STM JOURNALS

Journal of
Open Source Developments

ISSN: 2395-6704
Volume 12, Issue 2, 2025
May–August
DOI (Journal): 10.37591/JoOSD

https://journals.stmjournals.com/joosd

Review JoOSD

Nitro IDE: Exploring Integrated Development Environments: Current Trends and Innovations

Aditya More[1,*], Roshani Dhodare[2], Avantika Kaloge[2], Balalsaheb Tarle[3], Nikhil Kulkarni[4]

Abstract

Integrated Development Environments (IDEs) have become essential in modern software development, offering a centralized platform that brings together source code management, debugging tools, version control, and compilation features. This study offers an in-depth analysis of various IDE platforms, with a particular focus on Eclipse as a widely used open-source integration environment. It also explores specialized IDEs tailored for embedded systems and enterprise-level Java (J2EE) application development. Additionally, the study examines how web-based IDEs supported remote learning and coding practices during the COVID-19 pandemic. By reviewing five key research studies, we highlight the evolution of IDEs, current challenges, and emerging opportunities for both industry professionals and students. As software development continues to evolve, IDEs have kept pace by incorporating advanced tools, automation capabilities, and cross-platform support. These enhancements have made development environments more efficient, adaptable, and accessible across different user groups. Not only have these improvements benefited large-scale enterprise projects, but they have also played a vital role in computer science education, helping students learn and collaborate effectively. This review specifically considers how IDEs enable team-based development, assist in early error detection, and support cloud deployment workflows. The growth of online IDEs, particularly during the shift to remote education caused by the pandemic, has broken down location-based barriers. These platforms have made it easier for learners and developers to write, test, and share code in real-time, using shared environments and cloud resources. Features like live collaboration, classroom integration, and resource scalability have transformed how coding is taught and practiced in virtual settings.

Keywords: Nitro IDE, Anstric games, integrated development environment (IDE), Eclipse, embedded software, J2EE, online IDEs, open-source, education, software development

***Author For Correspondence**
Aditya More
E-mail: info@anstric.com

[1]Founder and CEO, Anstric Games Private Limited, Nashik, Maharashtra, India
[2]Student, Department of Computer Enginnering, Maratha Vidya Prasarak Samaj's Karmaveer Adv. Baburao Ganpatrao Thakare College of Engineering (MVPS's KBTCOE), Nashik, Maharashtra, India
[3]Head, Department of Computer Enginnering, Maratha Vidya Prasarak Samaj's Karmaveer Adv. Baburao Ganpatrao Thakare College of Engineering (MVPS's KBTCOE), Nashik, Maharashtra, India
[4]General Manager and Head - Incubation, Nashik Engineering Cluster, Nashik, Maharashtra, India

Received Date: May 13, 2025
Accepted Date: May 26, 2025
Published Date: June 13, 2025

Citation: Aditya More, Roshani Dhodare, Avantika Kaloge, Balalsaheb Tarle, Nikhil Kulkarni. Nitro IDE: Exploring Integrated Development Environments: Current Trends and Innovations. Journal of Open Source Developments. 2025; 12(2): 35–40p.

INTRODUCTION

Software development has always required efficient management of code, version control, testing, and deployment. Integrated Development Environments (IDEs) emerged as a solution to these needs, offering a single interface to integrate all necessary tools. With the increasing complexity of software, the role of IDEs has expanded, from simple code editors to comprehensive environments capable of supporting the entire software development lifecycle. The purpose of this study is to review various IDE frameworks by synthesizing the insights from four key research papers:

1. *Eclipse as an open platform:* Analyzing how Eclipse has become the foundation for various programming languages and tools, allowing third-party integrations through its extensible architecture [1].

2. *IDEs for embedded software:* Investigating the challenges of building IDEs for embedded software development, which require specialized toolsets [2].
3. *J2EE IDEs:* Evaluating IDEs built for J2EE, exploring their specific needs in managing enterprise applications [3].
4. *Online IDEs for education during COVID-19:* Analyzing the surge in online IDE usage for educational purposes during the pandemic and the effectiveness of these platforms [4].

LITERATURE SURVEY

Eclipse as a Platform for Development Tool Integration

The Eclipse platform, introduced in 2001, has become one of the most successful open-source IDE frameworks. Its modular and extensible plug-in architecture allows developers to extend its capabilities far beyond Java, the original target language. Through community contributions, Eclipse now supports languages like C++, PHP, Python, and even non-software engineering applications like scientific computation, which highlight that Eclipse was not just conceived as an IDE but as a platform for other tool integrations. This extensibility provides developers the freedom to customize the IDE according to their development needs, which has led to its widespread adoption in academic and professional contexts [1].

