

OTDK-DOLGOZAT

VODILA VIKTOR TAMÁS
MSC

2013

**PROJEKT MENEDZSMENT ESZKÖZÖK AZ IND KFT-NÉL
CSINÁLJUK AGILISAN!**

**PROJECT MANAGEMENT TOOLS AT IND PLC.
LET'S DO IT AGILE!**

Kézirat lezárva: 2012. December.20.

TABLE OF CONTENTS

TABLE OF CONTENTS	III
LIST OF FIGURES AND TABLES	IV
1. INTRODUCTION	1
2. THE MOST COMMON CONTROVERSIAL QUESTIONS IN THE PROJECT MANAGEMENT ENVIRONMENT	3
2.1. ARISING CHALLENGES WHEN DEALING WITH DIFFERENT KINDS OF PROJECTS.....	3
2.2. SOLUTIONS STILL NEED TO BE FOUND.....	4
3. AGILE PROJECT MANAGEMENT (APM)	5
3.1. THE HISTORY OF AGILE PROJECT MANAGEMENT	5
3.2. MANIFESTO FOR AGILE PROJECT MANAGEMENT	6
3.3. AGILE PRINCIPLES	7
3.4. EXAMPLE FOR AN AGILE APPROACH	10
3.5. TOP 10 OF HOW TO KNOW YOU ARE NOT DOING AGILE?	10
3.6. ADVANTAGES AND DISADVANTAGES OF USING AGILE PROJECT MANAGEMENT	11
3.7. CASES WHEN AGILE METHODOLOGIES ARE DIFFICULT TO USE	12
3.8. POPULAR FALLACIES ABOUT AGILE PROJECT MANAGEMENT	13
4. WHAT PMBOK® GUIDE PRACTICES CAN BE UTILIZED IN AGILE ENVIRONMENT?. 15	
4.1. AGILE VERSUS TRADITIONAL PROJECT MANAGEMENT	15
4.1.1. <i>Traditional approach - Waterfall model</i>	15
4.1.2. <i>Agile approach - Incremental development model</i>	16
4.1.3. <i>Comparison of traditional and agile project management roles</i>	17
4.1.4. <i>Merging Agility with Distribution</i>	20
4.2. HIGH LEVEL OVERVIEW OF THE CORE ASPECTS	21
4.3. THE SIX KEY KNOWLEDGE AREAS IN THE PMBOK® GUIDE WHICH HAVE THEIR COUSINS IN AGILE . 23	
5. INTRODUCTION OF THE IND GROUP	28
5.1. OVERVIEW.....	28
5.2. IND GROUP – ALWAYS ONE STEP AHEAD THE DEMANDS	29
5.3. INTERNATIONAL PRESENCE AND CLIENTS	29
5.4. STAFF	30
5.5. CORPORATE MILESTONES.....	30
5.6. AWARDS.....	32
6. AGILE PRACTICE AT IND GROUP	34
6.1. THE REASON	34
6.2. THE TRANSITION	34
6.3. LET’S DO IT AGILE – SCRUM, THE AGILE KNOWLEDGE PACKAGE	35
6.3.1. <i>The Scrum Guidelines</i>	36
6.3.2. <i>Scrum project roles</i>	36
6.3.3. <i>Scrum Artifacts</i>	40
6.3.4. <i>Meetings</i>	42
7. CONCLUSION	45
BIBLIOGRAPHY	47

LIST OF FIGURES AND TABLES

Table 1: Agile Manifesto Principles and their relationship to traditional project management practices	7
Table 2: Example for and agile approach	10
Table 3: Agile approach – Myth and Reality.....	14
Table 4: typical project management and governance roles in traditional, plan-driven projects and agile projects	19
Table 5: Characteristics of agile versus traditional distributed software development	20
Table 6: Project processes from the angle of PMBOK® Guide and Agile	21
Table 7: Project management knowledge areas from the aspect of PMBOK® Guide and Agile.....	24
Table 8: Corporate Milestones.....	31
Table 9: IND and client awards on a yearly basis	32
Figure 1: Modified waterfall model - Implementation phase with milestones.....	15
Figure 2: Incremental development approach	16
Figure 3: Agile project lifecycle	16
Figure 4: Typical role definition in traditional development projects	17
Figure 5: Typical role definitions in an agile project.....	17
Figure 6: Waterfall versus Agile: the paradigm shift	26
Figure 7: International presence and clients	30

1. INTRODUCTION

Today's fast moving, information driven world needs that the corporations and professionals to keep pace with the business requirements. Therefore one of the most important factors emerged to be the effective project management practices. Companies continuously seek cost-effective ways to realize their goals and achieve customer satisfaction. Business professionals are becoming more aware of each other's needs and concerns, which is resulting in partnerships that last throughout the project duration. On top of that, the escalating global competition needs an even faster turnaround of any kind of products.

There are many trends driving the development of new project management paradigms. Some of their key complaints are that existing project management processes are often too bureaucratic and slow to respond to the dynamic nature of development. Industries are faced with users who only know what they want after they see an initial version of the product. This means that requirements are not fully understood at the start of the project and change during the development phase. Based on this inequality and the need for faster product turnaround, organizations have started to implement empirical development processes such as the Agile methodologies.

In the first part of my present work I will introduce the core points of Agile Project Management as one of the new pioneers in the project management business area. After that in the next chapter I will also make a theoretical comparison between the Traditional Project Management and Agile Project Management highlighting the key ascertainments which form a contrast to the above mentioned practices, but we will see sometimes how these two methodologies can be merged and work effectively within the project. Here I also have to emphasize, that my work and the whole agile concept is concentrated on the software development business area since here I found agile methodology as a widespread practice. Although agile is known in other business spheres as well, firstly the software development sector implemented it successfully.

In the third and fourth part of my work I will bring into focus the practical side of the topic. I will introduce the IND Group where I carried out my study. The reason, why I chosen IND Group is quite versatile. First of all the company is one of the most successful Hungarian actors in the IT sphere leading the market with its phenomenal innovations, business and technical knowledge. The second reason of me choosing IND as the base company is my lucky situation of getting the opportunity to start my career at the company as a Junior Project Manager in August 2012. This means that during my work day-by-day I

practice those project management practices and tools what I have been learning. Moreover the company some years ago implemented the agile practices in its processes therefore nowadays we can find some projects using agile methodology. During my work I was always trying to draw conclusions by considering advantages and disadvantages of the different project management practices and in the end I found very interesting facts that I will present here in my work.

I carried out my research at the company by making extended personal interviews with competent project manager colleagues asking their opinion and experience of agile project management. Processing the interviews it makes me clear, what is the agile practice at the company, how it can be improved and mixed with the traditional and other project management methodologies.

2. THE MOST COMMON CONTROVERSIAL QUESTIONS IN THE PROJECT MANAGEMENT ENVIRONMENT

Our understanding of different kind of project management methodologies is constantly developing; however, there are still many misunderstandings and challenges that need to be cleared up. It is essential to clarify some of these misconceptions in order to fully understand the, limitations, benefits and requirements of implementing a project management strategy. In many organizations, chaos often results when a project's complexity is greater than its managers' ability to direct meaningful progress toward a goal. Progress might be made but it is often indiscernible and unsatisfactory. Here are some of the main barriers occurring during the process of different kind of projects.

2.1. ARISING CHALLENGES WHEN DEALING WITH DIFFERENT KINDS OF PROJECTS

- The demands are unknown, not clarified, unnecessary or contradictory.
- Changing the demands or adding new ones is difficult.
- Development is not customer oriented and the product does not satisfy the most important customer or market needs.
- High risk (because of using innovative technologies, inexperienced project team or high complexity of tasks).
- Projects often do not keep the planned deadlines.
- Documentation takes a lot of time.
- Integration takes place too soon which leads to unforeseen problems and delays.
- Testing is not complete due to the lack of time.
- Quality problems accumulate version after version
- Many failures are discovered too soon
- Communication between the stakeholders is complicated
- Conflicts and problems stay hidden and won't be solved

2.2. SOLUTIONS STILL NEED TO BE FOUND

- How can we handle constantly changing customers' needs with the lowest input
- How can we reduce overgrowing claims at the beginning of the analyze period
- How can we control the “real” progression of the project
- How can we recognize the unique and general risks
- How can we ensure transparency for every project stakeholder (project leader, team members, management, client)

For these questions we can almost completely find the answers in the conception of Agile project management.

“Agile thinking is an attempt to simplify things by reducing complexity of planning, by focusing on customer value, and by shaping a fruitful climate of participation and collaboration. There are a vast number of methods, techniques, best practices, that claim to be agile.” (Stober - Hansmann, 2010, p 35.) In the next chapter I want to give an overview of this new way of thinking in the project management sphere.

3. AGILE PROJECT MANAGEMENT (APM)

3.1. THE HISTORY OF AGILE PROJECT MANAGEMENT

“Agility is the ability to both create and respond to change in order to profit in a turbulent business environment” (Jim Highsmith)

As project management was becoming more and more important in the 1960s, the Project Management Institute (www.pmi.org) was founded in 1969 to advance the practice, science, and profession of Project Management. The PMI offers certification as certified project manager, which is currently the most widely recognized project management certification in the industry. The Project Management Institute also publishes A Guide to the Project Management Body of Knowledge (PMBOK® Guide), which contains the basic concepts and techniques for traditional project management. PMBOK® Guide was followed by many more new approaches re-thinking or totally differing from the traditional project management values. One of them was the concept of Agile Project Management which emerged to be the new way of handling project development tasks. The new approach triggered several debates about the contrariety with “The Traditional Way” however many experts said Agile Manifesto and PMBOK® are complementary.

Agile project management is a complex methodology which involves all the practices and values which are useful to be fast reacting, adaptive and ready to constantly develop when dealing with diverse projects. The concept “Agile” is not new - people have been doing different kinds of agile projects for a long time. Although, in today’s world, the movement to adopt more agile methodologies seems to be more and more important for several reasons:

- Some of the methodologies and tools for implementing agile projects such as Scrum, are becoming much more widely understood and accepted.
- Many companies have demonstrated success in implementing agile projects in actual practice and have added to the knowledge base of what works and what doesn’t work in different projects and environments.
- The tradeoffs associated with balancing a need for sufficient project control and achieving greater agility are also better understood, and there are many ways to achieve a balance of those objectives with the appropriate blend of agile and traditional methodologies.

