КОМПЬЮТЕРНЫЕ НАУКИ, ПРИБОСТРОЕНИЕ И АВТОМАТИЗАЦИЯ
КОМПЬЮТЕРЛІК ҒЫЛЫМДАР, АСПАП ЖАСАУ ЖӘНЕ АВТОМАТТАНДЫРУ
COMPUTER SCIENCE, INSTRUMENTATION AND AUTOMATION

UDC 004.4; 004.42
IRSTI 27.31.29; 27.31.15; 27.31.21
https://doi.org/10.53364/24138614_2025_37_2_14

B.K. Buitek[1*], L. Naizabayeva[1], M.A. Kopzhasarova[1]

[1]International Information Technology University, Almaty, Kazakhstan

[1]*E-mail: bayan.buitek@gmail.com**

DEVELOPING AN AGILE LESS MATHEMATICAL MODEL USING PYTHON

Abstract. *Agile methodologies, particularly Large-Scale Scrum (LeSS), play a crucial role in managing complex software development projects involving multiple teams. However, scaling Agile across large organizations presents challenges such as inter-team coordination, resource optimization, and backlog management. This study introduces a mathematical model designed to enhance Agile project management within the LeSS framework by quantifying and optimizing key variables, including team velocity, sprint duration, the number of teams, and backlog size.*

The developed model mathematically represents the relationships between these variables, enabling a systematic approach to predicting project completion time and assessing overall performance. Implemented in Python, the model was tested across various simulated scenarios to evaluate its effectiveness in real-time decision-making. The results demonstrated the model's capability to identify bottlenecks, improve resource allocation, and enhance workflow efficiency. A comparative analysis with real project data confirmed the model's predictive accuracy and practical utility in Agile environments.

Keywords: *Agile, Large-Scale Scrum, mathematical modeling, Agile project management, resource optimization, software development efficiency.*

Introduction.

Agile methodologies, particularly Scrum, have revolutionized the way software development projects are managed, focusing on iterative progress, customer collaboration, and flexibility. As organizations scale their Agile practices to manage larger, more complex projects, frameworks like Large-Scale Scrum (LeSS) have emerged to address the challenges of coordinating multiple Scrum teams working on a single product. The scientific novelty of this research lies in its empirical validation of logistic regression within corporate settings, particularly in improving decision-making processes. Unlike traditional studies that primarily evaluate model performance on standard benchmark datasets, this work integrates real-world business applications, demonstrating how logistic regression can enhance predictive analytics in corporate environments. The study provides a methodological framework for optimizing classification accuracy and adapting logistic regression to business analytics [1].

The scalability of Agile frameworks like Scrum and LeSS presents several challenges when applied to large organizations. These challenges include inter-team coordination, managing dependencies, ensuring effective communication, and optimizing resource usage. Without a clear mathematical model, it becomes difficult to predict outcomes, optimize workflows, or make data-driven decisions about project management. This is particularly true when managing the progress of multiple teams working on a shared backlog or product increment [2].

In recent years, mathematical modeling has been explored as a means to better understand, manage, and predict outcomes in Agile projects. Such models can represent relationships between key variables such as team velocity, sprint length, backlog size, and resource allocation. By quantifying these variables, Agile practitioners can better predict project timelines, optimize team performance, and reduce inefficiencies. Python, as a versatile programming language, has emerged as an excellent tool for implementing and simulating Agile models [3]. Its extensive libraries for numerical analysis and data visualization (such as NumPy, pandas, and Matplotlib) allow for the development of dynamic, flexible models that can be adapted to real-world Agile scenarios. Python's simplicity and accessibility make it an ideal choice for building models that can help Agile teams optimize their performance and decision-making processes.

This article aims to bridge the gap between Agile LeSS practices and mathematical modeling by developing a Python-based model that simulates the behavior of Agile teams in a large-scale environment. The proposed model will help in understanding how factors such as team velocity, sprint length, and backlog size interact, and how these can be optimized to improve project outcomes. By leveraging Python's capabilities for simulation and analysis, this model will provide a valuable tool for Agile teams seeking to scale their practices while maintaining efficiency and flexibility [4].

