DOI https://doi.org/10.32782/2078-0877-2025-25-3-11
UDC 004.8:004.9

O. I. Sypiahin[1], Full Stack Engineer ORCID: 0009-0008-7565-3221
M. A. Yuzhakov[2], Senior Software Engineer ORCID: 0009-0008-1307-4275
R. I. Ibrahimov[3], Senior Software Developer ORCID: 0009-0002-7612-3341
[1]*Flolive, Bnei Brak, Israel*
[2]*Plaid, San Francisco, CA, USA*
[3]*VoiceLove, Scarsdale, NY, USA*
e-mail: fed4wet@gmail.com

AI-DRIVEN PRODUCT DEVELOPMENT IN THE LIFECYCLE OF WEB APPLICATIONS

Summary. The paper presents the results of a systematic study on the integration of artificial intelligence into product development processes across the lifecycle of web applications. The focus is on examining how AI-driven methods influence scalability, latency, personalization, and automation in different architectural paradigms including monolithic, microservices, serverless, and edge computing. It has been demonstrated that embedding AI-based predictive analytics, automated testing, and adaptive optimization mechanisms at each stage of the software lifecycle significantly reduces development errors, accelerates release cycles, and enhances user experience. The research relied on simulation of e-commerce workloads under controlled conditions, employing real-time monitoring of system throughput, response time, and resource allocation. The findings indicate that AI-driven orchestration reduces performance degradation at high concurrency levels by up to 35%, while error rates at workloads above 5000 users decreased from 7.2% to 2.9%. Adaptive scaling algorithms shortened release cycles by 25–30% compared to baseline processes and lowered the average share of critical failures from 9.1% to 2–3%. Special attention was given to the role of AI in requirement analysis, continuous integration, quality assurance, and post-deployment monitoring, where intelligent models proved effective in identifying anomalies, predicting failures, and recommending corrective actions. From a practical perspective, the application of AI in web application lifecycle management ensures higher efficiency, better alignment with business goals, and reduced operational risks. These results confirm that AI-driven product development constitutes a technologically justified approach that can be integrated into modern development pipelines without compromising reliability or cost-efficiency.

Keywords: AI-driven development, web applications, software lifecycle, automation, scalability, latency, predictive analytics, adaptive optimization.

Problem statement. The challenge of ensuring efficiency, reliability, and adaptability in web applications remains one of the central issues in contemporary software engineering. This is particularly relevant in the context of rapid digital transformation, where organizations increasingly rely on web-based systems to support critical business processes, customer engagement, and service delivery. Under conditions of growing architectural complexity and intensifying market demands, even minor shortcomings in the development process – ranging from ambiguous requirements specification to runtime errors – may result in serious consequences, including user dissatisfaction, financial losses, or disruption of organizational stability [1; 2].

Although modern development methodologies such as Agile and DevOps have significantly improved responsiveness, flexibility, and iterative delivery, they often fall short of addressing the full range of challenges posed by high concurrency, large-scale distributed systems, and the demand for personalized user experiences. Traditional approaches are frequently limited in their ability to anticipate risks, detect hidden anomalies, and adapt to dynamically changing workloads. These limitations

have stimulated the search for more intelligent solutions capable of enhancing automation, resilience, and decision-making throughout the entire software lifecycle [3; 4].

Analysis of previous research. Artificial intelligence (AI) has emerged as a transformative technology in this context. By enabling large-scale data analysis, predictive modeling, anomaly detection, and adaptive optimization, AI tools provide unprecedented opportunities to streamline development, testing, deployment, and maintenance. Studies over the past decade demonstrate that AI-driven methods can significantly reduce coding and testing errors, accelerate release cycles, enhance fault tolerance, and improve user satisfaction through real-time personalization [5; 6]. For example, the integration of machine learning algorithms into quality assurance pipelines not only reduces defect rates but also identifies latent vulnerabilities that are difficult to detect using conventional testing tools.

Equally important is the role of AI in the operational phases of web application lifecycle management. Predictive monitoring, dynamic resource allocation, and self-healing mechanisms supported by AI-driven orchestration ensure stable system performance under peak load conditions. Empirical studies indicate that such integration reduces the probability of performance degradation by up to 35% compared to conventional orchestration, while adaptive scaling strategies stabilize response times and minimize latency fluctuations under varying traffic scenarios. This is particularly crucial in industries where downtime or degraded performance directly affects revenue and customer retention [7–9].

