Journal of Applied Economics and Business

OPEN ACCESS ARTICLE

Measuring agility in agile methodologies

Magdalena Maneva, Natasa Koceska, Saso Koceski

Faculty of Computer Science, University Goce Delchev, Stip, Macedonia

magdalena.21078@student.ugd.edu.mk, {natasa.koceska, saso.koceski}@ugd.edu.mk

Abstract

Agile software methodologies are among the most rational development patterns in global economic environment on which software development enterprises rely. They help software projects to face the dynamic changing business requirements of the customers. However, there is no standard tool for measuring agility and selecting the particular agile method for a particular project. The purpose of this paper is to review the existing tools for measuring agility in agile methodologies, as well as measuring agility in software development teams. This can help in decision-making processes regarding the adoption of an appropriate agile method for a particular project.

Keywords

Traditional Methodology; Agile Methodology; Software Development; Agility.

INTRODUCTION

Agile methodology in software development arises in response to meet the costumers need for receive a quality software in a short time. Traditional approaches are based on rigidly process that include a structured layout of step-by-step approach from requirements gathering to final testing and releasing the product [1]. These methods do not encompass or cannot changes until the whole cycle is complete. On the other hand, the agile development is based on the idea of incremental and iterative development, in which the phases within a development life cycle are revisited repeatedly [2, 3]. It advocates adaptive planning, early delivery, and continuous software improvement based of a customer feedback. Each iteration in agile development is treated as a separate mini-project with activities that include specifying requirements, design, implementation, and testing. The result of each iteration is the distribution of a small part of the software product that is functional and which can be the basis for specifying further requirements. Through these frequent iterations, the predictability and efficiency of the project itself increases.

Agile methods actually are a family of development processes, not a single approach to software development. The base of this is the Agile Manifesto, widely regarded as the canonical definition of agile development and accompanying agile principles [4]. The Agile Manifesto states that:

- **Individuals and Interactions** are over processes and tools
- **Working Software** is over comprehensive documentation
- **Customer Collaboration** is over contract negotiation
- **Responding to Change** is over following a plan

The methodologies that promote agility, basically have the same principles and differ only in their practical application. The agile methodologies are usually used for small and medium-sized projects. However, there are studies that analyse the implementation of ASDM in large projects [5].
In the following, we will briefly explain some of the most used agile methodologies.

TYPES OF AGILE METHODS

Extreme Programming (XP)

This method got its name since it raised the usual development principles to some extreme level. Extreme Programming is one of the most agile methodologies that aim to improve the quality of the software and at the same time to respond positively on changes in user requirements. As an agile methodology, extreme programming reduces the entire development process to several smaller development phases [Fig.1]. Therefore, developers can pay more attention to the developing phases and give more frequent reports related to the functionalities of the software product. Separating the development process in multiple phases and the ability to deliver reports to the user after each of those short phases opens up the possibility of receiving feedback from the users.
Extreme programming supports work in teams. Managers, clients and developers are equal partners in a collaborative team. Developers are in a constant communication with costumers that allows them to get quick feedback in order to make changes that are required by clients.