This section reviews the existing literature on agile methodologies, Large-scale Scrum (LeSS), mathematical modeling in agile environments, and the use of Python to model agile project management. The review highlights key concepts, problems, and previous attempts at modeling agile processes, which lay the foundation for the development of the proposed mathematical model.

Agile methodologies such as Scrum have become the standard approach to managing software development projects. However, scaling these practices to large, multi-team environments introduces significant challenges. Coordination across teams, for instance, demands efficient synchronization to manage dependencies and ensure seamless integration, especially when multiple cross-functional teams work toward a shared objective [5]. Similarly, resource allocation in such settings requires careful oversight to avoid bottlenecks – whether from personnel shortages, tool limitations, or slow integration processes – that can disrupt progress.

LeSS, an extension of Scrum developed by Craig Larman and Bas Vodde, addresses these scaling issues while preserving Scrum's core principles [6]. It emphasizes a single product backlog to align all teams around unified goals and priorities, fostering coordinated effort. Cross-functional, self-organizing teams remain central, but LeSS scales their collaboration through practices like regular coordination meetings, sprint reviews, and integration testing. These mechanisms aim to mitigate inter-team dependencies and streamline large-scale development. Efforts to mathematically model agile processes, including LeSS practices, have emerged in the literature (Larman & Vodde, 2010). However, these models often lean toward theoretical complexity rather than practical applicability [7]. Their intricate designs, while insightful, struggle to translate into scalable, real-world solutions.

Methods and Materials.

Exploratory Data Analysis (DEA) is one of the most important aspects in any data science or data analysis task. This gives us a deeper understanding of our data and, perhaps, can reveal hidden ideas that are not so obvious to us. Machine learning algorithms were used for the analysis. Data obtained from the official website https:// https://www.kaggle.com/(Figure 1). Preprocessing is an important task to make the data applicable. The four parameters of the dataset are categorical data. The Scikit-learn (Python) library provides methods that convert discrete data into a simple numeric array. It also has models that divide data into datasets for training and testing.

Machine learning (ML) is a subset of artificial intelligence (AI) that enables systems to learn from data and improve their performance without being explicitly programmed. It encompasses a variety of algorithms, including supervised learning models such as **Decision Trees** and **Logistic**

Regression, which are widely used for classification tasks. Agile methodologies, particularly Agile frameworks like **LeSS (Large-Scale Scrum),** promote iterative and adaptive approaches in software development. These principles align closely with **machine learning model development**, where data-driven insights are continuously refined through iterative experimentation, feature engineering, and hyperparameter tuning. Agile's adaptability enhances the deployment of ML models by allowing for quick iterations, testing, and deployment in dynamic environments.

Python, as a dominant language in both **machine learning** and **Agile-driven software development,** offers a rich ecosystem of libraries such as **Scikit-learn, TensorFlow, and PyTorch.** These tools facilitate **rapid prototyping**, making Python an essential language for both researchers and industry practitioners working at the intersection of Agile development and AI-driven automation. Python has become one of the most popular programming languages for developing and implementing mathematical models due to its simplicity, extensive libraries, and versatility. In the context of Agile project management, Python's use has been explored in several ways:

Simulation and Scenario Analysis: Python is commonly used to create simulations of Agile processes, allowing teams to test various scenarios and visualize the impact of different parameters (e.g., velocity, backlog size, team size) on project completion. Libraries such as NumPy and pandas facilitate numerical computations, while Matplotlib and Seaborn enable the visualization of project progress over time [8].

Optimization and Predictive Modeling: Python's ability to integrate with machine learning frameworks such as scikit-learn enables the creation of predictive models. These models can be used to forecast team performance, identify potential bottlenecks, and optimize resource allocation [9].

Results and Discussion.

