Appendix Project management

This appendix contains additional information on project management. These are un edited versions of earlier more elaborate chapters on either traditional- and agile project management.

Table of Contents

1 Traditional project management

This appendix chapter is an information dump on traditional project management and is the elaborated version of traditional project management.

1.1 *Traditional project management theory and principles*

Koskela, L. & Howell, (2002) argue that project uses narrow develop theory that should be broadened. In their paper they deduce that scope management is the reason project management exists. Project management theory focusses on planning and little on executing of the project. The execution is the communication of what needs to be done, any decisions that need to be made are largely taken care of in the planning phase. The communication is mainly one-way, from sender to receiver. Koskela, L. & Howell, in their paper found that TPM is based on three management theories; Mamanagement-as planning, dispatching model and the thermostat model (Fernandez & Fernandez, 2008; Koskela, L. & Howell, 2002; Koskela & Howell, 2002). According to Koskela, L. & Howell, (2002) Project management theory is focussed on the planning processes. Stoica, Mircea, & Ghilic-Micu, (2013) state that TPM's fundamental hypothesis is that; *"Systems are fully specifiable, predictable and are developed through extended and detailed planning".* Turner, (2009) defined five principles of TPM, which are:

- Mange through a structured breakdown, with a single point of responsibility;
- Focus on results;
- Balance results with the use of the breakdown structure;
- Organize the project through contract negotiations;
- Use a simple and clear reporting (Turner, 2009).

According to Fernandez, D.J. & Fernandez, (2009) the traditional approach uses a linear strategy without a feedback-loop. In this strategy it is assumed that projects have a clearly defined scope and changes are minimal (Fernandez & Fernandez, 2008; Owen, Koskela, Henrich, & Codinhoto, 2006; Špundak, 2014). The idea is that projects are simple and linear (Fernandez & Fernandez, 2008; Špundak, 2014). TPM assumes that a project is predictable (Verbruggen, 2017). The scope of the project can be clearly defined, which makes it easy to plan and follow the plan without making much changes. TPM methods should be applied to projects that have a clear goal and solution, where the goal and solution are well defined and understood (Nicholas & Steyn, 2017).The goal of TPM is to optimize and effective complete the detailed project plan in order to finish the project on time, within budget and fulfil the layout scope (Munns A. K. & Bjeirmi B. F., 1996; Špundak, 2014).

In TPM the project phases are completed in sequence (Nicholas & Steyn, 2017). This means project planning need to be done up-front because once a project phases is completed it is assumed that that phase is not revisited (Hass, 2007). These are often called stage-gates. Stage gates are assessment moments when the stage or phase of the project is assessed of it can move to the next stage or not. This mean the phase needs to be complete before the next phase can begin (Turner, 2009). Because of this TPM is nicknamed the waterfall approach (Hass, 2007). Any changes to the plan are expected to be resolved with the goal to get back on the planning (Owen et al., 2006). Therefore, TPM is sometimes referred to as a plan-driven approach (Verbruggen, 2017).Popular traditional project management methods used are PMBOK and Prince2 (Ghosh, Forrest, Dinetta, Wolfe, & Lambert, 2012). In the Netherlands the most methods are “projectmatig creeeren”, “projectmatig werken” and “Prince2” (de Jong, 2018). According to Joslin & Müller, (2015) these methods were developed in order to better project results and to predict project success.

1.2 Traditional project management the basics

Traditional projects make use of the iron triangle, where time, money and scope make up the points of the triangle and together define quality. The thought is that when one of the points of the triangle changes it will impact the others. The triangle is used to measure project success (Atkinson, 1999). Khan, (2010) identifies scope as the most important to manage because it covers the other key areas as cost, time and quality. In TPM the scope is fixed and time and resources are presumed flexible (Owen et al., 2006; Verbruggen, 2017). These criteria are decided early on in the project when these factors are still in development (Atkinson, 1999).

Scope creep is the refers to the unwanted growth and change of project requirements which lead to project being over budget and schedule. The project needs to be controlled because project have the tendency to change add specification to the scope, this increase time and cost (Nicholas & Steyn, 2017). Gerver, (2014) states *“From early phases onwards it is critical to manage project changes adequately to avoid scope creep, schedule delays and cost increase”.* It is consided that change is inefitatable (Lycett, Rassau, & Danson, 2004; Sohi, 2018).

Managing the unwanted growth is done through Scope management, which has three purposes, ensuring sufficient amount of work is completed, unnecessary work is not done and the work that is completed fulfils its business purpose. This scope is defined through the use of a work breakdown structure (Zandhuis & Wuttke, 2019). According to Nicholas & Steyn, (2017) meetings main function is to identify scope creep and quickly correct them in order to get back to the plan.

The work-breakdown structure (WBS) is used in TPM to establish a framework for the work and activities that need to be completed to realize the project (Globerson, 1994). And is used to define and control the project scope, isolate changes and isolate risks (Turner, 2009). A project consists of small sub-project/tasks/elements, these are depicted in a WBS. The WBS contains all the work that needs to be accomplished, and is often divided in different levels, each level has another level of detail. Depicting the work from sub-project level all the way to elements and activities that need to completed (Nicholas & Steyn, 2017). The work that is defined in the WBS used to established the planning. The WBS is the core of project management because the planning and control process are based on the WBS (Turner, 2009).

The requirements used to define the work that needs to be completed have the need to be clear (Špundak, 2014). A guideline often used is that the requirement should fulfil the acronym SMART. Which stands for specific; measurable; attainable; realistic and time-bound (Wasson, 2015).

Planning in TPM is done through the use of the Gannt chart. This commonly used scheduling technique is a bar chart that displays WBS and other planning information on a time-scale. Relationship between activities are often displayed, the chart shows the sequences of task that need to be completed. The Gannt chart gives the user also the possibility of measuring project progress by colouring in the activity bars. (Gerver, 2014)

The Gantt chart is often used in combination with the critical path method (CPM) and Program evaluation and review techniques (PERT) (H. L. M. Bakker & Kleijn, 2014; Nicholas & Steyn, 2017). PERT is a probabilistic method, that estimates the pessimistic, optimistic and most likely durations of a project. CPM is a method that determines the critical path and thus the critical activities. Critical activities are the activities that when they are delayed effected the schedule of the whole project (Gerver, 2014).

Because changes are inevitable risk management became an important aspect of TPM and is about managing the uncertainties of a project, the uncertainties that might affect the project are the risks. And is used to maintain better control over the project (Arkesteijn & Mooi, 2014). Risk can have either positive or negative effects on the project. A risk can or cannot happened, risks are calculated in percentages between 0% and 100%. A risk consists of the likelihood that it might happen and the impact of the event. Risks are identified by numerus techniques, as project analogy, through the WBS and process flowcharts. This estimation process is based on historical knowledge and previous experiences (Nicholas & Steyn, 2017). Risk are logged into a risk register, in here the risk are assed if any action is worthwhile taking and mitigation measure need to be implemented (Arkesteijn & Mooi, 2014; Nicholas & Steyn, 2017). Risk management is a soft science, highly depended on the human input (Arkesteijn & Mooi, 2014).