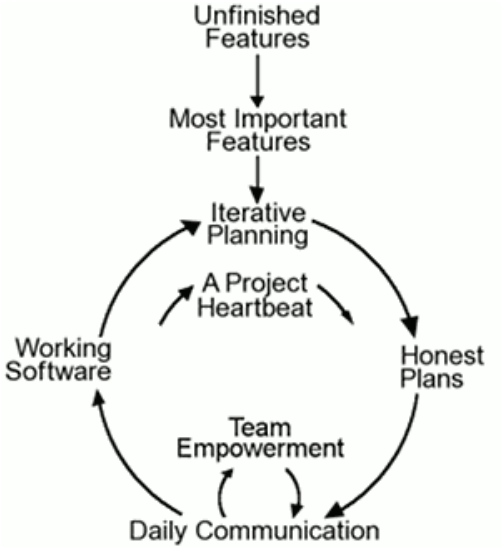

FIG.1. XP PROCESS

Scrum

Scrum supports the theory of an iterative and incremental approach to optimize predictability and risk control [Fig.2]. The three features that support this theory are: transparency, inspection, and adaptation. Transparency requires all aspects to be defined by a certain standard, for all participants in the process to understand what they are seeing. Participants in Scrum should make frequent inspections of the work done, yet these inspections should not be very frequent because they would interfere with the working process. If one of these inspections concludes that the outcome of the process / product will not be in line with expectations, then it should be adjusted. These adaptations are made as fast as they can to reduce further damage. Scrum uses four types of inspiration and adaptation: Sprint Planning, Daily Scrum, Sprint Review and Sprint Retrospective.
The Scrum teams consist of: Product, owner, Development team and Scrum master.

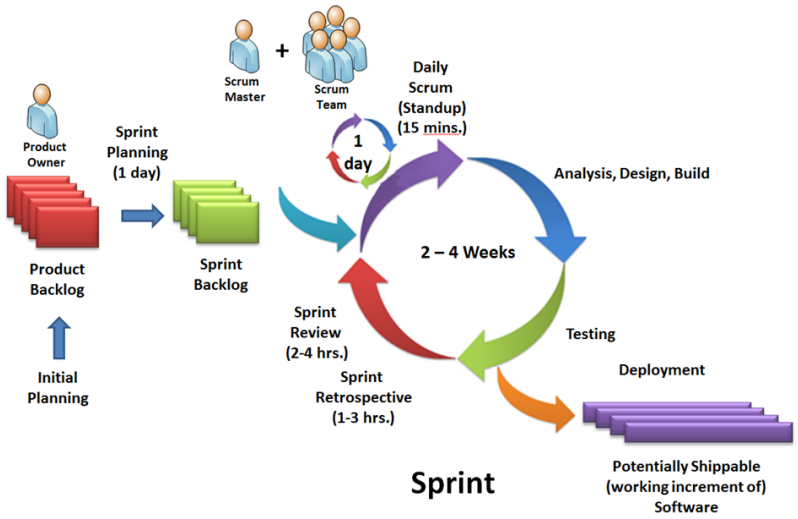

FIG.2. SCRUM DEVELOPMENT PROCESS

Crystal methods

Crystal Methods are a family of multiple methodologies created by Alistair Cockburn in 1998. Each methodology is specific and is used for different types of projects. However, all these methodologies emphasize the user's participation in software development and focuses mainly on people, interaction, skills, communication, etc. Crystal methodologies place a great emphasis on the communication of people involved in the project. Larger projects require more complex methodologies because they involve more people and therefore need better coordination, while projects that are more critical need a more rigorous approach [Fig.3].

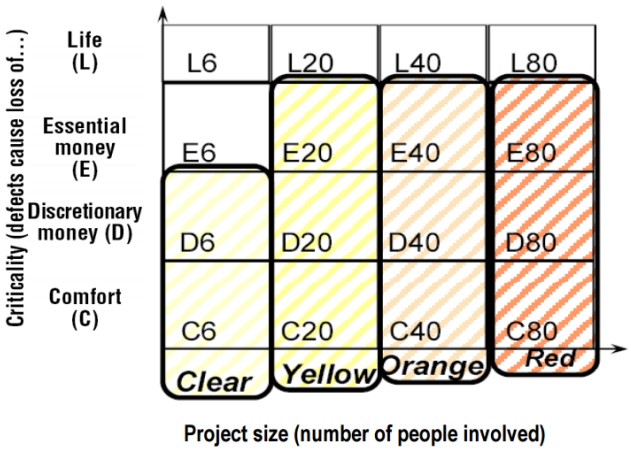

FIG.3. CRYSTAL FAMILY

Adaptive Software Development (ASD)

Adaptive software development was introduced by James Highsmith in 1997. It is based on Rapid Application Development (RAD). This methodology supports incremental and iterative development using prototyping. ASD replaces the traditional waterfall model with a repeating three-step series [Fig.4]:

- Speculate
- Collaborate
- Learn

ASD has no defined principles or procedures like other development methods and therefore does not present itself as a methodology for creating software projects. It is rather an approach that should be used by organizations that apply agile methods.

