M.EIC, 2022-23

Large Scale Software Development

Filipe Correia, Daniel Pinho, João Pedro Dias

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

DEI
DEPARTAMENTO DE
ENGENHARIA INFORMÁTICA

Managing in a VUCA World

Managing in a VUCA World

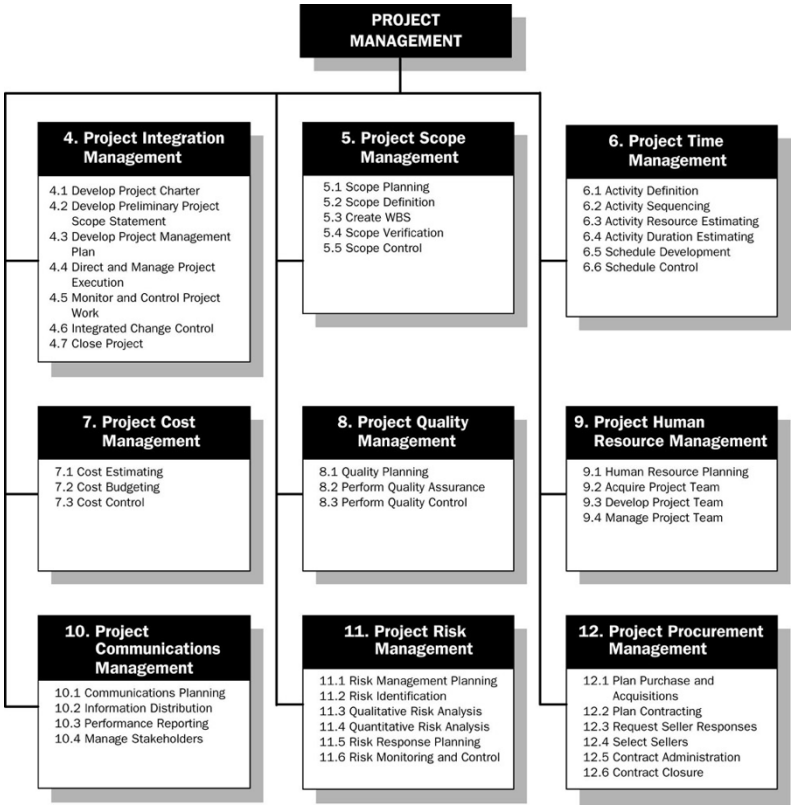
Volatility - change is rapid and **unpredictable** in its nature and extent

Uncertainty - the present is unclear and the future is **uncertain**

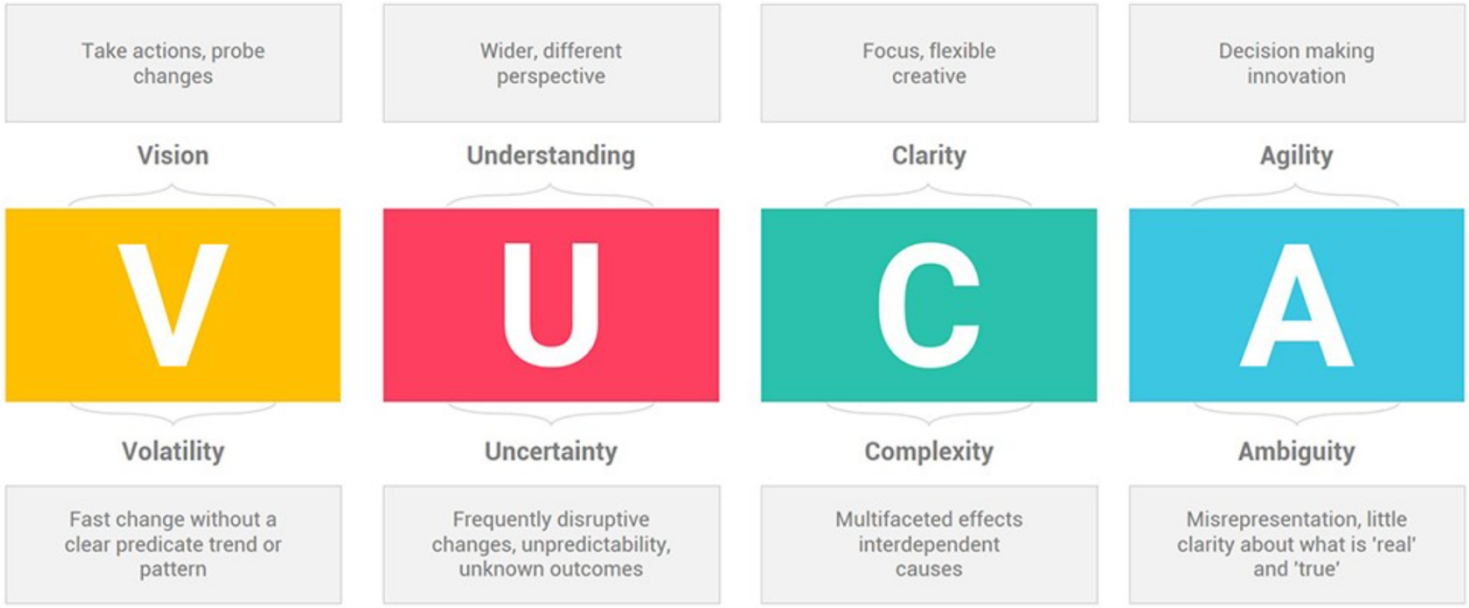
Complexity - many different, interconnected **factors** come into play, with the potential to cause chaos and confusion

Ambiguity - there is a lack of clarity or awareness about what reality is

PMBOK Guide



Managing in a VUCA World

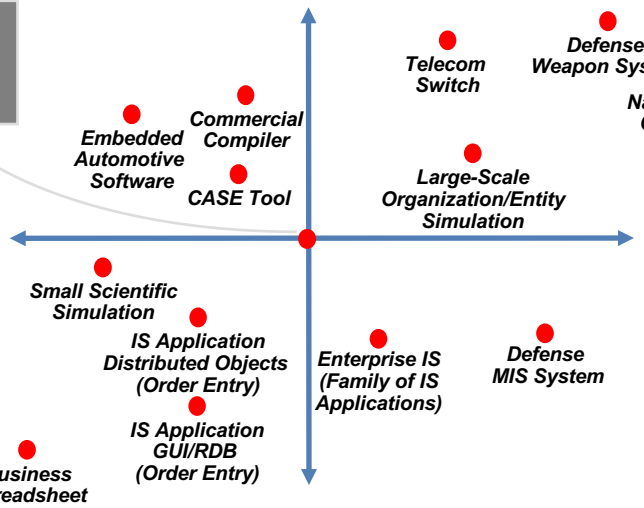


Software Complexity

Higher technical complexity

- Embedded, real-time, distributed, fault-tolerant
- Custom, unprecedented, architecture reengineering
- High performance

An average software project
- 5-10 people
- 3-9 month duration
- 3-5 external interfaces
- Some unknowns & risks



Lower management complexity

- Small scale
- Informal
- Single stakeholder
- "Products"

Higher management complexity

- Large scale
- Contractual
- Many stakeholders
- "Projects"

Lower technical complexity

- Mostly 4GL, or component-based
- Application reengineering
- Interactive performance

