Universal Library of Engineering Technology Research Article

ISSN: 3064-996X | Volume 2, Issue 3

Open Access | PP: 81-88

DOI: https://doi.org/10.70315/uloap.ulete.2025.0203015

Platform as a Product: How Turning Internal Platform Solutions into a Product Increases the Engagement of External Teams

Ruslan Tsyganok

Head of Development Team, ecom.tech.

Abstract

Against the backdrop of rapidly increasing software development complexity and widespread cloud migration, organizations face a productivity slump: engineers are compelled to spend substantial resources on infrastructure and operational tasks. This article examines the concept of Platform as a Product (PaaP) as a strategic response to this contradiction. The aim of the study is to build an integrated model of how a product approach to internal platforms increases the engagement and performance of external development teams. The methodological foundation includes a systematic review of academic publications (IEEE, ACM) and a content analysis of industry reports (Gartner, Google Cloud, Cortex) and practical cases (Spotify, Netflix, Zalando). The data obtained show that PaaP, by consistently applying the principles of customer centricity and UX design to internal developer users, specifically reduces their cognitive load. This, in turn, improves key dimensions of Developer Experience (DevEx) — satisfaction, flow state, and efficiency — which leads to increased voluntary engagement and accelerated value delivery. The conclusion confirms the proposed hypothesis and offers practical recommendations for technical leaders on implementing PaaP. The material will be useful to researchers in software engineering, as well as to executives responsible for technology strategy and organizational design in IT companies.

Keywords: *Platform as a Product, Platform Engineering, Developer Experience, Cognitive Load, Developer Productivity, Dora Metrics, Team Topologies, Conway's Law, Internal Developer Platforms, Product Management.*

INTRODUCTION

The modern digital economy is characterized by rapid technological dynamics and the increasing complexity of software landscapes. In such an environment, an organization's ability to design and deliver software quickly and reliably becomes a source of competitive advantage. Platform engineering addresses this challenge and is classified by leading analytical centers as a key strategic technology trend for 2024–2025 [1]. According to Gartner, by 2026, 80% of large software engineering organizations will have formed specialized platform teams that provide internal services, tools, and components to standardize and accelerate development [2].

Empirical evidence confirms the scale and persistence of this trend. A 2024 study by Google Cloud and Enterprise Strategy Group (ESG) recorded that 55% of organizations have already adopted platform engineering practices, and 90% of them intend to expand the corresponding initiatives [1]. These investments are unfolding against the backdrop of an overall market expansion: the forecast for global spending on software development in 2024 suggests an increase of 11,3%, and the custom development segment is estimated at 43,16 billion dollars [4]. At the same time, the problem of developer productivity is intensifying: according to an Atlassian report, engineers lose more than eight hours per week due to inefficiencies, including technical debt and insufficient documentation [6]. The Cortex State of Developer Productivity 2024 study shows that 58% of respondents estimate weekly losses of more than five hours per developer; the key causes are the need to spend extended time assembling project context and waiting for approvals [7]. Consequently, the widespread adoption of platform engineering is not a transient fashion but a strategic response to a measurable productivity crisis caused by the increasing complexity and fragmentation of development processes.

Despite the widespread popularity of the Platform as a Product (PaaP) approach as the most mature trajectory for the development of platform engineering [8], a noticeable gap remains in the academic discourse. Existing studies predominantly document the qualitative advantages of PaaP, but do not offer a systematic model that reveals the mechanisms by which this approach influences the

Citation: Ruslan Tsyganok, "Platform as a Product: How Turning Internal Platform Solutions into a Product Increases the Engagement of External Teams", Universal Library of Engineering Technology, 2025; 2(3): 81-88. DOI: https://doi.org/10.70315/uloap.ulete.2025.0203015.

engagement and effectiveness of developers external to the platform team. The linkage of PaaP with fundamental sociotechnical theories and contemporary frameworks for assessing Developer Experience, DevEx, is often absent, which leaves open the question of why and how exactly the product approach delivers its effect.