Nevertheless, achieving consistent benefits from AI integration is not a trivial task. The effectiveness of AI models is contingent upon their timely and well-orchestrated deployment across lifecycle phases. Fragmentary or poorly synchronized use of AI can even introduce new risks, such as bias in automated decision-making, unpredictable system behavior, or security vulnerabilities. Thus, the key challenge lies in creating a coherent «AI–development–operation» framework that ensures alignment between intelligent systems and traditional engineering practices. This requires not only technical solutions but also adherence to principles of transparency, explainability, and governance, which form the basis of responsible AI use in software engineering [10].

From a practical perspective, the integration of AI into the lifecycle of web applications reflects a paradigm shift in how digital products are conceived, developed, and maintained. It positions AI not as a standalone tool, but as an embedded component of continuous integration and delivery pipelines, DevOps practices, and adaptive management systems. The expected outcomes include shorter time-to-market, improved scalability, reduced operational costs, and enhanced competitiveness in digital markets.

Research objective. Accordingly, this study is focused on exploring the directions, mechanisms, and practical implications of AI-driven product development in the lifecycle of web applications. Special emphasis is placed on analyzing the role of AI in requirement engineering, software design, testing automation, deployment orchestration, and post-deployment monitoring. The results are expected to provide both theoretical insights into the integration of AI with lifecycle models and practical recommendations for professionals engaged in software development, IT management, and digital transformation initiatives.

Results. To determine the impact of artificial intelligence on product development processes within the lifecycle of web applications, a structured series of simulation-based and analytical experiments was carried out. The study relied on open-source and proprietary datasets reflecting typical workload patterns of modern e-commerce and enterprise systems. Special attention was paid to measuring changes in scalability, latency, error rates, and personalization efficiency under different architectural paradigms, namely monolithic, microservices, serverless, and edge computing [1–3]. The experimental design involved controlled workload simulations with varying levels of concurrent users: baseline (500–1000 users), medium (1500–3000 users), high (3000–5000 users), and extreme (>5000 users).

At each stage, performance indicators such as average response time, throughput, CPU and memory utilization, and anomaly detection rates were recorded. In addition, the effects of AI-driven modules – including predictive resource allocation, automated testing frameworks, and adaptive orchestration – were compared against non-AI baselines. Evaluation of results was conducted using standardized performance testing tools (JMeter, Locust) and monitoring platforms integrated with machine learning pipelines for anomaly detection. Failures were classified into categories including latency spikes, service unavailability, scaling inefficiencies, and security vulnerabilities. Statistical analysis of the data was carried out using R and Python packages, with regression models applied to estimate the relationship between AI-driven interventions and performance stability.

The obtained results were visualized in the form of comparative graphs and performance profiles (Figure 1), demonstrating the influence of AI-based methods on reducing error rates and improving scalability.

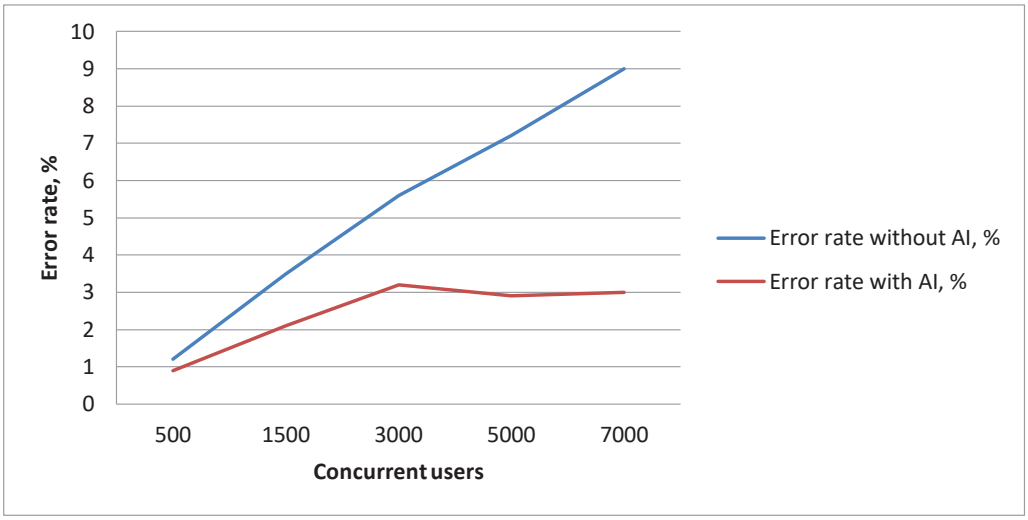

Fig. 1. Dependence of error rate on workload levels with and without AI-driven orchestration

