International Research Journal on Advanced Engineering and Management
https://goldncloudpublications.com
https://doi.org/10.47392/IRJAEM.2025.0460

e ISSN: 2584-2854
Volume: 03
Issue: 09 September 2025
Page No: 2915 - 2918

The Evolution of Bug Tracking Systems: A Review of Advancements in Software Quality

Dr. Pushparani.MK[1], Disha K[2], Kruthika S[3], Ranjitha S[4], Spoorthi N Naik [5]
[1]HOD, Dept. of CSD, Alva's Institute of Engg. & Tech., Moodbidri, Karnataka, India.
[2,3,4,5]UG Scholar, Dept. of CSD, Alva's Institute of Engg. & Tech., Moodbidri, Karnataka, India.
Emails: drpushparani@aiet.org.in[1], dishakumble08@gmail.com[2], skruthika21@gmail.com[3], ranjithas9481@gmail.com[4], nnaikspoorthi@gmail.com[5]

Abstract

Managing software defects has always been a crucial challenge within the software development lifecycle. This paper reviews major advancements in bug tracking systems (BTS) by consolidating findings from recent research. Rather than merely describing these systems, it highlights three areas of innovation: the transition from generic to specialized bug management platforms, the use of context-aware algorithms for detecting duplicate reports, and the role of visualization techniques in handling continuous streams of bug data. The discussion also considers human factors and the growing influence of machine learning in automated triage and predictive analysis. The study concludes that BTS are evolving from static repositories into intelligent systems that reshape how bugs are reported, managed, and resolved, thereby improving efficiency and overall software quality.

Keywords: *Duplicate Bug Report Detection, Information Overloading, Software, Defects/Bugs, Human Factors.*

1. Introduction

With software systems growing increasingly complex, efficient bug identification and resolution have become a central priority for developers and project managers. A bug tracking system (BTS) acts as the backbone of this process, providing a structured environment to log, prioritize, and resolve software issues [1]. While the fundamental role of BTS—tracking the status of defects from discovery to closure—remains unchanged, the requirements of modern development demand more advanced and flexible solutions. Traditional, generalized platforms often fail to handle the scale and dynamic needs of today's projects. 1. The reviewed literature emphasizes the necessity for BTS that are more adaptive, intelligent, and user-oriented [1, 2, 3]. This review connects contributions from multiple studies to form a comprehensive understanding of current challenges and solutions in bug tracking. It focuses particularly on duplicate bug reports, the impact of information overload, and the potential of machine learning and human factors to shape the future of defect management society. The core function of any BTS is to manage the lifecycle of a software defect. This lifecycle generally includes the following stages: discovery (when a bug is found), reporting (when it's logged in the system), analysis (when a developer investigates it), prioritization (determining its importance), resolution (fixing the bug), and verification/closure (confirming the fix and closing the report).

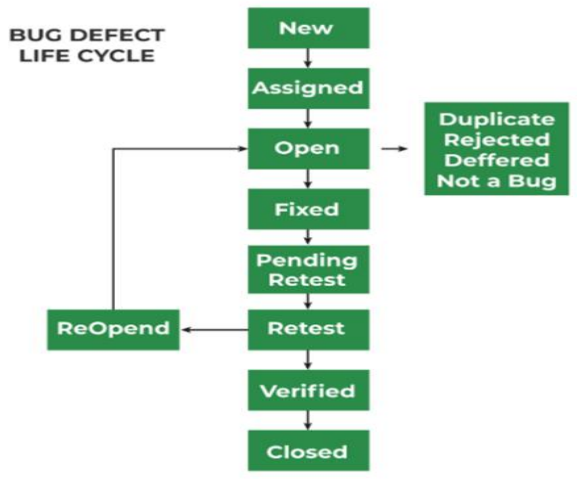

Figure 1 Bug Defect Lifecycle

2. Limitations of Conventional Bug Trackers

A BTS is intended to act as a centralized hub for defect management, covering processes such as reporting, triage, assignment, and verification [1]. While such systems provide visibility into bug ownership and progress, many current tools—commercial and open-source—are inadequate for the diversity of modern applications. The authors of BUGTRACK highlight a gap in existing solutions, noting that they often lack flexibility for different project needs [1]. Instead of forcing teams to perform rigid workflows, future workflows, future systems should support customizable and adaptable architectures. This points towards the development of platform-agnostic solutions that can handle a variety of bug types and reporting styles, ensuring scalability for diverse software environments.

