ISSN 2224-087X. Електроніка та інформаційні технології. 2023. Випуск 22. С. 70–82
Electronics and information technologies. 2023. Issue 22. P. 70–82

UDC 004.8 DOI: https://doi.org/10.30970/eli.23.7

COMPARISON USAGE OF AGILE, WATERFALL AND OTHER APPROACHES FOR SOFTWARE DEVELOPMENT UNDER UNSTABLE SITUATIONS CAUSED BY RUSSIAN INVASION OF UKRAINE

V. Franiv[1], S. Vasylyuk[2], I. Franiv[3]

[1]Ivan Franko National University of Lviv,
107 Tarnavskoho St., 79017 Lviv, Ukraine
volodymyr.franiv@lnu.edu.ua

[2]Lviv Polytechnic National University,
12 Bandery St, Lviv, 79013, Ukraine
sofiia.v.vasyliuk@lpnu.ua

[3]Lviv University of Trade and Economics,
10 Tuhan-Baranovskyi St, Lviv, 79005, Ukraine
ihor.franiv@lute.lviv.ua

This article compares the usage of agile, waterfall and other software development approaches under unstable situations caused by the Russian invasion of Ukraine. The study investigates how software development teams adapted to the crisis and the effectiveness of their methodologies in this context. The research is based on a survey of software development teams that were affected by the crisis, and it analyze the data collected from these surveys. The article discusses the advantages and disadvantages of mainly agile, waterfall for software development under unstable situations and provides recommendations for software development teams facing similar crises in the future. The findings of this study contribute to the understanding of software development methodologies and their effectiveness in unstable situations caused by geopolitical crises.

Keywords: agile, waterfall, software development, unstable situations, crisis, Russian invasion, Ukraine.

Introduction.

Ukraine has emerged as a popular destination for software development businesses since its independence in 1991. It has become a hub for outsourcing and outstaffing in the software development industry. Grammarly, Petcube, Depositphotos is just a short list of some examples of successful product software companies that have started in Ukraine. The main reason why Ukraine become so popular for software development business is caused by next factors:

1. Skilled and Educated Workforce: Ukraine has a large pool of highly skilled and educated software developers, with many holding advanced degrees in computer science and related fields. Ukrainian universities are renowned for their strong programs in science, technology, engineering, and mathematics (STEM) fields, producing a steady stream of talented software developers. Different high schools produce over 36,000 IT graduates annually, with a strong emphasis on practical skills development, making it an ideal location for hiring top-notch talent.

2. Competitive Pricing: Ukraine offers a competitive pricing advantage compared to other European countries, with lower labor costs and overhead expenses. This makes it an attractive destination for software development companies looking to save costs without sacrificing quality.
3. Cultural Similarities: Ukraine has a strong cultural affinity with Western Europe and North America, making it easier for software development companies to communicate and collaborate with Ukrainian teams.
4. Cost-Effectiveness: The cost of living in Ukraine is relatively low compared to Western European countries and the United States. This means that startup costs are lower in Ukraine, and entrepreneurs can benefit from lower wages, office rents, and other overheads.
5. Access to Western Markets: Ukraine has a favorable time zone for doing business with Western European and American clients. Additionally, Ukraine has a visa-free regime with the European Union, making it easy for Ukrainian product software companies to collaborate with European businesses.
6. Thriving Tech Ecosystem: Ukraine has a rapidly growing startup ecosystem, with many incubators, accelerators, and venture capital funds. This ecosystem provides support and resources to help entrepreneurs and product software companies succeed.

However, some factors also contribute to the prevalence of outsourcing and outstaffing rather than product software development companies in Ukraine. For example, the competitive pricing advantage may make it difficult for local companies to compete in the global market for software products. In addition, the focus on outsourcing and outstaffing may lead to a lack of investment in developing proprietary software products and intellectual property.

Overall, the dominance of outsourcing and outstaffing in Ukraine's software development industry can be attributed to a combination of factors, including a skilled workforce, competitive pricing, cultural similarities, and government support. While this focus has its advantages, it may also limit the growth and competitiveness of local product software development companies.

