A Design of an Agent Based System for Timetabling

[1]Kehinde Oluwatoyin Babaagba, [2]Samson Afolabi Arekete

[1]Assistant Lecturer, [2]Senior Lecturer
[1]Department of Computer Science
[1]Redeemer's University, Ede, Nigeria

***Abstract*—Timetabling forms an important part of most educational systems. This is because most educational institutes largely rely on time tables for their day to day activities. A number of these institutions rather use the very strenuous and slow manual time tabling systems. This results in problems such as inaccurate and poorly managed timetables. In this paper, we propose a design solution towards solving the timetabling problem which uses agent technology. We also examine how the use of an agent based system can help to solve some of the constraints in time table management. The constraints are categorized as both soft constraint (those constraints which can be violated without having a significant impact on the time table's efficiency) and hard constraints (those constraints that must not be violated else the time table generated will be inefficient). The system was designed using modeling tools from the Unified Modeling Language (UML) set. Program development was done in Java using the Netbeans 7.2 integrated development environment. The Java Agent Development Framework – JADE was adopted as the mobile agent platform. A rudimentary time table was generated as a test case.**

***Keywords*—Timetabling, Agent technology, Hard and Soft constraints**

I. INTRODUCTION

Timetabling is essential for teaching and learning. Time table management entails scheduling scarce resources in the most optimal way possible that guarantees effective teaching and learning while at the same time ensuring the comforts of the interest groups as much as possible. Most educational institutions lack reliable and efficient intelligent time table systems and they mostly rely on the manual system, which is a very difficult and time consuming task. Timetabling becomes a hot issue at the start of each new semester for the head of department and the time table committee. Such manual systems are lacking in historical data to make the job easy and comfortable. Other problems of a traditional manual system include lack of accuracy, slow speed, and poor information sharing. The goal of course timetabling is to find an appropriate timetable for a set of courses to be scheduled within limited resources such as classrooms and class time. This timetable must be found while simultaneously satisfying certain constraints such as efficient use of resources, convenience to students and instructors and so on.

According to [1], the timetable concept emanated from an image projected from above upon a large circular table, in which twelve dials were positioned around the perimeter of the table. The functions of each dial changes and mutates, depending on what is projected onto them at any given instance. The dials could sometimes become clocks, gauges, speedometers, switches, steering wheels, and so on. A real-time 3D scene, projected onto the central part of the table, was controlled and influenced by the movements of the dials. At the outset, the space of timetable seems to be rational and unified, but the longer the piece was used, the more complex and multi-dimensional it became, as perspectives and timeframes diverge and split off from each other. Timetable attempts to translate certain paradoxical or even impossible pseudo-scientific concepts into concrete experiences, such as time travel, multiple branching universes, alternate dimensions and shared hallucinations.

There are generally two types of constraints in timetabling: hard and soft constraints [2]. Hard constraints are those that must be satisfied and cannot be violated. Any violations of the hard constraints would render the timetable unusable. For example, a professor cannot be scheduled to give two distinct lectures at the same time in two different classrooms. Another example of hard constraint is to schedule two different lectures to hold in the same lecture room simultaneously, this would lead to a deadlock. Soft constraints on the other hand, are those that are preferred to be satisfied but may be relaxed under certain conditions, particularly in order to satisfy hard constraints. Violations of soft constraints may be unavoidable sometimes, as they can be mutually conflicting. Failure to meet a number of soft constraints may, however, lead to a poor quality timetable. Diverse approaches have been suggested for solving the course timetabling problem. [3] proposed decomposing the problem into a series of easier sub-problems. Genetic algorithms and Tabu Searches were used in handling the problem by some authors. Simulated annealing was also used by [4] while co-evolutionary algorithms were proposed by [5]. A number of authors have also applied the agent technology to timetabling ([6]-[8]).

An agent is defined as a specific software entity which acts autonomously, yet on behalf of a user or another agent ([9], [10]). Agents are active, persistent software components which are capable of perceiving, reasoning, acting and communicating [11]. According to [12], an agent constitutes an atomic software entity operating through autonomous actions on behalf of users without intervention. An agent may be harnessed to perform some information gathering or processing task in the background; usually the agent’s task is very small and well defined. The theory of agents though conceived for a long while, however, became more prominent with the growth of the Internet from the 1990s. A number of companies are now believed to sell software that enables you to configure an agent to search the Internet for certain types of information. The human mind is thought of essentially to consist of thousands or millions of agents all working in parallel. To produce real artificial intelligence therefore, this school of thought holds that we should build computer systems that also contain many agents and systems for arbitrating among the agents' competing results [13].