3. Advances in Duplicate Bug Report Detect

Duplicate bug reporting is one of the costliest inefficiencies in large projects. Users and testers frequently submit the same issue because they lack the time or tools to search for existing reports [2]. Consolidating these duplicates consumes significant time and resources. Earlier approaches used basic information retrieval (IR) methods, which relied on text similarity. However, these methods struggle with semantic variations—for example, two reports may describe the same issue using entirely different wording. Alipour et al. propose a more advanced solution that incorporates domain knowledge, architectural insights, and topic modeling (e.g., Latent Dirichlet Allocation) [2]. Their approach identifies duplicates more effectively, reducing triage effort and enabling teams to focus on complex cases. This work demonstrates that future BTS must be context-aware, leveraging semantic understanding to manage redundancy.

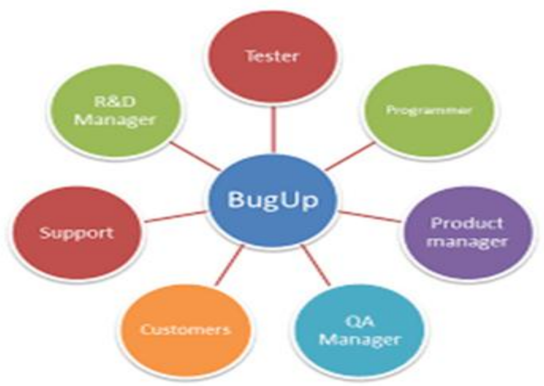

Figure 2 BugUp

4. The Role of Information Visualization.

Modern development teams deal with an overwhelming volume of bug-related updates. As noted by Takami and Kurosawa, continuous streams of textual updates often result in information overload, particularly for developers returning to a project after some absence [3]. Their study introduces visualization techniques that represent bug relationships dynamically. Animated displays can replay event sequences and highlight new updates, helping developers regain project context quickly. This goes beyond static dashboards or email notifications, turning raw data into an interactive, intuitive format. Such approaches illustrate a broader trend: integrating visualization into BTS to improve comprehension, reduce cognitive load, and enhance productivity.

5. Human Factors and the rise of Information Machine Learning

The effectiveness of BTS also depends heavily on the people using them. Clear, detailed bug reports are critical for efficient resolution, making user-friendly reporting essential. Looking ahead, machine learning (ML) is expected to drive the next wave of BTS. advancements. Beyond duplicate detection, ML can automate triage by analyzing bug reports and assigning them to the most relevant developers based on expertise and history. It can also estimate severity and priority levels, ensuring managers focus on critical issues first. By combining contextual understanding with predictive analytics, ML has the potential to transform BTS from reactive tools into proactive, intelligent assistants.

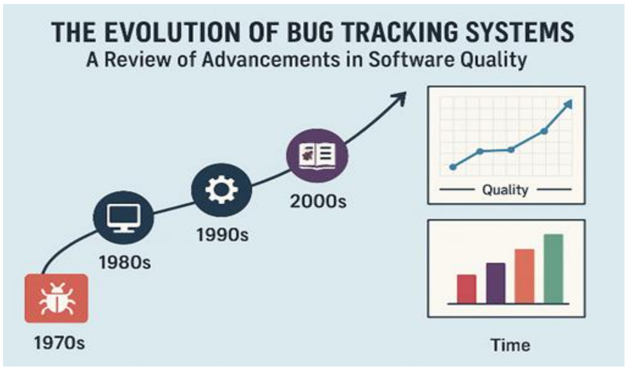
THE EVOLUTION OF BUG TRACKING SYSTEMS
A Review of Advancements in Software Quality
1970s
1980s
1990s
2000s
Quality
Time

