DOI: 10.46541/978-86-7233-406-7_260


27th International Scientific Conference
Strategic Management
and Decision Support Systems
in Strategic Management
SM2022
Subotica (Serbia), 20th May, 2022

**Anđelka Čarapić**
University of Belgrade, Faculty of Organisational Sciences
Belgrade, Serbia
andjelkacarapic6@gmail.com

**Mladen Čudanov**
University of Belgrade, Faculty of Organisational Sciences
Belgrade, Serbia
mladen.cudanov@fon.bg.ac.rs

**Ondrej Jaško**
University of Belgrade, Faculty of Organisational Sciences
Belgrade, Serbia
ondrej.jasko@fon.bg.ac.rs

# COMPARATIVE ANALYSIS OF WATERFALL AND AGILE APPROACH TO ORGANISATION IN THE STAR MODEL CONTEXT

**Abstract:** This article presents a comparative analysis of agile in relation to traditional approaches in the organisation of software systems development, shows the differences through the elements of the organisational system (strategy, structure, processes, values, human resources, reward system, etc.) using the STAR organisation model. Further, the goal is to present theoretical views in the context of a project that aims to establish an identity and data access management system. Our empirical insights come from the development team task management in the Jira software solution for supporting software development. The result of the research is presented in the form of a comparative analysis based on the search for optimal software development method approach during an identity management project. This paper can contribute to deciding in which situations teams can consider to apply Waterfall, and in which agile approaches for the development of software systems.
**Keywords:** Agile, Waterfall, STAR model, organisation, software development

## 1. INTRODUCTION

This paper uses the STAR model Kates and Galbraith (2010) proposed to assess the pros and cons of two different approaches (Waterfall and agile) in the software development project of the identity and data access management system. Software development is essentially a complex activity that requires the collective contribution of several individuals and requires significant coordination and project management (Maruping, 2009). Through creating ideas and decisions about all the functionalities that software will include, focus attention is on the possibility of changing requirements. Agility is about the willingness to re-examine the project and the software that is being implemented and continuously change it following the customer's requirements, where the focus is on the value he received. The development team is expected to be able to adapt and respond to the changes quickly.
Agile processes impact: software project management, software implementation management and many other aspects. The goal of agile processes is to make the customer satisfied. Sharma et al. (2012) argue that agile software processes support iterative and incremental development, where requirements vary according to customer needs. As a result, the software is delivered at shorter intervals, not after a few months, as with the Waterfall model (McCormick, 2012).
The article compares the agile approach to software development with other, traditional approaches and shows the differences between these two approaches. The advantages and disadvantages of agile processes over traditional ones are also discussed in this article, where the focus is on the advantages of the agile approach.The second chapter provides an overview of the agile approach focusing on Scrum and Extreme programming (XP), Crystal and Lean development methodologies. The third section describes the Waterfall approaches in detail, and the fourth section provides a comparative analysis of the Scrum and Waterfall approaches in terms of process, organisational structure, organisational culture, and staffing.

## 2. DIFFERENT APPROACHES TO SOFTWARE DEVELOPMENT

The beginning of the agile manifesto in 2001 brings the focus and top priority to the clients satisfaction, which is achieved by constant and continuous work on the project. Each part of the project is delivered as applicable and is