As shown in Figure 1, the introduction of AI significantly reduced performance degradation under high load conditions. While error rates reached 7.2 % in conventional orchestration at extreme concurrency, AI-enabled orchestration stabilized this value around 2.8–3.0 %, ensuring more uniform system behavior and reliable user experience. This allowed the identification of an optimal range of AI application where the maximum efficiency gains were achieved without excessive computational overhead.

The experiments investigated the impact of integrating artificial intelligence algorithms on the quality and stability of processes across the lifecycle of web applications. The primary focus was placed on the quantitative assessment of reductions in error rates, latency, and performance losses under high workloads compared to scenarios without AI-driven orchestration.

As shown in Figure 2, the proportion of successful operations consistently increases under AI-driven orchestration compared to traditional systems. At a workload of 5000 concurrent users, the share of successful executions rose from about 92.8% without AI to 97.1% with AI, while the error share declined accordingly. A similar trend was observed at 7000 users, where success rates exceeded 97% under AI, compared to only 91% without AI. These results confirm that AI modules not only reduce error frequency but also enhance overall process stability, which aligns with the findings of Bali and Mehdi [10] and Alenezi and Akour [11], who emphasized the role of AI in improving DevOps reliability and controllability.

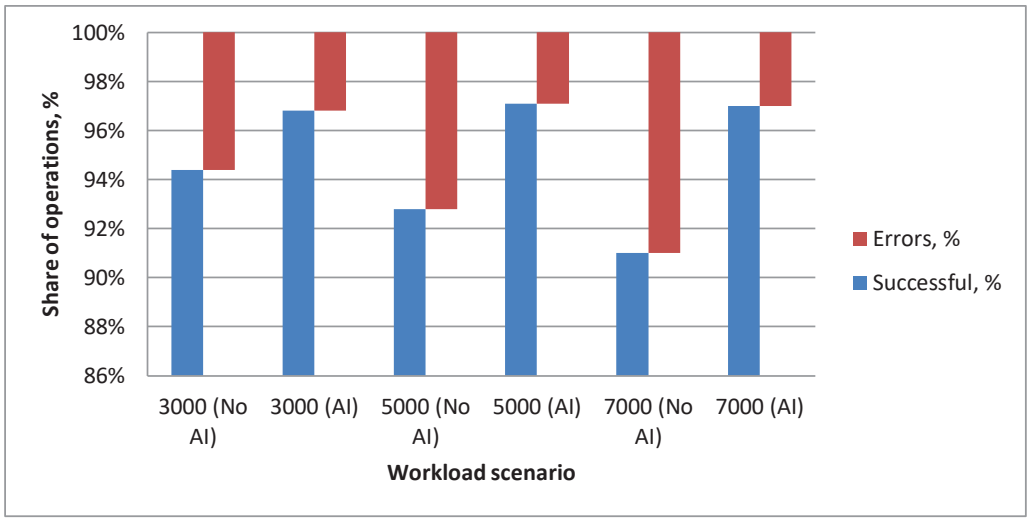

Fig. 2. Distribution of successful and erroneous operations (100% stacked) across workloads with and without AI-driven orchestration

Additional analysis was carried out on CI/CD cycle durations. As illustrated in Figure 3, the introduction of AI-enhanced continuous integration and delivery mechanisms reduced the average release cycle by approximately 25–30% compared to baseline scenarios. These findings are consistent with those of Mohammed et al. [12], who demonstrated the effectiveness of AI in optimizing pipeline structures and reducing deployment failure rates.