Walker Royce

DEI DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

U.PORTO

Agility

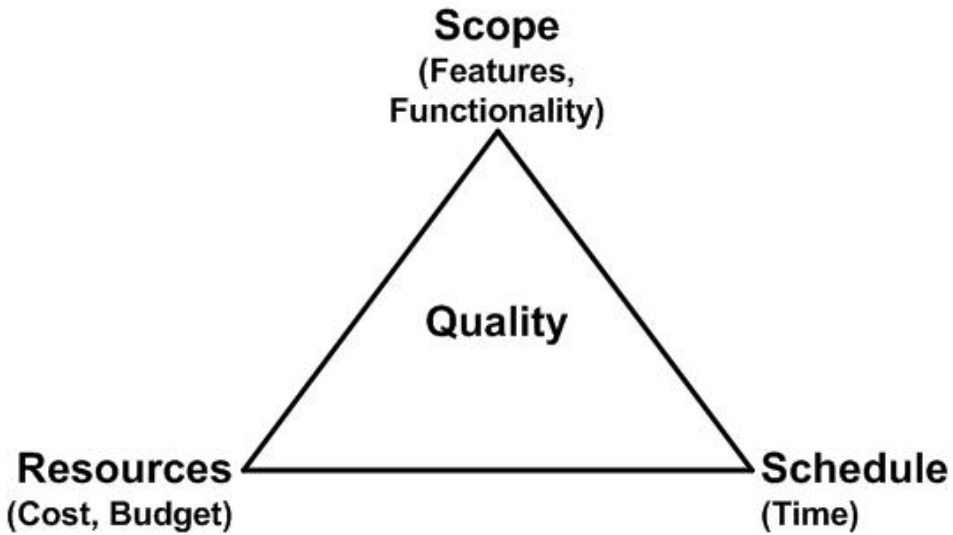
All we want is …

... **high quality** of the product/service.

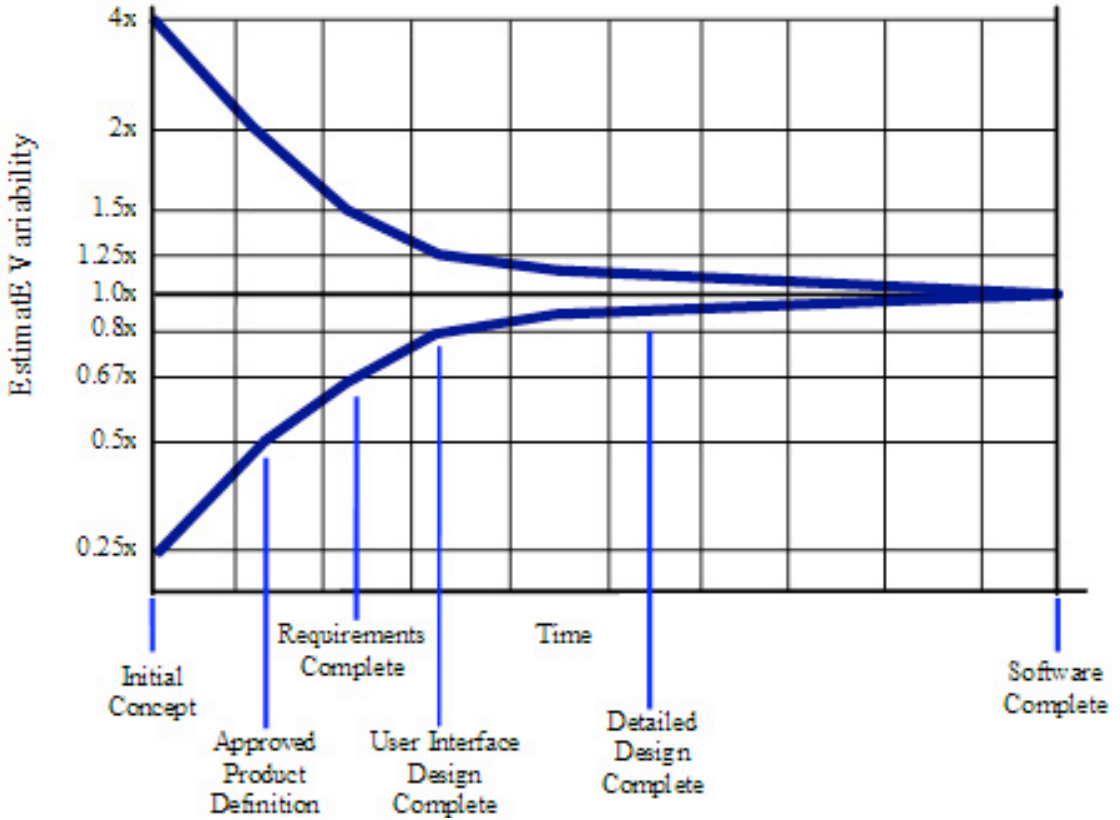
... **high productivity** of construction.

... **good predictability** of results.