Software development companies in Ukraine tend to value talent, experience, and expertise over age. As a result, there are people of all ages working in the IT industry in Ukraine, from recent university graduates to experienced professionals.

The Russian invasion of Ukraine in 2022 created a crisis that impacted many industries, including software development. The software development industry is known for its strict adherence to project management methodologies such as agile and waterfall. However, when a crisis arises, the efficacy of these methodologies may be called into question. This article aims to compare the usage of agile, waterfall and other software development methodologies under unstable situations caused by the Russian invasion of Ukraine. The study investigates how software development teams adapted to the crisis and the effectiveness of their methodologies in this context.

By analyzing the data collected from surveys of software development teams that were affected by the crisis, this article provides insights into the advantages and disadvantages of each methodology under unstable situations. The findings of this study will contribute to the understanding of software development methodologies and their effectiveness in unstable situations caused by geopolitical crises.

Literature review.

Software development approaches are a set of principles, practices, and procedures that guide the development of software. They provide a structured approach to software development, from the planning phase to the deployment phase. Two popular software development approaches are agile and waterfall [1-5].

The waterfall software development approach is a sequential approach to software development [2]. It involves a linear sequence of phases, with each phase completed before moving on to the next one. The phases typically include requirements gathering, design, implementation, testing, and maintenance.

One of the advantages of the waterfall methodology is its predictability. Since the requirements are gathered upfront and the phases are completed in a specific order, it is easier to estimate the time and cost of a project. This makes it ideal for large, complex projects with well-defined and stable requirements [6-8].

However, the waterfall methodology has several disadvantages. One of the major drawbacks is its inflexibility. Once a phase is completed, it cannot be revisited, which can lead to problems if changes are needed later in the project. This can lead to delays and cost overruns [1-10].

Additionally, the waterfall methodology does not provide much opportunity for customer feedback until the testing phase, which can lead to misinterpretation of requirements and dissatisfaction with the final product.

Several studies have compared the effectiveness of the waterfall methodology with other methodologies. A study [10] compared the waterfall methodology with the agile methodology and found that the waterfall methodology was less effective in managing projects with changing requirements.

Another studies [11-15] found that the waterfall methodology was effective in managing large, complex projects with well-defined requirements. However, in studies also noted that the methodology was less effective in managing projects with uncertain requirements.

The waterfall methodology is a popular approach to software development, particularly for large, complex projects with stable requirements. However, its inflexibility and lack of customer feedback can lead to problems in projects with changing requirements or uncertain scope. The choice of methodology depends on the specific needs and requirements of the project.

This approach typically involves several meetings throughout the project to ensure that each phase is completed successfully. Here are some of the typical meetings in the Waterfall software development methodology [8]:

- Requirements Gathering Meeting: In this meeting, the project stakeholders (including the client, product owners, and project managers) meet to define and document the project requirements. This includes identifying the project scope, goals, and deliverables.
- Design Meeting: In this meeting, the software architects and designers meet to create a high-level design of the system, including the software architecture, user interface design, and database schema.
- Development Meeting: In this meeting, the software developers meet to implement the design and create the software code. The team will typically discuss coding standards, version control, and testing methodologies.
- Testing Meeting: In this meeting, the quality assurance team meets to test the software and ensure that it meets the project requirements. This includes identifying any bugs or issues that need to be fixed before the software is released.

- Deployment Meeting: In this meeting, the project team meets to plan the deployment of the software. This includes identifying the deployment environment, creating a deployment plan, and testing the deployment process.
- Maintenance Meeting: In this meeting, the project team meets to discuss ongoing maintenance and support of the software. This includes identifying any issues that need to be addressed and planning for future updates and enhancements.

These meetings are typically scheduled at the beginning of the project and occur at regular intervals throughout the development process. The purpose of these meetings is to ensure that each phase of the project is completed successfully and that the software meets the project requirements.

Agile methodology, on the other hand, is an iterative and incremental approach that emphasizes flexibility and adaptability [7-11]. It involves breaking down a project into smaller parts and working on them in short cycles called sprints. The agile methodology is suitable for projects with changing requirements or uncertain scope.

