Mitigating Communication Cost In Building Management System

R. Srimathi Pandipriya[1], K.Geetha.M.E[2]
[1]Dept of Computer Science And Applications
[2]Assistant Professor, Dept of Computer Science And Applications
[1, 2] Periyar Maniammai Institute of Science And Technology, Vallam, Thanjavur,Tamil Nadu,India.

Abstract- *In recent years reducing communication costs became an important factor in building management systems, optimizing the communication system sometimes leads to loss of communication yet boosting algorithms gained massive popularity in data science or machine learning competitions. Most of the winners of these competitions use boosting algorithms to achieve high accuracy. Boosting algorithms such as AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier.*

***ALGORITHM**:*

In any given building management system, managing multiple requirements from multiple entities and resolving the issues or concerns in an efficient way has become challenging day by day. Optimizing concurrent requirements and classifying the concerns according to the requirements need a lot of computational efficiencies and still, BMS struggles to mitigate the cost of communication in a huge way. Thus by adopting one of the best classification algorithms in artificial intelligence (AdaBoost Classifier) we are able to create an environment in the building management system where not only communication is improvised but also the cost of communication is mitigated in a complex integrated communication system.

I. INTRODUCTION

The most common thing that every people faced during the lifetime is the problem that happens in the place where they reside or work. We can see the common problem that every building will be having a problem at any point and this will lead to discomfort for the people who make use of the building. As it temporarily halt the daily work of the people it will lead to a general and uneasy circumstances. Thus we can solve this managing issue in the building by raising a ticket will help is segregating the certain technician to the certain problem will help in finshing the problem on time without delays.

1.2 EXISTING SYSTEM:

In the existing system, based on the algebraic graph theory, the Lyapunov stability theory, and the matrix theory, some sufficient conditions are established to deal with the consensus problem in the nonlinear multiagent systems.

DISADVANTAGES OF EXISTING SYSTEM:

- Not applicable for complex communication systems while handling concurrent requirements.
- Computational efficiency decreases when complex of the requirement increases.
- The cost of communication is reduced but not as compared to the proposed system
- Even though a cost is reduced still there are missing requirements in the complex system.
- Even though the system is integrated still requires human labor compared to the proposed model

1.3 PROPOSED SYSTEM:

In the proposed system we utilize the full potential of the AdaBoost classifier, which classifies the concurrent requirements based on the priorities. In our case, we implement this classifier in the building management system where there is a lot of tickets regarding issues and concerns are flown into the integrated system.

ADVANTAGES OF PROPOSED SYSTEM:

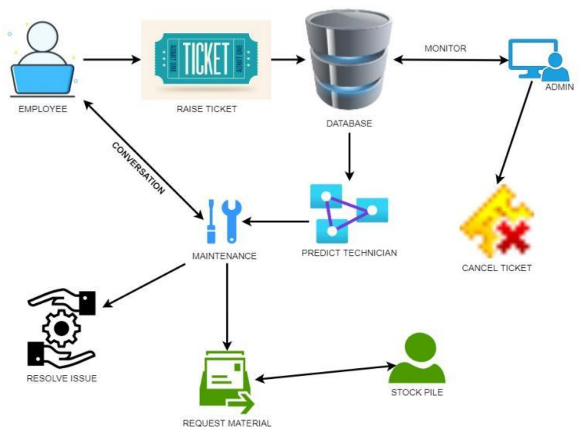
- Reduce the cost of communication on a huge scale compared to the existing system
- Classifying the multiple requirements is made easy and efficient.
- The time required to finish a job using an AdaBoost classifier is way higher compared to the existing system.
- Interaction between central managing systems to different entities is way smooth and efficient.

- The classification process is more precise that no requirements missed in the process. It dismisses human error while analyzing the document.

II. LOGICAL DEVELOPMENT

2.1 ARCHITECTURAL DESIGN:

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.