Internal and external communication

TPM often makes use of command and control style of project management (Owen et al., 2006; Serrador & Pinto, 2015; Verbruggen, 2017). This is caused because the project manager is responsible for the result, the plan, direct and integrate effects by the stakeholder (Nicholas & Steyn, 2017). This can only be achieved by communicating with the team and the stakeholders.

The PMBOK guide define 3 areas of project communication; plan communications management; manage communications; and monitor communications. The PMBOK guide defines 6 types of distributed information; communications to stakeholders; Project reports; Project presentations; Project documentation; Stakeholder feedback to improve performance; and documented experience for historical records to improve future performance (Zandhuis & Wuttke, 2019). Nicholas & Steyn, (2017) add communication to the chapter on evaluation, implementation and closeout. A communication plan is recommended for larger projects. Which should include a schedule with all details on documentation and formal communication including important milestones. According to Nicholas & Steyn, (2017) this plan should than be distributed to the whole project team before the project starts. Communication is important, especially at the start of projects were communication needs to be frequented and rapid (Bröchner & Badenfelt, 2011)

TPM makes use of progress reporting, these can be on a weekly, monthly or quarterly basis or any time in between. These reports are used to inform different stakeholders. Depending on their role and interest in the project the stakeholder receives their report with the goal of producing a progress report that satisfies the needs of most of the stakeholders and reducing making tailor made reports to a minimum (Gerver, 2014).

Client involvement is essential in defining requirements, this is done in the early stages of a project. In the execution phases of a project costumer involvement is seen as unwanted because they are seen as a necessary by irritating obstruction to efficiently completing and executing the project planning (Owen et al., 2006).

Learning

Projects are often approach as so unique that the lesson of history are ignored, but no project is ever so unique that it cannot learn from the past (Nicholas & Steyn, 2017). TPM uses a single loop of learning, this means only the action strategy is evaluated not the governing variables (Owen et al., 2006).

2 Agile project management

This chapter is the unedited version of the elaboration on agile project management.

2.1 Explaining agile project management ~ the agile manifesto

The term agile project management was coined in 2001, a two-day conference with 21 leaders of the then called lightweight project management methodologies led to the conception of the agile manifesto (Jim Highsmith, 2001). This meeting resulted in the conception of the agile manifesto, where the 21 leaders and later more, pledged to realise their IT project better through the ideas presented in the manifesto. This manifesto can be easily found on the website agilemanifesto.org. Here the 4 rules and 12 principles are presented on which APM is based (Beck et al., 2001a). The four rules of APM are:

- Individuals and interactions over process and tools;
- Working software over comprehensive documentation;
- Customer collaboration over contract negotiation;
- Responding to change over following the plan (Beck et al., 2001a).

Important to note is that traditional values; process and tools, comprehensive documentation, contract negotiation and, following the plan still hold value but not over the other values presented (Beck et al., 2001a).

APM is able to deal with real-world development. this is done in increasingly volatile environments as organizations need to adapt and deal with changing markets, technology and social conditions. Even changes that seem small may produce unexpected effect, especially in complex and interdepend systems (Augustine, Payne, Sencindiver, & Woodcock, 2005). Hamed & Abushama, (2013) state that the target of the agile manifesto is to ‘‘ *improve business and innovate new ideas to meet the market demands by quickly redefining resources when the requirements or technology changed, fast response to the market changes or insensitive customer interaction"*. They also highlight the importance to deal with the unpredictability of the process and changing requirements.

Table 1: the principles of the agile manifesto and the ideas it represents

Nr.	**APM principles according to Beck et al. (2001b)**	**Explanation according to** Verbruggen, (2017a)
1	Our highest priority is to satisfy the customer through early and continuous deliver of valuable software.	Achieve consumer satisfaction
2	Welcome changing requirements, even late in development. Agile process harness change for the customer’s competitive advantage.	Welcome change
3	Delivering working software frequently, from a couple of weeks to a couple of months, with a preference to shorter timescale.	Deliver frequently
4	Business people and developers must work together daily throughout the project.	Work together with the business
5	Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.	Motivate and empower people
6	The most efficient and effective method of conveying information to and within a development team is face-to-face communication	Face-to-face communication
7	Working software is the primary measure of progress	Working software is the definition of done
8	Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.	Maintain sustainable pace
9	Continuous attention to technical excellence and good design enhances agility.	Maintain a good design
10	Simplicity – the art of maximizing the amount of work not done – is essential	Keep it simple

11	The best architectures, requirements, and designs emerge from self-organizing teams	Work in self-organising teams
12	At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.	Reflect to become more effective

APM methods were developed in order to achieve better customer satisfaction, to shorten the development time of IT projects, reduce rework and to deal with changing environments (Leau, Loo, Tham, & Tan, 2012). These methods are based on the agile manifesto or created the agile manifesto (Stoica et al., 2013). The agile manifesto is there for the highest order and the true embodiment of APM. The rules and principles are the basis in which agile methodologies like scrum and extreme programming (XP) are built. Some of these methodologies found their conception before the realisation of the manifesto, but their core concepts find their way back into the manifesto. This is because some of the authors of the manifesto were founders of agile mythologies (back then often referred to as lightweight methodologies) (Beedle et al., 2001). Ken Schwaber and Jeff Sutherland, for example, are founders of the agile manifesto but also that of the agile methodology scrum (Sutherland, 2015). The same goes for Ron Jeffries creator of XP and Alistair Cockburn, founder of crystal, both agile mythologies (Abbas, Gravell, & Wills, 2008).

Therefore, these methodologies (Scrum, XP etc..) as interpretations on how to fulfil the goals set out in the agile manifesto and their way of implementing this agile belief system. There are many agile methods and it seems that important world players in the field of IT or other sectors that want to adopt APM make their own agile methodology. Scrum however is the most used APM method and is reported to is used by 37% - 58% of the companies reporting to the state of agile reports. A full analysis of the reports published between 2007 and 2019 can be found in Agile methodologies between 2006 and 2019.

Individuals and interactions *over* process and tools

The first value of the agile manifesto focusses on the relations between individuals. It highlights the importance of valuing people and trusting them to get the work done. This is reflected in four of the twelve agile principles:

- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- The best architectures, requirements, and designs emerge from self-organizing teams

Verbruggen (2017) states that an agile approach is people-oriented. That it has the focusses on the individuals of a team that are trusted through self-organizing teams to get the best result. Collaboration between individuals and developing individuals’ relationships and skill are central to this value.

Working software *over* comprehensive documentation

The second value refers to measuring progress through the delivery of working products. Focus needs to be on the product not on paper work. Griffiths, (2015) states that documentation needs to be done just on time, just enough and sometimes just because. Documentation is important but it needs to be just enough or because it is required or requested. All other documentation will bog down the process. Historically this value was created to stop the flood of specifications and requirements. APM tries the streamline documentation. According to Abbas et al., (2008) this value is best described as: “ a physical object conveys more information than any written specification''. This value is represented in the following 4 agile principles as stated in the manifesto.