Scrum and Kanban are two popular software development methodologies that are often used in agile software development.

Scrum is an iterative and incremental approach that emphasizes collaboration, flexibility, and continuous improvement. It involves breaking down a project into smaller parts called sprints, with each sprint lasting one to four weeks. During each sprint, the team focuses on delivering a working product increment that meets the customer's needs [11-15].

One of the advantages of Scrum is its flexibility. The methodology allows for changes in requirements, and the team can adjust their work accordingly. Additionally, Scrum emphasizes teamwork and collaboration, which can lead to better communication and a more cohesive team.

Kanban is another agile methodology that emphasizes continuous delivery and improvement. Kanban involves visualizing the workflow and limiting work in progress to avoid bottlenecks. The approach also involves measuring and analyzing the flow of work to identify areas for improvement [15].

Like Scrum, Kanban is its flexible approach. The methodology allows for changes in requirements, and work can be added or removed from the backlog as needed. Additionally, Kanban emphasizes visual management, which can lead to better communication and a clearer understanding of the work being done.

Several studies have compared the effectiveness of Scrum and Kanban. A study [15-20] found that Scrum was more effective than Kanban in managing large, complex projects. However, the study also noted that Kanban was more effective in managing smaller, less complex projects.

Another study [21] found that both Scrum and Kanban were effective in managing software development projects. The study found that the choice of methodology depended on the specific needs and requirements of the project.

Both Scrum and Kanban are popular software development approaches that are often used in agile software development. The choice of methodology depends on the specific needs and requirements of the project. Scrum may be more suitable for large, complex projects, while Kanban may be more suitable for smaller, less complex projects [22-30].

Here are some of the typical meetings in the Agile software development methodology [21]:

- Sprint Planning Meeting: In this meeting, the project team meets to plan the work for the upcoming sprint. This includes selecting the items from the product backlog that will be worked on during the sprint, defining the sprint goal, and creating a plan for how the work will be accomplished.

- Daily Stand-Up Meeting: Also called a daily scrum, this is a brief meeting held every day during the sprint. Each team member gives a brief update on their progress since the last meeting, including any challenges or roadblocks they are facing. The goal of this meeting is to keep everyone on the same page and identify any issues early on.
- Sprint Review Meeting: At the end of each sprint, the project team meets to review the work that was completed during the sprint. The team demos the working software to stakeholders and discusses what went well and what could be improved. This feedback is used to inform the next sprint planning meeting.
- Sprint Retrospective Meeting: Also called a retro, this is a meeting held after the sprint review to reflect on the sprint and identify areas for improvement. The team discusses what went well, what didn't go well, and what changes can be made to improve the process for the next sprint.
- Backlog Refinement Meeting: In this meeting, the project team meets to review and refine the product backlog. This includes adding new items, re-prioritizing existing items, and breaking down larger items into smaller, more manageable tasks.

These meetings are typically held at regular intervals throughout the project and are designed to encourage collaboration, feedback, and continuous improvement. The Agile methodology values communication and flexibility, and these meetings are a key part of that approach.

Some studies have compared the effectiveness of agile and waterfall methodologies. A study [29] found that the waterfall model was effective in managing large-scale projects with stable requirements. However, the model was less effective in managing projects with changing requirements.

On the other hand, a study [10] found that the agile methodology was effective in managing projects with changing requirements. The study found that the agile methodology enabled teams to respond quickly to changes in requirements and deliver software that met the customer's needs.

Other studies have also compared the effectiveness of agile and waterfall approaches in specific contexts. For example, a study by [29] compared the effectiveness of agile and waterfall methodologies in small software development teams. The study found that agile methodology was more effective in managing small software development teams.

