Intelligent Issue Assignment and Staff Efficiency Evaluation in Enterprise Issue Tracking System

Ritu Dagar
Dept. of Computer Science & Engg.
SRM University
Sonipat, Haryana

Naman Jain
Dept. of Computer Science & Engg.
SRM University
Sonipat, Haryana

Satwik Verma
Dept. of Computer Science & Engg.
SRM University
Sonipat, Haryana

Vaibhav Saini
Dept. of Computer Science & Engg.
SRM University
Sonipat, Haryana

ABSTRACT
In big companies, internal problems with the IT support, system maintenance and service requests are often reported. So, the conventional way of resolving these problems is through manual communication means like emails, spreadsheets or face-to-face interaction often causes delays in issue resolution, inadequate tracking and also unequal workload distribution. In the case of enterprise environments, they also have issue trackers that are in use to handle issues well. This paper, proposes IntelliTrack, an intelligent issue tracking system, designed to assist the organization to automate the issue assignment process. The system presents an algorithm of Priority-Weighted Staff efficiency which measures the priority of issues, the amount of work the staff have to do and the past performance of the staff to allocate the issue to the most appropriate staff. The system proposed will be able to create a centralized place for reporting, tracking and resolving issues, hence balancing workloads and increasing efficiency in issue handling.

Keywords
Issue tracking, Automated Assignment, Staff Efficiency, Workload Balancing, Enterprise Systems.

1. INTRODUCTION
In large companies, it is challenging to manage the internal problems effectively. These can range from system failures, technical support requests, to service-related issues, or any other issues that the technical support team find it necessary to address. As In many organizations, issue reporting and assignment are still handled manually through emails or by messaging platforms. So the administrators or team leads have to go through the issues reported and manually allocate them to staff members, which becomes inefficient as numbers of issues keep rising. In the software development setting, issue tracking systems are common tools that are used to effectively manage the track issues including Jira, Bugzilla and GitHub issues [1], [2]. However, assignment of issues by way of manual process poses several challenges, such as workload distribution of the staffs and inefficient handling of critical issues. There is existing work that has discussed ways to automatically assign issues and triage bugs using machine learning and data driven methods [3] and [4] respectively. These strategies aim at improving accuracy of assignments by analysing the description of the issues and previous records. There are, however, many current approaches that use text analysis mainly and that do not take into account many factors such as workload balancing and staff performance[4]. To overcome these drawbacks, this research presents an Intelligent issue tracking system named IntelliTrack which suggests a candidate solution by automating the distribution of issues based on priority, experience in a user, workload and historical performance. The proposed IntelliTrack system integrates an efficiency paradigm based on priority levels and a mechanism for workload-aware assignments to guarantee the equitable allocation of the workload among people and promote system's efficiency.

2. RELATED WORK
In software development environments, the issue tracking system plays an important role for efficient management of the bug reports and service requests [1]. Teams can use tools like Jira, Bugzilla, and GitHub issues to keep an eye on the reported bugs and run their tasks successfully. There are multiple research that have investigated automated issue assignment techniques. The first few attempts were based on bug triaging with machine learning algorithm like Naïve Bayes and Support Vector Machines to recommend appropriate developers [2] [3] [7]. Recent work includes the use of data-driven and historical analysis techniques that leverage issues that were previously solved and developer usage patterns to enhance the accuracy of the assignment [4], [6], [8]. These are methods that seek to improve decision making through the learning from data on issues resolved in the past. Furthermore, studies on mining software repositories and developer activities have enabled the acquisition of knowledge that could be used to enhance the ability of assigning issues and collaborating in software projects [5, 9]. Well-structured bug reports are also essential to make assigning more efficient and the quality of bug fixing better [10]. Current methods, however, mostly address textual description of issues and frequently fail to take into account other key considerations, like workload balancing, issue priority and staff performance [4]. To overcome these drawbacks, the proposed Intellitrack system presents a priority-based and performance-based assignment model which takes multiple parameters into consideration for an efficient assignment.

3. METHODOLOGY
The proposed system is a centralized issue management system named IntelliTrack which helps the organization to present their internal issues in an efficient manner. The system automatically sets up the assignment of the issue, based on various factors including issue priority, staff skills, workload and historical performance. The methodology used in IntelliTrack consists of three main components which includes: system workflow, staff efficiency and assignment decision model.

