Key elements for the success of the most popular Agile methods

Flaviu FUIOR

Institutul Național de Cercetare-Dezvoltare în Informatică – ICI București

flaviu.fuior@ici.ro

Rezumat: Metodologia Agile, care a cunoscut o creștere semnificativă a influenței și utilizării în industria de dezvoltare IT în ultimele două decenii, se bazează pe diverse metode mai noi sau mai vechi. În această lucrare, vor fi identificate și descrise cele mai populare și utilizate metode aparținând metodologiei Agile, bazate pe sondajele publicate de VersionOne în ultimii trei ani. În timpul prezentării metodelor, vor fi evidențiate principalele elemente care fac ca aceste metode să aibă succes. Abordarea din articol este ca după identificarea elementelor distincte ale fiecărei metode, să nu fie comparate aceste elemente cheie, ci mai degrabă să se evidențieze capabilitățile lor complementare. Aceasta este o condiție prealabilă pentru un viitor studiu pentru a propune o nouă metodă Agile bazată pe aceste elemente cheie.

Cuvinte cheie: metodologie Agile, Scrum, Kanban, Scrumban, hibrid Scrum / XP.

Abstract: The Agile methodology, who knew a significant increase in influence and use by the IT development industry in the past two decades, is based on various newer or older methods. This paper will identify and describe the most popular and used methods belonging to Agile methodology based on published surveys by VersionOne in the last three years. During methods presentation, the main elements that make these methods so successful will be highlighted. The approach proposed is to identify these distinctive elements of each popular method and not to compare these key elements but rather to point out their complementary capabilities. This is a pre-requisite for a future study to propose a new Agile method based on these key elements.

Keywords: Agile methodology, Scrum, Kanban, Scrumban, Scrum/XP hybrid.

1. Introduction

In February 2001, a group of seventeen developers gathered in Snowbird, Utah, to talk about the future of a new generation of software development methods, also known as light-weight software development methods.

Light-weight software development methods appeared all through the ’90s (Scrum in 1995, Extreme programming (XP) and Crystal Clear in 1996, Feature-Driven Development in 1997) and developed in response to the prevailing heavy-weight software development methods (such as Waterfall) that critics described as overly regulated and planned (Gibson, 2011), (O'Sheedy, 2012), (Wikipedia, 2019).

In this meeting, one of the major outcomes was the developers’ agreement to the term “agile” as the type of method that was capable to respond rapidly to modifications in software project requirements. Another fundamental result was that developers formed the Agile Alliance and wrote Manifesto for Agile Software Development and the Principles Behind the Agile Manifesto (O'Sheedy, 2012), (Williams, 2007).

The Manifesto for Agile Software Development it's a statement from the founders about what they value the most (Beck, 2001):

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

https://doi.org/10.33436/v29i4y201901

The twelve principles behind the Agile Manifesto have the following meanings (Beck, 2001):

- To obtain customer satisfaction via early and continuous software delivery;
- Integrate changing requirements throughout the entire development process;
- Deliver working software as frequently as possible;
- Teamwork between the business stakeholders and development team throughout the project;
- Motivate, trust and support all people involved;
- Encourage face-to-face interactions;
- The main measure of progress is to deliver working software;
- Agile processes to support a steady development pace;
- Permanent attention to technical detail and design increases agility;
- Simplicity – Deliver just enough to get the job done;
- Great architectures, requirements, and designs came out from self-organizing teams;
- Periodical reflections on how to become more effective and then apply changes.

The purpose of this article is to identify and to describe the most popular and used methods from the Agile methodology and from their presentation to point out and highlight the main elements that make these methods so successful.

In the second section, after the introduction section, are listed Agile methods and practices and the result of a survey about which methods are most used. Scrum, Kanban, Scrumban, and Scrum/XP hybrid methods are presented with details further on in the second section.

In the third section are identified and highlighted the key elements that make these four methods so popular and used. The last section presents conclusions that most of the key elements for the four methodologies are complementary rather than competitive and represent the foundation for a new Agile method that can be built.