There are many other software development approaches that exist besides Waterfall, Scrum, and Kanban. Here are a few examples [29]:

1. Lean Software Development: This methodology emphasizes delivering value to the customer and eliminating waste in the development process.
2. Extreme Programming (XP): XP is an agile methodology that emphasizes teamwork, frequent releases, and continuous testing.
3. Feature Driven Development (FDD): FDD is an iterative and incremental approach that emphasizes the development of specific features or functionality.
4. Rapid Application Development (RAD): RAD is a fast-paced methodology that emphasizes prototyping and iterative development.
5. Crystal: Crystal is a family of methodologies that emphasizes flexibility and adapting to the specific needs of the project.

These are just a few examples of the many software development methodologies and approaches that exist. Software development methodologies play a crucial role in managing software development projects. Both agile and waterfall approaches have their advantages and disadvantages and are suitable for different types of projects. [29-37]

Unpredicted consequences caused by Invasion.

The invasion of the Russian Federation into Ukraine in 2014 caused many unpredicted circumstances for software developers in Ukraine. These circumstances included disruptions in the political and economic environments, as well as challenges related to infrastructure and security.

One of the primary impacts on software developers was the disruption to the political and economic environment. The annexation of Crimea and the ongoing conflict in eastern Ukraine led to a loss of foreign investment, which affected the software industry. This led to reduced funding for startups, which in turn impacted the availability of new projects and job opportunities for software developers.

Infrastructure was another area that was impacted by the invasion. The conflict resulted in the destruction of key infrastructure such as power plants and telecommunication networks. This made it difficult for software developers to access essential resources such as electricity, internet connectivity, and server infrastructure. Furthermore, the disruption in transportation and logistics made it difficult to deliver software products to clients.

Security was also a major concern for software developers in Ukraine. The invasion created an unstable environment with an increased risk of cyber-attacks, espionage, and sabotage. This led to increased security measures being implemented, which impacted the development process by increasing the time and resources needed to ensure the security of software products.

Listed factors above impacted the software industry and made it difficult for software developers to access essential resources and deliver software products to clients.

Impact of blackout.

A blackout caused by the Russian invasion of Ukraine, which results in a lack of electricity and internet connection, had a significant impact on all software development approaches, as they heavily rely on technology and digital communication.

Lack of electricity cause significant impact on waterfall, scrum, and kanban software development approaches, as they all require a steady and reliable power supply.

Waterfall methodology, which follows a sequential process, involves different teams working on specific stages of the development process, from requirements gathering to testing and deployment. The lack of electricity cause delays and interruptions in the work of each team, leading to missed deadlines and reduced efficiency. In addition, Waterfall, being a traditional methodology, relies heavily on documentation, and a lack of internet connection made it impossible to access or update the necessary documents.

Scrum methodology, which emphasizes iterative development, relies on daily meetings and frequent communication between team members. Without electricity, team members were not able to access the necessary digital tools, such as project management software, to share updates and track progress, which leads to miscommunication and decreased productivity.

Similarly, Kanban methodology, which focuses on visualizing and optimizing workflows, heavily relies on electronic boards and digital tools for task management and tracking. A lack of electricity or internet connection cause interruptions in the flow of work and make it challenging to keep track of tasks and progress.

In summary, lack of electricity and internet connection significantly impact software development methodologies that rely heavily on technology and digital tools. Comparing aspects of agile and waterfall model under blackout circumstances is presented in Table 1. It causes delays, interruptions, miscommunication, and decreased productivity, making it challenging to meet project deadlines and deliver high-quality software. To mitigate the impact, teams were establishing contingency plans for working offline, such as setting up backup communication channels or using physical Kanban boards. On other hand software development companies was searching for independent power sources such as personal power generators which use fuel and satellite internet connection. Other solution was conducting meetings according with blackout plan provided by electricity suppliers. However, the impact on productivity and quality was still significant.

Table 1. Comparison of different aspects of agile (scrum, kanban) and waterfall approaches for software development under consequences of blackout.

Aspects	Agile (scrum, kanban)	Waterfall
Frequent regular meetings	High impact, approach requires daily meeting	Medium impact, approach doesn’t require everyday meeting
Requirements gathering and clarification	Medium impact, approach requires grooming meeting, review, and retrospective	High impact if blackout is during ‘Requirements Analysis’ and ‘System design’ phases. Low impact for ‘Implementation’ and ‘Testing’ phases.
Team communication	Medium impact, this approach requires regular customer feedback that should be discussed within a team	Low impact for phases ‘Implementation’ and ‘Testing’ High impact for phase ‘Maintenance’

Impact of air alarm.