The origins of much of today's agile movement can be traced to the Agile Manifesto, which was originally developed in February 2001, at The Lodge at Snowbird ski resort in the Wasatch Mountains of Utah.

Creating an agile team is usually motivated by benefits such as increased efficiency, innovation, and employee satisfaction. However the agile methodology is also encouraged by the accelerating complexity of development. As IT role in the modern economy has grown in importance, product developers have found themselves confronted with the challenges of exceptional complexity. An effective team needs to interact with and consider the viewpoints of a wide variety of stakeholders, many of whom have conflicting views on the product features and functionality. The team must then balance disparate needs of diverse stakeholders, a task far more challenging than merely fulfilling the functional requirements of a system. Therefore many developers welcome agile project management approach, which embraces these emerging realities.

The agile approach is built around self-organizing teams that organize their work themselves. Within these kinds of teams there is also a strict focus on cooperation and communication. Collaboration and coordination depend on communication, which is essential for successful development. These activities are supported through different agile practices.

3.2. MANIFESTO FOR AGILE PROJECT MANAGEMENT

Agile Manifesto defines four values and twelve principles. The four values are as follows: "We are uncovering better ways of developing software by doing it and helping others to do it. Through this work we have come to value" (Agile Manifesto):

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

"That is, while there is value in the items on the right, we value the items on the left more." (Agile Manifesto)

It's important not to misunderstand the purpose of these statements. The goal behind them was a shift in emphasis. That doesn't mean that the items on the right are unimportant:

"Over the years, the manifesto statements have been misinterpreted, primarily in confusing less important with unimportant: This should not be construed as indicating that tools,

process, documents, contracts, or plans are unimportant. Tools are critical to speeding development and reducing costs. Contracts are vital to initiating developer-customer relationships. Documentation aids communication. However, the items on the left are the most critical. Without skilled individuals, working products, close interaction with customers, and responsiveness to change, product delivery will be nearly impossible.” (Highsmith, 2002, p. 16.)

3.3. AGILE PRINCIPLES

The Agile Manifesto Principles elaborate on this overall value statement mentioned above. *Table 1.* is a summary of the Agile Manifesto Principles and their relationship to traditional project management practices.

Table 1: Agile Manifesto Principles and their relationship to traditional project management practices

Agile Manifesto Principle	Relationship to Traditional Project Management Practice
Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.	There is no real contradiction in principle; however, traditional practice in many cases has emphasized control, predictability, and thoroughness of testing over early delivery because the tradeoffs required to deliver early in many cases have been perceived to be severe. However, with the right approach and methodology, those tradeoffs may not be that severe and may be very manageable. Of course, the priority given to early delivery of software needs to be weighed against other factors in any project or business environment. For example, the process needs to be matched to the customer’s ability to actively engage in the project and provide feedback and inputs. It is difficult to make a statement that early and continuous delivery of software is always the highest priority in all projects
Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.	Change control has always been viewed as a very essential element of stabilizing a project, but a change control process does not necessarily have to be extremely rigid. If it is done properly, it does not equate to stifling or preventing change—it means ensuring that unnecessary change (as defined by the sponsor, ultimately) is rejected, but that necessary change is brought into the project with the full awareness of all concerned and that necessary adjustments to designs, plans, timescales, tests, contracts (etc.) are made, and with the minimum of wider disruption. Change control also can be valuable for configuration management and validating that any new changes are not inconsistent with other previously developed requirements and assumptions. For that reason, developing a more relaxed approach toward managing changes might not be acceptable in all projects.
Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.	Achieving these times with most traditional development methodologies may be difficult, and this is the area that agile really pays off; of course, the gains in this area need to be weighed against potential impact in other areas. It is certainly possible with traditional development approaches to break the project up into individual releases or iterations in order to deliver functionality early and that can be a very realistic approach in many cases; however, there is also typically some overhead involved in developing truly releasable software that should be considered.

Business people and developers must work together daily throughout the project.	This is a significantly increased role for the business people to play in the project that goes well beyond the typical requirements definition and testing phases of most traditional development methodologies. It also implies a very close and collaborative working relationship between the business and development staff and a strong commitment of the business side to fully participate in the project throughout the development process. Many business people are not prepared to make the kind of commitment that is needed and, in some cases, that level of participation may not be necessary. As a result, the project methodology needs to be tailored to the need for participation and the availability of business people to provide that participation.
Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.	There is no question that building a project around motivated individuals is a good thing for any project methodology; however, this particular principle might imply a higher level of delegation and trust than might normally be found in many traditional development projects. A higher level of delegation might involve some loss of control, but that could be a very acceptable tradeoff in the right environment with the right people on the project team. It does, of course, also imply that the people are well trained and capable of taking an increased level of responsibility. Some people have used this statement to justify having complete anarchy and chaos without any leadership at all and that clearly wasn't the intent behind it. Trust in motivated individuals is always a balance—you can move a long way back from micro-management without entirely sacrificing an appropriate level of leadership and management. Knowing what's on track, what's not on track, and deliberately deciding whether, when, and where to undertake corrective action (and what that should be) is essential to any project. Without that, you're "managing by hope" which is never a wise approach.
The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.	There is no question that maximizing the use of face-to-face communications is a good thing in any project. However, the ability to do that is dependent to some extent on the ability to co-locate members of the team to allow face-to-face communications and also might be limited by the size of the team. In many situations, tools such as desktop sharing (with or without video) and collaboration portals can alleviate some of the need for direct face-to-face communications for distributed teams especially when used with web-based development tools. This principle has often been interpreted to imply relaxing the emphasis on project documentation in favour of direct face-to-face communications. That can also be an acceptable tradeoff under the right circumstances based on project size, complexity, and other factors; however, some level of documentation may still be valuable and isn't inconsistent with an emphasis on face-to-face communications. Oral communication is great, and very quick, but places an overreliance on memory for information that may have a shelf life of no longer than a few days. Careful reading of the agile principle helps: it's about conveying information, not originating it, or keeping it available for future use. Not everyone who comes into contact with the product (or its components) is a member of the current development team. Think of the member of the support team in 5 years' time or other occasions when more formal and/or documented communication may be necessary to convey or preserve information outside of the development team or for future use after the development effort is finished.
Working software is the primary measure of progress.	There is no real contradiction with traditional development processes in this as long as the entire team (business and development) agrees on what "working software" means, what it consists of (including whatever documentation may be needed to support it), and what "done" means. What artifacts are deemed valuable by the customer apart from software? What makes up a working solution? Many traditional development processes fall short because the software is

	tested against documented requirements, and the requirements are either incomplete or inaccurate in reflecting the real needs of the users. What is implied here is “working software” in the eyes of the user, since the user is more directly involved throughout the development process. Working software is only the primary measure of progress when that’s the project’s only deliverable, which is probably true for a software coding team but not at all so for many projects. The primary measure of progress should be whatever the sponsor defines as value. Think about user training for example or process change, for example.
Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.	There is no real contradiction with traditional development processes, but this does imply that all the members of the team (both business and development) keep pace with each other throughout the whole duration of the project. In many traditional projects, the participants come in and out of the project as needed at different phases and, in some cases, that may be the best way to manage resources. In many projects, the business users may not have a sufficient capability to provide dedicated resources to participate in the project team on an ongoing basis and if the requirements are more certain and can be defined upfront, that level of participation may not be necessary. A key factor here is the motivation of the team—the theory behind this is that by empowering the team, they will be more highly motivated and energized and capable of more sustained development work. The need for leadership and skill to develop that kind of motivation is not limited to agile. It can also be done in a traditional environment if the development team understands the business need for additional control and is involved in the decision making behind the project.
Continuous attention to technical excellence and good design enhances agility.	There is no real contradiction with traditional development processes; however, it should be well understood that technical excellence and good design are not the only factors to determine a successful project development approach. This statement can be interpreted to be very developer-centric and needs to be understood in the context of the overall project goals.
Simplicity—the art of maximizing the amount of work not done—is essential.	There is no real contradiction with traditional development processes. This is basically the “keep it simple” principle. Projects should not be overburdened with overly complex features that may be unnecessary. The main advantages of simplification are speed and reduced complexity and reduced complexity can lead to better maintainability.
The best architectures, requirements, and designs emerge from self-organizing teams.	This is certainly a very arguable principle behind agile that will not always be true. For example, it is hard to make a categorical statement that the best architectures always emerge from self-organizing teams. The approach for determining the best architecture probably depends on the scope and complexity of the project. A similar argument could be made about requirements—this principle makes an implication that using self-organizing teams is the strongest factor in determining the quality of the requirements. It is certainly a factor, but not necessarily the only factor. Developing well-defined requirements that are testable and traceable to the design is a skill that is extremely important especially on large, complex projects.
At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.	There is no real contradiction with traditional development processes. This is probably a very strong point in favour of agile methodologies. Most methodologies emphasize the idea of a post-mortem to stop and reflect on what went right and what went wrong and to use that learning as a basis for ongoing continuous improvement of the process. A big advantage of the agile approach is that the sprints are much shorter so that learning and correction can take place much more frequently as the project progresses.

Source: Cobb, C. G., 2011, p. 40.

3.4. EXAMPLE FOR AN AGILE APPROACH

In the following chapter I collected some of the often used practices in agile teams and I tried to give examples of how it can work practically.

