PROCESS OF MOVING FROM WATERFALL TO AGILE PROJECT MANAGEMENT MODEL

Sanja Vukićević[1], Dražen Drašković[2]
[1]Faculty of Organizational Sciences, University of Belgrade, vukicevicsanja@yahoo.com
[2]Faculty of Electrical Engineering, University of Belgrade, drazen.draskovic@etf.bg.ac.rs

Abstract: *This paper focuses on the process of migrating from the Waterfall Project Management Model to the Agile Project Management Model. The main idea is to present the Waterfall and Agile Models, their advantages and disadvantages, as well as the procedure which should be followed in order to move from the Waterfall Model into the Agile Model. Expected risks of the process, which may jeopardize the project, have also been defined, and the means for their mitigation and elimination are put forward.*
The study has emerged as a result of insight into the project leaders and project consultants' reports. This paper will help project managers as well as developers to understand and be aware of all risks and benefits that may come up while adopting the Agile Project Management.

Keywords*: Waterfall Model, Agile Project Management, extreme programming, risk management*

1. INTRODUCTION

In addition to the right choice of technology, in software development it is extremely important to decide on the correct model of project management. In search for the pattern that would lead to the guaranteed success and safe project finalization, the first model appeared as the appropriate one – the Waterfall Model (Winston, 1970). Then, the other models kept appearing, such as: Prototyping Model, Incremental Model**,** Spiral Model, Rapid Application Development Model, Engineering Driven Model and finally in 2001 the Agile Software Development was created. The reason for such great number of models lies in the percentage of failure to manage projects which is still very high, but in the mean time, project management has also displayed certain betterment.

Table 1 shows the general percentage of project success, regardless of the project management model in the period from 1994 till 2009. In 1994 the Standish Group published the CHAOS Report asserting that 31% of projects were cancelled before they ever had got completed. Then, 53% of projects were cost and time overrun and only 16% of projects were completed on-time and on-budget. According to the Standish Group CHAOS Report in 2000, 23% of projects were cancelled, cost/time overrun had gone down to 49% and there were 28% of projects completing required features and functions on time. The last report, published in 2009, has shown that percentage of success has not that much progressed considering the time period. Namely, 32% of projects was completed successfully, 44% with delay and 24% were cancelled.

Table 1: Software Project Success Through Years According To Standish Group Chaos Reports

Year	Successful (%)	Challenged (%)	Failed (%)
1994	16	53	31
1996	27	33	40
1998	26	46	28
2000	28	49	23
2004	29	53	18
2006	35	46	19
2009	32	44	24

Project leaders use the Waterfall Model quite often. The majority of project leaders do not want to jeopardize the project by converting the management to agile methods. Nevertheless, the analyses have shown that the project success of the project managed by agile methods is almost guaranteed. According to the CHAOS Manifesto of Standish Group, published in 2011, as illustrated in Figure 1, the probability of project failure is even three times greater if the Waterfall Model was used, and not the Agile Model.

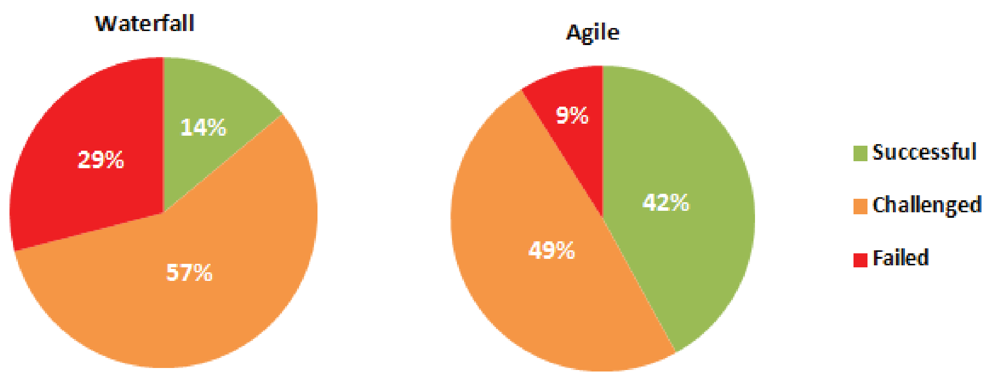

Figure 1: Differences Between Waterfall And Agile Model In Project Success

The basic problems which are encountered during software development are: failing to meet the deadlines, abandoning the projects mostly after failing to meet the deadlines, the piled up architecture, lots of bugs in code (therefore, the program is useless), project participants replacement. However, using agile methods in project management contributes to diminishing the number of the above stated problems, and the way this is achieved will be explained in the remainder of this paper.