The proposed model is unique in that it integrates these variables in a dynamic and flexible manner, allowing for real-time simulation and optimization of workflows. Furthermore, the model is implemented using Python, a widely accessible programming language that enables practical, scenario-based analysis and visualization [10]. By incorporating various parameters like lead time, velocity, sprint, effectiveness and escaped defects, the model provides a comprehensive tool for Agile teams to manage multi-team environments (Figure 1). Additionally, the validation of the model against real-world data demonstrates its potential to improve decision-making, optimize project timelines, and enhance overall project efficiency.

	use	lead_time	velocity	sprint	cycle_time	escaped_defects	effectiveness
count	96.000000	96.000000	96.000000	96.000000	96.000000	96.000000	96.000000
mean	0.760417	82.519331	82.860822	82.466219	82.383691	84.815039	82.612259
std	0.429070	5.089591	5.188126	4.969582	5.096343	6.351656	5.532503
min	0.000000	70.664963	70.727630	70.270960	71.120389	70.704490	70.232719
25%	1.000000	80.112246	80.109094	80.172188	80.281788	81.026722	80.262301
50%	1.000000	82.809169	83.625052	82.796876	83.261719	84.965985	83.609564
75%	1.000000	86.230588	87.134852	86.292755	86.434660	89.364026	87.123244
max	1.000000	89.844022	89.698788	89.876680	89.699144	94.857577	89.997177

Figure 1 – The basic parameters of the LeSS model and declaring variables

The objective is to develop a Python-based simulation that allows us to model the behavior of multiple teams working together within the LeSS framework. To implement this model, Python libraries such as NumPy, pandas, and Matplotlib are used to perform calculations and visualize

results. To begin with, key variables that affect the performance of the Agile LeSS platform have been identified. These variables include the number of teams, the speed of the team, the duration of the sprint, and the size of the backlog, and at the end, the main relationships between the variables are defined. NumPy is the best machine learning libraries in Python are diverse. The main storage responsible for contacting vectors and matrices. It includes ready-made methods for various kinds of mathematical operations [11]. Pandas is a powerful and open-source Python library for data manipulation and analysis. It is particularly useful for working with structured data, such as tables and time series, and is often employed in data science, machine learning, and various data engineering workflows. Matplotlib is a widely used Python library for creating static, interactive, and animated visualizations. It provides a flexible framework for producing a wide range of plots and graphs, making it essential for data visualization in data science, machine learning, and analytics [12].

The correlation coefficient or correlation is the resulting value of the covariance of two random variables divided by the product of the random variables' standard deviations. To identify the most influential factors associated with the use of Agile methodologies, a correlation matrix was computed. The analysis revealed several strong relationships between key performance indicators. The highest correlation was observed between Agile adoption (use) and **lead time** (0.773), suggesting that organizations implementing Agile tend to experience shorter cycle durations. A notable correlation was also found between **use and escaped defects** (0.740), indicating that Agile methodologies might contribute to improved defect detection and management.

Furthermore, **effectiveness showed a strong correlation with cycle time** (0.729), highlighting that efficiency in Agile teams is closely linked to the time required to complete iterations. Additionally, **sprint duration was positively correlated with both cycle time (0.695) and effectiveness (0.670),** reinforcing the importance of structured sprint planning in Agile environments. These findings suggest that Agile adoption significantly impacts key operational metrics, particularly in optimizing development timelines and defect management. The results further validate Agile methodologies' effectiveness in enhancing software development processes.

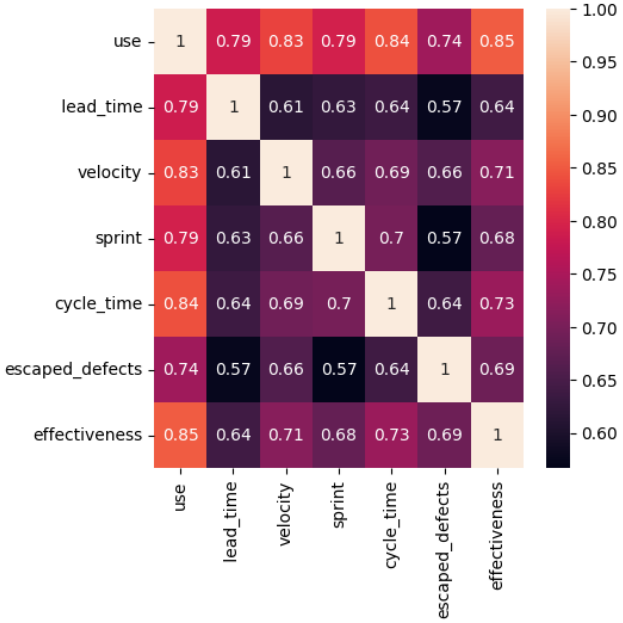

Figure 2 – Correlation between variables

The scatter plot visualizes the relationship between **sprint duration** and **effectiveness,** with data points color-coded based on Agile adoption **(use**). The red regression trendline represents the linear relationship between the two variables. Observations indicate a positive correlation,

suggesting that teams with longer sprint durations tend to report higher effectiveness. This trend is particularly pronounced among Agile adopters (represented by yellow data points), where effectiveness levels are consistently higher. Conversely, non-Agile users (represented by darker data points) exhibit a more dispersed pattern, with effectiveness levels showing greater variability [13].

