To Be or Not to Be Agile

Abstract - In this paper we take a look at two Traditional SDLC models and compare them with two Agile frameworks. We summarize information and define the advantages and disadvantages for Traditional SDLC and Agile. We check what developer teams expect to benefit from adapting Agile. In the end we make a conclusion if Agile brings more benefits than Traditional methods and will answer the question if teams should adapt Agile. We describe the obstacles which the development team might encounter before adapting Agile.

Introduction.

Many companies now days are using the Agile approach during their product development. Agile adoption in software development increased from 37% in 2020 to 86% in 2021 (Bill Holz, 2021).

Based on research which takes place in 2020 (Djurovic, 2020):

- Agile adoption has helped 98% of companies
- Agile failure rate is 8%
- 60% of companies experienced growth in profits after adopting Agile approach

According to Standish Group Chaos Study in 2020 (Group, 2020):

	Agile	Waterfall
Successful	42%	13%
Challenged	47%	59%
Failed	11%	28%

Table 1. Standish Group Chaos Study research on Agile vs Waterfall 2020 (Group, 2020)..

These facts show that Agile approach is not just popular, but brings more benefits to development.

Traditional SDLC.

Traditional SDLC (Software Development Life Cycle) might be based on predictive or more adaptive approaches. Model based on predictive approach is fully depending on the requirements analysis and careful planning in the beginning of the project development, as only the final version of the product will be available to users. Adaptive approach model is less dependent on requirements analysis then predictive model, and more ready for change implementation. In adaptive model early release is possible, and feedback might be received on early release.

Traditional SDLC is mostly associated with the oldest software development methodology - Waterfall Model, but it is worth to mention the existence of other old methodologies like Spiral or Prototype.

Waterfall Model.

Waterfall model was introduced in 1970 by Winston W Royce and became a popular software development model for decades (Royce, 1970). It helps organize workflow of software development through specific phases. Waterfall model is a linear predictive model, where transition from one development phase to another is done, when current development phase is complete.

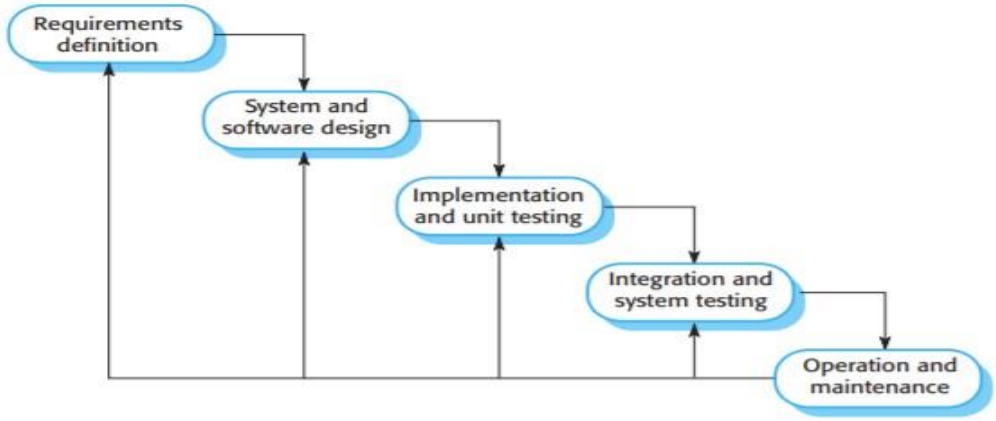

Figure 1. Waterfall Model phases scheme (Sommerville, 2015).

Waterfall model is easy to understand ant it has its own advantages and disadvantages:

Advantages	Disadvantages
Each phase should be completed and approved before moving further	Real projects rarely developed without requirements being changed
Tasks assigned to specialized teams	Changes are difficult
Identifying system requirements long before programming begins	Changes are costly and delivery date might move later
Necessary documentation produced for each phase	Long time between system proposal and system delivery

Table 2. Advantages and disadvantages of Waterfall Model (Karl Wiegers, 2013) (Sommerville, 2015).

Based on Table 1 we can make the conclusion that Waterfall model is more suitable for small projects, as large projects based on Waterfall model will often be delivered late. Development without user's feedback, might lead to lack of necessary features and failure to meet user's expectations (Karl Wiegers, 2013).

