Scientific Research Publishing

World Journal of Engineering and Technology, 2025, 13(4), 754-763
https://www.scirp.org/journal/wjet
ISSN Online: 2331-4249
ISSN Print: 2331-4222

The Synergy of Low-Code/No-Code and AI/ML: Enhancing Intelligent Automation

Nisar Ahmed[1], Muhammad Shahbaz[2], Hafiz Shafiq Ur Rehman Khalil[1], Zahid Javed[3], Muhammad Ali Khalid[4], Farhan Ajmal Khan[5], Zubair Maroof[6]

[1]Department of Computer Engineering, University of Coimbra, Coimbra, Portugal
[2]Department of Computer Science, Comsats University, Islamabad, Pakistan
[3]Department of Computer Science, National Textile University, Faisalabad, Pakistan
[4]School of Energy and Power Engineering, Xi'an Jiaotong University, Xi'an, China
[5]Department of Artificial Intelligence, Sapienza University of Rome, Rome, Italy
[6]School of Electronics and Information Engineering, Xi'an Jiaotong University, Xi'an, China
Email: nahmed@dei.uc.pt, fa14mcs151@vcomsats.edu.pk, hsurk@dei.uc.pt, zahid.javed@ntu.edu.pk, alikhalid2k18@gmail.com, khan.1923279@studenti.uniroma1.it, zubairxjtu15@hotmail.com

How to cite this paper: Ahmed, N., Shahbaz, M., Khalil, H.S.U.R., Javed, Z., Khalid, M.A., Khan, F.A. and Maroof, Z. (2025) The Synergy of Low-Code/No-Code and AI/ML: Enhancing Intelligent Automation. *World Journal of Engineering and Technology*, **13**, 754-763.
https://doi.org/10.4236/wjet.2025.134048

Received: August 18, 2025
Accepted: September 8, 2025
Published: September 11, 2025

Abstract

Software development has been revolutionized by low-code and no-code platforms, which make it possible for even non-programmers to create and launch apps rapidly. In contrast to traditional coding, they speed up development with drag-and-drop components, pre-built templates, and ready-to-use plugins. Their effects on accelerating innovation, reducing expenses, and facilitating digital transformation are examined in this article, together with issues including scalability, security, and restricted customization. It also investigates how similar platforms might be used in the future to promote cooperation between technical and non-technical teams. While LCNC platforms help close the gap between IT solutions and business needs, thorough integration is necessary for long-term success.

Keywords

Low-Code, No-Code Platforms, Software Development, Digital Transformation, Citizen Developers, Innovation

1. Introduction

Low-code and no-code platforms are transforming software development because they allow non-programmers to design apps. Drag-and-drop tools, visual interfaces, and prebuilt templates allow both non-technical and professional users to design solutions without fully relying on IT workers. These systems speed up

time-to-market, cut costs, and boost agility while enabling "citizen developers" from other departments to innovate and solve their own business issues. However, there are still problems with long-term viability, security, and scalability, especially in business settings. Despite these challenges, LCNC platforms are driving digital transformation, boosting productivity, and are anticipated to have a lasting effect on how companies adjust to changing demands.

From complicated binary machine code [1] to today's low-code/no-code (LCNC) platforms, programming has changed since the invention of the ENIAC (Electronic Numerical Integrator and Computer) in 1943. The conversion of low-level instructions into machine code was made much simpler by assembly languages (1949) and autocade (1952) [2]. High-level procedural languages like Fortran, Algol, and COBOL [2] [3] were developed in the 1950s, significantly lowering the complexity of coding. New paradigms were brought about by object-oriented languages like Simula and Smalltalk (1965). Platforms like Out Systems (2001) have surfaced more recently to use LCNC techniques to speed up digital transition [4].

Webratio is an example of a low-code platform for the visual modeling approach to developing applications, including data structures, user interfaces, and business processes. This is based on widely used modeling languages like Interaction Flow Modeling Language (IFML) and Business Process Model and Notation (BPMN) [5].

