELEASE BACKLOG**, subset of the requirements contained in the Product Backlog, which **REPRESENTS THE LIST OF THE FUNCTIONALITIES IMPLEMENTED IN THE RELEASE**.



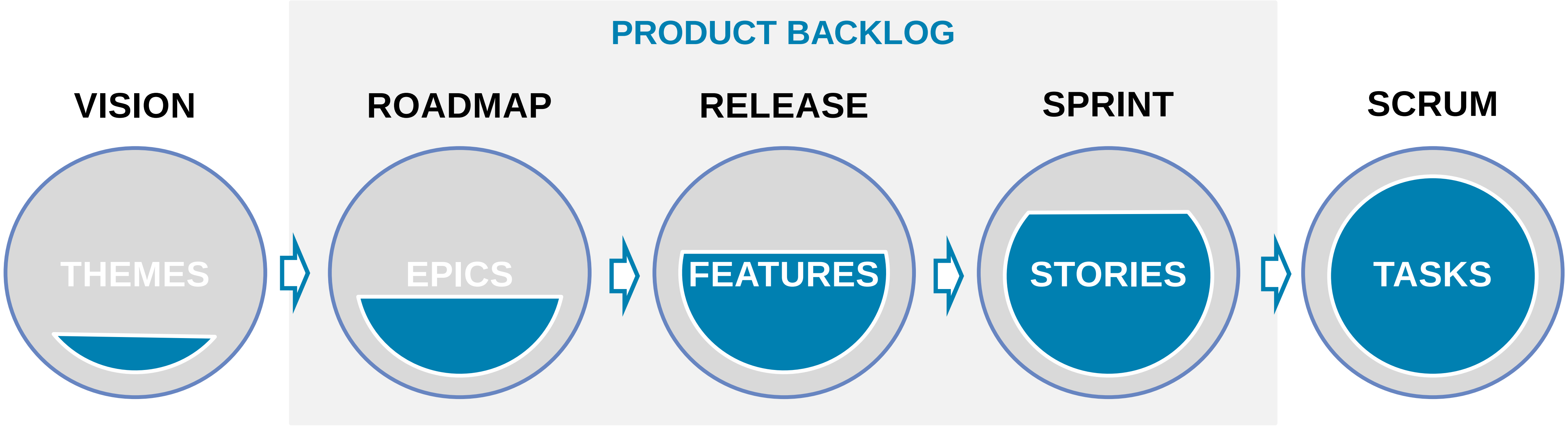
The Sprint Backlog

The Sprint Backlog is the set of Product Backlog items selected for the Sprint, plus a plan for delivering the product Increment and realizing the Sprint Goal. The Sprint Backlog is a forecast by the Development Team about what functionality will be in the next Increment and the work needed to deliver that functionality into a “Done” Increment. The Sprint Backlog makes visible all the work that the Development Team identifies as necessary to meet the Sprint Goal. To ensure continuous improvement, it includes at least one high priority process improvement identified in the previous Retrospective meeting.

The Sprint Backlog is a plan with enough detail that changes in progress can be understood in the Daily Scrum. The Development Team modifies the Sprint Backlog throughout the Sprint, and the Sprint Backlog emerges during the Sprint. This emergence occurs as the Development Team works through the plan and learns more about the work needed to achieve the Sprint Goal.

As new work is required, the Development Team adds it to the Sprint Backlog. As work is performed or completed, the estimated remaining work is updated. When elements of the plan are deemed unnecessary, they are removed. Only the Development Team can change its Sprint Backlog during a Sprint. The Sprint Backlog is a highly visible, real-time picture of the work that the Development Team plans to accomplish during the Sprint, and it belongs solely to the Development Team.

Abstraction levels for agile requirements



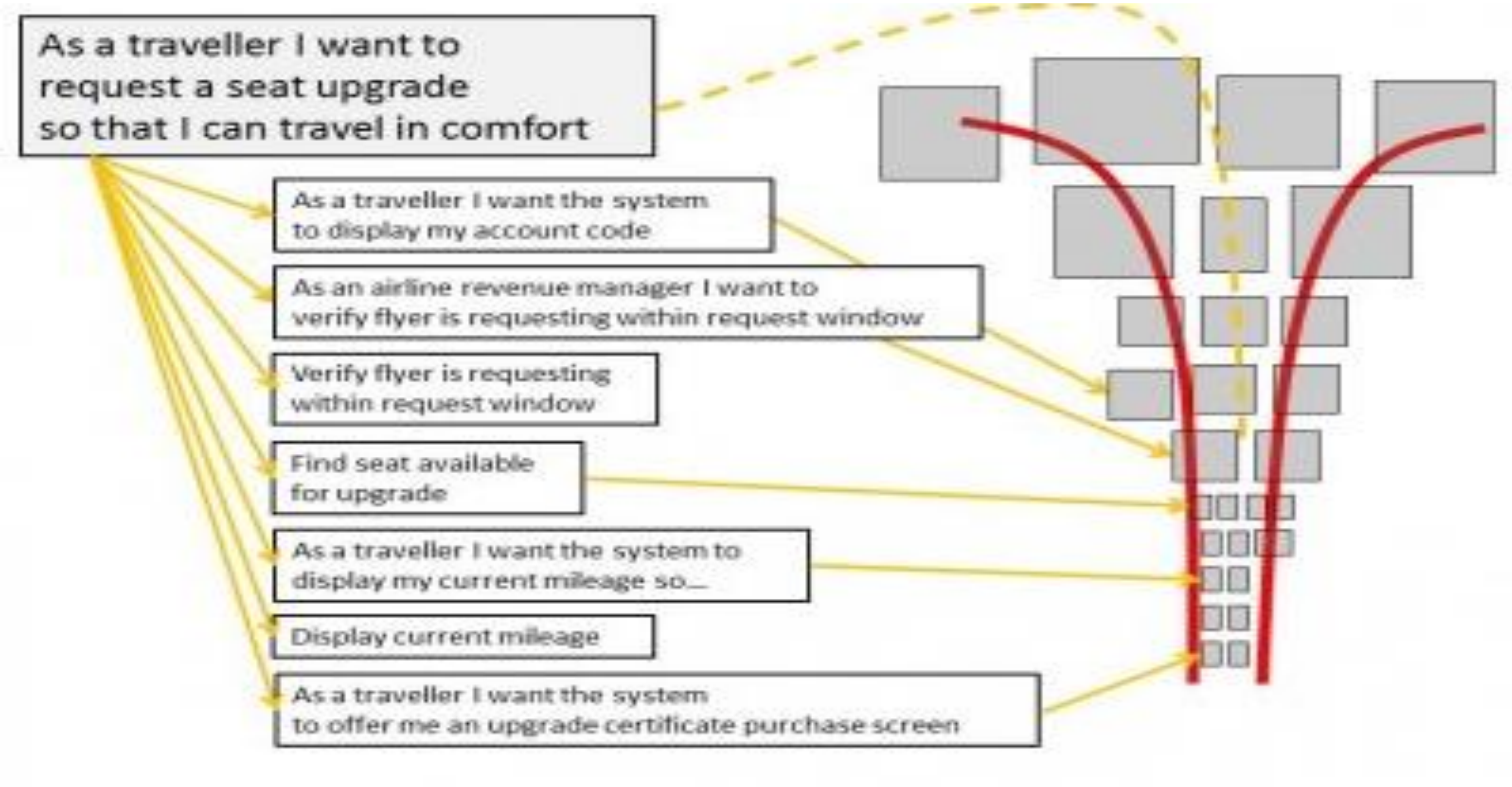
USER STORY briefly **describe what the system can do to solve a specific user problem**. The main feature of the stories is focused on the value that them produce for the subject, unlike the individual tasks because there are a segmentation of activities for the purpose of processing by the team.