delivered for an agreed period. We are talking about several weeks or several months, where the shorter time intervals are preferred.
Applicable software is a fundamental measure of progress, according to Beck et al. (2001). After visual displaying and testing the applicable software by the client, he can get ideas on how he wants to modify the software. According to Beck et al (2001), the most efficient and productive method of communication is face-to-face contact. The agile manifesto speaks about the willingness of the entire team to accept changes in client requirements, both at the beginning when changes are more frequent also at the later stages of software development (Beck et al., 2001). An agile process is an iterative approach where customer satisfaction comes first, as the customer is directly involved in software evaluation (Boehm & Turner, 2003; Sharma et al., 2012).
Highsmith and Cockburn (2001) said that the innovation in agile methods is their recognition of people as the primary drivers of project success. This, together with an intense focus on efficiency and manageability, gives a new combination of values and principles (Abrahamsson, 2017). The documentation is reduced to an appropriate level. It still exists, but it is maintained and updated following the course of work on the project.
Software development methodologies support fast development with better customer experience for dynamical requirements, using iterative and incremental development techniques (Boehm & Turner, 2005; Kulkarni et al., 2011). Williams (2007) explains that each iteration in an agile approach is a stand-alone, small project with activities that include requirements analysis, design, implementation, and testing. The clients states their requests for the next increment based on observations of the current release, instead of guessing at the beginning of the project (Williams, 2007; Van Casteren, 2017).
Agile approach in software development is highly results-oriented, which is why teams decide to use it. Software development is becoming increasingly complex, with new topics like AI (Devedzic 2021), Internet of Things and composable applications (Ruschby 2016; Chan 2021) or self integrating systems (Burger er al, 2020). There are many methodologies for system development, it means that for each system there may be a different software development methodology (Ericson et al., 2005). This means that choosing an approach to software development can be a challenging task that requires knowledge of methodological approaches. We will descibe only the most utilised methodologies: Scrum, Extreme programming, Cockburn's Crystal and Lean development.

## 2.1. Scrum

Scrum is a framework structured to help teams work together and apply an agile way of thinking to get the job done. It's based on communication, transparency and commitment to continuous improvement. Scrum term originally comes from the strategy of rugby game, where it means returning the ball to the game, through teamwork (Schwaber & Beedle, 2002; Abrahamsson, 2017). Encouraging the teams to learn through experience, self-organise in the process of problem solving, and continuously improve by thinking about their victories/losses connect the
Scrum is an agile methodology that manages software development in specific and short iterations called sprints. Sprint includes all software development lifecycle model phases, such as planning, design, implementation, testing, etc (Matharu et al., 2015).
People are the main link in the agile approach, and each person must contribute to the best software solution with their skills and experience. The multifunctional team includes a combination of developers, software architects, software analysts and quality assurance experts (Matharu et al., 2015). The Scrum team has more than 5 and less than 9 engineers (Schwaber & Beedle, 2002). If more people are available, more teams need to be formed (Abrahamsson, 2017). Team members are focused on how they need to function in order for software to evolve in an ever-changing environment.
Scrum framework offers great flexibility in dealing with changing constraints, whether financial or technological, where its key to success is focusing on the highest priority tasks (Andrei et al., 2019). The development process is complex and unpredictable, intending to deliver applicable software after each iteration.
The practices on which these agile methodologies are based recognise that changes in requirements are inescapable and they accordingly try to adapt on changing requirements as efficiently as possible (Maruping et al, 2009). A predetermined duration of an iteration is used in software development, and it's important for the development team because they need to complete all of the assigned tasks during that time.

## 2.2. Extreme programming (XP)

The use of the Extreme programming method, better known as XP, has been widely recognised as a starting point for various agile approaches to software development (Abrahamsson, 2017). It facilitates iterative and planned software development for small teams of developers to achieve higher software quality and increase productivity (Matharu et al., 2015). XP aims to enable successful software development despite constant changes in software requirements (Flora & Chande, 2014). Extreme programming is characterised by a high level of interaction with clients during the software development process and rapidly evolving requirements.
The founders aimed to develop a methodology suitable for object-oriented projects using teams of a dozen or fewer developers in one location (Abrahamsson, 2017).
XP is coding what a client wants and testing written code to ensure that previous steps in the development process have achieved what clients and developers intended (Erickson et al., 2005). The idea behind XP is that once the need for

specific functions arises and the user sees it, he can inform the development team. The developers don't worry about the missing features until the client does that. In the XP methodology, pre-requirements don't need to be specified.
The methodology is based on five core values: communication, simplicity, feedback, courage, and respect (Williams, 2007). Communication implies a culture of oral communication and aims to encourage interaction. They apply the design of the simplest product that meets customers' needs, which refers to simplicity. The development team receives feedback from clients at the end of each iteration and external release. The development team has to be brave because they need to resist pressur in certain situations. Regarding respect, team members need to show mutual care in order to take care of the project and pursue the same goals (Williams, 2007).

