Educational Administration: Theory and Practice
2024, 30(5), 7391 - 7400
ISSN: 2148-2403
https://kuey.net/

Research Article

GA Based Task Allocation In Agile Software Development InDistributed Environment

Pratyush Nayak[1], Junali Jasmine Jena[2*], Minakhi Rout[3]

[1,2*,3]School Of Computer Engineering, KIIT Deemed to be University, Bhubaneswar, India.
pratyush2244@gmail.com, junali.jenafcs@kiit.ac.in, minakhi.routfcs@kiit.ac.in

Citation: , Junali Jasmine Jena et.al (2024), GA Based Task Allocation In Agile Software Development InDistributed Environment *Educational Administration: Theory and Practice, 30(5), 7391 - 7400*
Doi: 10.53555/kuey.v30i5.4166

ARTICLE INFO

ABSTRACT

Distributed Agile Software Development (DASD) has emerged as a dynamic paradigm aimed at efficiently deliver- ing software products while tapping into global opportunities and ensuring customer satisfaction. However, within DASD teams, effective task allocation presents multifaceted challenges due to geographical dispersion, cultural diversity, and communication barriers. This paper addresses these challenges by proposing a quantitative approach to task allocation specific to DASD. The re- search delves into identifying the factors influencing DASD development processes and their intricate interdependencies. Notably, it explores dependencies within Agile teams and across distributed sites, underlining the crucial role of effective coordination in preventing rework and fostering team cohesion. Emphasis is placed on the expertise levels of team members as a pivotal deter- minant in task allocation, with the proposed method aiming to assign tasks to the most qualified experts. The paper presents a structured framework for task allocation in DASD, recognizing the complexity inherent in distributed teams and dependencies. By offering a quantitative method, this research contributes to enhancing the efficiency and quality of software development within the context of DASD, ultimately leading to heightened customer satisfaction and increased project success.

Key Word: DASD, Task allocation, Genetic algorithm.

INTRODUCTION

Distributed Agile Software Development (DASD) merges Agile methodologies with globally dispersed teams, pre- senting challenges in effective task allocation due to geographical dispersion, cultural diversity, and communication barriers."This research introduces a quantitative approach to address these challenges, aiming to optimize task alloca- tion, resource utilization, and project outcomes in DASD." Task allocation within DASD teams is critical for managing project timelines, costs, and software quality. The proposed quantitative method considers various factors, including task requirements, team members' expertise levels, and historical performance data, to assign tasks effectively. By matching tasks with the most qualified experts, the approach seeks to enhance project efficiency and reduce rework. Coordination among distributed Agile teams is essential to prevent bottlenecks and foster team cohesion. Effec- tive communication tools and techniques are vital for bridging time zone differences and cultural diversity within DASD teams. Additionally, maintaining Agile principles while adapting to the distributed nature of teams is crucial for successful DASD implementation. Agile methodologies such as Scrum and Kanban promote iterative develop- ment, continuous feedback, and customer-centric focus. While Agile values remain fundamental, DASD requires adjustments to traditional Agile practices to accommodate distributed teams. By embracing remote collaboration techniques, organizations can deliver responsive software solutions while adhering to Agile principles. Overall, this research contributes to advancing task allocation practices in DASD, ultimately leading to improved customer sat- isfaction and project success. By providing a structured framework for task allocation, organizations can navigate the complexities of DASD more effectively, ensuring efficient resource utilization and timely project delivery. The proposed quantitative approach offers a valuable tool for optimizing task assignments, managing dependencies, and enhancing communication among distributed Agile teams.

RELATED WORK

Traditional Machine Learning Techniques

Task allocation within Distributed Agile Software Development (DASD) presents formidable challenges, including communication barriers, coordination issues, and accountability gaps. These challenges result in time wastage and distractions, particularly within globally dispersed teams, leading to longer task completion times compared to collo- cated teams. Self-organizing Agile teams in DASD environments face additional hurdles such as undefined acceptance criteria and task dependencies, causing disruptions and potential work cancellations.To address these issues, variousmethodologies that emerged are explained below.