Table 2: Example for and agile approach

Practices	Example
Iterative and incremental	Create a software version which contains the most important data management logics, but not yet able to prepare detailed reports.
Continuous and close collaboration	Seat the programmers and the consultants in a common room
Frequent feedbacks	Furnish a development room at the client's office or ask her/him to visit us regularly.
Previously not predetermined changeable processes during the implementation	Make automatic test cases daily, and control the circumstances in every release. Every 2 weeks organize a "lessons learnt" meeting with the customer

Source: Self-edited table

3.5. TOP 10 OF HOW TO KNOW YOU ARE NOT DOING AGILE?

After collecting and understanding the basic concerns and issues of agile project management I would like to illustrate what is agile and what is not. For this I used a David Letterman-like Top 10 of how to know you are not doing agile listing by Alistair Cockburn. Alistair Cockburn is one of the initiators of the agile movement in software development, helping write the Manifesto for Agile Software Development in 2001, the agile Project Management Declaration of Interdependence in 2005, and co-founding the International Consortium for Agile in 2009.

As he describes in no particular order you know you are not doing agile if:

1. “The team is co-located, but people are not sitting within the length of a school bus to each other.
2. They’re distributed, and there is an absence of microphones and webcams and one or two meetings a day.
3. They have not delivered anything to real users in the last three months. Some of my agile friends would say that’s much too long, but I’m being generous there.
4. If no user has seen real running software inside the last month.
5. They don’t have the output of last month’s reflection workshop or retrospective on the wall.
6. They don’t have fully automated unit tests, and a large number of acceptance tests aren’t automated.
7. They’re not having a built integration at least once day. Good groups do it every half hour; there are groups that get away with it every other day.
8. They write big requirements documents, and they don’t know how to split those up into smaller pieces so they could deliver a piece of software every month.
9. They have itty-bitty requirements on the order of "here’s what happens when you click here," but they don’t have long-term vision for what they’re trying to accomplish.
10. People keep saying, "It’s not my job." One of the things about proper agile development is that there is group accountability for results. Very specifically, if the requirements are not coming in fast enough, whoever has a bit of spare time (programmers, testers, etc.) drops what they’re doing and helps gather requirements; if tests aren’t getting done, people with some spare capacity help test and so on.”

(Cockburn, 2010)

3.6. ADVANTAGES AND DISADVANTAGES OF USING AGILE PROJECT MANAGEMENT

Some think that being agile is just about some businesspeople sitting down with some developers over a box of pizza and starting develop code with no structure, process, tools, or methodology. In reality being agile does require a planned approach and a strict process for doing the work, and in a very fast-moving environment, having the right tools to manage all the upcoming tasks. Agile methodologies, by design, are not prescriptive. They don’t tell you exactly what needs to be done to implement the methodology or how to do it in many cases. In general, agile methodologies define some principles that intentionally

require interpretation for a given situation. This is the reason why in some cases there can be different perceptions of the methodology. Thus agile project management besides its wide range of advantages can have some disadvantages as well. In the following I assessed some of the most common advantages and disadvantages of using agile project management.

Disadvantages:

- Product owner sometimes is unreachable
- There is a lack of management support with the agile aspect
- The team members' agile approach is not sufficient
- The customer wants to change within the iteration
- It is hard to synchronize big, distributed teams
- It is difficult to form a complex end-product from the ones developed during the iterations
- There is a lack of device support for cooperation and automation (test, follow-up)

Advantages:

- High level of transparency for the team and the customer
- Early results; concentration of customers' needs (the client manages the development)
- Quick feedback cycles
- We can face problems at the early period of the project
- Motivation and success
- Agile practices can be used in non-agile projects as well
- Suitable in case of know-how transition and teambuilding

3.7. CASES WHEN AGILE METHODOLOGIES ARE DIFFICULT TO USE

After assessing the key elements of agile project management I collected some ideas which business actors have to take into account when thinking about agile transition. Implementing new methodologies is always a hard point for the company and the workers as well. Every transition starts with a preliminary assessment of needs, resources and capabilities which companies need to think about in order to adequately estimate the real possibil-

ities to change methodologies and processes. Here I collected some examples when companies can find hard to use agile project management approach:

- Unclear responsibilities
- Lack of management support
- Many customers with different requirements against one product
- There is no customer: brand new product on the market (e.g.: iPhone)
- The team members place their personal interests before the team's interest
- Lack of cooperation among the team members
- The customer does not require regularly deliverable product, just the end-product
- Offshore development

3.8. POPULAR FALLACIES ABOUT AGILE PROJECT MANAGEMENT

As a matter of fact agile approach is a brand new methodology to deal with workers, customers, resources and the project as a whole. Thus many misinterpretations have been already occurred regarding to agile documentation, change control, scope definition, quality assurance and other field of project management practical areas. In the followings I will list some misunderstood facts about agile approach and I also try to clarify them.

- Change is bad and need to be avoided
 - Change evolves feedbacks and more precise recognition
- Agile project can not be planned
 - Inversely! Cost and time is fixed. By definition there can not be delay in the deadline and cost can not be over limit
- All or nothing from the scope
 - An agile project delivers just the one that makes profit and eliminates the one which is useless
- The sequence of realizing the requirements can not be diversified arbitrarily
 - Inversely! Agile methodology enforces the real component driven solution and does not let the implied input to work

Table 3: Agile approach – Myth and Reality

Myth	Reality
Agile methodology can solve all problems	Agile methodology widens our problem-solving toolbar
We will never ever have to document anything	We are just dealing with documents aligned with the project size and complexity
Agile methodology can not be used for fix-price projects	The best way of making the most business value of fixed budget projects is the use of incremental development
We can realize the project earlier with less input	We must concentrate our energy in making the most business value in a safe way

Source: Hungarian Project Management Association (2009)

4. WHAT PMBOK® GUIDE PRACTICES CAN BE UTILIZED IN AGILE ENVIRONMENT?

In this chapter I will present the most important aspects of PMBOK® and I will make a comparison between the Traditional Project Management and Agile Project Management. In the end of this chapter I will make it clear that many elements of PMBOK® Guide are aligned with the Agile Project Management processes.

4.1. AGILE VERSUS TRADITIONAL PROJECT MANAGEMENT

4.1.1. Traditional approach - Waterfall model

There are several modifications of the waterfall model to address some of the disadvantages identified with the classic waterfall model. One method works with decomposing the implementation phase into multiple smaller phases with integration points called milestones.

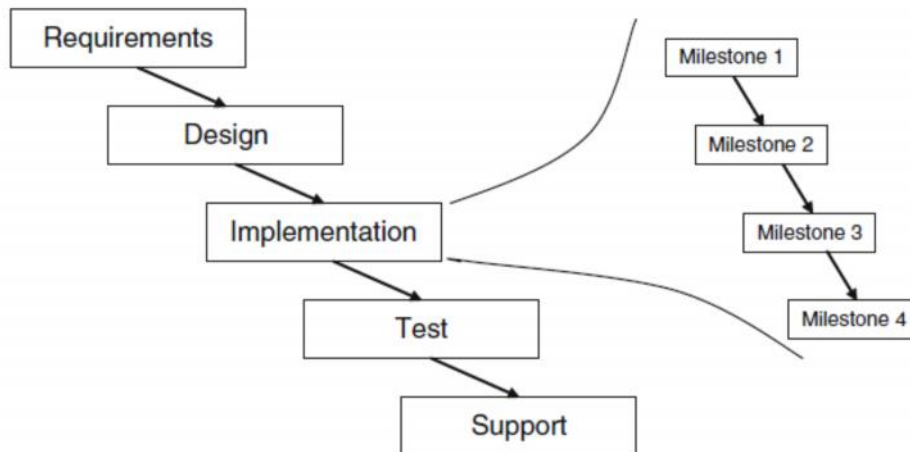


Figure 1: Modified waterfall modell - Implementation phase with milestones

Source: Stober – Hansmann (2010) p. 30.

4.1.2. Agile approach - Incremental development model

The incremental development approach developed the traditional waterfall model further by not only breaking the implementation phase into milestones but by taking the whole development process and repeating it several times on subsets of the overall project.

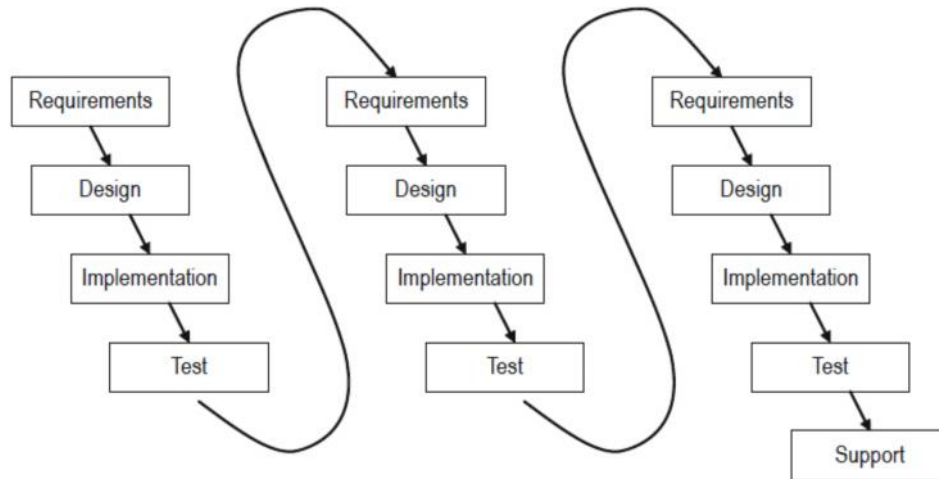


Figure 2: Incremental development approach

Source: Stober – Hansmann (2010) p. 31

Every iteration itself a project as defined by the PMBOK® Guide but just a phase of the overall project. That sometimes makes difficulties traditional project management practitioners go through as they make the transition from plan-driven approaches to the newer agile methodologies. *Figure 3.* shows us the Agile Project Lifecycle unwrapping and highlighting the iteration itself.