An air alarm, which signals the possible approach of enemy aircraft, significantly impact on software development methodologies such as waterfall, kanban, and scrum during the Russian invasion of Ukraine used by software development teams at Ukraine.

Waterfall methodology involves different teams working on specific stages of the development process. An air alarm cause interruptions and delays in the work of each team, leading to missed deadlines and reduced efficiency. The safety of team members is also a concern, as they need to evacuate the building or take cover during an air raid.

Kanban methodology, which focuses on visualizing and optimizing workflows, also was impacted by an air alarm. Team members needed to stop work and seek shelter, causing delays in task completion and potentially impacting the flow of work.

Similarly, Scrum methodology, which emphasizes iterative development, relies on daily meetings and frequent communication between team members. An air alarm disrupt communication, leading to miscommunication and decreased productivity. Team members also need to evacuate the building, leading to delays in completing tasks and potentially impacting sprint goals.

An air alarm significantly impacts software development approaches during a crisis such as the Russian invasion of Ukraine. It causes delays, interruptions, miscommunication, and decreased productivity, making it challenging to meet project deadlines and deliver high-quality software. The safety of team members is also a concern, and appropriate safety measures should be taken to ensure their well-being.

How does blackout impact on continuous integration and continuous delivery in terms of software development?

The lack of electricity and internet have a significant impact on continuous integration and continuous delivery (CI/CD) in software development.

Continuous integration refers to the practice of regularly merging code changes into a shared repository and ensuring that the integrated code builds and runs successfully. Continuous delivery refers to the process of continuously developing, testing, and deploying software changes to production.

Here are some ways that the lack of electricity and internet impact CI/CD in software development:

1. Slower development cycles: Without reliable electricity and internet access, software developers was unable to work efficiently, leading to slower development cycles. This results in missed deadlines and delayed releases.
2. Reduced collaboration: CI/CD relies on collaboration among team members to ensure that code changes are integrated correctly and tested thoroughly. A lack of electricity and internet access limit communication and collaboration among team members, making it harder to maintain the continuous integration and delivery process.
3. Increased risk of errors: Without a reliable power supply and internet access, developers had to work on local systems or with limited resources, which increase the risk of errors and bugs in the code.
4. Limited access to cloud services: Many CI/CD tools and services are hosted on cloud platforms, which require reliable internet access. Without a stable internet connection, developers was not able to use these services, which impact the efficiency and effectiveness of the CI/CD process.

The lack of electricity and internet had a significant impact on the efficiency, speed, and reliability of continuous integration and continuous development and delivery in software development.

To mitigate these challenges, developers was able to adopt alternative strategies, such as working offline and using local testing environments, until reliable electricity and internet access was restored.

Conclusion.

The article discusses the impact of a recent blackout caused by the Russian invasion of Ukraine on software development methodologies that rely heavily on technology and digital communication. Specifically, the article compares the impact on Waterfall, Scrum, and Kanban methodologies, which are commonly used by software development teams in Ukraine.

Waterfall methodology is a traditional approach that follows a sequential process, involving different teams working on specific stages of the development process, from requirements gathering to testing and deployment. The lack of electricity caused by the blackout led to delays and interruptions in the work of each team, resulting in missed deadlines and reduced efficiency. Additionally, Waterfall methodology relies heavily on documentation, and a lack of internet connection made it impossible to access or update the necessary documents.

Scrum methodology, which emphasizes iterative development, relies on daily meetings and frequent communication between team members. Without electricity, team members were not able to access the necessary digital tools, such as project management software, to share updates and track progress, which led to miscommunication and decreased productivity.

Similarly, Kanban methodology, which focuses on visualizing and optimizing workflows, heavily relies on electronic boards and digital tools for task management and tracking. The lack of electricity or internet connection caused interruptions in the flow of work, making it challenging to keep track of tasks and progress.