2. Agile Methods and Practices

The most known and used Agile methods are listed below with the remarks that this is not an exhaustive list of Agile methods and practices and the list is sorted on alphabetical criteria in each decade (Agile-Mercurial, 2019):

- In the 1940s:
    - Kanban;
- In the 1980s:
    - Iterative Development;
    - Lean Software Development;
    - Rapid Application Development (RAD);
    - Rational Unified Process (RUP);
    - Unified Software Development Process (USDP);
- In the 1990s:
    - Crystal family of methods;
    - Dynamic Systems Development Methodology (DSDM);
    - Extreme Programming (XP);
    - Feature Driven Development (FDD);
    - Scrum;
    - Test-Driven Development (TDD);
- In the 2000s:
    - Adaptive Software Development (ASD);

- Complex Adaptive Systems (CAS);
- Disciplined Agile Delivery (DAD);
- Scrumban;

- In the 2010s:
  - Nexus;
  - Scaled Agile Framework (SAFe)
  - Sustainable Cultural Agile Release in the Enterprise (SCARE)

It is clear that the Agile methodology with its numerous methods is a vast domain and cannot be covered in just one article, and this is the reason a selection criterion was used.

In Table 1 are presented the results of the annual survey organized by VersionOne in "State of Agile Report" for three consecutive years (2015, 2016 and 2017) regarding Agile methods and practices used (VersionOne, 2016, 2017, 2018). Survey respondents declared in a very high percentage that their organization is using Agile methods and practices (95% in 2015, 94% in 2016, 97% in 2017) (VersionOne, 2016, 2017, 2018).

Table 1. Results of the annual survey regarding Agile methods and practices used

Method	**Survey 2017**	**Survey 2016**	**Survey 2015**
Scrum	56%	58%	58%
Hybrid (multiple methodologies)	14%	8%	8%
Scrumban	8%	8%	7%
Scrum/XP hybrid	6%	10%	10%
Kanban	5%	5%	5%
Iterative Development	3%	2%	3%
Extreme Programming (XP)	1%	<1%	1%

By using the result of the survey as the selection criteria, in the following are presented four Agile methods: Scrum, Scrumban, Scrum/XP hybrid, and Kanban. Hybrid (multiple methodologies) will be left outside this presentation because is itself a subject for a paper since we need to understand all Agile methods and how to combine them. The presentation will start with Scrum, Kanban, and will follow by the hybrid methods Scrumban and Scrum/XP.

2.1. Scrum

The first mention in the literature for the expression Scrum was in an article written by Takeuchi and Nonaka in the year 1986 (Abrahamsson, 2002). Scrum, called after the scrum in Rugby, was at the beginning developed by Ken Schwaber and Jeff Sutherland in the early '90s and later join the collaboration, Mike Beedle.

Scrum is a process framework used to continually improve product features, team efficiency, and the working environment. Scrum Teams are the main elements of the Scrum framework and are self-organizing teams. The team engages in fulfilling a goal in a period (sprint) and is given the command, autonomy, and responsibility to act in meeting the target in the timeframe (Williams, 2007).

Artifacts

Artifacts outlined by Scrum are precisely designed to maximize clarity of crucial information so that everyone interested has a common understanding of the artifact. There are three major artifacts created by the Scrum team: the Product Backlog; the Sprint Backlog; and the Increment.

The Product Backlog is a sorted list of all features that are known to be needed in the future for a product. Product Backlog is the unique source of requirements for a specific product, it will never be completed, and the responsible actor for the content and sorting is the Product Owner (Schwaber, 2017). In the Product Backlog, for each specified requirement should be present at least the following elements: a unique identifier, type of demand (new feature, improvement, issue), the current status, the priority, and the effort estimation for the requirement.

The Sprint Backlog is a list with the requirements from the Product Backlog that are selected to be part of the current iteration, also known as Sprint (Schwaber, 2017). In the Sprint

Backlog, requirements are split into tasks, and all are detailed at least with the following information: task description, what requirement generated the task, who is responsible for solving the task, the current status, and how much time is needed to finish the task. Sprint Backlog is presented and discussed in the Daily Scrum meeting and needs to be updated with the latest information daily.

The Increment is a leap toward the goal and represents the amount of all the Product Backlog tasks finished during a Sprint, and the value of the increments done in all previous Sprints (Schwaber, 2017).

Roles

The **Product Owner** is the responsible person for creating, updating, prioritizing, and offering clearness and visibility for the Product Backlog. Also, the Product Owner together with other stakeholders are in charge of deliverable inspection at the end of the Sprint.

