

Daria Tikhomirova


PROJECT MANAGEMENT IN THE
IMPLEMENTATION OF A NEW
FUNCTIONALITY IN A CRM SYS-
TEM USING THE SCRUM METH-
ODOLOGY

Bachelor's Thesis
Information Technology

May 2016



DESCRIPTION

		Date of the bachelor's thesis 30.05.2016.
Author(s) Daria Tikhomirova	Degree programme and option Information Technology	
Name of the bachelor's thesis Project management in the implementation of a new functionality in a CRM system using the Scrum methodology		
Abstract <p>The purpose of this study is to compare different methodologies of project management and on the basis of the research select the most appropriate methodology for the management of a project in development and implementation of a new functionality to an existing CRM system.</p> <p>In the study the following models were considered: traditional waterfall model, iterative model, spiral model, agile methodologies Extreme Programming, Scrum, Thrifty development, Feature driven development and Kanban.</p> <p>On the basis of the analysis the methodology Scrum was chosen for project management in development of new functionality in the a CRM system, as well as features of using Scrum methodology were analyzed in details.</p> <p>As a result of this study the IT project in development and implementation of new functionality to an existing CRM system was presented. Project management process was reviewed from establishing customer requirements to the deployment to the Production Server. The results of the project were obtained and analyzed.</p>		
Subject headings, (keywords) Project, IT project, project management, agile, flexible methodologies, Scrum		
Pages 54	Language English	URN
Remarks, notes on appendices		
Tutor Matti Koivisto	Bachelor's thesis assigned by Mikkeli University of Applied Sciences	

CONTENTS

1. INTRODUCTION.....	1
2. CHARACTERISTICS AND FEATURES OF IT PROJECT MANAGEMENT	3
2.1. The definition of the project.....	3
2.2. The definition of the project management and project team.....	4
2.3. The definition of IT project and its specifics	6
2.4. Models of life cycles of IT projects	7
3. FLEXIBLE METHODOLOGIES OF IT PROJECT MANAGEMENT	13
3.1. Description and principles of flexible management methodologies of IT projects	13
3.2. Justification for choosing the Scrum methodology for modeling the management of IT projects.....	17
3.3. Description of the scrum methodology	20
4. PROJECT MANAGEMENT IN THE IMPLEMENTATION OF A NEW FUNCTIONALITY IN AMOCRM USING SCRUM	29
4.1 Characteristics of the implemented project.....	30
4.2 Project management using Scrum	32
5. CONCLUSION	45
BIBLIOGRAPHY	47

1. INTRODUCTION

Project management nowadays is quite different from the project management in the 1950s, when this science was firstly implemented in construction and engineering companies for more effective organization of work. Today project management is used in almost every area of activity of each company, as it improves efficiency of companies.

With the development of information technologies many companies realized that using a variety of automated information systems can improve the profitability of the business. Therefore, higher level of integration of IT products and solutions to the companies' workflow, made effective management of IT projects very popular.

Many companies use traditional waterfall model when it comes to life cycle of a project. In this model every new stage of a project begins only after the end of the previous stage. This model is so often chosen for its simplicity rather than for its real efficiency. Recently so-called flexible management methodologies of IT projects, most of which are based on the advantages of iterative and spiral models of the project life cycle, have become increasingly popular. In addition, these flexible methodologies are promoted as "ready for change", which is in most cases better suited for IT-managed projects in rapidly changing environments.

The aim of this study is to analyze the feasibility of using a flexible methodology called Scrum to manage IT projects instead of the traditional waterfall model in the example IT project.

The case covered in this study is a project of implementing new functionality to an existing CRM system. As this system is very customer-oriented, the workflow of the project management faced many challenges such as increased working hours, incorrect operation of the new functionality and incorrect performance of the other parts of the system after implementation, etc.

The structure of the study is as follows:

Chapter 2 focuses on basic terminology related to projects including such aspects as definition of terms, analyzing success criteria of a project, determining characteristics of IT projects and analysis of life cycle models of IT projects.

Chapter 3 discusses flexible management methodologies, performs a comparative analysis of them and selects the methodology called Scrum for modeling and managing specific IT projects in the practical part.

Chapter 4 describes an IT project of implementing new functionality to an existing CRM system.

Finally, the conclusions are introduced in Chapter 5.

2. CHARACTERISTICS AND FEATURES OF IT PROJECT MANAGEMENT

2.1. The definition of the project

The term “project” comes from Latin “projecere” – to go further. The English word project keeps the base from Latin, narrowing, at the same time, the basic meaning – an activity aimed to achieve the goal. In modern society operational management of the business, which aims at managing well-established business processes, is losing its relevance. This is primarily due to the fact that external environment is constantly undergoing dramatic and significant changes. Respectively, projects become increasingly popular. Projects are such activities that are characterized by a certain degree of uniqueness in the solution of any problem. Also a company might have a very big range of different goals, and therefore, the number of projects is growing respectively. They differ by their nature and the future of the company depends on the form of their management. An activity should have several characteristics to be called a “project”:

- a well-defined goal with a quite certain result
- defined duration of the project with start and end time
- defined resources that will be needed for the project
- coordination of work of different specialists in one project

In all the books definition of the term “project” is based on these characteristics. As a result, a project is:

1. “complex, non recurring, one-time event for a limited time, budget, resources and clear guidelines for the implementation, designed to the needs of the customer” (Gray 2007)
2. “a complex system planning (financial, technological, organizational and other) documents containing complex-system model of actions aimed at achieving the original purpose” (Razu & Bronnikova 2006)

Projects are also characterized by the balance of the four indicators: budget, duration, field and quality (Figure 1):



FIGURE 1. Project Triangle (Wolfson 2012)

The change of one of these elements affects the other three. Limited budget represents the resources which are reserved for the project implementation. Limited duration means the time scale of the project. Limited area means the decisions and activities for finishing the project. The balance of these parameters represents successfulness of the project.

2.2. The definition of the project management and project team

Before the 20th century project management was performed by the project developers like architects, engineers, etc. But very soon other organizations understood that good project management was a very important part of their work. Especially clearly this was understood in the fields of construction, consultation and defense. With time the number of people interested in this field was growing. More and more universities included the subject Project management to the study program. Thus, graduates of the faculties of marketing, information technology, economics, finance, medicine and other sciences now have an idea of project management, which is certainly important nowadays, because project management encompasses not only the scope of business and construction, as before, but any side of society.

In professional literature the term “project management” implies a variety of methods, techniques and ideas on how to make the business more efficient. Here are some definitions for the term “project management”:

1. “methodology (also called - art) of organization, planning, management, coordination of manpower, financial and material resources throughout the project cycle, aimed at the effective achievement of its objectives through the use

of modern techniques, technology and management techniques to achieve specific project results on the composition and scope of work, cost, time, quality and satisfaction of the participants of the project” (Mazur & Shapiro 2004)

2. “management, which covers the areas of industrial activity, in which the creation of a product or service is implemented as a unique set of interrelated activities targeted at certain timing requirements, budget and the characteristics of the expected results” (Tovb & Tsipes 2003)
3. “professional activities to direct resources (human and material) through the application of methods, tools and management for the successful achievement of pre-set goals as a result of a set of interrelated activities under certain requirements, deadlines, budget and specifications of the expected results of the projects” (Gorbovtsov 2009)
4. “special kind of management, based on the pre-collegiate development of complex system model of action to achieve the original purpose and direction for the implementation of this model” (Razu & Bronnikova 2006)

In this study project management is defined as a completion of project tasks with the highest level of quality to meet the customer's requirements, to monitor the compliance with the terms of the project, project tasks' distribution between developers depending on their qualifications as well as to demonstrate completed tasks to the customer and get feedback.

Within the framework of project management there is constant interaction between the project participants: initiator, product owner, project manager, stakeholders and development team.

The concept of the project is set by the initiator. Afterwards, it passes to the product owner who is interested in determining the outcome of the project. The product owner determines the time scale of the project and expected results of the project. Financial issues are managed by the product owner or by the stakeholders. The project manager manages the activities needed to implement the goals of the project. He is directly responsible for achieving the goals and for possible errors to the customer. Formation of the project team, depending on the complexity of the organization and responsibilities assigned to the project team is under the leadership of the project manager. When recruiting people to the team project managers should take into ac-

count their professional qualifications and work experience. Thus, the more complex a project is, the more qualified the team should be.

2.3. The definition of IT project and its specifics

With the development of information technologies one part of organizations began to automate their work and the other part started to produce software for organizations. There are IT projects in all types of organizations.

The term “IT project” can be described the same way as the general term project but with some important additions related primarily to the fact that IT projects involve an activity aimed at the creation or use of information technology (Grekul & Korovkina 2011). As this study will discuss the management of an IT project related to the development of a new functionality to an existing system, we need to determine the difference between a project and an IT project.

Firstly, reticence between a product owner and development team. For most companies development of IT solutions is done by outsourcing. In this case there might be difficulties in determining business specifications and the outcomes of the project, because the IT sphere is very specific. This means that at the first steps of the project discussions between a customer and a project manager could be very blurred, and requirements could change in the future. When the customer reads technical specification of the project first time, he could have many questions, because the developer has his own vision of the IT project.

Another characteristic of IT projects is the uniform distribution of the blame on the participants of the project in case of its unsuccessful implementation. Thus, the responsibility for the result lies not only on the developers (with the possibility of incorrect implementation) and not only on the customer (for possible errors in the formation of the requirements), but on all parties together, because one way or another, they have to build together effective communication, minimizing possible subjectivity in the future.

The third characteristic of the IT projects is their high cost. If the construction of a building can have minimal deviations from the final requirements and expectations,

in IT projects they are very likely. Accordingly, there is a risk of significant changes, as well as very often the relationship between a customer and developers is regulated by the previously signed contract and any change entails a so-called “change request”, the result of which is a certain amount of money. A large number of changes imply a higher cost, and this is why IT projects are among the most expensive types of projects.

2.4. Models of life cycles of IT projects

2.4.1. The definition of life cycle of IT project

Each IT project goes his way from its creation through a series of milestones before its completion, when information technology is fully established and implemented. In other words, a project has phases which are called “life cycle”. Project Management Body of Knowledge (shortly PMBoK) gives the following definition of the project life cycle:

“The life cycle of the project - a set of usually sequential and sometimes overlapping phases of the project, the name and number are determined by the needs of the management and control of the organization or organizations involved in the project, the nature of the project and its area of application” (PMBoK Guide 2008).

Life cycle helps to coordinate the efforts of participants in the project, as it allows for some specifics. For example, the project manager should understand how input and output of some stages of the cycle influence the other stages. The life cycle of the project has a direct connection with its financing, as it is only possible to predict the cost and expenses with planned phases of its implementation.

The general structure of the life cycle consists of four phases: initiation, planning, implementation (or execution) and project closure (Figure 2):



FIGURE 2. Scheme of an IT project life cycle

The phases can shortly be described as follows:

- Initiation – determination of the project idea, business specifications and goals.
- Planning - drawn up a project plan, financial plan, quality plan and resource plan; the work to be done is determined in the framework of the project.
- Execution – execution of tasks; the results are evaluated by the project manager and subsequently compared with the project plan.
- Project closure – the final result of the project and documentation are submitted to the customer.