The author's hypothesis is that transforming an internal platform into a product — when oriented toward the user, with a clear value proposition and continuous feedback — directly reduces the cognitive load on developer teams. This leads to improvements in key DevEx metrics (satisfaction, flow state, feedback speed), which in turn strengthens voluntary engagement, increases productivity, and accelerates the delivery of business value.

The aim of the study is to construct a holistic model of how the product approach to internal platforms increases the engagement and effectiveness of external developer teams.

The scientific novelty lies in synthesizing PaaP with sociotechnical theories (Conway's Law, cognitive load, team topologies) and contemporary performance assessment frameworks (SPACE, DevEx) to explain the mechanisms of growth in developer engagement.

MATERIALS AND METHODS

Modern research in the field of platform engineering and the development of internal platforms as a product brings together several scientific and applied strands: the theory of product platform design, issues of improving developer productivity, product lifecycle management practices, as well as empirical studies and reports by industry leaders. For analytical consideration, these sources are reasonably grouped into four conceptual blocks: the conceptual and methodological foundations of the platform approach; studies of developer experience (DX) and productivity; applied cases and industry analytics; interdisciplinary extensions of the platform paradigm.

In the first group, a number of works are devoted to the systematization of approaches to the design and management of platforms. Thus, Bortolini M. et al. [3] propose a two-stage methodology for designing and evaluating product platforms under conditions of high production variability, which lays the groundwork for transferring these principles to the software industry. Stark J. [19], within the classical PLM paradigm, emphasizes strategic product lifecycle management, highlighting the role of platforms as a linking element between engineering and business objectives. Similarly, Abate E. et al. [20] consider digital platforms in the context of the industrial Internet of Things, demonstrating the strategic nature of platform logic.

The bibliometric analysis by Zhang X., Yang Y., Chen Y. [10] shows the historical formation of the ecosystem approach, revealing the relationship between platforms and business ecosystems. These conceptual works set the frame within which internal platform solutions can be considered as products that create value not only within the organization but also for external participants.

In the second direction, DX studies focus on the fact that turning a platform into an internal product is directly connected to the developer experience. Aune A. A. W. [11], in an empirical study, shows that the successful implementation of internal developer platforms depends on cultural and organizational factors. Razzaq A. et al. [13] systematize factors affecting productivity through improvements in DX, identifying practices that can increase team engagement. Lopes J. G. M., Oliveira J., Figueiredo E. [14] draw attention to the relationship between DX and code quality, expanding the understanding of the business value of internal platforms. These findings resonate with industry reports on the state of developer productivity (for example, Cortex [7]) and platform engineering (Google [1]; Gartner [2]; Gitpod [15]). These publications demonstrate a consensus that internal platforms, considered as products, reduce cognitive load and risks while simultaneously strengthening team autonomy.

The practical dimension of the topic is presented through case studies and industry analysis. For example, Humanitec [16] describes the role of internal platform teams as a product to ensure standardization and accelerate processes. The Netflix case [17] illustrates how the Gradle platform becomes a strategic asset that increases delivery speed and quality. Leaddev [6] examines the dilemma of allocating investments among AI, DX, and platform engineering, emphasizing the importance of balance. Statistical reports on trends in software development (Vrinsoft [4]; Itransition [5] Conf.researchr [12]) record a growing demand for platform solutions oriented toward a product approach, which is corroborated by rising interest in platform teams and DevEx practices.

The final group comprises interdisciplinary extensions of the platform paradigm. Several studies demonstrate the transfer of platform engineering principles to adjacent disciplines. For example, Karamthulla M. J., Malaiyappan J. N. A., Prakash S. [8] investigate AI-based self-healing systems in the context of platform fault tolerance, which underscores autonomy as a characteristic of a product platform. The literature on analytic methods (Allsop D. B. et al. [18]) points to the growing importance of qualitative studies for assessing developers' perception of platforms. Even work outside the immediate field — for instance, a review of liposomes as drug-delivery systems (Liu P., Chen G., Zhang J. [9]) — shows that the idea of a platform as a reproducible yet customizable solution has universal significance across industries.