3.1 System Overview
Before an issue can be assigned to a specific staff member, the system will go through a few steps to determine what happened when the report is made. A considerable amount of data is typically generated in an issue tracking system that can be used

to allocate tasks and analyze them [1], [4]. The key steps identified are:

- Receiving the issue in the system database Recognizing its priority level.
- Organizing the staff by skills and expertise
- Identifying existing workloads of each member of staff.
- Establishing the efficiency of staff on the basis of their previous work.
- The automatic allocation of the issue to the best staff member.
- This automated process reduces manual effort and ensures fair and efficient task allocation.

This automated process eliminates manual work and guarantees task allocation is carried out appropriately and fairly.

Fig. 1. IntelliTrack System Workflow

3.2 Staff Efficiency Model

Please use a 9-point Times Roman font, or other Roman font with serifs, as close as possible in appearance to Times Roman in which these guidelines have been set. The goal is to have a 9-point text, as you see here. Please use sans-serif or non-proportional fonts only for special purposes, such as distinguishing source code text. If Times Roman is not available, try the font named Computer Modern Roman. On a Macintosh, use the font named Times. Right margins should be justified, not ragged.

$$\Sigma\ (Priority\ Weight\ Resolved\ Issues)$$

$$Efficiency\ (\%)\ =\ ---------\times\ 100$$
$$\Sigma\ (Priority\ Weight\ \times\ Assigned\ Issues)$$

$$\ldots\ldots\ldots\ldots(1)$$

This formula allows that high priority issues have larger impact on the efficiency score than low priority issues.

Priority Weight mapping:

Table 1. Priority Weight mapping

Priority Level	Weight
Critical	4
High	3
Medium	2
Low	1

The system will be balanced by analyzing the workload of all staff. The workload score is determined by:

$$Workload\ Score\ =\ Active\ Issues\ /\ TotalCapacity\ \ldots\ldots\ldots\ldots\ldots\ldots(2)$$

The lower the workload score, the greater the availability to address new issues.

3.3 Assignment Decision Model

Issues are assigned through a multi-step decision process with the IntelliTrack system. The priority of the issue is determined first. After that the members of staff are filtered according to the skills they need. A comparison is made with the successful team members' efficiency score, which is determined on the basis of historical data, and an assessment is made of the workload of each member of the staff. The final assignment decision is made based on the following:

$$Score\ =\ \alpha\ \times\ Efficiency\ +\ \beta\ \times\ (1\ -\ Workload)\ +\ \gamma\ \times\ Priority$$

$$\ldots\ldots\ldots\ldots\ldots\ldots(3)$$

The weighting factors α, β, and γ indicate the importance of each of the parameters. Existing literature has been done about multi factor decision making models [4], [5]. The task at hand is assigned to the staff member who scores most:

$$Assigned\ Staff\ =\ argmax(Score)$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(4)$$

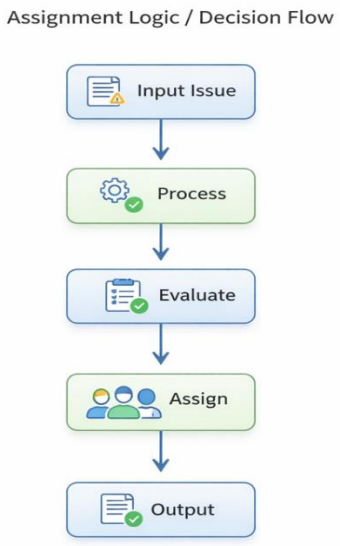

Fig. 2. Assignment Decision Flow in IntelliTrack

The process of assignment is done in the following steps:

1. Identify issue priority

2.Here filter workers according to their experience.

3. Calculate efficiency score

4. Evaluate workload

5. Compute final score

6. Rank staff members

7. Assign issue to highest scored staff member

4. SYSTEM ARCHITECTURE

The IntelliTrack system is designed on the basis of the principle of modular lay-out, whereby different functional components are separated from each other to ensure scalability, maintainability and efficient processing problems. The Architecture includes three layers - the Presentation layer, Application layer and Data layer.