2.4.2. The waterfall model of an IT project life cycle

The waterfall model was introduced in 1970 and it is a traditional methodology of IT project management. The waterfall process is an ordered sequence of the various phases of the project, and each phase has a set of inputs and outputs (Holtsnider 2010).

The main phases of the waterfall model are shown in Figure 3. According to Ledbrook (2012) and Melonfire (2006) the phases are:

- Requirements development: Collection of business requirements from the customer, followed by their conversion to the functional requirements of the software product.
- Architecture and design: Determination of requirements related to software (databases, security settings, data model and interface circuit);
- Implementation: Development of the software product based on the specifications.
- Testing: Checking the absence of any defects in a software product, as well as the corresponding functional requirements of the customer
- Deployment: Product installation.
- Maintenance: Validation in the future work of the program.

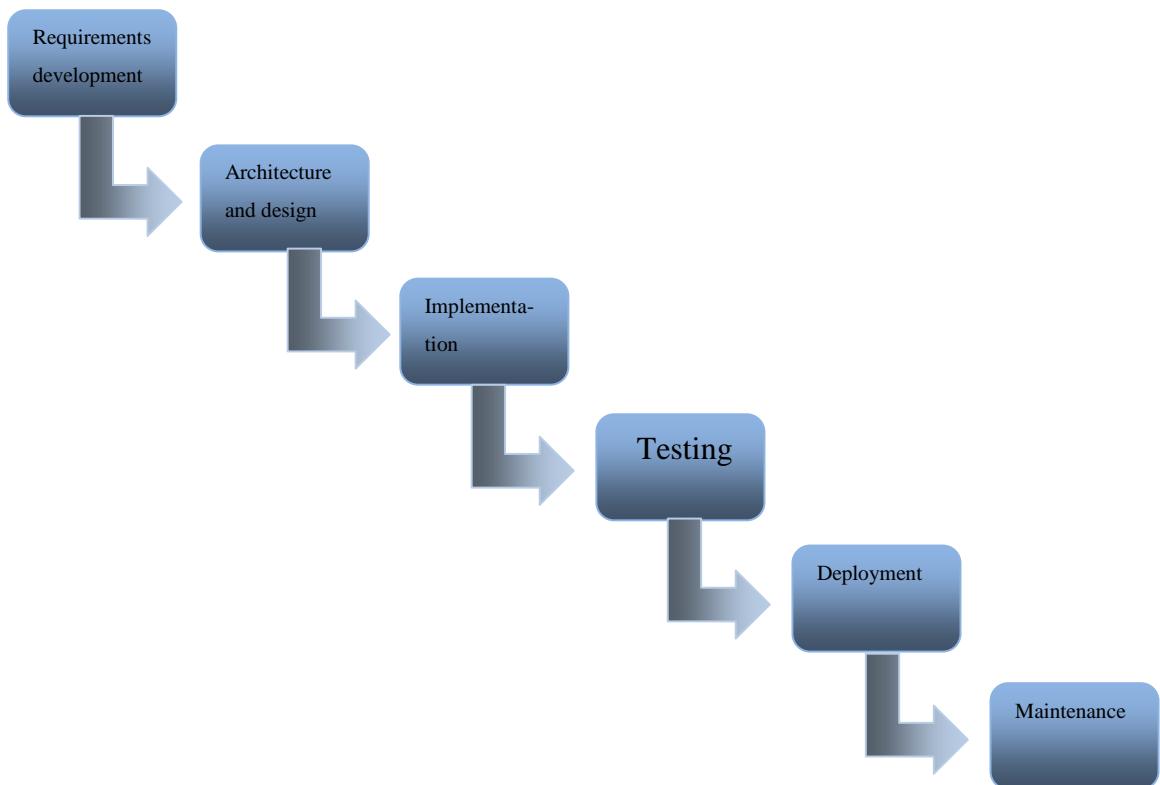


FIGURE 3. Waterfall model of an IT project life cycle (Holtsnider 2010)

The key point of the waterfall model is its serial nature, which means that each phase cannot start if you do not complete the previous one. This model requires a clear plan at the beginning of the project, and it is always accompanied by great technical writing jobs, but the result of running the program can be seen only at the end. The management of IT projects in the waterfall model is supervised by the project manager, who assigns tasks to developers and sets the time and budget. During the life cycle of the project the team reports on the work done only to the manager, direct communication with the customer is not available. If the specifications which were determined in the starting phase are not changed, it is very useful to follow the technical specification. The problem is that usually none of the drafts of technical specifications can cover all needed software functionality, users' requirements, and so on. In addition, if something has been missed in the project on its initial stages, it is very difficult to fix this in the final stages and requires additional costs.

As a conclusion, I should point out that the benefits of this model are only seen in small projects with clearly defined and unchangeable requirements. In other cases, corrected plans can contribute to non-compliance with the project.

2.4.3. Iteration model of an IT project life cycle

The iterative lifecycle model differs significantly from the waterfall. This approach implies a cyclic sequence of iterations, each of which consists of four phases: requirements gathering, planning, development and testing (Figure 4).

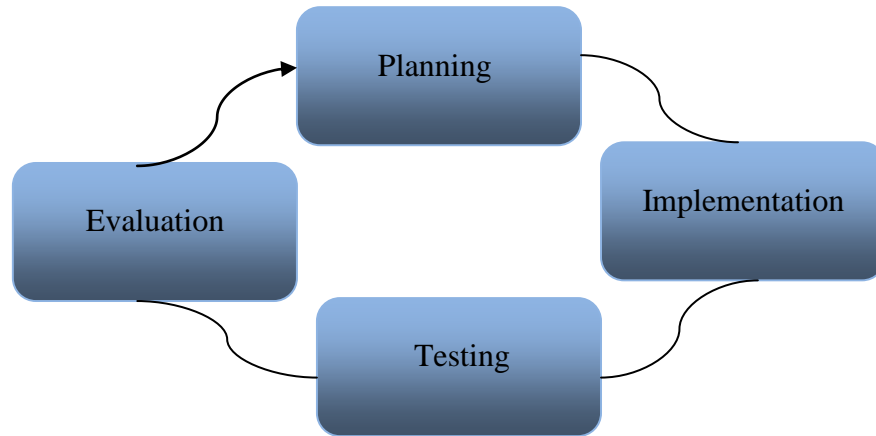


FIGURE 4. Detail of iteration of the iterative model of an IT project life cycle (Jalote 2004)

The main characteristic of the iterative model is that each iteration adds new features to the software. Forwarding Jalote (2004) “new iteration begins before the end of the current, thereby being a continuation of the current version of the software”. In this case, direction of the project activity can be modified at the beginning of each iteration, which is quite convenient in the case of a rapidly changing environment. The functionality of the whole system can be globally evaluated at the beginning of the project (at the first iteration), and required minor changes may be made in the next iterations. Furthermore, the iterations can be seen as means of getting feedback to measure the quality of the project and the productivity of the team members. An iterative approach to planning life cycle suggests that different types of work on the project are not tied to any specific stages of development. Instead, they are performed as required.

2.4.4. Spiral model of an IT project life cycle

The development of the idea of iterative model of software development is a spiral lifecycle model proposed by Barry Boehm in 1986. The main feature of this model is to minimize the risks at the beginning of every iteration, by improving communica-

tion within the project team. At the beginning of the iteration in a spiral model, the developers identify:

- goals which will be developed for this iteration of the piece of software;
- main and alternative ways to achieve these goals;
- analysis of possible restrictions.

After this, the project team begins to identify problems that may occur during the iteration as well as their causes (lack of information, lack of qualified staff). Next, the team begins to develop a strategy for the implementation of project activities during this iteration based on detected risks.

Each “spiral helix” is a complete part of the development product, respectively, with each new “turn of” project reaches a deeper level of detalization and specification (Figure 5):

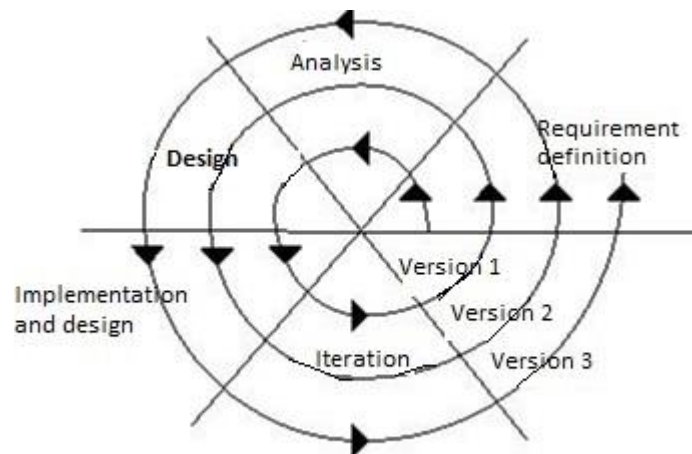


FIGURE 5. Spiral Model IT project life cycle (Izbachkov & Petrov 2008)

Special attention is devoted to planning of each spiral helix, as Boehm believed that the unsuccessful implementation of iterations is caused primarily by low level of communication between specialists in the project team. One of the obvious shortcomings of the spiral model is the difficulty of determining when to move to the next phase (Grekul 2010).

With the development of the theory of IT project management developers and their managers realized that a combined version of the iterative and spiral models of the life cycle may be the best way to organize the management of IT projects. Step by

step development and the availability of ready-made software running fragments taken from an iterative model as well as prioritizing the human factor, and risk analysis, taken from the spiral model were combined into a new **flexible methodology** called agile. According to Cobb (2011) flexibility means:

- the ability to respond quickly to changes in order to succeed in the unstable business environment;
- the possibility of rapid changes in the degree of priority use of resources in response to changes in requirements, technology and knowledge;
- the ability to quickly respond to any threats of the market as well as to any changes caused by the influence of customers;
- using an incremental approach in the delivery of the product for maximum customization;
- maximizing the profitability of the project by the desire to complete all the work in time.

3. FLEXIBLE METHODOLOGIES OF IT PROJECT MANAGEMENT

3.1. Description and principles of flexible management methodologies of IT projects

As previously mentioned, the theory of the flexible methodology came after realizing the advantages of iterative and spiral models of the life cycle compared to the traditional waterfall model. In February 2001, there was a meeting of the founders of the methodology about combining basic principles of iterative and spiral models. At this meeting it was decided to call these methodologies by the common word agile. The Agile methodology – an iterative process using a unique practice for obtaining new software features every 1 to 4 weeks (Rubio 2008).

At that time in 2001, two fundamental documents were created: “Manifesto for Agile Software Development” and “Principles of agile development”. The first one of these, agilemanifesto.org (2001), clearly defined high-level values which pay attention to the management of IT projects:

“Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan”

The first value indicates that no matter how important processes and tools are the success of the project depends, first of all, on the people involved in the project. In other words, in flexible methodologies human factor is very important, namely the possibility of organizing effective communication, ease of overcoming conflicts and so on.

The second value does not exclude the documents, on the contrary, it facilitates interoperability and cooperation as well as establishes the transfer of knowledge, but the effective interaction between people has much better impact on the project, rather than well-described documents without interaction within the team.