Overall, the lack of electricity and internet connection had a significant impact on software development methodologies that rely heavily on technology and digital tools. Teams had to establish contingency plans for working offline, such as setting up backup communication channels or using physical Kanban boards. Additionally, software development companies were searching for independent power sources such as personal power generators that use fuel and satellite internet connections. Another solution was conducting meetings according to a blackout plan provided by electricity suppliers. However, the impact on productivity and quality was still significant.

In addition to the blackout, the article discusses the impact of an air alarm on software development methodologies during the Russian invasion of Ukraine. An air alarm signals the possible approach of enemy aircraft and requires team members to evacuate the building or take cover. This significantly impacted Waterfall, Kanban, and Scrum methodologies, as team members were forced to stop work and seek shelter, causing delays in task completion and potentially impacting the flow of work. The safety of team members is also a concern during such situations, and appropriate safety measures should be taken to ensure their well-being.

In summary, the article highlights the importance of contingency planning and safety measures during crisis situations that can significantly impact software development methodologies. It also emphasizes the need for software development companies to have independent power sources and alternative communication channels in place to mitigate the impact of blackouts and other similar situations.

References.

[1] *Li, Min.* (2019). Application on agile technology for provoding international joint scientific projects. Management of development of complex systems, 38, 103–110. DOI: 10.6084/m9.figshare.9788555

[2] *Bushuev S. D., Bushuev S. A., Bushueva N. S., Kozir B. J.* (2018) Informatsiini tekhnolohii rozvytku kompetentsii menedzheriv zupravlinnia proektamy na osnovi hlobalnykh trendiv

Informatsiini tekhnolohii i zasoby navchannia [Information technologies for the development of competencies of project management manag-ers based on global trends Information technologies and teaching aids]. Informatsiini tekhnolohii i zasoby navchannia, vol. 68, no. 6, pp. 218–234.

[3] *Martin R., Newquark J., Coss R.* Rapid development of programs. Principles, examples, - Williams, 2004. - 752 pp.

[4] *Barjtya S.* A detailed study of Software Development Life Cycle (SDLC) Models / S. Barjtya, A. Sharma, U. Rani // International Journal of Engineering And Computer Science. – 2017. – Vol. 6, Issue. 7. – P. 22097.

[5] *Vale, J. W. S. P., & Carvalho, M. M. D.* (2017). Risk and uncertainty in projects management: literature review and conceptual framework. Revista GEPROS, 12(2), 93.

[6] *Lalsing V., Kishnah S., Pudaruth S.* People Factors In Agile Software Development And Project Manage-ment. International Journal of Software Engineering & Applications (IJSEA), vol. 3, no. 1, January, pp. 117-137.

[7] *Hanif, T., Limbachiya, M.* (2010). Selecting the right project management approach using 6P. 24th World Conference IPMA (International Project Management Association). Istanbul, Turkey, Pp. 183–189.

[8] *Lijoi G.* Can we combine Agile and Waterfall development strategies? [Electronic resource]. - Access mode: www.projectsmart.co.uk/can-we-combine-agile-and-waterfall-developmentstrategies.php

[9] The State of Agile study has been published – 2019. Available at: https://www.pmservices.ru/project-management-news/opublikovano-issledovanie-state-of-agile-2019/.

[10] *Casteren V.* The Waterfall Model and the Agile Methodologies: A comparison by project characteristics — short. 10.13140/RG.2.2.10021.50403, 2017. URL: https://www.researchgate.net/publication/313768860_The_Waterfall_Model_and_the_Agile_Methodologies_A_comparison_by_project_characteristics_-_short

[11] *Christopher M.* (2010). The Agile Supply Chain: Competing in Volatile Markets. Available at: https://dspace.lib.cranfield.ac.uk/bitstream/handle/1826/2658/Agile%20supply%20chain-2000.pdf?%20sequence=1

[12] *Whitaker S.* (2014). How to Build Your Own Project Management Methodology [Electronic resource]. Available at: www/URL: http://seanwhitaker.com/ how-to-build-your-own -project- management-methodology/

[13] Agile methodology. Top 10 Mistakes When Using Agile // Official Website of the Gant BPM Consulting Company [Electronic Resource]. URL: https://gantbpm.ru/metodologiya-agile/

[14] *Bhuvaneswar T.* A Survey on Software Development Life Cycle Models / T. Bhuvaneswar, S. Prabaharan // International Journal of Computer Science and Mobile Computing. – 2013. – Vol. 2, Issue. 5. – P. 262-267.