FUNCTIONALITY is the **intermediate level between the user's needs and the overall design solution**, connecting the individual activities carried out by the Scrum Team with a terminology used by professionals such as **Program Manager**, **Product Manager** or **other manager**.

EPICS represent the **highest level of expression of a user's need**. They are the pillars of the *big picture*, there are the first elements that emerge from the project concept. These are stories have great or very large dimensions that allow us to **communicate the perimeter of the project in a few bars** but do not allow us to make an initial estimate given the high level of abstraction.

The progressive deepening of the requirements

An example of a progressive deepening of the requirements through the detail of the needs, passing from the epics to the individual stories that will be included in the product backlog.



As a traveller I want to request a seat upgrade so that I can travel in comfort

As a traveller I want the system to display my account code

As an airline revenue manager I want to verify flyer is requesting within request window

Verify flyer is requesting within request window

Find seat available for upgrade

As a traveller I want the system to display my current mileage so...

Display current mileage

As a traveller I want the system to offer me an upgrade certificate purchase screen

The main rules of the game

DOD – DEFINITION OF DONE

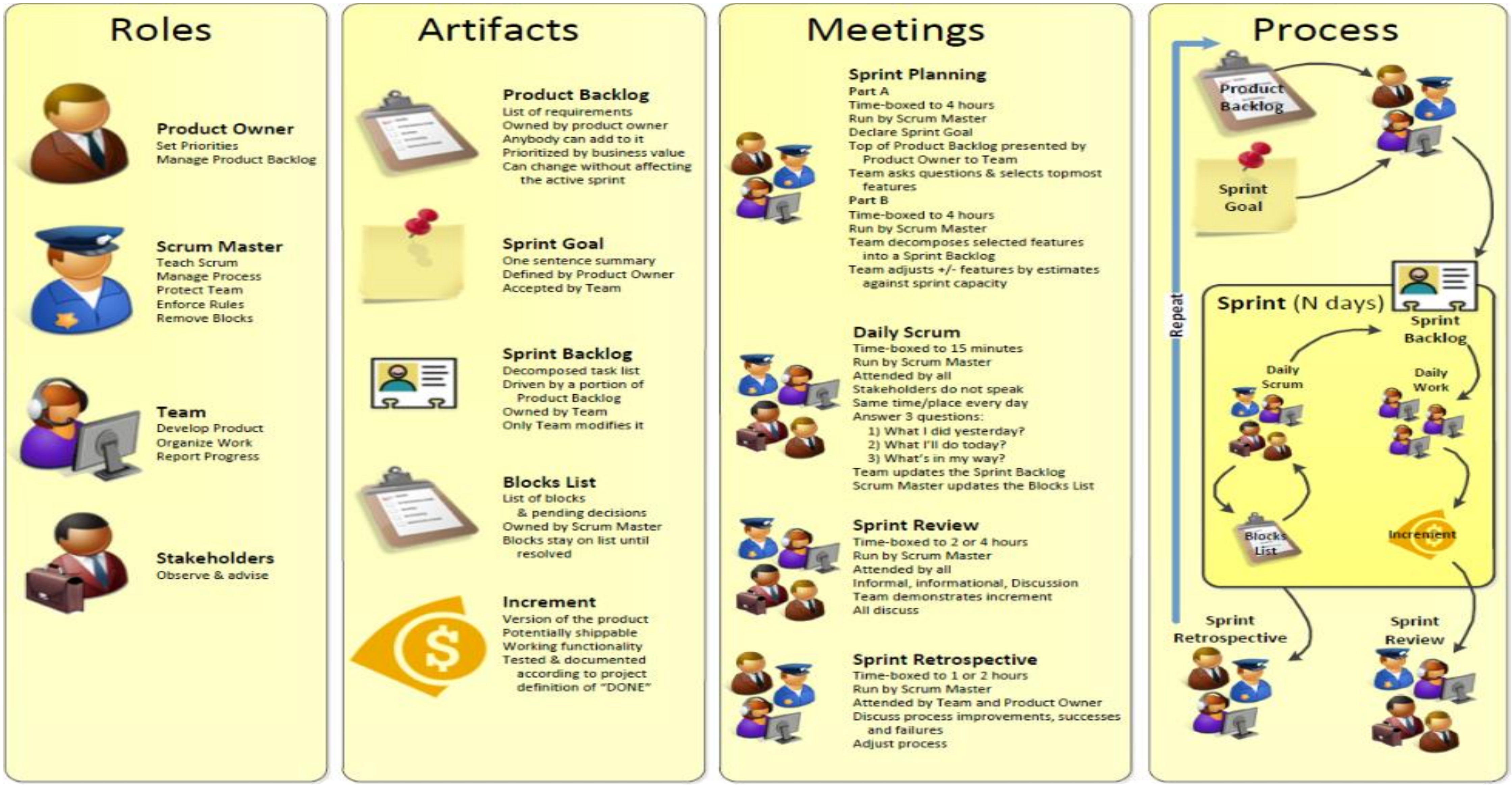
In order to guarantee the right acceptance of developments it is important to consolidate the criteria for DOD (definition of fact):

- The code must be commented, checked and tested
- The build must not have errors
- **CODE INSPECTION** must have been carried out during development activities
- Unit tests carried out and executed successfully before the test
- **USER ACCEPTANCE TEST** performed successfully and confirmed by the Product Owner
- The necessary documentation has been created and released correctly

OTHER RULES

- All Scrum events are mandatory
- It is not possible to change team members during the sprint
- Product Owner and Scrum Master can not be the same person
- The scope, within the current sprint, is frozen
- Sprints are consecutive and do not include breaks or suspensions between sprints and the other

Scrum… In One Page



Roles

Product Owner
Set Priorities
Manage Product Backlog

Scrum Master
Teach Scrum
Manage Process
Protect Team
Enforce Rules
Remove Blocks

Team
Develop Product
Organize Work
Report Progress

Stakeholders
Observe & advise

Artifacts

Product Backlog
List of requirements
Owned by product owner
Anybody can add to it
Prioritized by business value
Can change without affecting the active sprint

Sprint Goal
One sentence summary
Defined by Product Owner
Accepted by Team

Sprint Backlog
Decomposed task list
Driven by a portion of Product Backlog
Owned by Team
Only Team modifies it

Blocks List
List of blocks & pending decisions
Owned by Scrum Master
Blocks stay on list until resolved

Increment
Version of the product
Potentially shippable
Working functionality
Tested & documented according to project definition of "DONE"

Meetings

Sprint Planning
Part A
Time-boxed to 4 hours
Run by Scrum Master
Declare Sprint Goal
Top of Product Backlog presented by Product Owner to Team
Team asks questions & selects topmost features
Part B
Time-boxed to 4 hours
Run by Scrum Master
Team decomposes selected features into a Sprint Backlog
Team adjusts +/- features by estimates against sprint capacity

Daily Scrum
Time-boxed to 15 minutes
Run by Scrum Master
Attended by all
Stakeholders do not speak
Same time/place every day
Answer 3 questions:
1) What I did yesterday?
2) What I'll do today?
3) What's in my way?
Team updates the Sprint Backlog
Scrum Master updates the Blocks List

Sprint Review
Time-boxed to 2 or 4 hours
Run by Scrum Master
Attended by all
Informal, informational, Discussion
Team demonstrates increment
All discuss

Sprint Retrospective
Time-boxed to 1 or 2 hours
Run by Scrum Master
Attended by Team and Product Owner
Discuss process improvements, successes and failures
Adjust process

Process

Agenda

- What is the Agile Methodology
- The Agile manifesto and its principles
- Business Analysis Agile
- Scrum
  - Scrum Roles
  - Scrum Events
  - Scrum Artifacts
- **Scaled Agile Framework**

The Methodologies for Programs And Portfolio

The companies have chosen different solutions for the management of Agile projects within the internal organization.