## 2.3. Crystal

In 2001, Alistair Cockburn, one of the creators of the Agile manifesto (Highsmith &Cockburn 2001) who developed the Crystal methodology, which gives precedence to people involved than to the process. Crystal is characterised by frequent software delivery, a technical environment where automated tests are applied, close communication, focus on tasks, frequent integration, etc. This methodology doesn't require the use of certain techniques or special tools. Moreover, the increments are adaptable in length, for large projects they can last even 4 months (Flora & Chande, 2014).
Crystal is a family of methodologies used on different types of projects, different complexities and sizes. Mnkandla & Dwolatzky, B. (2004) argued that Crystal can be applied to teams off any size. Larger projects, which require better coordination and communication, are mapped to darker colors (clear, yellow, orange and red). When it's necessary to maintain better communication and coordinate with more people, darker colors are used.

## 2.4. Lean development

Lean development focuses on aspects of project management and doesn't go into technical practice (Flora & Chande, 2014). Due to that it is often integrated with other agile methodologies. Aimed at optimisation, this approach should optimise the whole organisation and product for the whole time in order to avoid suboptimisation (Alahyari et all. 2019).Lean development has a focus on creating value for the customer, eliminating waste, continuous improvement, optimising value flows and empowering people (Ebert et al., 2012). It was used for the first time in production, with clear goals to optimise workflows, and above all to maintain the market and customer needs as the primary goal.
Specifications related to team size are not specified, as Lean development is considered as a software development management philosophy, rather than a methodology (Flora & Chande, 2014).
Some of the principles Lean is based on are:

- respecting all the people in the team,
- assembling multifunctional teams,
- reducing the time required to identify business problems and faster system delivery,
- eliminating everything that doesn't add value to the customer,
- applying short iterative development cycles and so on (Flora & Chande, 2014).

The principles have similarities with the agile manifesto, and philosophy of Lean methodology is based on the foundation of the agile approach (Cawley et al., 2010).

# 3. WATERFALL APPROACH

Waterfall approach can describe a set of traditional software engineering methodologies (Mirza & Datta). It stands out as an nonflexible approach where tight control is maintained all through the project. This is reflected in the extensive written documentation required, formal reviews, signatures and approvals from clients. With the Waterfall approach, the emphasis is on planning, timelines, dates, budgets, and implementing the entire system at once (Flora & Chande, 2014). It focuses on stability and greater security for predefined sets of requirements.
Traditional software development methods depend on a set of predefined processes and documentation that describes software development progress and leads to further development (Nikiforova et al., 2009; Leau et al., 2012). It's a sequential approach, so there are stages that software goes through during development, which in the traditional approach are followed in identical order, and approvals appear at the end of one and before the beginning of the next phase. When the phase is over, there are no revisions and changes in the results subsequently. Also, no task belonging to a later phase can be started until the results of the previous phases have been completed and approved. Weisert (2003) said that the incapability of water to flow uphill suggests the metaphor of a waterfall. Once a certain point is passed, there is no going backwards because costs increase beyond acceptable, often 10 times more with each phase (Anitha & Prabhu 2012)
However, Weisert (2003) also argues that the point of any formal life cycle is the phased limited commitment that is essential to the success of large projects. When the development team knows exactly what functionalities should be enabled, that they won't change, and during which time they should be completed, that's the motive to complete the