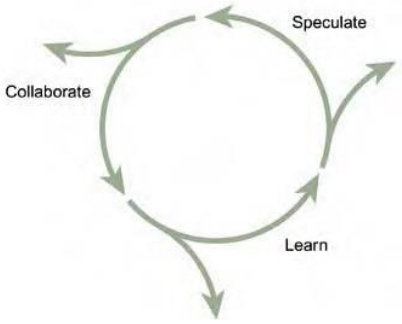

FIG.4. ASD LIFECYCLE

Dynamic System Development Method

Dynamic System Development Method is one of the leading agile methodologies today that gives a greater discipline to the RAD method. The main idea of this methodology is to define detailed strategic goals and to focus on early delivery. Although this methodology is successful in managing the small projects, it also focuses on larger projects that are important for the business environment .

This method includes many of the basic concepts of the Agile Manifesto such as iteration, incremental delivery, and client involvement [Fig.5].

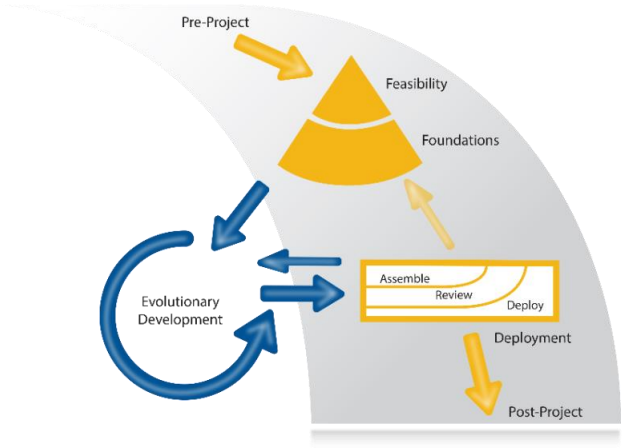

FIG. 5. DSDM PROCESS

Lean Software Development (LSD)

Lean software development is a translation of lean manufacturing principles and practices to the software development domain. Adapted from the Toyota Production System it is emerging with the support of a pro-lean subculture within the Agile community. Lean offers a solid conceptual framework, values and principles, as well as good practices derived from the experience that supports agile organizations.

The basic principle of "Lean" production is to produce exactly what the client likes, in the context of type, quality and quantity of products.

Lean development can be summarized by seven principles, very close in concept to lean manufacturing principles:

- Eliminate waste
- Amplify learning
- Decide as late as possible
- Deliver as fast as possible
- Empower the team
- Build integrity in
- See the whole

Kanban

Kanban is a lean software development methodology [6], [7] that focuses on just-in-time delivery of functionality and managing the amount of work in progress. When used for software development, Kanban uses the stages in the software development lifecycle (SDLC) to represent the different stages in the manufacturing process. The aim is to control and manage the flow of features (represented by Kanban cards) so that the number of features entering the process matches those being completed [Fig.6].

Kanban allows the software be developed in one large development cycle. Despite this, Kanban is an example of an agile methodology because it fulfils all twelve of the principles behind the Agile manifesto, because whilst it is not iterative, it is incremental.