The **Scrum Master** has the responsibility to promote and support Scrum and to act as a servant-leader for the Scrum Team (Schwaber, 2017). The Scrum Master is not only a management position; he/she is also a team member and is involved in product development. The main responsibilities for the Scrum Master are: to understand and reinforce the product Sprint and goals and the Scrum practices and values, to lead the Daily Stand up Meeting and the Sprint Review, to monitor the progress, to remove any impediments that appear for the Scrum team during the Sprint and to provide resources to meet Sprint objectives (Williams, 2007).

The Scrum Team members, known as **Developers,** are the professionals that are committing to accomplish the Sprint Goal and have all the authority needed to take all the measure to achieve the goal.

Process

An overview of the Scrum process is presented in Figure 1.

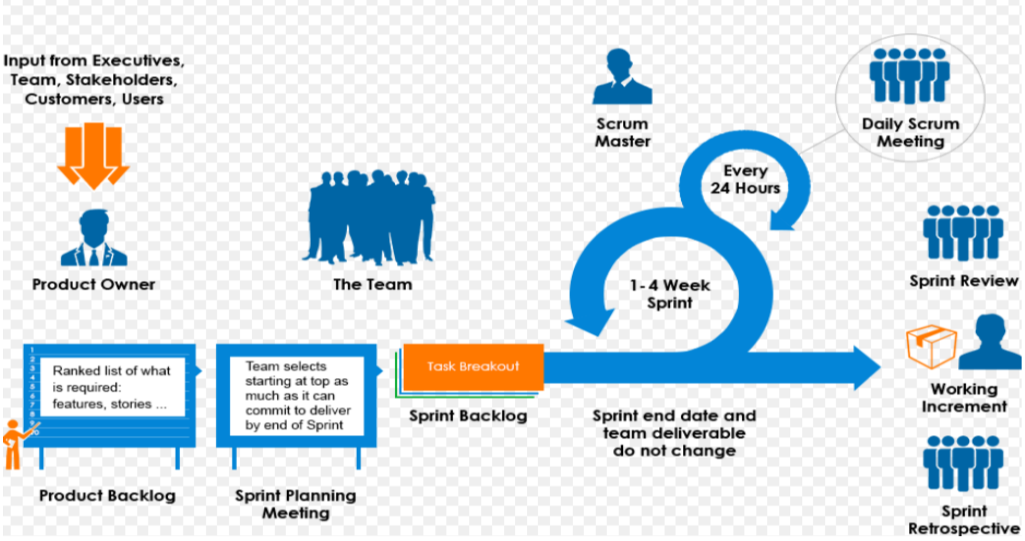
Input from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
Scrum Master
Daily Scrum Meeting
Every 24 Hours
1-4 Week Sprint
Sprint Review
Ranked list of what is required: features, stories ...
Team selects starting at top as much as it can commit to deliver by end of Sprint
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Working Increment
Product Backlog
Sprint Planning Meeting
Sprint Retrospective

Figure 1. Scrum process (Imgbin, 2019)

The Scrum process has the following elements in its composition:

- At the Sprint Planning meeting are attending Product Owner, Scrum Master and the development team and the main support document is the Product Backlog which was already updated and prioritized by Product Owner. In this meeting is establish what features are taken to be developed in the next Sprint (increment period of 1-4 weeks) and also is set a Sprint Goal which represents high-level success criteria for the Sprint in order to provide the team with an overview image. Once the perimeter of the Sprint is established, no re-prioritization will take place during the Sprint in which features sliced in tasks are designed, implemented, and tested.

- Every day of the Sprint, are held Daily Scrum Meeting of 15 minutes, and the mandatory attendances are Scrum Master and Development team. This meeting is an important element of Scrum methodology because it is reviewed work done by the team in the last 24 hours and is anticipated work to be done in the next 24 hours. For this meeting to have the expected effect, the number of attendances should be no more than seven persons. In the case of a bigger team, the team will be split into more Scrum teams. During this short meeting, every team member needs to answer the following three questions:
  - What have I done yesterday to help the team achieve the Sprint Goal?
  - What will I do today to help the team achieve the Sprint Goal?
  - Are any impediments that prevent the team achieve the Sprint Goal?
- At the end of each Sprint, a Sprint Review meeting takes place to see the outcome of the Sprint and to run a demo with the new features developed during the Sprint in front of Product Owner and stakeholders.
- The Sprint Retrospective meeting takes place after the Sprint Review and before the next Sprint Planning. The Scrum Master provides a positive and productive climate for this meeting. The Sprint Retrospective meeting is a chance for the Scrum Team to examine itself and create a plan for improvements to be assimilated during the next Sprint.