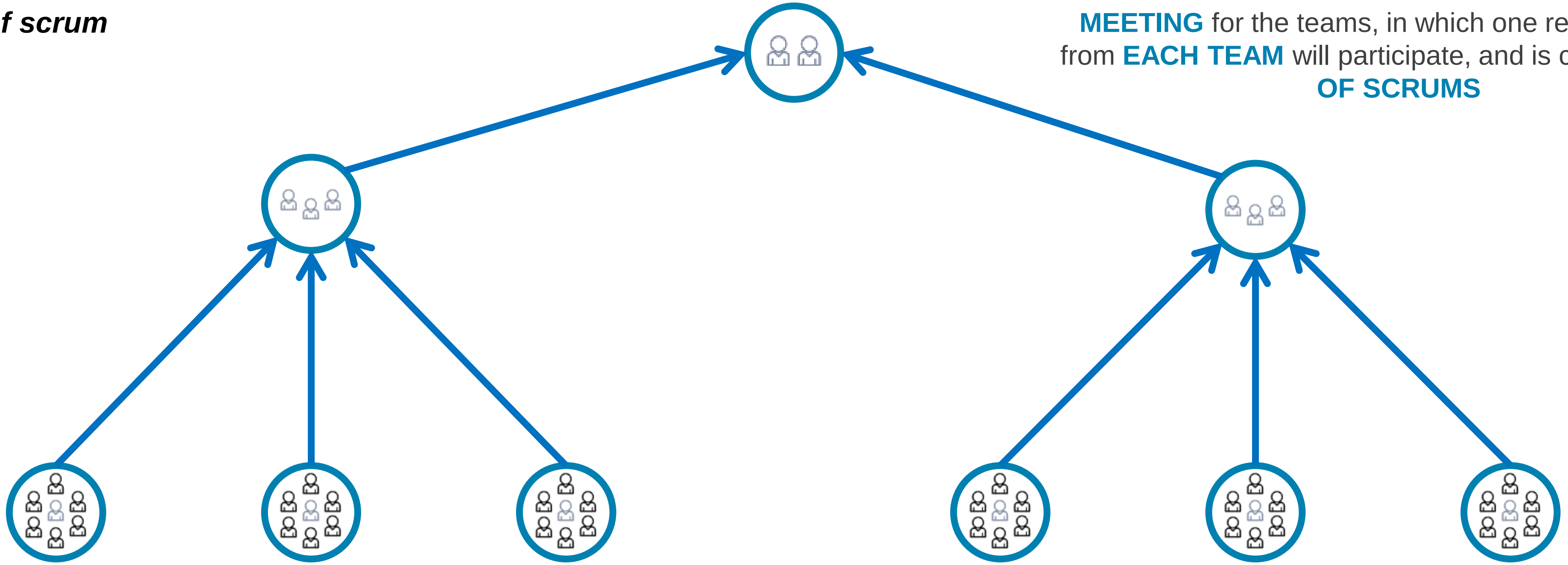
In some companies, several solutions have been adopted simultaneously.

Metodologie ed approcci	**%**
Scrum/Scrum of scums	72
SAFe – Scaled Agile Framework	27
Soluzioni custom interne	23
Lean Management	9
Enterprise Agile	9
Enterprise Scrum	9
Agile Portfolio Management (APM)	9
Large Scale Scrum (Less)	6
Disciplined Agile Delivery (DAD)	4
Recipes for Agile Governance in the Enterprise (RAGE)	1

Data source 10 VersionOne Annual State of Agile Survey (2016)

Scrum Of Scrum

Described the first time in 2001 in an article by Jeff Sutherland and also known as "meta Scrum“, Scrum of Scrum is a **TECHNIQUE TO SCALE SCRUM UP** to large groups (over a dozen people), consisting in dividing the groups into **AGILE TEAMS** of 3-9 people. In more complex scenarios, it could be useful to **INTRODUCE ANOTHER LEVEL** for different coordination objectives, as represented below, with a scaled composition of teams.



Scrum of scrum of scrum

Scrum of scrum

Scrum team

There should BE **A HIGHER LEVEL DAILY STANDUP MEETING** for the teams, in which one representative from **EACH TEAM** will participate, and is called **SCRUM OF SCRUMS**

There should be **ONE SCRUM MASTER FOR EACH TEAM.** One person can be the Scrum Master for more than one team.

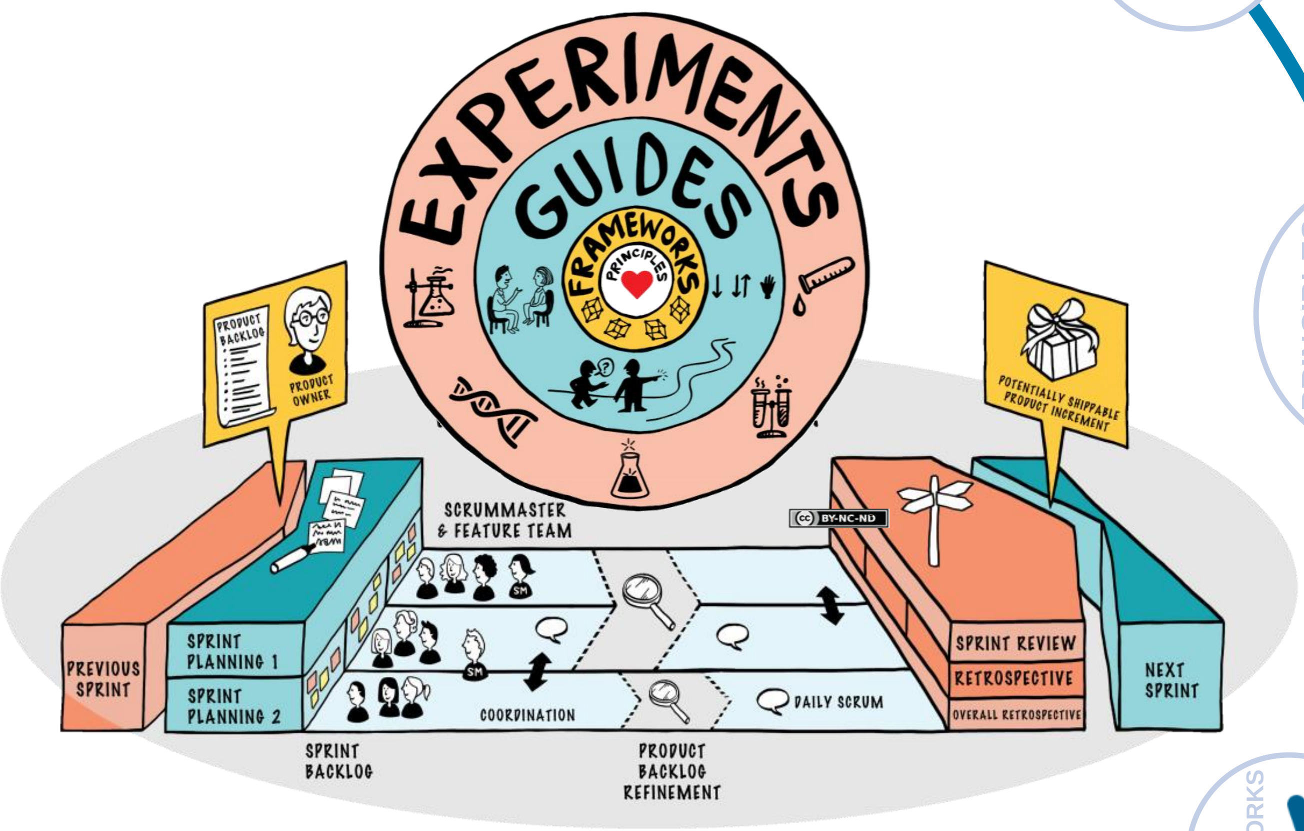
There should be **ONLY ONE PRODUCT BACKLOG** in scaled scrum.