Figure 3 The Evolution of Bug Tracking Systems: A System of Advancements in Software Quality

6. Results And Discussion

6.1. Results

Proactive Deduplication: Modern systems are moving beyond basic text-based similarity checks to use context-aware algorithms for identifying duplicate bug reports. Techniques like Latent Dirichlet Allocation (LDA) and the integration of domain and architectural knowledge are improving accuracy and reducing the significant time and effort spent on consolidating duplicate reports. Enhanced Visualization: The problem of information overload is being addressed through visualization techniques that turn continuous streams of bug data into intuitive, interactive formats. These animated displays and visualizations help developers quickly regain project context, reducing cognitive load and improving productivity, which goes beyond static dashboards or email notifications. Machine Learning Integration: Machine Learning (ML) is an influential force shaping the future of BTS. Beyond simple duplicate detection, ML can automate triage by analyzing bug reports to assign them to the most relevant developers based on their expertise and history. It also has the potential to estimate severity and priority levels, enabling managers to focus on critical issues first.

6.2. Discussion

The findings confirm that the evolution of bug tracking is a response to the increasing complexity of software systems and the inherent challenges of modern development, such as large data volumes and information overload. The reviewed literature consistently emphasizes the need for more adaptive, intelligent, and user-oriented BTS. The move from generic to specialized platforms reflects this need for more flexible and customizable workflows that support diverse project needs. The identified innovations collectively aim to transform BTS from reactive tools into proactive, intelligent assistants. The human factor remains crucial, as the effectiveness of any BTS ultimately depends on the quality of bug reports submitted. Therefore, the push for human-centric design and intuitive interfaces is essential for encouraging accurate reporting and ensuring that the intelligent features, like automated triage, have reliable data to work with. The integration of ML and advanced algorithms demonstrates a shift toward automating manual, time-consuming tasks like duplicate report consolidation and initial triage, allowing developers to focus on actual bug resolution rather than administrative overhead. The future of BTS is characterized by a blend of technological sophistication and a focus on improving the user experience, ultimately leading to greater efficiency and higher software quality. This progression indicates that BTS are becoming an indispensable part of the development ecosystem by actively enhancing productivity rather than just tracking issues.

Conclusion

Bug tracking has advanced from simple defect logging to a complex, data-driven discipline. Challenges such as high data volume, duplicate reporting, and developer overload have spurred significant innovations. The future direction of BTS can be summarized in four principles:

- **Intelligent Triage:** Automating prioritization and assignment through machine learning. Proactive Deduplication: Leveraging semantic and contextual analysis to detect redundant reports.
- **Visual Communication:** Using interactive visualization to provide real-time insight into bug activity.
- **Human-Centric Design:** Ensuring intuitive workflows that encourage accurate reporting. Adopting these principles will enable next-generation BTS to not only track issues but actively enhance productivity and software

e ISSN: 2584-2854
Volume: 03
Issue: 09 September 2025
Page No: 2915 - 2918

quality, making them indispensable in modern development ecosystems.

References

Journal reference style:

[1]. M. K. Gopal, P. Shetty, G. M., S. Raj, and P. 2. Chandra, "BUGTRACK – A New Improved Bug Tracking System," in 2022 IEEE Delhi Section Conference (DELCON), 2022.

[2]. A. Alipour, A. Hindle, and E. Stroulia, "A Contextual Approach towards More Accurate Duplicate Bug Report Detection," in Proceedings of the 10th Working Conference on Mining Software Repositories (MSR), 2013.

[3]. Y. Takama and T. Kurosawa, "Application of Monitoring Support Visualization to Bug Tracking Systems," in Proceedings of the 3rd International Conference on Pervasive Computing and Applications (ICPCA), 2008.