III. PROGRAM DESIGN

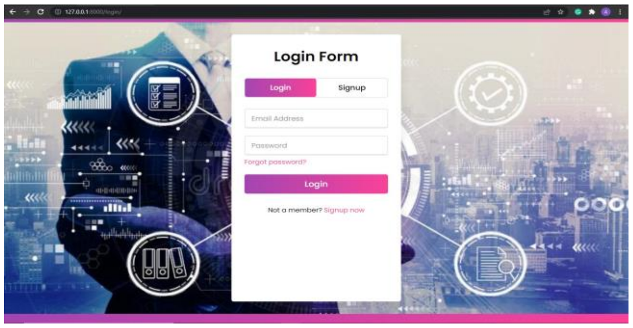
3.1 MODULES:

1) Employee
2) Maintenance
3) Stockpile
4) Admin

3.2 MODULES DESCRIPTION:

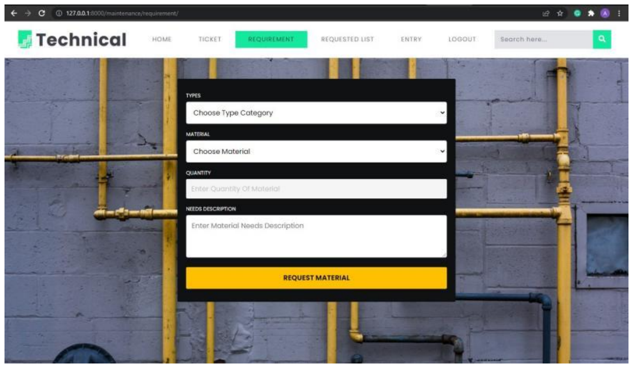
1.Employee:

Initially employee, register for the process of raising the ticket. That registration process will have the information of employee floor, block, name of the employee, email address and password. Once register process is finished then admin have to allow the employee for the login process. Once, this process is done then employee will login using email id and password. After login process home page for the employee page will be shown. In this, the form which shows the employee name, email id, floor, block, type of issue which is in the list and additional description field is used to give some additional details of issues. Once raise button is utilized then the above given details are registered in raised queue.



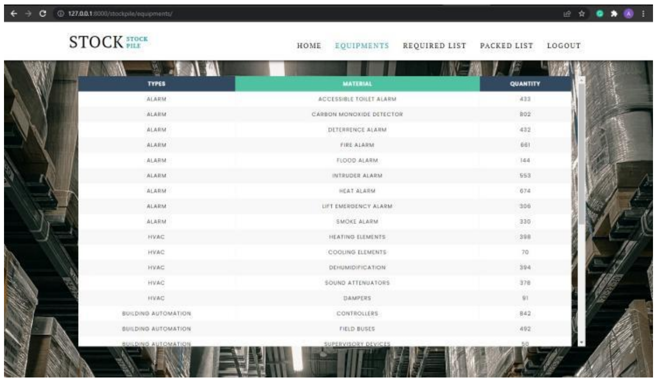
2.Maintenance:

Maintenance module is used for the maintenance team to view the raised ticket and also to contact the respective employee who raised the ticket. This module has four sub modules where the sub modules are ticket, requirement, requested list and entry. First of all the technician person should register with the type under, where the technician person will work, name of the technician, email id and password.After the acceptation process of register by admin only, the technician person can login to their respective page. The ticket sub module shows the ticket raised by the employee where the ticket belongs to the respective technician. In the ticket details table there is a chat button to make the conversation with the raised employee.



3.Stockpile:

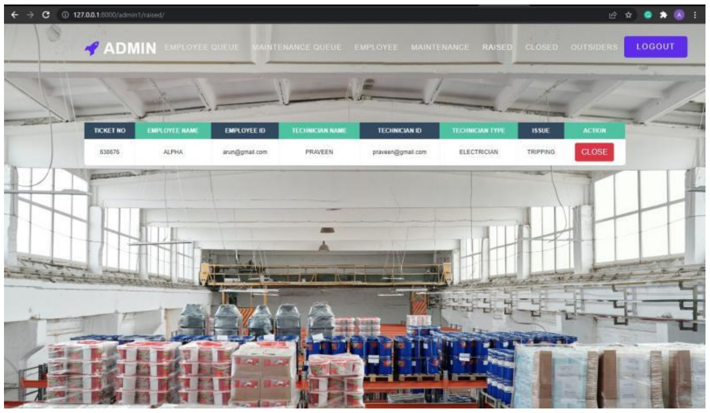
Stock pile is a module where the maintenance team can request for getting the materials or a parts of equipment to solve the issue raised by the employee. In stock pile there are three sub modules where equipments, required list and packed list. It has many stock pile employee for different categories. Finally equipments sub module shows the list of equipments with the types in the form of table. Then the stock pile employee can logout from the logout module which will delete the session details from the usage.