Section 2 describes the Waterfall Model principles, while in section 3 Agile Model principles are described. The way how the Waterfall Model-led project may be transferred into the Agile Model-led project is described in Section 4. Section 5 describes specific problems arising in the course of transition to agile methods. The final section summarizes our research and provides the conclusion.

2. WATERFALL MODEL

This model is called in such a way because the model develops systematically from one phase to the other in a downward fashion, like a waterfall. The phases are: defining requirements, designing system an software, implementation and unit testing, integration and system testing and operation and maintenance (Sommerville, 2011). The linearity of this method indicates that each phase may be assigned to the separate team, and only after one phase is completed, another one would begin.

Advantages of this model lie in the fact that a lot of emphasis is dedicated to paperwork in this method when compared to the newer methods. When new workers enter the project, it is easier for them to carry on the work from where it has been left. The newer methods don't document every step in their developmental process which makes it difficult for a recently new member of the team to understand which step is going to follow. The second advantage is that Waterfall Model is well known amongst the software developers, and therefore it is easy to be employed.

There are many disadvantages of this model. Many software projects are dependent upon external factors, amongst which the product purchasers is the biggest factor. It could happen that the product purchaser changes the project requirements thereby influencing an alteration in the normal construction plan, and hence the functionality as well. In such a situation, team must return to the first phase and go through all phases again. The second negative aspect of this model is that a huge amount of time is wasted while running project. If designers are still designing software, time of developers is completely wasted. Also, because the outcome is expected at the end of the phase, usually workers are not so hardworking at the beginning of the phase, and then as the deadline is closer, they overwork in order to finish in time. The phase deadline is often missed. Another disadvantage of this method is that the testing period comes quite late in the developmental process and for that reason the design flaws are found late and all phases have to be repeated. Elaborate documentation during the Waterfall method has its advantages, but it is not without the disadvantages as well. It takes a lot of effort and time to maintain valid versions of documents.

Due to the various disadvantages there are couples of modified versions of this model. For example Sashimi Model overlaps the phases so the developers start to work on implementing design while designers are creating documents for the rest of the requirements. Since they overlap, one can return to the previous step if desired.

3. AGILE METHODS

Agile methods care for different values in relation to Waterfall Model. Individuals and interactions are more important than processes and tools. Working software is more important than comprehensive documentation. Customer collaboration is more important than contract negotiation and quick responding to change is more important than following a plan (Cohn, 2006). It may be freely said that solutions created by agile methods evolve during the development.

According to Agile Alliance there are twelve principles showing what 'agile' really means. Principles are given in Table 2 below.

Table 2: Twelve Principles Of Agile Model

No.	Agile principles
1.	The most important is to satisfy the customer through early and continuous delivery of valuable software. Hence, the satisfied customer is priority.
2.	Change request is allowed even late in development. This can be used as advantage over the competitors.
3.	Functional software is delivered often, from a couple of weeks to a couple of months. Shorter period is preferred.
4.	Business people and developers must work together daily throughout the project.
5.	Projects are built around motivated individuals. Principle is to give them the environment and support they need, and trust them to get the job done.
6.	Face-to-face conversation is the most efficient and effective method of conveying information to and within a development team.
7.	Primary measure of progress is working software.
8.	Agile processes promote sustainable development.
9.	Continuous attention to technical excellence and good design enhances agility.
10.	Keeping simplicity is essential.
11.	The best architectures, requirements, and designs emerge from self-organizing teams.
12.	The team inspects periodically good and bad procedures and accordingly tunes and adjusts its behavior in order to become more effective.

The overall product development is divided into small iterations which require minimum planning time. Iteration is the short time interval – time box (which mostly lasts one to four weeks). In every iteration a team will go through all the phases of software development (planning, analysis, coding and testing). The iteration result is product release which is functional and which may be presented to the clients. The finalized iteration implies the set of features, and each feature is precisely specified with regard to its function and is tested in details. There is no iteration duration prolongation, and no new requests addition in the course of iteration. If it is observed that the team cannot fulfill all planned requirements, the request of lower significance is tasked and they are switched off the release iteration.

Agile methods should be implemented in high dynamics environment conditions. If there is no dynamic environment, agile methods should not be applied. It could easily happen that there is lack of documentation, because the design is sometimes discussed at meetings and then no person writes it down. Also, the risk is if the customer representative is not clear about the final outcome they want. Then the project can easily get taken off track.

4. PROCESS OF MIGRATING WATERFALL TO AGILE

Agile methodology migration is viewed as a type of experiment by managers. With the beginning of the migration process from waterfall to agile project managers display fear of failure. This is the reason why this process develops gradually and there are many transitional phases. We may say that Waterfall Model mutates into Agile Model at speed which depends on project participating team capabilities.