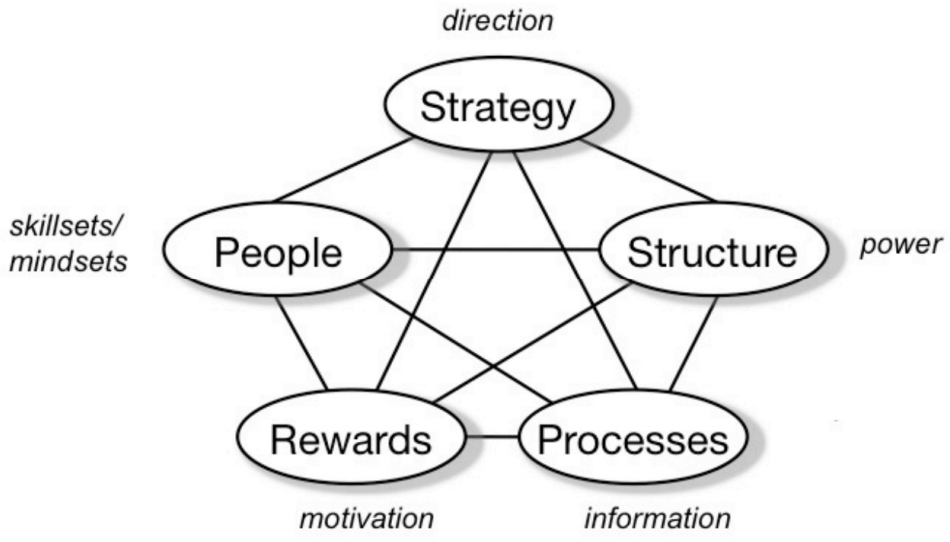
work within the given deadline. Precise and fixed definition of the project requirements before development even starts is imperative for the success of this approach (Leau et al., 2012), and it facilitates project costing, scheduling, and resource allocation accordingly. The client's advantages by applying this approach and the reasons he decides on the traditional approach are reflected in his confidence that the project will not take more resources than planned. In this case, resources primarily include finances, but also time, people, equipment and more. The goal is to be productive, to avoid unproductive hours and unwanted guidelines. The control over the project, that the client has when approving one complete phase at a time, is also the reason for applying this approach. Client can make a rational choice estimating costs/benefits of the future project if faced with well-grounded life cycle waterfall methodology, which is a rare case in practice.
The waterfall approach would be unproductive when project requirements are not fully understood or defined, which assumes that they will change during the project. If the development team starts designing and implementing uncomplete requirements, in most cases there will be an increase in costs that are not in line with the original plan. Most problems would occur in the later stages of software development.

## 3.1. The phases of a Waterfall Model

The software development life cycle is a process that includes various stages, from preliminary development analysis to post-development testing and software evaluation (Leau et al., 2012).
The traditional model implies a sequential development approach, where development proceeds through the phases of collecting requirements, analysis, design, coding, testing, and maintenance (Flora & Chande, 2014). The phases are followed in the listed order and it is necessary to approve and verify one phase to move on to the next.
First of all, project requirements are collected and defined, and then, based on the quantity and complexity, the duration of the project is defined by adding estimation for each of the development phases and anticipating problems. Afterwards, the team defines the technical infrastructure using diagrams/models in the design and architectural planning phase (Leau et al., 2012). When these architectural models and diagrams are accepted, the coding phase begins. Software development is often divided into smaller tasks given to different teams according to their competences. Near the end of that phase, the customer is included as the project slides into the testing and feedback cycle. When the customer declares satsfaction with the final result, the project becomes delivered (Leau et al., 2012).
The analysis and coding phases give the most direct value to the final product (Van Casteren, 2017), but it's important to dedicate enough time to each phase because the next one depends on the performance in the previous phase. The results of each phase are documented.
In the last phase, the client is provided with a certain type of software support. Eventual additional improvements to the existing software require that the team must restart activities from the first phase (McCormick, 2012).

# 4. COMPARISON OF AGILE AND WATERFALL METHDS ON GALBRAITH'S STAR MODEL

Strategy, structure, processes, rewards and people, as the five key areas that need to be linked and harmonised in order to successfully shape decisions and behaviors, have been analysed. These five elements form the basis of the *Star Model* (Galbraith, 2011), shown in graph below.




**Picture 1:** Star model

The following is an overview of all five categories using agile and traditional approaches.

## 4.1. Strategy

Vision, mission, core values, and the main set of proposed goals define organisation's strategy as the element of this model. It determines the direction and talks about the sources of competitive advantage (Galbraith, 2011), which in this case are the advantages of one approach over another.