There should BE **ONE MAIN PRODUCT OWNER**. Some resources suggest having local Product Owners for teams, and some other resources suggest limiting ourselves to only one Product Owner

There’s **ONLY ONE INCREMENT** after each Sprint, which is created by the output from all the different teams

Large-scale scrum

LeSS



One Sprint | One 'Done' | Many Teams

BIRTH

Described for the first time in **2013** by Craig and Bas, the Large-Scale Scrum (**LeSS**), is a set of organization and workflow patterns intended to **guide enterprises in scaling Lean and Agile practices.**
It is one of a growing number of frameworks that seeks to address the **problems encountered** when **scaling** beyond a **single team.**

PRINCIPLES

LARGE SCALE SCRUM IS SCRUM

LeSS is Scrum and it is **not a new or improved** Scrum.
LeSS is also not a framework that applies to the team, rather it is a Scrum scaled on all the levels.

- Large Scale Scrum, like regular Scrum, is a framework for development in which the detail needs to be filled in by the teams and evolved **iteration by iteration**.
  It is a collection of suggestions for inspecting and adapting the product and process when there are many teams - at least two teams and up to group of 500 or 1000 people.

FRAMEWORKS

LeSS provides **two** different **large-scale Scrum frameworks**.
Most of the scaling elements of LeSS are focused on directing the attention of all of the teams onto the whole product instead of "my part." Global and "end-to-end" focus are perhaps the dominant problems to solve in scaling. The two frameworks – which are basically single-team Scrum scaled up – are:

☑ **LeSS**: Up to eight teams (of eight people each).

☑ **LeSS Huge**: Up to a few thousand people on one product.

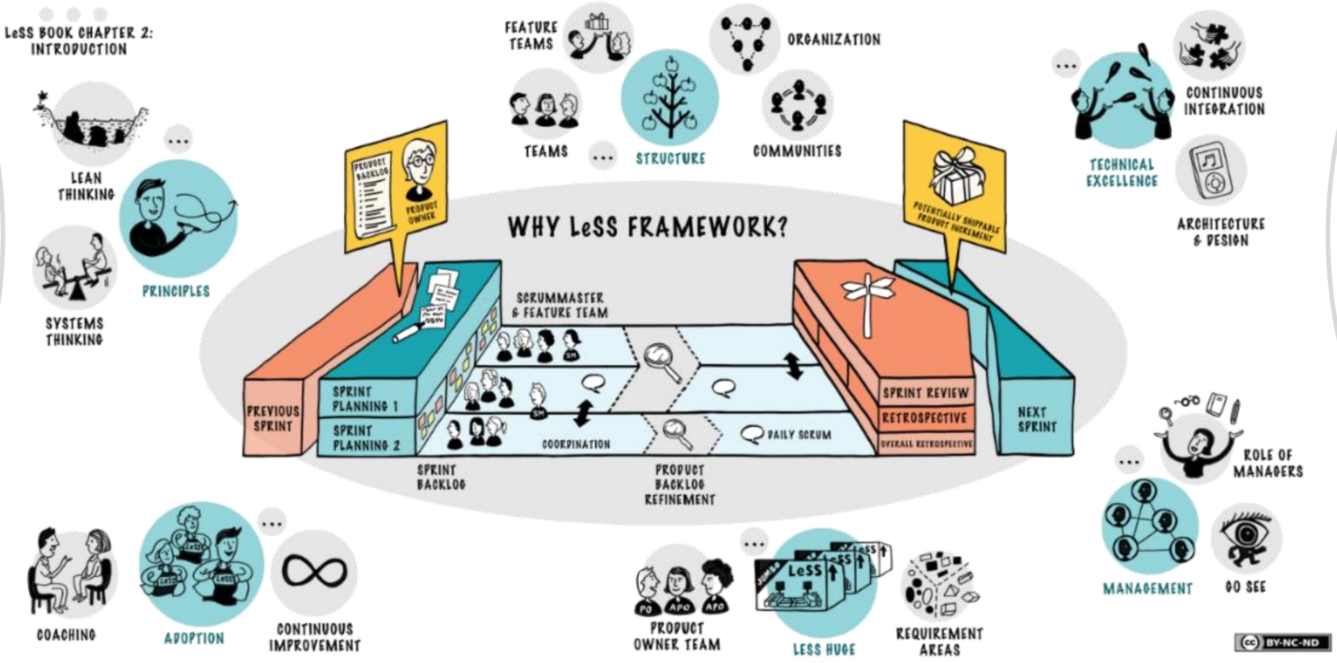
Less framework

LeSS is **more** than **a set of principles and experiments**. It also provides a framework with rules. The LeSS Rules define what is LeSS (and what isn’t) and they provide a concrete framework for applying LeSS. Within the LeSS Framework, product groups can apply the experiments and discover what works best for them at a certain moment.

This way ensures that LeSS is simple and stays true to the Scrum nature. Yet, like Scrum, LeSS provides enough concrete practices to start and enough flexibility and power to scale. LeSS is a scaled up version of one-team Scrum, and it maintains many of the practices and ideas of one team Scrum.

In LeSS, you will find:

- **A** single **Product Backlog** (because it’s for a product, not a team)
- **One** Definition of **Done** for all teams
- **One** Potentially Shippable **Product Increment** at the end of each Sprint
- **One Product Owner**
- **Many** complete, cross-functional **teams** (with no single-specialist teams)
- **One Sprint**



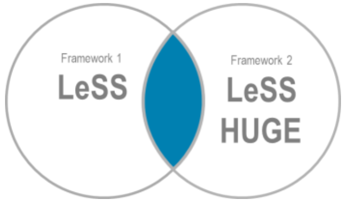
In LeSS **all Teams are in a common Sprint** to deliver a common shippable product, every Sprint