- Delivering working software frequently, from a couple of weeks to a couple of months, with a preference to shorter timescale.
- Working software is the primary measure of progress
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity – the art of maximizing the amount of work not done – is essential

Customer collaboration *over* contract negotiation

Third value sets customers as a priority. Aiming to get the client to involved in the process. The value tries to counter the running-to-the-contract effect, where parties don't work together but through the contract. When something goes wrong and issues emerge the contract is consulted not the customer. It also facilitates to the ability to deal with the changing customer needs. 3 principles reflect this value.

- Our highest priority is to satisfy the customer through early and continuous deliver of valuable software.
- The most efficient and effective method of conveying information to and within a development team is face-to-face communication
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Responding to change *over* following the plan

Last of the 4 values allows change. Instead of a 100% set plan that needs to be fulfilled, change is dealt with and in some cases even welcomed. Responding to change was added to reflect the reality of project. Initial plans are made before hand when not much is known and information is limited (Verbruggen, 2017). This is reflected in 2 of the 12 principles form the agile manifesto:

- Welcome changing requirements, even late in development. Agile process harness change for the customer's competitive advantage.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

To summarize APM is build-up of the 4 values and 12 principles also known as the agile manifesto this is the foundation of APM. From this manifesto agile methodologies were developed, using agile practices to fulfil the requirements set-out in the agile manifesto.

2.2 Typical agile mechanics, ideas and tools ~ agile in practice

As established is Agile project management any project management methodology that fulfills the 4 values and 12 principles established in the agile manifesto. This led to common practices and ideas to facilitate and fulfill the guidelines set out by the agile manifesto. This sub-chapter makes use of numbers between brackets, the number with in the bracket reflects the agile principles that is satisfied with that practiced.

According to J Highsmith & Cockburn, (2001) agile is about creating and responding to change. The importance of people and understanding that they are the primary driver of project success in combination with effectiveness and manoeuvrability in projects. Key techniques of agile development as defined by Cockburn, (2006):

Appendix project management

- Collocate people to facilitate osmotic communication
- Access to expert users to acquire early feedback
- Timebox & deliver to acquire feedback and celebrate accomplishments
- Burn-down charts to facilitate the visibility of progress
- Distribute responsibilities between managers and programmers
- Daily stand-up to relay current information and change
- Simpler design to respond faster and easier to change
- Concurrent development to respond to change
- Automated testing to permit change

2.2.1 Iterative development ~ iterative and incremental planning

APM is both iterative and incremental (Fernandez & Fernandez, 2008). Iterative development is a making progress through refinement and go's therefor through a cycle over and over again until the project is finished or terminated. Meaning that products goes through a development cycle multiple time to keep improving the product. Incremental is when a product is built and delivered in pieces. Each piece is a chuck or subset of the overall product that needs to be delivered.

This incremental and iterative process are used to fulfil two of the agile principles [2] and [3]. By using the cycles changes can be incorporated in the next cycle. Delivering from weeks to a couple of months is just one of the principles of APM. This incremental development delivers an incomplete solution to the client and the result is discussed, evaluated and changed according to the client's specifications welcoming change in the meantime.

Through the use of these iterations also referred to as "time-boxes" or "sprints" (DSDM Consortium, 2014; Sutherland, 2015) Delivering value as fast as possible to the client and showing the progress the project has made [1],[7]. This progress is show through the time-boxes, at the end of each time-box a demonstratable outcome has to be presented to the client who can give feedback on the delivered value (DSDM Consortium, 2014). Often hold reflection or lessons learned sessions to find out wat they can improve in the next iteration (Hass, 2007).This means that continues learning is used to improve the end product desired by the client and the team performance (Owen et al., 2006)[12].
Through the use of increments the project is broken up in to pieces in a fixed amount of time. These smaller project pieces make the project more manageable and the keep the tasks at hand simpler. Teams are not overwhelmed with all the work they need to complete [8],[10].

These sprints often utilize different tool to help with planning the time-box, understanding what needs to be done. Agile methodologies often make use of feature definitions that are made by the client instead of tasks. This is done because the clients better understand the features he or she needs (J Highsmith & Cockburn, 2001). A feature is an unit of functionality described by the consumer in her or his words in terms of what they need (Karlesky & Vander Voord, 2008). XP coined the term user stories and is adopted by many agile methodologies (Lucassen, Dalpiaz, Werf, & Brinkkemper, 2016), and mentions of this can be found in the books of DSDM Consortium, (2014) and Sutherland, (2015). These stories use a simple template to define a feature which is then written on a story card. User stories usually follow the "as a <role> I want <something> so that <benefit> (Lucassen et al., 2016). Due to this APM methods are often described as feature-driven approaches (Hass, 2007). User stories are used in the planning to estimate and forecast the project progress (Haugen, 2006).

APM methods use prioritization to manage their requirements/features(Hass, 2007; Owen et al., 2006). Different prioritization methods are used. But the goal is to understand what features are important to the client and which ones are not. This way value to the client is maximized and can changes be incorporated in the design [2], [9] (Daneva et al., 2013; Racheva, Daneva, Sikkel, Wieringa, & Herrmann, 2010).

Often the teams that need to develop these features need to plan when then need to what. The prioritization helps the teams to understand what needs to be developed first but not how long it might take. Features are often but not always estimated on their size before a the time to complete the feature is estimated (Coelho & Basu, 2012). Other techniques for the feature time estimating are used such as expert judgement and linear regression (Usman, Mendes, Weidt, & Britto, 2014). Basically two types of estimating are used either size, effort or a combination of both (Usman et al., 2014).

These estimations are used to plan the project on three levels, Release, Iteration and Daily. Release project level, at the end the project is finished. Release level determines the needed amount of iterations. Iteration level determine what needs to be done that iteration. And on daily level it is decided what is done that day (Usman et al., 2014).

The planning takes place during the three levels. Daily meetings usually lasting only 15 minutes are done to plan what needs to be done that day. Iteration meeting that is done at the start or end of an iteration cycle. Here the next iteration is planned and the last iteration cycle is evaluated, done by a team working on a specific feature. Release planning is the overall planning which is conducted before a few iteration cycles are planned out. Important dates and deliverables are discussed, done with multiple teams to build the bigger picture (Smits, 2006).

Some agile project makes use of a burn down chart to measure their velocity and estimate if they are on schedule or not. The amount of features are plotted against time, the amount of features should decline over time, a straight line from the start to the end of the project is drawn against this line the velocity of the project is measured. This is used to visualise project progress (Karlesky & Vander Voord, 2008).

Cockburn states the use of burn-down charts as a key agile technique. Burn-down charts are used to visualise the progress and measure the statutes of a project (Woodward, Cain, Pace, Jones, & Kupper, 2013).

I would add the other common tool used to the list created by Cockburn and that is the Kanban board (sometime progress graph) in which sticky notes with tasks are placed on a board (Agarwal & Majumdar, 2012).