The third value implies that the ongoing communication between the customer and the development team is a more appropriate strategy than the “war” for the correct interpretation of the previously signed contract. “The current practice of projects usually is accompanied by protracted negotiations between the parties. Of course, the negotiations – is one of the most important elements of business practice. But often they become a zero sum game: one side is trying to ensure their own interests by the expense of the other side” (Chernykh 2008). Flexible methodologies oppose such a “struggle” dialogue between developers and customers. This approach allows the two sides to win by common efforts.

The last, the fourth, value of the project calls for the team to be constantly ready for changes. No IT projects can be pre-designed entirely with all the nuances. The flexible methodology does not deny the need for early planning. It only allows for adjustments to the time available for the implementation of IT project.

“Principles of agile development” details the core values of the “Manifesto”. These twelve values are concretized synthesis of the fundamental principles of iterative and spiral models. They allow you to organize a disciplined flexible process management of IT projects, to carry out all the work iteratively with intermediate checks, creating a harmonious and self-organizing team of developers and being in constant communication with the customer (“frequent deliveries of the product”, “frequent deliveries of new versions of software”, “methods of interaction and exchange of information”, “technical excellence and quality architecture”). It is important to note that a well-established interaction between participants of the project is able to guarantee fast performance.

Gary Chin (2014) lists in his book “Agile Project Management: How to Succeed in the Face of Changing Project Requirements” in more detail the factors of the project. Availability of those should serve as a signal for the use of the agile methodology for the management of IT projects. These factors are divided by the author into two groups: “internal uncertainties” and “external uncertainties” (Figure 6):

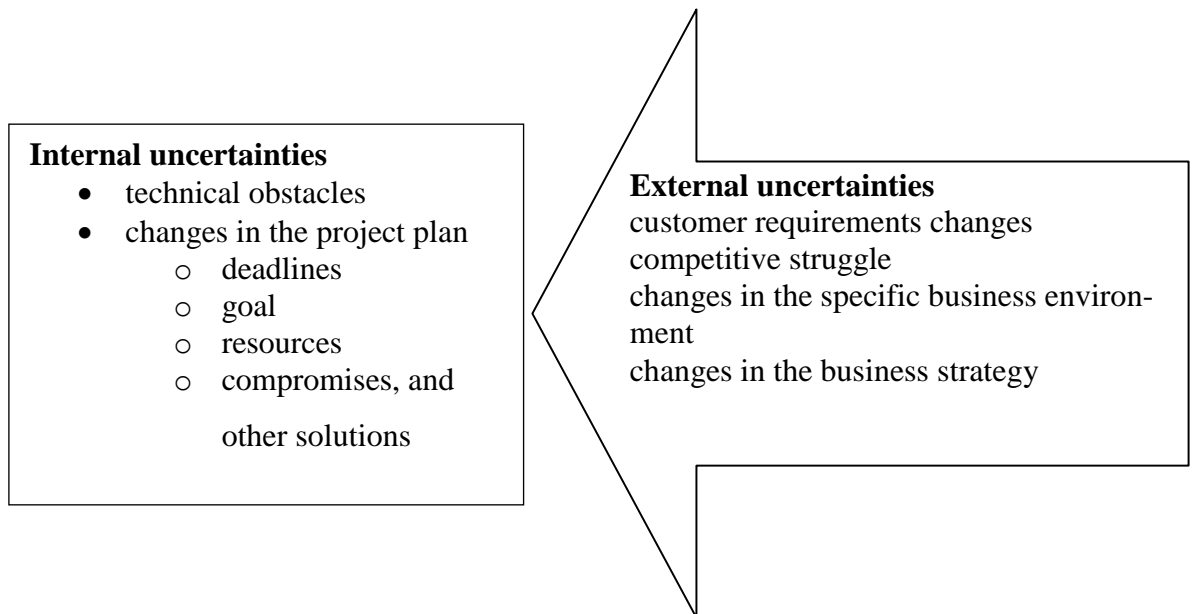


FIGURE 6. Internal and external project uncertainties (Chin 2004)

If the project is exposed to some internal or external uncertainty, it is prudent to plan its life cycle by using a flexible methodology.

When using flexible methodologies for a project tasks are broken into small pieces (iterations) with careful short-term planning and almost negligible long-term planning. If in the classical iterative model the iteration length can be arbitrarily long (but within reasonable limits), the flexible methodology limits the time available to perform a particular iteration up to five weeks.

Flexible methodologies are based on empirical control, that is, such a control where decisions are made on the basis of the intermediate results of the project. In addition, these results are transparent, which means that all the people involved in the project are aware of the project status, number of changes and potential problems (Layton 2012). It should be noted that in the case of IT projects empirical control allows to quickly make adjustments to the software, and it is a great advantage compared to the waterfall model of the life cycle.

During the last few years in the United States a lot of studies were conducted to understand the benefits of flexible methodologies compared to the traditional waterfall model. For example, according to the seventh annual public research of flexible development by Internet resource Version One, 84% of respondents (the developers)

have observed that flexible approaches seem to them more effective than traditional approaches (7th Annual State of Agile Development Survey 2013).

In addition, 70% of respondents of the same survey acknowledged that IT projects implemented with the help of flexible methodologies are realized much **faster** than IT projects implemented by a waterfall life cycle model (Figure 7):

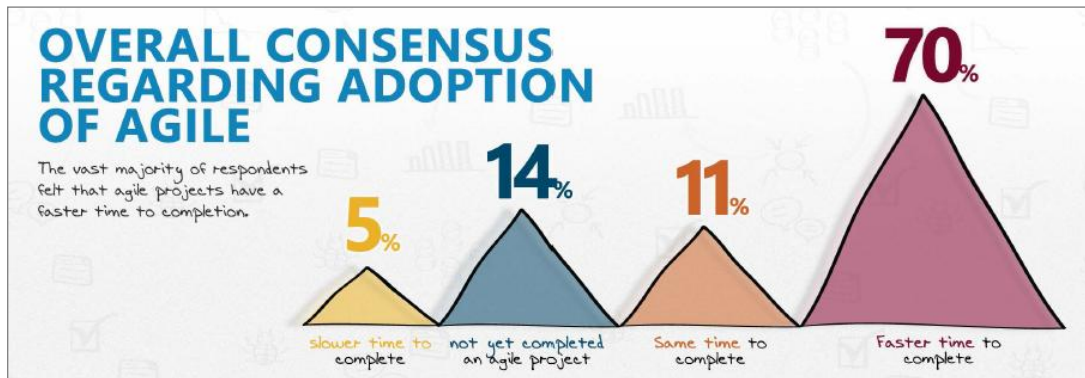


FIGURE 7. Projects are implemented faster if use the flexible methodology (7th Annual State of Agile Development Survey 2013)

As the final stage of the analysis of the nature of flexible methodologies and their benefits, the table below once again shows the main differences between flexible methodologies and waterfall models that were not reflected clearly in the “Manifesto”:

TABLE 1. Key differences between flexible methodologies and the waterfall model

	Flexible methodologies	Waterfall model
Customer requirements	Iterative gathering	Detailed requirements clearly defined before the start of the implementation phase
Cost of improvements	Low	High
Direction of the development	Can be changed at any time	Fixed
Testing	After each iteration	During the corresponding phase, following the implementation / development
Customer Involvement	High	Low

3.2. Justification for choosing the Scrum methodology for modeling the management of IT projects

Currently, there are many different agile methodologies, but the most popular are Extreme programming (XP), Scrum, Thrifty development (lean), Feature driven development (FDD) and Kanban. Table 2 shows a comparative analysis of these methodologies.

TABLE 2. Comparative analysis of agile methodologies (Abrahamsson 2002)

Name	Key points	Unique features	Disadvantages
Extreme programming (XP)	Small teams, daily meetings, informal communication within the team, minimum documentation	Permanent correction of the project for improving its efficiency and for adaptation to the changes	More suited to the individual practices than for global management, with the risk of forming undisciplined teams
Scrum	Independent, small, self-organizing team of developers, iteration length is 2 to 4 weeks, the team deciding how much time it takes to complete the task, informal communication within the team, daily meetings, with only basic documentation	A high level of communication and interaction within the team, clearly-written formal organization of this methodology, the presence of “supervisor” of the team, a complete orientation to the customer's requirements	A risk of increasing the time of the project due to the Scrum activities
Thrifty development (lean)	Using visualization tools, development through testing, short iterations	Main functions of the software valuable to the customer, there is a permanent motivation of the team	Decisions are made by a long time, lack of discipline, suitable only for small projects
“ Feature driven development” (FDD)	A five-step process, object-oriented development, very short iterations (may reach several hours in duration)	The simplicity of the method, object modeling	Focuses only on the design and implementation, very little attention is paid to its development
Kanban	A self-organizing team, without the meaning of the term iteration - all is focused on the task	No restrictions on the execution time, but there is a limit on the number of “working at the moment” tasks, effective, when nobody knows what the customer may require from the software in the future	Lack of discipline, protracted nature

The FDD focuses on the five-step approach which is based on the identification, development and implementation. The FDD also assumes that part of the work of the project is already done. As a result, many phases of the project are not fully implemented.

Kanban can be effective in a particular organization, even within a certain department. In the case of complex contractual relationship between customers and developers it is very difficult to implement.

Lean development and extreme programming also lose their effectiveness when more than one organization is involved in the project: the absence of explicit observers could create confusion of responsibilities.

The Scrum methodology is aimed at interaction with the customer, and, despite the fact that the team decides what tasks will be performed during one iteration, there is an observer (Scrum Master) in the methodology, who monitors compliance with the scrum process. In addition, scrum is well documented: there is a special “Scrum Guide” (Schwaber & Sutherland 2011), translated into many languages.

A certain degree of formalization in terms of flexible development methodology is the best strategy for managing IT projects when the project involves people from different organizations.