IDEs for Embedded Software

Tailored to address the unique requirements of embedded software engineering: Unlike general-purpose applications, embedded systems interact directly with hardware, requiring cross-compilation, real-time debugging, and hardware simulation tools integrated into the IDE. Their case study on open-source IDEs for embedded software emphasized the use of Eclipse's modular framework, which can accommodate specialized plug-ins for hardware interaction and real-time testing environments. However, limitations in real-time performance monitoring and the need for additional hardware interfaces pose challenges in developing such IDEs [2].

Analysis of J2EE IDEs

Analyzed IDEs built specifically for J2EE, focusing on their role in enterprise-level software development: J2EE applications involve web-based services, which require extensive server management, database connectivity, and complex deployment strategies. IDEs such as NetBeans, IntelliJ IDEA, and Eclipse provide specific tools for handling these tasks, streamlining the development and deployment process. Their study identifies the strengths and weaknesses of various J2EE IDEs, concluding that while Eclipse provides powerful tools for server management, IntelliJ IDEA outperforms in terms of ease of use and code refactoring features [3].

Online IDEs For Education During COVID-19

Several online IDE platforms, focusing on their usage during the pandemic: As universities and schools transitioned to online learning, educators faced challenges in maintaining student engagement and ensuring effective learning outcomes for computer programming courses. Online IDEs, such as Replit and CodeSandbox, emerged as viable solutions due to their accessibility and simplicity. While these platforms provided an immediate solution, the study found that online IDEs lacked some advanced functionalities such as local debugging and real-time performance tracking, which could impact the learning experience for more advanced students [4].

EVOLUTION OF INTEGRATED DEVELOPMENT ENVIRONMENT

The evolution of Integrated Development Environments (IDEs) reflects the increasing complexity and specialization in software development. This section explores the contributions of the Eclipse framework, its significance in industrial software development, and a comparison with proprietary IDEs like Microsoft Visual Studio [5].

Eclipse Framework: Achitecture and Extensibility

Eclipse's modularity is one of its defining characteristics. As an open-source platform, Eclipse allows developers to install plug-ins tailored to their project needs. For example, Java developers can use

the Java Development Tools (JDT), while C++ developers can leverage the C/C++ Development Tooling (CDT) [1].

Plug-in Architecture

Each component of Eclipse is a plug-in, even the core functionalities like the user interface or the code editor. This makes Eclipse highly customizable.

Community Contributions

Being an open-source project, Eclipse benefits from a large community that develops and maintains plug-ins for various technologies.

Cross-Language Support

Initially designed for Java, Eclipse now supports a broad range of programming languages through community-developed plug-ins, making it a multipurpose IDE.

Eclipse's Role in Industrial Development

Industries have adopted Eclipse not only for its versatility but also for its collaborative potential. In large-scale projects, Eclipse facilitates integration with build automation tools like Jenkins, source control systems like Git, and various project management tools. The flexibility provided by Eclipse's architecture allows teams to align the IDE with their development methodologies, whether Agile, Waterfall, or DevOps.

Comparison with Proprietary IDEs

Proprietary IDEs like Microsoft Visual Studio are known for offering direct access to vendor-specific features and deep integration with Microsoft's ecosystem. In contrast, Eclipse's vendor-neutral, open-source approach offers flexibility and adaptability but may require more initial configuration effort to meet specific needs.

IDEs FOR EMBEDDED SOFTWARE DEVELOPMENT

Embedded software development differs significantly from general-purpose software engineering due to its inherent hardware dependencies. Embedded systems differ from standard desktop or cloud-based applications by needing to function within tight limitations on memory, computing performance, and power usage. This section discusses the specialized requirements for embedded software development, the role of open-source IDEs, and the challenges developers face when working with embedded IDEs [2].

Unique Requirements for Embedded Systems

Embedded software development is distinct from general software development due to its tight coupling with hardware. Due to the strict limitations in memory, processing capabilities, and power usage, embedded systems require IDEs to offer dedicated tools and features tailored to these challenges [2].

Cross-Compilation

Embedded systems often run on processors that differ from standard desktop processors, requiring cross-compilation from one architecture to another.

Real-Time Debugging

Embedded IDEs need to provide tools for real-time debugging and performance monitoring, as real-time constraints are critical in systems like automotive or industrial control systems.

Open-Source IDEs in Embedded Development

It was demonstrated that open-source IDEs, like Eclipse with its CDT plug-in, can support embedded software development. They provided a case study showing how Eclipse was adapted for hardware-specific tasks, including the integration of cross-compilers and hardware simulation tools.

