OCTOBER 28, 2019

SKILLS DEMONSTRATION 2

SOFTWARE ARCHITECTURE

TAHA AFLOUK
5N2772

Table of Contents

1 – History and Evolution of Agile

First came the crisis

In the early 1990s, as PC computing began to proliferate in the enterprise, software development faced a crisis. At the time, it was widely referred to as "the application development crisis," or "application delivery lag." Industry experts estimated that the time between a validated business need and an actual application in production was about three years.

The problem was, businesses moved faster than that, even 25 years ago. Within the space of three years, requirements, systems, and even entire businesses were likely to change. That meant that many projects ended up being cancelled partway through, and many of those that were completed didn't meet all the business's current needs, even if the project's original objectives were met.

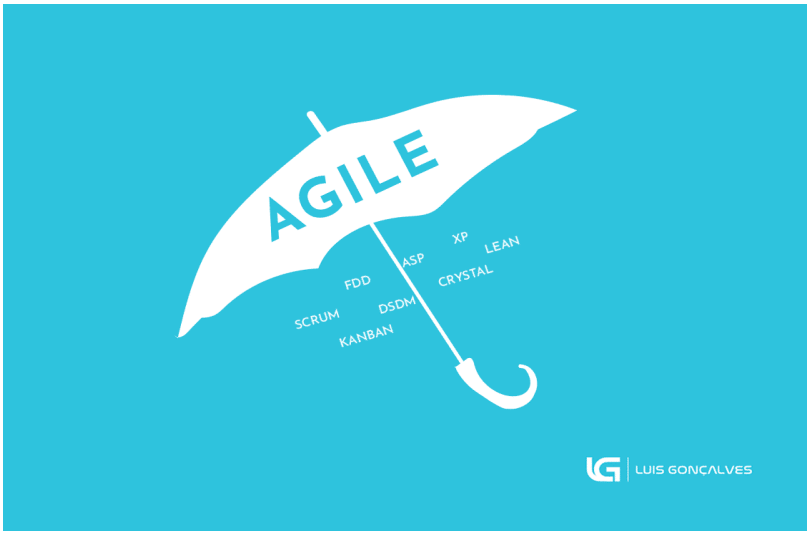
In certain industries, the lag was far greater than three years. In aerospace and defense, it could be 20 or more years before a complex system went into actual use. In an extreme but by no means unusual example, the Space Shuttle program, which operationally launched in 1982, used information and processing technologies from the 1960s. Highly complicated hardware and software systems were often designed, developed, and deployed in a time frame that spanned decades.

1.1 – What is Agile?

In 2001, a small group of people, tired of the traditional approach to managing software development projects, designed the agile manifesto. It is a more improved method for managing the progress of software projects.

1.2 - Where did Agile come from?

In 2001, a small group of people, tired of the traditional approach to managing software development projects, designed the agile manifesto. It is a more improved method for managing the progress of software projects.



“Agile is a time boxed, iterative approach to software delivery that builds software incrementally from the start of the project, instead of trying to deliver it all at once near the end.”

1.3 - Reason for its development

Here are 10 good reasons to apply agile development principles and practices…

The iterative nature of agile development means features are delivered incrementally, enabling some benefits to be realized early as the product continues to develop.

Research suggests about 80% of all market leaders were first to market. As well as the higher revenue from incremental delivery, agile development philosophy also supports the notion of early and regular releases, and ‘perpetual beta’.

A key principle of agile development is that testing is integrated throughout the lifecycle, enabling regular inspection of the working product as it develops. This allows the product owner to adjust if necessary and gives the product team early sight of any quality issues.

Agile development principles encourage active ‘user’ involvement throughout the product’s development and a very cooperative collaborative approach. This provides excellent visibility for key stakeholders, both of the project’s progress and of the product itself, which in turn helps to ensure that expectations are effectively managed.

Small incremental releases made visible to the product owner and product team through its development help to identify any issues early and make it easier to respond to change. The clear visibility in agile development helps to ensure that any necessary decisions can be taken at the earliest possible opportunity, while there’s still time to make a material difference to the outcome.

In traditional development projects, we write a big spec up-front and then tell business owners how expensive it is to change anything, particularly as the project goes on. In fear of scope creep and a never-ending project, we resist changes and put people through a change control committee to keep them to the essential minimum. Agile development principles are different. In agile development, change is accepted. In fact, it's expected. Because the one thing that's certain in life is change. Instead the timescale is fixed and requirements emerge and evolve as the product is developed. Of course for this to work, it's imperative to have an actively involved stakeholder who understands this concept and makes the necessary trade-off decisions, trading existing scope for new.

The above approach of fixed timescales and evolving requirements enables a fixed budget. The scope of the product and its features are variable, rather than the cost.

The active involvement of a user representative and/or product owner, the high visibility of the product and progress, and the flexibility to change when change is needed, create much better business engagement and customer satisfaction. This is an important benefit that can create much more positive and enduring working relationships.