In 2013, W.-N. Chen et al. introduced "Ant colony optimization for software project scheduling and staffing with an event-based scheduler [5]. The Event-Based Scheduler methodology introduced principles from ant colony optimiza- tion algorithms to optimize task scheduling, aiming to enhance efficiency by considering employee plans and task requirements. This approach mimicked nature's optimization techniques to improve task allocation. In 2014, J. Lin et al. conducted a study on task allocation decisions of novice agile teams using data from agile project management tools [11]. DASD (Distributed Agile Software Development) integrated Agile project management principles into task allocation within DASD, considering factors such as team competence, workload distribution, and confidence levels through exploratory data analysis to ensure improved task allocation and project success. In 2015, J. Sutanto et al. investigated task coordination in globally dispersed teams from a structural contingency perspective [12] , in- troducing a methodology centered on task coordination portfolios. This methodology leveraged IT-mediated optimal task synchronization portfolios, considering factors such as task dependencies and time constraints to optimize task allocation and synchronization across distributed teams. The Verbal Decision Analysis Method, introduced by R. Hoda et al. in 2016, offered a quantitative approach to task allocation [13] . It involved rank-ordering influencing factors to enhance decision-making. This method was outlined in the Journal of Systems and Software.This method facilitated informed decision-making and optimized allocation outcomes by assigning values based on significance. In 2017, W. Aslam et al. introduced "Risk aware and quality enriched effort estimation for mobile applications in dis- tributed agile software development " [14] , aiming to enhance software effort estimation and knowledge management through data and analytics methodologies. Various other Evolutionary algorithms could be used for task allocation such as Social Group Optimization[15], Differential Evolution[16], Particle Swarm Optimization[17] etc. Usefulness of these algorithms to task allocation could be proven from their range of applications. Task Allocation in large scale is a complex task and belong to the group of hard problems. Evolutionary algorithms have found a profound role in providing optimal or near optimal solutions to these problems like application of SGO to solve TSP[18], SGO for engineering design problems[19] etc. Evolutionary algorithms have got wide range of applications such as in civil engineering [20], Machine learning [21,25]. Not only Evolutionary algorithms but other new models could be used to solve such dynamic problems such as AI [22], Deep Neural Network[23].This could be modelled as a multi-objectiveproblem too and could be solved using multi-objective optimization[24]

Factors influencing task allocation in DSD

In Distributed Agile Software Development (DASD), task allocation is influenced by a wide array of factors spanning project and people-related considerations, expertise requirements, site characteristics, task attributes, and cost con- siderations. Effective coordination among distributed teams, prioritization based on business value, and addressing geographical and cultural differences are pivotal. The type of DSD environment, whether offshoring or outsourcing, affects communication protocols, while control structures, whether centralized or distributed, impact team interac- tions. These factors, classified as success or failure factors, shape task allocation efficiency. The influencing factors, categorized into six dimensions (Project, People, Site, Task, Agile, and Environment), encompass various specific aspects, including project complexity, communication skills, site specificity, task deadlines, and Agile principles. Un- derstanding and considering these dimensions and factors is fundamental for optimizing task allocation in DASD, ultimately influencing task allocation, project success, and the overall efficiency of software development endeavors.

Types of required capabilities for software roles

In Agile software development, roles like the Product Owner, Scrum Master, development team members, Quality Assurance/Testers, Architects, Designers, and Business Analysts require specific capabilities for effective task allo- cation. For instance, Product Owners need domain knowledge and stakeholder communication skills, while Scrum Masters require Agile expertise and conflict resolution abilities. Development team members should be technically proficient and collaborative, while Quality Assurance/Testers need testing expertise and attention to detail. Archi- tects and Designers must possess problem-solving skills and creativity, and Business Analysts require requirements analysis proficiency and business acumen. These capabilities align with Agile principles, facilitating efficient task

allocation, effective teamwork, and successful project outcomes. Additional roles and skills may be necessary de- pending on project requirements. Regarding the research methodology, various academic sources were considered, and data collection involved multiple stages, with 59 articles identified as primary studies relevant to the research. Reporting on secondary research questions involved discussing methodologies adopted by other authors, such as qualitative observations, case studies, surveys, interviews, and focus groups.

PROPOSED METHODLOGY

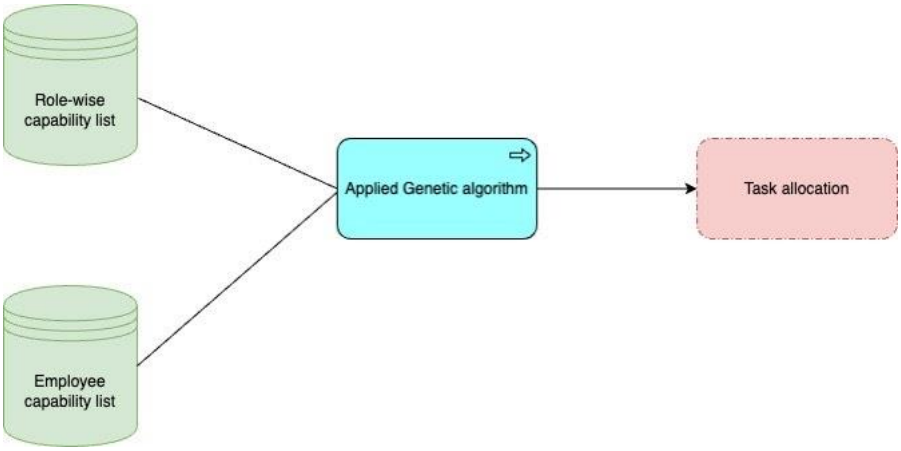

FIGURE 1. Proposed Application Model

Assigning human resources with different skill sets and efficiency factors to several projects that require different skill sets is a complex task. The proposed model shown in fig1 , collection of different human resources as per the required skill set of a project or task are represented as candidate solution or chromosome in the G.A. An initial population of potential staff allocations required for the project is generated, and a fitness function, denoted as Eq1, is employed to evaluate the quality of each allocation based on their efficiency factors , is considered :

$$\text{Fitness Function for Chromosome} = \frac{0.4 \times \sum Experience_i + 0.6 \times \sum Rating_i}{10}$$

Here we considered two factors i.e performance of the employee rated in the range of 1 to 10 and experience of the employee in years. Here we assume that employee with more rating and more experience or fitter than others. Stopping criteria is considered to be the maximum number of iteration provided as input by the user. By intelligently allocating task, enhancing team collaboration, and aligning project goals with customer needs, the model aims to enhance project success.