Challenges in Embedded Development with IDEs

The authors noted several challenges in creating effective embedded IDEs. These include the integration of real-time debugging tools and simulators that can accurately mimic hardware behavior, as well as the steep learning curve associated with configuring the IDE for different hardware environments.

INTEGRATED DEVELOPMENT ENVIRONMENTS FOR J2EE APPLICATIONS

Enterprice-level Application Development

J2EE offers a framework for building enterprise-level applications that are large-scale, distributed, and structured across multiple tiers. These types of applications are inherently complex which involve web servers, databases, and business logic layers, require robust IDE support for code management, debugging, and deployment [3].

Integrated Server Management

IDEs for J2EE, such as Eclipse and IntelliJ IDEA, offer built-in server management tools, allowing developers to deploy their applications directly to web servers like Apache Tomcat or Glassfish [6].

Database Integration

J2EE applications often interact with relational databases. IDEs offer tools to streamline this interaction, from generating SQL queries to managing database connections.

Comparison of J2EE IDEs

Comparative analysis of popular J2EE IDEs, focusing on their effectiveness in managing enterprise-level applications. The study highlights Eclipse's flexibility, as it can be configured with various enterprise plug-ins, while IntelliJ IDEA is praised for its intuitive user interface and advanced code analysis tools [2].

ONLINE IDEs AND REMOTE EDUCATION DURING COVID-19

Shift to online Learning

The COVID-19 pandemic forced educational institutions to rapidly adapt to online teaching. For programming courses, this shift necessitated the adoption of online IDEs that could be accessed from anywhere, without the need for local installations [4].

Comparison of Online IDEs

Evaluated online IDEs such as Replit, CodeSandbox, and others: These platforms offer features like code collaboration, integrated debugging, and cloud-based execution, making them ideal for remote learning environments [7].

Collaboration Tools

Online IDEs enable real-time collaboration, allowing students to work together on projects from different locations.

Ease of Use

Online IDEs often have simplified interfaces, making them accessible to students with limited programming experience.

Limitations and Challenges

Despite their benefits, online IDEs also face limitations, particularly in terms of performance, scalability, and the range of features compared to traditional desktop IDEs. Additionally, online IDEs can struggle with real-time debugging and offer limited offline support, which can impact learning outcomes for more advanced students.

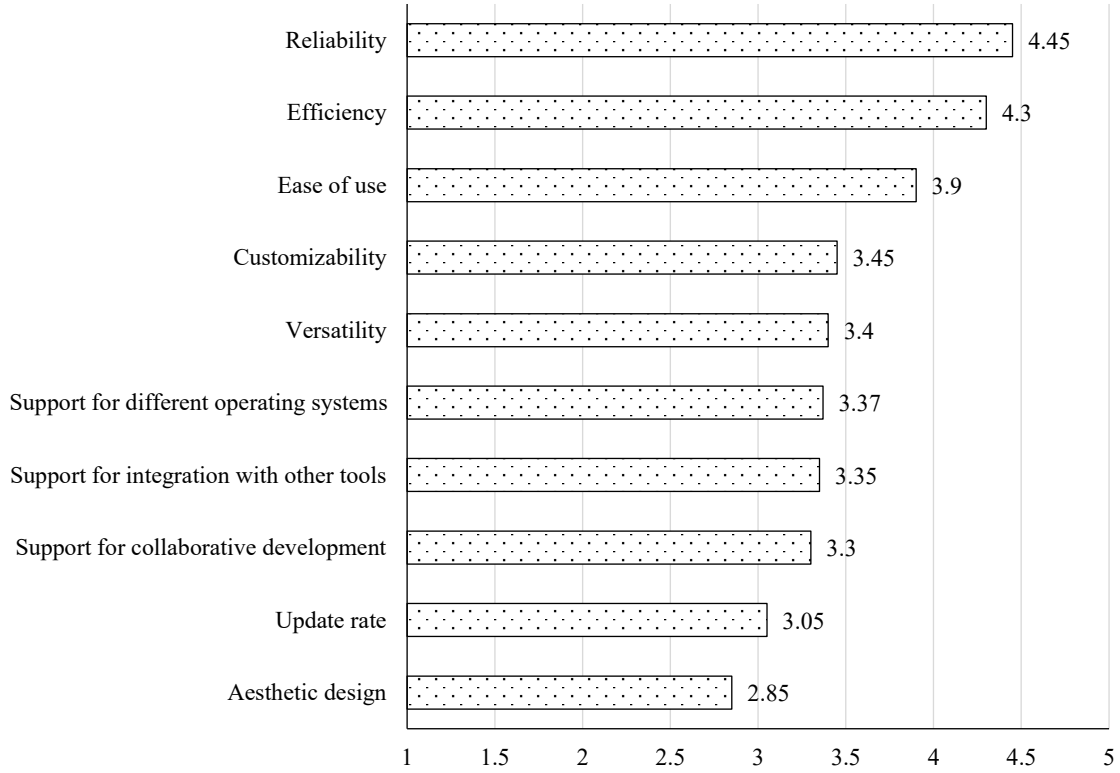

Figure 1. Average rating of importance of qualities.