A task or a kanban board are often used by agile teams to manage the features they need to deliver. In it simplest form this task board consist of three lanes: To-Do; In progress; Done. This board is then filled each irritation with the features that need to be delivered (Hass, 2007). These features are made on-by-one until the iteration is over. Features not developed are discussed and moved to the next iteration of terminated. The amount of features delivered for that iteration are feedback into the burndown chart. Now the chart shows the amount of features delivered and what should have been delivered. Visualizing if the needed amount of features are delivered, or more or to little features are developed. Sometimes instead of the amount of features, time or effort are used in the burndown chart. Now the features that are on the top of the prioritization list are developed and the list is shortened until completed or until the time or money ran out.

2.2.2 Communication, learning and visualisation

These topics are intertwined with the way agile project are planned and estimated. As established in the planning section three planning levels can be distinguished. For each of these levels a meeting is planned, a daily meeting, one after each iteration cycle. In this las cycle value is delivered to the client who can react to the value, improve or change the scope of the project. Communication and visualisation help to

develop the project, this way the client is more engaged in the project. Client also can reprioritize the features needed after each completed iteration (Daneva et al., 2013). Client and developing parties need to collaborate in order to effectively develop the clients desired product (Hass, 2007).
Agile teams are often co-located, this mean that all key team members including consumers, client work at the same facility. With the preference that this work is done in the same room [6],[4] (Hass, 2007).

For the team's communication is often conducted in short daily meetings, here the progress of the project is reflected. Team members help each other when needed and tasks are divided often with the use of the task board (cards-on-the-wall method). The use of this board is to create visual control for the team members. Members can see where they are during that iteration and what still needs to be done (Hass, 2007).Giving the team control over their own planning and pace. But also trusting the team they know what they are doing and that they will deliver the features the client [5],[11]. This can only be achieved when team are collaborating with team members and other teams in the same project. Most agile team are small to improve communication with in the team (Santos, Bermejo, Oliveira, & Tonelli, 2011).

Pekka Abrahamson, Outi Salo, Jussi Ronkainen, & Juhani Warsta, (2002) state that the developer should keep the code as simple as possible. Simplicity is a direct principle from the agile manifesto. And relates to all aspects of the project. I would argue that they are after parsimony, as simple as possible but not simpler than that. However, this principle is important not only because it lessens the needed amount of documentation (Pekka Abrahamson et al., 2002). But it also helps communicate faster, because the presented material is easier understood [10].

Self-organizing teams is also one of agile principle. This is principle is because APM wants to rely on the people in the project and their creativity over tools and process (Moe, Dingsøyr, & Dybå, 2008). These teams have autonomy to decide over their own work and organize the details around this work. Through this leadership is often distributed in the teams and managers function as facilitators (Hoda, Noble, & Marshall, 2010; Verbruggen, 2017). Member of these teams make use of informal, implicit and spontaneous roles to satisfy their organizational needs [11] (Hoda & Murugesan, 2016).

Some agile project makes use of a product owner. This is a person who has the task to communicate between the development team and the client. She or he have the responsibility to anticipate client needs, prioritizing client needs as a few other task (Bass, 2013).

All these meetings facilitate the opportunities for the team to develop and reflected on how the team is doing. This is done to continuously keep improving (Hass, 2007).Team evaluation help with and trying to be better help with developing a sustaining pace [8], through communication and evaluation teams can better understand themselves and the workings of the teams.

Testing is often mentioned in agile reading material. Verbruggen, (2017) states that agile methods use test-driven development. After ever iteration and early in the process testing should be done. And then there is the importance of defining the acceptance test early on in development. However, this testing refers to code testing. A specific element of IT projects. This doesn't mean testing has not value to other types of projects or is less important, just different. Testing of delivered value and that it adds value for the client are essential in evaluating if a project is delivering what it needs to do. However, this type of testing is often placed on, lessons learn, evaluation or feedback.

Teams are usually managed by a project manager but one of the agile principles states: *The best architectures, requirements, and designs emerge from self-organizing teams* (Beck et al., 2001b). This might lead to the idea that a project manager is not necessary anymore, because evidently the teams would organize everything themselves. In reality this is not the case, sometimes this project manager role is changed and/or called different. In scrum for example this person is the scrum master, in XP this is merely the manager. Verbruggen, (2017) states that the core role of the agile project manager is similar to that of the traditional project manager. She also describes the need for the agile manager to be human-centric and have an agile mindset. The agile manager is often regarded as a facilitator rather than a manager with a command-and-control style (Verbruggen, 2017).

2.2.3 Project results and project success

Often the success of the projects is judged against the iron triangle. When comparing at the values of the iron triangle against APM it is found that APM methods often fix time and resources and scope is kept flexible to accommodate change (Verbruggen, 2017). This is mechanic is used to keep control over the project, and giving focus to the end result. This means that the client almost always beforehand knows when the project is finished and what it will cost. No more delays and no cost overruns, however not all features that the client asked for might have been delivered. This will only work is the client accepts this idea. Client and the development teams need to work together to achieve the most value for the client in the set time and recourses limit. Working together will show the client what is possible in the set constraints and for the team it is important to communicate with the client to achieve consumer satisfaction. By having a flexible scope, it is also so that the end result of the project is not clearly defined. Only what the client’s needs are are established.

Jim Highsmith & Wooden, (2005) propose to use the agile triangle instead, for which the points are made up of: Value, Quality and Constraints. This last one will than embody the scope of the iron triangle. Highsmith & Wooden argue that this way of thinking will deliver better value to the customer.

The points of the iron triangle determine the quality of the project (Jha & Iyer, 2007). This quality is then tested. APM often defines test at the same test as the requirements are defined. This is done to ensure the requirement is testable and understood enough to be build (Hass, 2007). Writing test before the execution of the projects starts ensures quality and controls the requirements.

Customer satisfaction stands central with in APM. Close customer collaboration is needed, the projects is all about the features the consumer needs. The client is closely involved in the development through meetings and value delivery by the development teams. This is therefor also an important measure for agile practitioners to determine if there project was successful or not (Bass, 2013; Serrador & Pinto, 2015). Working with agile methods increase job satisfaction (Tripp, Riemenschneider, & Thatcher, 2016).