Genetic Algorithm for Task Allocation

The task allocation process in Agile software development, particularly in Distributed Agile environments, relies heavily on efficient methodologies such as genetic algorithms (GAs) to optimize task allocation, resource utilization,and project outcomes. Here's a breakdown of key components and their significance:

- **Task pool:** It is a list of tasks aggregated according to the skill set, performance rating, and years of experiencefor each member assigned to the task.
- **Chromosome Representation:** The chromosome serves as a potential solution to a particular project. It defines how tasks are distributed among team members. Choosing the right representation, Here we adopted array- based structures, significantly impacts algorithm performance.
- **Initialization:** The GA begins by generating an initial population of potential job allocations. This can involverandom assignment o approaches to kick start the learning process.
- **Fitness Function:** The fitness function is crucial in evaluating task allocations' effectiveness. It considers two factors are employee rating and employee experience. The goal is to evolve task assignments towards higher fitness values.
- **Crossover:** Crossover involves task exchange between different allocations. In task allocation, this could entail swapping tasks between team members to explore new combinations and improve assignment strategies.
- **Mutation:** one task from a specific allocation is randomly selected and replaced with another task from the taskpool.
- **Selection:** After cross-over and mutation the fitter candidate is selected for the next iteration. It favors solutionswith higher fitness, akin to "survival of the fittest."
- **Evaluation:** The priorities of employee experience and employee rating are assessed after evaluation.The flowchart shows how we assess employee experience and rating in task allocation,

prioritizing these factorsin our methodology, as described below.

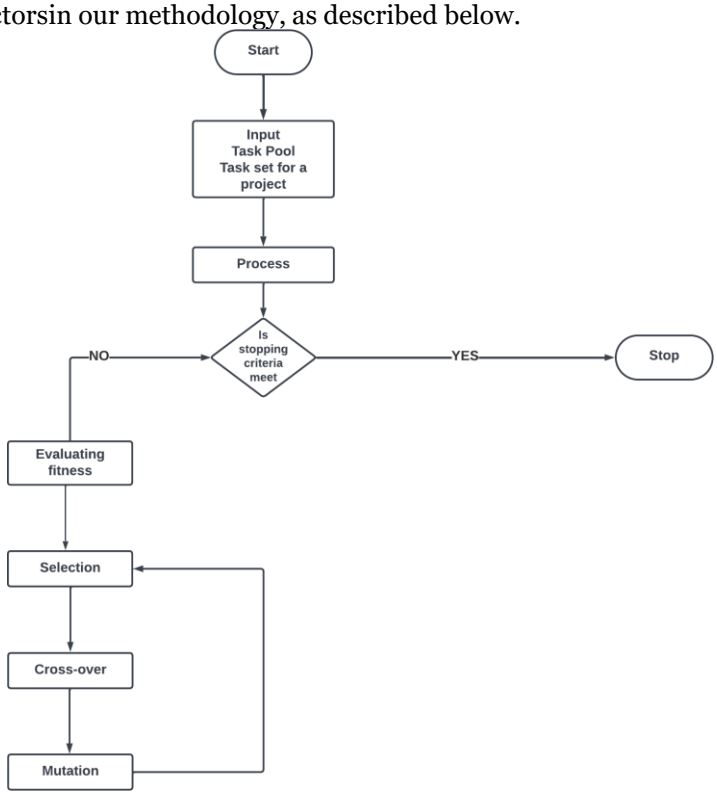

FIGURE 2. Flow Chart of the proposed methodology

ILLUSTRATING THE PROCESS

In the admin phase, we filter and add employees with the same skills in our database. Since many have similar skills, task assignments get tricky. There are many factors but we consider only two factors are employee rating and experience for the initial phase of our project. These help us choose the right candidates for tasks. In our project, suppose we randomly select 50 employees based on their ratings and experience. However, only 10 employees are required for the project. Therefore, we need to filter out the excess of 40 employees. Let's suppose we choose four random chromosomes, each consisting of 10 genes(employee), from the initial
population.
Chromosome 1: 10, 25, 37, 4, 49, 12, 30, 5, 22, 46.
Chromosome 2: 18, 43, 7, 36, 2, 15, 28, 50, 9, 4.
Chromosome 3: 33, 20, 48, 14, 3, 31, 11, 27, 45, 1.
Chromosome 4: 35, 16, 44, 24, 19, 47, 6, 39, 26, 8.
Chromosome 5: 13, 21, 34, 17, 38, 23, 29, 40, 41, 42.
In this context, each chromosome cannot have duplicate genes.
Fitness function = weighted sum average of exp and rating. The fitness function for a chromosome can be represented as:

$$\text{Fitness Function for Chromosome} = \frac{0.4 \times \sum Experience_i + 0.6 \times \sum Rating_i}{10}$$

where i = 1 to 10,and experience and rating will take from our database and here we priortize rating more.
FF for Chromosome 1 = 0.4 * (7 + 13 + 9 + 11 + 15 + 8 + 10 + 12 + 6 + 14) + 0.6 * (4 + 3 + 2 + 5 + 4 + 3 + 2 + 5 + 4 + 3) / 10 = 6.3.
FF for Chromosome 2 = (0.4 * 95 + 0.6 * 37) / 10 = 6.02.
FF for Chromosome 3 = (0.4 * 95 + 0.6 * 35) / 10 = 5.9
FF for Chromosome 4 = (0.4 * 95 + 0.6 * 35) / 10 = 5.9.
FF for Chromosome 5 = (0.4 * 97 + 0.6 * 32) / 10 = 5.8.
Crossover-To perform crossover between two chromosomes, we'll select a crossover point randomly and exchange the genetic information beyond that point between the two chromosomes.
Let's consider the two fittest chromosomes, Chromosome 1 and Chromosome 2, and perform crossover:
Chromosome 1: 10, 25, 37, 4, 49, 12, 30, 5, 22, 46
Chromosome 2: 18, 43, 7, 36, 2, 15, 28, 50, 9, 4
Let's say we randomly select the crossover point after the 4th index position.After the crossover, we swap the

segments after the 4th position between the two chromosomes.
Before crossover:
Chromosome 1 : 10, 25, 37, 4, 49 | 12, 30, 5, 22, 46.

Chromosome 2 : 18, 43, 7, 36, 2 | 15, 28, 50, 9, 4. After crossover:
Chromosome 1 : 10, 25, 37, 4, 49, 15, 28, 50, 9, 4.
Chromosome 2 : 18, 43, 7, 36, 2, 12, 30, 5, 22, 46.
Before crossover, the chromosomes are considered as parents. After crossover, the chromosomes produced from the crossover process are referred to as children.
FF for child Chromosome 1 = (0.4 * 111 + 0.6 * 35) / 10 = 6.54.
FF for child Chromosome 2 = (0.4 * 91 + 0.6 * 39) / 10 = 5.98.
From the comparison of the two fittest chromosomes are: Child Chromosome1 and Parent Chromosome 1.
Mutataion-To perform mutation, let's randomly select a gene (employee) in each of the two fittest chromosomes (Parent Chromosome 1 and Child Chromosome 1) and change its value to another randomly selected value(1 to 50). Then, we'll recalculate the fitness for the mutated chromosomes and determine the most fittest chromo-some.
Let's say we randomly select position 7 (index 6) and change its value to a randomly selected value from the range of possible genes.
Child Chromosome 1 : 10, 25, 37, 4, 49, 15, 28, 50, 9, 4.
Chromosome 1 (After Mutation): 10, 25, 37, 4, 49, 44, 15, 50, 9, 4.
FF = (72.6 + 21.6) / 10 = 94.2 / 10 = 9.42.
Parent Chromosome 1: 10, 25, 37, 4, 49, 12, 30, 5, 22, 46
Chromosome 1 (After Mutation): 10, 25, 37, 4, 49, 12, 26, 5, 22, 46.
FF = (86.8 + 25.6) / 10 = 112.4 / 10 = 11.24.
After one iteration with the 5 chromosomes, it appears that the employees listed in Parent chromosome 1 are more suitable for the job. However, the number of iterations needed to find the best solution can vary dependingon the problem's complexity and the number of employees.

Functional Requirements

- **User Authentication:** All users, including team members, team leaders, and managers, should have individuallogin credentials(username and password).
- **Role-Based Access Control (RBAC):** Implement Role-Based Access Control to differentiate between userroles. Roles can include "User," "Team Leader," and "Manager."
- **Login Page:** Create a central login page accessible from the application's homepage.Users are prompted toenter their credentials, including username and password.
- **User Role Selection:** Upon successful login, users are redirected to a role-specific dashboard or landing page.Include a role selection option if users have multiple roles.
- **Dashboard Differentiation:** Design distinct dashboards for each role with functionalities tailored to their responsibilities. Users should only have access to features and information relevant to their role.
- **Remember Me Option:** Include a "Remember Me" option on the login page to provide convenience for userswho want to stay logged in across sessions.
- **Task Assignment Algorithm:** Implement a task allocation algorithm, potentially using genetic algorithms or heuristics. Optimize task assignments based on employee skills, workload, and project priorities.
- **Notifications and Alerts:** Send automated notifications for new task assignments, updates, and approachingdeadlines. Allow users to set preferences for notification frequency.
- **Historical Tracking:** Maintain a historical record of task allocations, completions, and any changes. Supportdata analysis for continuous process improvement.
- **Intuitive User Interface:** Provide a user-friendly interface for easy task creation, assignment, and tracking.Prioritize simplicity and clarity in design.
- **Mobile Compatibility:** Ensure compatibility with mobile devices to support remote access and flexibility.Optimize the user interface for various screen sizes.