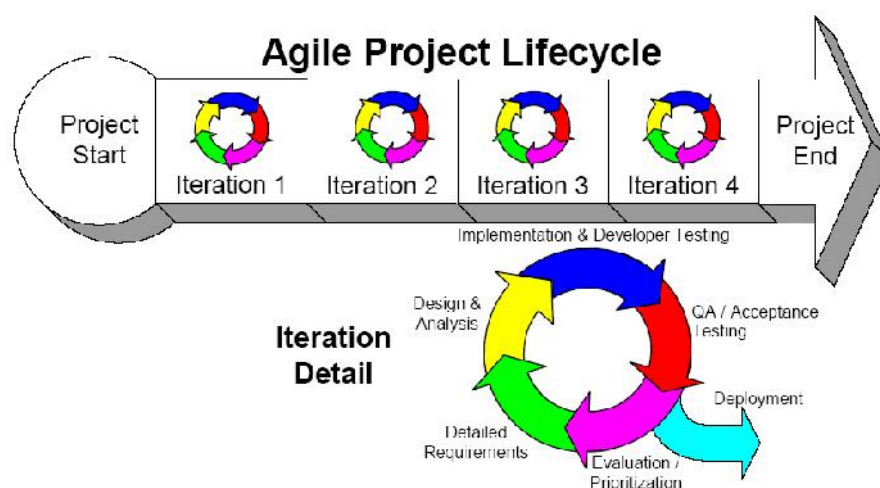


Figure 3: Agile project lifecycle

Source: Hungarian Project Management Association (2009)

4.1.3. Comparison of traditional and agile project management roles

The figures below show a comparison of the way roles that are typically implemented on traditional development projects and agile development projects. *Figure 4* shows the general roles in traditional projects and *Figure 5* shows the general roles in agile projects. Please note that these figures only show the roles and responsibilities within single teams for agile projects. An additional layer of project management would normally be needed to coordinate the efforts of projects that require multiple teams.

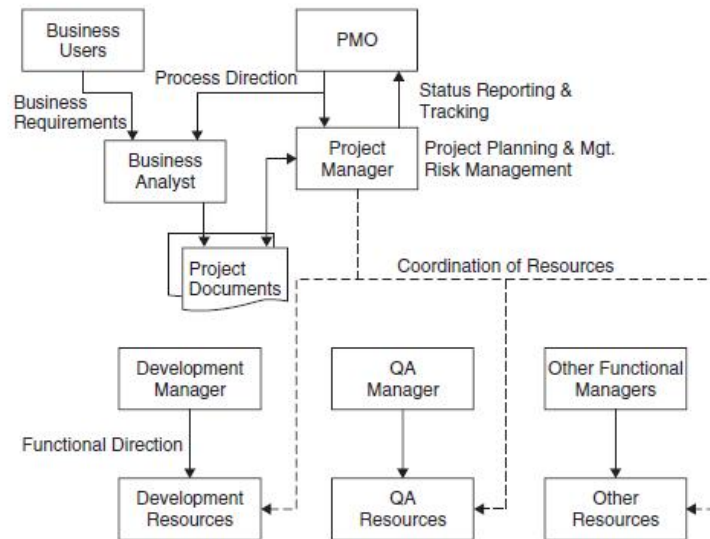


Figure 4: Typical role definition in traditional development projects

Source: Cobb (2011) p. 103.

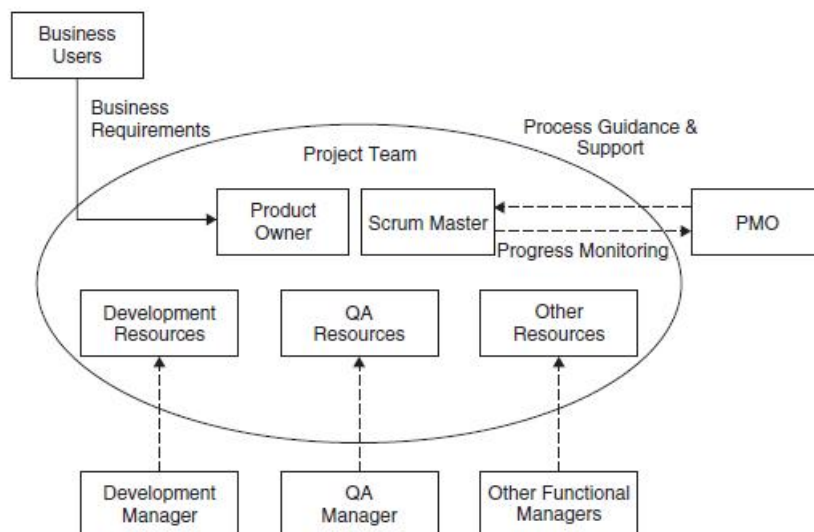


Figure 5: Typical role definitions in an agile project

Source: Cobb (2011) p. 104

In the following I summarized the key differences in an agile project (Cobb, 2011, p. 103.):

- A Project Management Office (PMO) is often not involved at all in the agile project. If a PMO were involved at all, it might play much more of a value-added process consulting role rather than a process control function in an agile environment.
- In some cases the Scrum Master replaces project manager functions performs team facilitation role and the team, as a whole, makes project decisions that are generally made individually by the project manager. The idea of involving the team in the decision-making phase is a good technique that many project managers use anyway on traditional projects.
- The Product Owner may relay some of the functions that might have been done by a Business Analyst to provide the comprehensive requirements definition for the project. Here I have to note that this role is a bit different from a Business Analyst role. The Product Owner is expected to be a subject matter expert, represents the interests of all the business stakeholders, and shows all-round business direction to the project. It is very feasible that a Business Analyst may also be required for the team to assist the Product Owner in performing that role.
- The functional department managers such as the Development Manager and the Quality Assurance Manager play a remarkably reduced role and provide only very limited functional direction to the team. The functional members of the team are expected to provide most of their own functional direction. Naturally, that requires fairly highly skilled people to fill those roles.

The table below shows a comparison of the typical project management and governance roles in traditional, plan-driven projects and agile projects

Table 4: typical project management and governance roles in traditional, plan-driven projects and agile projects

Responsibility	Traditional	Agile
Process Direction and Control	The PMO provides process direction and control (standardize the utilization of processes).	The Project Team is responsible for defining, implementing, and continuously improving the process. The PMO may play a value-added consulting or supporting role if necessary.
Progress Tracking	The Project Manager is responsible for tracking and reporting of project progress and control and management of overall costs and schedules. The PMO may provide overall oversight and control for all projects.	The Project Team is responsible for tracking and monitoring of their own progress. Progress reporting metrics such as the “burn down chart” are built into the process. The PMO may play a consolidation role.
Functional Management Responsibility for Resources	Functional managers (QA, Dev, etc.) provide direct management for all functional resources involved in projects. The Project Manager is responsible for defining and assigning tasks to functional resources and obtaining commitments for task performance from functional managers as necessary.	The Project Team is responsible for functional management of team resources, including assigning tasks, obtaining commitments, and making any functional decisions that may be needed. Functional managers may play a supporting role if necessary.
Coordination of Resources	The Project Manager is responsible for coordinating the availability of resources from supporting departments.	Most resources are assigned full time to the project; therefore, less resource coordination is needed. The Scrum Master will coordinate availability of shared resources from other departments.
Business Analysis and Requirements Elicitation	Requirements are documented and controlled. Business Analysts work directly with the users to analyse, define, and document requirements.	Documentation of requirements may be less formal and may be less controlled. The Product Owner is dedicated to the team and is the representative of user requirements to the team. The Product Owner might be trained as a Business Analyst or use Business Analysts to help if required
Change Management	The Project Manager is responsible for controlling and managing changes to the project requirements as the project progresses.	The Product Owner is responsible for continuously managing the Product Backlog in response to changes as the project progresses. The Product Owner may be assisted by the Scrum Master in performing that role. The Project Team and the Product Owner are responsible for continuously updating release plans as necessary as the project progresses in response to changes in the

Risk Management		Product Backlog.
	The Project Manager is responsible for risk management on behalf of the business stakeholders.	The Project Team is responsible for risk management. The Product Owner is expected to play a major role in risk management decisions on behalf of the Business Stakeholders.

Source: Cobb (2011) p. 104.

4.1.4. Merging Agility with Distribution

Addressing the problems related to the complexity of product development and the strong focus on collaboration, coordination and communication, are primary reasons for the growing interest in exploring the applicability of agile approaches in distributed development. Despite the popularity of the topic, the practice of agile development has been well ahead of research. “Thus, there is still no consensus or deep, theoretically grounded, understanding of the applicability of agile methods to different types of projects and the flexibility in application of agile methods necessary to realise the benefits promised.” (Šmite – Moe – Ågerfalk, 2010, p. 4.) The table below details the characteristics of agile versus traditional distributed product development.

Table 5: Characteristics of agile versus traditional distributed software development

Characteristics	Agile Development	Distributed Development
Communication	Informal Face-to-face Synchronous Many-to many	Formal Computer-mediated Often asynchronous Tunneled
Coordination	Change-driven Mutual adjustment self-management	Plan-driven Standardization
Control	Lightweight Cross-functional team	Command-and-control Clear separation of roles

Source: Šmite – Moe - Ågerfalk (2010) p. 4.

4.2. HIGH LEVEL OVERVIEW OF THE CORE ASPECTS

The main characteristics of Agile development and the PMBOK® Guide are discussed by process group below.

Table 6: Project processes from the angle of PMBOK® Guide and Agile

	PMBOK® Guide	Agile
<i>Initiating</i>	Authorization of new project or next project phase	Focus on identifying the business needs for new product development
<i>Planning</i>	Rolling wave planning for the whole project (sequential)	Preliminary whole project planning followed by separate planning for each iteration
<i>Executing</i>	Execution of project plan	Per iteration development of identified feature set.
<i>Controlling</i>	Emphasizes on change control management to minimize changes	Open to scope changes throughout the process with limitations on changes during iterations.
<i>Closing</i>	Formal acceptance.	Customer acceptance per iteration

Source: Udo – Koppensteiner (2003) p. 3.

Initiation:

In this starting phase the PMBOK® focuses on the formal approval in order to continue or start the project. That is why high level of authorization is important in this period. Moreover PMBOK® identifies the stakeholder and their needs, objectives and expectations, but does not go into details about stakeholder success expectation.

Agile methodology focuses on the prioritization of customer requirements and the way how the team is formed.