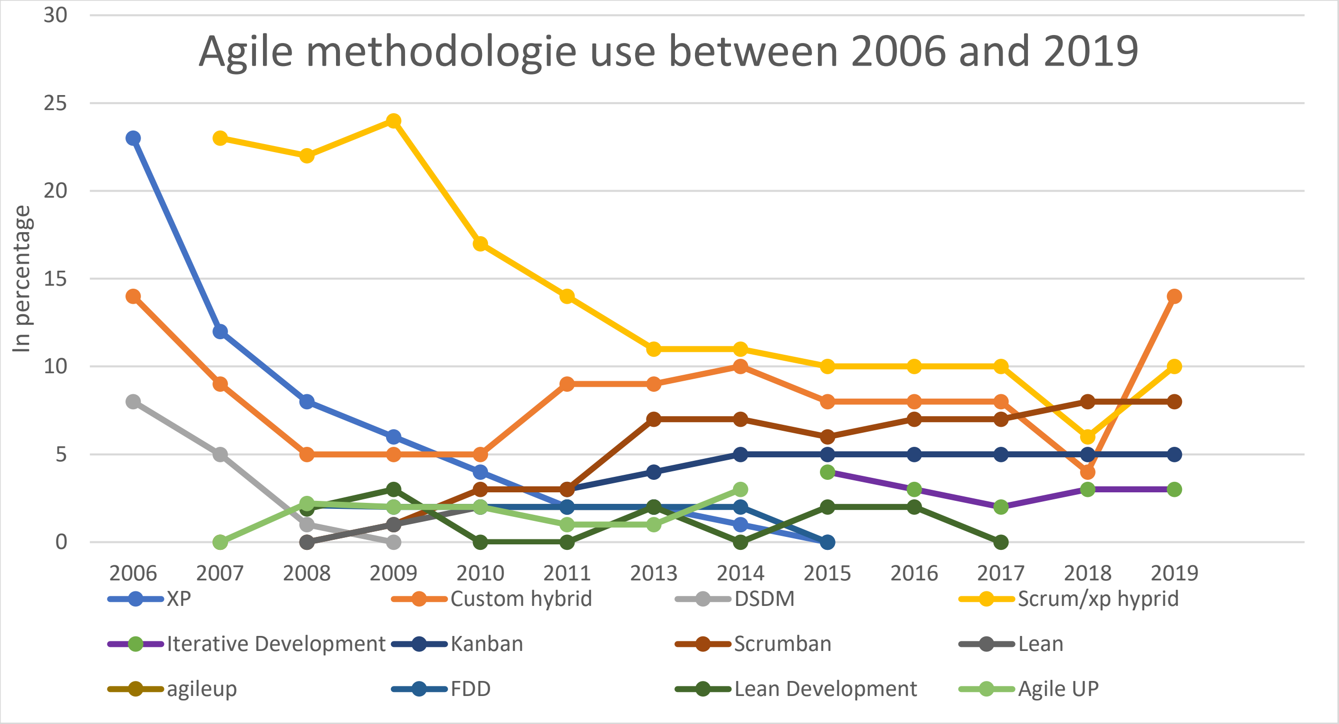
2.2.4 Documentation

Through the use of value delivery and user stories documentation is minimized. By using visualisation after each iteration and showing the client the produced value documentation is minimized. Simply because there is less need for reporting project progress. Through defining features and prioritization of these features, the client does not have to determine the whole project at the start of the project. But can be incorporated through the development cycles.

3 Agile methodologies between 2006 and 2019

In 2001 is the agile manifesto was founded. Some of the writers of the manifesto developed their own methodologies. Like Scum, XP and DSDM all methodologies fulfil the aspects of the agile manifesto. Some agile methodologies found their conception before the publication of the manifesto. Back then these methodologies where called lightweight project management methodologies (Jim Highsmith, 2001). Since 2006 (with the exception of 2012) the state of agile project management publicises a report on the use of APM. So far 13 reports are conducted and published online at stateofagile.com. This website is run by the global company CollabNet VersionOne a software development and delivery solutions provider (CollabNet VersionOne, n.d.). The global survey's where conducted by VersionOne until they were acquired by CollabNet in 2017, where the practise was continued (CollabNet VersionOne, 2017; VersionOne Inc., 2006). The survey is sent to all sizes of companies, with the goal to summarize the state of agile development in organizations. The survey focuses on every industry that uses agile globally. The company implemented a agile methodology. The survey also asked if there are any improvements made by the use of APM and the barriers encountered by the practitioners (VersionOne Inc., 2006). Over the years the amount of respondents has grown from 722 in 2006 to 1319 in 2019 (Stateofagile.com, 2019; VersionOne Inc., 2006). In the lasted report it is stated that 17% of the respondents is a customer of CollabNet VersionIne.

From these 13 reports the used methodologies are gathered. 17 different types of APM methodologies where mainly used. In the last 14 years Scrum rained supreme. With the least user in 2007 with 37% of the respondents and in 2007 its highest point with 58% (VersionOne Inc., 2007, 2010, 2016, 2017).Graph 1 shows the use of methodologies over the past 15 years (without 2012, when there was no report) Scrum is excluded because of its major share in used APM methodologies. Graph 1 also doesn't show the small shares to make the graph more readable.



Graph 1: Agile methodology use between 2006 and 2019

4 Original comparison table by Verbuggen, (2017)

Comparison table create by Verbruggen, (2017). On this comparison the comparison in chapter **Error! Reference source not found.** is based. Due to its size it is on the next page.

Appendix project management

Comparators	Aspects	Agile project management	Traditional project management
Philosophy			
Mindset		Individuals Working software Customer collaboration Responding to change	Processes & tools Comprehensive documentation Contract negotiation Following the plan
Organisation and management			
Organisation	*Structure*	Flat team-based structure	Hierarchical structure
	Form	Flexible & participative encouraging cooperative social action *(organic)*	Bureaucratic with high formalization *(mechanic)*
	Culture	Comfort and empowerment via many degrees of freedom *(thriving on chaos)*	Comfort and empowerment via framework of policies and procedures *(thriving on order)*
Management	*Management style*	Leadership-and- collaboration	Command-and-control
	Decision making	Decentralized & pluralist decision making	Centralized & managerial decision making
Development process			
Development methods	*Development style*	Iterative, adaptive, extreme	Linear, incremental
	Development model	Evolutionary delivery model; *e.g. Scrum, XP, DSDM, Crystal, FDD*	Life cycle model; *e.g. Waterfall model, spiral model, V-model*
Development approach	*Project cycle*	Guided by project features	Guided by tasks or activities
	Iron triangle	Resources and time are fixed	Scope (solution) is fixed
Development direction & nature of planning	*Development direction*	Adaptable; readily changeable	Pre-planned; fixed
	Planning approach	Planning is done prior and for every iteration	Rigorous planning for the entire project
Value delivery frequency	*Value delivery*	Frequent value delivery; after every iteration (timebox) value is delivered to the customer	At the end of each phase/ at the end of the project the value accepted by the customer
Dealing with change	*Change*	Change is inevitable, dealt with after every iteration	Threat for meeting the plan, not dealt with until the next release
People and team			
Teamwork	*Team composition*	Small teams, collaborative work	Large teams, individual work
	Team location	Co-located teams	Not always co-located teams
	Role assignment	Self-organising teams & encourages role interchangeability	Individual & favours specialisation
	Skills	Interpersonal & multidisciplinary skills	Specialized skills
	Reward systems	Team award systems	Individual award systems
Customer involvement	*Customer involvement*	High customer involvement; dedicated customers focused on prioritized increments	Low customer involvement; as-needed customer interactions focused on contract provisions
	Customer location	Co-located customer	Not always co-located customer
Attitude to learning	*Learning type*	Double loop learning	Single loop learning
Technology			
Requirements	*Definition of requirements*	Requirements can undergo unforeseeable change, and consist of prioritized informal stories	Requirements undergo a foreseeable evolution and are formalized *(e.g. projects, capabilities, interfaces, quality)*
	Clarification of requirements	Requirements discussed and clarified ‘‘just-in-time’’	Requirements at the beginning of the project *(Contract driven; requirements serve as contract)*
Testing		Executable test cases define requirements testing	Documented test plans and procedures
		Write test prior to code *(test-driven development)*	Write code prior to test
	Timing of testing	Testing early in the development process	Testing late in the development process
	Frequency of testing	Testing on every iteration *(incl. customer acceptance testing)*	Testing after coding phase completed *(incl. customer acceptance testing)*
Release frequency		High release frequency *(‘‘go live’’ release per one to four weeks)*	Low release frequency *(‘’ go live’’ release per six or more months)*
Project metrics and documentation	*Documentation*	Minimal, up-to-date metrics	Emphasis on data collection
	Knowledge & communication	Tacit knowledge & informal communication	Explicit knowledge & formal communication