2.2. Kanban

Kanban word means in the Japanese language "signboard", and it is used in manufacturing as a scheduling system with the purpose to have products delivered in time and to use workers capabilities at their full strength. Kanban has more technical and industrial character, and this method has the origins in the Toyota Production System (TPS) and Lean Manufacturing (also known as Lean Production) (Stoica, 2016).

Kanban is not such an old concept when we are referring to the software development field. The main visual element of Kanban methodology is the Kanban board (Figure 2), where the entire software development project lifecycle is displayed. On the Kanban board are put usually sticky notes that represents Kanban cards and contain small work items (frequently user stories). The board has a couple of columns, and each one represents a workflow phase in the software development process. In order to determine developers first to finish what had been started and only after to pick new work items, the number of work items existing simultaneously in each column is limited. When a work item is finished in one specific phase, it skips to the next column as "to be done" and on the current column-free slot, another work item can be pulled from the previous column (Khan, 2014).

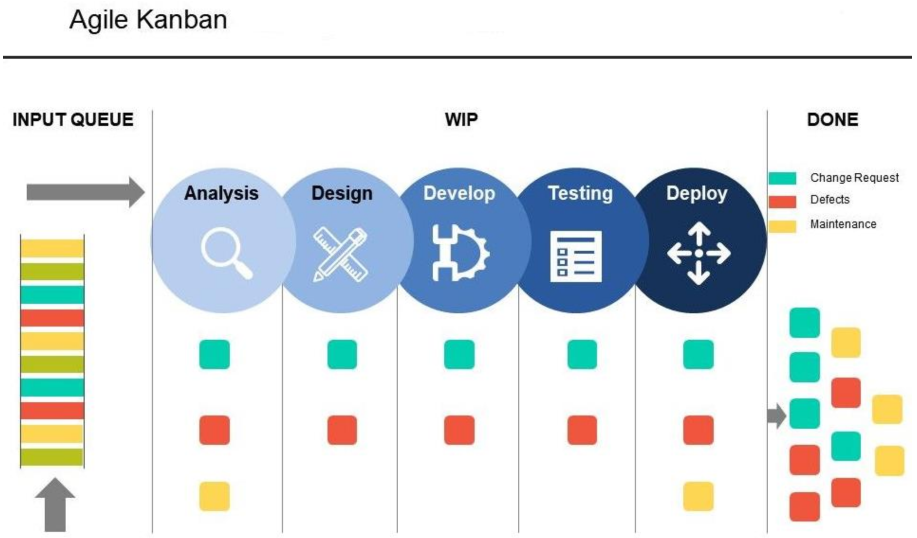

Figure 2. Kanban board (Slideteam, 2019)

Kanban method has three important principles, and these are workflow visualization for the process, limit work in progress (WIP), and measurement lead time.

Process Visualization: It is done by using the Kanban board to draw important workflow phases, and this is offering a visual control mechanism. Visualization helps team members to know exactly the progress of each task and to quickly observe if in some phases are impediments in tasks advancement.

Limiting Work in Progress: Kanban philosophy is based on the idea that it is better not to start work on a new item until the current item work is not finished for that phase. This is why Kanban introduce a maximum limit on the Work in Progress (WIP) items for every phase of the workflow. The “pull" mechanism allows new work item to be pulled only when is availability to be handled properly, and this is an important method to avoid overloading different phases of the workflow.

Measure Lead Time: Lead Time is one of the most relevant metrics in Kanban methodology and measures average time to complete one work item. In Kanban, the approach is to improve the process and to make lead time smaller and more predictable. By improving lead time, the development team can provide better forecast how many work items can be completed within a time frame, and this drives to an improvement of the global process and finally enhances team performance (Khan, 2014).

2.3. Scrumban

Scrumban is a hybrid methodology that resulted from a combination of Scrum and Kanban methods and takes from Scrum the basic features and Kanban the desire for improvement and flexibility.