The literature review demonstrates consistency in that transforming internal platforms into products contributes to reducing coordination costs, increasing developer engagement, and accelerating value delivery. However, there are contradictions:

- conceptual works [3, 19, 20] emphasize the strategic and systemic role of platforms, whereas empirical studies [11, 13, 14] focus on the operational and organizational levels;

- industry reports [1, 2, 15, 16] portray a more optimistic picture of the adoption of platform practices, whereas academic research points to substantial cultural and technical barriers;

- most publications center on the developer experience, whereas issues of economic efficiency, metrics for return on investment in the platform as a product, and long-term ecosystem governance remain underexplored.

Thus, the literature has not comprehensively addressed the problems of strategic valuation of internal platforms as products, nor the methods for integrating platform teams into broader business ecosystems. These problems open avenues for future research at the intersection of engineering, management, and organizational theory.

RESULTS AND DISCUSSION

Traditionally, internal technology platforms were treated as infrastructure initiatives or toolkits created to solve narrow technical tasks. The Platform as a Product (PaaP) approach radically reinterprets this stance. A product is understood as any outcome of activity offered to the market to satisfy a need, regardless of its tangible or intangible nature [19]. A platform, in turn, is understood as a product, service, or technology on the basis of which external actors (innovators, developers) create complementary solutions [20]. Consequently, PaaP is the transfer of the principles and practices of product management to the development of internal technology platforms, where the role of clients is played by internal development teams [18].

The explanation of the effectiveness of this approach rests on fundamental sociotechnical theories that link organizational structure, communication, and the architecture of technologies. Conway's law and the reverse Conway maneuver. Formulated by Melvin Conway in 1968, the principle states that the architecture of the systems being built reproduces the communication structure of the organization that designs them [3, 4]. If a company is dominated by isolated functional units (for example, database, operations, and development teams), then the product architecture inherits these gaps, turning any end-to-end change into a slow and labor-intensive approval procedure. PaaP materializes the reverse Conway maneuver: the organization deliberately constructs communication pathways by forming a platform team that provides other teams with clearly defined, documented interfaces (APIs, self-service portals, tools). As a result, a desired loosely coupled architecture emerges, in which consuming teams work autonomously and at high speed, without delving into the implementation details of the platform.

Next, we turn to the theory of cognitive load. This concept in cognitive psychology posits the limited capacity of working memory and, accordingly, the amount of mental effort that can be processed simultaneously. In software development, three types of load are distinguished: intrinsic (related to the complexity of the domain task), germane (directed at meaningful problem solving), and extraneous (caused by tools, processes, and environment). PaaP is aimed at radically reducing extraneous load: by providing ready-made, reliable, and convenient means for deployment, monitoring, testing, etc., the platform frees the cognitive resources of developers for creating business value.

Speaking of team topologies, in the model of Matthew Skelton and Manuel Pais four basic types of engineering teams are distinguished, among which the Platform Team plays a system-forming role. Its purpose is to reduce the cognitive load of other teams, primarily Stream-aligned Teams focused on a specific stream of business value. This is achieved by providing the internal platform as an easily consumable service (an X-as-a-Service interaction model).

The synthesis of these ideas suggests that PaaP is not merely a method for creating internal tools, but a purposeful organizational design strategy that uses the platform as the primary lever for optimizing the company's entire sociotechnical system. Instead of reactive bespoke production of tools on request, the organization proactively designs the platform as a product, shaping desired communication patterns, reducing inter-team dependencies, and thereby increasing overall efficiency. This paradigm shift is clearly reflected in Table 1.

Table 1. Comparison of traditional and product approaches to internal platforms (compiled by the author based on [9, 11, 16, 18]).