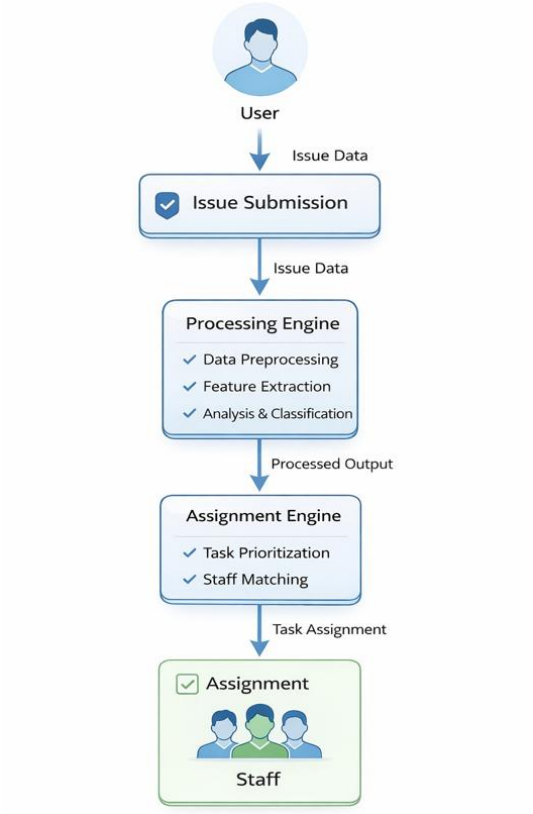

Fig. 3. System Architecture of IntelliTrack

The presentation layer allows users to report and track issues in a straightforward manner, while the application layer takes care of the main functions like processing issues, evaluating their priority, and deciding how they should be assigned. On top of that, the data layer plays a crucial role by storing issue records, staff information, and historical data, which altogether helps in achieving more intelligent and efficient task allocation [4], [5]. The layered architecture is a practical and effective approach, as it ensures the system remains scalable, modular, and capable of performing efficiently as system demands grow over time.

5. IMPLEMENTATION

The IntelliTrack system employs current Web technologies to achieve scalability, performance and integration into enterprise environments. Such a system has been designed Works according to a modular design with various components performing individual tasks like issue management, processing and assignment. The frontend is built with Next.js to deliver a fast and responsive interface for creating and tracking issues, ensuring a smooth user experience. The backend extends the infrastructure using the nodeJs platform to process assignments, manage api communication and implement backend logic. The issue database uses MongoDB for it's records, staff records are stored in the same database, along with historical performance data. Gain secure access control and authentication based on middleware-based techniques. Moreover, it also has notification facility by linking with Gmail API to ensure that workers are updated instantly every time they are assigned with tasks. The above technologies are selected for the management of large volume of issues quickly as well as to adhere real-time decision-making process, which is a certain way in modern issue tracking system [4]. Table 2 shows the technologies used to employ the IntelliTrack system.

Table 2. System Implementation Technologies

Component	Technology
Frontend	Next.js
Backend	Node.js
Database	MongoDB
Authentication	Middleware-based authentication
Notifications	Gmail API

These technologies can be deployed to provide enterprises with high performance and reliability, even when they are deployed on a large scale.

6. RESULT AND DISCUSSION

The IntelliTrack system is a significant performance improvement over the conventional methods of assigning issues. The system automates the assignment process, thereby saving time and workload during the issue resolution process. The score is now being determined on a scale of 1–4 and this applies to the qualitative performances. Reporting the number of test cases executed, the number passed and failed, and the resulting success rate for each module. Of the 70 total test cases executed, 64 passed and 6 failed, yielding an overall system success rate of 91.43%.

Table 3. Module-wise test case summary and success rate.

Module	No. of Test Cases	Passed	Failed	Success Rate (%)
Login	5	5	0	100.00
Registration	5	4	1	80.00

Module	No. of Test Cases	Passed	Failed	Success Rate (%)
Dashboard	5	5	0	100.00
Create Issue	6	5	1	83.33
Automatic Issue Assignment	8	7	1	87.50
Manual Assignment	5	5	0	100.00
Issue Status Update	6	5	1	83.33
Comments	5	5	0	100.00
Search & Filter	6	5	1	83.33
Notifications	5	5	0	100.00
Report Generation	5	5	0	100.00
User Management	6	5	1	83.33
Logout	3	3	0	100.00
Total	**70**	**64**	**6**	**91.43**

The proposed priority-based efficiency model provides that serious issues are allocated to more experienced worker, whereas the workload aware mechanism prevents from overloading the worker. This helps to spread tasks across the system evenly and increase overall system efficiency. IntelliTrack benefits over the existing methods like manual allocation and popular bug triaging techniques by considering many factors, such as workload, staff efficiency and priority, which result in more accurate and fair task allocation [2],[4]. In addition, the use of historical data will further improve decision making and be more consistent in assignments. Table 3 shows a comparison between the different issue-assignment methods and the advantage offered by the proposed IntelliTrack system.