Non Functional Requirements

- **Performance:** Response Time: The system should provide quick responses to user authentication requests androle-based access control actions, ensuring a seamless and responsive user experience.
- **Scalability:** The system should be scalable to accommodate a growing user base and increasing data volumeswithout a significant degradation in performance.
- **Reliability:** The authentication and RBAC system should be highly reliable, minimizing downtime and ensur-ing continuous availability to users.
- **Error Handling:** The system should gracefully handle errors and provide meaningful error messages to usersto assist in troubleshooting.

- **Security:** The system should adhere to industry-standard security protocols to protect user credentials andsensitive data during authentication and authorization processes.
- **Encryption:** Employ encryption mechanisms to secure data transmission between the client and the server,especially during the authentication process.
- **Usability:** The user interface for authentication and role assignment should be intuitive and user-friendly,requiring minimal training for users to navigate the system effectively.
- **Accessibility:** Ensure that the system adheres to accessibility standards to accommodate users with disabilities.
- **Scalability:** The system should be designed to handle a large number of concurrent authentication requests androle-based access control actions without a significant decrease in performance.
- **Maintainability:** The system should be designed with maintainability in mind, facilitating easy updates,patches, and modifications to adapt to evolving security requirements.
- **Compatibility:** The authentication and RBAC system should be compatible with various browsers and devicesto accommodate a diverse user base.
- **Auditability:** Implement logging and auditing mechanisms to track authentication attempts, role changes, andany security-related events for compliance and troubleshooting purposes.
- **Compliance:** Ensure that the authentication and RBAC system complies with relevant industry standards andregulatory requirements regarding user data protection and privacy.
- **Interoperability:** The system should be designed to seamlessly integrate with other systems and technologieswithin the Agile software development environment.
- **Capacity:** Specify the maximum number of users and roles the system should be able to handle efficiently.
- **Availability:** Define the required system availability, including planned downtime for maintenance and updates.

THE METHODLOGY OF THE EFFICENT TASK ALLOCATION:

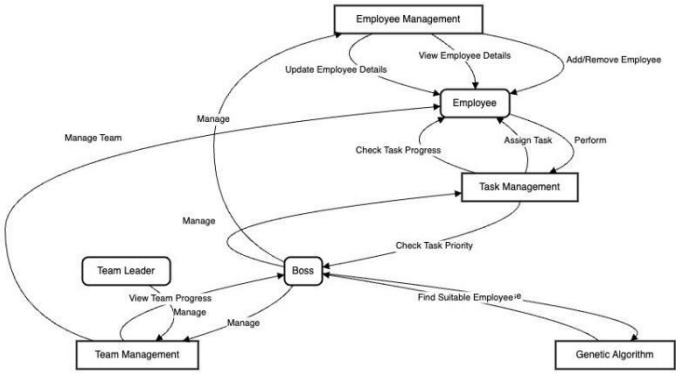

FIGURE 3. Block Diagram

Figure 3 displays the components and interactions within a system or process using a block diagram.

RESULT

Here is a brief summary of the developed application for task allocation, along with screenshots illustrating variouspages.
The first page shown in figure 7 is the login page of all the users including manager,team leader and employee.
The second page shown in Figure 8 is for adding a new employee.
The third page shown in figure 9 is for manage all the employee.
The fourth page shown in Figure 10 is used for assigning the most suitable employees to the project, determined bytheir rating and experience.
The fifth page shown in Figure 11 is dedicated to receiving updates on the project's progress.