Criterion	Traditional approach (Platform as a project/tool)	Product approach (Platform as a product)
Primary objective	Meeting a set of requirements, centralization of infrastructure.	Increasing developer productivity and satisfaction, accelerating value delivery.
Users	Captive audience, required to use the platform.	Customers who must be attracted and retained by the product's quality and usability.
Success metrics	Uptime, cost, SLA compliance.	DORA metrics, adoption rate, developer satisfaction (NPS).

Team	Predominantly infrastructure engineers (Operations).	A balanced team with competencies in Product Management, UX, Engineering.
Evolution	Reactive, based on tracker tickets (ticket-driven).	Proactive, based on a roadmap (roadmap), user research, and feedback.
Funding	Cost center.	Value center.

The principal channel through which PaaP affects development teams is the purposeful improvement of the quality of their professional experience (Developer Experience, DevEx). Engagement here should be understood not as formal, imposed use of the platform, but as voluntary and active adoption that arises when the platform makes work noticeably easier, faster, and subjectively more pleasant. This effect emerges through its impact on the fundamental components of DevEx.

Contemporary academic models make it possible to delineate and operationalize this seemingly diffuse category. In particular, the DevEx framework proposed by Microsoft researchers identifies three key dimensions [13]. First, feedback loops: the speed and quality of the system's and colleagues' responses to a developer's actions. PaaP radically shortens these cycles through automated CI/CD pipelines that return build and test results within minutes, and self-service tools that instantly provide the necessary resources. Second, cognitive load: the amount of mental effort required to perform tasks. As shown earlier, PaaP directly reduces extraneous cognitive load by offering golden paths—standardized and well-supported routes for solving common tasks [5, 10]. Third, the state of flow: sustained immersion and concentration. By eliminating blockers, waits, and context switches (for example, from writing code to handwritten YAML configuration), PaaP helps maintain this highly productive state for longer.

Another influential framework — SPACE — offers a broader five-component lens for assessing productivity [13]: satisfaction and well-being (Satisfaction & well-being); performance/quality (Performance); activity (Activity); communication and collaboration (Communication & collaboration); and efficiency and flow (Efficiency & flow). The impact of PaaP on most of these dimensions is positive and straightforward. Satisfaction increases due to the removal of irritating and routine aspects of everyday work [1]. Quality (performance) rises thanks to built-in standards and automated checks that set a default threshold. Efficiency and flow improve as a result of accelerated development cycles and reduced cognitive load.

As a result, a transparent causal chain takes shape: the product approach to the platform systematically reduces cognitive load and accelerates feedback; this directly improves measurable DevEx/SPACE indicators, which in turn triggers the growth of voluntary engagement and, consequently, productivity. Schematically, this logic is presented in Figure 1.

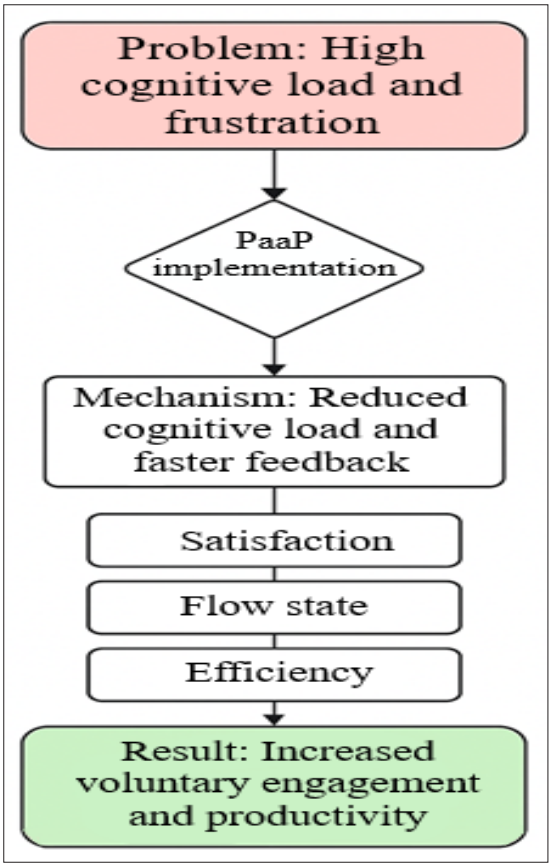

Fig. 1. Model of the influence of PaaP on engagement through reducing cognitive load (compiled by the author based on [1, 5, 10, 13]).