If there is no one in team who is experienced in agile methods, than it is advised that the consultant should be employed thus monitoring the project flow and helping the team to grasp the agile project management by learning from their own mistakes. So, it is very difficult to implement all agile methods from the beginning, and that is why they are being acquired gradually during each iteration (Sureshchandra & Shrinivasavadhani, 2008). The team usually learns in the more difficult manner by deciding not to accept a method which seems complex to them and which they evaluate as being time consuming. It is only when failure occurs, the team realizes that it happened due to lack of that method usage. Thus, this is usually the most common way of adopting agile methods.

The process begins by agreeing on the way of meeting management. Agile methods favor stand-up meetings. On the first meeting, the team should make a plan defining a number of iterations to be done and which functionalities are being planned for each of them. This plan should not be elaborate, but rather an outline according to which the things would function and which would be updated after each iteration.

Iteration is composed of the following steps:

- Iteration duration time is specified and functional specification requirements for that iteration are written down. The meetings should include also the product purchasers in order to define functionalities correctly thus avoiding the huge number of change requests.
- The team is having daily meetings while there is a meeting with a product purchaser once a week.
- Coding standards should be defined before the beginning of the coding. In this way, developer hesitation and confusion is being prevented from occurring, uniformity code is achieved, code quality is enhanced and the code review and refactoring time are being reduced.
- Automatic tests are composed and used during the code development.
- Daily builds should be made during the development phase.
- Iteration result is the product version and the implemented functionalities list. The product is submitted to the team which tests the product like a user. If the bugs are detected, they are submitted to the team and the team should decide whether they would be corrected in the next iteration or be left for some later iteration.

Iterations are executed one after the other. But one iteration finalization should not be waited for in order to begin with the other, but rather, iterations should overlap (Ming et al., 2004). Figure 2 displays iterations overlapping. The team members may be divided into three groups: those composing functionalities specifications, those coding and testing and those verifying the product. Therefore, one iteration may be displayed through three phases: functional specifications writing (FS), coding with testing (C+T) and verification (V). In the first iteration, when developers work on coding phase, a product architect in collaboration with the product purchaser prepares functional specifications for the other iteration. Thus the developers after finalization of coding phase deliver a release to the verifiers and immediately initiate coding phase of the second iteration. After the verification finalization the verifiers obtain the new product release and this lasts up until the final release has been obtained.

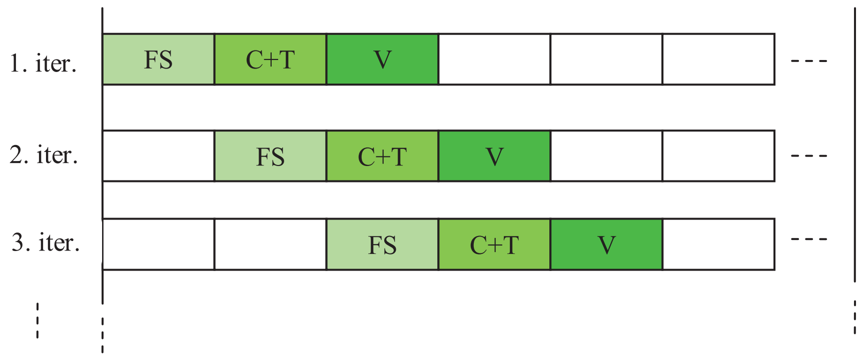

Figure 2: Overlapping Of Agile Iterations

The above mentioned agile methodologies are described in detail in the part which follows: Stand-up meetings, Burn-Down charts, Coding standards, Test Driven Development, Behavior Driven Development and Daily builds.

Stand-up meetings and Burn-Down Charts

The simplest agile methodology easily accepted by team members is in the form of stand-up meetings. A stand-up meeting is a daily team meeting held to provide a status update to the team members. During these meetings each member is asked three questions: 1. What did s/he do yesterday?, 2. What does s/he plan to do today?, 3. Are there any problems? This lasts 15 to 20 minutes at maximum and gives rhythm to iteration.

Another agile method shows how project develops by means of the Burn-Down chart. A Burn-Down chart is a graphical representation of work left to do, versus time. The outstanding work (or backlog) is often on the vertical axis, with time along the horizontal.

Coding standards

Individual style is great when you're working alone. In team software development, however, the goal is to create a collective work that is greater than any individual could create on his/her own. Because requirements are changing very offer and code is amended or fixed in iterations, it is needed to have clear code. That’s where coding standards are welcome. Code standards are guidelines on which all the developers agree to adhere while programming. However, when you are putting together your coding standards, don't fall into the trap of arguing about formatting. There are more important issues.