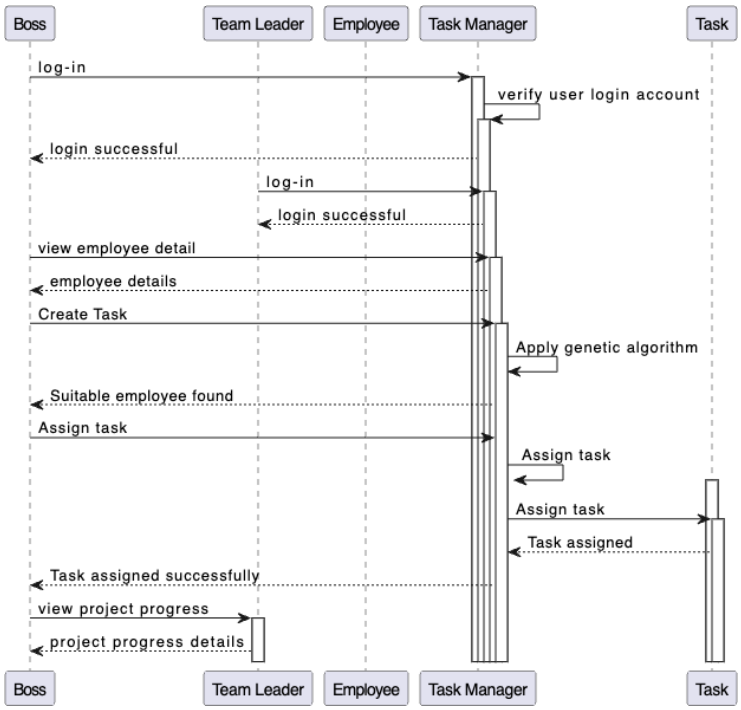

FIGURE 4. Sequence Diagram

A scenario showing the use of utility of the task allocation in application is given in sequence diagram in figure 4. Figure 5 provides an objective view of different classes used for the Task allocation application. utility provided by the application is graphically represented through a use case diagram given in figure 6 and the snapshots of the developed applications are presented in Figure7 (a-e).