Less HUGE framework

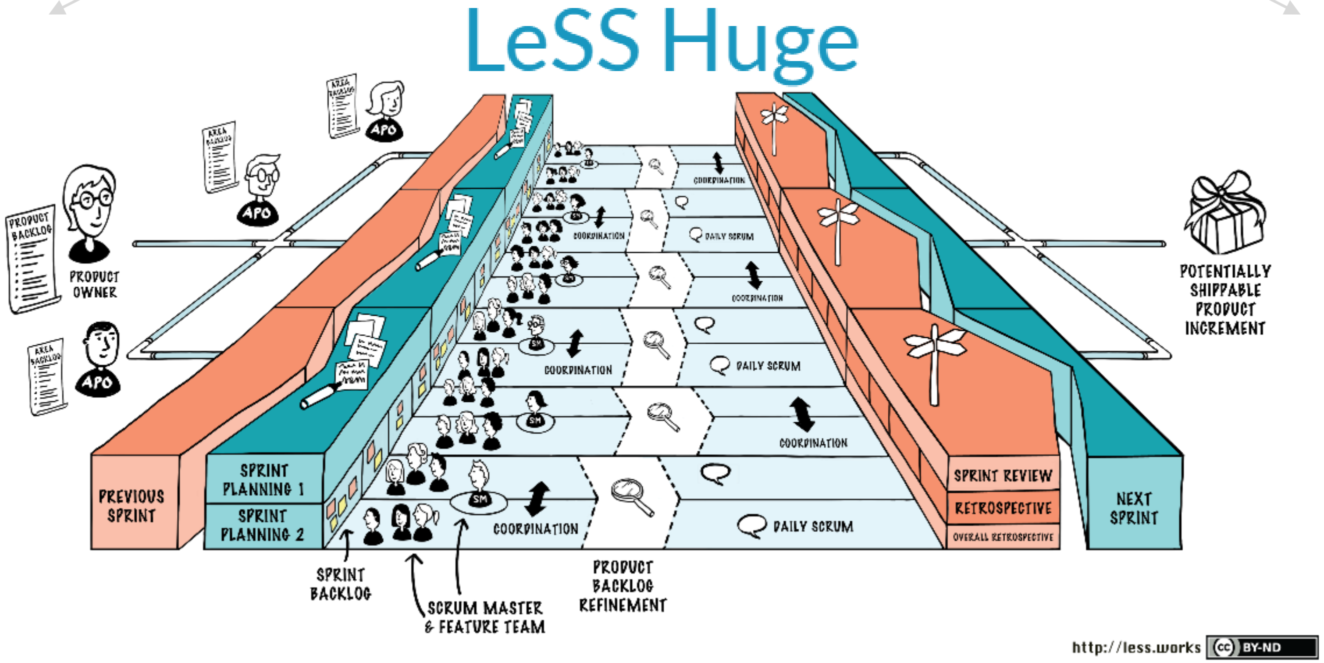
LeSS Huge is the **second LeSS Framework**, which is suitable for LeSS adoptions of **more than eight teams**.
Conceptually it is LeSS scaled up further by having multiple (smaller) LeSS frameworks stacked on top of each other.

As the smaller LeSS Framework

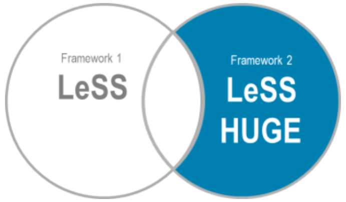
What's the Same?



- One Product Backlog
- One Definition of Done
- One Product Increment
- One Product Owner
- One Sprint
  All Teams in one Sprint
  with one delivery



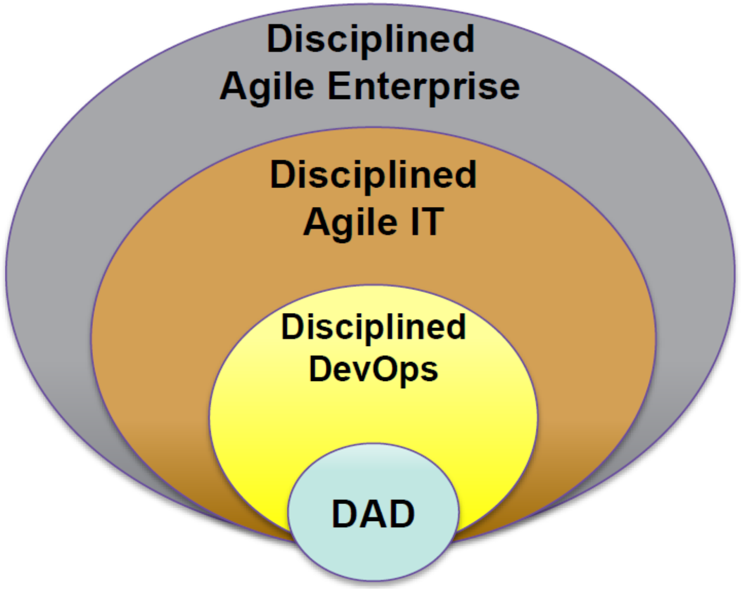
What's Different?

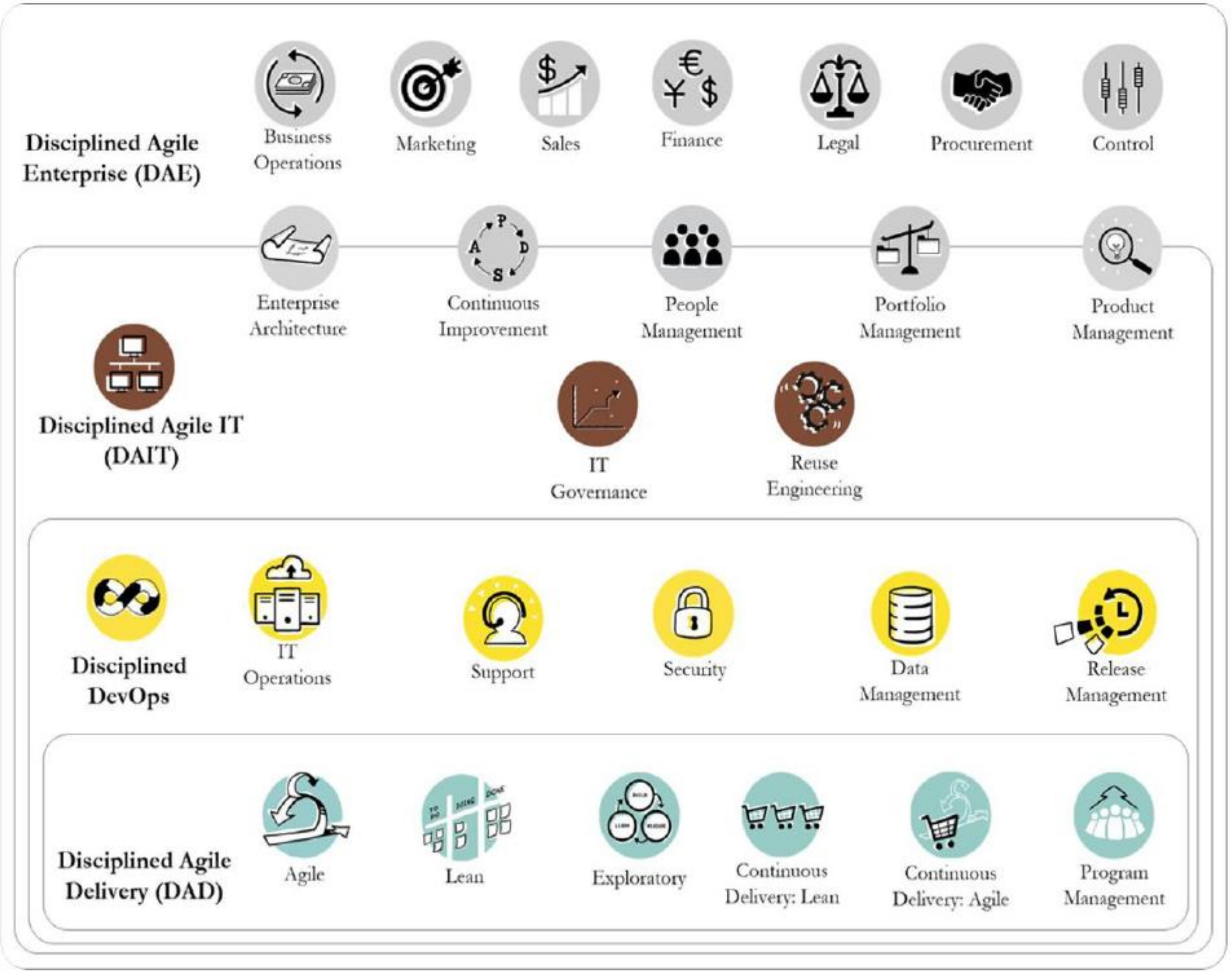


- Role changes → **Area PO**
- Artifact changes →
  -"**Requirement areas**" in Product Backlog
  -**Area Backlog**.
- Meeting changes→
  LeSS Huge is a **set of parallel** (per requirement area) LeSS **Sprint** executions

Disciplined Agile – the framework

Disciplined Agile is a framework divided into 4 components for different aspects, from enterprise management to delivery with the best known DAD.