**Table 1:** Comparative analysis based on strategy

| Agile approach strategy | Traditional approach strategy |
|---|---|
| ▪ Based on the principles of being highly adaptive, continously improving, open relation with the client to ensure good communication and feedback;<br>▪ It is argued that "applicable software is a key measure of progress";<br>▪ It is based on an iterative and incremental model of software development, where software is delivered at regular and equal intervals, most often at time intervals from one to six weeks;<br>▪ The most important are technical expertise, fewer shortcomings, continuous testing, integration and collaborative approach;<br>▪ There is direct communication between the development team and the client (Mora et all, 2020);<br>▪ Finding a balance of response to the changing project requirements and ratio of software quality / time for delivering the project; | ▪ The emphasis is on planning, timelines, dates, budgets, and implementing the entire system at once;<br>▪ The systems are predictable and built with extensive planning where the requirements are known at the very beginning of the project and will not change during development (Cho, 2020);<br>▪ It focuses on stability and greater security for predefined sets of requirements (Kulkarni et al, 2011). It is a sequential approach, so there are stages that software goes through during development, which are respected in identical order and approvals are applied at the end of one and before the beginning of the next phase;<br>▪ Comprehensive documentation is mandatory;<br>▪ The goal is to be productive, to avoid unproductive hours and unwanted guidelines, and deliver software with all agreed functionalities on time; |

## 4.2. Structure

The organisation's structure is the defined by Henry Mintzberg (1993) as the sum of ways organisation divides its work and coordinates it towards the mutual goal (Jaško et al 2017) Galbraith (2011) claims that structural policies are classified into specialisation, form, distribution of power, and departmentalisation.
Specialisation refers to the type and number of jobs. The form refers to the number of people who are part of the department at each level of the structure (Khan at al 2016). The distribution of power balances between the centralisation or decentralisation in decision making of the organisational members. Departmentalisation is the process of joining jobs into department of the lower level, or joining the departments of the lower level into those of the higher level e.g. sectors, divisions for each level of the organisational structure (Jaško et al, 2013). They can be formed on the basis of various criteria, such as functions, geography, customers, products etc.

**Table 1**: Comparative analysis based on structure

| Structure in agile approach | Structure in traditional approach |
|---|---|
| ▪ Specialisation: small teams, up to 10 employees (ideally 5 to 9);<br>▪ Distribution of power: decentralisation, authority and decision-making is transferred to the development team, which during the project can independently make decisions in agreement with the client, without consulting senior management;<br>▪ Departmentalisation: People-oriented, agile teams are self-organised. In case there are more than 10 developers, it is recommended to form two teams; | ▪ Specialisation: larger teams;<br>▪ Distribution of power: centralised, every decision is documented and verification by senior management and stakeholders is necessary;<br>▪ Departmentalisation: Process-oriented and guided by set goals. Teams are structured. The structure is formalised so that each employee knows the clear tasks; |

## 4.3. Processes

Management processes can be vertical or horizontal. Vertical processes are the processes of business planning and budget planning. Horizontal are processes such as developing new products or entering and fulfilling customer requirements (Galbraith, 2011).

**Table 3**: Comparative analysis based on processes

| Processes in agile approach | Processes in traditional approach |
|---|---|
| ▪ Flexibility and adaptability are present in software development. Usually, during the project's first phase, a vision is developed, which is agreed upon by the business owner, product owner, and stakeholders (Rawsthorne & Shimp 2009; Van Casteren, 2017). In the second step of the process, a framework plan supplements that vision. It includes multiple iterative developments, and in each one the output is improved through all-round activities of the analysis, design, coding, and testing (Alshamrani, & Bahattab 2015). The design is open to change in every part of the iterative implementation. Less importance is given to documentation, and higher to development rapidity. When each iteration has finished, a meeting with the clients is organised. The next iteration of the process is determined by the client given feedback. The team repeats the abovementioned cycle until a product is delivered to the customer. | ▪ In this approach the process follows the linear life cycle model, a software development process where the production cycle progresses sequentially and unidirectionally through several stages. The stages of software development are: requirements specification, conception, analysis, design, coding, testing and debugging, installation and finally, maintenance (McCormic, 2012). The team moves on to the next stage only after the completion of the previous stage is accepted by the relevant stakeholders. The team must maintain the documentation during all stages of development. |

## 4.4. Reward system

The purpose of the reward system is to align the employee's goals with the organisation's goals, according to Galbraith (2011). The organisation's reward system includes basic employee salaries, variable rewards and bonuses, criteria for advancement and promotion system, employee stock options and profit sharing etc. (KErr & Slocum Jr, 2005). In order to influence strategic direction, the reward system must be consistent with structure and processes, according to the STAR model (Benjamin et al, 2012). It provides motivation and encouragement to complete the strategic goals that have been set.