Planning:

The biggest difference between PMBOK® Guide and Agile can be recognised in the planning processes. PMBOK® puts a lot of effort in the planning processes. The number of planning processes demonstrates this: there are 11 core and 10 facilitating planning processes out of all 39 processes described in the PMBOK® Guide. The final product of the planning process is a project plan containing a schedule.

The general essence of Agile is small teams working in short iterative cycles comprised of planning, development, testing, and acceptance (Schwaber 2001). Therefore the planning processes in Agile are very simple. A preliminary estimate is made of each requested feature. These estimates are added up to the total amount of days it will take to complete the full project. Every iteration is a fixed amount of days and starts with a joint meeting be-

tween the customer and the development team. The customer prioritizes the requested feature list and the development team chooses the feature set it thinks it can develop within that iteration. The tasks for the iteration are planned and the workload is tracked on a regular basis against the remaining work and time left in the iteration. After the iteration, time to complete the full project is recalculated based on the results of the previous iteration and any changes the customer has made, including priority changes or adding or deleting features.

Execution:

PMBOK® Guide during the execution processes focuses on carrying out the project plan which was clearly defined in the Planning phase.

Agile focuses on carrying out the feature set promised to be delivered in the current iteration. During that time the project team is empowered to make decisions according to controlling, testing or correcting the product feature. The result of an iteration is a product increment.

Controlling:

The PMBOK® Guide controlling processes ensure that project objectives are met through frequent monitoring and measurement of progress to identify variances from plan so that corrective action can be taken when necessary (PMI 2000, p. 30). The scope is defined at the beginning of the project so strict change management control process is in use to assess the requirement changes. PMBOK's focus is on the cost reduction during the controlling period. Thus it recommends change control systems for scope, schedule, cost, and contract changes.

Agile methodologies focus on prioritizing the requirements and then define the scope per iteration. After each iteration features can be added or removed, but changes can not be made during iterations. This characteristic of Agile is based on the Agile philosophy that response to change is more important than following a plan. Agile methodologies focus on reducing the cost of change rather than reducing the amount of change. There is no need for change control processes since the customer usually sets priorities when changes occur.

Closing:

In PMBOK Guide important tasks are completion and settlement of contracts. The product of the project is signed off and the project and product information are archived.

Agile approach favours customer collaboration over contract negotiation. The client has the right and opportunity to review the product during the development phases. After each iteration the customer accepts the product feature and at the same time can make comments about the requirements and priorities for the next iteration. Documentation is seen as a requirement just like any other feature. As such it will need to be prioritized in the work queue. There is no standard formal closure process at the end of the project.

In conclusion, the biggest differences between PMBOK® Guide and Agile are in the planning and controlling phase. Even though there is overlap in the PMBOK® Guide process groups the methodology is based on first defining scope, second getting customer signoff on the scope definition and subsequently managing tasks against the scope definition including a strict change control system. Agile on the other hand starts with a preliminary planning and continues planning after each iteration. Scope changes are possible before every iteration, but usually not during iterations. Control is not based on scope but on remaining work hours. If the work needed is more than what is remaining, the customer is requested to drop features or add resources.

4.3. THE SIX KEY KNOWLEDGE AREAS IN THE PMBOK® GUIDE WHICH HAVE THEIR COUSINS IN AGILE

There are six key knowledge management areas defined in the PMBOK® Guide and deserving special attention. These are Project Integration Management, Project Scope Management and Project Time Management, Project Quality Management and Project Risk Management, and Project Human Resource Management. For each of these areas, the practices advocated by the PMBOK® Guide have their cousins in agile, but they are different in their implementation.

Table 7: Project management knowledge areas from the aspect of PMBOK® Guide and Agile

Knowledge areas	PMBOK® Guide	Agile
<i>Integration management</i>	Ensures that various elements of the project are properly coordinated.	Need for formal coordination is limited due to reduced use of processes in various elements.
<i>Scope management</i>	Ensures that project includes only the work required to complete the project successfully. Focuses on defining and controlling what is or is not in the project. Uses rigid change management processes.	Scope is only fixed when iterations are in progress. No formal scope control is needed.
<i>Time management</i>	Focuses on defining activities for project schedule and schedule control to ensure timely project completion.	Time per iteration is fixed. Focus is on delivering value (features) as fast as possible. Overall schedule is based on features instead of activities.
<i>Quality management</i>	Ensures that the project will satisfy the need for which it was undertaken. Focus is on conformance to requirements.	Project success criteria are set by the customers, which also deliver feedback after iterations. Focus is realization of purpose or fitness for use.
<i>HR management</i>	Processes to make most effective use of the people involved with the project.	Focus on team and not individual. Incentives are based on group productivity: e.g. jointly recognize tester and developers for a low defect ratio.
<i>Risk management</i>	Focuses on identifying, analyzing and responding to project risk.	The Agile methodologies treat risk management differently. There is not one common Agile approach.

Source: Udo – Koppensteiner (2003) p. 6.

Integration management

A key deliverable in integration management is the project plan document, prepared and owned by the project manager. In agile software development, with its emphasis on just-in-time design and participatory decision making, this activity translates into several different envisioning and planning exercises done on an iterative basis. Rather than defining all of the elements of a project plan at the beginning of the project. The agile project manager will focus on planning using a level of detail that is more appropriate and realistic for the time horizon. Waterfall's big up-front planning translates into agile's continuous planning, where gross-level features and estimates are planned at the beginning of a release. Low-ceremony, or low-tech documentation of agile release and iteration plans can consist of several white boards in a war room with color-coded markings, or it can be flip-chart paper taped to the wall with brightly coloured notes on each. The team must create, own, and commit to the plan. Release plans, iteration plans, and other planning outputs from the

initiating and planning phases are then shared with all stakeholders in the most highly visible way. Integrated change control also changes dramatically in agile methodologies. In keeping with the idea of minimum process to achieve maximum value, the change control process is streamlined and integrated into the daily routine of agile teams. At the end of each iteration, the working code that was developed is demonstrated, and feedback that may affect future decisions about the items in the backlog and ranking is gathered from stakeholders. Process changes are also made at the end of the iterations, allowing the team to make course corrections not only in the product, but also in the way they work. The team becomes the equivalent of a change control board. They streamline the process so that decisions are made quickly, collaboratively, and with little to no ceremony.

Scope management

Scope has always been the hard point for traditional project managers. Requirements constantly change in response to customer business needs, shifts in the industry, advances in technology, and things that were learned during the development process. Scope planning, definition, verification, and control are all knowledge areas that are given a great deal of attention in the PMBOK® Guide. These areas have great attention in agile methodologies as well but the philosophy on managing scope is completely different. Plan-driven approaches work hard to prevent changes in scope, whereas agile approaches expect and embrace scope change. The agile strategy is to fix cost and schedule and then work to implement the highest value features as defined by the customer, so that scope remains flexible.

This is contrasted against a typical waterfall approach, where features (scope) first are defined in detail, driving the cost and schedule estimates. Agile has simply flipped the triangle as follows:

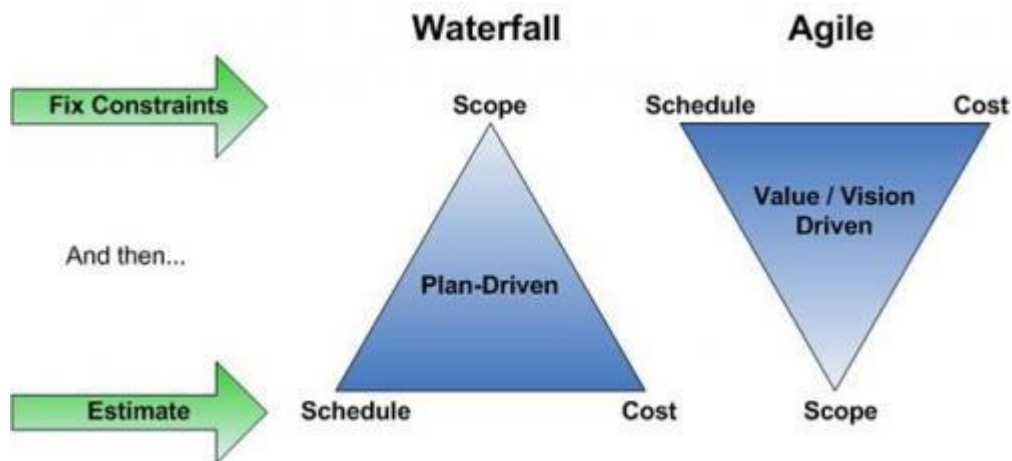


Figure 6: Waterfall versus Agile: The paradigm shift

Source: Sliger (2006)

Time management

Scope definition and many of the practices defined in the PMBOK® Guide knowledge area of Project Time Management are done as part of iteration planning, where features are elaborated, tasked out, estimated, and assigned. Again, planning and design work is done only for the features in that iteration, not on the entire system. Duration is no longer of interest in an agile project, because it is always the length of the iteration. Estimations are now based on the amount of work effort needed for each of the tasks, and it's the entire team that does the estimating. Team members sign up for the tasks instead of simply being assigned to them. This gives the team an opportunity to commit to the schedule and take ownership of the work.

We can see that PMBOK® Guide best practices aren't really that different from agile methods; they're simply done more often, iteratively and incrementally, with the attention to detail that's appropriate for the timeframe.

Risk management

The objectives of Project Risk Management are to increase the probability and impact of positive events and decrease the probability and impact of events adverse to the project. (PMI, 2000). The framework of the agile software development process fosters these objectives by making risk management an intrinsic part of the project lifecycle. Continuously identifying, analysing, monitoring, and responding to risk triggers and risk events are part of the agile team's iterative planning discussions, which is to say that risks are addressed by everybody all the time. Agile teams can perform these risk management activities either overtly or organically. Organic risk management is that which is intrinsic to the agile process, and risk management emerges out of iterative planning and review activities. Potential risks are instead labelled as "obstacles," "assumptions," or "concerns."