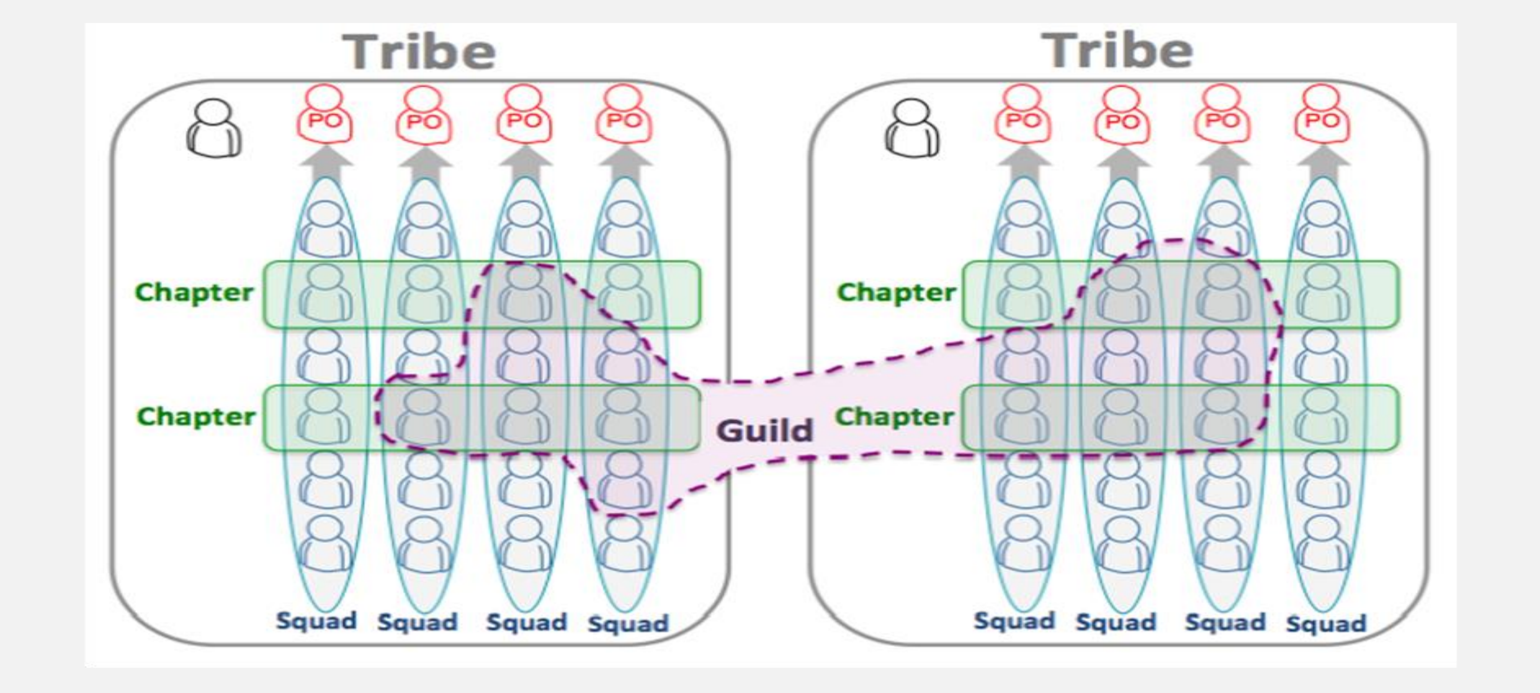




Organization, The Spotify Experience

SPOTIFY "has developed its own vision about the organization, known as " **SPOTIFY ENGINEERING CULTURE** ".
The base unit is the «**TEAM**», which consists in a limited number of resources, from a minimum of 5 to a maximum of 10 persons, in order to develop the projects as mini-startup.

More "**TRIBE**", coordinated by a tribe leader, allowing their teams to achieve high performance.
The "**Guilds**" represent the transversal skills of the tribes as centers of competence.



SAFe®

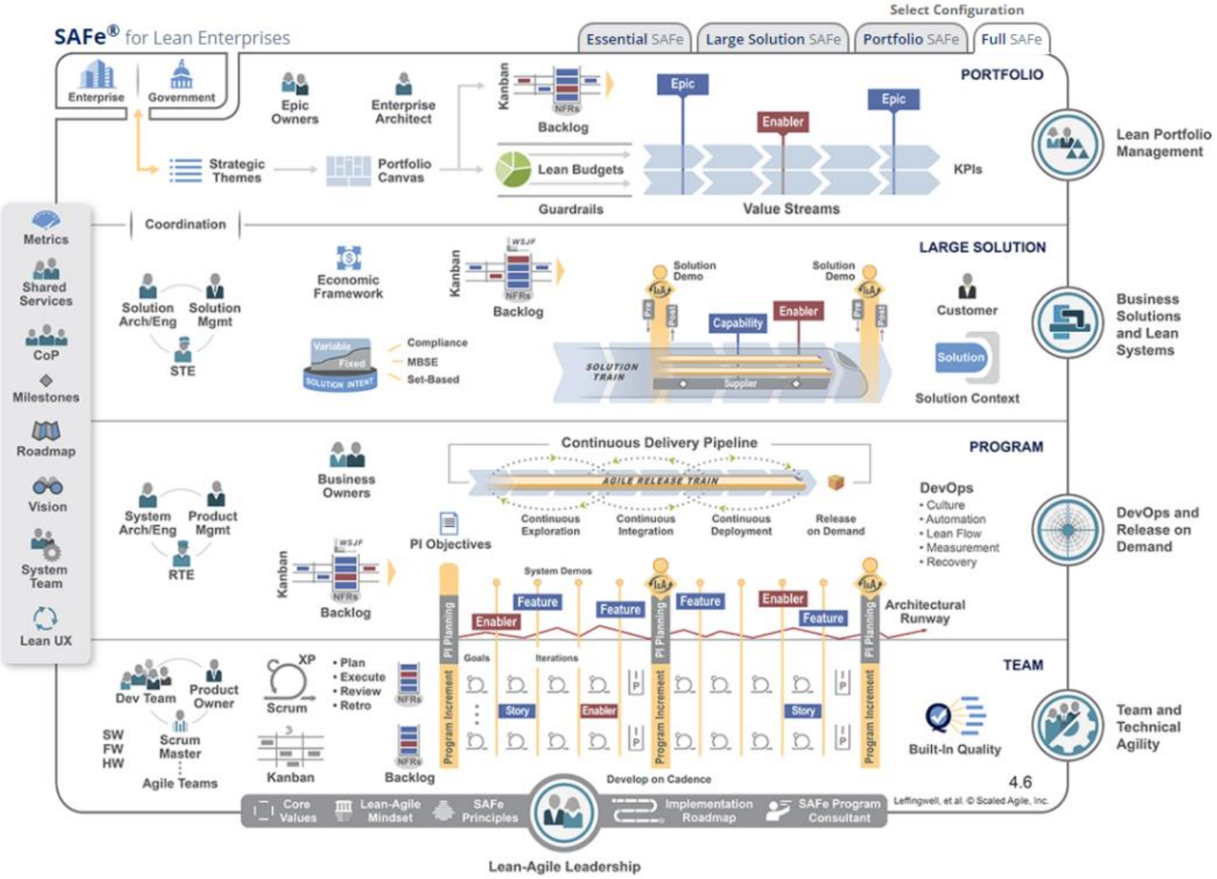
SAFe® - Scaled Agile Framework, first model was realized by Dean Lefflingwell in 2011 with the aim of extending the Agile methodologies to the whole ICT context since the real benefits of Agile can be appreciated only if the company or the ICT department know and use Agile methodologies.