YOLOv2 is one of the code-based object identification models used in ML/AI [6]. YOLOv2 is not a low-code or no-code model; rather, it is a code-based object detection paradigm. Effective implementation and use necessitate a high level of programming expertise. This deep learning model demands a thorough comprehension of training pipelines, data preparation, and machine learning principles. Numerous systems provide low-code, visual tools for annotating photos and getting datasets ready for training. AI support is frequently included in these technologies to expedite the manual procedure.

Low-code or no-code models are not those such as Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Support Vector Machine (SVM), and Invertible Neural Networks (INN) [7]. It takes a great deal of coding knowledge to apply and utilize these mathematical and computational ideas. These models can be used in a low-code environment, but they cannot be built without code or even low-code. The sophisticated code required to execute these algorithms is abstracted away by built-in modules or graphic elements seen in low-code machine learning platforms.

The increasing use of LCNC platforms is democratizing access to sophisticated technology, hence transforming the field of automation and artificial intelligence. A streamlined, visual interface for managing the full machine learning lifecycle, from data ingestion to deployment and monitoring, has made it possible for no-code AI tools to leverage complex machine learning operations (MLOps) [8]. In specialized areas of computer vision, such as 3D-Dual Contextualized Model (DCM) and Pedestrian Visual Acuity Module (PVAM), LCNC platforms can as-

sist in integrating sophisticated models into useful applications without requiring developers to write a lot of code for the underlying algorithms themselves [9]. Low-code platforms have emerged as a crucial tool for programmers to quickly create and implement automated workflows, optimize business procedures, and integrate disparate systems, thereby speeding up digital transformation across multiple industries [10]. This capability goes beyond artificial intelligence and into broader industry automation.

2. Research Methodology

To have a comprehensive understanding of low-code/no-code (LCNC) platforms, this research will do a thorough literature review, examining the platforms' historical development, influence on the software development lifecycle (SDLC), and potential future applications. A systematic literature review process will be used to accomplish this. Using specific keywords, the method starts with a targeted search strategy across industry sources, including reports and white papers, as well as academic databases like IEEE Xplore and the ACM Digital Library. Strict inclusion criteria will guide the selection of pertinent literature, avoid promotional content and require sources to be peer-reviewed or published within a certain timeframe. Following identification, important information will be taken out and combined using thematic analysis to find recurring themes and trends. The anticipated results of this thorough analysis include a clear timeline of LCNC's evolution, a summary of its reported effects on the SDLC, and a discussion of its future trajectory, including the role of AI and machine learning. Ultimately, the findings will serve as a foundational resource for both researchers and practitioners.

Thorough research was carried out throughout the academic databases IEEE Xplore, ACM Digital Library, Scopus, Web of Science, and SpringerLink in order to guarantee replicability [11]. Publications from January 2015 to December 2023 were included in the search, which corresponds to the time frame when LCNC platforms became widely used in tandem with advances in AI and ML.

In the first search, 432 records were found. Titles and abstracts were used to screen 172 articles for relevance after duplicates (96 records) and non-peer-reviewed or promotional content (164 records) were eliminated. 68 studies that satisfied the inclusion criteria after full-text evaluation were added to the theme analysis.

A practical demonstration of the benefits of combining LCNC with AI/ML can be seen in Siemens' use of the Mendix low-code platform for predictive maintenance [12]. By integrating ML algorithms for fault detection, Siemens reduced unplanned downtime by approximately 30%, generating annual cost savings in the millions. Similarly, a Forrester Total Economic Impact (FTEI) study (2021) reported that organizations adopting Microsoft Power Apps with AI Builder achieved an 188% return on investment over three years, with app development cycles accelerated by 74% compared to traditional coding. These results provide concrete evidence that AI-enhanced LCNC platforms can deliver measurable business value

beyond theoretical advantages [13].