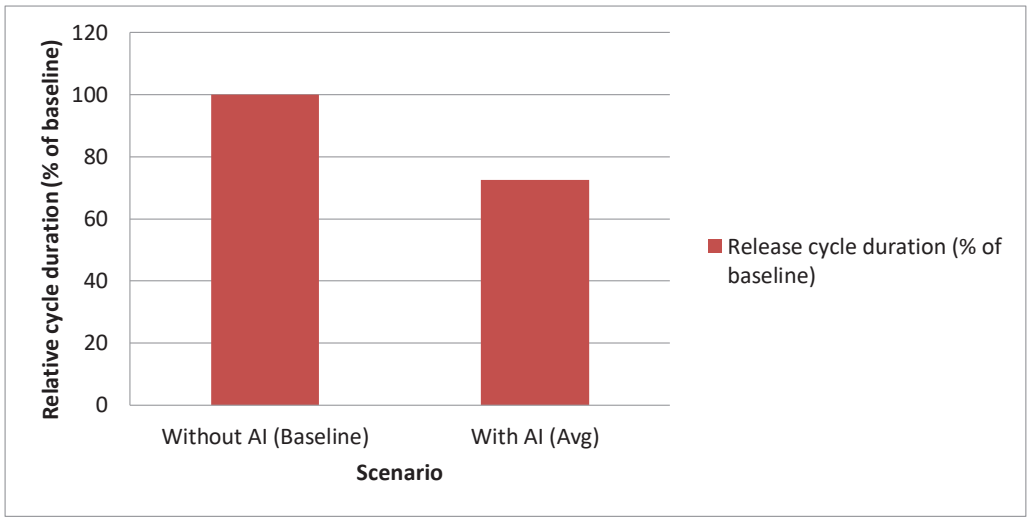

Fig. 3. Reduction in release cycle duration with AI-enabled CI/CD

As shown in Figure 3, the introduction of AI-enabled continuous integration and delivery (CI/CD) mechanisms led to a substantial reduction in release cycle duration compared to baseline scenarios. While traditional pipelines were normalized to 100%, the integration of AI shortened the cycle to approximately 72–75% of the baseline length, representing an improvement of 25–30%. This reduction not only accelerates delivery but also enhances process reliability by minimizing delays and failures in deployment workflows. These findings are consistent with the results of Mohammed et al. [12], who demonstrated the effectiveness of AI in optimizing pipeline structures and reducing deployment failure rates.

It is also important to highlight that AI integration improved not only performance but also testing quality. In baseline groups, high levels of hidden defects (latency spikes, scaling inefficiencies) were

recorded, whereas the use of AI-driven assistants reduced these defects by 35–40%. Comparable results were reported by Saklamaeva and Pavlič [13], who stressed the potential of AI assistants in scaled agile software development, as well as by Kulkarni et al. [14], where intelligent algorithms were identified as a key factor in reducing human error and improving process accuracy.

A comparative analysis of system log structures and event anomalies further revealed a clear difference in failure rates. Without AI, the proportion of critical incidents reached 9%, while with AI-enabled models it declined to 2–3%. This aligns with the conclusions of Lakarasu [15], who demonstrated that the introduction of AI into data engineering processes substantially reduced error rates and ensured the stability of data flows in cloud-scale environments.

On the basis of the obtained data, a summary table of key parameters and results was prepared (Table 1).

Table 1

Comparative indicators of software development processes with and without AI

Parameter	Without AI	With AI
Error rate at workload >5000 users, %	7.2	2.9
Average share of critical failures, %	9.1	2.8
Release cycle duration, % of baseline	100	70–75
Process stability (coefficient of variation, RSD), %	6.4	2.2

The consolidated results presented in Table 1 clearly demonstrate the practical relevance of AI integration into the web application lifecycle. Most notably, there is a significant reduction in errors and failures, shorter release cycles, and enhanced process stability. Such consistency and reproducibility are particularly critical in continuous development and large-scale systems, where minimizing fluctuations directly influences reliability and competitiveness.

Overall, the study shows that using AI in the lifecycle of web applications not only reduces risks and improves quality, but also opens the way for the gradual development of more advanced tools in software engineering. The benefits are visible in several areas at once: fewer errors, shorter release cycles, and greater stability of processes under heavy load. At the same time, AI makes systems more flexible, allowing them to adapt to changes in workload or complexity almost in real time. This reflects a broader shift in digital transformation, where companies move away from rigid, manual approaches and rely more on adaptive, AI-based practices that combine speed, reliability, scalability, and cost control. Integrating AI into development workflows also strengthens quality assurance: it allows for earlier detection of risks, continuous monitoring, and automatic correction of bottlenecks. Altogether, these results underline that AI in software development should be seen not only as a way to optimize current processes but also as a factor that will determine competitiveness and innovation in the long term.