The theoretical model presented above receives convincing empirical validation from the practices of leading technology companies that were the first to institutionalize the product approach to the development of internal platforms.

A representative case is Spotify, where the internal developer platform evolved into the popular open-source product Backstage [9]. The consistent application of product thinking to internal tools yielded measurable effects. A key indicator of success is a sharp reduction in onboarding: the time required for a new engineer to achieve the first deployment to a production environment decreased from several weeks to several days [9]. Additionally, the platform standardized development practices across hundreds of teams and established a unified market of internal tools and services, enabling a full-fledged self-service model of infrastructure management [9, 12].

As another example, consider Netflix. This global streaming service applies the product approach to a wide spectrum of

internal means, from analytics platforms to build and testing systems, which makes it possible to sustain high reliability alongside the pace of innovation. The assessment of the effectiveness of tools for improving developer productivity demonstrates exceptional returns: the use of the Develocity platform to accelerate builds and tests saves more than 280 000 hours of developer working time per year. The metrics include a 50% reduction in total build time and a 60% reduction in sequential test execution time; in a number of projects the average build time was reduced from more than 10 minutes to 1–2 minutes [14, 17].

The European e-commerce leader Zalando implemented PaaP to unify the developer experience and increase operational efficiency [15, 16]. To assess maturity and outcomes, the platform is measured using KPI trees that link high-level business goals with specific technical and product metrics. The key measurement directions include productivity (productivity), lead time (lead time), deployment frequency (deployment frequency), developer happiness (developer happiness), stability (stability), operational efficiency (efficiency), and risk (risk). This framework not only improved technical indicators but also ensured compliance by default by embedding regulatory requirements directly into the platform [16].

The summary results of the case-study analysis are presented in Table 2.

Table 2. Summary results and metrics from the PaaP implementation case study (compiled by the author based on [9, 16, 17]).

Company	Platform/Initiative	Key results and metrics
Spotify	Backstage (internal developer portal)	Reduction of onboarding time for new engineers from weeks to days; standardization of practices across hundreds of teams.
Netflix	Developer Productivity Tools (including Develocity)	Savings of >280,000 developer hours per year; 50% reduction in build time; 60% reduction in test time.
Zalando	Internal Developer Platform	Management via a KPI tree (productivity, lead time, developer happiness); DevEx unification; compliance by default.

Transition to the PaaP approach requires a fundamental overhaul of the metrics system. Traditional IT indicators, such as uptime or the number of tickets closed, cannot capture the platform's primary value — accelerating and improving the work of development teams. Successful PaaP adoption implies an evolution of metrics from measuring activity to measuring outcomes and business value. To this end, it is advisable to use a multi-level assessment system.

Level 1: Delivery efficiency. At the basic level, it is necessary to measure the health and speed of the software delivery pipeline. The de facto standard here consists of four key DORA metrics (DevOps Research and Assessment) [20]:

- Deployment frequency: how often the organization successfully releases to the production environment.

- Lead time for changes: the time from commit to the successful running of code in production.

- Change failure rate: the share of deployments that cause failures.

- Mean time to recovery: how quickly the organization can restore the service after a failure.

Level 2: Developer experience and productivity (DevEx). This level measures the direct impact of the platform on its clients. The metrics here should reflect how convenient and useful the platform is, and how it supports productive work [9]:

- Adoption level: the percentage of teams that voluntarily (if the platform is not mandatory) choose to use the platform.