While LCNC platforms accelerate development, several challenges limit their long-term adoption. Security vulnerabilities are frequently reported, as non-technical developers often misconfigure access controls; a recent study found that over 40% of LCNC-built enterprise apps exposed sensitive data through weak authentication policies [14]. Scalability is another limitation; for instance, an ACM case study (2021) demonstrated that LCNC applications integrating IoT data experienced performance degradation at scale, necessitating hybrid coding approaches. Vendor lock-in has also been highlighted in comparative studies (Sahay *et al.*, 2020), which showed that proprietary architectures and lack of interoperability prevent organizations from easily migrating between platforms. Finally, maintenance issues emerge as applications mature: SoftwareX (2022) reported that many LCNC applications developed by business users later created significant technical debt when integrated into IT-managed systems. These findings suggest that while LCNC platforms empower rapid development, their long-term viability requires careful governance and hybrid approaches [15].

3. Addressing Research Gap

Although the use of LCNC systems is expanding quickly, there are still a lot of unanswered questions. Applications developed on these platforms must be sustainable and scalable over the long run. Their capacity to sustain enterprise-level performance, dependability, and maintenance as they become more complicated and integrate with other systems is still unknown.

The security and compliance implications of LCNC are another area that has not received enough attention. There is a greater chance of vulnerabilities and misconfigurations when non-technical people create more applications. The strength of integrated security features and the efficiency of governance systems in guaranteeing secure development processes require more investigation.

The socio-economic impacts of LCNC also need more research. Although technology can be made more accessible through these platforms, its wider effects on workforce skills, professional development job positions, and the digital divide are not well understood.

Filling the research gaps in low-code/no-code (LCNC) platforms requires a blend of practical and theoretical approaches. To address scalability and sustainability, researchers should conduct long-term studies and use real-world case studies, comparing LCNC solutions to traditional development methods to understand their performance with enterprise-level workloads. For security and compliance, new frameworks and tools are needed; experts should collaborate to design governance models and investigate common vulnerabilities inherent in these platforms, with a focus on creating better user training and automated security checks. On the socio-economic front, interdisciplinary research is crucial to examine LCNC's impact on labor markets, including how it changes the roles of professional and citizen developers and whether it promotes inclusion for un-

derrepresented groups. Ultimately, filling these gaps requires close collaboration between academia, industry, and policymakers to ensure the responsible and sustainable adoption of this technology.

4. Evolution of Low-Code/No-Code Platforms

A key motivation for low-code/no-code (LCNC) platforms is faster application development. A big shift happened in the 1980s and 1990s with the introduction of visual development tools like Microsoft Access and Visual Basic. By moving software development away from intricate coding and toward a graphical environment, these platforms allowed developers to produce simple apps with less code. This paved the way for more advanced platforms. The development of LCNC was further accelerated in the 2000s by the rise of cloud computing and Software as a Service (SaaS). Companies like Salesforce and Zoho provided low-code capabilities with drag-and-drop tools and prebuilt components, enabling users to customize workflows and expand functionality. As a result, non-technical people found it easier to create their own programs. The 2010s saw a substantial evolution in low-code/no-code (LCNC) platforms because of the shift to mobile-first development, APIs, and AI breakthroughs. Emerging vendors such as Mendix, Out Systems, and Bubble provided scalable options for developing applications of all complexity levels.

LCNC platforms became popular in a variety of industries as companies placed a higher priority on agility, a quicker time to market, and a decreased dependency on specialized IT teams. They are increasingly seen as essential to digital transformation because they allow businesses to quickly innovate and adapt. Looking ahead, LCNC will continue to evolve due to the deeper integration of AI and machine learning, which will result in more intelligent, automated, and tailored applications (Figure 1).