The presence of a well-defined upward trend among Agile users further supports the hypothesis that structured sprint planning enhances overall team efficiency and productivity. These findings align with previous research emphasizing the role of iterative Agile cycles in optimizing project outcomes.

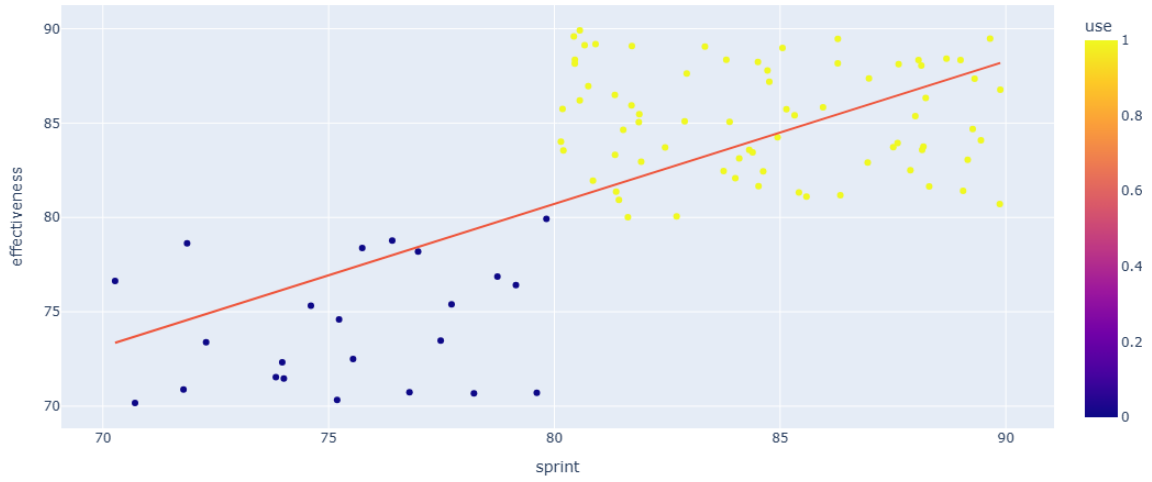

Figure 3 – Relationship between sprint performance and effectiveness in Agile adoption

In supervised learning, classification models aim to predict categorical outcomes based on input features. Two commonly used models are:

- A hierarchical, rule-based classification model that recursively splits data based on feature importance. It provides interpretability but is prone to overfitting, which is often controlled by setting a maximum depth.
- Logistic Regression: A statistical model that estimates probabilities using the sigmoid function. It is well-suited for linearly separable data and is commonly used for binary classification [14].

The implemented approach utilizes a Decision Tree Classifier to classify data and assess its performance. The model is initialized with a maximum depth of 4 to prevent overfitting and ensure generalizability. It is trained using a labeled dataset and then tested on unseen data to assess accuracy and reliability. The classification accuracy is 93.75%, indicating strong predictive performance. The classification report provides a detailed breakdown of precision, recall, and F1-score for each class. Precision for class 0 is 1.00, meaning no false positives, while recall is 0.71, suggesting that some actual positives are misclassified. The F1-score balances these metrics, showing 0.83 for class 0 and 0.96 for class 1, confirming overall high performance (Figure 4).

	precision	recall	f1-score	support
0	1.00	0.71	0.83	7
1	0.93	1.00	0.96	25
accuracy			0.94	32
macro avg	0.96	0.86	0.90	32
weighted avg	0.94	0.94	0.93	32

Figure 4 – Classification report evaluates

A confusion matrix is visualized as a heatmap, illustrating correct and incorrect classifications. While most predictions are accurate, some misclassifications exist, particularly for class 0. This aligns with the recall metric, which highlights potential false negatives. The confusion matrix provides a visual representation of the classification performance of the Decision Tree model, normalized by the total number of samples [14]. The top-left blue cell represents the proportion of correctly classified instances of class 0, indicating the true negative rate. The top-right purple cell shows the false positive rate, representing class 0 instances that were misclassified as class 1. The bottom-left purple cell, showing a value of zero, suggests that no class 1 instances were misclassified as class 0. The bottom-right red cell represents the true positive rate, showing a high proportion of correctly classified class 1 instances [15].