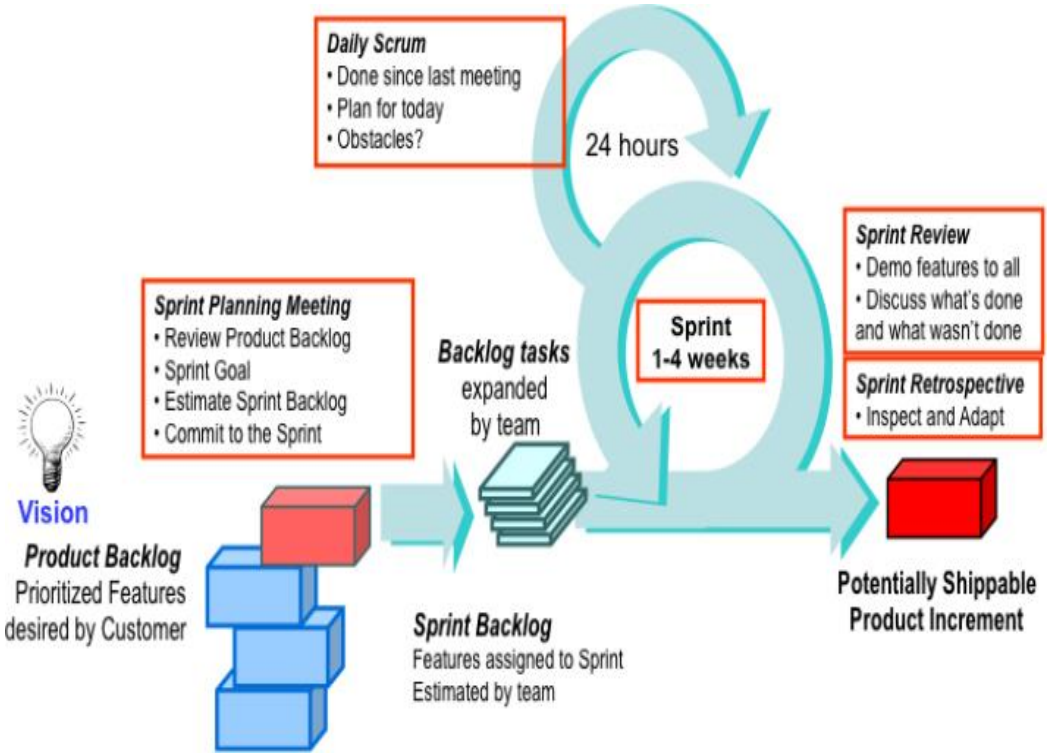
Above all other points, the ability for agile development requirements to emerge and evolve, and the ability to embrace change (with the appropriate trade-offs), the team build the right product. It's all too common in more traditional projects to deliver a "successful" project in IT terms and find that the product is not what was expected, needed or hoped for. In agile development, the emphasis is absolutely on building the right product.

The active involvement, cooperation and collaboration make agile development teams a much more enjoyable place for most people. Instead of big specs, we discuss requirements in workshops. Instead of lengthy status reports, we collaborate around a task-board discussing progress. Instead of long project plans and change management committees, we discuss what's right for the product and project and the team is empowered to make decisions. In my experience this makes it a much more rewarding approach for everyone. In turn this helps to create highly motivated, high performance teams that are highly cooperative.

1.4 – Other Models

Scrum

“Scrum is an agile way to manage a project, usually software development. Agile software development with **Scrum** is often perceived as a **methodology**; but rather than viewing **Scrum** as **methodology**, think of it as a framework for managing a process.”



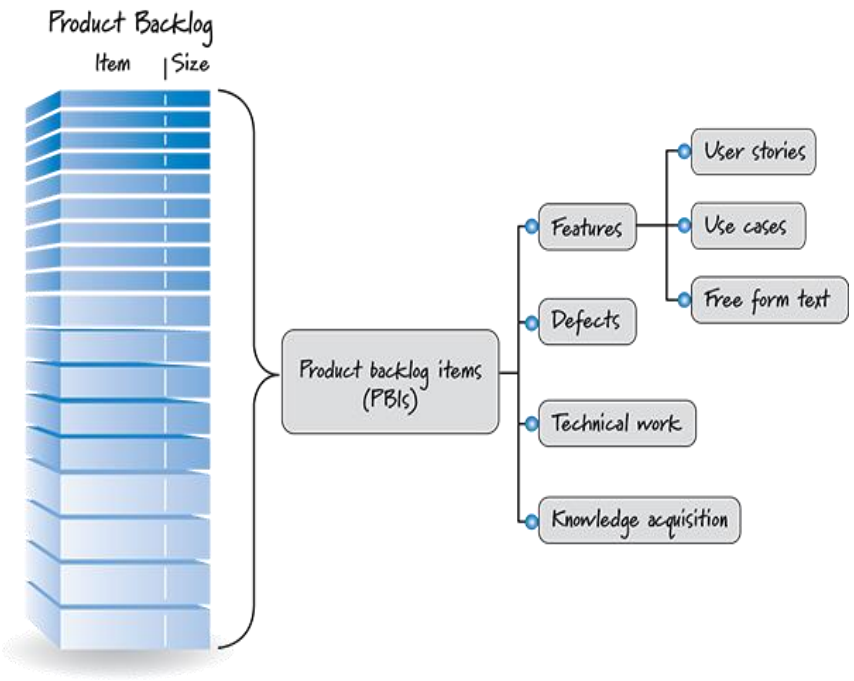
- Gathering Requirement
- Team Roles
- Release Planning
- Sprints
- Burndown chart

Gathering Requirements-1

Get requirements from the customer and put them in this format. This called a user story.

- As a [who]
- I want [what]
- Because [why]

Gathering Requirements-2



Team Roles

Product owner:

Picks features

Represents the customer

Scrum master:

Sets up meetings

Make sure everything is working smoothly

Release planning

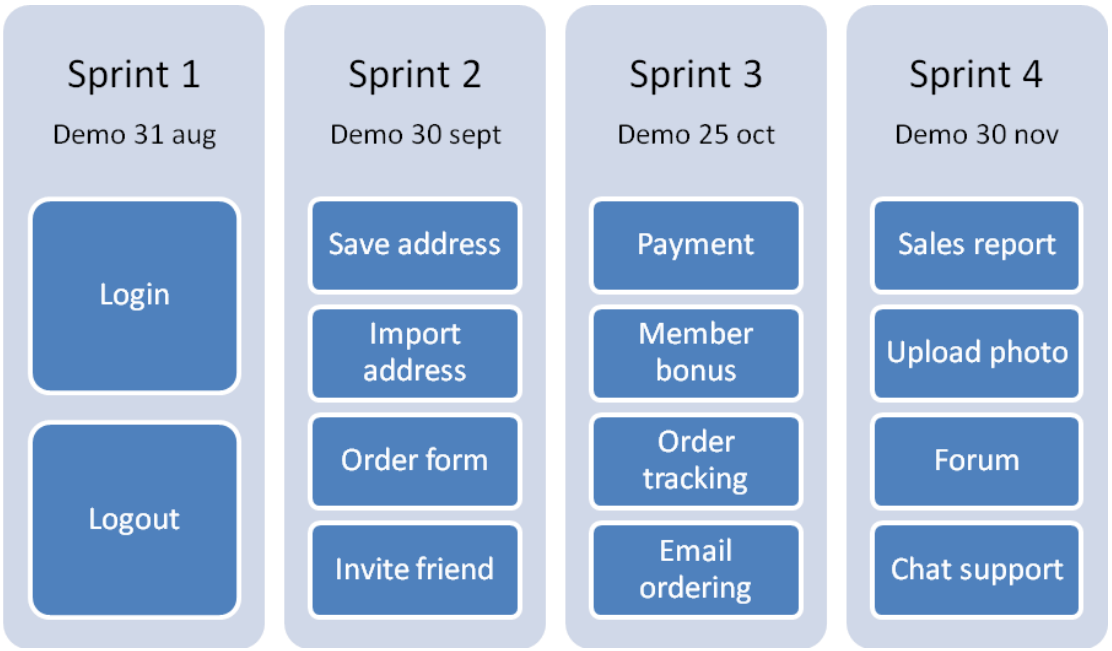
Pick user stories for next release

Prioritize

Estimate (and possibly breakup

Now you have a release backlog and a time estimate of how work to do.

Sprints-1



Sprints-2

You work on some stories and get them to a ship-ready state (100% complete).

This includes doing all planning, development and testing needed for the user stories.

Sprints-3

Daily scrum meeting

Quick 15-minute (standing) meeting everyday:

What did you do since last meeting?

Are you facing any problems?

Sprints-4

Sprint retrospective

A meeting after a sprint is done:

What was good?

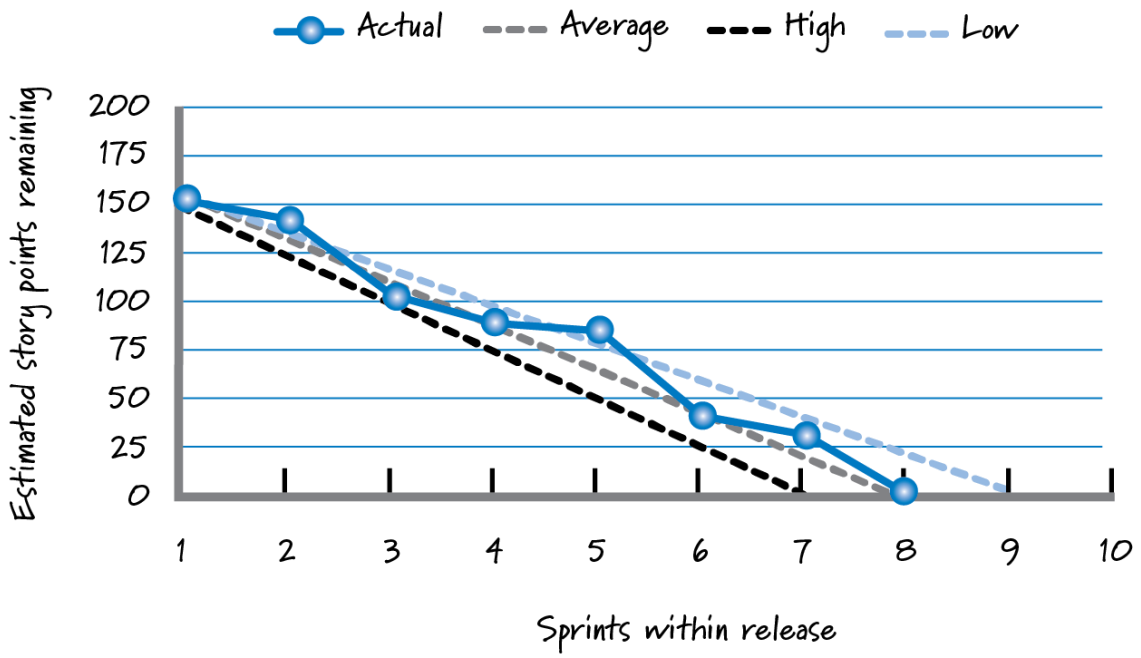
What was bad?

How to improve?

Burndown chart

Using the time estimate, we know the amount of work in the sprint (or release).

Update this value every day and plot it on the chart. It helps you know if you are on track or late.