Waterfall model success mostly depends on stakeholder's clear view of final product and requirements analysis in Requirement definition phase in the beginning of the project.

In reality stakeholders often change requirements for large projects, as business needs will change with time. Those changes are expensive, unpredictable, and might need full project refactoring, meaning that release date will move.

Spiral Model

Spiral model combines the idea of Waterfall and Prototype models, but in iterative development. It is adaptive, more modern SDLC, where cycles will go over and over until the project is finished. Spiral model allows incremental early releases at each iteration (Barry Boehm, 2014).

Spiral model consists of four phases:

- Determine objectives
- Identify and resolve risks
- Development
- Plan the next iteration

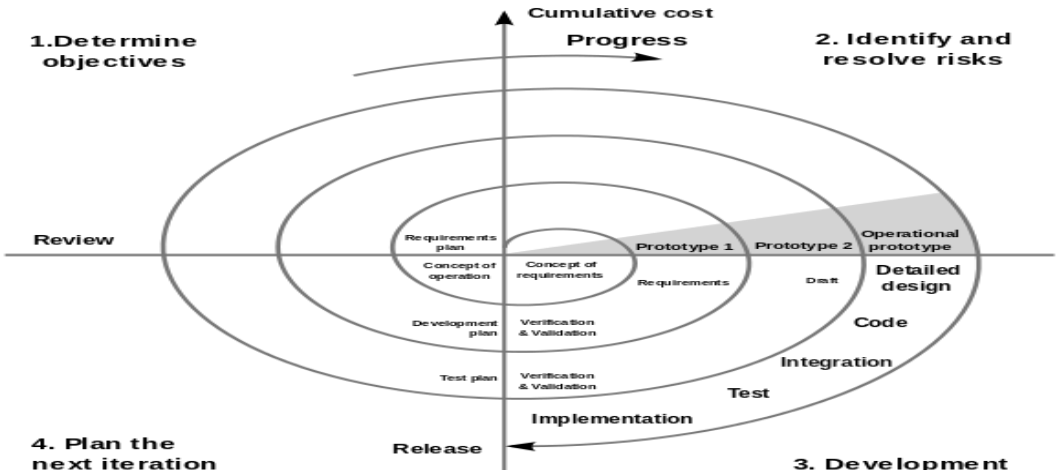

Figure 2. Spiral Model development scheme (Wikipedia).

Advantages	Disadvantages
Changes in requirements are possible	Final release date not known on early stages
Feedback is received, as early releases possible	It is a complex process to manage
Risky parts might be developed early, to improve risk management	Requires excessive documentation
Allow extensive use of prototypes	High cost of development
Review of phase and planning next phase	No suitable for small or low risk projects

Table 3. Advantages and disadvantages of Spiral Model (Barry Boehm, 2014).

Based on Table 2 we can make conclusions that the Spiral model is minimizing risks of developing the final product with lack of necessary features, as user's feedback might be available from early releases. Current model is suitable only for larger and medium to high-risk projects. Spiral model development might be expensive, as process of risk analysis is complex.

Conclusion on traditional methodology's

Taking Waterfall and Spiral models as an example, we can make the conclusion that traditional methodologies have their own advantages and disadvantages.

Advantages	Disadvantages
Good documentation	Might be slow in development
Not much training needed, as work is done by specialized teams	No collaborative teamwork, as work is done by specialized teams
Clearly defined objectives	Might be expensive
Some models specially designed for high-risk projects	In some models changes are not easy

Has predictive and adaptive approach models	In some models testing performed in a late stage, with possibility that product release will be postponed until bugs are fixed

Table 4. Advantages and disadvantages of traditional methodologies based on research.

Agile

Agile approach does not require detailed planning. Teams are easy adapting to changes if new requirements appear. Agile approach is change friendly and has the opportunity to make modifications during development (Karl Wiegers, 2013).

Agile Manifesto

On the 11th of February, 2001, a group of industry experts met at "The Lodge at Snowbird" ski resort, to discuss a better way of building software. That group call themselves "Agile Alliance". They have been working on that problem together over the past months, and during their meeting they introduced the "The Manifesto of Agile Alliance" (Martin, 2014).