TYPES	MATERIAL	QUANTITY
ALARM	ACCESSIBLE TOILET ALARM	433
ALARM	CARBON MONOXIDE DETECTOR	802
ALARM	DETERRENCE ALARM	432
ALARM	FIRE ALARM	661
ALARM	FLOOD ALARM	144
ALARM	INTRUDER ALARM	553
ALARM	HEAT ALARM	674
ALARM	LIFT EMERGENCY ALARM	306
ALARM	SMOKE ALARM	330
HVAC	HEATING ELEMENTS	398
HVAC	COOLING ELEMENTS	70
HVAC	DEHUMIDIFICATION	394
HVAC	SOUND ATTENUATORS	378
HVAC	DAMPERS	91
BUILDING AUTOMATION	CONTROLLERS	842
BUILDING AUTOMATION	FIELD BUSES	492
BUILDING AUTOMATION	SUPERVISORY DEVICES	50

4.Admin:

Admin module which has all the details and the process throughout the application. Admin module has six modules. They are employee queue, maintenance queue, employee, maintenance, raised and closed. Once the admin logged in to their respective pages, they have the first sub module as employee queue has the table contains the details of the employee which the employee given for the registration. In that admin can allow for the employee login. In this also admin can allow for the technician login. Employee module is the third sub module where details of the employee who are in active status. From this admin can know about the employee details. Next is the maintenance sub module where it shows the table of information of the technician details in the maintenance team. At last there is out-comer sub module where out-comers list which is registered by the employee has been shown in the table. And also admin can close the registration for the out-comers or approve for them.



TICKET NO	EMPLOYEE NAME	EMPLOYEE ID	TECHNICIAN NAME	TECHNICIAN ID	TECHNICIAN TYPE	ISSUE	ACTION
538676	ALPHA	arun@gmail.com	PRAVEEN	praveen@gmail.com	ELECTRICIAN	TRIPPING	CLOSE

4.TESTING:

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

4.1STRATEGIC APPROACH TO SOFTWARE TESTING:

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

1. UNIT TESTING:

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing we have is white box oriented and some modules the steps are conducted in parallel.

WHITE BOX TESTING

This type of testing ensures that

- All independent paths have been exercised at least once
- All logical decisions have been exercised on their true and false sides
- All loops are executed at their boundaries and within their operational bounds
- All internal data structures have been exercised to assure their validity.

To follow the concept of white box testing we have tested each form .We have created independently to verify that Data flow is correct, All conditions are exercised to check their validity, All loops are executed on their boundaries.

2. BASIC PATH TESTING:

The established technique of flow graph with Cyclamate complexity was used to derive test cases for all the functions. The main steps in deriving test cases were:

Use the design of the code and draw correspondent flow graphs.

Determine the Cyclamate complexity of the resultant flow graph, using formula:

$V(G) = E - N + 2$ or
$V(G) = P + 1$ or
$V(G) =$ Number of Regions

Where V (G) is Cyclomatic complexity,
E is the number of edges,
N is the number of flow graph nodes,
P is the number of predicate nodes.
Determine the basis of set of linearly independent paths.

3. CONDITIONAL TESTING:

In this part of the testing each of the conditions were tested to both true and false aspects. And all the resulting paths were tested. So that each path that may be generated on particular condition is traced to uncover any possible errors.

4. DATA FLOW TESTING:

This type of testing selects the path of the program, according to the location of the definition and use of variables. This kind of testing was used only when some local variable were declared. The definition-use chain method was used in this type of testing. These were particularly useful in nested statements.

5. LOOP TESTING:

In this type of testing all the loops are tested to all the limits possible. The following exercise was adopted for all loops:

- All the loops were tested at their limits, just above them and just below them.
- All the loops were skipped at least once.

V. CONCLUSION

In our proposed model our highly efficient AdaBoost classification helps to classify multiple requirements and concerns from multiple entities at the same time. Eventually, our proposed system mitigating our cost of communication on a huge scale and minimizing the missing requirements, and maximizing accuracy. Interaction between multiple entities is made smooth and efficient in a highly complex communication system.