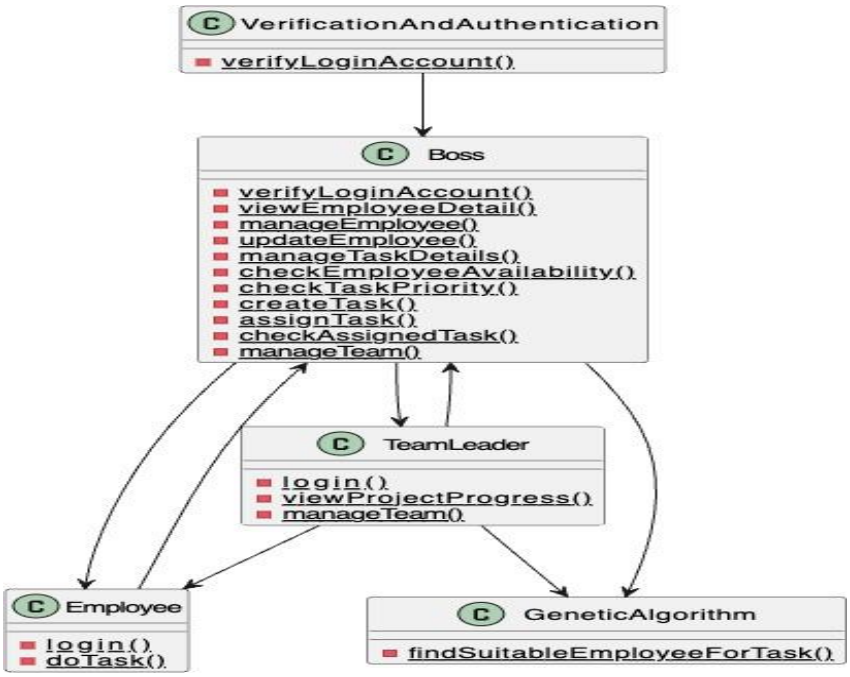

FIGURE 5. Class Diagram

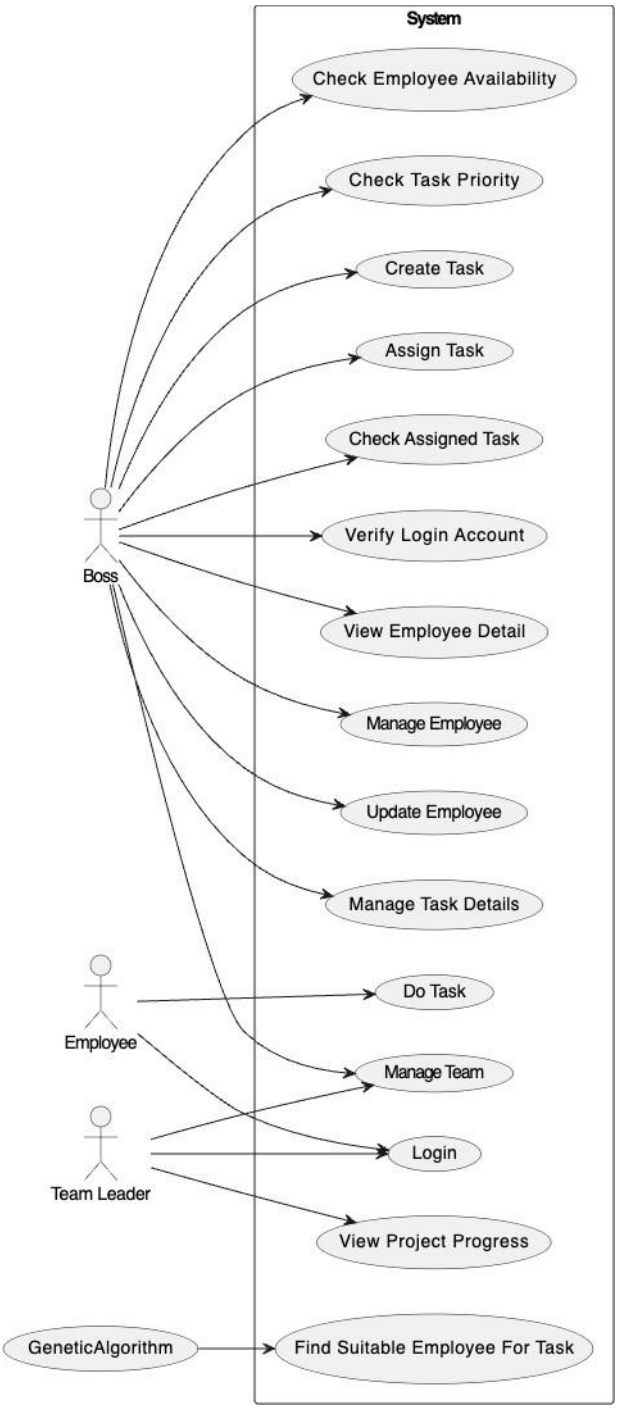

FIGURE 6. Usecase Diagram

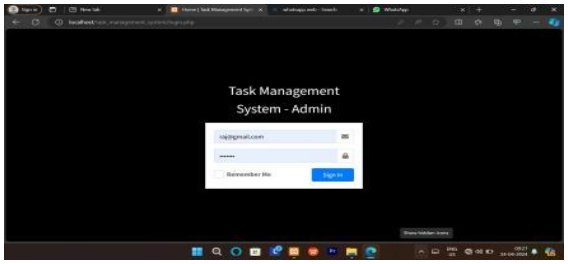

FIGURE 7(a).

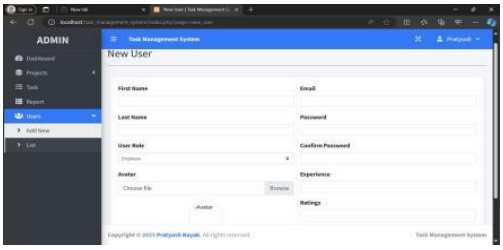

FIGURE 7(b).

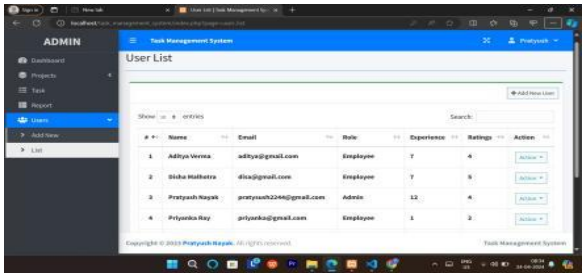

FIGURE 7(c).

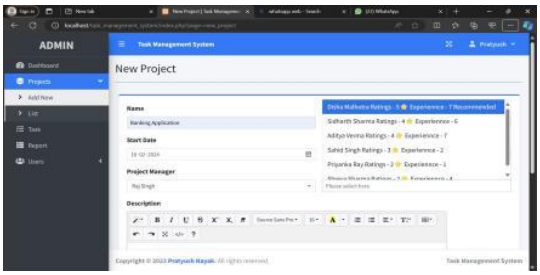

FIGURE 7(d).

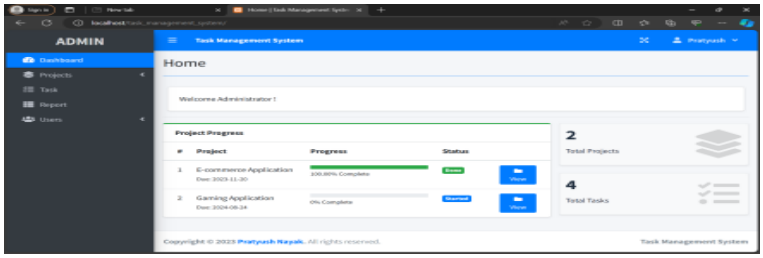

FIGURE 7 (e).

FIGURE 7. Snapshots of the developed application

CONCLUSION AND FUTURE WORK

In the realm of Agile software development, where task allocation is critical for project success, A application has been developed. Steam line process- by using genetic algorithm the task allocation has been done by two factors that is experience and rating of the employee. However, there are room for improvements by considering various other factors such as involvement interest of the employee, site, type of the project etc. Various other quantitative factors may be considered depending on the requirement and suitability of the project. Also genetic algorithm is the primitive algorithm more such evolutionary algorithm could be explored for enhance efficiency in task allocation.

ACKNOWLEDGMENT

We express our sincere gratitude and appreciation to the administrators of the School of Computer Engineering at KIIT Deemed to be University in Bhubaneswar, Odisha, India, for their generous provision of resources that supported ourstudy.

REFERENCES

1. K. N. Rao. et al. G. K. Naidu et al., and h Chakka et al., A study of the agile software development methods, applicability and implications inindustry," Int. J. Softw. Eng. Appl., vol. 5, no. 2, pp. 35-45, 2011.
2. Y. I. Alzoubi et al., A. Q. Gill et al., and A. AI-Ani et al., "Empirical studies of geographically distributed agile development communication challenges: A systematic review," Inf. Manage., vol. 53, no. 1, pp. 22-37, Jan. 2016.