Agile Manifesto consists of a set of four values and twelve principles. Which helps the developer teams to reach their goal, using simple technique.

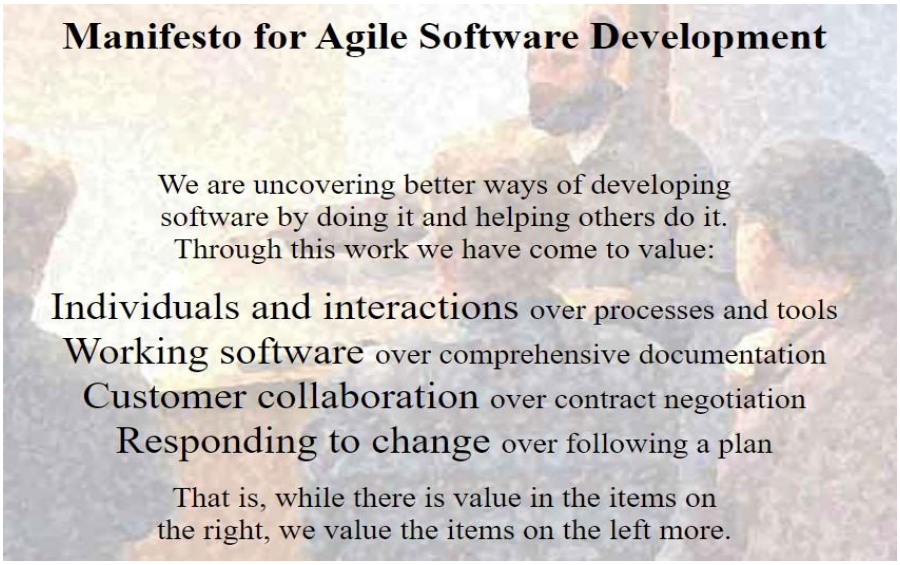

Figure 3. Four Values of Agile Manifesto (Kent Beck, 2001).

Idea of Agile Software Development

Idea of Agile software development is to break down complex problems into a small manageable goals. Agile team is working on these goals, while adding new, based on customer feedback or stakeholder changed requirements.

Benefits of Agile development (Ken Schwaber, 2020; Karl Wiegers, 2013):

- At any stage the ability to adapt and change depending on feedback and supply only relevant product to the market
- Very flexible as quickly adapts to changes
- Quick product launch

- Enables fast decision making through a flexible organizational structure and simple communication
- Improved team communication through daily meetings

What is agile framework

Agile framework is a specific adaptive software development approach based on the agile philosophy articulated in Agile Manifesto. Some of this framework ideas existed even before Agile Manifesto was introduced (Sommerville, 2015).

There are many frameworks based on Agile philosophy. Most popular in 2021 are (Bill Holz, 2021):

- Scrum (66%)
- ScrumBan (9%)
- Scrum/XP Hybrid (6%)
- Kanban (6%)

Various organizations use many popular frameworks, depending on the project needs. Many factors may influence the identification of suitable framework for the project:

- Company size
- Available resources
- Team structure
- Needs of stakeholders
- Previous experience

Scrum

At the moment Scrum is the most popular framework to implement Agile (Bill Holz, 2021). Scrum is iterative and incremental development. Scrum was introduced in 1995 by Jeff Sutherland and Ken Schwaber.

Scrum has three important roles in the process:

Role	Definition
Product Owner	Individual (or a group) who identifies requirements and prioritizes requirements after discussing with the development team. Creates and reviews product backlog to ensure that project is continuously developing.
Scrum Master	Individual who ensures that Scrum process is done the right way in organization. Guides the team with understanding of Scrum. Plays an important role in all Scrum ceremonies.
Development team	Self-organized team of maximum seven developers. Responsible for documentation and software development

Table 5. Roles and responsibilities in Scrum (Sommerville, 2015).