Test Driven Development or Behavior Driven Development

One of the differences between agile and traditional testing methods, such as the Waterfall Model of software design, is that testing of the software is conducted at different points during the software development lifecycle. Waterfall Model applies testing after complete system development. Agile Model applies techniques such as Test Driven Development where the tests are written before the code. Since the result of entire iteration is one final product version which implements certain functionalities in order to be sure that these functionalities function correctly and that their introduction has not jeopardize product remaining part, the products must be frequently tested during the iteration. The common practice is that each developer before code committing, starts up all the tests or at least those which s/he regards that cover changes. Before release announcement all the tests should be implemented successfully.

In agile approach, the other testing way is also frequent, and it takes into account sense. This one is called Behavior Driven Development where the scenarios are written before the code, after which automated suite of tests are written. Scenario represents a natural sequence of actions which are possible within the product verifying whether the code meets the required functionality and/or quality standards.

Daily builds

Frequent releases provide product purchasers with current product functionalities, so that they may check out the earlier phase functionality accuracy. Product purchasers may also give their own comments and requests which are used for next versions planning. Both daily builds and automated testing ensure that the code base is not broken after code has been checked in.

5. POTENTIAL PROBLEMS AND HOW TO HANDLE THEM

The risks accompanying stand-up meetings refer to the instances when meetings are prolonged due to problems which are elaborated by a team member and the others willing to help by providing advice. This problem is easy to solve by discussion interruption and if further solving is required additional meeting should be scheduled (Hines et al., 2009).

Many project teams insist on trying to define all of the requirements in advance. This is not just dangerous, but is actually waste of time, since you can almost be assured that a certain percentage of those requirements will be invalidated and the time spent for analyzing them will be for naught (Churchville, 2006). An agile approach to requirements definition is to defer detailed analysis as late as possible, which is typically just before the work is about to begin. Until then, you capture requirements in the form of "user stories", which are brief descriptions of customer relevant functionality.

It is really important to change project manager role from controlling the team to removing roadblocks which impend to the team's progress. Team has to suggest the action and project manager has to approve or disapprove it. Also, the estimation of remaining effort has to be left to the team. The estimation would be more realistic if it comes from the team and there would be more ownership and enthusiasm to complete the iteration in time. Also, decision making is faster. It is better to make a decision quickly even if being a bad one and thus achieve continuity, than wait for weeks on decision which may again be bad and require changes.

It is very hard to make daily builds of a product, especially in the early stage of adopting agile methods. Usually, weekly builds are done and it is satisfying modification of agile methodology as well as there is a build at the end of the week. Something is better than nothing. If project leader notice that none of the developers had checked in their code for the weekly build because they were not sure if the code was working, there is a time to include Test Driven Development. It will ensure that developed code actually meets the requirements.

6. CONCLUSION

This research is base on published reports of team members, project leaders and project consultants, all of whom migrated from Waterfall to Agile Project Management Model. The authors themselves participated in one such project.

On previous projects, we were involved in projects that were Waterfall Model-led. Frequent problems were: the team overworks during 4 to 5 weeks preceding the release, many times the release date has had to be postponed due to delay in development and testing, too many defects have been raised by the customer during acceptance testing, etc. After this, first project was carried out successful by combining agile methods - Extreme Programming (XP) and Scrum. The results were: high productivity of team members, production of high quality code, product purchasers were satisfied because they were involved in project from start to finish.

Process described in this paper is useful for the teams migrating from Waterfall Model to Agile Model. However, knowing this process could be useful for the teams willing to migrate from any other project management model to Agile.

REFERENCES

Winston, R. (1970, August). *Managing the development of large software systems. IEEE WESCON,* 329-338.
Sommerville, I. (2011). *Software Engineering.* United States: Addison-Wesley.
Cohn, M. (2006). *Agile estimating and planning*. United States: Prentice Hall PTR
Sureshchandra, K. & Shrinivasavadhani, J. (2008, August). *Moving from Waterfall to Agile*. Paper presented at AGILE '08 IEEE Conference.
Ming, H., Verner, J., Liming, Z. & Babar, M.-A. (2004, September). *Software quality and agile methods*. Paper presented at Computer Software and Applications IEEE Conference.
Hines, L., Baldwin, S., Giles, M. & Peralta, J. (2009, July) Implementing agile development in a waterfall project. *IBM WebSphere Developer Technical Journal.*
Churchville, D. (2006). *The Seven Deadly Sins of Software Project Management*. Retrieved from http://syberspawn.itgo.com/it/docs/7-PM-sins.pdf