Tools

What tools can I use to watch the scrum:

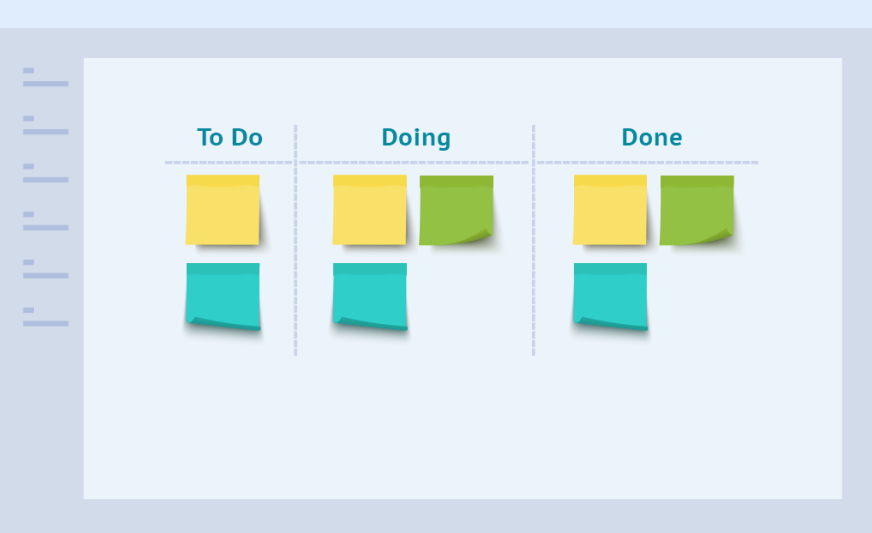
Trello is simple and free to use

Trello

https://trello.com

Kanban

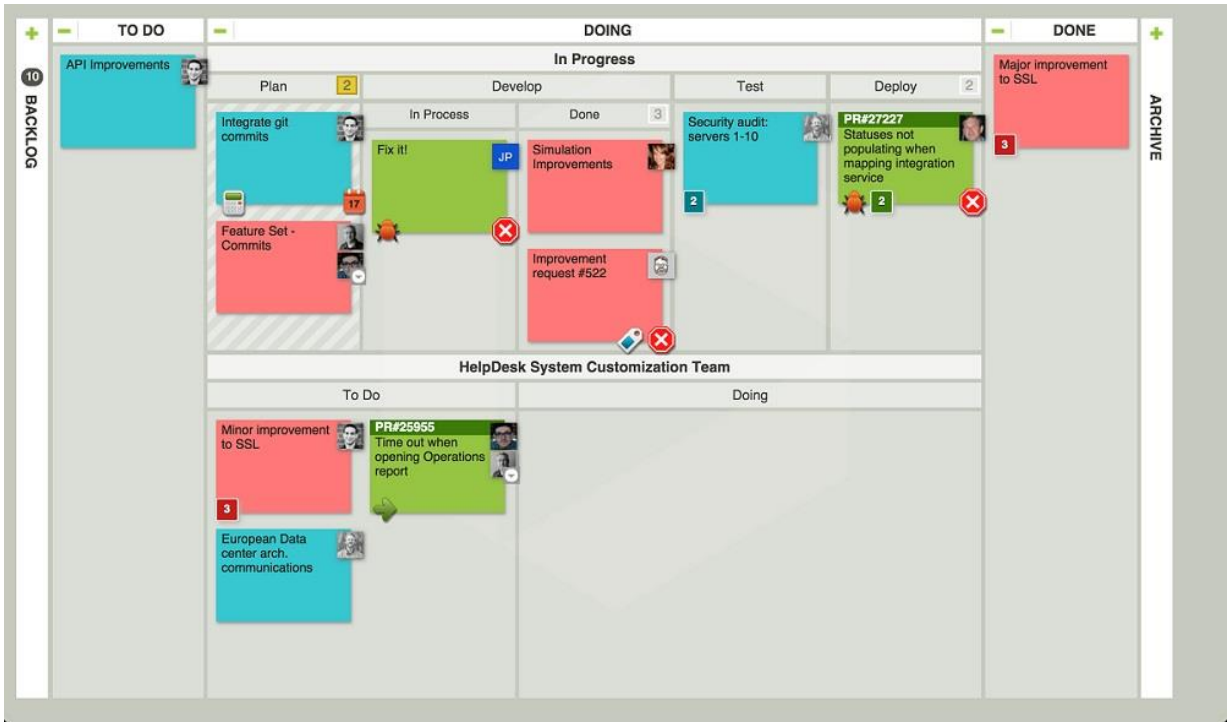
“Kanban is a visual system for managing work as it moves through a process. Kanban visualizes both the process (the workflow) and the actual work passing through that process. The goal of Kanban is to identify potential bottlenecks in your process and fix them so work can flow through it cost-effectively at an optimal speed or throughput.”



“An online Kanban board is a tool that helps visualize work and workflow, as well as optimize the way work gets done. Then name comes from the Japanese word Kanban, meaning “visual signal” or “card,” and references the process improvement approach known as the Kanban Method.”

How an Online Kanban Board Works

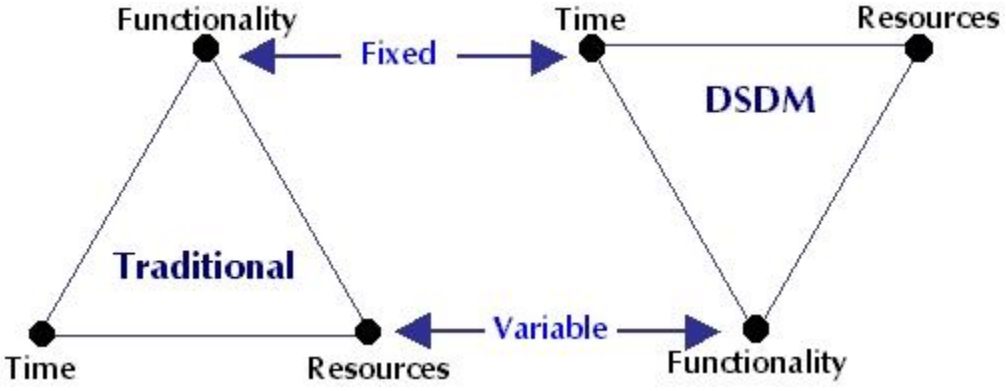
“Like a physical Kanban board, a Kanban tool uses a series of vertical and horizontal lanes to represent workflow or process (i.e. the steps that work takes to advance from start to finish). Kanban cards, representing tasks and work items, are moved through the lanes to reflect progress, using drag-and-drop functionality.”