An agent system consists of the agent and its environment [14]. The agent receives stimuli from the environment and carries out actions on the environment. An agent has a body and a controller. The controller receives percepts from the body and sends commands to the body. A body includes sensors that convert stimuli into percepts and actuators that convert commands into actions. Stimuli can be in the form of light, sound, words typed on a keyboard, mouse movements, or physical bumps. The stimuli can also include information obtained from a web page or from a database. Common sensors include touch sensors, cameras, infrared sensors, sonar, microphones, keyboards, mice, and XML readers used to extract information from web pages. Agents have been variously classified as: Collaborative Agents, Interface Agents, Internet/Information Agents, Mobile Agents, Reactive Agents, Hybrid Agents, and Heterogeneous Agents [15]. A mobile agent is a computer program that acts autonomously on behalf of an application, a person or an organization. Its unique ability is mobility. It can move from one place to another in a network. This ability allows the intelligence or application to be moved from a network management centre to managed devices. Furthermore, compared with traditional distributed computing paradigms, software developed using mobile agent is done in a one-side-programming, application-oriented way. This leads to extreme flexibility and efficiency [16]. [9] remarked that mobile agent combines agency with mobility.

Agent technology may be classified as single-agent and multi-agent systems ([9], [10]). In single-agent systems, an agent performs tasks on behalf of a user or some process; while performing a task, it may communicate with a user and a local or remote system resources. The mandatory attributes of single-agent systems are autonomy, decision making, temporal continuity and goal-oriented. In addition, they may also possess intelligence attribute. On the other hand, the agents in multi-agent system may extensively cooperate with each other to achieve their individual goals, and also may interact with users and system resources. Multi-agent systems may use either a set of static agents, a set of mobile agents or a combination of mobile and static agents to achieve their goals. The mandatory attributes of multi-agent systems are autonomy, decision making, temporal continuity, goal-oriented, communicative and collaboration. In addition, they may also possess learning (for intelligent distributed agents) and mobility (for mobile agents) attributes. An agent based system can handle the time timetabling problem effectively, hence its application in this study.

The rest of the paper is organized into three sections. The second section includes the materials and methods used when carrying out the research work. Furthermore, the third section includes the results and discussions. Finally, section four concludes our findings as well as presents possible future work.

II. MATERIALS AND METHODS

This section presents the architecture for the agent based system, the tools and technologies adopted, the system algorithm, the system flowchart as well as the Unified Modeling Language diagrams used in the system development.

A. System Architecture

The agent based system includes the timetable agent that launches the login form and schedules the courses based on the information entered by the users, which include: Teacher, Course Manager and Room Manager. The teacher or lecturer is responsible for allocating time and venue, the course manager is in charge of managing and adding courses, the room manager on the other hand is in charge of managing venues (adding and assigning venues). The system architecture is shown below.

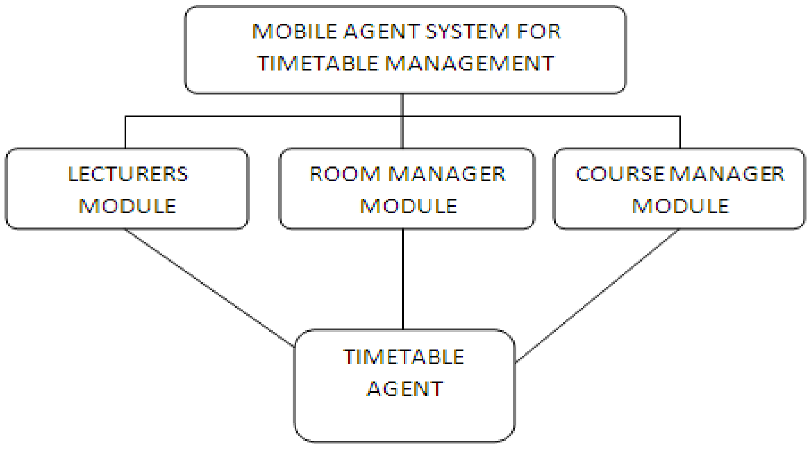


Figure 1: System Architecture

B. Tools and Technologies

The system was developed using the following development tools and technologies: Java is adopted as the programming language, Microsoft Access is used as the Database Management System, Unified Modeling Language (UML) was employed to model the system interactions and behaviours, and NetBeans 7.2 was used as the IDE (Integrated Development Environment). JADE was adopted as the mobile agent platform.

C. *Time table development Algorithm*

a) The Timetable agent is launched via the JADE platform
b) This launches the login form. There will be three users of the system; the Course Manager (who will add the courses), the Room Manager (who will add the lecture venue) and the Teacher (who will allocate the time and venue for the course). Each will have to select their login role.
c) The teacher allocates the time and venue for the courses
d) The course manager adds the courses and the size of students taking the courses using the add course form.
e) The room manager adds the venues and their capacities using the add room form.
f) The Timetable agent schedules the courses using the schedule course form.
g) The Timetable is generated.