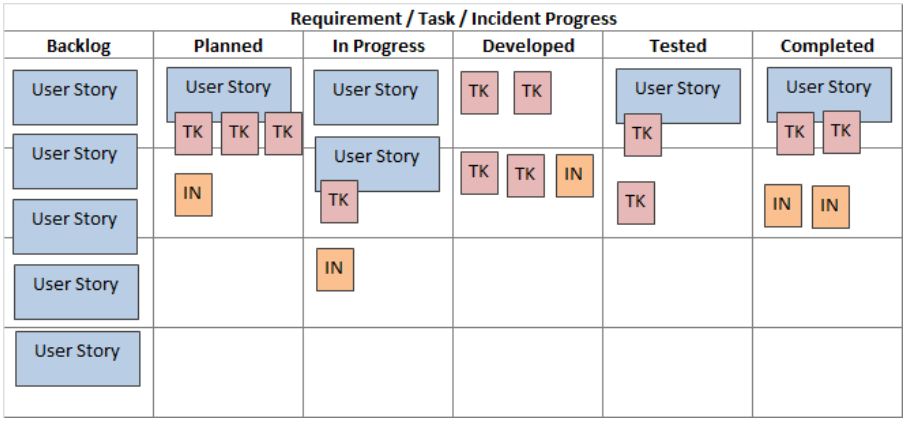

FIG. 6. KANBAN METHODOLOGY

TOOLS FOR MEASURING AGILITY

"Agility" can be defined as ability to respond to unpredictable changes with quick response and profitability. As agility is present in all the industries, it is very important to measure it in order to determine responsiveness of an enterprise to external turbulences. Although it is very difficult to measure agility, there are still some tools that are used and are worth mentioning.
The tools for measuring agility are divided into two categories, those that measure agility of agile methodologies and those that measure the agility of software development teams.

Measuring agility of agile methodologies

Bohem and Turner [8] presented a tool for creating a balance between agility and discipline. According to them, discipline is a key success factor for any project, while agility it as part of the discipline. The combination of these two values contributes to the success of the organization. In their research they defined five "critical decision factors" that can be used to estimate whether it is better to apply agility or detailed planning for a given software project.

- The size of the project team
- Critical damage from unplanned defects
- The necessary culture for balancing between chaos and order
- The dynamics of the team working in chaos or with detailed planning
- Staff dealing with Cockburn's ranking skills [9]

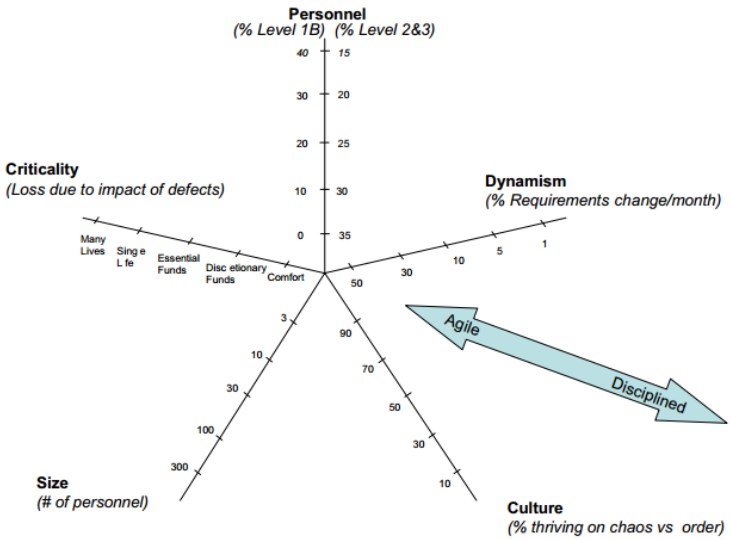

FIG. 7. DIMENSIONS AFFECTING METHOD SELECTION

According to the diagram in Fig. 7, if the rating of the five factors is closer to the center, then it can be said that the team is agile. Otherwise, the team monitors the approach of discipline.

Philip Taylor [10] has made a modification of the tool created by Boehm and Turner and added a sixth axis called "Client Involvement", while Williams proposed to access XP practices that were appropriated by the organization. The Williams approach consists of three parts:

1. XP-CF (Context Factors) - team size, project size, experience of an employee;
2. XP-AM (Adherence Metrics) - displaying the practices of the team in a precise way
3. XP-OM (Outcome Measures) - tool to estimate the outcome of the project using all or part of the XP practices.

Datta introduced a metric that would help determine which agile methodology best corresponds to a given project by identifying five dimensions: duration, labor, risk, interaction, and news. By assigning values to each of these dimensions, it identifies which model for software development best suits.