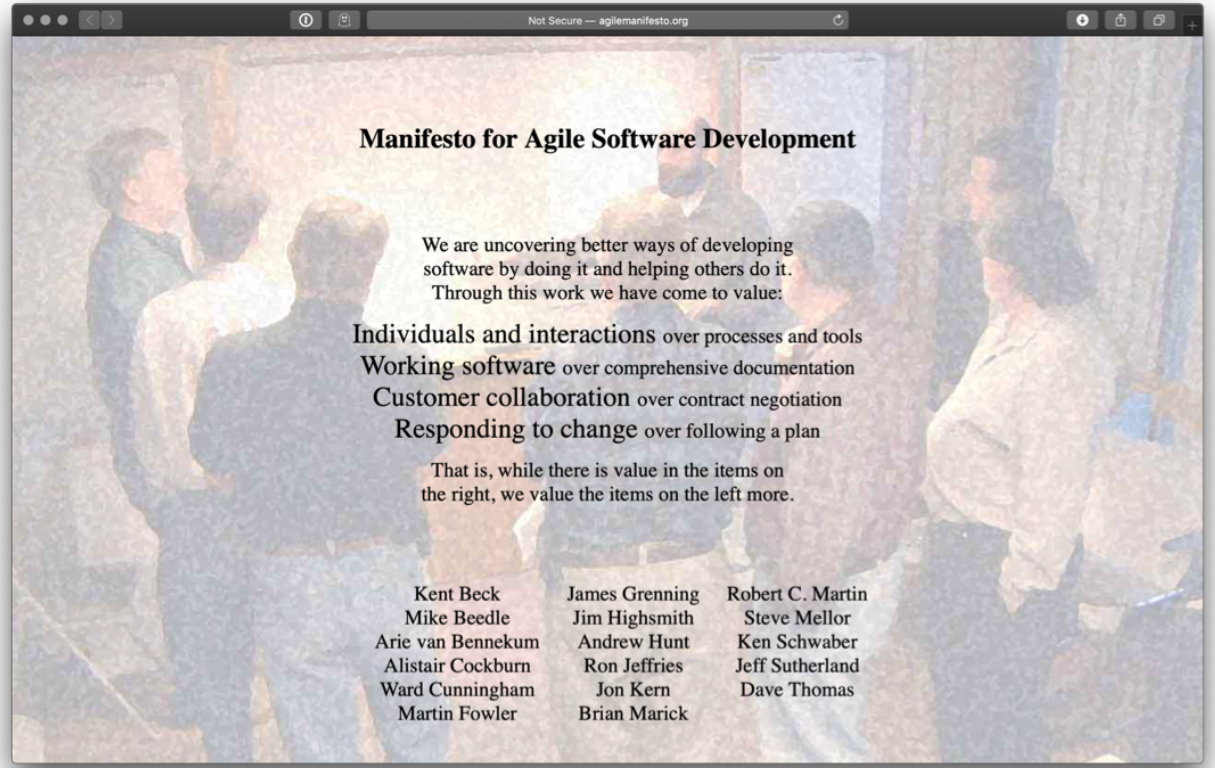
The Iron Triangle



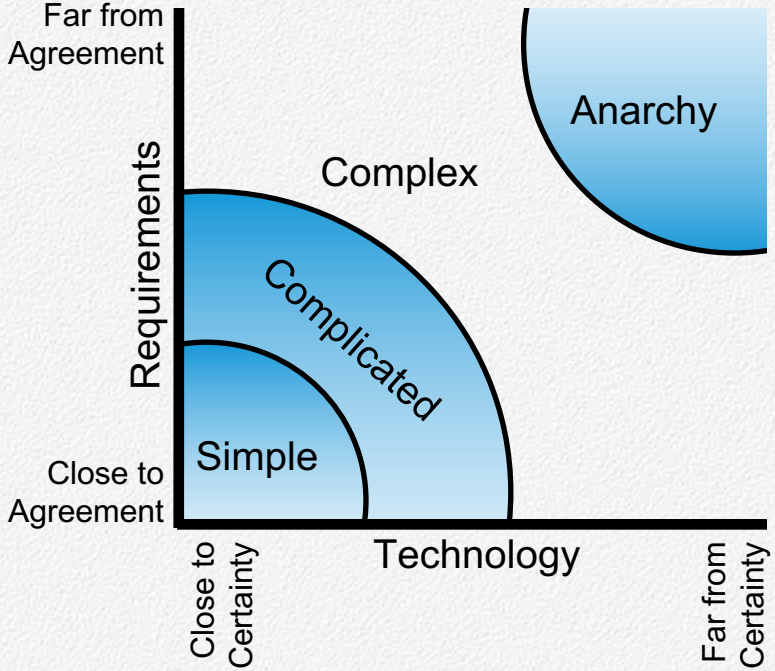
Scope & Variability



Agile: ASD, Crystal Clear, Scrum, XP…

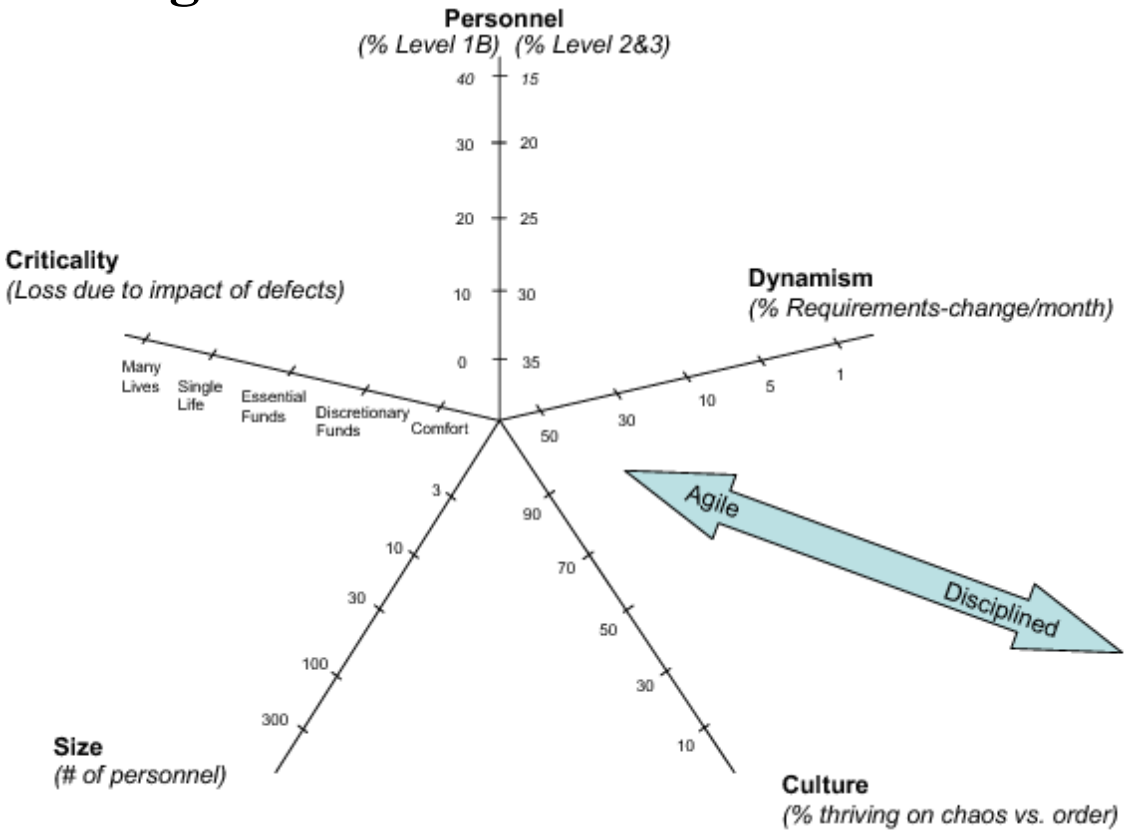


Project Uncertainty



Source: *Strategic Management and Organizational Dynamics* by Ralph Stacey in *Agile Software Development with Scrum* by Ken Schwaber and Mike Beedle.

How much agile?



The Right Conditions for Agile

CONDITIONS	FAVORABLE	UNFAVORABLE
Market Environment	Customer preferences and solution options change frequently.	Market conditions are stable and predictable.
Customer Involvement	Close collaboration and rapid feedback are feasible. Customers know better what they want as the process progresses.	Requirements are clear at the outset and will remain stable. Customers are unavailable for constant collaboration.
Innovation Type	Problems are complex, solutions are unknown, and the scope isn't clearly defined. Product specifications may change. Creative breakthroughs and time to market are important. Cross-functional collaboration is vital.	Similar work has been done before, and innovators believe the solutions are clear. Detailed specifications and work plans can be forecast with confidence and should be adhered to. Problems can be solved sequentially in functional silos.
Modularity of Work	Incremental developments have value, and customers can use them. Work can be broken into parts and conducted in rapid, iterative cycles. Late changes are manageable.	Customers cannot start testing parts of the product until everything is complete. Late changes are expensive or impossible.
Impact of Interim Mistakes	They provide valuable learning.	They may be catastrophic.