- Developer satisfaction: measured through regular surveys, for example using the Net Promoter Score (NPS) metric for the platform.

- Time to first meaningful contribution: an onboarding efficiency indicator measuring how quickly a new employee can start delivering value [7].

- Proxy metrics of cognitive load: indirect indicators such as the number of platform-related support requests, the time spent searching for documentation, or the number of steps required to complete a typical task.

Level 3: Business value. The highest level of metrics links platform investments to key business indicators. These metrics enable the platform team to speak the language of business and to demonstrate its value as a center of value creation rather than a cost center [16]:

- Time to market: the extent to which the platform accelerates the launch of new products or functionality [1].

- Return on Investment: calculated through developer time savings (as in the Netflix case), reduced infrastructure costs due to standardization, and lower costs of defect remediation.

- Talent attraction and retention: a mature platform that provides a strong DevEx becomes a competitive advantage in the labor market [1].

This hierarchical system of metrics, presented in Figure 2, provides a comprehensive view of platform effectiveness and its contribution to the success of the entire organization.

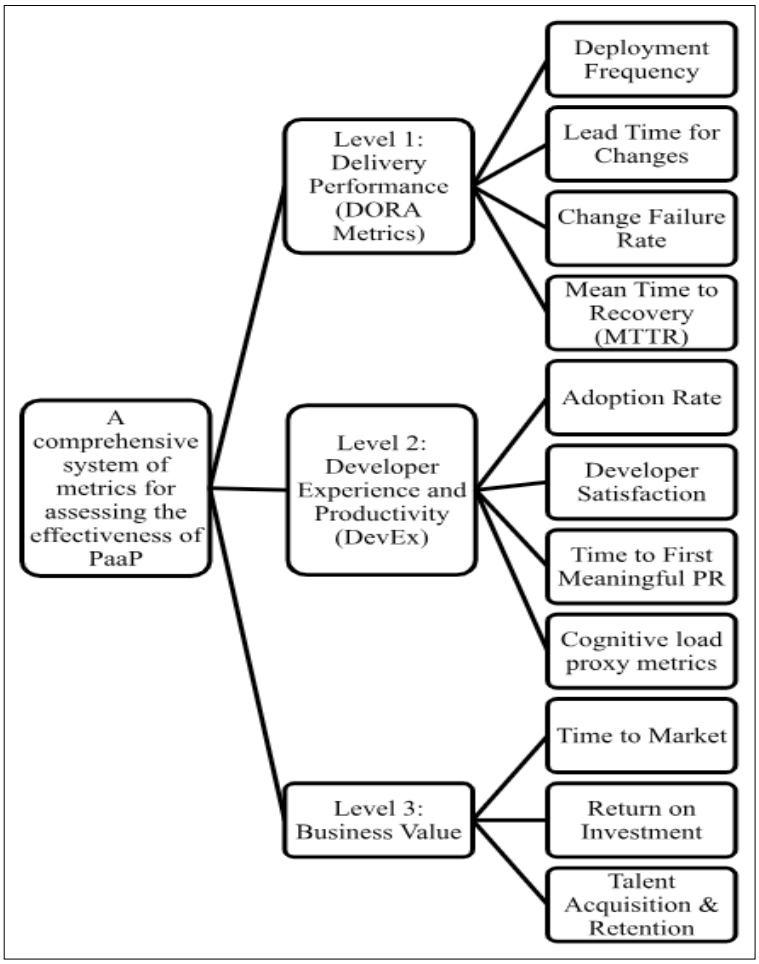

Fig. 2. A comprehensive system of metrics for assessing the effectiveness of PaaP (compiled by the author based on [1, 7, 9, 16, 20]).

Despite the obvious advantages, the transition to the PaaP model is associated with substantial risks and obstacles that are appropriately considered along three dimensions: cultural, technical, and strategic. Table 3 presents a classification of the barriers and the corresponding strategies for overcoming them.