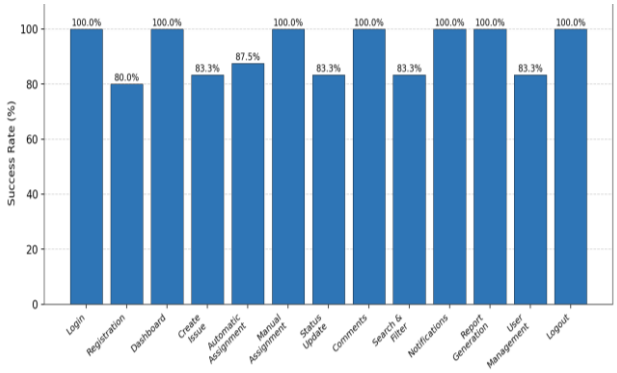

Fig. 4 Module-wise success rate (%) across the thirteen functional modules.

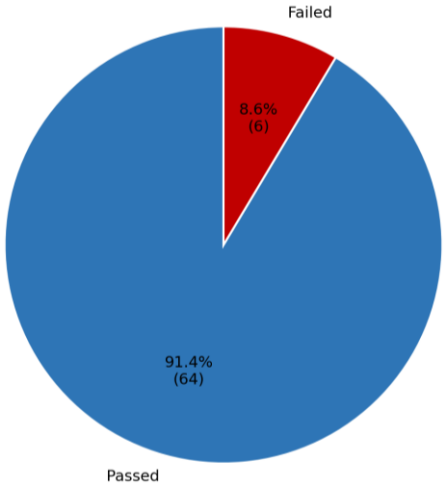

Fig 5. Overall distribution of passed versus failed test cases (n = 70).

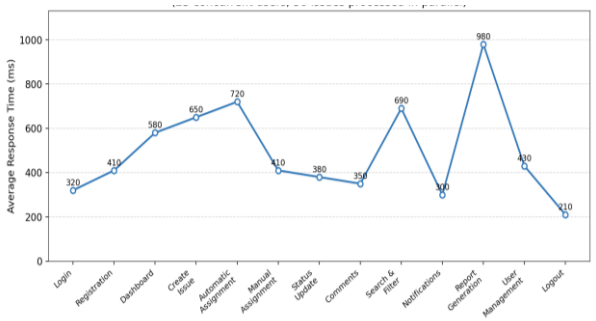

Fig 6. Average API response time (ms) per module under 25 concurrent users and 50 parallel issues

To execute the load and performance discribed or showed above , a combination of open source testing tools were used.

1. Apache J meter to simulate the users like we have tested over 25 users for scalability

2. K6 was used to test the load and traffic means that software sustained when many user use it at once.

3. Postman for api endpoints performance.

Table 4. Comparative Analysis of Issue Assignment Approaches

Approach	Method Type	Factors Considered	Workload Handling	Automation Level
Manual Assignment	Human-based	Limited	Poor	Low
Bug Triaging Models [2], [3]	Machine Learning	Text-based	Limited	Medium
Data-driven Approaches [4]	Historical Analysis	Partial	Moderate	Medium
IntelliTrack (Proposed)	Hybrid Model	Priority + Efficiency + Workload	Balanced	High

The performance comparison also shows that IntelliTrack is more efficient, and more automated than traditional and existing processes.

Obviously, IntelliTrack's implementation is better than the traditional methods because it combines multiple decision factors and makes efficient workload distribution.