SOURCE BAIN & COMPANY

FROM "EMBRACING AGILE," MAY 2016

Harvard Business Review

CHANGE MANAGEMENT

How Agile Teams Can Help Turnarounds Succeed

by Darrell K. Rigby, Simon Henderson, and Marco D'Avino

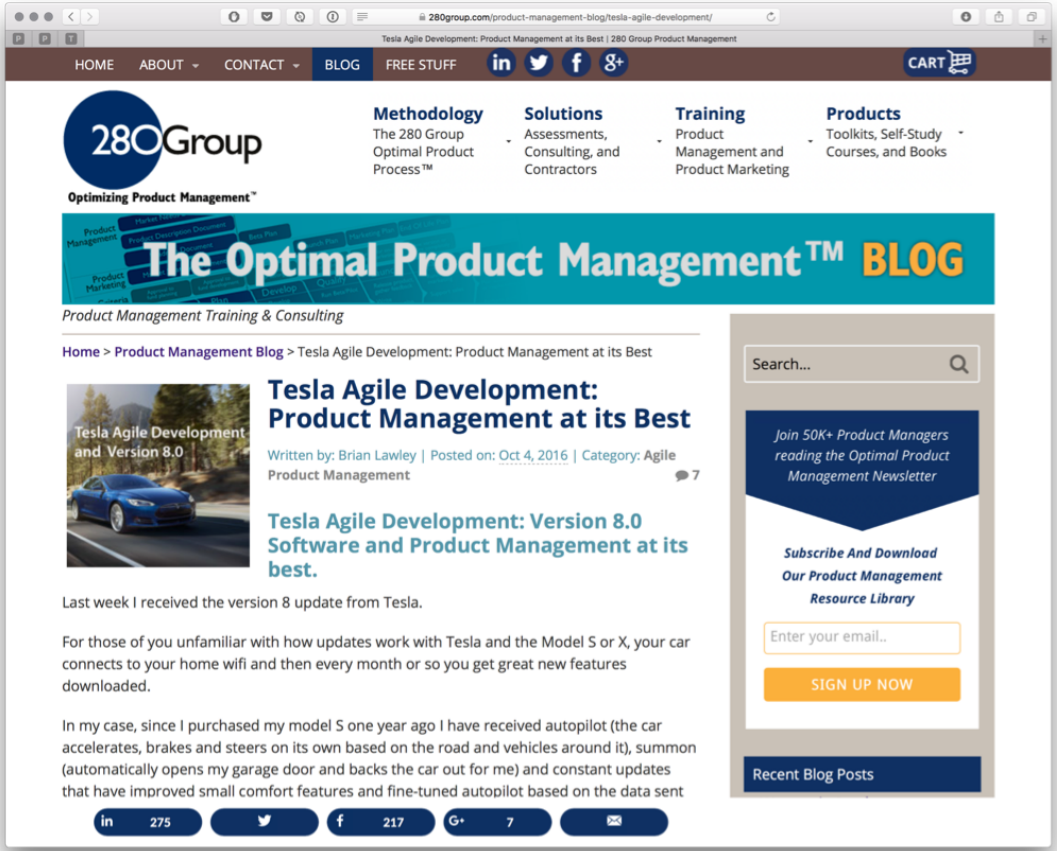
JULY 02, 2018

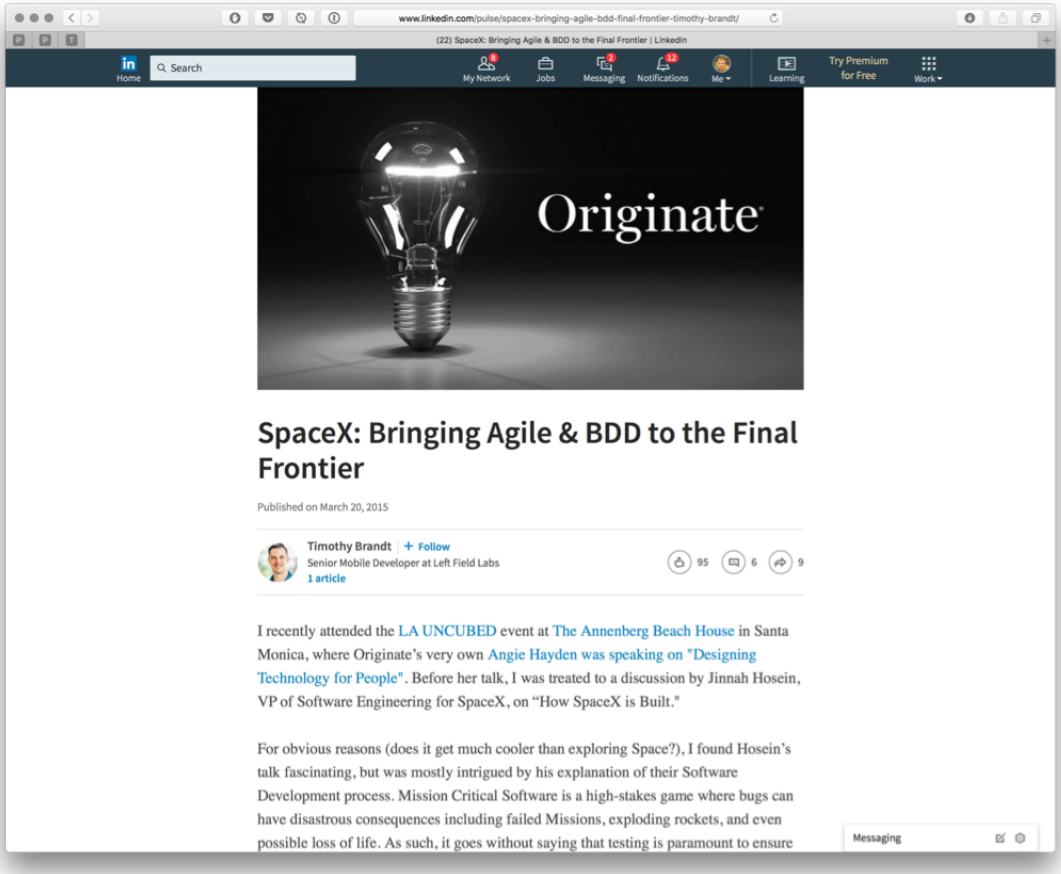
PATRICK SMITH/GETTY IMAGES

Agile — the management approach that relies on small, entrepreneurial, close-to-the-customer teams — has a reputation that reflects its rapid adoption in software development. It's for techies. It's for hip Silicon Valley startups. It is most definitely not

COMPANIES AT THE FOREFRONT ARE GIVING GLIMPSES INTO THE FUTURE

TESLA'S PLAN TO KILL RANGE ANXIETY WITH A SOFTWARE UPDATE





Harvard Business Review

SPONSOR CONTENT FROM CA TECHNOLOGIES

Survey Data Shows That Many Companies Are Still Not Truly Agile

MARCH 22, 2018

In Today's Digital Economy, Agile Practices Can't Be Limited to Just the IT and Development Realms

By Surya Panditi, SVP and GM, Agile Management, CA Technologies

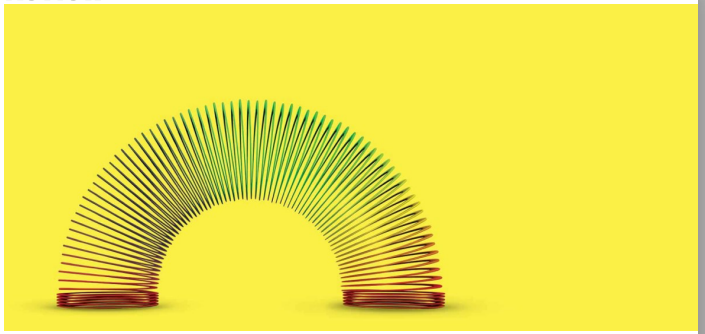
Agile practices have a vital part to play in the rapid delivery and continuous maintenance of software-driven products and services.

When software is ubiquitous, agile needs to be likewise. It's no use having technology that's responsive to the business if the business can't respond to technology and the demands coming from its customers.