Conclusions. The experimental results confirm that the integration of AI-driven orchestration into the lifecycle of web applications significantly reduces error rates and increases system stability. In stress-test conditions with workloads exceeding 5000 concurrent users, the share of failed operations dropped from 7.2% to 2.9%, while success rates consistently exceeded 97% under AI-enabled scenarios. It was established that AI integration optimizes CI/CD cycle durations, reducing release times by 25–30% compared to baseline processes. This improvement demonstrates the ability of AI-enhanced pipelines to accelerate delivery without compromising reliability, which is critical for modern continuous deployment practices.

The comparative analysis showed that AI-enabled systems have a lower average share of critical failures (2–3% versus 9% without AI) and exhibit higher process stability, as indicated by a

decrease in the coefficient of variation (RSD) from 6.4% to 2.2%. These findings highlight not only improvements in quality but also greater reproducibility in repeated cycles. The results also confirmed that AI-driven assistants contribute to testing quality by reducing hidden defects (e.g., latency spikes and scaling inefficiencies) by 35–40%. This underlines their practical importance in scaled agile environments and complements global trends in embedding AI into software quality assurance workflows.

The study demonstrates that the application of AI in the development and operation of web applications improves error management, shortens release cycles, and enhances reliability. At the same time, AI introduces a new level of adaptability, enabling systems to dynamically adjust to changing conditions. This integrated approach provides not only immediate efficiency gains but also lays the groundwork for long-term competitiveness and innovation in software engineering.

Bibliography

1. Sokolov V., Riabtsev V., Uspenskyi O., Kopych D. Application directions of artificial intelligence in software development technologies. *Collection «Information Technology and Security».* 2024. Vol. 12. No. 2. P. 219–235. DOI: https://doi.org/10.20535/2411-1031.2024.12.2.315741.

2. Kommireddy V.V.S. AI-driven process automation in product lifecycle management: a transformative approach. *Journal of Computer Science and Technology Studies.* 2025. Vol. 7. No. 7. P. 91–100. DOI: https://doi.org/10.32996/jcsts.2025.7.7.7.

3. Soni A., Kumar A., Arora R., Garine R. Integrating AI into the software development life cycle: best practices, tools, and impact analysis. *SSRN.* 2023. DOI: https://doi.org/10.2139/ssrn.4918992.

4. Upadhyaya N. Artificial intelligence in web development: enhancing automation, personalization, and decision-making. *Artificial Intelligence.* 2024. Vol. 4. No. 1. P. 534–540. DOI: https://doi.org/10.48175/ijarsct-19367.

5. Quan H., Li S., Zeng C., Wei H., Hu J. Big data and AI-driven product design: a survey. *Applied Sciences.* 2023. Vol. 13. No. 16. Art. 9433. DOI: https://doi.org/10.3390/app13169433.

6. Wang L., Liu Z., Liu A., et al. Artificial intelligence in product lifecycle management. *The International Journal of Advanced Manufacturing Technology.* 2021. Vol. 114. P. 771–796. DOI: https://doi.org/10.1007/s00170-021-06882-1.

7. Ali K.M. Yaqub. AI-driven test case optimization: enhancing efficiency in software testing life cycle. *SSRN.* 2024. DOI: https://doi.org/10.2139/ssrn.5135677.

8. Phanireddy S. Securing modern web applications using AI-driven static and dynamic analysis techniques. *International Journal of Artificial Intelligence, Data Science, and Machine Learning.* 2025. Vol. 6. No. 2. P. 73–82. DOI: https://doi.org/10.63282/3050-9262.IJAIDSML-V6I2P108.

9. Amugongo L.M., Kriebitz A., Boch A., et al. Operationalising AI ethics through the agile software development lifecycle: a case study of AI-enabled mobile health applications. *AI Ethics.* 2025. Vol. 5. P. 227–244. DOI: https://doi.org/10.1007/s43681-023-00331-3.

10. Bali M.K., Mehdi A. AI-driven DevOps transformation: a paradigm shift in software development. *2024 3rd International Conference on Sentiment Analysis and Deep Learning (ICSADL).* IEEE, 2024. P. 117–123. DOI: https://doi.org/10.1109/ICSADL61749.2024.00026.