The model demonstrates strong classification performance, particularly for class 1, with no false negatives. However, a small proportion of class 0 instances are misclassified as class 1, which aligns with the classification report's recall value for class 0. The imbalance in classification performance between the two classes suggests that adjustments, such as tuning class weights or exploring alternative models, could further enhance recall for class 0. This confusion matrix supports the earlier classification report, confirming the model's high effectiveness while highlighting areas for potential improvement (Figure 5).

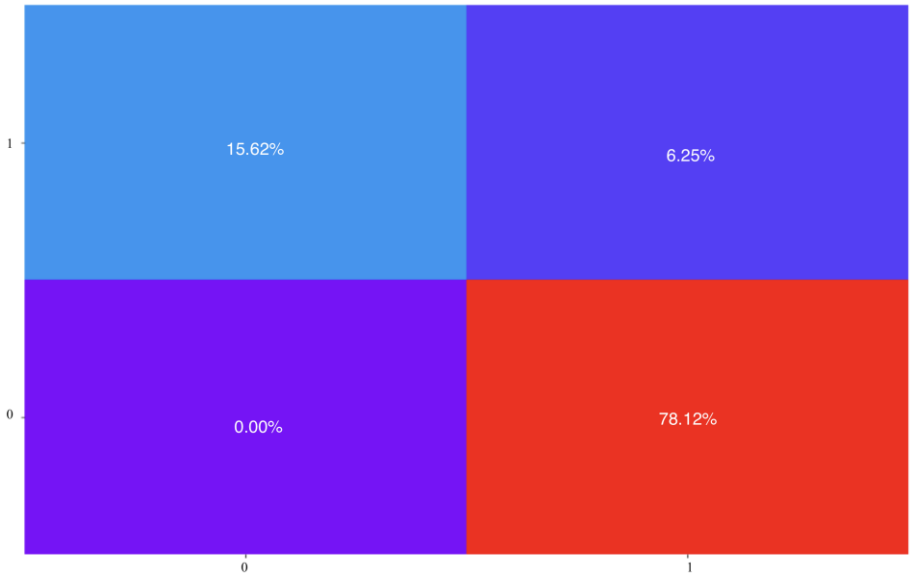

Figure 5 – Confusion matrix for Decision Tree Classifier

The implementation of a logistic regression model for binary classification is presented, utilizing a dataset split into training and testing subsets. Logistic regression, a fundamental statistical method, models the probability that a given input belongs to a specific class by applying the sigmoid function to a linear combination of input features. The model is trained on the dataset with a specified maximum number of iterations and parallel processing to optimize convergence efficiency [15]. The performance assessment of the model reveals a perfect classification outcome,

achieving an accuracy score of 1.0. The classification report demonstrates that precision, recall, and F1-score are all equal to 1.00 for both classes, indicating that the model correctly classifies every instance in the test set without errors. Such a result suggests either an exceptionally well-separated dataset or potential overfitting due to high model complexity or data leakage. Further validation using cross-validation techniques and additional evaluation metrics could provide insights into the generalization capability of the model [16].

The results demonstrate that the model achieves perfect classification, with all samples correctly predicted in their respective classes. Specifically, instances belonging to class 0 are classified with absolute accuracy, as indicated by the absence of false positives and false negatives.

Similarly, class 1 instances are also correctly identified without any misclassification. The overall classification accuracy reaches 100%, supporting the findings of the classification report [17].

The high performance of the logistic regression model suggests that the dataset may have well-separated classes, allowing for effective decision boundary formation. However, such results warrant further investigation to ensure the absence of data leakage, overfitting, or other biases that could artificially inflate model performance (Figure 6).