Coding	*Design*	Simple design; refactoring assumed inexpensive	Extensive design; refactoring assumed expensive
	Code ownership	Collective code ownership	Not always collective code ownership

5 New table

Table 2, Table 3 and Table 4 are the fully reshuffled and complemented tables.

Table 2: Philosophy differences between agile- and traditional project management

Comparators	Aspect	Agile project management	Traditional project management	Source
Theory	Fundamental basis	Agile manifesto	Management-as-planning, Dispatching model and thermostat model used in combination.	Koskela, L. & Howell, 2002
Philosophy	Fundamental assumption	High-quality adaptive software is developed by small teams that use continuous improvement of design and testing based on fast feedback and change.	Systems are fully specifiable, predictable and are developed through extended and detailed planning.	Dybå & Dingsøyr, (2008); Nerur et al., (2005) ,Stoica et al., (2013)
	Project driver	Quick value	High safety	Stoica et al., (2013)

Table 3: Principles differences between agile- and traditional project management

Comparators	Aspect	Agile project management	Traditional project management	Source
Origination & People				
Teamwork	Developers	Agile, with advanced knowledge, co-located and cooperative	Oriented on plan, with adequate abilities, access to external knowledge	Stoica et al. (2013)
	Team location	Co-located teams	Not always co-located teams	Verbruggen (2017)
	Role assignment	Self-organising teams & encourages role interchangeability	individual & favors specialization	Verbruggen (2017)
Attitude to learning	Learning type	Double-loop learning	Single-loop learning	Verbruggen (2017)
Customer involvement	Customer location	Co-located customer	Not always co-located costumer	Verbruggen (2017)
Organization	Customer role	Critical	Important	Dybå & Dingsøyr, (2008)
Development				
Release frequency		High release frequency	Low release frequency	Verbruggen (2017)
Value delivery frequency	Value delivery	Frequent value delivery after every iteration (timebox) value is delivered to the customer	At the end of each phase/ at the end of the project the value accepted by the customer	Verbruggen (2017)
Project realisation				
Coding	Design	Simple design; refactoring assumed inexpensive	Extensive design; refactoring assumed expensive	Verbruggen (2017)
Dealing with change	Change	change is inevitable, dealt with after every iteration	threat for meeting the plan, not dealty with unitl next release	Verbruggen (2017)
Project metrics and documentation	Documentation	Minimal, up-to-date metrics	Emphasis on data collection	Verbruggen, (2017)
	Communication	Mostly Informal communication	Mostly formal communication	Verbruggen, (2017)
Quality	Quality control	Continuous control of requirements designs and solutions	Heavy planning and strict control	Dybå & Dingsøyr, (2008); Stoica et al., (2013)
Requirements	Type of requirements	Functional requirements	Technical requirements	
	Definition of requirements	Requirements can undergo unforeseeable changes and consist of prioritized informal stories	Requirements undergo for seeable evolution and are formalized	Verbruggen (2017)
	Clarification of requirements	Requirements discussed and clarified "just-in-time", emergent with rapid changes	Requirements at the beginning of the project, stable known in advance	Verbruggen (2017)
Testing	timing of testing	Testing early and continuously during development	testing late in the development process	Verbruggen (2017)

	Frequency of testing	testing on every iteration (Inc. customer acceptance testing)	testing after development phase is complete (Inc. customer acceptance testing)	Verbruggen (2017)

Table 4: Practice differences between agile- and traditional project management

Comparators	Aspect	Agile project management	Traditional project management	Source
Orginsation & People				
Customer involvement	Customer involvement	High customer involvement	Low customer involvement; as-needed customer interactions focused on contract provisions	Verbruggen (2017)
	Clients	Dedicated, knowledgeable	With access to knowledge	Stoica et al., 2013
	User involvement	Close and frequent collaboration	Not involved	Špundak, 2014
Management	Management style	Leadership-and- collaboration	Command-and-control	Verbruggen, (2017)
	Decision making	Decentralized & pluralist decision making	Centralized & managerial decision making	Verbruggen (2017)
	Control	People centric	Process centric	Nerur et al., 2005
Organisation	Structure	Flat team-based structure	Hierarchical structure	Verbruggen (2017)
	Form	Flexible & participative encouraging cooperative social action (organic)	Bureaucratic with high formalisation (mechanic)	Verbruggen (2017)
	Culture	Comfort and empowerment via many degrees of freedom (thriving on chaos)	Comfort and empowerment via a framework of policies and procedure (thriving on order)	Verbruggen (2017)
Project metrics and documentation	Knowledge	Tacit knowledge	Explicit knowledge	Verbruggen (2017)
Teamwork	Team composition	Small teams	Large team, fluctuation expected	Verbruggen (2017)
	Team cooperation	Collaborative	Individual	Verbruggen, (2017)
	Skills	Interpersonal & multidisciplinary skills	Specialized skills	Verbruggen, (2017)
	Reward systems	Team award system	Individual award system	Verbruggen, (2017)
	Developer extra abilities	Interpersonal abilities and basic knowledge of the business	-	(Stoica et al., 2013
Development				
Development approach	Project cycle	Guided by project features	Guided by tasks or activities	Verbruggen (2017)
	Iron triangle	Resources and time are fixed	Scope is fixed	Verbruggen (2017)
Development direction & nature of planning	Planning approach	Planning is done prior and for every iteration	Rigorous planning for the entire project	Verbruggen (2017)
	Planning visualisation	Kanban board or burndown chart	Gantt chart	Hass (2007)
Development methods	Development style	Iterative, adaptive, extreme	Linear, incremental	Verbruggen (2017)
	Development model	Evolutionary delivery model	Life cycle model	Verbruggen, (2017)
Project realisation				
Coding	Code ownership	Collective code ownership	Not always collective code ownership	Verbruggen (2017)
Requirements	User requirements	Interactive acquisition	Detailed and defined before implementation	(Stoica et al., 2013
Rework	Rework cost	Low	High	Leau et al., 2012; Stoica et al., 2013
Testing	Testing fundamentals	Executable test cases define requirements testing	Documents test plans and procedures	Verbruggen, (2017)
	Defining test	Write test prior to code	Write code prior to test	Verbruggen (2017)
Delivery	Finished project	Definition-of-done	Verified requirements	Lindstrom & Jeffries, 2004

6 References

Abbas, N., Gravell, A. M., & Wills, G. B. (2008). Historical Roots of Agile Methods: Where Did “Agile Thinking” Come From?, 94–103. https://doi.org/10.1007/978-3-540-68255-4_10

Agarwal, M., & Majumdar, R. (2012). Tracking Scrum projects Tools, Metrics and Myths About Agile. *International Journal of Emerging Technology and Advanced Engineering*, *2*(3), 97–104.