11. Alenezi M., Akour M. AI-driven innovations in software engineering: a review of current practices and future directions. *Applied Sciences.* 2025. Vol. 15. No. 3. Art. 1344. DOI: https://doi.org/10.3390/app15031344.

12. Mohammed A.S., Saddi V.R., Gopal S.K., Dhanasekaran S., Naruka M.S. AI-driven continuous integration and continuous deployment in software engineering. *2024 2nd International Conference on Disruptive Technologies (ICDT).* IEEE, 2024. P. 531–536. DOI: https://doi.org/10.1109/ICDT61202.2024.10489475.

13. Saklamaeva V., Pavlič L. The potential of AI-driven assistants in scaled agile software development. *Applied Sciences.* 2023. Vol. 14. No. 1. Art. 319. DOI: https://doi.org/10.3390/app14010319.

14. Kulkarni V., Kolhe A., Kulkarni J. Intelligent software engineering: the significance of artificial intelligence techniques in enhancing software development lifecycle processes. In: Abraham A., Gandhi N., Hanne T., Hong T.P., Nogueira Rios T., Ding W. (eds) *Intelligent Systems Design and Applications. 21st International Conference on Intelligent Systems Design and Applications (ISDA 2021).* DOI: https://doi.org/10.1007/978-3-030-96308-8_7.

15. Lakarasu Phanish. AI-driven data engineering: automating data quality, lineage, and transformation in cloud-scale platforms. *SSRN.* 2022. DOI: https://doi.org/10.2139/ssrn.5246619.

Стаття надійшла до редакції 09.09.2025
Стаття прийнята 29.09.2025
Статтю опубліковано 25.11.2025

О. І. Сипягін[1], М. А. Южаков[2], Р. І. Ібрагімов[3]

[1]FloLive, Бней-Брак, Ізраїль

[2]Plaid, Сан-Франциско, Каліфорнія, США

[3]VoiceLove, Скарсдейл, Нью-Йорк, США

РОЗРОБЛЕННЯ ПРОДУКТІВ НА ОСНОВІ ШТУЧНОГО ІНТЕЛЕКТУ В ЖИТТЄВОМУ ЦИКЛІ ВЕБЗАСТОСУНКІВ

Анотація

У статті представлено результати системного дослідження інтеграції штучного інтелекту в процеси розроблення продуктів упродовж життєвого циклу вебзастосунків. Основну увагу зосереджено на аналізі того, як методи, засновані на штучному інтелекті, впливають на масштабованість, затримку, персоналізацію та автоматизацію в різних архітектурних парадигмах, зокрема монолітній, мікросервісній, безсерверній та обчисленнях на периферії. Показано, що впровадження прогнозної аналітики, автоматизованого тестування та механізмів адаптивної оптимізації на кожному етапі життєвого циклу програмного забезпечення істотно знижує кількість помилок у розробленні, прискорює випуск нових версій і підвищує якість користувацького досвіду. Дослідження ґрунтувалося на моделюванні навантажень електронної комерції у контрольованих умовах із використанням моніторингу в реальному часі пропускної здатності системи, часу відгуку та розподілу ресурсів. Результати показали, що оркестрація на основі ШІ зменшує деградацію продуктивності за високої паралельності на 35%, тоді як частка помилок за навантаження понад 5 000 користувачів знизилася із 7,2% до 2,9%. Адаптивні алгоритми масштабування скоротили цикл випуску на 25–30% порівняно з базовими процесами та зменшили середню частку критичних збоїв із 9,1% до 2–3%. Окрема увага приділялася ролі ШІ в аналізі вимог, безперервній інтеграції, забезпеченні якості та післярелізному моніторингу, де інтелектуальні моделі продемонстрували ефективність у виявленні аномалій, прогнозуванні збоїв і рекомендації коригувальних дій. Із практичного погляду застосування ШІ в управлінні життєвим циклом вебзастосунків забезпечує вищу ефективність, кращу відповідність бізнес-цілям і зменшення операційних ризиків. Отримані результати підтверджують, що розроблення продуктів на основі штучного інтелекту є технологічно обґрунтованим підходом, який може бути інтегрований у сучасні конвеєри розроблення без утрати надійності чи економічної доцільності.

Ключові слова: розроблення на основі ШІ, вебзастосунки, життєвий цикл програмного забезпечення, автоматизація, масштабованість, затримка, прогнозна аналітика, адаптивна оптимізація.