Table 3. Classification of barriers to PaaP implementation and strategies for overcoming them (compiled by the author based on [16, 20]).

Barrier type	Barrier description	Mitigation strategy	Barrier type
Cultural	Cost center mindset	Demonstrate ROI through business metrics (hours saved, accelerated TTM); educate and engage stakeholders.	Cultural
Cultural	Captive audience syndrome	Adopt product practices: user research, journey mapping, feedback collection (NPS), regular demos.	Cultural
Technical	Integration complexity	Use an API-first approach; provide flexible paved roads rather than rigid rails; modernize the stack before building the IDP.	Technical
Strategic	Lack of product competencies	Hire or train product managers and UX designers; form balanced cross-functional teams.	Strategic

Thus, the conducted analysis allows us to conclude that the transition from the traditional understanding of internal platforms as infrastructural tools to the Platform as a Product model constitutes a fundamental shift in organizational design and development management. In contrast to a reactive and technically oriented approach, PaaP is grounded in product thinking, where internal developers are regarded as customers, and the key success criterion becomes the quality of their experience (DevEx). This is achieved through the reduction of incidental cognitive load, the acceleration of feedback, and the formation of loosely coupled architectural and communication structures. Empirical data from the

practices of leading technology companies (Spotify, Netflix, Zalando) confirm the measurable effectiveness of the approach in shortening onboarding time, standardizing processes, and increasing developer productivity and satisfaction. A comprehensive system of metrics (from DORA to ROI) demonstrates that PaaP transforms the platform from a cost center into a value creation center that directly influences business outcomes. At the same time, successful implementation requires overcoming cultural, technical, and strategic barriers, which makes PaaP not only a technological transformation but also an organizational and cultural one.

CONCLUSION

The empirical data obtained confirm the proposed hypothesis: the Platform as a Product interpretation serves as a powerful lever for increasing the engagement and effectiveness of development teams. This is achieved not through the administrative imposition of tools but through the deliberate development of developer experience (DevEx). The primary channel of impact is the systematic reduction of extraneous cognitive load, which enables engineers to direct mental resources to solving business problems instead of overcoming infrastructural and procedural barriers.

The research objective has been achieved: a coherent model is proposed and substantiated that links the PaaP concept with foundational socio-technical theories (Conway's law, cognitive load theory, team topologies) and contemporary measurement frameworks (DORA, DevEx/SPACE). The model demonstrates that the success of PaaP rests on three interrelated foundations: product (customer centricity, UX priority, a clear value proposition for internal users), organization (properly designed platform teams, a culture of feedback and trust), and measurement (a shift from IT activity metrics to indicators of impact on productivity, developer experience, and business outcomes).

The practical significance of the study lies in providing managers and technical leaders with a structured approach to implementing PaaP. Platform creation should be viewed not as a purely technical project but as a strategic transformation aimed at increasing the operational efficiency of the entire engineering function.

Key steps include:

- Secure executive sponsorship by demonstrating the expected ROI and direct alignment with business objectives.

- Form a balanced platform team with strong product expertise.

- Begin with in-depth research into the needs and pain points of internal developer users to define the MVP (Minimum Viable Platform).

- Implement a multi-level metrics system for continuous monitoring of progress and demonstration of the value created.

Promising directions for further research include quantitative analysis of the relationship between specific DevEx metrics (for example, satisfaction survey results) and long-term business indicators such as talent retention and the pace of innovation in organizations that have adopted PaaP. Of additional interest is the assessment of the impact of generative AI technologies on the evolution of platform engineering, since AI assistants may become an integral component of future platforms, further reducing cognitive load and accelerating development.