Framework is open and can be read at the link: http://scaledagileframework.com

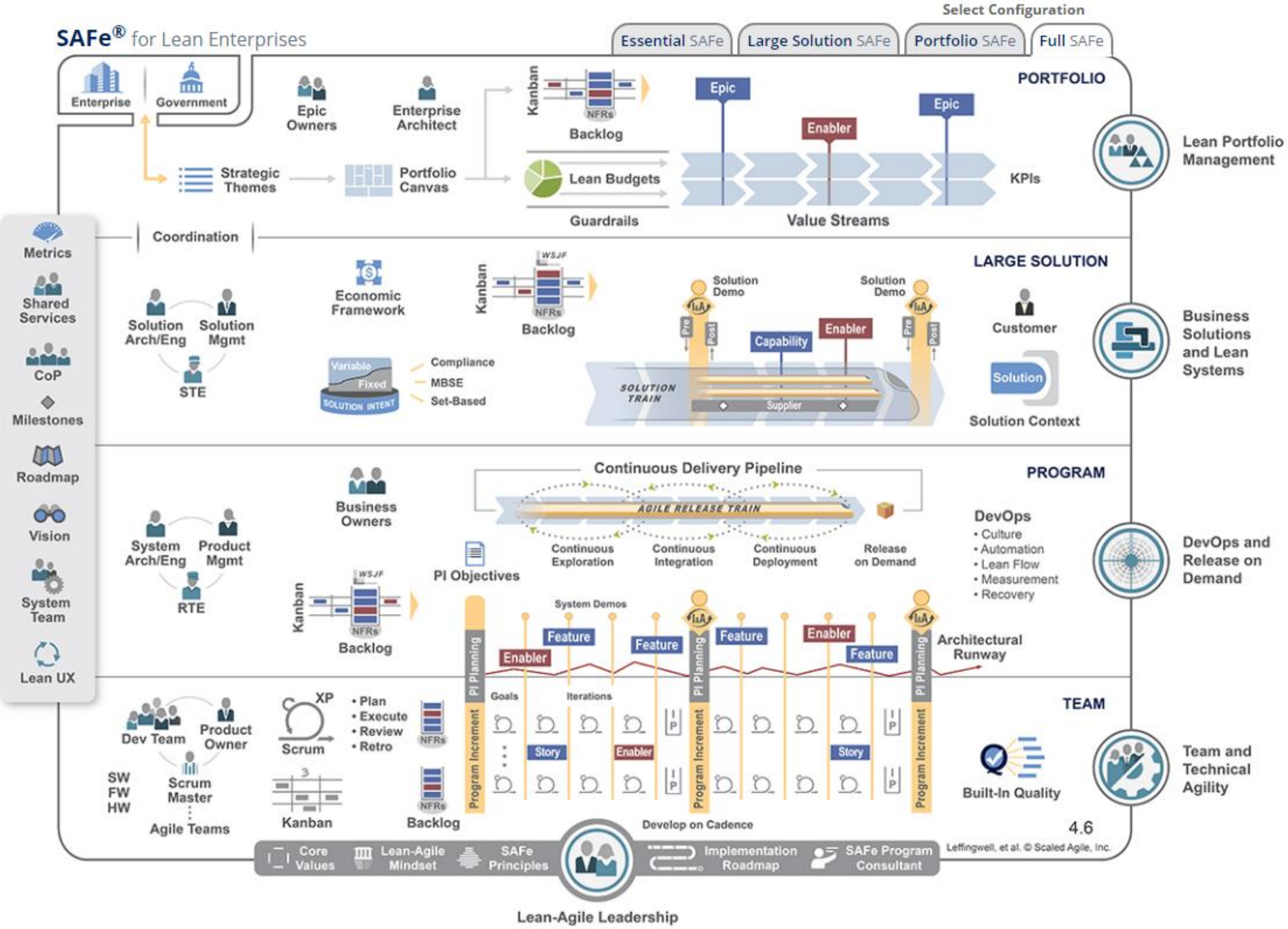
Now it's avalilable the 4.6 Version.

To train people on SAFe is mandatory to use the SAFe documentation from a SPC Certified.

Those slide contains some abstract and screenshots in order to share internal Knowledge.

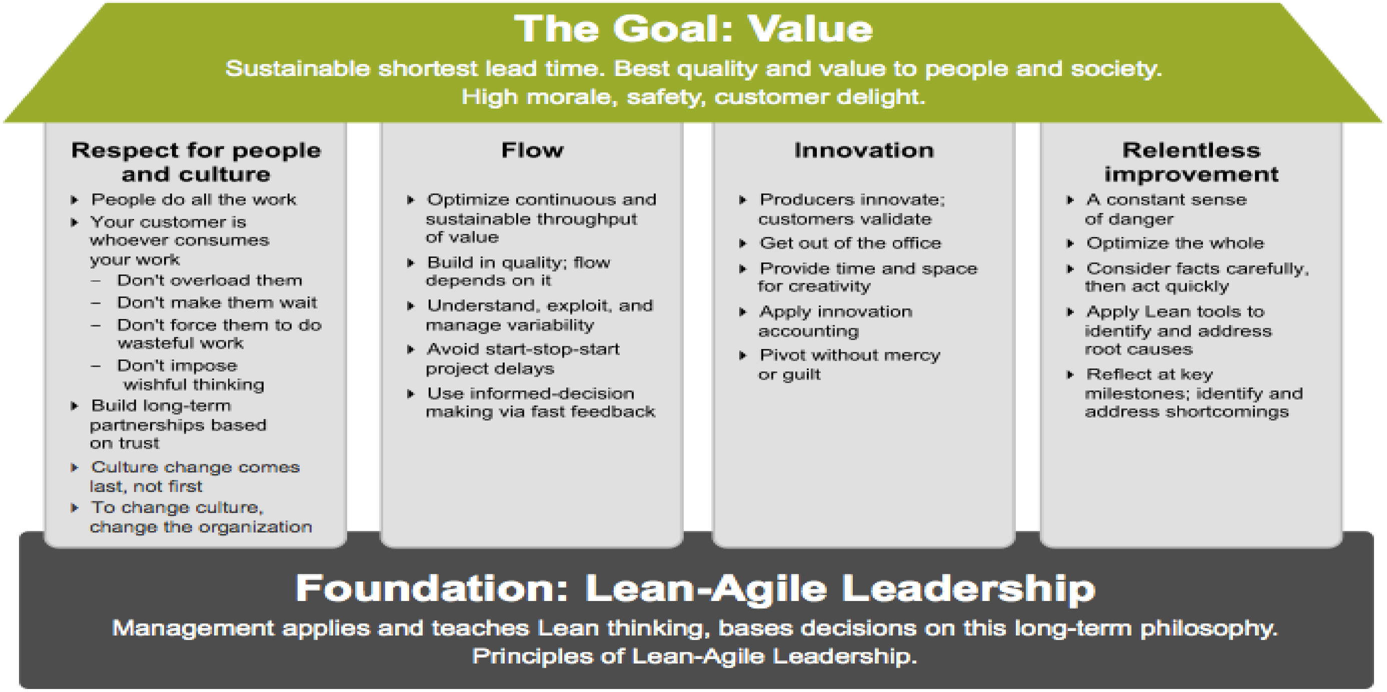


SAFe® 4.6 framework



House od Lean

SAFe® House of Lean starts from Lean principles.