**Table 4**: Comparative analysis based on the rewarding system

| Rewarding system in agile approach | Rewarding system in traditional approach |
|---|---|
| ▪ The approach focuses on client satisfaction, so the reward system is potentially maintained in cases where the client indicates a successful cooperation and collaborative approach with the employee.<br>▪ Employee performance can be noticed after each increment because the results are visible and testable every few weeks, so employee rewards can occur more often. | ▪ The approach is oriented towards planning, time schedules, budget and implementation of the entire system at once following the plan, so the reward system can be maintained when the employee has completed the tasks at earlier stages, before agreed deadline.<br>▪ The results are visible only after the entire phase is completed and verified, so the employee has the opportunity to present his results only in a longer period of time. |

## 4.5. People

This area regulates employment policy, personnel policy, training, and development. Human resources policy creates skills and ways of thinking for the implementation of requirements, where it simultaneously improves people and organisational skills (Galbraith, 2011).

**Table 5**: Comparative analysis based on employee predispositions

| People with an agile approach | People with a traditional approach |
|---|---|
| ▪ Flexible approach requires flexible people;<br>▪ They are characterised by courage, focus, dedication, respect and communication;<br>▪ They are agile, aware of everything that happens on the project and willing to cooperate;<br>▪ Proactive and adaptable to the circumstances and changes an agile approach brings;<br>▪ Have relevant experience; | ▪ Distributed teams of experts;<br>▪ Plan-oriented;<br>▪ They have appropriate and necessary skills;<br>▪ Trusted employees;<br>▪ Their focus is on completing tasks on time; |

| | |
|---|---|
| ▪ Their focus is on customer satisfaction and delivery of high quality software; | |

# 5. CONCLUSION

This article presents conclusions based on the experience of identity management system development, a large scale information systems and technologies project. Its theoretical background is the Galbraiths's Star model which identifies five basic elements of the organisational system: strategy, structure, processes, reward system and people. First part of the article gives brief description of agile vs waterfall approaches. In the following part, elements of the organisational system are described, according to the Galbraiths's Star framework. Conclusions are given from the perspective of team members, participating both in the development under the SCRUM methodology, which was used to implement this specific system, and the Waterfall approach used previously on earlier projects of the organisation.
Main conclusions are, in the respective areas:

- Strategy –the strategy of organisation where agile methodology is implemented is based on the flexibility, responsiveness and close interaction with the client, while the strategy of the organisation where Waterfall is implemented is aimed at efficiency, predictability and ex-ante fixed estimations.
- Structure – the organisation's structure when agile methodology is implemented is based on the small decentralised teams, which should not limit self-organising potential with the size above 10 members. On the other hand, waterfall approach tend to produce more centralised and formalised structure, with high power distance and stricter hierarchy / chain of command. Teams are structured and departmentalisation is based on the criteria of process phase (regarding waterfall stages)
- Processes- in the organisation where the agile methodology is applied, processes are a less important factor for other organisational elements. Main process is defined by the applied agile methodology (in this case, SCRUM) and defines roles and job description of the key methodology facitilitators. However, the core process of software development does not define the organisation with such strength such as in waterfall approaches. Multifunctional teams finish the tasks within the SCRUM sprint without strict specialisation or belonging to the department defined by the process phase. On the other hand, Waterfall approach does not have strong influence based on its main process which defines the methodology, but the core process of software development has strong influence on the other elements of organisation, which are largely defined by the stage – e.g. requirements specification, design, coding or testing.
- Reward system is much more dynamic in an organisation when agile methodology is applied. Results are visible in several times smaller increments than in the waterfall method, so the delay between the performance and the reward is shorter, as it i connections with the reward. So, according to the Vroom (1966) expectancy theory motivation in total should be higher, because the instrumentality – or the connection between the performance and the reward is much higher in the organisations where agile, compare to the waterfall methodologies are applied.
- People – when agile methodologies are applied in the organisation, adaptable and proactive employees are needed and much more likely to perform well, achieve good performance levels and be promoted. Out of the competence triad, knowledge is mostly the same as expected in the organisations where the Waterfall is implemented, the knowledge of methodology mostly differentiates skills and the most observable difference is in the attitudes. The organisation is in much higher need of the employees who do not tend to do what they are told but what is right.