REFERENCES

1. New platform engineering research report [Electronic resource] Access mode: https://cloud.google.com/blog/products/application-modernization/new-platform-engineering-research-report (date of access: 10.08.2025)
2. Platform Engineering That Empowers Users and Reduces Risk [Electronic resource] Access mode: https://www.gartner.com/en/infrastructure-and-it-operations-leaders/topics/platform-engineering (date of access: 12.08.2025)
3. Bortolini M. et al. A two-step methodology for product platform design and assessment in high-variety manufacturing //The International Journal of Advanced Manufacturing Technology. – 2023. – Vol. 126 (9). – pp. 3923-3948.
4. 50+ Essential Software Development Statistics and Trends for 2024 - Vrinsoft Technology [Electronic resource] Access mode: https://www.vrinsofts.com/software-development-statistics-and-market-trends/ (date of access: 24.08.2025)
5. Software Development Statistics for 2025: Trends & Insights [Electronic resource] Access mode:https://www.itransition.com/software-development/statistics (date of access: 07.08.2025)
6. AI, platform engineering, or DX? How to choose where to invest – LeadDev [Electronic resource] Access mode:https://leaddev.com/technical-direction/ai-platform-engineering-dx-choose-where-invest (date of access: 07.08.2025)
7. The 2024 State of Developer Productivity [Electronic resource] Access mode: https://www.cortex.io/report/the-2024-state-of-developer-productivity (date of access: 24.07.2025)
8. Karamthulla M. J., Malaiyappan J. N. A., Prakash S. AI-powered self-healing systems for fault tolerant platform engineering: Case studies and challenges //Journal of Knowledge Learning and Science Technology ISSN: 2959-6386 (online). – 2023. – Vol. 2 (2). – pp. 327-338.
9. Liu P., Chen G., Zhang J. A review of liposomes as a drug delivery system: current status of approved products,

regulatory environments, and future perspectives // Molecules. – 2022. – Vol. 27 (4). – 1372.

10. Zhang X., Yang Y., Chen Y. History and future of business ecosystem: a bibliometric analysis and visualization // Environment, Development and Sustainability. – 2024. – pp. 1-24.
11. Aune A. A. W. Towards Enhanced Developer Experience: An Empirical Study on Successful Adoption of Internal Developer Platforms : дис. – NTNU, 2024.
12. Research Papers - ASE 2025 [Electronic resource] Access mode: https://conf.researchr.org/track/ase-2025/ase-2025-papers (date of access: 10.08.2025)
13. Razzaq A. et al. A systematic literature review on the influence of enhanced developer experience on developers' productivity: Factors, practices, and recommendations //ACM Computing Surveys. – 2024. – Vol. 57 (1). – pp. 1-46.
14. Lopes J. G. M., Oliveira J., Figueiredo E. Evaluating the Impact of Developer Experience on Code Quality: A Systematic Literature Review //Congresso Ibero-Americano em Engenharia de Software (CIbSE). – SBC, 2024. – pp. 166-180.
15. Gartner® Hype Cycle™ for Platform Engineering Report - Gitpod [Electronic resource] Access mode: https://www.gitpod.io/gartner-platform-engineering-hype-cycle-2024 (date of access: 12.08.2025)
16. Internal Platform Teams: What Are They and Do You Need One? | Humanitec [Electronic resource] Access mode: https://humanitec.com/blog/internal-platform-teams-what-are-they-and-do-you-need-one (date of access: 14.08.2025)
17. Netflix | Case Study | Gradle [Electronic resource] Access mode: https://gradle.com/customers/story/netflix/ (date of access: 10.08.2025)
18. Allsop D. B. et al. Qualitative methods with Nvivo software: A practical guide for analyzing qualitative data //Psych. – 2022. – Vol. 4 (2). – pp. 142-159.
19. Stark J. Product lifecycle management (PLM) //Product lifecycle management (volume 1) 21st Century paradigm for product realisation. – Cham : Springer International Publishing, 2022. – pp. 1-32.
20. Abate E. et al. Strategic Sourcing of Digital Platforms in the Industrial Internet of Things. pp.62-105.