To mitigate such risks, agile turnaround leaders typically take five actions:

- **They communicate — even over-communicate — the strategic ambition to a broader range of people.** Since leaders know they will be delegating far more decisions, they ensure that people making those decisions are fully aligned on what to do and why to do it. That way, how they do it can be flexible yet faithful to the strategy.
- **They serve as coaches, not commanders.** In a turnaround, people are afraid to make mistakes, so they bring decisions to their boss. Strong leaders act as coaches and trainers to expand the quantity and quality of decision makers.
- **They strengthen lines of communication among the teams.** To avoid becoming a bottleneck, they develop tools that help everyone see what all the teams are doing at any time.
- **They accelerate learning loops, emphasizing progress over perfection.** They embrace unpredictability and don't get slowed by excessive precision. Adequate approximations will do.
- **They shift measurement and reward systems to larger teams.** One of the biggest problems in a crisis is that people focus on doing what is best for the individuals they know and trust — which often means people in their own silos. Effective turnaround leaders enlarge circles of trust and collaboration.

Harvard
Business
Review



CHANGE MANAGEMENT

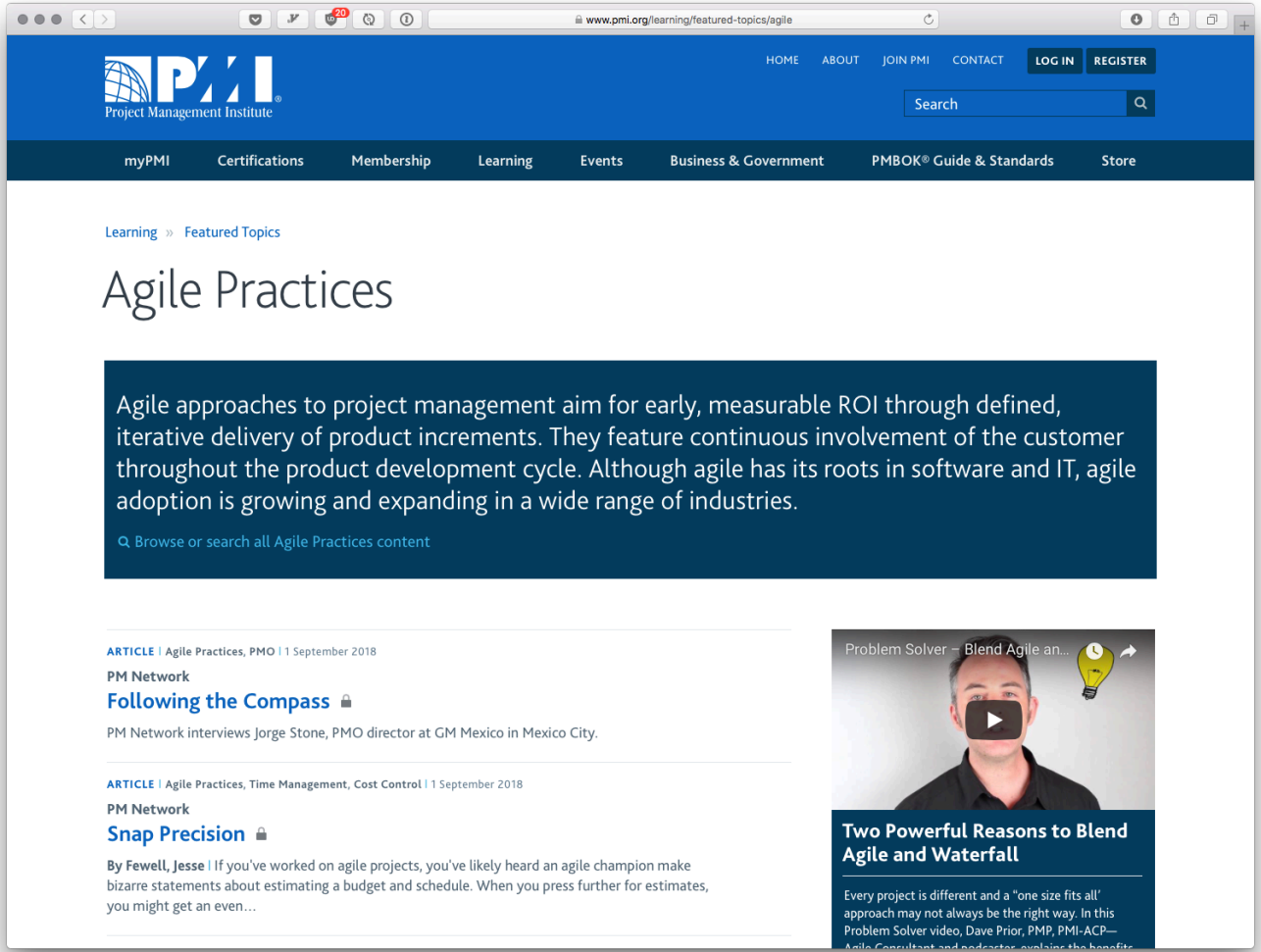
Agile at Scale

by Darrell K. Rigby, Jeff Sutherland, and Andy Noble

FROM THE MAY–JUNE 2018 ISSUE

By now most business leaders are familiar with agile innovation teams. These small, entrepreneurial groups are designed to stay close to customers and adapt quickly to changing conditions. When implemented correctly, they almost always result in higher team productivity and morale, faster time to market, better quality, and lower risk than traditional approaches can achieve.

Naturally, leaders who have experienced or heard about agile teams are asking some compelling questions. What if a company were to launch dozens, hundreds, or even thousands of agile teams throughout the organization? Could whole segments of the



www.pmi.org/pmbok-guide-standards/practice-guides/agile

Project Management Institute

HOME ABOUT JOIN PMI CONTACT LOG IN REGISTER

Search

myPMI Certifications Membership Learning Events Business & Government PMBOK® Guide & Standards Store

PMBOK® Guide and Standards » Practice Guides

Agile Practice Guide

Like Tweet Share G+

PRACTICE GUIDE | **Agile Practices** | September 2017

How to cite this article:
Agile Practice Guide (2017).

Receive the *Agile Practice Guide* when you purchase the *PMBOK® Guide* – Sixth Edition.

ORDER NOW

Created in partnership with Agile Alliance®, the *Agile Practice Guide** provides tools, situational guidelines and an understanding of the various agile approaches available to enable better results. It is especially useful for those project managers accustomed to a more traditional environment to adapt to a more agile approach.