Arkesteijn, R., & Mooi, H. (2014). *Management of Engineering projects: People are Key, Chapter 8*. (H. Bakker & J. P. Kleijn, Eds.) (1st ed.). NAP - The Process Industry Competence Network.

Atkinson, R. (1999). Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria, *17*(6), 337–342. https://doi.org/10.1016/S0263-7863(98)00069-6

Augustine, S., Payne, B., Sencindiver, F., & Woodcock, S. (2005). Agile project management: Steering From the Edges, *48*(12), 85–89. Retrieved from http://delivery.acm.org.libezproxy.open.ac.uk/10.1145/1110000/1101781/p85-augustine.pdf?ip=137.108.145.45&id=1101781&acc=ACTIVE SERVICE&key=BF07A2EE685417C5.402DE054EF770E8A.4D4702B0C3E38B35.4D4702B0C3E38B35&__acm__=1532679831_814f7bfd3651dcba332ca147dc27

Bakker, H. L. M., & Kleijn, J. P. (2014). *Management of Engineering Projects: People are Key* (1st ed.). Nijkerk: NAP - The Process Industry Competence Network.

Bass, J. M. (2013). Agile method tailoring in distributed enterprises: Product owner teams. *Proceedings - IEEE 8th International Conference on Global Software Engineering, ICGSE 2013*, (August), 154–163. https://doi.org/10.1109/ICGSE.2013.27

Beck, K., Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001a). Manifesto for Agile Software Development. Retrieved June 15, 2018, from http://agilemanifesto.org/

Beck, K., Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001b). Principles behind the Agile Manifesto. Retrieved May 22, 2019, from https://agilemanifesto.org/principles.html

Beedle, mike, van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Highsmith, J., … Thomas, D. (2001). Signatories: The Agile Manifesto. Retrieved May 22, 2019, from https://agilemanifesto.org/authors.html

Bröchner, J., & Badenfelt, U. (2011). Changes and change management in construction and IT projects. *Automation in Construction*, *20*(7), 767–775. https://doi.org/10.1016/j.autcon.2011.01.001

Cockburn, A. (2006). Agile Software Development: The Cooperative Game (2nd Edition) (Agile Software Development Series). *Subversion.Assembla.Com*, (February), 220. Retrieved from http://www.ccs.neu.edu/home/lieber/com1205/w03/lectures/lec1d-short-agile-sd.ps

Coelho, E., & Basu, A. (2012). Effort estimation in agile software development. *Proceedings of the 10th International Conference on Predictive Models in Software Engineering - PROMISE ’14*, *3*(7), 82–91. https://doi.org/10.1145/2639490.2639503

CollabNet VersionOne. (n.d.). About. Retrieved June 11, 2019, from https://www.collab.net/company/about

CollabNet VersionOne. (2017). CollabNet and VersionOne Merge to Accelerate Innovation Across Enterprise Software Development and Delivery. Retrieved June 11, 2019, from https://www.collab.net/news/press/collabnet-and-versionone-merge-accelerate-innovation-across-enterprise-software

Daneva, M., Van Der Veen, E., Amrit, C., Ghaisas, S., Sikkel, K., Kumar, R., … Wieringa, R. (2013). Agile requirements prioritization in large-scale outsourced system projects: An empirical study.

Journal of Systems and Software, *86*(5), 1333–1353. https://doi.org/10.1016/j.jss.2012.12.046

de Jong, C. E. (2018). *Fit-for-purpose project management*. TU Delft.

DSDM Consortium. (2014). *Agile Project Management Handbook v2.0* (2nd ed.). DSDM Consortium. Retrieved from https://apmg-businessbooks.com/books/project-programme-management/agile-v20-project-management-handbook

Fernandez, D. J., & Fernandez, J. D. (2008). Agile project management - Agilism versus traditional approaches. *Journal of Computer Information Systems*, *49*(2), 10–17.

Gerver, M. (2014). *Management of Engineering projects: People are Key, Chapter 10*. (H. L. . Bakker & J. P. Kleijn, Eds.) (1st ed.).

Ghosh, B. S., Forrest, D., Dinetta, T., Wolfe, B., & Lambert, D. C. (2012). Enhance PMBOK ® by Comparing it with P2M , ICB , PRINCE2 , APM and Scrum Project Management Standards. *PM World Today*, *XIV*(I), 1–77. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&db=bth&AN=74028654&site=bsi-live

Globerson, S. (1994). Impact of various work-breakdown structures on project conceptualization. *International Journal of Project Management*, *12*(3), 165–171. https://doi.org/10.1016/0263-7863(94)90032-9

Griffiths, M. (2015). *PMI-ACP Exam Prep (second ed.)*. (RMC Publications, Ed.). United States of America.

Hamed, A. M. M., & Abushama, H. (2013). Popular agile approaches in software development: Review and analysis. *Proceedings - 2013 International Conference on Computer, Electrical and Electronics Engineering: "Research Makes a Difference", ICCEEE 2013*, (February), 160–166. https://doi.org/10.1109/ICCEEE.2013.6633925

Hass, K. B. (2007). The Blending of Traditional and Agile Project Management. *PM World Today*, *IX*(V). Retrieved from http://chelsoftusa.com/uploads/3/4/1/3/34136265/agile_well_explained.pdf

Haugen, N. C. (2006). An empirical study of using planning poker for user story estimation. *Proceedings - AGILE Conference, 2006*, *2006*, 23–31. https://doi.org/10.1109/AGILE.2006.16

Highsmith, J, & Cockburn, A. (2001). Agile software: The business of innovation. *Computer*, *34*(9), 120–122.

Highsmith, Jim. (2001). History: The Agile Manifesto. Retrieved May 22, 2019, from https://agilemanifesto.org/history.html

Highsmith, Jim, & Wooden, J. (2005). *Agile project management: Creating innovative product. The Agile revolution chapter 1* (2nd ed., Vol. 44). Addison-Wesley.

Hoda, R., & Murugesan, L. K. (2016). Multi-level agile project management challenges: A self-organizing team perspective. *Journal of Systems and Software*, *117*, 245–257. https://doi.org/10.1016/j.jss.2016.02.049

Hoda, R., Noble, J., & Marshall, S. (2010). Organizing Self-Organizing Teams. *ACM/IEEE 32nd International Conference on Software Engineering*, *1*, 285–294.