CEFAM was created by Taromirad and Ramsin in order to cover the most important aspects of agile methodologies. This tool consists of multiple evaluation criteria that are divided into five groups: Process, Agility, Usage, Modeling Language, and Context [Fig. 8]. Each group has multiple questions that can be answered with a numerical value, Yes or No or a selection of multiple offered answers, but in the end the responses are evaluated according to the scale:

- Unacceptable≤0.25
- 0.25 <Low <0.5
- 0.5 <Average ≤ 0.75
- 0.75 <High ≤ 1

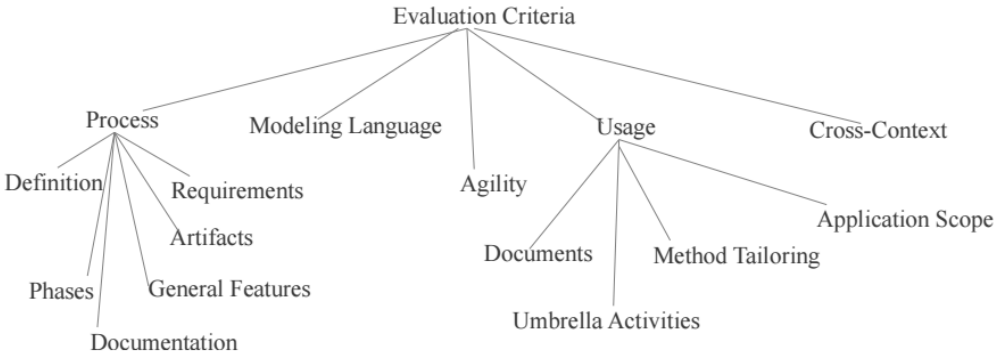

FIG.8. EVALUATION CRITERIA ACCORDING TO CEFAM

Measuring agility of software development teams

Thoughtworks [11] is a worldwide widespread consulting company that developed an online questionnaire for accessing agility based on 20 multi-choice questions. These issues are based on demand analysis, business response, collaboration and communication, and project management. If the user answers the questions, the website calculates the level of agility of their team.

Another similar tool is the 42-Point Test tool, which is a test with 42 questions. This tool, created by Waters, is created for Scrum or XP teams to help determine whether they follow the most agile practices.

The Escobar - Vasques model for agility assessment was created by Escobar Sarmiento and Linares Vasques and consists of four stages. For the first three phases, they used models and tools proposed by other researchers:

- Agile Prject Management Assessment - proposed by Quimer and Henderson Sellers [12]
- Project Agility Assessment - proposed by Taylor [10]
- Workteam Agility Assessment - proposed by Leffingwell [13]
- Agile Workspace Coverage

In order to collect data from measurements, tool-based surveys were used in each phase, while in the last step they used their own survey. Then they split the data into four axes on a radar diagram in order to show the agility of the organization [Fig. 9].

SAMI (Sidky Agile Measurement Index) was created by Sidky [14] in order to measure the agility as a work from the "Agile Adoption Framework". SAMI is actually a scale used by an agile manager to identify the team's potential and the project. SAMI consists of 5 agile levels and 5 agile principles that together form a 5 x 5 matrix. These principles come from the Agile Manifesto.