Quality management

In a change that pleasantly surprises most quality personnel, agile software development brings QA back into the analysis and design of the product, keeping QA heavily involved in decision making throughout the entire lifecycle. QA is now testing at the very beginning of the project lifecycle instead of waiting for something to be "thrown over the wall" at the end. And because throughput is increased as a result of agile's iterative and incremental development, QA finds itself with a need to become more technical as it must automate testing at all levels. Typically, quality planning includes defining how to perform quality assurance and quality control activities as they relate to standard QA practices, organization policies, and procedures. Quality assurance, with its focus on preventing defects, is translated into the agile practice of having committed QA resources on the development team that participate in daily decision making throughout the lifecycle of the project. Another aspect of quality assurance is the idea of continuous improvement, which the agile principles of constant feedback and responding to change support. Stopping points meant for inspection and adaptation, are built into all agile methodologies. Continuous improvement is practiced in planning meetings, daily activities, and iteration reviews and retrospectives. A recognized exercise in the iteration retrospective is determining what went well in the iteration, what did not go well, and what changes the team wishes to make in the next iteration. Quality control places its emphasis on finding defects that have already slipped into the system and working with developers to eliminate those defects.

5. INTRODUCTION OF THE IND GROUP

5.1. OVERVIEW

IND Group, with its unique philosophy for “The Banking Front-Office Technology” is a leading developer and provider of banking front-office products on the European e-finance market. As IND enters its second decade in operation they are witnessing a radical revolution in the global economy. The implications of the economic slowdown are widespread for banks: new business models, a changing competitive landscape and customer demand mean that adjusting business to new expectations has never been so important. As IND Group’s professional and business success is secured by their ability to adapt to the ever changing world of the e-finance market, the company can help its banking clients to accelerate these necessary and unavoidable changes in their frontline operations. IND Group adds value to its clients by giving them the tools to deliver a high-level customer experience and optimize sales-oriented processes with reduced acquisition and retention costs on all banking channels.

The main directions of IND developments:

- Banking Front-Office (E-banking sales channels and branch user interface on the same platform)
- Mobile channels in Banking and Trading
- Complex multi-channel e-business solutions
- Corporate and retail e-banking
- Portals and content management (CMS)
- Customer authentication

The company’s flagship product, the IND Banking Front-Office, is a comprehensive range of services that empowers front-office operations. IND BFO delivers a complete ‘sales and banking’ suite of channels including branch, internet, mobile and contact centre, all on the same centralized platform. With a foundation of profitable growth since its inception in 1997, IND Group is operating now from several countries: Austria, Germany, Hungary, Romania and UK. Through their local business partner network the company is represented in Bahrain, Dubai, Egypt, Italy, Switzerland and Ukraine.

5.2. IND GROUP – ALWAYS ONE STEP AHEAD THE DEMANDS

During its operation, IND was characterized by extraordinary growth. This proves that the structure created by the company is efficient: competence, persistence, client-orientation, flexibility based on knowledge and market-sensible preparedness are the key directions. Apart from the intention of its clients being able to have recourse to high quality services through all internet channels, the main conditions of success have been the appropriate and accurate recognition of market requirements, moreover the specification by these. The company wanted more than just ride the current trends, they wanted to create them. IND's aim is to facilitate the activity of their clients and to support their client handling, transactions and also their sales process with the help of IND innovative products. The company's Innovation Lab monitors and uses all of the web technologies and is trying to cover every possible internet based solutions. In accordance with the demands of company costumers they are able to supply everything from web applications to highly secure E-commerce and E-banking systems. In order to keep the company's innovation they have used most of their fast growing profit to update and develop their technical background. Over the past 5 years IND Group has delivered more than 100 projects, like online banking, contact centre solutions, mobile banking, and a series of branch renewal solutions, like loan origination, or CRM transactional front-end.

5.3. INTERNATIONAL PRESENCE AND CLIENTS

During the past decade IND has gained significant international experiences. The company delivered its solutions to clients in many different cultures and in many different languages, and they were performing well in every country. Development work takes place mainly in the Miskolc headquarters. The office in Budapest handles Hungarian sales, and the task of our German and Austrian colleagues and UK colleagues is the requirement analysis of international demands. The company is a major player on the European e-finance market. 27 banks, brokerage firms and insurance companies use our products, including the biggest corporations of the European banking industry. The clients of IND Group finished top in international competitions with IND software technology.



Figure 7: International presence and clients

Source: Internal company document

IND employs 160 people, out of which around 110 experts are directly involved in the design, programming, testing and support of the IND Banking Front-Office. There are close to 50 people who are exclusively involved in the product development, more than 60 people in the Professional Services team (making the project implementations) and 12 people covering the support services. All experts have an average of 4 years experience in delivering internet and mobile banking solutions.

5.5. CORPORATE MILESTONES

IND (Interactive Net Design) Group was founded by 4 university students in Miskolc in 1997. The management is very proud that the company has kept its 100% Hungarian ownership and they can operate in the market with self-financed without any external investment need. The remarkable growth IND has made over the past years marks a true success story in the international financial IT market. The company provides proficiency and state-of-the-art technology in more than 15 countries on three continents.

Table 8: Corporate Milestones

1997	IND Group was founded in 1997 by Roland Pecsénye, Balázs Vinnai and József Nyíri. The focus of the new company was immediately on e-business technology and web solutions.
1999	New strategic partnership and multichannel focus - Strong partnership with Brokat Technologies AG Focus: multi-channel banking technology
2001	Multi-country focus. Multi-country Internet Banking delivered (HVB Group, 8 countries)
2002	Multi-channel banking. Multi-channel banking delivered to Bank of Valletta. Quote: “The fastest growing technology company in Central- and Eastern Europe” (DT)
2003	New target market: Middle-East. New clients in the Middle-East (Arab Bank, Riyadh Bank). Acquisition of Datadesign GmbH (former Brokat Technologies AG)
2004	Market leader in Hungary. IND Group becomes the market leader of online banking provider in Hungarian market, with clients such as MKB, CIB, Fundamenta and others
2005	IND Group reached the milestone of 20 implementations. The strengths of the company are online banking, online brokerage, contact center banking and WAP banking
2006	Start to designing IND Banking Front-Office and renewal of the product portfolio
2007	IND Innovation Lab was founded. Business Partner Network expansion (CEE and ME regions). Venture Capital Investments (Euroventures)
2008	IND iMobile Banking was launched. New offices opened in London and Bucharest. IND Group wins the Technology Innovation Award / Innovation Grand Prize.
2009	PFM: IND Group launched its Personal Finance Manager product. First customer in Romania: Banca Transilvania.
2010	First comprehensive virtual bank project: Zuno Bank
2011	Launch of IND Gamification concept, a unique enhancement of PFM with social elements and loyalty programs. Launching BT24 project on Romanian market.
2012	World’s First Facebook transfer initiated from online banking @ Alior Sync bank; Second Romanian customer ProCredit Bank. At the e-Finance Awards Gala 2012

	in Bucharest, IND Group have received the ‘E-payments’-award for the development of the best e-banking and m-banking solution in Romania for Banca Transilvania.
--	--

Source: Internal company document

5.6. AWARDS

The company with its international widespread presence, and wide range of customer network always becomes proud to hear and announce all the awards that they or one of the clients have received.

Table 9: IND and client awards on a yearly basis

2009	• KNIGHT’S CROSS ORDER OF MERIT OF THE REPUBLIC OF HUNGARY (civil division) to dr. Balazs Vinnai • ‘Technology Innovation Award’ for IND iMobile Banking / ‘Innovation Grand Prize’by Hungarian Association for Innovation • CIB Internet Bank: ‘Best Retail Internet Bank’ – Gold-level Award / Privatbankar • Unicredit: ‘Best Bank Website’ – Gold-level Award / Privatbankar • "IT and Consulting Partner of the Year" of CIB Bank a Bank of Intesa Sanpaolo • "Innovation Partner of the Year" of CIB Bank for mobilCIB (IND iMobile Banking)
2008	• eFestival 2008 awards: in category „Tourism/Settlement“ for the implementation of tourism portal “vendegvaro.hu” and in category „e-Services“ for the multifunctional banking portal of Unicredit Bank. • CIB Bank: „Bank of the Year 2008 – 1st place” / Mastercard • CIB Bank: „Electronic Retail Banking Service of the Year 2008 –1st place”(among others for mobilCIB) / Mastercard
2007	• The Hungarian Venture Capital and Private Equity Association awarded the IND Group as the „Company of the Year” • The CIB Internet Bank won the „Innovative Bank of the Year” and the „Bank of the Year” award on MasterCard’s – The Bank of the Year competition

2006	• The CIB Internet Bank won the „Direct Banking Service of the Year” award on MasterCard’s – The Bank of the Year competition. • On the Ernst & Young Entrepreneur Of The Year contest Dr. Balázs Vinnai, the CEO of IND Group won the „Promise of the Future” award.
2005	• CIB portal awarded again as “The homepage of the year”.
2004	• The IND STS obtained the certificate: "Hungarian Innovation Award – new solutions: product of 2004". • CIB portal awarded as “The homepage of the year”.
2003	• Index.hu, HP, International Banker School Center: CIB Internet Bank is the "Internet Bank of the Year in Hungary". • Bucharest Business Week’s Financial Awards for Excellence 2003: HVB Bank wins “Best e-banking product in Rumania” award with IND technology.
2002	• CIB Bank won the competition of Privátbankár.hu: “Financial services from the point of view of the client”. • Index.hu, HP, International Banker School Center: CIB Internet Bank is the "Internet Bank of the Year in Hungary". • IND was selected as number one on the Deloitte & Touche Fast 50 list of companies in Central Eastern Europe, winning the title: “The fastest growing technological company in Central- and Eastern-Europe”.