SATISFACTION WITH IDE AND NEEDED IMPROVEMENTS

Participants were requested to evaluate their overall satisfaction with the IDE they used. As illustrated in Figure 1, the responses indicate a clear trend, with approximately 70% of respondents expressing that they were 'very satisfied' with their chosen IDE [8].

Among the various IDE qualities evaluated by participants, 'reliability' received the highest average rating (4.45 out of 5), followed closely by 'efficiency' and 'ease of use'. In contrast, attributes such as 'aesthetic design', 'update frequency', and 'collaborative development support' were rated as less important. Beyond the numerical ratings, participants who had personally selected their IDEs were also asked to explain their reasons for doing so [7]. Of the 14 individuals surveyed, six responded to this question. Four of them, who had opted for Eclipse, cited its 'customizability', 'versatility', and 'reliability' as key factors. The remaining two respondents indicated that their IDE choice was constrained by the specific platform requirements of their projects [9, 10].

CONCLUSION AND FUTURE WORK

This study has explored the development and applications of different Integrated Development Environments (IDEs), highlighting their progress over time, their use in embedded systems and J2EE projects, and their growing importance in education, especially during the COVID-19 pandemic. Tools like Eclipse have shown how adaptable open-source IDEs can be, while cloud-based IDEs have played a critical role in supporting remote learning environments. Looking ahead, the integration of artificial intelligence and machine learning into these platforms has the potential to greatly improve the developer workflow by offering smart suggestions, early bug detection, and better code performance. Moving forward, it is important that future studies aim to bridge the gap between online and desktop IDEs, making web-based platforms just as powerful and feature-rich. Improving these tools is essential to keeping pace with the changing demands of software development for both students and industry professionals.

REFERENCES

1. Wiegand J. Eclipse: A platform for integrating development tools. IBM Syst J. 2004; 43(2): 371–83.
2. Ertl D, Krapfenbauer H. A case study of developing an IDE for embedded software using open source. In 2009 IEEE Fourth International Conference on Software Engineering Advances. 2009 Sep 20; 191–196.
3. Dagdeviren H, Juric R, Ogunleye O, Tesanovic I. Analysis of integrated development environments for J2EE applications. In Proceedings of the 11th IASTED International Conference on Software Engineering and Applications. Anaheim, CA, USA: ACTA Press; 2007 Nov 6; 349–354.
4. Kusumaningtyas K, Nugroho ED, Priadana A. Online integrated development environment (ide) in supporting computer programming learning process during covid-19 pandemic: A comparative analysis. International Journal on Informatics for Development (IJID). 2020; 9(2): 66–71.
5. Mavroudi A, Economides AA, Fragkou O, Nikou SA, Divitini M, Giannakos M, Kameas A. Motivating students with Mobiles, Ubiquitous applications and the Internet of Things for STEM (MUMI4STEM). In 2017 IEEE Global Engineering Education Conference (EDUCON). 2017 Apr 25; 37–38.
6. Zaman M, Puryear N, Abdelwahed S, Zohrabi N. A review of IoT-based smart city development and management. Smart Cities. 2024 Jun 20; 7(3): 1462–501.
7. Joshi K, Mudliar P. Reselling Practices in a Textile Bazaar: Translating E-Commerce Platforms to WhatsApp. In Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems. 2025 Apr 26; 1–18.
8. Satria H, Wibowo B, Kwon JB, Lee JB, Hwang YS. VDEES: A virtual development environment for embedded software using open source software. IEEE Trans Consum Electron. 2009 May; 55(2): 959–66.
9. Portillo-Rodríguez J, Vizcaíno A, Piattini M, Beecham S. Tools used in Global Software Engineering: A systematic mapping review. Inf Softw Technol. 2012 Jul 1; 54(7): 663–85.
10. Lin B, Zampetti F, Bavota G, Di Penta M, Lanza M, Oliveto R. Sentiment analysis for software engineering: How far can we go? In Proceedings of the 40th international conference on software engineering. 2018 May 27; 94–104.