The *Agile Practice Guide* contains the following sections:

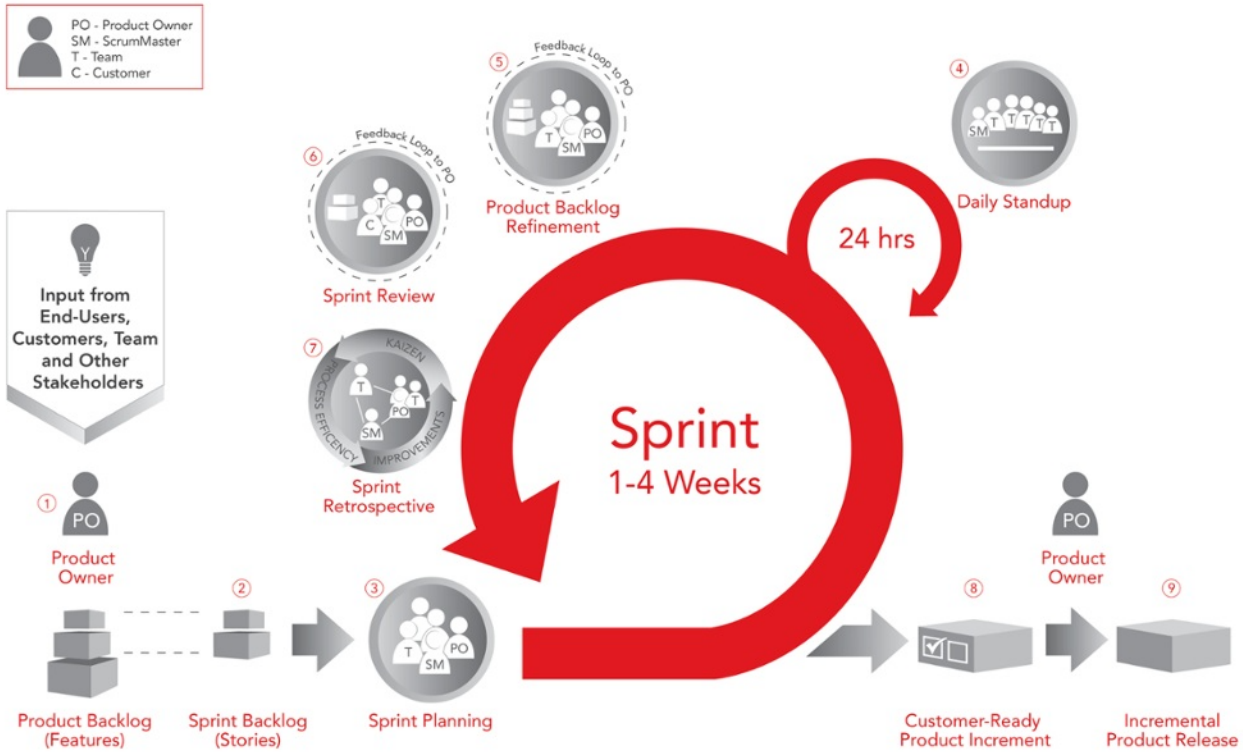
- **An Introduction to Agile** describes the *Agile Manifesto* mindset, values and principles. It also covers the concepts of definable and high-uncertainty work, and the correlation between lean, the Kanban Method and agile approaches.
- **Life Cycle Selection** introduces the various life cycles discussed in the practice guide and covers suitability filters, tailoring guidelines and common combinations of approaches.
- **Implementing Agile: Creating an Agile Environment** talks about critical factors to consider when creating an agile environment such as servant leadership and team composition.

Errata Sheets

Find the latest corrections and updates to the Agile Practice Guide

Scrum

Scrum



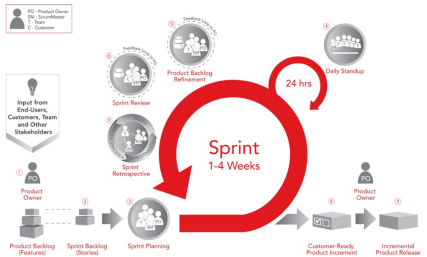
https://www.scruminc.com/scrum-framework/

Scrum: foundations

- Trust
- Focus
- Transparency
- Courage
- Respect
- Commitment
- Trust :)

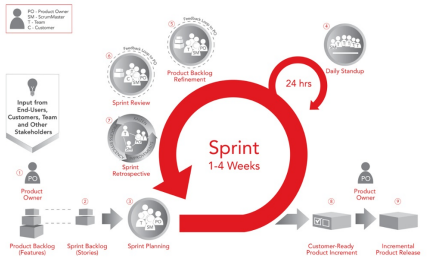
Team

- Typically 5-9 people
- Cross-functional:
  - Programmers, testers, user experience designers, etc.
- Members should be full-time
  - May be exceptions (e.g., expert on a specific API)
- Teams are self-organizing
  - Ideally, no titles but rarely a possibility
- Membership should change only between sprints



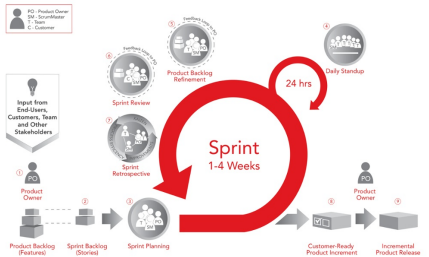
Scrum Master

- Represents management to the project
- Responsible for enacting Scrum values and practices
- Removes impediments
- Ensure that the team is fully functional and productive
- Enable close cooperation across all roles and functions
- Shield the team from external interferences

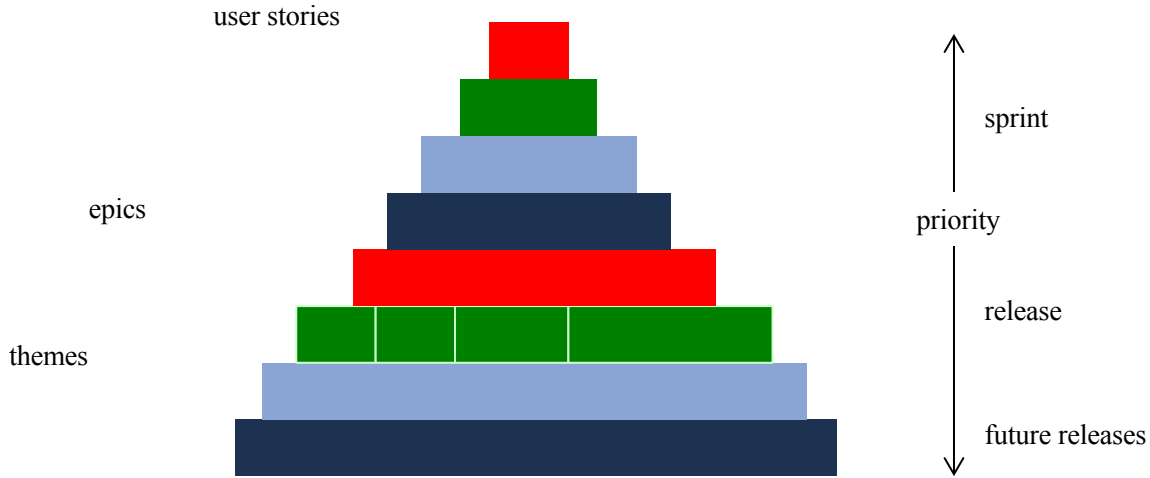


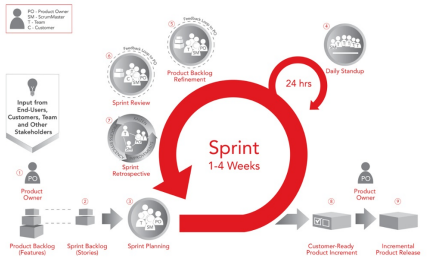
Product Owner

- Define the features of the product
- Decide on release date and content
- Be responsible for the profitability of the product (ROI)
- Prioritize features according to market value
- Adjust features and priority every iteration, as needed
- Accept or reject work results