Main limitations of this study are the restricted sample and the qualitative methods. Employees included in this research were using the agile approach with the SCRUM, as the improvement the previous Waterfall approaches. While the organisation is large and dozens of employees participated, methods for data gathering were qualitative – interviews with the employees and direct observation. Therefore, the interpretation of organisational elements could be subjective. Future research should focus on one or a few of the hypotheses proposed in this article and recheck it using the quantitative methods and available measurements of the organisation and its elements.

# 6. LITERATURE


Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2017). Agile software development methods: Review and analysis. arXiv preprint arXiv:1709.08439.

Alahyari, H., Gorschek, T., & Svensson, R. B. (2019). An exploratory study of waste in software development organisations using agile or lean approaches: A multiple case study at 14 organisations. Information and Software Technology, 105, 78-94.

Alshamrani, A., & Bahattab, A. (2015). A comparison between three SDLC models waterfall model, spiral model, and Incremental/Iterative model. International Journal of Computer Science Issues (IJCSI), 12(1), 106.

Andrei, B. A., Casu-Pop, A. C., Gheorghe, S. C., & Boiangiu, C. A. (2019). A study on using waterfall and agile methods in software project management. Journal Of Information Systems & Operations Management, 125-135.

Anitha, P. C., & Prabhu, B. (2012). Integrating requirements engineering and user experience design in product life cycle management. In 2012 First International Workshop on Usability and Accessibility Focused Requirements Engineering (UsARE) (pp. 12-17). IEEE.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... & Thomas, D. (2001). Manifesto for agile software development.

Benjamin, B. M., Naimi, L. L., & Lopez, J. P. (2012). Organizational Change Models In Human Resource Development. Leadership & Organizational Management Journal, 2012(2).

Boehm, B., & Turner, R. (2003). Using risk to balance agile and plan-driven methods. Computer, 36(6), 57-66.

Boehm, B., & Turner, R. (2005). Management challenges to implementing agile processes in traditional development organizations. IEEE software, 22(5), 30-39.

Burger, A., Cichiwskyj, C., Schmeißer, S., & Schiele, G. (2020). The Elastic Internet of Things-A platform for self-integrating and self-adaptive IoT-systems with support for embedded adaptive hardware. Future Generation Computer Systems, 113, 607-619.

Chan, V. W. (2021). Internet of Things and Smart Cities. IEEE Communications Magazine, 59(10), 4-6.

Cho, J. J. (2010). An exploratory study on issues and challenges of agile software development with scrum. Utah state University, Theses and dissertations repository. Available at: https://digitalcommons.usu.edu/etd/599/ .

Cockburn, A. (2004). Crystal clear: A human-powered methodology for small teams: A human-powered methodology for small teams. Pearson Education.

Cawley, O., Wang, X., & Richardson, I. (2010, October). Lean/agile software development methodologies in regulated environments–state of the art. In International Conference on Lean Enterprise Software and Systems (pp. 31-36). Springer, Berlin, Heidelberg.

Devedžić, V. (2021). Quo Vadis, AI?. Management:Journal Of Sustainable Business And Management Solutions In Emerging Economies, 26(1), 1-12. doi:10.7595/management.fon.2021.0001

Ebert, C., Abrahamsson, P., & Oza, N. (2012). Lean software development. IEEE Computer Architecture Letters, 29(05), 22-25.

Erickson, J., Lyytinen, K., & Siau, K. (2005). Agile modeling, agile software development, and extreme programming: the state of research. Journal of Database Management (JDM), 16(4), 88-100.

Flora, H. K., & Chande, S. V. (2014). A systematic study on agile software development methodologies and practices. International Journal of Computer Science and Information Technologies, 5(3), 3626-3637.

Fowler, M., & Highsmith, J. (2001). The agile manifesto. Software development, 9(8), 28-35.

Galbraith, J. R. (2011). The star model. The STAR Model. Available on: http://www.jaygalbraith.com/images/pdfs/StarModel.pdf

Highsmith, J., & Cockburn, A. (2001). Agile software development: The business of innovation. Computer, 34(9), 120-127.