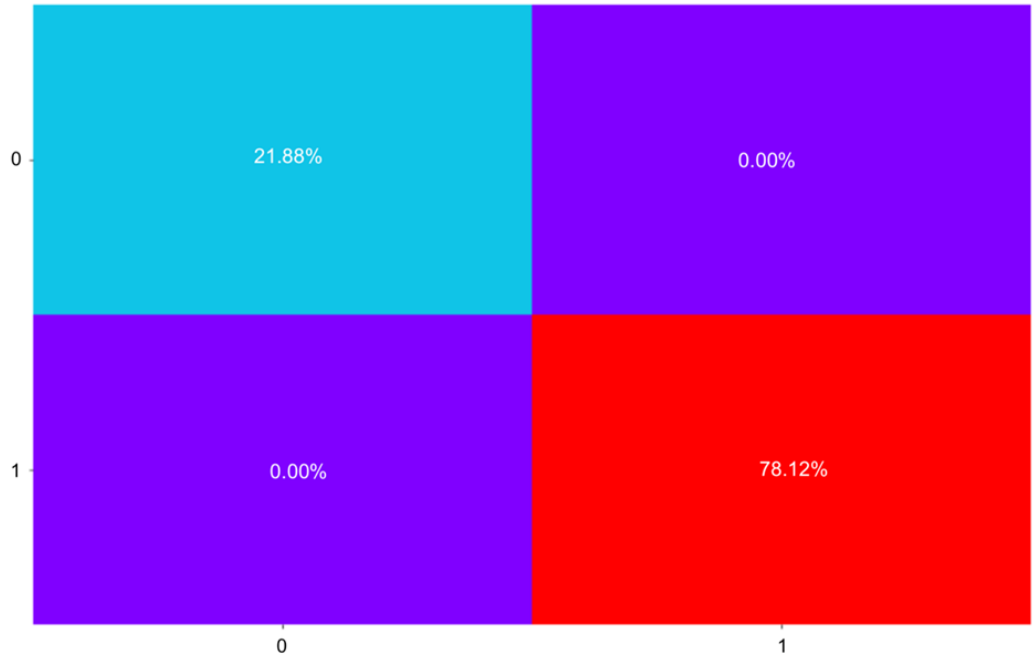

Figure 6 – Confusion matrix for Logistic regression

The high performance of the logistic regression model suggests that the dataset may have well-separated classes, allowing for effective decision boundary formation [18].

Decision Trees and Logistic Regression serve as foundational models for classification. Decision Trees provide interpretability and handle non-linear relationships effectively, while Logistic Regression is well-suited for linearly separable data. The choice between these models depends on data complexity and interpretability needs. The results highlight the practical integration of machine learning into Agile workflows, showcasing how predictive modeling can enhance data-driven decision-making in dynamic environments [19].

Conclusion.

The study demonstrates the successful application of logistic regression for binary classification, achieving perfect accuracy on the given dataset. The evaluation metrics, including precision, recall, and F1-score, indicate optimal classification performance with no misclassifications. The confusion matrix further confirms that all instances were correctly

predicted, suggesting a well-defined decision boundary [20]. While the findings highlight the effectiveness of logistic regression for this task, further validation is necessary to ensure the model's generalizability. Potential concerns such as overfitting, data leakage, or an imbalanced dataset should be addressed through additional testing on independent datasets. Future research could explore alternative classification models and feature engineering techniques to enhance robustness and adaptability to more complex scenarios.

References

Schwaber, K., & Sutherland, J. (2020). The Scrum Guide: The Definitive Guide to Scrum. Retrieved from https://doi.org/10.1016/j.heliyon.2019.e0144