Themes, epics, user stories -> priorities



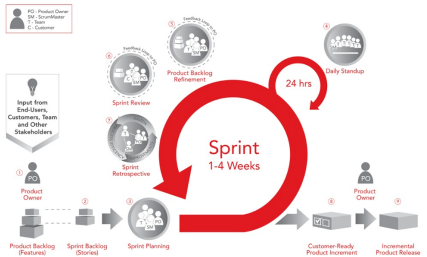


User Stories

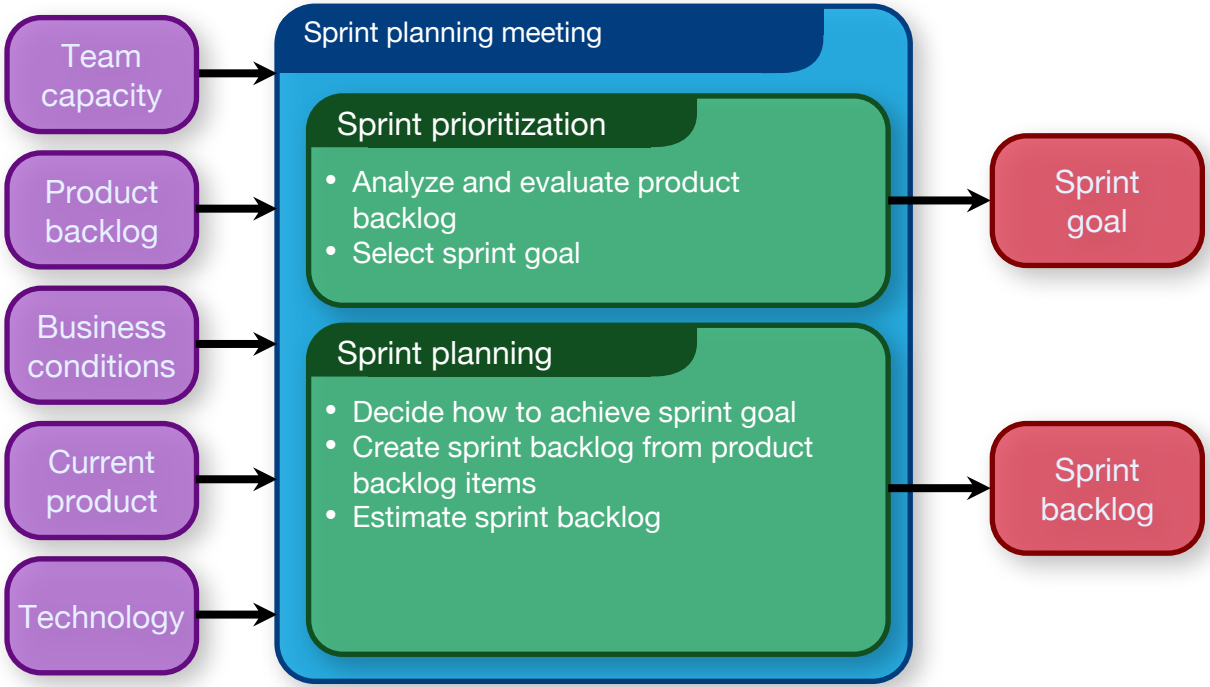
- Write the user stories in a template such as:

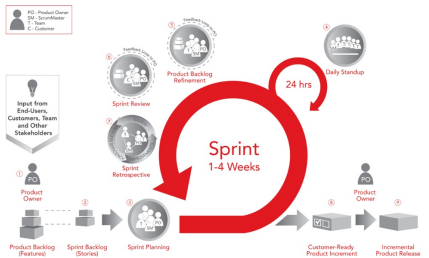
  <user role>
  need to <goal>,
  so that <reason>.

- **INVEST** in high quality user stories, i.e.:
  - > **I**ndependent
  - > **N**egotiable
  - > **V**aluable
  - > **E**stimable
  - > **S**mall
  - > **T**estable



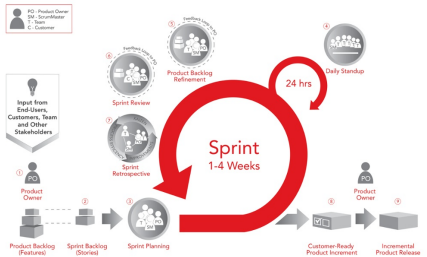
Sprint Planning





Sprint Review

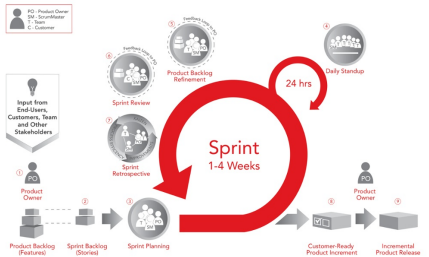
- Team presents what it accomplished during the sprint
- Typically takes the form of a **demo** of new features or underlying architecture
- Informal
  - 2-hour prep time rule
  - No slides
- Whole team participates
- Invite the world



Scrum: key practices

- Organize work in ***short cycles***.
- The management ***doesn't interrupt*** the team during a work cycle.
- The team reports to ***the client*** not the manager.
- The team estimates ***how much time*** work will take.
- The team decides ***how much work*** they can do in an iteration.
- The team decides ***how to do the work*** in the iteration.
- The team ***measures its own*** *performance*.
- Define work goals ***before*** each cycle starts.
- Define work goals through ***user stories***.
- Systematically ***remove impediments***.

Source: Steven Denning: Scrum is a Major Management Discovery. Forbes Blog. April 29, 2011.



Large Scale Scrum

- How far can we grow a Scrum team?
- What if we have multiple teams rather than just one?
- How do these teams collaborate together towards a same overarching goal?
- What concrete practices can help us do that?

... topics for another class!

M.EIC, DS 2022-23

Large Scale Software Development

U.PORTO
DEI DEPARTAMENTO DE ENGENHARIA INFORMÁTICA
INESCTEC

Goals

- Apply an **agile process**, using agile practices to the **full software development life cycle** of a **real** software system of medium–large complexity and scale.
- Use **tools to automate and support the practices** used along the lifecycle.
- Use **infrastructures and services** for large scale software systems to develop and deploy solutions with strong integration of components and applications.
- **Collaborative** work with different participants (team, customers, external experts), promoting design negotiation and participatory decision-making.
- Consolidate knowledge of other courses, in particular: Software Engineering, Software Design and Testing Laboratory, Databases, and User Interfaces.

Classes

Theoretical (T)

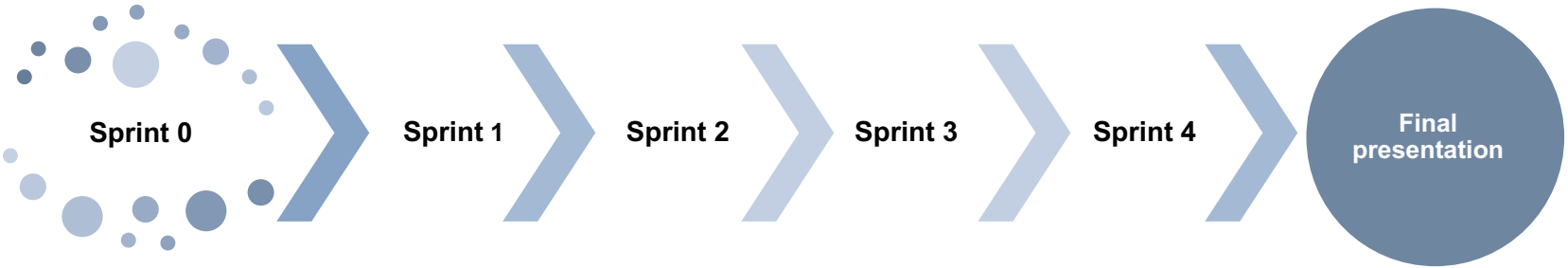
- 13 weeks x 2h = 26h
- General information about the project
- Presentation of theory (e.g., development processes, agile practices, DevOps, architecture and technology)
- Seminars with invited speakers

Theoretical-practical (TP)

- 12 weeks x 2h = 24h
- Project development

Projects

Project stages



Sprint 0 – Understand the product vision; initial tools setup and experiment with technologies.