DSDM

“Dynamic Systems Development Method (DSDM) is an organized, common-sense process focused on delivering business solutions quickly and efficiently. It is similar in many ways to SCRUM and XP, but it has its best uses where the time requirement is fixed.

DSDM focuses on delivery of the business solution, rather than just team activity. It makes steps to ensure the feasibility and business sense of a project before it is created. It stresses cooperation and collaboration between all interested parties. DSDM makes heavy use of prototyping to make sure interested parties have a clear picture of all aspects of the system.”



DSDM in more detail

“DSDM has been developed to address common problems faced by projects such as late delivery, cost overruns or the final deliverable not being completely fit for purpose.

DSDM addresses these problems by creating an agile environment which is collaborative and flexible yet remaining focused on hitting deadlines and maintaining the appropriate level of quality and rigour.

DSDM involves all stakeholders such as the business representatives throughout an iterative and incremental lifecycle.

All personnel involved in a project are given clear roles and responsibilities and work together in timeboxes to ensure the project is kept on schedule.

DSDM strikes the balance between performing a lot of ‘up-front’ design and performing none. DSDM believes in doing ‘enough design up-front’ in order to reduce risk yet still allow for the inevitability of change.

DSDM is particularly good at being used with other approaches. Two prominent examples of this are using DSDM with PRINCE2 which enables PRINCE2 to be easily run with an agile setting and running DSDM with Scrum which enables Scrum to be scaled up to run on more than just product development.”

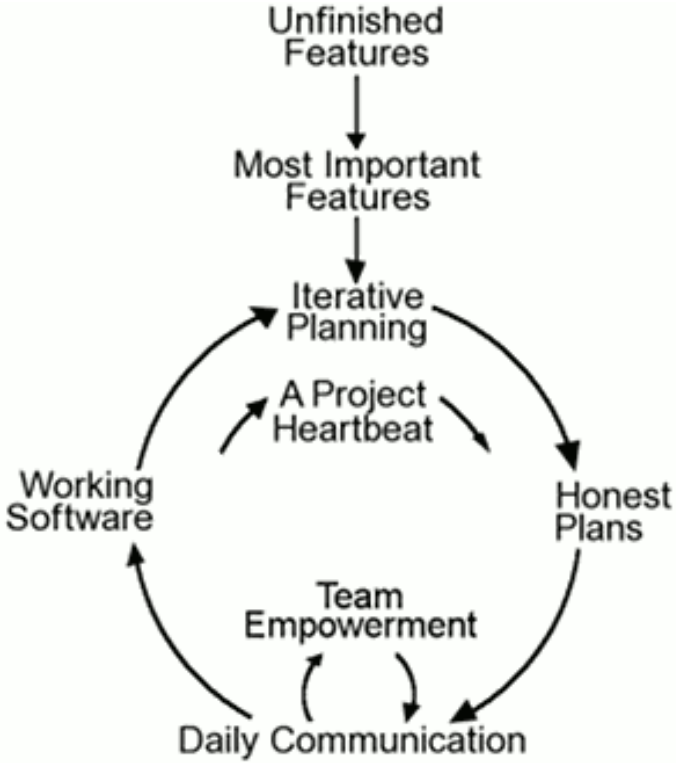
The Eight Principles of DSDM

DSDM has eight principles. They represent an ethos, a culture, a way of working. The principles are always actively managed, because if a principle becomes compromised it represents a risk to the successful execution and completion of a project.

The eight Principles of DSDM are as follows:

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

Extreme programing (XP)



“The first Extreme Programming project was started March 6, 1996. Extreme Programming is one of several popular Agile Processes. It has already been proven to be very successful at many companies of all different sizes and industries worldwide. Extreme Programming is successful because it stresses customer satisfaction. Instead of delivering everything you could possibly want on some date far in the future this process delivers the software you need as you need it. Extreme Programming empowers your developers to confidently respond to changing customer requirements, even late in the life cycle. Extreme Programming emphasizes teamwork. Managers, customers, and developers are all equal partners in a collaborative team. Extreme Programming implements a simple, yet effective environment enabling teams to become highly productive. The team self-organizes around the problem to solve it as efficiently as possible.

Extreme Programming improves a software project in five essential ways; communication, simplicity, feedback, respect, and courage. Extreme Programmers constantly communicate with their customers and fellow programmers. They keep their design simple and clean. They get feedback by testing their software starting on day one. They deliver the system to the customers as early as possible and implement changes as suggested. Every small success deepens their respect for the unique contributions of each team member. With this foundation Extreme Programmers can courageously respond to changing requirements and technology.

The most surprising aspect of Extreme Programming is its simple rules. Extreme Programming is a lot like a jigsaw puzzle. There are many small pieces. Individually the pieces make no sense, but when combined a complete picture can be seen. The rules may seem awkward and perhaps even naive at first but are based on sound values and principles.
Our rules set expectations between team members but are not the end goal themselves. You will come to realize these rules define an environment that promotes team collaboration and empowerment, that is your goal. Once achieved productive teamwork will continue even as rules are changed to fit your company's specific needs. This flow chart shows how Extreme Programming's rules work together. Customers enjoy being partners in the software process, developers actively contribute regardless of experience level, and managers concentrate on communication and relationships. Unproductive activities have been trimmed to reduce costs and frustration of everyone involved. Take a guided tour of Extreme Programming by following the trail of little buttons, starting here."

2.0 – Agile Manifesto

2.1 – Contributors

"The document, formally called the "Manifesto for Agile Software Development," was produced by 17 developers during an outing on Feb. 11-13, 2001, at The Lodge at Snowbird ski resort in Utah.