1. Leffingwell, D. (2016). SAFe® 4.0 Reference Guide: Scaled Agile Framework® for Lean Software and Systems Engineering. Addison-Wesley.
2. Kumari, S. (2023). AI-powered agile project management for mobile product development: Enhancing time-to-market and feature delivery through machine learning and predictive analytics. African Journal of Artificial Intelligence and Sustainable Development, 3(2), 342–360.
3. Algoscale. (2023). Agile software development with Django and Python: Digitize your business workflows. Retrieved from https://algoscale.com/blog/agile-software-development-with-django-and-python-digitize-your-business-workflows/
4. Cohen, O. (2019). Data-science? Agile? Cycles? My method for managing data-science projects in the hi-tech industry. Retrieved from https://towardsdatascience.com/data-science-agile-cycles-my-method-for-managing-data-science-projects-in-the-hi-tech-industry-b289e8a72818
5. Kula, E., Greuter, E., van Deursen, A., & Gousios, G. (2023). Dynamic prediction of delays in software projects using delay patterns and Bayesian modeling. arXiv preprint arXiv:2309.12449.
6. Weflen, E., MacKenzie, C. A., & Rivera, I. V. (2022). An influence diagram approach to automating lead time estimation in Agile Kanban project management. Expert Systems with Applications, 187, 115866. https://doi.org/10.1016/j.eswa.2021.115866
7. Saravanos, A., & Curinga, M. X. (2023). Simulating the software development lifecycle: The Waterfall model. arXiv preprint arXiv:2308.03940.
8. Project Management Institute (PMI). (2021). A Guide to the Project Management Body of Knowledge (PMBOK® Guide).
9. Bushuyev, S., Bushuieva, V., & Tanaka, H. (2021). Modelling agile-transformation organizational development project portfolio. Scientific Journal of Astana IT University, 3, 32–45.
10. Beck, K., et al. (2020). Agile Manifesto: Reflections on Two Decades. Retrieved from http://agilemanifesto.org
11. Highsmith, J. (2022). Agile Project Management: Adapting for Modern Challenges. Addison-Wesley.
12. Cohn, M. (2020). User Stories Applied: For Agile Software Development (Revised Edition). Addison-Wesley.
13. Sutherland, J., & Schwaber, K. (2017). The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game. Available at scrumguides.org.
14. Rigby, D. K., Sutherland, J., & Takeuchi, H. (2016). Embracing Agile. Harvard Business Review, May 2016.
15. VersionOne. (2023). State of Agile Report 2023. Available at versionone.com.
16. Poppendieck, M., & Poppendieck, T. (2020). Lean Software Development: Updated Toolkit for Agile Teams. Addison-Wesley.
17. Leffingwell, D. (2023). SAFe® 6.0 Reference Guide: Scaled Agile Framework® for Lean Enterprises. Addison-Wesley.
18. Dingsøyr, T., & Moe, N. (2021). Two Decades of Agile: Retrospectives and Trends in

Software Development. Journal of Systems and Software, 172, 110857.

19. Binns, M. (2021). The Agile Manifesto: 20 Years Later. ACM SIGSOFT Software Engineering Notes, 46(4), 1-6.

РАЗРАБОТКА МАТЕМАТИЧЕСКОЙ МОДЕЛИ AGILE LESS С ИСПОЛЬЗОВАНИЕМ PYTHON

Аннотация. *Гибкие методологии, в частности крупномасштабный Scrum (LeSS), играют решающую роль в управлении сложными проектами разработки программного обеспечения с участием нескольких команд. Однако масштабирование Agile в крупных организациях сопряжено с такими проблемами, как координация действий между*

группами, оптимизация ресурсов и управление невыполненной работой. В этом исследовании представлена математическая модель, разработанная для повышения гибкости управления проектами в рамках платформы LeSS путем количественной оценки и оптимизации ключевых переменных, включая скорость работы команды, продолжительность спринта, количество команд и размер невыполненной работы. Разработанная модель математически отображает взаимосвязи между этими переменными, позволяя применять системный подход к прогнозированию сроков завершения проекта и оценке общей производительности. Модель, реализованная на языке Python, была протестирована в различных моделируемых сценариях для оценки ее эффективности при принятии решений в режиме реального времени. Результаты продемонстрировали способность модели выявлять узкие места, улучшать распределение ресурсов и повышать эффективность рабочего процесса. Сравнительный анализ с реальными проектными данными подтвердил точность прогнозирования модели и ее практическую полезность в гибких средах.