Jha, K. N., & Iyer, K. C. (2007). Commitment, coordination, competence and the iron triangle. *International Journal of Project Management*, *25*(5), 527–540. https://doi.org/10.1016/J.IJPROMAN.2006.11.009

Joslin, R., & Müller, R. (2015). Relationships between a project management methodology and project success in different project governance contexts. *International Journal of Project Management*, *33*(6), 1377–1392. https://doi.org/10.1016/j.ijproman.2015.03.005

Karlesky, M., & Vander Voord, M. (2008). Agile Project Management (or, Burning Your Gantt Charts). *Embedded Systems Conference Boston (Boston, Massachusetts)*, (October 2008), 1–16. https://doi.org/10.1145/1101779.1101781

Khan, A. (2010). Project Scope Management. *Case Studies in Project, Program, and Organizational Project Management*, *48*(6), 83–84. https://doi.org/10.1002/9780470549179.ch5

Appendix project management

Koskela, L. & Howell, G. (2002). The Underlying Theory of Project Management is Obsolete. *Proceedings of the PMI \Research Conference*, 293–302.

Koskela, L., & Howell, G. (2002). The theory of project management: Explanation to novel methods. *Proceedings 10th Annual Conference on Lean Construction*, 1–11. https://doi.org/10.1109/EMR.2008.4534317

Leau, Y., Loo, W. K., Tham, W. Y., & Tan, S. F. (2012). Software Development Life Cycle AGILE vs Traditional Approaches (Vol. 37, pp. 162–167).

Lucassen, G., Dalpiaz, F., Werf, J. M. E. M. van der, & Brinkkemper, S. (2016). The Use and Effectiveness of User Stories in Practice (pp. 205–222). Springer, Cham. https://doi.org/10.1007/978-3-319-30282-9_14

Lycett, M., Rassau, A., & Danson, J. (2004). Programme management: A critical review. *International Journal of Project Management*, *22*(4), 289–299. https://doi.org/10.1016/j.ijproman.2003.06.001

Moe, N. B., Dingsøyr, T., & Dybå, T. (2008). Understanding self-organizing teams in agile software development. In *19th Australian Conference on Software Engineering (aswec 2008)* (pp. 76–85). IEEE.

Munns A. K., & Bjeirmi B. F. (1996). The role of project management in achieving project success. *International Journal of Project Management*, *14*(2), 81–87.

Nicholas, J. M., & Steyn, H. (2017). *Project management for engineering, business and technology* (5th ed.). Retrieved from https://www.sciencedirect.com/book/9780080967042/project-management-for-engineering-business-and-technology

Owen, R., Koskela, L., Henrich, G., & Codinhoto, R. (2006). Is agile project management applicable to construction? *Public Money and Management*, *28*(1), 41–48. https://doi.org/10.1111/j.1467-9302.2008.00617.x

Pekka Abrahamson, Outi Salo, Jussi Ronkainen, & Juhani Warsta. (2002). Agile software development methods: Review and analysis. *VTT Publications*. https://doi.org/10.1076/csed.12.3.167.8613

Racheva, Z., Daneva, M., Sikkel, K., Wieringa, R., & Herrmann, A. (2010). Do we know enough about requirements prioritization in agile projects: Insights from a case study BT - 2010 18th IEEE International Requirements Engineering Conference, RE2010, September 27, 2010 - October 1, 2010. *18th IEEE International Requirements Engineering Conference*, 147–156. https://doi.org/10.1109/RE.2010.27

Santos, M. de A., Bermejo, P. H. de S., Oliveira, M. S. de, & Tonelli, A. O. (2011). Agile Practices: An Assessment of Perception of Value of Professionals on the Quality Criteria in Performance of Projects. *Journal of Software Engineering and Applications*, *04*(12), 700–709. https://doi.org/10.4236/jsea.2011.412082

Serrador, P., & Pinto, J. K. (2015). Does Agile work? — A quantitative analysis of agile project success. *International Journal of Project Management*, *33*(5), 1040–1051. https://doi.org/10.1016/J.IJPROMAN.2015.01.006

Smits, H. (2006). 5 Levels of Agile Planning : From Enterprise Product Vision to Team Stand-up. *Rally Software Development Corporation*, 1–11. Retrieved from https://www.klopsgruppe.de/~franz/wiki/index.php/5_Levels_of_Agile_Planning:_From_Enterprise_Product_Vision_to_Team_Stand-up

Sohi, A. J. (2018). *Flexibility in project management: Towards improving project performance*. TU Delft. https://doi.org/10.4233/uuid

Špundak, M. (2014). Mixed Agile/Traditional Project Management Methodology – Reality or Illusion? *Procedia - Social and Behavioral Sciences*, *119*, 939–948. https://doi.org/10.1016/j.sbspro.2014.03.105

Stateofagile.com. (2019). *13th annual state of agile report*. Retrieved from https://www.stateofagile.com/#ufh-c-473508-state-of-agile-report

Stoica, M., Mircea, M., & Ghilic-Micu, B. (2013). Software Development: Agile vs. Traditional. *Informatica Economica*, *17*(4/2013), 64–76. https://doi.org/10.12948/issn14531305/17.4.2013.06

Sutherland, J. V. (2015). *Scrum: the art of doing twice the work in half the time* (1st ed.). Cornerstone.

Tripp, J. F., Riemenschneider, C., & Thatcher, J. B. (2016). Job Satisfaction in Agile Development Teams: Agile Development as Work Redesign. *Journal of the Association for Information Systems*, *17*(4), 267–307. Retrieved from https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1741&context=jais

Turner, J. (2009). *handbook of project-based management* (3rd ed.). The McGraw-Hill Companies, Inc. Retrieved from http://llrc.mcast.edu.mt/digitalversion/Table_of_Contents_374.pdf

Usman, M., Mendes, E., Weidt, F., & Britto, R. (2014). Postprint Effort Estimation in Agile Software Development : A Systematic Literature Review. *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*, 82–91. https://doi.org/10.1145/2639490.2639503

Verbruggen, A. (2017). *A project manager' s journey towards agile project management*. TU Delft.

VersionOne Inc. (2006). *Survey : " The State of Agile Development ." State of Agile*.

VersionOne Inc. (2007). *State of Agile Development 2th*. *Network*.

VersionOne Inc. (2010). State of Agile Development 5th. Retrieved from http://www.versionone.com/pdf/2010_State_of_Agile_Development_Survey_Results.pdf

VersionOne Inc. (2016). *State of Agile Development 10th*.

VersionOne Inc. (2017). State of Agile Development 11th.

Wasson, C. S. (2015). *System engineering analysis, design, and development: Concepts, principles, and practices*. John Wiley & Sons.

Woodward, C. J., Cain, A., Pace, S., Jones, A., & Kupper, J. F. (2013). Helping students track learning progress using burn down charts. *Proceedings of 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering, TALE 2013*, 104–109. https://doi.org/10.1109/TALE.2013.6654409

Zandhuis, A., & Wuttke, T. (2019). *A pocket companion to PMI's PMBOK® Guide* (6th ed.). Van Haren Publishing. Retrieved from https://www.ako.nl/a-pocket-companion-to-pmi-s-pmbok-guide-sixth-edition-9789401801102.html