5.1 FUTURE ENHANCEMENTS:

In the future, we can apply real-time image processing where an entity can upload their requirements or issues in image format, based on the image processing output, requirements can be sent to the concerned maintenance team.

REFERENCES

[1] K. H. Movric and F.L. Lewis, “Cooperative optimal control for multi-agent systems on directed graph topologies,” IEEE Trans. Autom. Control,vol.59,no.3,pp.769–774,Mar.2014.
[2] J. Qin, C. Yu, and H. Gao, “Coordination for linear multiagent systems with dynamic interaction topology in the leader-following framework,” IEEE Trans. Ind. Electron., vol. 61, no. 5, pp. 2412–2422,May 2014.
[3] G. S. Seyboth, W. Ren, and F. Allgower, “Cooperative control of lin- ear multi-agent systems via distributed output regulation and transient synchronization,”Automatica,vol.68,pp.132–139,Jun.2016.
[4] L. Shi, J. L. Shao, M. T.Cao, and H. Xia, “Distributed contain- ment of heterogeneous multi-agent systems with switching topologies,” Neurocomputing,vol.312,pp.41–48,Oct.2018.
[5] H. Xia, T. Z. Huang, J. Shao, and J. Yu, “Group consensus of multi- agent systems with communication delays,” Neurocomputing, vol. 171, pp. 1666–1673, Jan.2016.
[6] H. B. Du, Y. Y. Cheng, Y. G. He, and R. T. Jia, “Second-order con- sensus for nonlinear leader-following multi-agent systems via dynamic output feedback control,” J. Robust Nonlinear Control, vol. 26, no. 2, pp. 329–344,2016.
[7] X. Ge, Q.-L. Han, and X.-M. Zhang, “Achieving cluster formation of multi-agent systems under aperiodic sampling and communication delays,” IEEE Trans. Ind. Electron., vol. 65, no. 4, pp. 3417–3426, Apr.2018.
[8] X. Liu, D. W. C. Ho, Q. Song, andW. Xu, “Finite/fixed-time pin-ningsynchronizationofcomplexnetworkswithstochasticdisturbances,” IEEETrans.Cybern.,vol.49,no.6,pp.2398–2403,Jun.2019.
[9] H. Shen et al., “Nonfragile dissipative synchronization for Markovian memristive neural networks: A gain-scheduled control scheme,” Nonlinear Anal. Hybrid Syst., vol. 34, no. 6, pp. 92–107,2019.
[10] C. Li, W. Yu, and T. Huang, “Impulsive synchronization schemes of stochastic complex networks with switching topology: Average time approach,”NeuralNetw.,vol.54,no.6,pp.85–94,2014.
[11] X. He, C. Li, and X. M. Pan, “Impulsive control and Hopf bifurcation of a three-dimensional chaotic system,” J. Vib. Control, vol. 20, no. 9, pp. 1361–1368,2014.
[12] T. Huang, C. Li, S. Duan, and J. A. Starzyk, “Robust exponential sta- bility of uncertain delayed neural networks with stochastic perturbationand impulse

effects," IEEE Trans. Neural Netw. Learn. Syst., vol. 23, no.6,pp.866–875,Jun.2012.

[13] Y. Zhu, S. Li, J. Ma, and Y.Zheng, "Bipartite consensus in networks of agents with antagonistic interactions and quantization," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 65, no. 12, pp. 2012–2016, Dec.2018.

[14] Y. Zheng, J. Ma, and L. Wang, "Consensus of hybrid multi-agent systems," IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 4, pp. 1359–1365, Apr.2018.

[15] H. Shen, Y. Wang, J. W. Xia, J. H. Park, and Z. Wang, "Fault- tolerant leader-following consensus for multi-agent systems subject to semi-Markov switching topologies: An event-triggered control scheme," IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 6, pp. 1841–1853, Nov.2019.

[16] T. Han, Z.-H. Guan, M. Chi, B. Hu, T. Li, and X.-H. Zhang, "Multi- formation control of nonlinear leader-following multi-agent systems," ISA Trans., vol. 69, pp. 140–147, Jul.2017.

[17] X. W. Dong, Q. K. Tan, Q. D. Li, and Z. Ren, "Necessary and sufficient conditions for average formation tracking of second-order multi-agent systems with multiple leaders," J. Frankl. Inst., vol. 354, no. 2, pp. 611–626,2017.