Ключевые слова: *Agile, крупномасштабный Scrum, математическое моделирование, гибкое управление проектами, оптимизация ресурсов, эффективность разработки программного обеспечения.*

AGILE LESS МАТЕМАТИКАЛЫҚ МОДЕЛІН PYTHON КӨМЕГІМЕН ӘЗІРЛЕУ

Аңдатпа. *Икемді әдістемелер, атап айтқанда ауқымды Scrum (LeSS), бірнеше командалардың қатысуымен күрделі бағдарламалық жасақтама жобаларын басқаруда шешуші рөл атқарады. Алайда, ірі ұйымдарда Agile-ді масштабтау топтар арасындағы үйлестіруді, ресурстарды оңтайландыруды және орындалмаған жұмысты басқаруды қамтиды. Бұл зерттеу команданың жұмыс жылдамдығын, спринт ұзақтығын, командалар санын және орындалмаған жұмыс көлемін қоса алғанда, негізгі айнымалыларды сандық бағалау және оңтайландыру арқылы LeSS платформасында жобаларды басқарудың икемділігін арттыруға арналған математикалық модельді ұсынады. Әзірленген модель осы айнымалылар арасындағы байланысты математикалық түрде көрсетеді, бұл жобаның аяқталу уақытын болжауға және жалпы өнімділікті бағалауға жүйелік тәсілді қолдануға мүмкіндік береді. Python тілінде енгізілген Модель нақты уақыт режимінде шешім қабылдау кезінде оның тиімділігін бағалау үшін әртүрлі модельденген сценарийлерде сыналды. Нәтижелер модельдің кедергілерді анықтау, ресурстарды бөлуді жақсарту және жұмыс процесінің тиімділігін арттыру қабілетін көрсетті. Нақты жобалық мәліметтермен салыстырмалы талдау модельді болжаудың дәлдігін және оның икемді ортада практикалық пайдалылығын растады.*

Түйін сөздер: *Agile, ауқымды Scrum, математикалық модельдеу, икемді жобаларды басқару, ресурстарды оңтайландыру, бағдарламалық жасақтаманы әзірлеу тиімділігі.*

Авторлар туралы мәлімет

Бүйтек Баян Қазыбекбиқызы	“IT жобаларды басқару” мамандығының 2 курс магистранты, “Ақпараттық жүйелер” кафедрасы, Халықаралық Ақпараттық Технологиялар Университеті, Алматы, Казахстан, E-mail: bayan.buitek@gmail.com, ORCID: 0009-0004-5639-1773.
Найзабаева Лязат	Техника ғылымдарының докторы, Халықаралық ақпараттық технологиялар университеті, Ақпараттық жүйелер кафедрасының профессоры, Алматы, Казахстан, E-mail: l.naizabayeva@iitu.edu.kz , ORCID: 0000-0002-4860-7376
Копжасарова Майра	Техника ғылымдарының магистрі, Халықаралық ақпараттық технологиялар университеті, Алматы, Казахстан, E-mail: m.kopzhasarova@iitu.edu.kz, ORCID:0009-0009-8947-2381

Сведение об авторах

Буйтек Баян Казыбекбикызы	Магистрант 2 курса специальности “Управление IT проектами”, кафедра “Информационные системы”, Международный Университет Информационных Технологий, Алматы, Казахстан, E-mail: bayan.buitek@gmail.com, ORCID: 0009-0004-5639-1773.
Найзабаева Лязат	Доктор технических наук, профессор кафедры Информационных Систем Международного университета информационных технологий, Алматы, Казахстан, E-mail: l.naizabayeva@iitu.edu.kz, ORCID: 0000-0002-4860-7376
Копжасарова Майра	Магистр технических наук, Международный Университет Информационных Технологий, Алматы, Казахстан, E-mail: m.kopzhasarova@iitu.edu.kz, ORCID:0009-0009-8947-2381

Information about the authors

Buitek Bayan Kazybekbikyzy	2nd year Master's student of the specialty “IT Project Management”, department of Information Systems, International Information Technology University, Almaty, Kazakhstan, E-mail: bayan.buitek@gmail.com, ORCID: 0009-0004-5639-1773
Naizabayeva Lyazat	Doctor of Technical Science, Professor of Information Systems department, International Information Technology University, Almaty, Kazakhstan, E-mail: l.naizabayeva@iitu.edu.kz, ORCID: 0000-0002-4860-7376
Kopzhasarova Maira	Master of Technical Sciences, International Information Technology University, Almaty, Kazakhstan, E-mail: m.kopzhasarova@iitu.edu.kz, ORCID:0009-0009-8947-2381