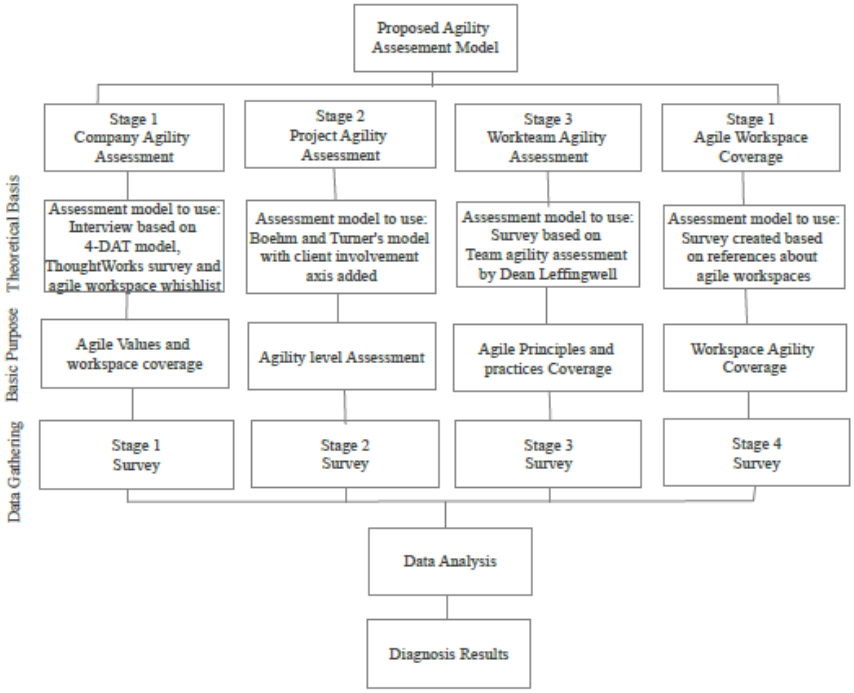

FIG.9. ESCOBAR-VASQUEZ MODEL

CONCLUSION

Agile project management is about the ability to manage and adapt to change. Agile methodologies were developed to provide more customer satisfaction, to shorten the development life cycle, to reduce bug rates, and to accommodate changing business requirement during the development process. They emphasize on teams, working software and customer collaboration.

In this paper, we explained some of the most used agile methodologies, as well as some tools for measuring the agility. As agility is present in all the industries, it is very important to know the comprehensive tools for measure it, as a necessity in order to determine responsiveness of an enterprise to external turbulences.

REFERENCES

[1] W. Cunningham (2001). Manifesto for Agile Software Development. http://www.agilemanifesto.org Accesed on: 10.08.2017

[2] Szalvay, Victor. An Introduction to Agile Software Development. Danube technologies Inc. 2004.

[3] Larman, C. & Basili, V. "A History of Iterative and Incremental Development," IEEE Computer, vol. 36, no. 6, pp. 47-56, June 2003.

[4] Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Humt, A., Je®ries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S.,

Schwaber, K., Sutherland, J., Thom, D.: Manifesto for agile software development. (2001) http://agilemanifesto.org Accesed on: 10.08.2017.
[5] Stober T., Hansmann U. (2010) Traditional Software Development. In: Agile Software Development. Springer, Berlin, Heidelberg.
[6] Maneva, M., Koceska, N., Koceski, S. "Introduction of Kanban methodology and its usage in software development." In: ITRO 2016, 10 June 2016, Zrenjanin, Serbia (2016) pp: 52-54.
[7] Kirovska, N., Koceski, S. "Usage of Kanban methodology at software development teams." Journal of Applied Economics and Business 3, no. 3 (2015): 25-34.
[8] B. Boehm, R. Turner (2003). Observations on balancing discipline and agility. In: Proceedings of the Agile Development Conference, 2003. ADC 2003. pp. 32–39
[9] A. Cockburn (2002). Agile software development. Agile software development series, Addison-Wesley
[10] P. Taylor, D. Greer, P. Sage, G. Coleman, K. McDaid, I. Lawthers, R. Corr (2006). Applying an agility/discipline assessment for a small software organisation.
[11] ThoughtWorks (2016). https://www.thoughtworks.com/ Accesed on: 10.08.2017.
[12] A. Qumer, B. Henderson-Sellers (2006). Measuring agility and adoptability of agile methods: A 4-dimensional analytical tool. Procs. IADIS International Conference Applied Computing 2006.
[13] D. Lengwell (2007). Scaling Software Agility: Best Practices for Large Enterprises (The Agile Software Development Series). Addison-Wesley Professional.
[14] A. Sidky (2007). A structured approach to adopting agile practices: The agile adoption framework. Ph.D. thesis, Virginia Polytechnic Institute and State University.