Scrum is the most popular flexible methodology, which is widely used by international companies such as Yahoo!, PayPal, Nike, Google, SAP and GE for project management (Deemer 2007). According to the sixth and seventh annual public research of flexible development by Internet resource Version One, scrum is more likely to be used as a method of “flexible” management of IT projects in several times more cases than the rest of methodologies (see Figure 8 and Figure 9):

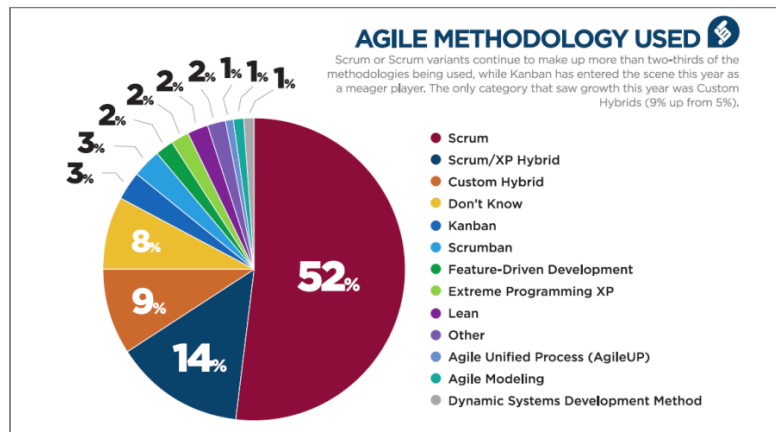


FIGURE 8. Rating of the scrum methodologies in 2011 (6th Annual State of Agile Development Survey 2011)

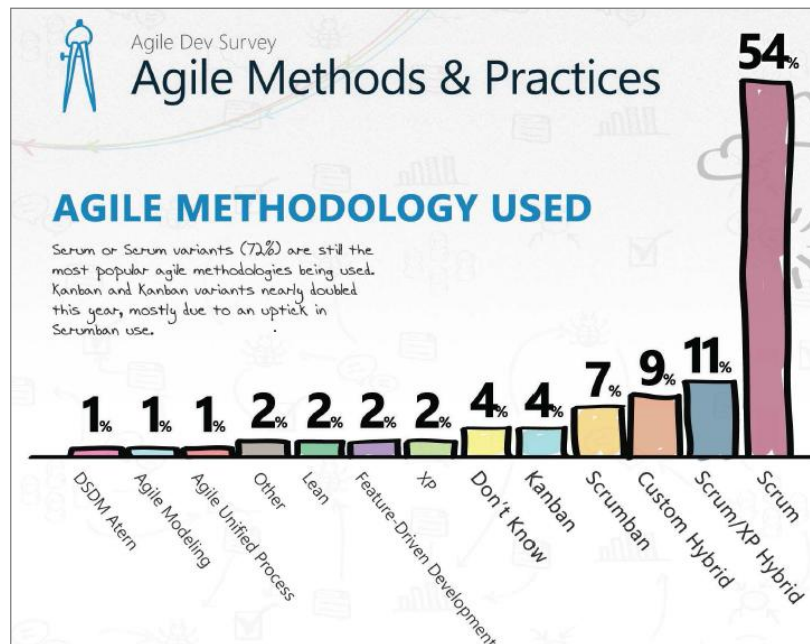


FIGURE 9. Rating of the scrum methodologies in 2012 (7th Annual State of Agile Development Survey 2013)

3.3. Description of the scrum methodology

Scrum methodology was firstly introduced in the 1990s by Ken Schwaber and Jeff Sutherland in a clearly documented and formalized “Manual of Scrum”.

According to Cervone (2012) Scrum is a flexible and easy process of managing and controlling software and product development in a rapidly changing environment. This methodology establishes certain rules for the administration of IT projects (the development or implementation of information technology) which are based on the possibility of a permanent adjustment requirements and making tactical changes.

Before the start of the project participants discuss global goals and basic strategy to achieve these goals. A person from the customer side of the project represents the interests of his company by describing business functions of the implemented or customized software. On the performer side the development team and other resources are determined, after which the parties discuss the approximate scope of the project as well as the start date of the first iteration. Iteration in scrum is called a **sprint** and it lasts two to four weeks. All sprints have a relatively fixed duration, which means that all the work of the sprint must end at the end date of the sprint (with minimum error), regardless of whether the project team managed to do the job or not. Minimum error means that, in practice, some delays are still possible, but, they should not however exceed two or three days for a sprint, the duration of which is equal to two weeks.

The model of the scrum methodology is made up of three main components: roles, artifacts and activities. In this methodology there are three roles: Product Owner, Scrum Master and Scrum Team.

- **Product Owner** is often a client that maintains an up-to-date list of the project requirements. The better and more clearly the owner describes the product requirements, the fewer questions there will be from developers and the less software features will be changed over time. Product owner determines the priority of the requirements. So, the customer decides which software features are the most important and urgent for the company at any particular time;
- **Scrum Master** is responsible for the understanding of the nature of all the scrum process by other members. Scrum master is something similar to a traditional project manager, but there is one difference: Scrum Master does not give explicit orders and does not set deadlines for the developers. On the contrary, he helps them in the event of any difficulties and ensures that their

work is coordinated. On Scrum Master depends the atmosphere in the team of developers, and as a consequence, their initiative, satisfaction and the overall result. Scrum Master solves any problems that may somehow affect the team, whether it's broken equipment or the external pressure from the customer.

Scrum Master also ensures that all scrum events (which will be described below) took place regularly and with maximum effectiveness;

- **Scrum Team** consists of professionals who are working on the development of a new, potentially “ready” to release version of the product at the end of each sprint (Schwaber & Sutherland 2011). Usually the number of developers and programmers does not exceed eight, and they are all interested in the successful implementation of the IT project. Before the start of each iteration the Scrum Team sets an achievable and meaningful to the customer purpose and during the iteration directs all its efforts to achieve this goal in time with the declared quality. Achieving the goal is characterized by a scheduled code that is tested later, and any defects of the iteration could be corrected. Scrum Team members themselves set deadlines (discussing them with Scrum Master and Product Owner) and evaluate the opportunities and allocate time.

Among the artifacts of the scrum methodology are distinguished Product backlog, Sprint backlog and Combustion chart (Figure 10):

Artefacts		
<u>Product backlog:</u> a list of all business requirements of the project, ranked in order of importance	<u>Sprint backlog:</u> business requirements from the product backlog selected by scrum team for a particular sprint	<u>Combustion chart:</u> It shows the amount of done and remaining work

FIGURE 10. Artifacts of the scrum methodology (Wolfson 2012)

Product backlog is compiled by Product Owner (customer) after analyzing the needs of the business as a list of software requirements. In other words, the customer describes the main desired functionality of a developed or implemented program.

After that, each request is ranked by the level of importance for the customer, which is the priority; the highest priority task must be higher in the Product backlog list than that of the least priority. Throughout the project Product Owner has the right to add new requirements or modify the old ones in a Product backlog. Such capability allows this approach to be flexible to changes. In real life, the customer is constantly changing the list of software requirements. Respectively, Product Owner can keep their immediate consideration and serve as a “connecting” bridge between the customer company interests of which the Product Owner represents and Scrum Team (developers). Requirements which are selected for a specific sprint from the Product backlog are recorded in **Sprint backlog**.

Schematically it can be represented as follows (Figure 11):

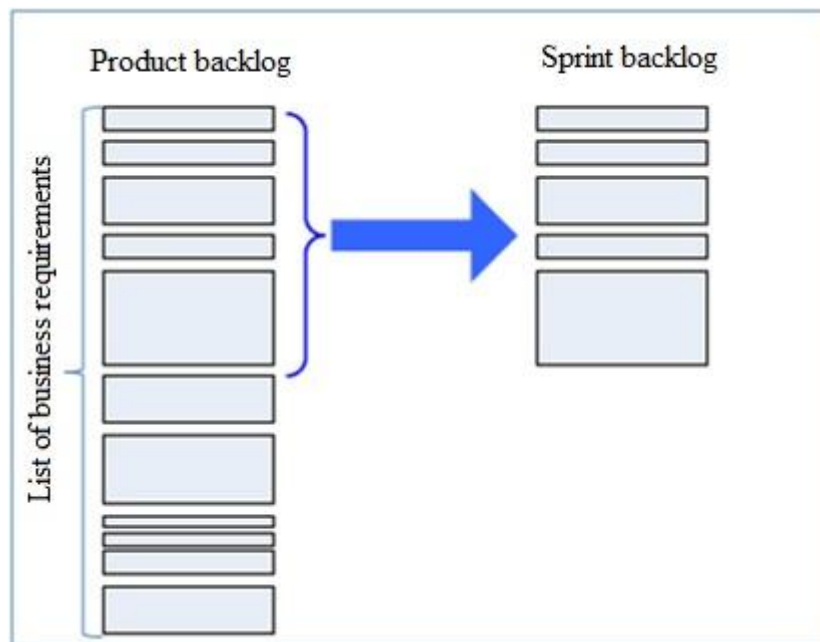


FIGURE 11. Formation of Sprint backlog (Wolfson 2012)

Unlike Product backlog, which can be changed or refilled by Product Owner at any time of the project, Sprint backlog for a particular iteration is fixed and approved by the Team. Tasks from Sprint backlog can be excluded only in cases of force majeure, when their implementation is no longer aimed at the success of the IT project. As mentioned earlier, scrum methodology does not exclude the documents at all. Simply, there is much less documentation than in the traditional waterfall development approach. In practice, everything is solved by the Product Owner and Scrum Team: if both parties come to an agreement about the process of implementing some tasks from the product backlog, the developers create an appropriately small document.

This document will not be similar to the technical specifications. It will not be referenced when discussing the contract conditions. It is necessary for a more concrete understanding of the operation of any part of the program.

In order to keep a record of the amount of work and volume of the work, **Combustion charts** are used. Before each iteration, the team plans the amount of effort required to perform the tasks from the Sprint backlog. After the completion of each task Scrum Master counts the amount of the remaining work for the sprint. These values are marked on the graph where the horizontal axis shows the duration of the sprint, and the vertical axis the amount of remaining work. Combustion chart is useful as a support tool that allows evaluating whether the team is working in the right direction in a specific sprint.

In addition to the sprint there are four events in the scrum methodology: sprint planning, daily meetings, reviews of the sprint outcomes and retrospective meetings.

Before the start of the sprint its **planning** takes place. At first, Product Owner shows the product backlog which contains software requirements sorted by priority at the moment. At the same time Product Owner discusses possible nuances of tasks with the Scrum Team, thereby minimizing the possibility of subjectivity in the development. After that, the team selects the most important tasks of the Product backlog, which are placed in Sprint backlog. The key point here is to determine the time required to perform these tasks by the developers. As in this case, they decide all by themselves, without external forcing by the project manager, the probability of unjustified overestimation of time is minimal.

After the next sprint tasks are assigned Scrum Team starts to discuss each of them in more detail. For successful completion of the task, it can be divided into mini-tasks in an understandable way for developers. In addition, Scrum team discusses the code development process, the interdependence between the various components and so on. In other words, developers decide how they will implement customer's requirement.

Scrum methodology aims at organizing coordinated work within the team. Sprint planning lasts about four hours, and the main result of this event is the agreement

between Product Owner and Scrum Team about the number of tasks for the planned sprint, as well as assurance that the understanding has come to all the participants of the meeting.

Despite the flexibility of the scrum methodology the team must be assured of the immutability and stability of tasks within a specific sprint. If Product Owner wants to change the requirements, he should wait for the next sprint planning. *This approach is both flexible and at the same time protects the project from possible confusion and disagreement.*

After planning the sprint begins. Every working day Scrum Master organizes a short **meeting**, not exceeding fifteen minutes. The purpose of the **daily meeting** is to allow team members to share their own progress and the difficulties being encountered.

Each team member answers three questions:

1. What has been done since the last meeting?
2. What do you plan to do before the next meeting?
3. What are the difficulties hindering the progress of the task?

If any problems occur, Scrum Master is doing everything possible to eliminate them after the end of the daily meeting. Thus, the 15-minute meetings helps to keep Scrum Team participants up-to-date with what is done by their colleagues and on what stage are the tasks of the Product backlog. Product Owner can also attend this meeting, but these should not turn into discussion between him and developers: a meeting is organized only for brief statements (Wolfson 2012).