Contrary to Scrum, Scrumban does not use a clear definition of roles but is still promoting short daily meetings and kaizen events (a team workshop that facilitates process related problems solving) that intend to resolve everyday impediments. Iterations period in Scrumban should not be longer than two weeks but opposite like in Scrum, are allowed long tasks that can be developed on several iterations and this can drive to an unfinished product at the end of each iteration. This is the reason why the Feature Freeze (FF) concept is used in Scrumban and it means that when the iteration finish is coming, the team is focusing on completing tasks already in progress and features that are still unfinished will be removed from the final product in order to provide an incremental release (Brezočnik, 2018).

Scrumban iteration is presented below in Figure 3.

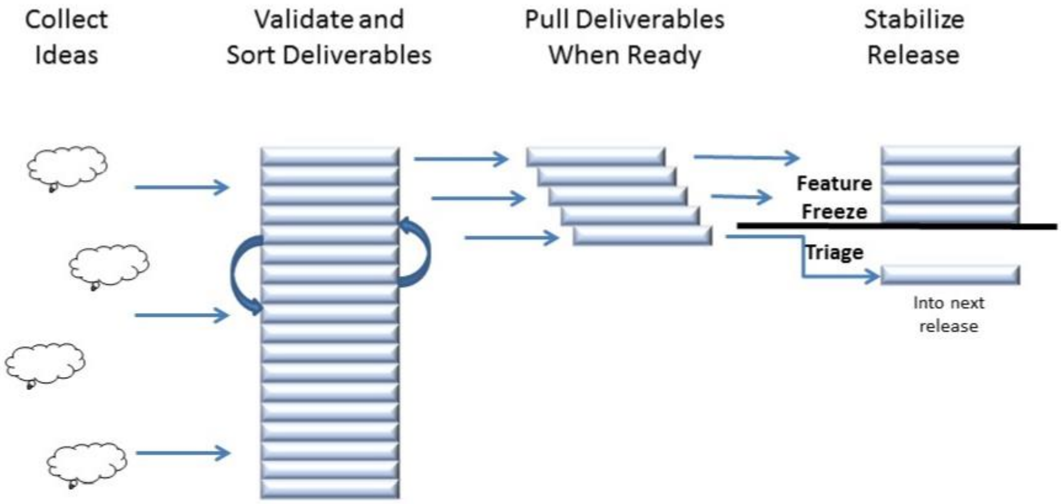

Figure 3. Scrumban iteration (Medium, 2019)

The main elements defining Scrumban are:

Visualize the workflow: Team visualizes different phases for the workflow on board is the most significant instrument borrowed from Kanban.

Limited Items with status Work-In-Progress (WIP): Scrumban is setting a limit for the number of work in progress items that can be in every phase based on team capacity. When the limit is reached within a particular phase of the workflow, better than start working on a new item, it is time to help someone else within the team. Doing this will significantly improve collaboration within the team (Khan, 2014).

Pull Work: Work is pulled from a queue in Scrumban, and an important improvement in Scrumban is that it allows teams to detach the assignation of work from the prioritization of work and this is done by adding a new queue called ready queue in between the backlog and work in progress queue. In this ready queue are items with high priority that are pending from the backlog, and in this queue, no tasks are linked to the team members, and the first who gets available should take one of these tasks (Khan, 2014).

Make Team Rules Explicit: In Scrumban, team rules are made clear so that everybody in the team is allowed to manage flow, coordinate, and self-organize in order to obtain a smoother workflow.

Planning Meetings: Scrumban is using shorter planning meetings with the purpose to update the backlog queue as frequently is required, and this is one of the biggest advantages of Scrumban because it provides the ability to respond to demands faster.

Daily Stand-up, Review, and Retrospectives: In Scrumban, these are meaningful ceremonies and are imported from the Scrum method.

2.4. Scrum/XP hybrid

Scrum/XP hybrid (Figure 4) is a combined method between Scrum methodology and extreme programming (XP) methodology which are quite complementary methods and practices except for the part of planning where these two are overlapping. By combining these two methodologies, the hybrid method takes from Scrum an effective project management framework and adding XP elements, enriched the method with engineering capabilities during phases like planning, designing, developing, testing and integration phases (Mushtaq, 2012).

The two methodologies, which are the building blocks for the new hybrid one, have in common some practices like the work is done in iterations, a product increment is delivered after each iteration, the customer has the wheel to steer what to be built in the next iteration, the development team is self-organized and good collaboration between customer and development team.