Using Scrum development is a process of going through a series of short-length iterations until product is done. Those iterations called sprints, and early software releases are common when using Scrum. Each sprint has Sprint Backlog, with tasks needed to be done in current sprint. Duration of a sprint may vary, but usually one sprint duration is 1-4 weeks. Once sprint begins, adding new tasks in the current sprint is not allowed. The important part of Scrum is that the team is motivated, as they see achieved progress after each sprint.

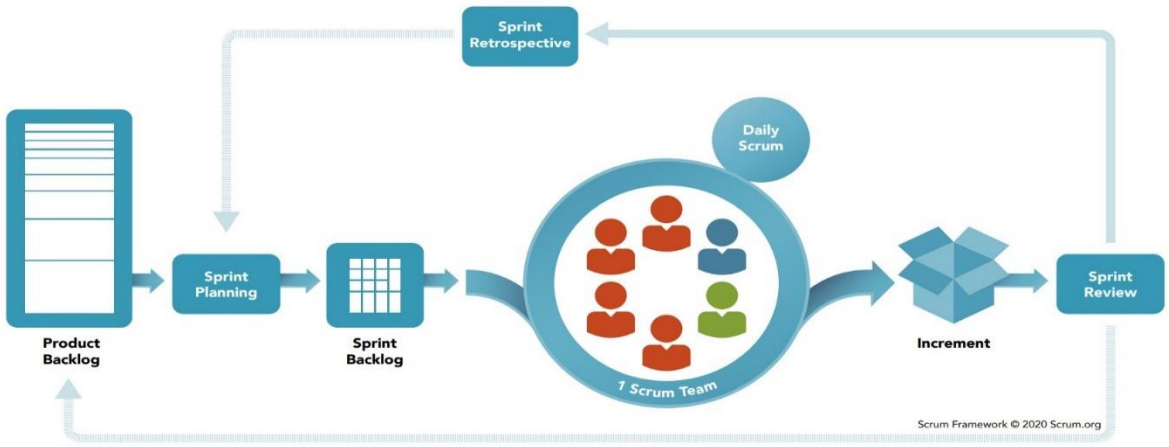

Figure 4. Review of processes in one Sprint cycle (Trademark, 2021).

Each sprint has four ceremonies (Ken Schwaber, 2020):

- **Sprint planning** – team meeting, where team identifies necessary current sprint tasks (user stories) and time needed for sprint. Tasks being moved from Product Backlog into current Sprint Backlog.
- **Daily Scrum** - limited to 15 minutes stand up meetings, where development team and Scrum Master discuss progress and problems of the sprint. Scrum team uses Scrum Board to have a visual clear view of the workflow. Scrum board usually consist of four columns: To Do, Build, Test, Done. Tasks being moved from left to right, based on progress made on current task.
- **Sprint review** – team meeting at the end of the sprint where current sprint is being reviewed with the stakeholder's participation.
- **Sprint retrospective** – team meeting where they discuss what the team did right or wrong, and what the team might do better in the next sprint. By doing that, team improves and becomes more efficient in the next sprints.

Advantages	Disadvantages
Quick and efficient development	Project might fail if individuals aren't cooperative
Projects divided in small, easy manageable sprints	Hard to adopt in large teams
Clear visibility on tasks for the team during Scrum meetings	Hard to implement for large projects, more efficient for small and medium size projects
Possible feedback after each sprint	Hard for new members to join in the middle of the project
Fast moving development as team collaborates	Team's success depends on the Scrum Master and team's experience
Good for teams to try Agile	Not suitable for development where immediate actions need to be performed on change

Table 6. Advantages and disadvantages of Scrum (Sommerville, 2015) (Ken Schwaber, 2020).

Cost of final project can be calculated by multiplying fixed cost per sprint by amount of sprints.

Kanban

Idea of Kanban was introduced by Toyota engineer Taiichi Ohno in 1940. In 2004 David J. Anderson was the first to apply Kanban into software development.