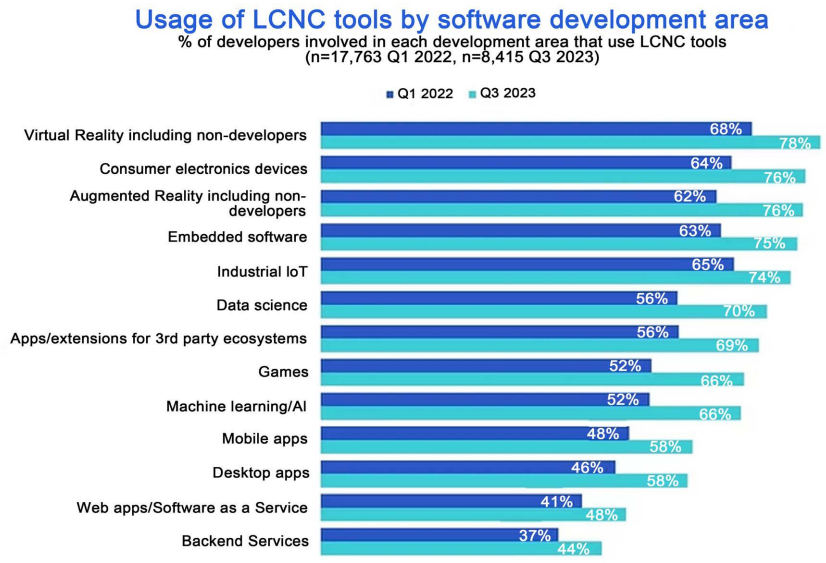

Figure 1. Usage of LCNC tools by software development area.

By reducing the need for manual coding through the use of pre-built components and visual interfaces, LCNC systems present an appealing answer to development problems. By facilitating application development through drag-and-drop capabilities and modular components, these platforms reduce the entrance barrier for a larger audience [16]. Additionally, LCNC improves business-IT alignment by enabling analysts and business leaders with in-depth domain knowledge to translate their ideas into IT solutions without the need for IT specialists [17]. Employees from many departments can develop customized solutions without requiring a lot of technical assistance thanks to LCNC platforms, which empower non-coders [18].

5. Impact on Software Development Life Cycle

Through the introduction of new workflows and efficiency, low-code and no-code (LCNC) platforms are transforming the Software Development Life Cycle (SDLC). By streamlining conventional SDLC phases, these systems shorten development times and improve communication between technical and non-technical teams. They do, however, also bring with them new factors to consider while organizing, carrying out, and sustaining projects.

5.1. Requirement Gathering and Planning

Through direct participation from non-technical users, low-code/no-code (LCNC) platforms streamline the requirements collecting stage of the development cycle. It takes a lot less time to translate requirements to developers when business users can prototype applications themselves. This tight cooperation guarantees that the finished result meets user expectations and enhances everyone's comprehension of the project's objectives.

5.2. Development

Platforms that are low-code/no-code (LCNC) radically alter the development process. They greatly accelerate the building of applications using visual interfaces, drag-and-drop capability, and prebuilt components. Additionally, LCNC automates tedious processes like data integration and UI design, freeing your teams to concentrate on more important duties like innovation and customization.

5.3. Testing

One important component of low-code/no-code (LCNC) platforms is integrated automated testing tools, which greatly accelerate the testing stage. These platforms frequently have preconfigured debug settings and testing environments, which help you find and fix problems fast. However, for more complicated applications, it can still be necessary to employ external tools or manual labour to test for edge situations and confirm performance under stress.

5.4. Deployment

Most of the contemporary low-code/no-code (LCNC) platforms provide a single-

click deployment functionality, making the deployment process simpler. These platforms take care of infrastructure and environmental management, which is frequently made even easier by vendor-based cloud hosting. Applications may be swiftly deployed and effectively scaled as needed by enterprises thanks to this.

5.5. Maintenance and Updates

Low-code/no-code (LCNC) platforms are designed to simplify updating and maintenance using features like centralized dashboards and real-time editing. However, while they ease initial development, long-term maintainability can be challenging. Key risks include cumbersome backwards compatibility, the potential for a project to outgrow the platform, vendor lock-in, and a dependency on proprietary technology.

Business-IT alignment is improved via low-code/no-code (LCNC) platforms, which enable domain-savvy analysts and business leaders to translate their ideas into IT solutions without the need for IT specialists [19]. These platforms, sometimes referred to as low-code development platforms (LCDPs), allow citizen developers to create software systems even if they have no prior programming experience by using intuitive visual environments [20].

Present-day low-code/no-code (LCNC) platforms, such as Microsoft Power Platform, Mendix, and OutSystems, offer interactive data visualization features to researchers who are not programmers. This facilitates studies' ability to present their research findings in an effective manner [21] [22].