The developers, who called themselves the Agile Alliance, were seeking an overhaul of the software development processes that they saw as cumbersome, unresponsive and too focused on documentation requirements.

According to agilemanisfesto.org, the online home of the proclamation, the developers' stated goal was not anti-methodology, but rather "to restore credibility to the word methodology."

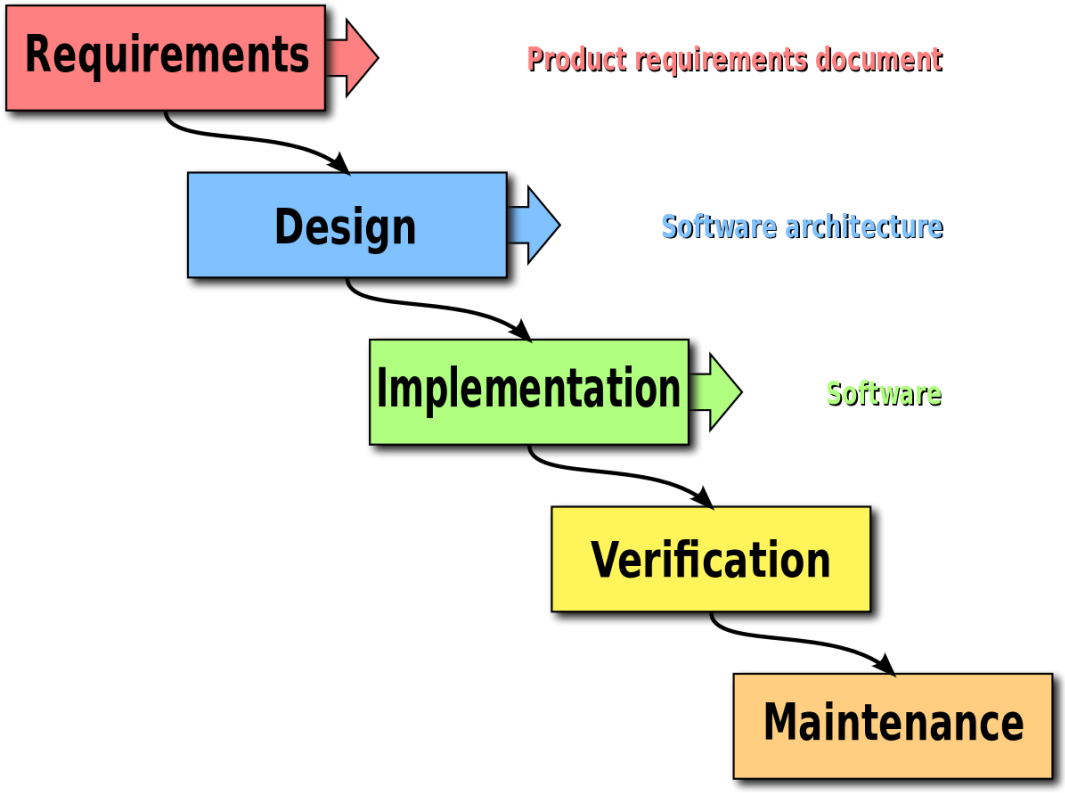
It further states: "We want to restore a balance. We embrace modeling, but not in order to file some diagram in a dusty corporate repository. We embrace documentation, but not hundreds of pages of never maintained and rarely used tomes. We plan but recognize the limits of planning in a turbulent environment."

2.2 – Principals

The 12 principles articulated in the Agile Manifesto are:

1. Satisfying customers through early and continuous delivery of valuable work.
2. Breaking big work down into smaller tasks that can be completed quickly.
3. Recognizing that the best work emerges from self-organized teams.
4. Providing motivated individuals with the environment and support they need and trusting them to get the job done.
5. Creating processes that promote sustainable efforts.
6. Maintaining a constant pace for completed work.
7. Welcoming changing requirements, even late in a project.
8. Assembling the project team and business owners on a daily basis throughout the project.
9. Having the team reflect at regular intervals on how to become more effective, then tuning and adjusting behaviour accordingly.
10. Measuring progress by the amount of completed work.
11. Continually seeking excellence.
12. Harnessing change for a competitive advantage.

3.0 – Agile vs Waterfall



“Waterfall Model is a sequential model that divides software development into different phases. Each phase is designed for performing specific activity during SDLC phase. It was introduced in 1970 by Winston Royce.

3.1 – Advantages

The advantages of the Agile methodology

The Agile methodology was firstly developed for the software industry. The task was to optimize and improve the development process and to try to identify and quickly correct problems and defects. This methodology allows to provide **a** better output, more quickly, through short and interactive sessions / sprints.

In the era of digital transformation, where many organizations are migrating to a digital workplace, the Agile methodology suits perfectly in companies that are looking to transform the way in which projects are managed and the way they operate.

If we consider the benefits for the company, the digital workplace and the Agile methodology provide:

- More flexibility;
- More productivity;
- More transparency;
- Products of superior quality;
- Decreased risk of missed goals;
- Greater involvement and satisfaction of stakeholders.

In the field of project management, the Agile methodology gives teams, sponsors, project managers and customers many specific advantages, including:

- Faster implementation of solutions;
- Waste reduction thanks to the minimization of the resources;
- Greater flexibility and adaptability to change;
- More success thanks to more focused efforts;
- Faster delivery times;
- Faster detection of problems and defects;
- Optimized development processes;
- A lighter/less complicated structure;
- Excellent project control;
- Greater attention to specific customer needs;
- Increased collaboration frequency and feedback.

3.2 – Disadvantages

The disadvantages of Agile

As with any other methodology, even the Agile approach is not suitable for any project. It is therefore recommended to do an adequate analysis in order to identify the best methodology to apply in every situation. Agile may not work as expected, for example, if a client is not clear about the goals, if the project manager or the team has no experience or if they do not “work well” under pressure.