Kanban's main focus is to maintain a continuous task flow and continuous delivery. The team will never have more work than it can process. This is achieved using two main principles of Kanban (Anderson, 2010):

- **Visualise your work** – process of collecting the required work to be done and documenting it on cards, place them on a Kanban board. The board is divided in columns, representing progress of tasks. Kanban board contains columns: To Do, In Progress, Done. Teams can add more columns to better visualise process. Task is being moved from left to right, representing progress made on current task. This is how team sees the progress on work being done.
- **Limiting Work-In-Progress** – limit for number of tasks can be processed in one step (Kanban board column). If one step WIP already reached the limit and another task is ready to be moved in, team stops the work and together fix "bottleneck". This helps the team focus on current tasks before staring new, and maintain continuous flow.

Kanban development is incremental and allows development in one large development cycle. Progress in Kanban is defined in Cycle time – amount of time taken for one task, from start to finish.

Differences between Scrum and Kanban

Main differences between Scrum and Kanban:

	Scrum	**Kanban**
Cadence	Duration of sprint	Continuous flow
Releases	At the end of the sprint. Product owner approval needed.	Continuous delivery or by the team decision
Roles	Product owner, Scrum Master, development team	No specific roles. Some teams might have agile coach
Key metrics	Velocity - measure which determines the amount of work team can do in one sprint	Cycle time - measure which determines time spent as Work-In-Progress (WIP).
Change philosophy	Changes is not acceptable until the end of sprint.	Changes can happen anytime, and new task can be added in development if it does not create a bottleneck

Table 7. Main differences between Scrum and Kanban (Atlassian, 2021; Ken Schwaber, 2020).

Scrum is more structured, allows accountability, forecasting emphasizes teamwork, iterative progress toward well-defined goal (Rob Cole, 2015).

Kanban is a visual system for managing work which moves through a continuous process. Kanban is more flexible than Scrum (Rob Cole, 2015).

Conclusion on Agile development

Taking Scrum and Kanban frameworks as an example we can make the conclusion that Agile development has its own advantages and disadvantages.

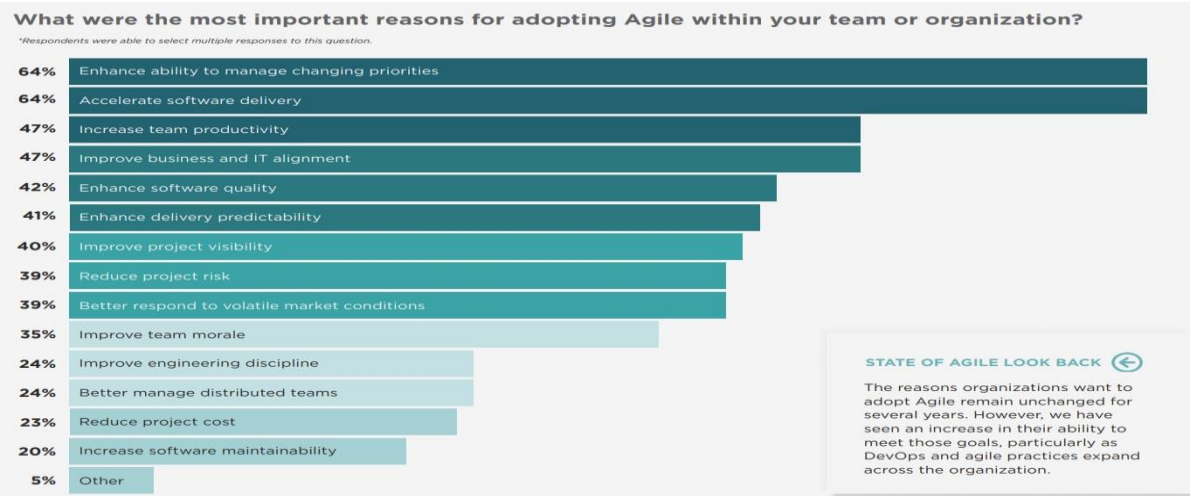

Figure 5. Social survey results on an Agile adoption reason (Bill Holz, 2021).

Advantages	Disadvantages
Very adaptive and flexible	Possibility of scope creep
Quality improvement, as developers know that quality is only their responsibility	Low amount of documentation, possible wrong understanding of second Agile Manifesto value
Reduce cost of development, as no risk management is needed and teams are more productive when working on small tasks	Possibility to not see the clear end of the project
Accelerates delivery. Quick releases, with necessary customer feedback	Possible team conflict might lead to failure
Space for experimentation and test ideas	Requires Agile training, especially for individuals who are responsible for agility in the company

Table 8. Advantages and disadvantages of Agile development based on research.