[15] *Ernest M.* About Software Engineering Frameworks and Methodologies / M. Ernest // IEEE AFRICON. – 2009. – P. 1-5. https://doi.org/10.1109/afrcon.2009.5308117.

[16] *Royce W.* Managing the development of large software systems, 1970 [Electronic resource]: - Access mode: www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf

[17] Agile methodology. Examples, when to use it, advantages and disadvantages. Available at: https://twproject.com/ blog/agile-methodology-advantages-disadvantages-inno-vative-method/

[18] *Ryabokon N. P., Ryabokon A. A., Ryabokon B. A*. (2019) Vprovadzhennia metodolohii agile: tsinnisno orii-entovanyi pidkhid [Implementation of agile methodolo gy: value-oriented approach]. Zbirnyk naukovykh prats ChDTU. Seriia «Ekonomichni nauky», no. 49, pp. 34–42.

[19] *Miller R., & Lessard D.* (2001). Understanding and managing risks in large engineering projects. International Journal of Project Management, 19(8), 437–443.

[20] *Hass K. B.* The blending of traditional and agile project management. PM World Today, IX (V), 1-8 [Electronic resource]. - Access mode: https://www.mx1.chelsoftusa.com/uploads/2/8/3/8/2838312/agile_well_explained. pdf

[21] *Smolich D. V.* (2019) Innovatsiini metody upravlinnia proektamy [Innovative methods of project management]. Ekonomichnyi forum, no. 4, pp. 50–53.

[22] Hybrid project management manifesto // Official site of the manifest of hybrid software development [Electronic resource]. URL: https://www.binfire.com/hybrid-project-management-manifesto/

[23] *Silkina Y. O.* Agile-menedzhment – efektyvna praktyka systemy upravlinnia pidpryiemstvom [Agile management is an effective practice of the enterprise management system]. DOI: http://doi.org/10.31617/k.knute.2019-04-12.19

[24] *Pich M. T., Loch C. H., & De Meyer A.* (2002). On uncertainty, ambiguity, and complexity in project management. Management Science, 48(8), 1008–1023.

[25] Manifesto for Agile Software Development // Official website of the flexible software development manifesto [Electronic resource]. URL: http://www.agilemanifesto.org/

[26] *Atkinson R., Crawford L., & Ward S.* (2006). Fundamental uncertainties in projects and the scope of project management. International Journal of Project Management, 24(8), 687-698.

[27] Principles behind the Agile Manifesto. Available at: http://agilemanifesto.org/principles.html.

[28] *Meredith J. R. & Mantel S. J. Jr.* (2005). Project Management A Managerial Approach (6th ed.). New York: John Wiley & Sons.

[29] *Alshamrani Adel*. (2021). A Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model. http://www.academia.edu/ 10793943/A_Comparison_Between_Three_SDLC_Models_Waterfall_Model_Spiral_Model_and_Incremental_ Iterative_Model.

[30] *Bassil Y.* A Simulation Model for the Waterfall Software Development Life Cycle / Y. Bassil // International Journal of Engineering & Technology (iJET). – 2012. –Vol. 2, No. 5. – P. 16.

[31] *Ward S., & Chapman C.* (2003). Transforming project risk management into project uncertainty management. International Journal of Project Management, 21(2), 97-105.