If Scrum Team could not achieve the goal of the sprint, developers recognize this fact on the last daily meeting and try to understand the causes of the incident. It is possible that at the stage of the sprint planning capabilities of the team and the time required to complete tasks were evaluated incorrectly. Very often it happens at the very first sprints, which makes sense: the team could not yet fully work well with the Product Owner and with the new IT project in general. However, in future, such failure must be corrected with the help of the necessary influence of Scrum Master.

After the end of the sprint an **overview of its results** is provided, which demonstrates the basic functionality, which was introduced to the project during this sprint. Any concerned person can attend this meeting (for example, someone from the management of the customer wants to see the new features). It is important to note that a review of the results of the sprint is not intended for an organization and presentation planning. In fact, it is a direct demonstration of what has been done during the last sprint. Product Owner determines the completeness and quality of the job, thus establishing feedback with developers (Figure 12):

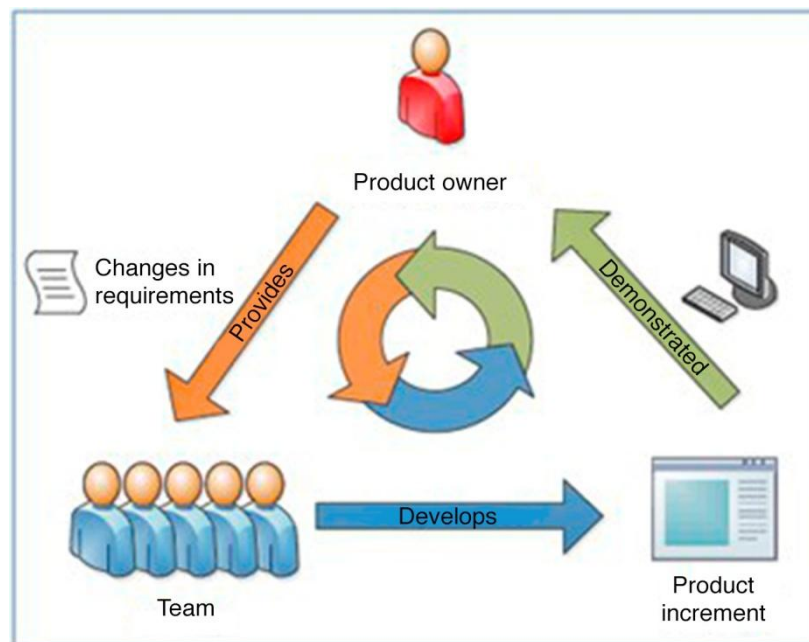


FIGURE 12. Feedback organization within the sprint results review (Wolfson 2012)

Subsequently, Product Owner corrects the Product backlog and makes some conclusions about further timing of the project. The results of the meeting are corrected Product backlog and plans for further sprint.

Retrospective meeting is also held at the end of the sprint, but at this meeting, there are only Scrum Team participants, Scrum Master and Product Owner (for additional feedback).

Length of the meetings is varying from fifteen minutes to two hours, depending on how long the sprint was, how many problems were found, and so on. During the meeting the developers are discussing among themselves the problems and the posi-

tive aspects of the last sprint. If the team concludes that in the project there is a problem that affects all parties, then decision is made about changes that can improve the work flow in the next sprint. An example of the results of a retrospective meeting is shown in Figure 13:



FIGURE 13. Visualization of positive and negative moments during retrospective meetings (Kniberg 2006)

In general, scrum process may be schematically represented as follows (Figure 14):

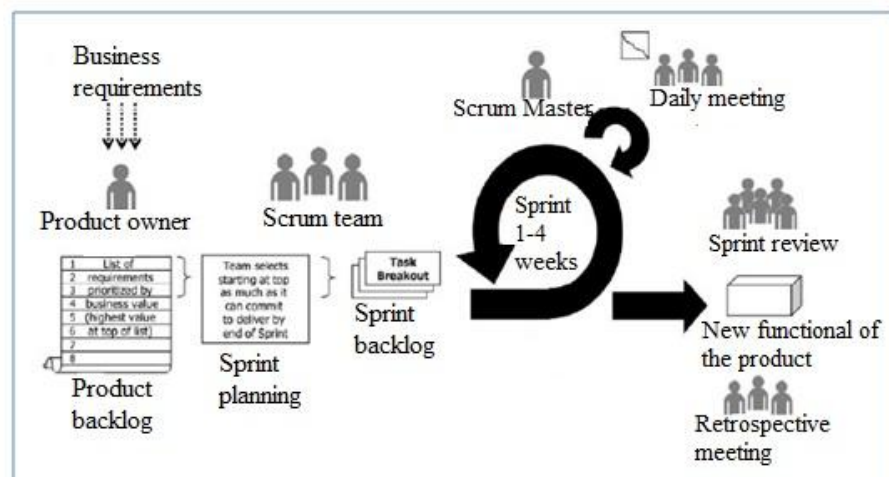


FIGURE 14. A full scrum process (Schwaber 2011)

To sum up, it should be noted that because of the **constant analysis of the work performed and the possibility to correct the direction of the project between iterations (sprints)** scrum methodology allows achieve results more productively and, consequently, develop the software with higher **quality**. The methodology assumes that the team selects those tasks which are of the highest priority for the business and meet technical possibilities. In addition, the absence of external pressure

and independence in the choice of the amount of work contribute to the fact that the developers feel fully involved in the process of development, they are experts from whom the quality of the product depends, not just performers who only carry out orders of their manager. This approach stimulates enthusiasm and activity of the developers. In addition, self-selected amount of work assumes that the employee knows its capabilities, and thus the probability of deviations from the schedule is much lower than in the same traditional IT project management model. The project cost depends on the duration, respectively, if the customer has to pay for additional work time of the developer (if the developer runs out of his own set time frame), then the cost will be much lower than the cost of a query to a change that would occur in a similar situation in the waterfall model.

The scrum methodology enables permanent change of priorities of business requirements, which increases the profitability of the project, even at the earliest stages. In addition, thanks to the regular meetings of Scrum Team and Product Owner, in which participants receive feedback from each other, the team manages to develop precisely such software that **suits customer's expectations best**. As in the project satisfaction of the customer plays an important role, the IT project can be considered as qualitative, not only when it runs smoothly after the implementation, but also when it meets the requirements of the customer. A significant advantage is the ability to monitor the intermediate product, developed and implemented within a certain sprint. This makes it possible to **detect and correct errors in embedded fragments at the early stages**. This is the level of quality of the project. Product Owner together with his colleagues can see the work of the program, even with the minimum capacity.

4. PROJECT MANAGEMENT IN THE IMPLEMENTATION OF A NEW FUNCTIONALITY IN AMOCRM USING SCRUM

To begin with, what is the Customer Relationship Management (CRM) system? This is online software that helps to increase sales, helps to bring potential customers to purchase, organize the work of the sales department, increase the amount of the bills and to repeat purchases or orders.

amoCRM – is a company which makes a CRM system with the same name for small and medium-sized businesses. This is a SaaS product, when the client does not need to install anything on his devices. He works in the browser online. The distinguishing features of the product are easy-to-use and focus on sales management.

I am working as a Project Manager in this company. If we use the terms of scrum methodology, I am a Project Manager and Scrum Master at the same time. My duties include discussion of the project and getting requirements from the Product Owner (this role is played by the company's management), writing technical specification for the project from the business logic point of view for developers (for large and innovative projects such document is written by the customer), discussion with the technical director about resources which are needed for performing the task, formation of the development team for the project. As well as holding of meetings, tasks completion control, quality control of completed tasks, test tasks of the most likely used-cases from the user's perspective, presenting the work to the customer, receive feedback and maintaining up-to-date product backlog. As a Scrum Master I provide a comfortable working environment for the team, decide on arising administrative matters (like controlling of buying a new monitor for the developer) and interaction with other company departments for resolving team issues.

One of the operating parameters in a CRM system is a **lead**. Lead - is the contact information of potential client. This may be name, company, phone, email, the application from a third-party resource or customer's action on the company project. Also in the sales theory, this term used to allocate the contact to one of the stages of the sales pipeline. Combination of these definitions refers to the lead, used in the CRM systems.

4.1 Characteristics of the implemented project

As part of the next release of amoCRM a new system function called Digital Pipeline was developed. With this functionality, users can set up automatic movement of the lead on the sales pipeline and use digital options for communication with customers like sending emails with special offers, setting to-dos for managers for every lead, as well as adding customers to advertising campaigns and mailing lists in social networks (Figure 15).

From a visual point of view this functionality is a separate interface where automatic action settings are made. Also in this interface users can configure the sales stages and create or change existing sales pipelines.

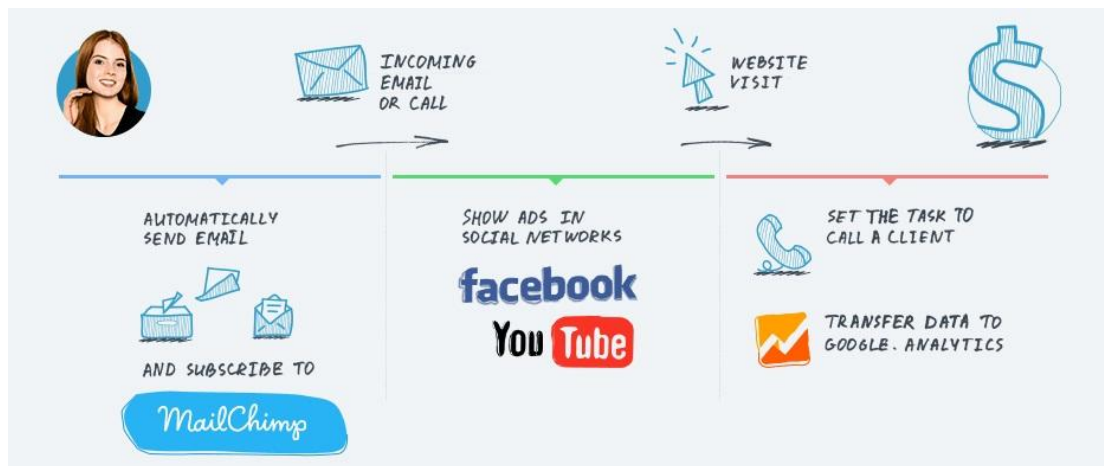


FIGURE 15. The concept of Digital Pipeline

From a technical point of view functionality resides on a separate virtual machine. Communication of this system with the rest of the services, which are implemented in amoCRM, is established through API requests.