Sprints 1-4 – Plan, develop, deliver, gather feedback on the new functionalities.

Final presentation – Final result, to be assessed from *product* and *technology* perspectives.

Continuous feedback and evaluation

Sprint 0	15%		
Sprints 1-4	45%	Sprint 1	10%
		Sprint 2	20%
		Sprint 3	30%
		Sprint 4	40%
Final outcomes	30%	Product	75%
		Presentation	25%
Individual class participation	10%		

Project-related items are evaluated at three levels:

- *Class*
- *Team*
- *Individual*

Scrum roles in the project

Dev Team

All the students in the team are part of the Dev Team.

Scrum Master (SM)

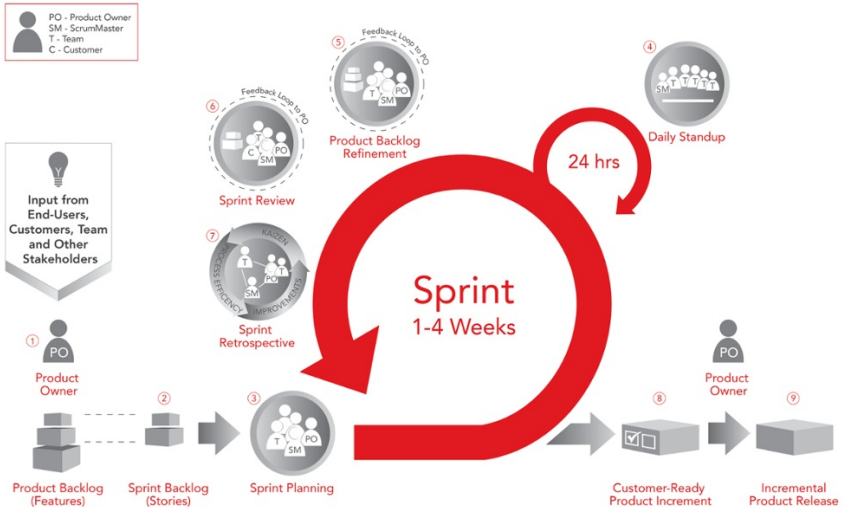
A student chosen by the team. It is not required, but often works well, to rotate this role among team members in each sprint.

Product Owner (PO)

The proponent (client) of the project.

Surrogate Product Owner (SPO)

A student chosen by the team to be in regular contact with the PO, and to make product-related decisions when the PO is not available. The SPO should not be the only team member in touch with the PO, but is his/her privileged point of contact.
Note: this one is not one of the Scrum roles, but we will use it in the context of DS.



https://www.scruminc.com/scrum-framework/

Project characteristics

Real-world project, proposed by the software industry / professionals
No two projects will be alike (they are real-world projects!)
Challenge: how to interact (and sometimes negotiate) with clients?

Large (humanly), each class working together on the same project
Around 25 students per project, organized in 4 teams per class
Challenge: how to collaborate effectively and efficiently with so many?

Project proposals

P1: **Ecommerce Logistics API and Applications**, for Maersk
1MIEIC1, supervised by João Pedro Dias

P2: **Circular Economy for the Built Environment Platform**, for Built COLAB
1MIEIC2, supervised by Daniel Pinho

P3: **Shift Planner**, for Centro Hospitalar do Médio Ave
1MIEIC3, supervised by João Pedro Dias

P4: **Case Management for Finance**, for LOQR
1MIEIC4, supervised by Filipe Correia

P5: **Dynamic Pricing Strategies for admira**, for B6 Software Solutions
1MIEIC5, supervised by Daniel Pinho

P6: **HotSquare**, for DevScope
1MIEIC6, supervised by Filipe Correia

In the next practical classes

- Define constitution of the teams (done by the professors)
- Hands-on work on Sprint 0
- Kickoff meeting with PO (will be invited by professors to be present in a class)

Sprint 0

Sprint 0 (4 weeks)		15%
Setup	Git repository and board are created (GitHub). Project docs created and available on the repository(ies) according to templates. README.md on the root of the repository. CHANGELOG.md on the root of the repository. The SPO and SM roles are assigned to team members. There is a "factsheets/teamX.md" for each team. There is a "factsheets/firstname_lastname.md" for each team member. Team collaboration tools setup (e.g., slack, discord).	5%
Initial Product Vison	Kickoff meeting with PO. Product vision refined. Market research (survey of similar projects and comparative analysis). Domain analysis (includes high-level class diagram with key domain concepts).	25%
Product Backlog	Scrum Board has draft of Product Backlog Items (PBIs) ready for Sprint 1. PBIs have mockups (when applicable). PBIs have acceptance tests (not automated yet) covering key usage scenarios. PBIs validated by the PO (including stories, mockups, and acceptance tests). PBIs ordered by the PO.	25%
Initial Technical Vision	First architectural definition (component/deployment diagrams & using packages if needed). Identify main risks and justify choices showing the soundness of the technical vision. Choose development tools/environments/languages/frameworks. Learn and practice the technologies to be used; use what you learn to justify your choices. Development of a base implementation and guidelines to to be used by the different teams.	45%

Tools and technologies

- Will depend on the concrete needs of each project, but...
- We will use **GitHub** for version control and project management
- We will favor the use of **containers** (e.g., Docker and Docker Compose)
- Automation of ***unit* and *acceptance* tests** will need appropriate tools
- Linters, and other **static analysis tools**, including security analysis tools
- Continuous integration **pipelines** (e.g., GitHub actions, others)
- Application **telemetry** (e.g., exception tracking, real-time monitoring)

Homework

- Learn
  - Read the scrum guide (see link in moodle).
  - Install and learn the essentials of using containers (see resources in moodle).
- Project logistics
  - Provide your GitHub username (link to spreadsheet in moodle)
- Hands-on Sprint 0, where to start
  - Study the list of items of Sprint 0 (slide #49) and bring any questions to next class
  - Study de description for your project and ...
    - > Search for similar projects and differentiators.
    - > Collect a list of doubts to clarify with your client at the first opportunity.
    - > Sketch your own understanding of the project as a *domain model*, using a class diagram.

FEUP.DEI

DEI - DEPARTAMENTO DE ENGENHARIA INFORMÁTICA
dei.fe.up.pt | secdei@fe.up.pt | +351 225 082 134
Director **João MP Cardoso** | Full Professor
jmpc@fe.up.pt | +351 220 413 284

FEUP - FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO
Rua Dr. Roberto Frias | 4200-465 Porto | PORTUGAL
www.fe.up.pt | feup@fe.up.pt | +351 225 081 400
@DEI_FEUP

DEI DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

U.PORTO