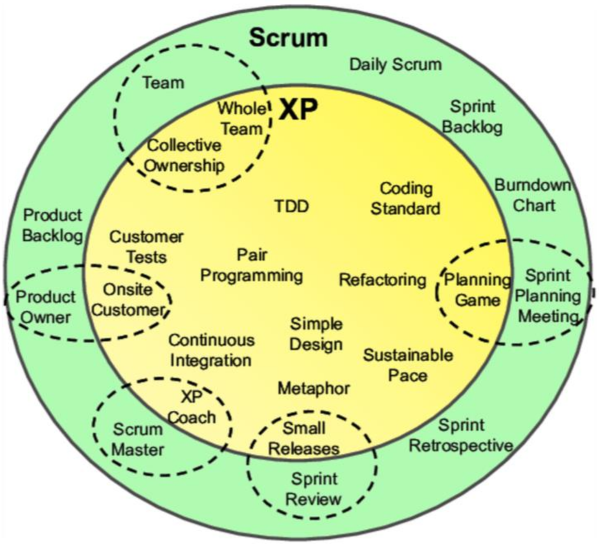

Figure 4. Scrum/XP hybrid (Medium, 2019)

Scrum/ XP hybrid method has the following distinguished elements:

Product owner, Scrum Master, development team -roles taken from Scrum.

Product backlog, Sprint backlog- these artifacts are taken from Scrum.

Sprints with all ceremonies included like Sprint planning meeting, Daily scrum meeting, Review, and Retrospective meetings.

Simple system design - focus exclusively only on what is strictly needed to support currently implemented functionality.

Test first -this means that tests are prepared before the code is written and this thing helps developers to understand better what is expecting from their application and to faster and cleaner write the code (Mar, 2002).

Pair programming -it's a technique imported from XP and consists of two developers work on the same computer, one is writing the code, and the other one is reviewing the code, and the roles are often changed (Mar, 2002).

Refactoring -it's usually more efficient when pair programming is applied and permits incremental code simplification and improvement.

Continuous integration - it means that several times a day merges all developers works, build the code, and run the set of tests.

3. Key elements of the most popular Agile methods

Cumulated, the four methods presented above represent 75% of all Agile methodologies used by the 97%, from the 1492 respondents, who declare they use Agile in the 2017 survey (VersionOne, 2016, 2017, 2018). This fact means that these four methods are currently the most popular ones from the entire wide range of Agile methodologies.

In this paper, the approach is not to compare these four popular methods but rather to identify and highlight the key elements that make these methods so popular and used. Bellow, Table 2 presents exactly these key elements.

Table 2. Key elements of the most popular Agile methods

Agile method	Key elements
Scrum	Product Backlog, Sprint Backlog, Increment
	Product Owner, Scrum Master, Scrum development team
	Sprint with the Sprint goal
	Sprint Planning, Daily stand up, Review and Retrospective meetings
Kanban	Process Visualization
	Limiting Work in Progress (WIP)
	Measure Lead Time
Scrumban	Visualize the workflow
	Limited Items with status Work-In-Progress (WIP)
	Pull Work
	Make Team Rules Explicit
	Sprint Planning, Daily stand up, Review and Retrospective meetings
Scrum/XP hybrid	Product backlog, Sprint backlog
	Product owner, Scrum Master, the development team

	Sprint Planning, Daily stand up, Review and Retrospective meetings
	Simple system design
	Test first
	Pair programming
	Refactoring
	Continuous integration

4. Conclusions

With the key elements for each methodology identified, the next steps in our future work will be to deeply analyze the hybrid (multiple methodologies) model used by 14% of respondents (VersionOne, 2016, 2017, 2018) in order to identify its particularities and the key elements. Having all the above elements, the next natural step will be to propose a new Agile method based on elements of the most successful and popular Agile methodologies existent.

It is clear now that most of the key elements for Scrum, Kanban, XP, hybrid methodologies are complementary rather than competitive and can be used as a foundation for a new Agile method.

Agile is a mindset described by four values, defined by twelve principles and manifested through an unlimited number of methods and practices. In the next article, we will seek the right mix between these key elements already identified and will add new ones to define a new customized Agile method.

Acknowledgment

This work was supported by project PN 19 37 04 01“New solutions for complex problems in current ICT research fields based on modeling and optimization”, funded by the Romanian Core Program of the Ministry of Research and Innovation (MCI), 2019-2022.