Jaško, O., Čudanov, M., Jevtić, M., & Krivokapić, J. (2013). Projektovanje organizacije. Beograd, Srbija: Fakultet organizacionih nauka (FON).

Jaško, O., Čudanov, M., Jevtić, M., & Krivokapić, J. (2017). Organizacioni dizajn–pristupi, metode i modeli. Beograd, Srbija: Fakultet organizacionih nauka (FON).

Kates, A., & Galbraith, J. R. (2010). Designing your organisation: Using the STAR model to solve 5 critical design challenges. John Wiley & Sons.

Kerr, J., & Slocum Jr, J. W. (2005). Managing corporate culture through reward systems. Academy of Management Perspectives, 19(4), 130-138.

Khan, S., Nicho, M., & Takruri, H. (2016). IT controls in the public cloud: Success factors for allocation of roles and responsibilities. Journal of information technology case and application research, 18(3), 155-180.

Kulkarni, V., Barat, S., & Ramteerthkar, U. (2011). Early experience with agile methodology in a model-driven approach. In International Conference on Model Driven Engineering Languages and Systems (pp. 578-590). Springer, Berlin, Heidelberg.

Leau, Y. B., Loo, W. K., Tham, W. Y., & Tan, S. F. (2012). Software development life cycle AGILE vs traditional approaches. In International Conference on Information and Network Technology (Vol. 37, No. 1, pp. 162-167).

McCormick, M. (2012). Waterfall vs. Agile methodology. MPCS, Available online: http://www.mccormickpcs.com/images/Waterfall_vs_Agile_Methodology.pdf .

Mintzberg, H. (1993). Structure in fives: Designing effective organisations. New York, USA: Prentice-Hall, Inc.

Mirza, M. S., & Datta, S. (2019). Strengths and Weakness of Traditional and Agile Processes-A Systematic Review. Journal of Software., 14(5), 209-219.

Mnkandla, E., & Dwolatzky, B. (2004). A survey of agile methodologies. Transactions of the South African Institute of Electrical Engineers, 95(4), 236-247.

Mora, M., Gómez, J. M., O'Connor, R. V., & Buchalcevová, A. (Eds.). (2020). Balancing Agile and Disciplined Engineering and Management Approaches for IT Services and Software Products. IGI Global.

Maruping, L. M., Venkatesh, V., & Agarwal, R. (2009). A control theory perspective on agile methodology use and changing user requirements. Information systems research, 20(3), 377-399.

Matharu, G. S., Mishra, A., Singh, H., & Upadhyay, P. (2015). Empirical study of agile software development methodologies: A comparative analysis. ACM SIGSOFT Software Engineering Notes, 40(1), 1-6.

Nikiforova, O., Nikulsins, V., Sukovskis, U.: Integration of MDA Framework into the Model of Traditional Software Development. In: Frontiers in Artificial Intelligence and Applications, Databases and Information Systems V, vol. 187, pp. 229–239. IOS Press, Amsterdam (2009)

Rushby, J. (2016, January). Trustworthy self-integrating systems. In International Conference on Distributed Computing and Internet Technology (pp. 19-29). Springer, Cham.

Rawsthorne, D., & Shimp, D (2009). Scrum in a nutshell. Available online: https://www.scrumalliance.org/community/articles/2009/december/scrum-in-a-nutshell .

Schwaber, K., & Beedle, M. (2002). Agile software development with Scrum (Vol. 1). Upper Saddle River: Prentice Hall.

Sharma, S., Sarkar, D., & Gupta, D. (2012). Agile processes and methodologies: A conceptual study. International journal on computer science and Engineering, 4(5), 892.

Vroom, V. H. (1966). Organisational choice: A study of pre-and postdecision processes. Organisational behavior and human performance, 1(2), 212-225.

Van Casteren, W. (2017). The Waterfall Model and the Agile Methodologies: A comparison by project characteristics. Research Gate, 2, 1-6.

Weisert, C. (2003). There's no such thing as the Waterfall Approach. Retrieved from http://www.idinews.com/waterfall.html

What is Scrum? Retrieved from https://www.atlassian.com/agile/scrum

Williams L. (2007) A survey of agile development methodologies. Available at: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.709.7775&rep=rep1&type=pdf