Because the Agile methodology has less formal and more flexible processes, it may not always be easily included into larger and more traditional organizations.

Here, in fact, processes, policies or teams could be rigid**.** The Agile methodology is also difficult to implement when clients follow rigid processes or methods**.** Furthermore, given that this methodology focuses mainly on the short term, the risk that the long-term vision will be lost does exist.

At this point it is appropriate to make some small considerations according to the fact that the PmBok, i.e. the bible of Project Manager, is mainly based on the so-called “Waterfall” approach – which explains a sequential development in phases, in the life cycle of the project.

In some of these phases, the PmBok contemplates possible application of an Agile approach, if this is in line with the goals of the project.

The advantages of the Waterfall approach are:

- Defined, agreed and formalized requirements;
- Possible defects or risks are already assessed in the initial phases of the project;
- Detailed and punctual documentation;
- Due to the detailed project documentation, even non-expert colleagues can manage the project.

On the other hand, the disadvantages of this approach are the following:

- Analysis and planning activities can take a long time and thus delay the actual launch of the project;
- The requirements, as soon as they are formalized, can only be modified through another process, which – again – takes time;
- During project development, new needs or new tools may arise that can require more flexibility.

The Agile methodology focuses mostly on optimizing the process. The PmBok, and therefore the Waterfall method, focuses more on managing goals and risks and on forecasting and controlling costs.

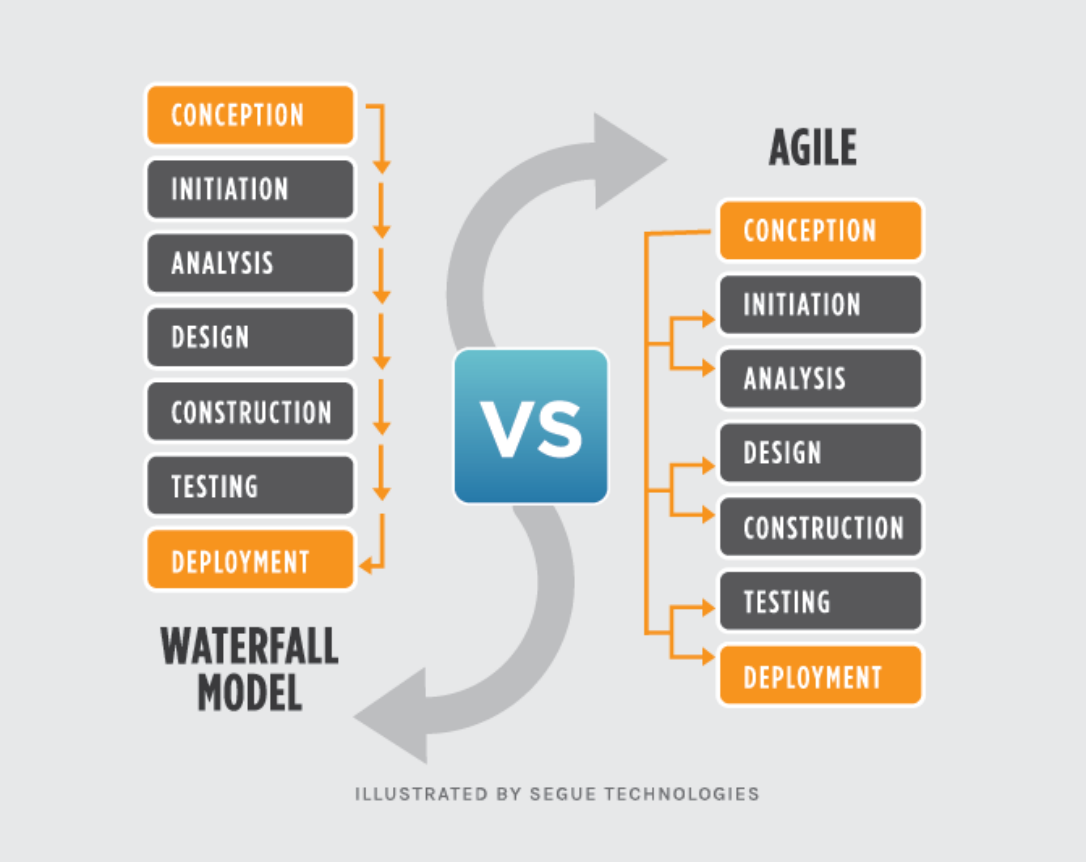
An Agile approach works at its best in situations that have a relatively high level of uncertainty, where creativity and innovation in order to find the appropriate solution are more important than predictability. A very simple and clear example is the research for a cure for cancer. In this case, for instance, it would be ridiculous to develop a detailed plan on the strategy to follow.

A traditional approach, such as Waterfall, works well in situations that have a relatively low level of uncertainty and where predictability, planning and control are essential.

Here the best example can be the building of a bridge that must always follow the same system. Many project managers have seen – and still see – these two approaches as competitive with each other. A high level of skill is needed in order to see these two approaches in a new perspective, as complementary to each other. “In fact, both methodologies are valid, but require a great interpretative capacity – beyond experience – in order to apply the correct principles in every situation. In the development of the Twproject software, we came to a very important consideration.