Fake Agile

It is worth mentioning, that practising Agile in a team doesn't mean that your team is Agile.

Short description for Scrum is "Scrum is lightweight framework, which is easy to understand, but hard to master". In Ken Schwarber's book "Agile project management with Scrum", he analyses situations where team uses wrong understanding of Scrum philosophy in practice, making them not Agile (Schwaber, 2004).

Factors Agile popularity increase

We see significant increase in Agile adoption within software development teams in the last year (Bill Holz, 2021).

By adopting Agile, software development teams want to be more prepared for changes in projects, accelerate delivery and increase productivity.

It is worth noticing that according to one of co-creators of Scrum Jeff Sutherland, that 47% of Agile transformations fail because of bad Agile philosophy implementation (Sutherland, 2020).

Conclusion.

During research we see that Agile and Traditional methodologies have their own advantages and disadvantages and there is no one-size-fits-all model in software development.

From our research we can make the conclusion that Agile is a very good choice for startup projects, as development is fast and flexible and the product will be faster presented to the users.

Small to medium size projects are ideal for Agile implementation. Large projects are hard to implement with Agile because of lack of important documentation, complexity and possible team burnout, but still possible in an Agile experienced team.

Based on research we can agree that Agile brings more valuable advantages like flexibility, accelerated development of process, ability to implement user expectations with less expenses, compared to Traditional methods. Those facts make Agile more interesting to developer teams.

If we ask the question – To be or not to be Agile? It all depends on the team, project size and complexity, stakeholders. Agile adoption shouldn't be based just on Agile popularity, but based on an important fact, if Agile development will bring more benefit into the current project than Traditional models.

Before adapting Agile development team needs to be sure that all current factors are fulfilled:

- stakeholders agree with the fact that less documentation will be produced during development
- team has clear understanding of how to implement Agile philosophy into practice. Ideally team needs to be Agile certified
- understanding which Agile framework benefits the project development more
- does the development require Risk Management
- size and complexity of the project
- recommended size of the team is less than nine individuals

References

Anderson, D. J., 2010. *Kanban. Succesful Evolutionary Change for Your Technology Business..* First Edition ed. s.l.:Blue Hole Press.

Atlassian, 2021. *Kanban.* [Online]
Available at: https://www.atlassian.com/agile/kanban

Barry Boehm, J. A. L. S. K. R. T., 2014. *The Incremental Commintment Spiral Model.* s.l.:Pearson Education.

Bill Holz, D. B. P. H. S. K. B. D. K. M., 2021. *15th State of Agile Report,* s.l.: Gartner Inc.

Djurovic, A., 2020. *goremotely.net.* [Online]
Available at: https://goremotely.net/blog/agile-adoption/

Group, S., 2020. *Chaos report ,* s.l.: s.n.

Karl Wiegers, J. B., 2013. *Software Requirements.* Third Edition. ed. s.l.:Microsoft Press.

Ken Schwaber, J. S., 2020. SCRUM guide.

Kent Beck, M. B. A. v. B. A. C. W. C. M. F. J. G. J. H. K. S. J. S. D. T., 2001. *Agile Manifesto.* [Online]
Available at: https://agilemanifesto.org/

Martin, R. C., 2014. *Agile Software Development Principles, Patterns, and Practices.* First Edition ed. s.l.:Pearson.

Rob Cole, E. S., 2015. *Brilliant Agile Project Management.* First Edition ed. s.l.:Pearson.

Royce, W., 1970. *Managing the development of large software systems: Concepts and techniques..* s.l.:IEEE Computer Society Press.

Schwaber, K., 2004. *Agile project management with Scrum.* First Edition ed. s.l.:s.n.

Sommerville, I., 2015. *Software Engineering..* Tenth Edition. ed. s.l.:Pearson..

Sutherland, J., 2020. *Why 47% of Agile Transformations Fail,* s.l.: s.n.

Trademark, S., 2021. *scrum.org.* [Online]
Available at: https://www.scrum.org/resources/scrum-framework-poster