Source: Internal company document

6. AGILE PRACTICE AT IND GROUP

In this chapter I will demonstrate the Agile practices and tools used in the IND Group. I carried out my research through deep and long interviews with the project leader colleagues. The main topic of the interview was the agile implementation to the projects with a strong focus on the agile practices, tools and responsibilities. I also consulted with some colleagues from the developer teams using agile methodologies as I wanted to get first-hand experience of using agile tools and I found the direct verbal contact with the actors would be the best way to receive such information. As I collected and processed the interview answers, it made me clear what kind of agile practices are in use at the company. I have come to the conclusion that IND agile methodology is the Scrum package which is a well-known agile software development method. In the following I will demonstrate the need of changing to scrum methodology after that I will shortly detail the transition period and then I will show the scrum roles, and scrum artefacts.

6.1. THE REASON

There were several reasons why the company about 2 years ago implemented agile methodologies to the projects. One of them was the fast pacing customer business environment which required quick responding software development and product delivery. Secondly the management came to the conclusion of having unmotivated project groups because of long lasting projects with no visible working product which would have given success and motivation to the workers. The third reason was that the company in the last period widened its product portfolio in such a big scale, that customer orientation became the first priority to meet clients' requirements. It was hard to respond to the fast changing environment with the previous project management methodologies and company experts started to think about innovative ways to renew project processes and tools.

6.2. THE TRANSITION

The transition was started with a deep analysis to point out what worked well and what worked wrong. The company also carried out some benchmarking analysis and after outlining the weak areas, and the processes and tools in need of improvement they hired a

coaching company to implement the chosen software development and project management processes.

Basically there were two types of teams and working cycles where IND experts defined a foreseeable agile transition. One was the Platform team which was responsible for developing and managing the company product portfolio as a whole. The second was the development team assigned to a particular project. The project team generally develops according to special client needs. There is collaboration between the two teams which we call *Product Traceback*. The topic of product traceback and its process would deserve a single chapter to discuss but this is not the subject of my current work.

The complete transition was eventually carried out during half a year. The platform team nowadays still works with agile methodologies and there is one project team whose client accepted the agile methodology so the team could also start to implement scrum processes. Here I need to highlight, that launching an agile development project is always a 2 party decision, because the client also has to become committed to the new process which involves frequent communication, continuous feedback, automated tests and many more new tasks from client side as well.

6.3. LET'S DO IT AGILE – SCRUM, THE AGILE KNOWLEDGE PACKAGE

The company was very excited about the adoption of Agile (adaptive) project management and development and about the adoption of Scrum in particular.

Scrum is an incremental and iterative software development method for handling software projects and development, and is one of the most commonly used agile methods. Unlike other agile methods, scrum is not an acronym. The method was coined as such by Hirotaka Takeuchi and Ikujiro Nonaka in a HBR article titled “The New Product Development Game” written in 1986, in reference the sport of rugby. The process involved was developed by Ken Schwaber and Dr. Jeff Sutherland.

Scrum focuses on projects where it is difficult to plan ahead. It represents a radically new approach for planning and managing projects, bringing decision-making authority to the level of operation properties and certainties. Scrum has been used for a variety of products, but has initially been most popular for software products using object-oriented technologies. It is particularly suited to high risk endeavors where traditional efficiency concerns are secondary to the ability to deliver the right product, or any product, by the re-

quired date. However scrum is generally a software development method nowadays it is also used to produce financial products as well as medical products.

6.3.1. The Scrum Guidelines

Here I collected the most important guidelines said by project managers who took part in the agile transition and still work with scrum

- Iterative
- Strongly time-boxed
- Empirical
- Non-hierarchical
- Always keeps sustainable pace
- Focuses on strict quality requirements of the deliverables
- Focuses on continuous integration
- Prioritization (both from project team and from customer side)
- Fast respond to change
- Verbal communication
- Open and common work area

6.3.2. Scrum project roles

6.3.2.1. Product owner

Product owners define the “what” of product. That is, they describe and prioritize what they want done based on the vision of the product and their role of maximizing the return on investment of what is developed. The Product Owner basically represents the stakeholders and is the voice of the customer. The product owner is the person responsible for communicating the customer requirements. He or she is accountable for ensuring that the team delivers value to the business.

While considered part of the customer unit, the product owner role is critical to the success of an agile product development effort. In addition to communicating requirements, the product owner is responsible for defining user acceptance criteria for requirements, making the prioritization, as well as collaborating with the development unit throughout the iterative development cycle. Scrum teams should have one Product Owner.

6.3.2.2. Team

Scrum teams define the how and how long of the product. No one else gets to estimate development work or gets to make changes to the estimates except for the developers. Scrum teams determine the effort estimates for the product function items on which they work. More importantly, they determine how much work they can reliably do during a sprint. No one else defines how much work a team will do during a sprint—that has to come from the team itself.

The Team is responsible for delivering potentially shippable product increments. A Team is made up of 3–9 people with cross-functional skills who do the actual work (analyse, design, develop, test, technical communication, document, etc.). The Team in Scrum is self-organizing, even though they may interface with project management organizations (PMOs). It is very important to mention that there is no hierarchy within the team, and the accountability is distributed.

A team member is anyone on a scrum team who is not a Product Owner or Scrum Master, who is responsible for specific iteration deliverables and is accountable for contributing to team iteration commitments and goals. A team member may be a developer, a technical writer, an architect, quality analyst or any other role essential to the production and delivery of working software. Scrum teams work best when the team members are 100% dedicated to their teams. Studies have clearly demonstrated that attempting to work on multiple projects at the same time significantly reduces their effectiveness, quality, and productivity.

6.3.2.3. Scrum Master

A key role in the Scrum product development framework, the Scrum Master is the person who is primarily responsible for facilitating all scrum meetings, removing team impediments, protecting teams from external distractions, keeping the team honest and on-track to ensure that the team is best able to deliver against the sprint goals. As scrum teams are self-organizing and self-managed, it is important to differentiate between a Scrum Master and a traditional manager. Rather than attempt to manage the scrum team, effective Scrum Masters work for the team, and are often best described as servant-leaders. The Scrum Master is not the team leader, but acts as a buffer between the team and any distracting influences. The Scrum Master ensures that the Scrum process is used as intended. The

Scrum Master is the enforcer of rules. A key part of the Scrum Master's role is to protect the Development Team and keep it focused on the tasks at hand. Many a time, a Scrum Master is mistaken for a Project Manager. The difference is that, while the Project Manager may also have people management responsibilities in addition to the role of a Scrum Master, a Scrum Master shall not hold any such additional people responsibilities.

As I was told in the company scrum masters are developers who do their scrum role in 20% of their work and in the other part (80%) they do the basic developer team roles.

There was a very interesting way of assigning the scrum master at the company for a particular project as one of the company project managers explained to me. They asked the team directly who wants to be the scrum master. The ones who raised their hands definitely didn't get the position and from the ones staying in the background were selected to the scrum master. There is another option as the coaching firm suggested. If a project has a female team member we need to assign her to be the scrum master.

6.3.2.4. Other stakeholders of the scrum driven project

The above mentioned roles are the so called "pig roles". Other ancillary roles are the so called "chicken roles". Here is a short story about the chickens and pigs:

A pig and a chicken are walking down a road. The chicken looks at the pig and says "Hey, why don't we open a restaurant?" The pig looks back at the chicken and says "Good idea, what do you want to call it?" The chicken thinks about it and says "Why don't we call it 'Ham and Eggs?'" "I don't think so" says the pig, "I'd be committed but you'd only be involved."

So the "pigs" are committed to building software regularly and frequently, while everyone else is a "chicken" - interested in the project but really indifferent because if it fails they are not the pigs - that is, they were not the ones that committed to doing it. The needs, desires, ideas and influences of the chicken roles are taken into account, but are not in any way allowed to affect, distort or get in the way of the actual Scrum project.

“Chicken roles” in Scrum:

- End users
- Client
- Project Management Office (PMO)
- Project manager
- Consulting expert
- Architect
- Other stakeholders

In this chapter we mustn't be regardless of the project manager role. However agile and scrum methodology decline the project manager role on the scrum processes, at IND it works a bit differently. As I mentioned above, there is one client oriented project which uses the scrum practices. Here there is a project manager assigned to control and handle some project tasks which is also different from traditional project manager tasks. The agile project manager basically is not the boss of the team who is commanding and controlling the team. He is more like a facilitator who would protect and serve the team in order to giving full effort to develop quality software on time.

What does the agile project Manager NOT do?

- The agile project manager does not manage the team
- The agile project manager does not direct team-members
- The agile project manager does not assign tasks
- The agile project manager does not “drive the team” to hit its goals
- The agile project manager does not make decisions for the team
- The agile project manager does not overrule team-members
- The agile project manager does not direct product strategy, decide technical issues...etc.

6.3.3. Scrum Artefacts

6.3.3.1. *Product backlog*

The product backlog is a prioritized and estimated list of all outstanding product/project requirements, features, defects and other work items. The product backlog is typically owned and managed by the product owner who reviews it on a regular cadence to ensure that the development unit is focusing on the completion of those items that represent the highest impact on the overall product value. (ASPE SDLC Training, 2010, p. 18.).

During the sprint planning meeting the product owner prioritizes the items in the product backlog and describes them to the team. The team then determines which items they can complete during the coming sprint (iteration). The team then moves items from the product backlog to the sprint backlog. In doing they expand each product backlog item into one or more sprint backlog tasks so they can more effectively share work during the sprint. Conceptually, the team starts at the top of the prioritized product backlog list and draws a line after the lowest of the high priority items they feel they can complete. In practice it is not unusual to see a team select, for example, the top five items and then two items from lower on the list but that are associated with the initial five. Product backlog items can be technical tasks or more user-centric.

The product backlog can be maintained in an Excel spreadsheet as IND scrum project team does. Their Excel spreadsheet shows each product backlog item assigned a general priority (very high, high, etc.) by the product owner. Estimates have been developed by the developers but it is understood that they are very imprecise and are useful only for rough assignments of tasks into the various sprints.

6.3.3.2. *Iteration or sprint backlog*

First of all here I need to show a short definition of the Sprint: A time-boxed cycle, or iteration, during which the development team is supposed to turn some selected requirements (user stories) into a potentially shippable product increment. (Pham – Pham, 2010, p. 272.).