The Goal: Value

Sustainable shortest lead time. Best quality and value to people and society.
High morale, safety, customer delight.

Respect for people and culture

- People do all the work
- Your customer is whoever consumes your work
  - Don't overload them
  - Don't make them wait
  - Don't force them to do wasteful work
  - Don't impose wishful thinking
- Build long-term partnerships based on trust
- Culture change comes last, not first
- To change culture, change the organization

Flow

- Optimize continuous and sustainable throughput of value
- Build in quality; flow depends on it
- Understand, exploit, and manage variability
- Avoid start-stop-start project delays
- Use informed-decision making via fast feedback

Innovation

- Producers innovate; customers validate
- Get out of the office
- Provide time and space for creativity
- Apply innovation accounting
- Pivot without mercy or guilt

Relentless improvement

- A constant sense of danger
- Optimize the whole
- Consider facts carefully, then act quickly
- Apply Lean tools to identify and address root causes
- Reflect at key milestones; identify and address shortcomings

Foundation: Lean-Agile Leadership

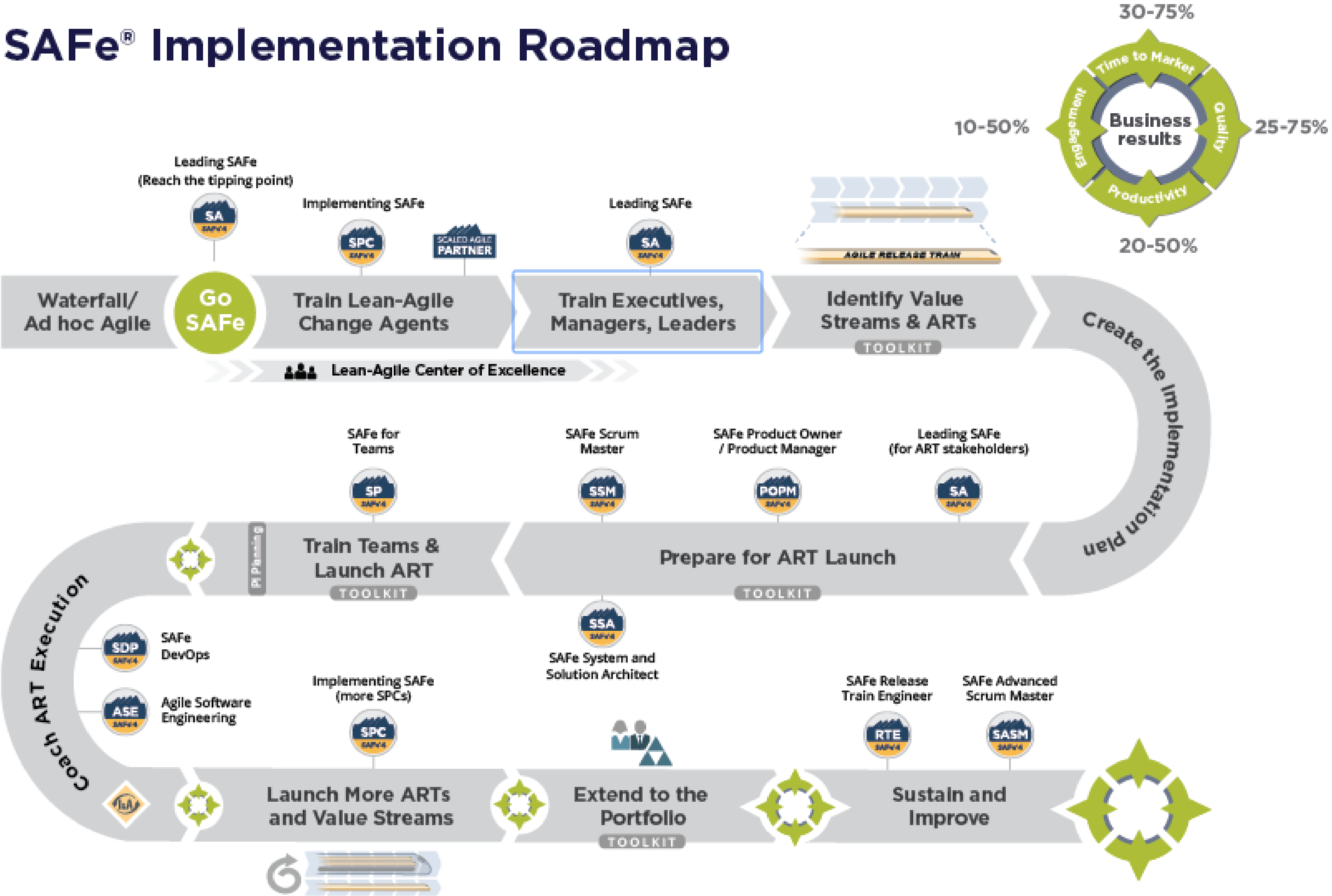
Management applies and teaches Lean thinking, bases decisions on this long-term philosophy.
Principles of Lean-Agile Leadership.

People are already doing their best; the problems are with the system. Only management can change the system.

W. Edwards Deming

SAFe® accepts and includes the Agile Manifesto values and principles.

How to introduce safe


SAFe® Implementation Roadmap
30-75%
Time to Market
10-50%
Engagement
Business results
Quality
25-75%
Productivity
20-50%
Leading SAFe (Reach the tipping point)
SA
Implementing SAFe
SPC
SCALED AGILE PARTNER
Leading SAFe
SA
AGILE RELEASE TRAIN
Waterfall/ Ad hoc Agile
Go SAFe
Train Lean-Agile Change Agents
Train Executives, Managers, Leaders
Identify Value Streams & ARTs
TOOLKIT
Lean-Agile Center of Excellence
Create the Implementation Plan
SAFe for Teams
SP
SAFe Scrum Master
SSM
SAFe Product Owner / Product Manager
POPM
Leading SAFe (for ART stakeholders)
SA
PI Planning
Train Teams & Launch ART
TOOLKIT
Prepare for ART Launch
TOOLKIT
SSA
SAFe System and Solution Architect
Coach ART Execution
SDP
SAFe DevOps
ASE
Agile Software Engineering
Implementing SAFe (more SPCs)
SPC
SAFe Release Train Engineer
RTE
SAFe Advanced Scrum Master
SASM
Launch More ARTs and Value Streams
Extend to the Portfolio
TOOLKIT
Sustain and Improve

NTT DATA

Trusted Global Innovator