Functionality allows to:

- automate the movement of leads by statuses of the pipeline. This means that functionality allows you to track client activity (his calls, incoming emails, website visits or correspondence in online chatrooms). Depending on the oc-

curred event and the user's settings in the Digital Pipeline functionality will change the status of the leads;

- automate the work with clients. In the Digital Pipeline interface the user can set up pipeline statuses to send emails, add to-dos for managers, add customers to the advertising campaigns in Facebook and VK social networks and add customers to promotional mailing list on MailChimp. With these options warming of a client is reached with maximum automation and with the minimum participation of the sales manager;
- minimize the human factor when working with a client. When the automatic transition of the leads from status to status is configured, the head of the department always has a clear view of what stage the lead is on, because the manager cannot forget to change the status of the lead upon the occurrence of any events. Also, automatic sending of emails and adding to the advertising campaigns minimizes the annoying desire of a manager to “warm” the client, which, in most cases, only scares potential successful customers. At the same time adding to-dos for a manager automatically in a particular lead status allows for communicating with the client at an exact point of time and bringing the transaction to a successful conclusion;
- reduce company's expenses on the sales department. As a lot is now done automatically, on certain stages of the sales the work of manager can be completely excluded, which reduces the number of man-hours and payroll costs;
- increase overall conversion of leads from the first referral to the successful conclusion. Since now the mechanism of movement of leads by the sales steps is almost completely implemented, managers can see actual customers who are interested in the product or service. Managers do not lose these clients in a common base, but on the contrary, will help them to determine their wishes and to choose the most advantageous offer;
- track conversion with Google Analytics. Digital Pipeline allows users to configure automatic sending of information to the goals in Google Analytics and to monitor the conversion there.

After the analysis of existing CRM systems and the identification of the needs of users, it was concluded that nowadays none of the services, a CRM system or similar to it, is not implemented with this functionality. Therefore, the implementation of

this project enabled amoCRM company to become unique on the CRM system market.

In addition to the already implemented functions, we allow our partners to implement their own projects and automatic functions via webhooks and API. In this case, when the lead has passed a certain status, the webhook notification will be sent to the user. User can change the status of the transaction through API requests by entering its parameters. Accordingly, when writing some program code, our customers and partners can use the Digital Pipeline not only within the functions implemented by us, but they can also build their own unique usage patterns (Figure 16).

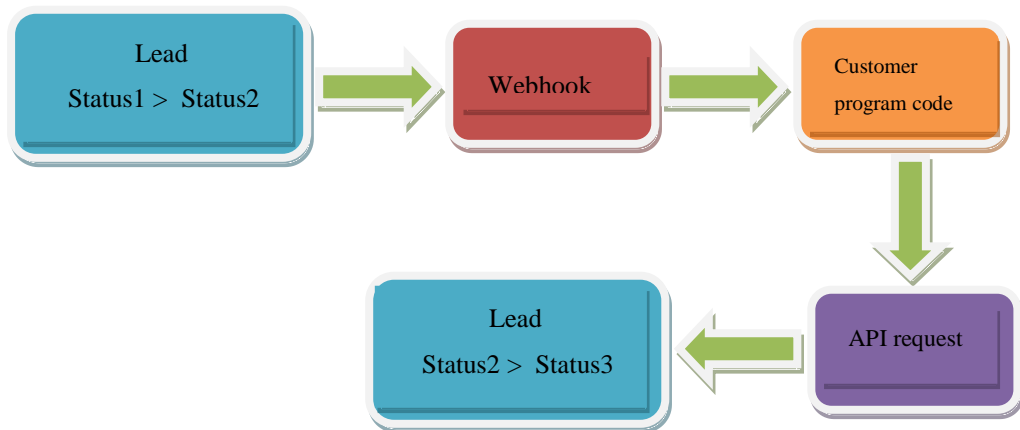


FIGURE 16. The concept of customer functions with Digital Pipeline

To sum up, we can say that Digital Pipeline is a platform-designer which implements the basic steps and components and tools for the implementation of more complex and narrowly focused tasks.

4.2 Project management using Scrum

The CEO and co-founder of the company played the role of Product Owner. Employees of the company represented Scrum Team, which consisted of three developers - two backend developers and a frontend developer.

The project had two strict deadlines that could not be changed - the introduction of the first version of the product in company's sales department and release of the product for users, which took place on public presentation. A few thousand users of

the system were invited to the presentation, there was an online broadcast of the event. At the presentation not only the result of the Digital Pipeline project, but also the results of other teams were presented.

Based on the fact that it is very difficult to predict in advance the entire scope of the necessary work that is required for the successful implementation and delivery of the IT project, flexible management methodology has allowed the most efficient management of the project. Flexible methodology allows the collection of requirements during a sprint, not to allocate additional time after the implementation of the project. Also a flexible methodology allows to change the number, priority and task sequence that allow quick response to changing requirements and to making quick adjustments to the work of the team. Testing and debugging of the functionality must be carried out immediately after development, because the quality of the developed functionality was an important point for the customer. This contributes to the minimal deviation from the planned timing and size of the project. In addition, using scrum makes it possible to organize work in parallel, which cannot be done in the classic waterfall approach.

When using scrum the team determines the timing of each task by itself. In future, on the basis of these assessments and priorities established by Product Owner, sprints are formed.

Before the start of work on the project a meeting was held to discuss the project between Product Owner, Project Manager, executive director, head of the production department and Scrum Team. At this meeting the essence of the project (Figure 17) and its purpose were explained, design layouts for implemented functionality were demonstrated and Product Owner explained necessary basic functions that will be implemented.

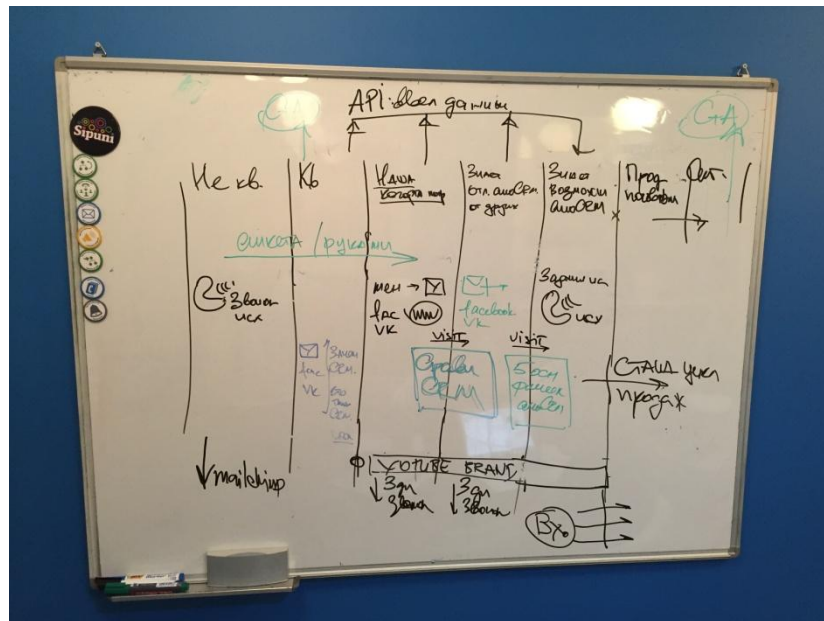


FIGURE 17. The schematic picture of the result of the project from first meeting

After the meeting Scrum Team has developed the technical documentation of the project, which included a description of the technical specifications of the functional and integration with the core system and other services (Unsorted, mail, web forms server and API). The next step was the approval of the documentation by the technical director of the company and software architect. These actions were implemented in the “zero sprint”.

Scrum project was held as follows:

- Product Owner has prepared Product backlog for each sprint, with tasks sorted by priority;
- Each sprint starts with its planning, where the team determines the amount of time needed for each member to perform tasks from the Product backlog if they have not yet been evaluated or confirmed, the accuracy of evaluation of the tasks which had been already discussed;
- Selected tasks from the Product backlog are transferred to Sprint backlog and, if necessary, are detailed by the developers and broken up into smaller tasks;
- During the sprint planning team members discuss the general idea of the tasks, their correlation with other specialists, services and departments, etc;

- During the sprint the team performs the tasks, reporting to each other the work done on the Daily Meetings, as well as talking about emerging problems and questions;
- Control of the duration of the meetings, as well as maintaining the atmosphere in the team, was provided by Project Manager, who is also a Scrum Master;
- Each completed task the team (if the task was carried out by all the team members) or the developer (if the task was carried out by him) passes to the Project Manager immediately after the execution. Project Manager tests the result from the user point of view in the most likely used cases and makes a decision on the degree of fulfillment of the task based on the requirements set out in the sprint planning;
- At the end of the sprint the team and Project Manager demonstrates the functionality to Product Owner and any concerned person (for example, it could be the executive director of the company or the head of the production department);
- If the Scrum Team did not complete the task at a sprint, the unfinished part of the functionality of this task once again is added to the Product backlog;
- During the sprint Product Owner can have new requirements for software which he wrote in the Product backlog, and then waited for the planning meeting of a new sprint in order to make changes to the work.

The project was divided into two iterations. Each iteration consisted of multiple sprints, and the sprint length was one working week.

The first iteration – the development of basic functionality of the product and the introduction of the product for use in the company’s sales department. The first iteration lasted from 22nd February till 1st April 2016.

Tasks for the first iteration were:

1. The implementation of the core functionality by the backend
 - realization of all necessary Germans and Workers for service
 - realization of API for interaction with other parts of the system
 - realization of base for the subsequent implementation of events and actions

2. Digital Pipeline interface by design layout of the page (Figure 18)

- css for the basic elements
- frontend implementation of page elements (buttons, links, add items, etc.)

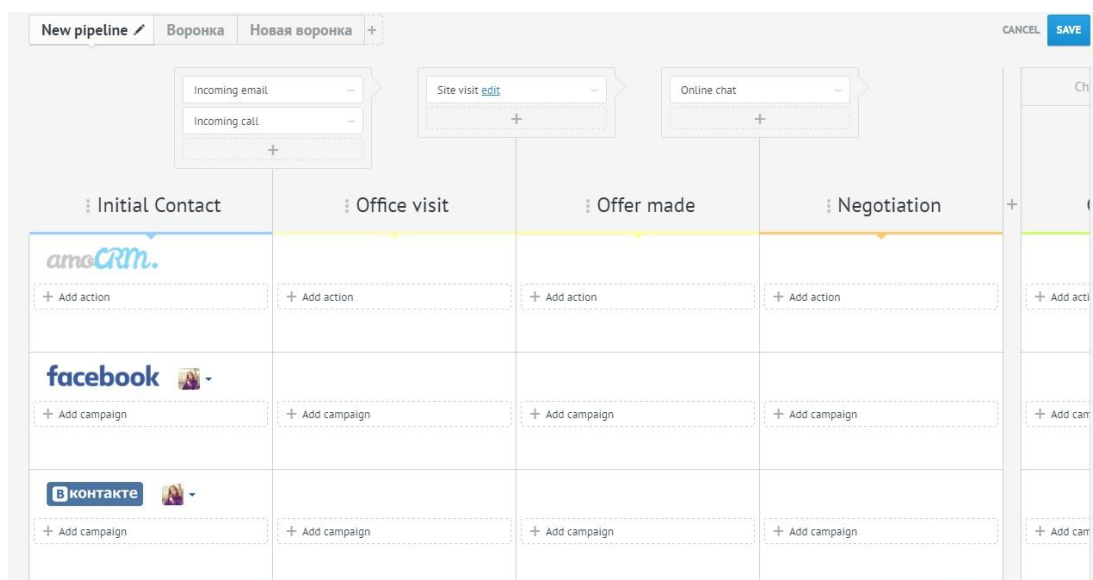


FIGURE 19. Interface of Digital Pipeline

3. Integration with advertising campaigns in Facebook

- realization of service for working with Facebook - creation of application in Facebook, which is working as an API client, and implementation of API for amoCRM interacting with advertising accounts in Facebook
- realization of oauth authentication for connecting integration with Facebook and receiving user's data
- realization of the mechanism for creating and deleting auditoriums in the advertising account for further cooperation with amoCRM
- realization of adding and removing mechanism of people to audiences
- getting information about the user account and the user's profile photo (Figure 19)
- realization of interface by design layout to configure advertising campaigns in Digital Pipeline (Figure 20)