The iteration or sprint backlog is a subset of user stories from the product backlog that contains the planned scope of a specific iteration. Generally, the iteration backlog reflects the priority and order of the release plan and product roadmap.

The sprint backlog is the list of work the development team must address during the next sprint. The list is derived by selecting stories/features from the top of the product backlog

until the development team feels it has enough work to fill the sprint. This is done by the development team asking "Can we also do this?" and adding stories/features to the sprint backlog. The development team should keep in mind the velocity of its previous sprints when selecting features for the new sprint, and use this number as a guide line of how much "effort" they can complete.

The features are broken down into tasks by the development team, which, as a best practice, should normally be between four and sixteen hours of work. With this level of detail the development team understands exactly what to do, and potentially, anyone can pick a task from the list. Tasks on the sprint backlog are never assigned; rather, tasks are signed up for by the team members as needed during the daily scrum, according to the set priority and the development team member skills. This promotes self-organization of the development team, and developer buy-in. The sprint backlog is the property of the development team, and all included estimates are provided by the development team. Often an accompanying task board is used to see and change the state of the tasks of the current sprint, like "to do", "in progress" and "done".

6.3.4. Meetings

Since agile methodology strictly focuses on the frequent and physical communication therefore scrum defined some meeting forms which need to be applied during scrum processes. At the company I received first-hand experience because I had the chance to attend some of the meetings as an observer. In the following I will show theoretical background of these meetings and I also present the IND experience.

6.3.4.1. *Daily scrum meeting*

The Daily Scrum, also referred to as “the daily stand-up”, is a brief, daily communication and planning forum, in which agile/scrum teams come together to evaluate the health and progress of the iteration/sprint. It is also considered to be the fifth and final level of the agile planning process. As a daily, team planning meeting, an effective daily scrum should be a tightly focused and time boxed meeting that occurs at the same time and place, on a daily basis. (ASPE SDLC Training, 2010, p. 7.). The intent of the daily scrum is to better understand the progress of the iteration, by all contributing team members honestly answering the following three questions:

- What did I accomplish yesterday?
- What will I commit to, or complete, today?
- What impediments or obstacles are preventing me from meeting my commitments?

Conversation in these meetings should remain focused on answering these three questions only. For the sake of brevity and greater team efficiency, additional discussion stemming from these three questions should be handled independently of the daily scrum, and should be limited to those team members who are directly involved.

This meeting has specific guidelines:

- All members of the development team come prepared with the updates for the meeting
- The meeting starts precisely on time even if some development team members are missing
- The meeting should happen at the same location and same time every day
- The meeting length is set (timeboxed) to 15 minutes
- All are welcome, but normally only the core roles speak

6.3.4.2. *Backlog grooming*

The team should spend time during a sprint doing product backlog grooming. This is the process of estimating the existing backlog using effort/points, refining the acceptance criteria for individual stories, and breaking larger stories into smaller stories. Meetings should not be longer than an hour. Meeting does not include breaking stories into tasks. The team can decide how many meetings are needed per week. The Scrum Guide time-box for grooming is 10% of the sprint (so 1 day for a 2 week sprint, 2 days for a 4 week sprint...etc.). IND agile team only spends about 1-2 hours a week once they get used to the practice.

Typical activities at IND weekly backlog grooming meetings:

- Discussing product backlog items
- Splitting product backlog items
- Showing and stabilizing acceptance tests for product backlog items
- Estimating product backlog items
- Looking further down the road by identifying and discussing external dependencies

6.3.4.3. *Sprint planning meeting*

It is a two-phase meeting. During the first part, the product owner lets the team know which requirements he/she thinks should be part of the sprint, and during the second part, the team decides how to turn the selected requirements into an increment of potentially shippable product. (Pham – Pham, 2010, p. 272.)

At the beginning of the sprint cycle (every 7–30 days), the sprint planning meeting is held.

Typical activities at IND sprint planning meeting:

- Select what work is to be done
- Prepare the Sprint Backlog that details the time it will take to do that work, with the entire team
- Identify and communicate how much of the work is likely to be done during the current sprint
- Eight-hour time limit
 - (1st four hours) entire team: dialog for prioritizing the product backlog
 - (2nd four hours) development team: hashing out a plan for the sprint, resulting in the sprint backlog

At the end of a sprint cycle, two meetings are held:

- Sprint review meeting: Meeting during which the team provides a demonstration of what they have built to the product owner and the stakeholders.
- Sprint retrospective: Meeting during which the scrum team will go through what worked and what did not work during the sprint they just finished and determine whether there is anything they can learn from their experience that will make the process even better for the next sprint.

7. CONCLUSION

According to the agile movement, software development is not intended to produce repeatable results. It is not a “get it right the first time” effort. There are a too many unknowns at the start of software projects to be able to fix the scope and plan the project to completion. New business processes, development languages or architectures are used that have never been used before. Most projects are attempting something that has never been done before, using the latest tools and architectural approaches. The effects of implementation are unclear due to integrations with old customized legacy systems. In this situation, rapid development, test and fix cycles will produce a better product than focusing on perfecting the design and then performing the work. Many short development cycles maximize the learning potential for the organization and the development team. In agile an important integration point is that customers have many feedback opportunities.

Don’t throw the PMBOK Guide out of the window, but combine the knowledge and experience with the new agile approaches. Companies need to let go of the rigid control systems. The project manager’s role needs to change from “police officer” controlling the “law” to managing relationships and empowering the team. He or she needs to create an environment for open communication, with faster responses to market changes and new demands.

I found agile case at IND a very successful implementation and good step to improve software development and project processes. As the teams still effectively use agile and scrum practices, it made the management clear that the new methodology has many ways to better serve the customers’ requirements and contribute to overall company business success. I can conclude that the agile/scrum practices eventually proved up. And it is not yet over. The company continuously improves its processes resources and tools in order to strive for excellence.

In the future I would like to widen the subject of my current work and make a deeper research at the company. I would like to make a comparison between agile project methodology and RUP (Unified Rational Process) methodology which is also a standalone project management and software development package in use at the company and also has the particular pros and cons. Further I would like to widen IND’s and my agile knowledge base by examining larger companies where multisite project work is always an issue. Another important topic of my future work will be the project manager’s role in agile project.

The official literature sometimes excludes the project manager from the agile team, however I have already seen effectively working projects with an integrated, core work of a project manager.

BIBLIOGRAPHY

Official Literature:

1. ASPE SDLC Training (2010): *Agile Glossary*. Davisbase, Springville, 32 p.
2. Augustine, S. (2005): *Managing Agile Projects*. Prentice Hall, Annandale, Virginia, 253 p.
3. Cobb, C. G. (2011): *Making Sense of Agile Project Management: Balancing Control and Agility*. John Wiley & Sons, Hoboken, New Jersey, 245p.
4. Cockburn A. (2010): *What's agile, what's not*. Search Software Quality, <http://searchsoftwarequality.techtarget.com/news/1255480/Alistair-Cockburn-on-whats-agile-whats-not> (downloaded: 12.06.2012.)
5. Highsmith, J. (2002): *Agile Project Management*. Addison Wesley, Boston.
6. Hungarian Project Management Association (2009): *Circuit switching, 12th conference*. PMSZ, http://www.pmsz.hu/system/files/korkapcs_12Last.pdf (downloaded: 14.09.2012)
7. Larmann, C. – Vodde (2010): *Scaling Lean and Agile Development: Thinking and Organizational Tools for Large-Scale Scrum*. Pearson Education, Boston, 348 p.
8. Pham, A. – Pham, P.-V. (2010): *Scrum in Action: Agile Software Project Management and Development*. Course Technology, Boston, 321 p.
9. Pries, K. H. – Quigley, J. M. (2011): *Scrum Project Management*. CRC Press, Boca Raton, 196 p.
10. Project Management Institute (2000) *A guide to the project management body of knowledge (PMBOK® Guide)*. Project Management Institute, Newtown Square, 496 p.
11. Schwaber, K – Beedle - M. & Martin - R.C. (2001), *Agile Software Development with SCRUM*. Prentice Hall, Upper Saddle River, 175 p.
12. Sliger, M. (2006): *Relating PMBOK Practices to Agile Practices*. StickyMinds, <http://www.stickyminds.com/sitewide.asp?ObjectId=10365&Function=DETAILBROWSE&ObjectType=COL&sqry=%2AZ%28SM%29%2AJ%28MIXED%29%2AR%28relevance%29%2AK%28simplesite%29%2AF%28Relating+PMBOK+Practic-es+to+Agile+Practices%29%2A&sidx=3&sopp=10&sitewide.asp?sid=1&sqry=%2AZ%28SM%29%2AJ%28MIXED%29%2AR%28relevance%29%2AK%28simp>

le-

site%29%2AF%28Relating+PMBOK+Practices+to+Agile+Practices%29%2A&si
dx=3&sopp=10 (downloaded: 2012.08.22.)

13. Sliger, M. – Broderick S. (2007): *The Software Project Manager's Bridge to Agility*. Pearson, Boston, 354 p.
14. Šmite, D. - Moe, N. B. - Ågerfalk, P. J. (Eds.) (2010): *Agility Across Time and Space*. Springer, s.l., 341 p.
15. Stober, T. – Hansmann, U. (2010): *Agile Software Development*. Springer, Heidelberg, 179 p.
16. Udo, N. – Koppensteiner, S. (2003): *Agile form a PMBOK Guide perspective (Abstract)*. Projectway, s.l., 7p.
17. Vereczki, B. (2009): *Agile Software Development (v0.1)*. s.n., s.l., 3p.

Internet sources:

1. Agile Manifesto: <http://agilemanifesto.org> (downloaded: 12.07.2012)
2. Agile Alliance: <http://www.agilealliance.org/> (downloaded: 12.07.2012)
3. Agilis Szoftverfejlesztők Egyesülete: <http://www.agilealliance.hu/> (downloaded: 12.07.2012)
4. Traditional vs. Agile Project Management:
<http://pmpalyazat.wordpress.com/2011/08/13/hagyomanyos-vs-agilis-projektmenedzsment/> (downloaded: 07.10.2012)