7. CONCLUSION AND FUTURE WORK

In this paper, an intelligent issue tracking system called Intellitrack which automates issue assignment by making use of a priority weighted staff efficiency model is proposed. The proposed system solves various problems associated with traditional issue management, such as manual task assignment, workload imbalance, and transparency issues. IntelliTrack supports effective and a balanced assignment of tasks taking into account various features like staff skills, workload, historical performances, etc., and issue done priorities. These results show the feasibility of achieving an efficient operation and equitable allocation of work and tasks between the different operators compared to current [2]–[4] systems. The centralized platform also enables organizations to follow the entire process of issues, promoting responsibility and decision-making. The system can be expanded by using machine learning algorithms in future to calculate the probable time to resolve the issue and classify the issues according to its descriptions automatically. Other features you may want to consider are advanced analytics dashboards, adaptive experience evaluation system with dynamic daily evaluations through real-time.

8. REFERENCES

[1] J. Anvik, L. Hiew, and G. C. Murphy, “Who should fix this bug?” in *Proceedings of the 28th International Conference on Software Engineering (ICSE)*, Shanghai, China, 2006, pp. 361–370.

[2] G. Jeong, S. Kim, and T. Zimmermann, “Improving bug triage with bug tossing graphs,” *IEEE Transactions on Software Engineering*, vol. 38, no. 5, pp. 1115–1131, 2012.

[3] P. Bhattacharya and I. Neamtiu, “Fine-grained incremental learning and multi-feature tossing graphs to improve bug triaging,” in *Proceedings of the IEEE International Conference on Software Maintenance (ICSM)*, Timisoara, Romania, 2010, pp. 1–10.

[4] F. Zhang, I. Keivanloo, and Y. Zou, “Data-driven approaches for issue assignment in software projects,” in *Proceedings of the IEEE International Conference on Software Maintenance and Evolution (ICSME)*, Raleigh, USA, 2016, pp. 187–198.

[5] A. Mockus, R. T. Fielding, and J. Herbsleb, “Two case studies of open-source software development: Apache and Mozilla,” *ACM Transactions on Software Engineering and Methodology*, vol. 11, no. 3, pp. 309–346, 2002.

[6] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan, “Mining email social networks,” in *Proceedings of the 2006 International Workshop on Mining Software Repositories (MSR)*, Shanghai, China, 2006, pp. 137–143.

[7] X. Xia, D. Lo, X. Wang, and X. Yang, “Who should fix this bug? Automatically recommending developers based on bug reports,” in *Proceedings of the IEEE International Conference on Software Maintenance (ICSM)*, Eindhoven, Netherlands, 2013, pp. 361–370.

[8] S. Wang, D. Zhang, and Y. Zhang, “Developer recommendation for bug resolution: A ranking approach,” in *Proceedings of the IEEE International Conference on Software Maintenance and Evolution (ICSME)*, Victoria, Canada, 2014, pp. 285–294.

[9] R. Shokripour, J. Anvik, and B. Adams, “Why so complicated? Simple term filtering and weighting for bug triage,” in *Proceedings of the 10th Working Conference on Mining Software Repositories (MSR)*, San Francisco, USA, 2013, pp. 2–11.

[10] N. Bettenburg, S. Just, A. Schröter, C. Weiss, R. Premraj, and T. Zimmermann, “What makes a good bug report?” in *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE)*, Atlanta, USA, 2008, pp. 308–318.

[11] T. Zimmermann and N. Nagappan, “Predicting defects using network analysis on dependency graphs,” in Proceedings of the 30th International Conference on Software Engineering (ICSE), 2008.

[12] J. Jiang, L. Tan, and S. Kim, “Personalized defect prediction,” in Proceedings of the 25th IEEE/ACM International Conference on Automated Software Engineering (ASE), 2010.

[13] E. Murphy-Hill, T. Zimmermann, and N. Nagappan, “Cowboys, ankle sprains, and keepers of quality,” in Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering (ICSE), 2010.

[14] T. Menzies, Z. Milton, B. Turhan, B. Cukic, Y. Jiang, and A. Bener, “Defect prediction from static code features,” Automated Software Engineering, vol. 17, no. 4, pp. 375–407, 2010.

[15] N. Nagappan and T. Ball, “Using software dependencies and churn metrics to predict field failures,” in Proceedings of the 1st International Symposium on Empirical Software Engineering, 2002.

[16] X. Xia, D. Lo, E. Shihab, X. Wang, and B. Zhou, "Automated bug report assignment using multi-feature tossing graphs," IEEE Transactions on Software Engineering, 2015.

[17] D. Cubranic and G. C. Murphy, "Automatic bug triage using text categorization," in Proceedings of the 16th International Conference on Software Engineering & Knowledge Engineering (SEKE), 2004.