FIGURE 19. Facebook User Profile

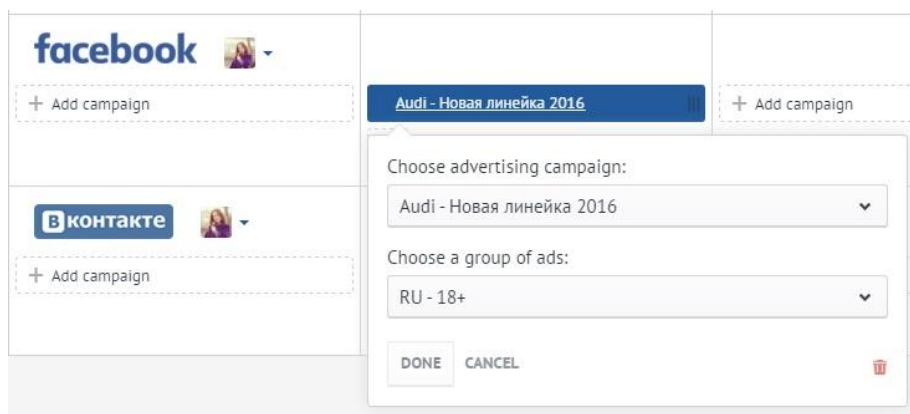


FIGURE 20. Facebook Settings

4. Integration with advertising campaigns in VK

- realization of service for working with VK - creation of application in VK, which is working as an API client, and implementation of API for amoCRM interacting with advertising accounts in VK
- realization of oauth authentication for connecting integration with VK and receiving user's data
- realization of the mechanism for creating and deleting auditoriums in the advertising account for further cooperation with amoCRM
- realization of adding and removing mechanism of people to audiences
- getting information about the user account and the user's profile photo (Figure 21)
- realization of interface by design layout to configure advertising campaigns in Digital Pipeline (Figure 22)

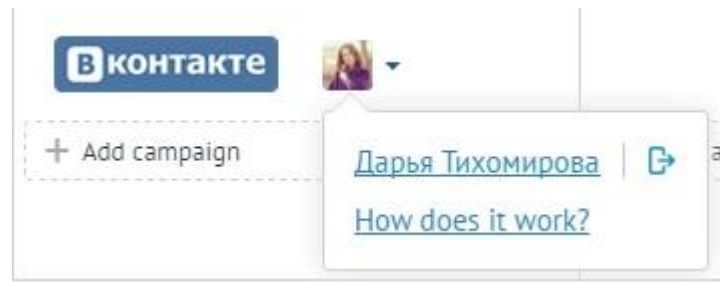


FIGURE 21. VK User Profile

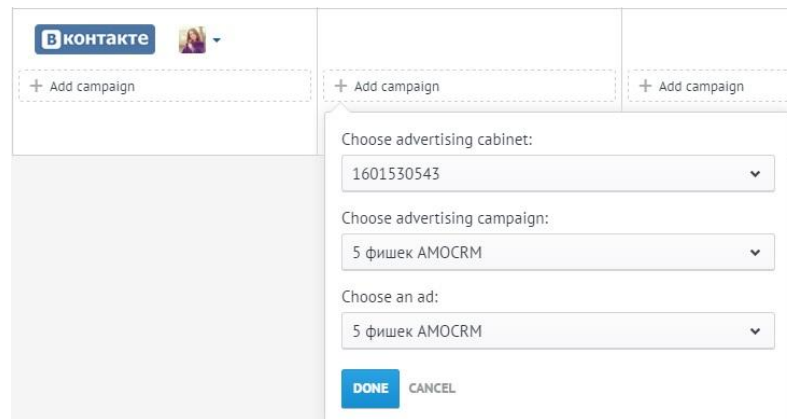


FIGURE 22. VK settings

5. Action Add a To-Do

- realization of the automatic mechanism for adding a to-do
- realization of interface of configuration of setting tasks by design layout (Figure 23)

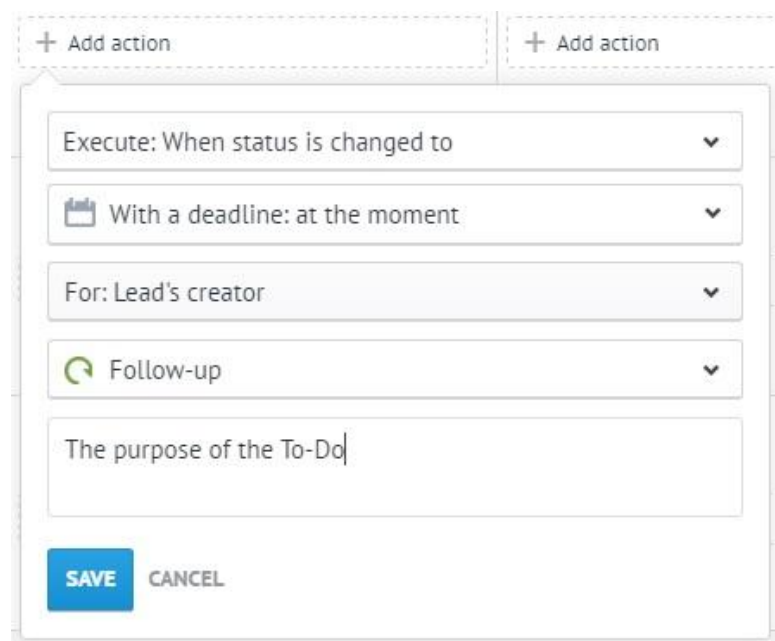


FIGURE 23. Settings of To-Do action

6. Action Send a letter

- realization of the mechanism for interaction with the mail service to send emails
- realization of interface of configuration settings to send letters by design layout (Figure 24)

FIGURE 24. Setting of the Send letter action

7. Events Incoming call, Incoming email and Online Chat

- realization of mechanism for responding to an incoming call to the system, an incoming letter, emergence of online chat with a customer (Figure 25)
- realization of the mechanism for automatically changing the status of the lead when an event occurs

FIGURE 25. Events settings

After completing the tasks of the first iteration (it took 5 sprints), one sprint was devoted to the implementation of the developed functional to Production server and making the Digital Pipeline settings in the account of company's sales department. This implementation allowed identifying the weaknesses and defects of the functionality and making necessary changes in the next iteration before the release.

For example, completion of the VK integration was required. Since communication of amoCRM and VK occurs through the API, amoCRM sent various requests while maintaining the settings, removing the ads, and adding users to the audiences. In the VK API client there are limitations on the number of requests per second and per hour per user. In the first version of the product we sent a lot of different requests within a single action. Therefore, query optimization was required to reduce the number of blocks and for the smooth operation of the service.

The second iteration – the general release of new functionality. The iteration lasted from the 4th of April till 23rd of April.

Tasks for the second iteration were:

1. Implementation of Time-machine
 - mechanism for setting deferred actions
 - realization of interface to configure deferred actions in accordance with the design layout (Figure 26)

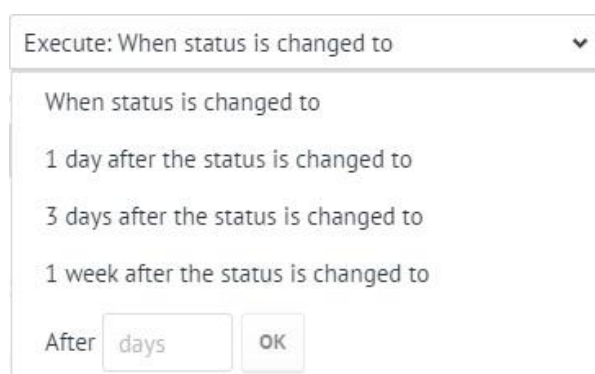


FIGURE 26. Time machine settings

2. Implementation of advertising campaigns for a few statuses
 - as well as drag already set advertising campaigns to other statuses

3. Implementation of Pixel to track visits of users to the sites

- development of the script for implementing on the site to track conversions
- implementation of script for issuing cookies to users in the feedback form for the website (this is our form integrated with the amoCRM account) and in online chat JivoSite
- realization of matching mechanism for cookies, customized websites and sending notifications to the Digital Pipeline service

4. The event Visit to the site

- integration with developed Pixel
- realization of interface for settings of the event in accordance with the design layout (Figure 27)



FIGURE 27. Settings of the Site visit event

5. Implementation of the integration with Google Analytics

- implementation of existing mechanism of sending information to the Google Analytics
- realization of additional logic for Digital Pipeline

6. Implementation of action Webhook

- realization of data sending mechanism when an event occurs
- realization of interface for action settings (Figure 28)

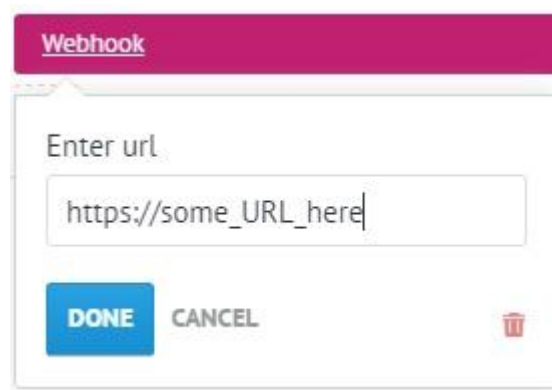


FIGURE 28. Webhook settings

7. Integration with mailing lists in MailChimp

- realization of service for working with MailChimp - creation of application in MailChimp, which is working as an API client, and implementation of API for amoCRM interacting with advertising accounts in MailChimp
- realization of oauth authentication for connecting integration with MailChimp and receiving user's data
- realization of adding and removing mechanism of people to mailing lists
- getting information about the user account and the user's profile photo (Figure 29)
- realization of interface by design layout to configure mailing lists in Digital Pipeline (Figure 30)



FIGURE 29. MailChimp User Profile



FIGURE 30. MailChimp settings

8. Implementation of editing statuses and pipelines in the Digital Pipeline interface
 - Editing, creating, deleting statuses
 - Editing, adding, deleting pipelines

9. Separation of the functionality for Russia and the USA
 - separation of the databases
 - separation of available integrations (for example, integration with the VK is not available in the US)

10. Optimizing the number of API requests to the VK

11. Checking user subscription plan and turning off the Digital Pipeline service if the plan is lower than Advanced

After performing the tasks of the second iteration (it took 2 sprints), the last sprint was devoted to the implementation of the functional to the Production server and testing the developed functional integrated with the results of the work of rest of the teams. During testing some problems and inconsistencies were identified and they were corrected before the release.

After analyzing the entire project development and implementation of new functionality to the system, it is possible to note the following:

- the main criterion of the success of the project is the quality. This includes the ability to detect and correct errors on the early stages of the implementation of tasks, as well as full compliance with the results of customer's project requirements. As a result, the duration of the project increased due to the con-

stant improvements and adjustments to the requirements of the functional.