BIBLIOGRAPHY

1. Abrahamsson Pekka, Salo Outi, Ronkainen Jussi (2002). Agile *software development methods*. VTT Electronics, 2002.

2. *Agile-Mercurial* - URL: https://agile-mercurial.com/2019/02/06/agile-frameworks-fact-sheet/ (visited on 03/04/2019).

3. Beck Kent, Beedle Mike, van Bennekum Arie, 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*, 2001.

4. Brezočnik Lucija, Majer Črtomir (2018). *Comparison of agile methods: Scrum, Kanban, and Scrumban*. Faculty of Electrical Engineering and Computer Science, University of Maribor, Maribor, Slovenia, July 2018.

5. Gibson Paul (2011). *Agile Methods of Software Development*, 2011. Departament LOR, TSP, Evry, France.

6. *Imgbin* - URL: https://imgbin.com/png/f4Np5UYr/agile-software-development-with-scrum-computer-software-png (visited on 04/09/2019).

7. Khan Zahoor Ahmad (2014). *Scrumban - Adaptive Agile Development Process*. Helsinki Metropolia University of Applied Sciences, May 2014.

8. Mar Kane, Schwaber Ken (2002). Scrum with XP – 2002.

9. *Medium* URL: https://medium.com/agile-outside-the-box/team-working-agreement-for-a-team-transitioning-to-scrumban-8c197190e4c7 (visited on 04/11/2019); https://medium.com/agile-outside-the-box/better-together-xp-and-scrum-c69bf9bffcff (visited on 05/11/2019).

10. Mushtaq Zaigham, Qureshi M. Rizwan Jameel - *Novel Hybrid Model: Integrating Scrum and XP* - I. J. Information Technology and Computer Science.

11. O'Sheedy Daniel Glen (2012). *A study of agile project management methods used for IT implementation projects in small and medium-sized enterprises,* 2012. Southern Cross University.

12. Schwaber Ken, Sutherland Jeff (2017). *The Definitive Guide to Scrum: The Rules of the Game*, November 2017.

13. *Slideteam* - URL: https://www.slideteam.net/agile-kanban-showing-input-wip-limit-analysis-design-and-develop.html (visited on 31/10/2019).

14. Stoica Marian, Ghilic-Micu Bogdan, Mircea Marinela, Uscatu Cristian (2016). *Analyzing Agile Development – from Waterfall Style to Scrumban*. Bucharest University of Economics, Romania - Informatica Economică vol. 20, no. 4/2016.

15. *VersionOne - The 10th,11th,12th annual State of Agile Report, 2016, 2017, 2018* - available at https://agilebb.nl/wp-content/uploads/2018/04/versionone-12th-annual-state-of-agile-report.pdf; http://www.agile247.pl/wp-content/uploads/2017/04/versionone-11th-annual-state-of-agile-report.pdf; http://www.agile247.pl/wp-content/uploads/2016/04/VersionOne-10th-Annual-State-of-Agile-Report.pdf.

16. https://en.wikipedia.org/wiki/Agile_software_development (visited on 03/04/2019).

17. Williams Laurie (2007). A *Survey of Agile Development Methodologies*, 2007.

Flaviu FUIOR a absolvit Facultatea de Finanțe, Asigurări, Bănci și Burse de Valori din cadrul Academiei de Studii Economice București în anul 2003. În 2005 a terminat un master în management la Școala Națională de Studii Politice și Administrative București. În prezent își desfășoară activitatea în cadrul Departamentului CDI Modelare, Simulare, Optimizare la Institutul Național de Cercetare-Dezvoltare în Informatică – ICI București. Domeniile principale de interes sunt: securitate cibernetică, internetul obiectelor (IoT), testare automată, protecția infrastructurilor critice, metodologii pentru managementul proiectelor.

Flaviu FUIOR graduated the Faculty of Finance, Insurance, Banking and Stock Exchanges of the Bucharest Academy of Economic Studies in 2003. In 2005 he completed a Master in Management at the National School of Political and Administrative Studies Bucharest. He is currently working in the Department CDI Modeling, Simulation, Optimization at the National Institute for Research and Development in Informatics – ICI Bucharest. His main areas of interest are cybersecurity, Internet of Things, automated testing, critical infrastructure protection, project management methodologies.