[32] *Silva V. B. S., Schramm F., & Damasceno A. C.* (2016). A multicriteria approach for selection of agile methodologies in software development projects. 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Budapest, Hungary, 2056-2060. https://doi.org/10.1109/SMC.2016.7844542

[33] *Pammer V. Bratic M.* Surprise, surprise: activity log based time analytics for time management. In CHI ‘13 Extended Abstracts on Human Factors in Computing Systems (CHI EA ‘13). Association for Computing Machinery, New York, NY, USA, 2013. P. 211–216. URL: https://doi.org/10.1145/2468356.2468395

[34] *Cohen S.* A Software System Development Life Cycle Model for Improved Stakeholders’ Communication and Collaboration / S. Cohen, D. Dori, U. de Haan // International Journal of Computers, Communications & Control. – 2010. – Vol. V, No. 1. – P. 20-41. https://doi.org/10.15837/ijccc.2010.1.2462.

[35] Agile software development: Impact on productivity and quality / A. Ahmed, S. Ahmad, N. Ehsan, E. Mirza and S.Z. Sarwar // Management of Innovation and Technology (ICMIT), IEEE International Conference. – 2010. – P. 287-291. https://doi.org/10.1109/icmit.2010.5492703.

[36] *Ruparelia N. B.* Software Development Lifecycle Models / N.B. Ruparelia // ACM SIGSOFT Software Engineering Notes. – 2010. – Vol. 35, No. 3. – P. 8-13. https://doi.org/10.1145/1764810.1764814.

[37] *Rastogi V.* Software Development Life Cycle Models Comparison, Consequences / V. Rastogi // International Journal of Computer Science and Information Technologies. – 2015. – Vol. 6, No. 1. – P. 168.

ПОРІВНЯННЯ ВИКОРИСТАННЯ AGILE, WATERFALL, ТА ІНШИХ ПІДХОДІВ ДО РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ В УМОВАХ НЕСТАБІЛЬНОЇ СИТУАЦІЇ, СТВОРЕНІЙ ВТОРГНЕННЯМ РОСІЙСЬКОЇ ФЕДЕРАЦІЇ В УКРАЇНУ.

В. Франів[1], С. Василюк[2], І. Франів[3]

Львівський національний університет імені Івана Франка,
вул. Тарнавського 107, 79026 Львів, Україна
volodymyr.franiv@lnu.edu.ua

[2]*Національний університет «Львівська Політехніка»,*
вул. Бандери 12, 79013 Львів, Україна
sofiia.v.vasyliuk@lpnu.ua

[3]*Львівський торгово-економічний університет,*
вул. Тугана-Барановського 10, 79005 Львів, Україна
ihor.franiv@lute.lviv.ua

У даній статті здійснюється компаративний аналіз використання підходів Agile, Waterfall та інших методологій розробки програмного забезпечення в умовах нестабільної ситуації, що виникла через російське вторгнення в Україну. Дослідження базується на опитуванні команд розробників програмного забезпечення, які зазнали впливу цієї кризи.

Автори статті розглядають, як різні команди розробників адаптували свої підходи до розробки програмного забезпечення в умовах кризової ситуації. Порівнюються ефективність Agile та Waterfall підходів в умовах планових відключень електроенергії, роботи без постійного інтернет з’єднання та непередбачуваних повітряних тривог. Виділяються переваги та недоліки кожної з них. Особливу увагу приділяється Agile підходу до розробки програмного забезпечення оскільки саме він являється найбільш поширеним серед Українських IT команд.

Результати цього дослідження можуть виявитися корисними для команд розробників програмного забезпечення, які можуть стикнутися з подібною кризою в майбутньому. В статті надаються рекомендації, які допоможуть командам зрозуміти, який підхід до розробки програмного забезпечення може бути більш ефективним та адаптованим до нестабільних умов.

Такий аналіз важливий, оскільки допомагає зрозуміти, які зміни в методології розробки можуть бути корисними в періоди кризи, коли стандартні умови роботи можуть бути порушені. Отримані висновки можуть стати основою для подальшого вдосконалення підходів до розробки програмного забезпечення в умовах геополітичних небезпек.

Ключові слова: agile, waterfall, розробка програмного забезпечення, нестабільна ситуація, криза, війна в Україні.

Стаття надійшла до редакції 20.07.2023
Прийнята до друку 15.08.2023