6. Emerging Trends

As they spearhead digital transformation across industries, low-code and no-code (LCNC) platforms are poised for exponential growth, propelled by the integration of AI and ML.

It will be much simpler to create apps and make smarter decisions with enhanced AI-driven capabilities like predictive analytics and automatic code development. It is expected that this development would lead to a wider range of complex use cases, such as AI-powered chatbots, Internet of Things apps, and sophisticated workflow automation, enabling both professional teams and citizen developers to utilize these potent tools.

Enterprise-grade feature expansion is linked to the future growth of low-code/no-code (LCNC) platforms. Key issues like scalability, security, and compliance are being prioritized by vendors as more companies use these platforms for mission-critical applications. Offering powerful integrations with ERP and CRM systems, multi-cloud/hybrid compatibility, and strong governance capabilities are all examples of this.

By democratizing technology, low-code/no-code (LCNC) platforms are having a big social impact in addition to their increasing technical interoperability. They are contributing to closing the digital gap and fostering creativity for individuals, entrepreneurs, and schools throughout the world by making it possible for non-

technical users to develop applications.

7. Low Code/No Code & Gen AI

The emergence of generative AI is revolutionizing low-code/no-code (LCNC) platforms, making them more intelligent and adaptable than before. These technologies are improved by generative AI, which makes application development even simpler by automating difficult processes like code authoring, design creation, and workflow optimization. This potent mix enables businesses to develop more sophisticated, customized, and intelligent solutions more quickly. By adopting these AI-powered solutions, companies can achieve unprecedented levels of creativity, productivity, and innovation, enhancing LCNC's position as a vital part of contemporary digital ecosystems.

New difficulties arise when generative AI is integrated with low-code/no-code (LCNC) systems. These difficulties include data privacy concerns, ethical use issues, and excessive dependence on automation. To guarantee responsible growth, these risks need to be controlled. But this combination also presents enormous potential to transform the way we develop technology. More people will be able to access and profit from the digital revolution if these platforms, which are propelled by astute governance and ongoing innovation, usher in an era of inclusive and equitable growth. By reducing the technical skills needed for domain experts to use artificial intelligence (AI), low-code/no-code (LCNC) AI systems are crucial for obtaining actionable knowledge. To guarantee that the models produce pertinent and useful information, future research will concentrate on customizing generative models to certain domains, creating specialized architectures, and training techniques and assessment measures [23].

8. Conclusions

Low-code and no-code (LCNC) platforms are revolutionizing software development by facilitating innovation and opening up application creation to a larger user base. These platforms facilitate communication between technical and business teams by streamlining intricate procedures, which drastically cut down on development time and foster digital transformation.

But there are still issues, especially with security, scalability, and vendor lock-in. The way that LCNC converges with technologies like AI, ML, and IoT will determine its future, increasing its application in small enterprises, education, and other areas. In the end, LCNC platforms are poised to transform the development and application of technology, propelled by research and astute governance.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] CHM (2022) Computer History Museum, Birth of Computer-ENIAC.

https://www.computerhistory.org/revolution/birth-of-the-computer/4/78

[2] HP (2018) Computer History: A Timeline of Computer Programming Languages. https://www.hp.com/us-en/shop/tech-takes/computer-history-programming-languages

[3] Rizwan, O. (2018) A Snapshot of Programming Language History, Increment. https://increment.com/programming-languages/language-history/

[4] Outsystems (2001) It Began with a Vision. https://www.outsystems.com/evaluation-guide/it-began-with-a-vision/

[5] Acerbis, R., Bongio, A., Brambilla, M. and Butti, S. (2016) Model-Driven Development of Cross-Platform Mobile Applications with Web Ratio and IFML. 2015 2*nd ACM International Conference on Mobile Software Engineering and Systems*, Florence, 16-17 May 2015, 170-171.