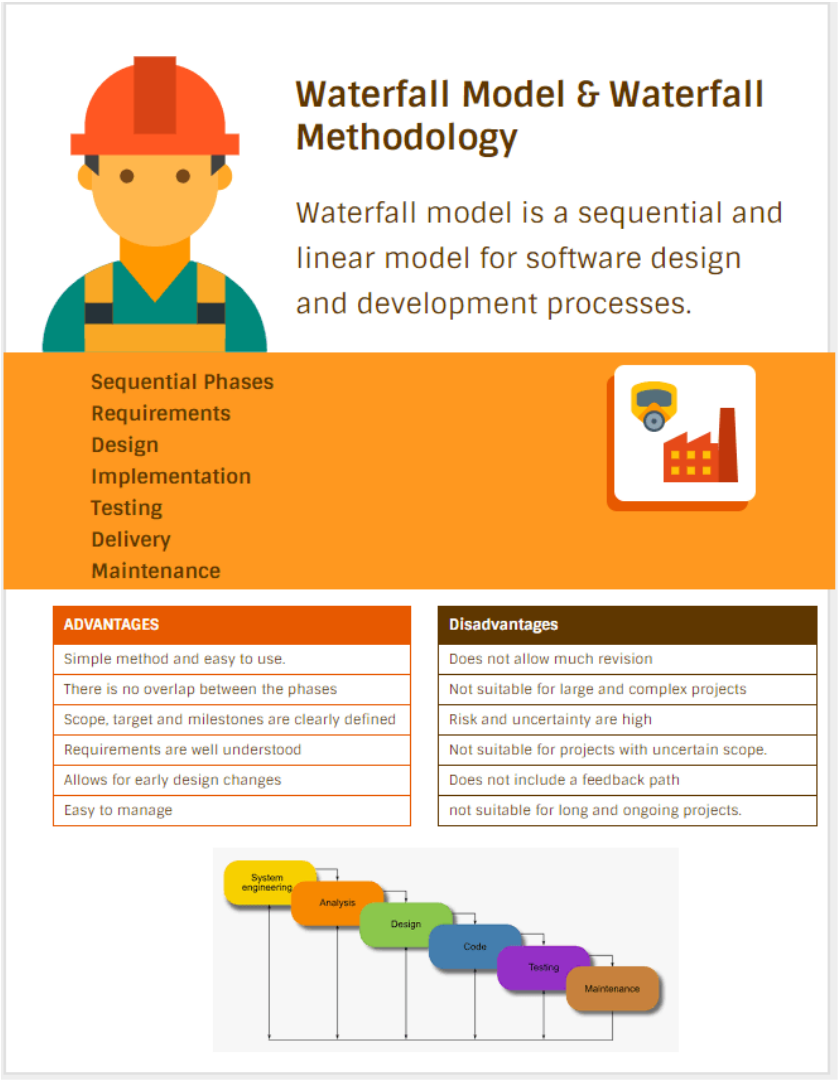
Approaches can help to solve certain classes of problems, but they will never cover all the work activities of a company. Therefore, it would be extremely non-agile to have a specific software for “agile” projects, and one for others. And even “agile” projects can present many variations, which will fit into the agile metaphor at different stages, and hardly in a single “software model”. Therefore, we have reached a basic assumption: agility is in the methodology, not in software.

A software should be flexible enough to let you map projects, tasks, issues, to people and customers, in endless ways, but so that all data from different projects and methodologies are collected in the same place.”



“The disadvantages of the waterfall model typically surround risk associated with a lack of revision, including:

- Design is not adaptive; often when a flaw is found, the entire process needs to start over.
- Ignores the potential to receive mid-process user or client feedback and make changes based on results.
- Delays testing until the end of the development life cycle.
- Does not consider error correction.
- Does not handle requests for changes, scope adjustments or updates well.
- Reduces efficiency by not allowing processes to overlap.
- No working product is available until the later stages of the life cycle.
- Not ideal for complex, high risk, ongoing or object-oriented projects.”



Bibliography

Waters, K. (2019). *10 Good Reasons To Do Agile Development* | *101 Ways*. [online] 101 Ways. Available at: https://www.101ways.com/2007/06/11/10-good-reasons-to-do-agile-development/ [Accessed 13 Dec. 2019].

Digite. (2019). *What Is Kanban? An Overview Of The Kanban Method*. [online] Available at: https://www.digite.com/kanban/what-is-kanban/ [Accessed 14 Dec. 2019].

Gonçalves, L. (2019). What Is Agile Methodology. [online] luis-goncalves. Available at: https://luis-goncalves.com/what-is-agile-methodology/ [Accessed 14 Dec. 2019].

Rouse, M. (2019). *What is Agile Manifesto? - Definition from WhatIs.com*. [online] SearchCIO. Available at: https://searchcio.techtarget.com/definition/Agile-Manifesto [Accessed 14 Dec. 2019].

Staff, T. (2019). *Agile methodology: advantages and disadvantages of an innovative method*. [online] Twproject: project management software, bug tracking, time tracking, planning. Available at: https://twproject.com/blog/agile-methodology-advantages-disadvantages-innovative-method/ [Accessed 14 Dec. 2019].

TechBeacon. (2019). *The complete history of agile software development*. [online] Available at: https://techbeacon.com/app-dev-testing/agility-beyond-history-legacy-agile-development [Accessed 15 Dec. 2019].

Youtube.com. (2019). *The 4 Values of Agile Manifesto Explained!*. [online] Available at: https://www.youtube.com/watch?v=gf7pBZxOCtY&t=7s [Accessed 15 Dec. 2019].

Terry, J. (2019). *What is an Online Kanban Board?* | *Planview LeanKit*. [online] Planview. Available at: https://www.planview.com/resources/articles/what-is-an-online-kanban-board/ [Accessed 15 Dec. 2019].

Agilebusiness.org. (2019). *What is DSDM?*. [online] Available at: https://www.agilebusiness.org/page/whatisdsdm [Accessed 15 Dec. 2019].

Extremeprogramming.org. (2019). *Extreme Programming: A Gentle Introduction.*. [online] Available at: http://www.extremeprogramming.org/ [Accessed 15 Dec. 2019].