3. S. V. Shrivastava et al. and U. Rathod et al., "Categorization of risk factors for distributed agile projects," Inf. Softw. Technol., vol. 58, pp.373-387,Feb. 2015.
4. A. Aslam et al., "Decision support system for risk assessment and management strategies in distributed software development," IEEE Access,vol. 5, pp. 20349-20373, 2017.
5. W.-N. Chen et al. and J. Zhang et al., "Ant colony optimization for software project scheduling and staffing with an event-based scheduler,"IEEE Trans. Softw.Eng., vol. 39, no. 1, pp. 1-17, Jan. 2013.
6. Hashmi AS et al., Hafeez Y et al., Jamal M et al., Ali S et al., Iqbal N et al. (2019) Role of situational agile distributed model to support modern software development teams. Mehran Univ Res J Eng Technol 38(3):655-666.
7. Sauer J et al. (2010) Architecture-centric development in globally distributed projects. In: Agility across time and space, pp 321-329.
8. Ijaz F et al., Aslam W et al. (2019) Identification of dependencies in task allocation during distributed agile software development. Sindh UnivRes J SURJ (Sci Ser) 51(01):31-36.
9. Aslam W et al., Ijaz F et al. (2018) A quantitative framework for task allocation in distributed agile software development. IEEE Access6:15380-15390.
10. Mak DK et al., Kruchten PB et al. (2006) Task coordination in an agile distributed software development environment. In: 2006 Canadian conference on electrical and, computer engineering, pp 606-611.
11. J. Lin et al., H. Yu et al., Z. Shen et al., and C. Miao et al., "Studying task allocation decisions of novice agile teams with data from agileproject management tools," in Proc. 29th ACM/IEEE Int. Conf. Autom.
12. J.Sutanto et al., A. Kankanhalli et al., and B. C. Y. Tan et al., "Investigating task coordination in globally dispersed teams: A structural contingency perspective," ACM Trans. Manage. Inf. Syst., vol. 6, no. 2, pp. 1-31, Jul. 2015.
13. R.Hoda et al. and L. K. Murugesan et al., "Multi-level agile project management challenges: A self-organizing team perspective," J. Syst.Softw., vol. 117, pp. 245-257, Jul. 2016.
14. W. Aslam et al., F. Ijaz et al., M. I. Lali et al., and W. Mehmood et al., "Risk aware and quality enriched effort estimation for mobile applications in distributed agile software development," J. Inf. Sci. Eng., vol. 33, no. 6, pp. 1481-1500, 2017.
15. Satapathy, Suresh, and Anima Naik. "Social group optimization (SGO): a new population evolutionary optimization technique." ComplexIntelligent Systems 2.3 (2016): 173-203.
16. Pant, Millie, et al. "Differential Evolution: A review of more than two decades of research." Engineering Applications of Artificial Intelligence90 (2020): 103479.
17. Kennedy, James, and Russell Eberhart. "Particle swarm optimization." Proceedings of ICNN'95-international conference on neural networks.Vol. 4. ieee, 1995.
18. Mohanty D., Jena, B. Khuntia,T., Mohanty, P.K., Mohapatra, S., Behera, S. (2024). Green Transit: Harnessing Renewable Energy For Sustainable Integration. Educational Administration: Theory and Practice, 30(4), 7242–7254. Retrieved from https://www.kuey.net/index.php/kuey/article/view/2552
19. Verma, S., Jena, J. J., Satapathy, S. C., Rout, M. (2020). Solving travelling salesman problem using discreet social group optimization. Journalof Scientific Industrial Research, 79(10), 928-930.
20. Jena, Junali Jasmine, and Suresh Chandra Satapathy. "A new adaptive tuned Social Group Optimization (SGO) algorithm with sigmoid- adaptive inertia weight for solving engineering design problems." Multimedia Tools and Applications 83.1 (2024): 3021-3055.
21. Das, S., Saha, P., Satapathy, S. C., Jena, J. J. (2021). Social group optimization algorithm for civil engineering structural health monitoring.Engineering Optimization, 53(10), 1651-1670.
22. Jena, Junali Jasmine, and Suresh Chandra Satapathy. "Mortality Prediction of Victims in Road Traffic Accidents (RTAs) in India using Opposite Population SGO-DE based Prediction Model." Journal of Scientific Industrial Research 80.11 (2021): 1001-1007.
23. Darshana, Subhashree, Siddharth Swarup Rautaray, and Manjusha Pandey. "AI to Machine Learning: Lifeless Automation and Issues." Ma- chine Learning: Theoretical Foundations and Practical Applications (2021): 123-135.
24. Dash, Adyasha, et al. "A clinical named entity recognition model using pretrained word embedding and deep neural networks." DecisionAnalytics Journal (2024): 100426.
25. J. Lin et al., H. Yu et al., Z. Shen et al., and C. Miao et al., "Studying task allocation decisions of novice agile teams with data from agileproject management tools," in Proc. 29th ACM/IEEE Int. Conf. Autom.
26. Jena, Junali Jasmine, et al. "Evolutionary algorithms-based machine learning models." Trends of Data Science and Applications: Theory and Practices (2021): 91-111.