Since the project had a clear deadline (presentation of the company) and quality was a key point for the customer, part of the tasks from the original Product backlog has not been realized. These tasks will be implemented in the third iteration of the project, after the presentation;

- in the middle (after the first iteration) and at the end (after the second iteration) the development team was in constant stress, as when new functionality was moved to the Production server, unfinished parts of the functional were detected, arising from the differences between the Development server and the Production server. After the second iteration the results of teams of other companies were integrated, which resulted in unpredictable errors and conflicts in the system.

As has been noted previously, an iterative method of project management is effective, because specification and demonstration of each version of the software is quite frequent, after the end of each sprint. With this approach, it is obvious that errors occurred on the early stages can be corrected immediately (unlike waterfall model where errors can be detected only at the testing phases). In addition, the product obtained at the end of the project is more consistent with customer requirements, than the “black box”, as in the case of the waterfall model. Accordingly, based on these findings, it may be noted that the deviation of the quality of the product obtained by the scrum methodology is much smaller than deviations in quality of the product obtained by the waterfall model.

5. CONCLUSION

The results of the study are the following.

In this study definitions of “project” and “IT project” were analyzed to identify distinguishing characteristics of the last one. It was found out that the management of an IT project is a lot different from the management of the project not related to information technology.

Here the indicator-criteria were analyzed, which show the effectiveness of the project (budget, duration, area of coverage, quality). The study revealed that changing one of these parameters has a significant impact on all others.

The concept of “IT project life cycle” was considered and the advantages and disadvantages of the main models were analyzed: the waterfall (traditional), iterative and spiral models. As a result, it was discovered that the strengths of iterative spiral model became the basis for the creation of flexible methodologies.

As a result, considering the characteristics and benefits of flexible methodologies for IT project management, as well as after a comparative analysis of the flexible methodologies types the scrum methodology was chosen, which is the most well-defined and most suitable for the kind of “principal-agent” relationship.

The analysis shows that in a case with a rapidly changing and bad-defined requirements (such conditions as in IT projects) as well as with a low level of communication between the departments of the company, using of scrum methodology in the management of IT-projects is the most effective both for the developers and for the customer.

Speaking about developers, it should be mentioned that the coordinated work, which is defined by the specialists who perform it is much more stimulating for quality problem solving, rather than permanent project manager assignments and constant stress. For the customers the benefit is obvious: the project is being implemented faster, better and at a lower cost compared to the traditional method.

Based on the fact that this study must be written and submitted by a certain deadline, only a part of the Digital Pipeline project has been considered, which was carried out before the release (the first two iterations of the project). The third iteration was started after the release. In this iteration the team is working on tasks that were not included in the previous two iterations, but which were announced in the Product backlog. For example, integration with the advertising accounts in YouTube will be done in this iteration.

Of course, there are no perfect projects in the world. Therefore, project management can be the ideal and follow all dogmas only in theory. In practice, it is necessary to deviate from the rules, to change approaches to planning, to use several different methodologies in one project and sometimes within a single sprint. In discussed project we had to use several different methodologies: use Scrum as the basic methodology and Extreme Programming during sprints, which were devoted to the implementation of developments at the Production server. In the Sprint backlog in this case there was only one task – to implement developed functionality on the Production server. Other tasks have been identified during carrying out of this task.

Also my duties were varied depending on the needs of the team. I was not only Project Manager and Scrum Master but also acted as a tester of intermediate versions of the product and I was also able to identify errors in the functionality during implementation at the Production server.

BIBLIOGRAPHY

1. Abrahamsson, Pekka, Salo, Outi, Ronkainen, Jussi & Warsta, Juhani 2002. Agile Software Development Methods - Review and Analysis. Espoo: Otamedia Oy.
2. Beck, Kent, Beedle, Mike, Bennekum, Arie van, Cockburn, Alistair, Cunningham, Ward, Fowler, Martin, Grenning, James, Highsmith Jim, Hunt, Andrew, Jeffries, Ron, Kern, Jon, Marick, Brian, Martin, Robert C., Mellor, Steve, Schwaber, Ken, Sutherland, Jeff & Thomas, Dave 2001. Manifesto for Agile Software Development. WWW document. <http://agilemanifesto.org/>. No update information. Referred 17.03.2016.
3. Benefield, Gabrielle 2008. Rolling out Agile in a Large Enterprise. Proceedings of the 41st Hawaii International Conference on System Sciences. Conference paper.
4. Bogdanov, Vadim 2012. Project Management. A corporate system - step by step. Moscow: Mann, Ivanov and Ferber.
5. Brooks, Frederick 2010. The mythical Man-Month: Essays on Software Engineering. Moscow: Symbol-plus.
6. Cervone, H. Frank 2011. Understanding agile project management methods using Scrum. OCLC Systems & Services: International digital library perspectives, 18-22.
7. Chernykh Evgenyi 2008. Agile Project Management and its history. Quality management, 84-95.
8. Chin, Gary L. 2004. Agile Project Management: How to Succeed in the Face of Changing Project Requirements. New-York: AMACOM.

9. Cobb, Charles G. 2011. Making Sense of Agile Project Management: Balancing Control and Agility. Hoboken: John Wiley & Sons, Inc.
10. Cockburn, Dr. Alistair 2008. Using Both Incremental and Iterative Development. The Journal of Defense Software Engineering, 27-30.
11. Cohn, Mike 2009. Succeeding with Agile: Software Development Using Scrum. Boston: Addison-Wesley Professional.
12. Deemer, Pete, Benefield, Gabrielle, Larman, Craig & Vodde, Bas 2012. The Scrum primer: A Lightweight Guide to the Theory and Practice of Scrum Version 2.0. PDF document.
<http://www.scrumprimer.org/scruprimer20.pdf>. No update information.
Referred 13.03.2016.
13. Dithelm, Gerd 2004. Project management. Part 1. Saint-petersburg: Business-Press.
14. Dithelm, Gerd 2004. Project management. Part 2. Saint-petersburg: Business-Press.
15. Gorbvtsov, Gennady 2009. Project Management: Educational-methodical complex. Moscow: EAOL.
16. Gray, Clifford F. & Larson, Erik W. 2008. Project Management: The Managerial Process, 4th Edition. New-York: McGraw-Hill/Irwin.
17. Gray, Clifford F.& Larson, Erik W. 2007. Project Management: The Managerial Process. 4th Edition. Moscow: Business and Service.
18. Grekul, Vladimir 2010. Information Systems Project Management. WWW-resource. <http://www.intuit.ru/studies/courses/2195/55/info>. No update information. Referred 16.02.2016.

19. Grekul, Vladimir, Korovkina, Nina & Kupriyanov, Uriy 2011. Methodical bases of management of IT projects. Moscow: Binom.
20. Hodgetts, Paul R. & Agile Logic 2001. Absolute Agile. www.agilelogic.com. Updated 2015. Referred 17.02.2016.
21. Holtsnider, Bill, Wheeler, Tom, Stragand, George & Gee, Joseph 2010. Agile Development & Business Goals. Burlington: Morgan Kaufmann Publishers is an imprint of Elsevier.
22. Izbachkov, Uriy & Petrov, Vladimir 2008. Information Systems. Saint-Petersburg: Piter.
23. Jalote, Pankaj, Palit, Aveyjeet, Kuriyen, Priya & V.T. Peethamber 2004. Timeboxing: a process model for iterative software development. The Journal of System and Software, 117-127.
24. Kniberg, Henrik 2006. Scrum and XP from the trenches. WWW document. http://www.k2x2.info/delovaja_literatura/scrum_i_xp_zametki_s_peredovoi/p15.php. No update information. Referred 27.03.2016.
25. Koltsov, Alexander 2005. Project management. <http://www.koltsov.by/category/projectmanagement/page/5>. Updated 02.10.2015. Referred 27.02.2016.
26. Layton, Mark C. 2012. Agile Project Management for Dummies. Hoboken: John Wiley & Sons.
27. Leau, Yu Beng, Loo, Wooi Khong, Tham, Wai Yip & Tan, Soo Fun 2012. Software Development Life Cycle AGILE vs Traditional Approaches. International Conference on Information and Network Technology. PDF document. <http://ipcsit.com/vol37/030-ICINT2012-I2069.pdf>. No update information. Referred 7.03.2016.

28. Ledbrook, Louise 2012. Waterfall Project Management: An Overview. WWW document. <http://projectcommunityonline.com/waterfall-project-management-an-overview.html>. Updated 13.06.2012. Referred 7.02.2016.
29. Lewis, James P. 2006. Fundamentals of Project Management. 3rd Edition. New-York: AMACOM.
30. Mazur, Ivan, Shapiro, Vladimir & Olderogge, Natalia 2004. Project Management: Student book. Volume 2. Moscow: Omega-L.
31. Melonfire 2006. Understanding the pros and cons of the Waterfall Model of software development. WWW document. <http://www.techrepublic.com/article/understanding-the-pros-and-cons-of-the-waterfall-model-of-software-development/6118423/>. Updated 22.09.2006. Referred 17.02.2016.
32. Portny, Stanley E. 2005. Project Management For Dummies. Moscow: Williams.
33. Portny, Stanley E. 2007. Project Management For Dummies. 2nd Edition. Indianapolis: Wiley Publishing Inc.
34. Project Management Institute, 2008. A Guide to the Project Management Body of Knowledge (PMBOK Guide). Fourth Edition. Newtown Square: Project Management Institute, Inc.
35. Razu, Mark, Bronnikova, Tatiana, Razu, Boris, Titov Sergey & Yakutin, Uriy 2006. Project Management: Fundamentals of Project Management. Moscow: Knorus.
36. Roberts, Paul 2007. Guide to Project Management. London: The Economist in association with Profile Books Ltd.

37. Rubio, Xavier Amatriain & Cirera, Gemma Hornos 2008. Agile Methods in Research. Presentation. Telefonica. <http://www.slideshare.net/xamat/agile-science>. Updated 06.2008. Reffered 14.03.2016.
38. Schuh, Peter 2004. Integrating Agile Development in the Real World. Newton Center: Charles River Media.
39. Schwaber, Ken & Sutherland, Jeff 2011. The Scrum Guide. PDF document. <http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf>. Updated 07.2013. Referred 23.03.2016.
40. Tovb, Aleksandr & Tsipes Grigoriy 2003. Project Management: standards, methods, experience. Moscow: Business-Olimp.
41. Version One 2011. 6th Annual State of Agile Development Survey. PDF document. <https://www.versionone.com/pdf/6th-Annual-State-of-Agile-Development-Survey.pdf>. No update information. Referred 17.04.2016.
42. Version One 2011. 7th Annual State of Agile Development Survey. PDF document. <https://www.versionone.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf>. No update information. Referred 17.04.2016.
43. Westland, Jason 2006. Project Management Life Cycle. London, Philadelphia: Kogan Page Ltd.
44. Wolfson, Boris 2012. Flexible development methodologies. Version 1.2. PDF document. <http://adm-lib.ru/books/10/Gibkie-metodologii.pdf>. No update information. Reffered 26.01.2016.