[6] Mujeeb, A., Ahmed, N., Arshed, H. and Khan, F. (2022) Electronic Toll Collector Framework. *Advances in Science and Technology Research Journal*, **16**, 294-302. https://doi.org/10.12913/22998624/144537

[7] Ahmed, N., Khan, F., Ullah, Z., Ahmed, H., Shahzad, T. and Ali, N. (2021) Face Recognition Comparative Analysis Using Different Machine Learning Approaches. *Advances in Science and Technology Research Journal*, **15**, 265-272. https://doi.org/10.12913/22998624/132611

[8] Sundberg, L. and Holmström, J. (2023) Democratizing Artificial Intelligence: How No-Code AI Can Leverage Machine Learning Operations. *Business Horizons*, **66**, 777-788. https://doi.org/10.1016/j.bushor.2023.04.003

[9] Ul Abideen, Z., Ahmed, N., Khalil, H.S.U.R. and Shahbaz, M. (2025) Advancing Pedestrian Trajectory Prediction with Interaction-Aware 3D-Dual Contextualized Modeling. *Egyptian Informatics Journal*, **31**, Article ID: 100742. https://doi.org/10.1016/j.eij.2025.100742

[10] Tsahat, C.O.O., Ngoulou-A-Ndzeli and Kwai, P.A.A. (2023) Prospects of Using Low-Code in the Creation of Automated Systems. *Open Journal of Applied Sciences*, **13**, 1864-1869. https://doi.org/10.4236/ojapps.2023.1310147

[11] Wilde, M. (2016) IEEE Xplore Digital Library. *The Charleston Advisor*, **17**, 24-30 https://doi.org/10.5260/chara.17.4.24

[12] Emma, L. (2025) AI-Driven Predictive Maintenance for Smart Manufacturing and Industry 4.0. https://www.researchgate.net/publication/389498658

[13] Shrivastava, A. (2024) Learning Microsoft Power Apps: Building Business Applications with Low-Code Technology. O'Reilly Media, Inc.

[14] John, B. (2025) Low-Code and No-Code Platforms: Accelerating Enterprise Software Development. https://www.researchgate.net/publication/389634372

[15] Alfonso, I., Conrardy, A. and Cabot, J. (2025) Towards the Interoperability of Low-Code Platforms. In: Pufahl, L., Rosenthal, K., España, S. and Nurcan, S., Eds., *Intelligent Information Systems*, Springer, 3-11. https://doi.org/10.1007/978-3-031-94590-8_1

[16] Gartner Research (2021) Low-Code Development Technologies Evaluation and Trends. Gartner, Inc. https://www.gartner.com

[17] McKendrick, J. (2022) Low-Code No-Code Market Keeps Growing, and That Means Shifts in Technology Roles. ZDNET.

[18] IDC Research (2020) Low-Code and No-Code Platforms in the Age of Digital Transformation. IDC Report.

[19] Sufi, F. (2023) Algorithms in Low-Code-No-Code for Research Applications: A Prac-

tical Review. *Algorithms*, **16**, Article 108. https://doi.org/10.3390/a16020108

[20] Sahay, A., Indamutsa, A., Di Ruscio, D. and Pierantonio, A. (2020) Supporting the Understanding and Comparison of Low-Code Development Platforms. 2020 46*th Euromicro Conference on Software Engineering and Advanced Applications* (*SEAA*), Portoroz, 26-28 August 2020, 171-178. https://doi.org/10.1109/seaa51224.2020.00036

[21] Sahinaslan, E., Sahinaslan, O. and Sabancıoglu, M. (2021) Low-Code Application Platform in Meeting Increasing Software Demands Quickly: SetXRM. *AIP Conference Proceedings*, **2334**, Article ID: 70007. https://doi.org/10.1063/5.0042213

[22] Chhor, J., Fischer, V., Kröppel, F. and Schmitt, R.H. (2022) Rule-Based Decision Support for No-Code Digitalized Processes. *Procedia CIRP*, **107**, 258-263. https://doi.org/10.1016/j.procir.2022.04.042

[23] Ramdurai, B. and Adhithya, P. (2023) The Impact, Advancements and Applications of Generative AI. *International Journal of Computer Science and Engineering*, **10**, 1-8. https://doi.org/10.14445/23488387/ijcse-v10i6p101