D. *System Flowchart*

Given below is the flow chart of the proposed system;

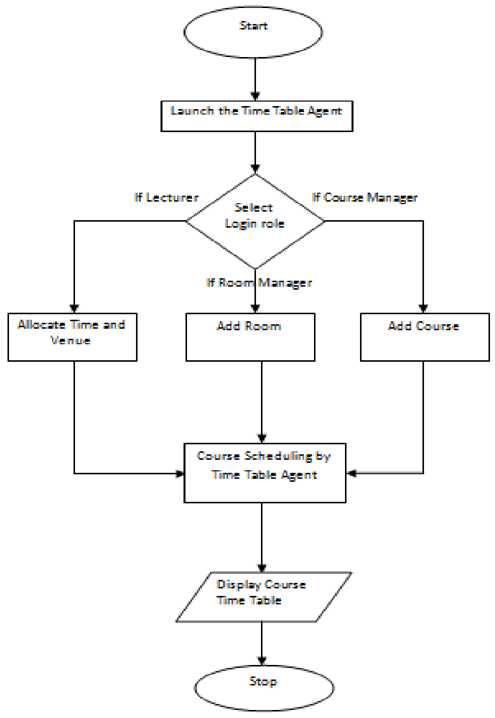
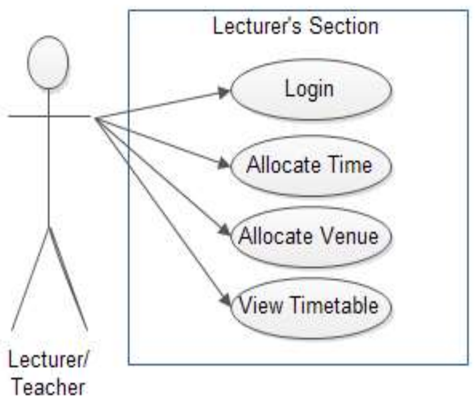
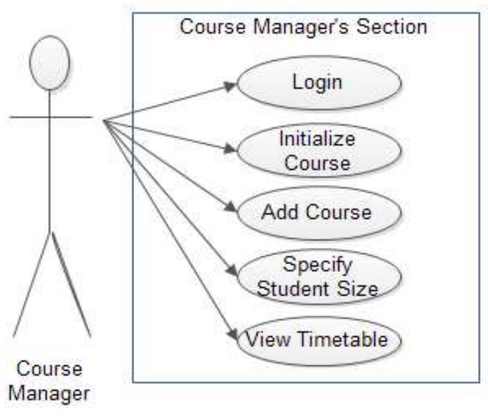


Figure 2: System Flow Chart

E. *Object Oriented Design Using Unified Modeling Language*

The actors of the system are the lecturers/teachers, course manager, room manager and students. The Use Case diagrams are given below;





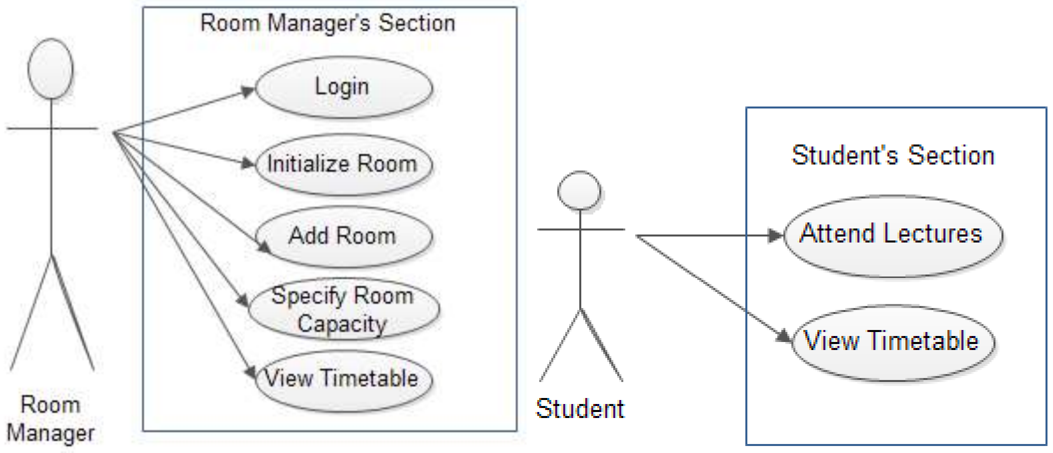


Figure 3: Use Case Diagrams

The class diagram is given below;

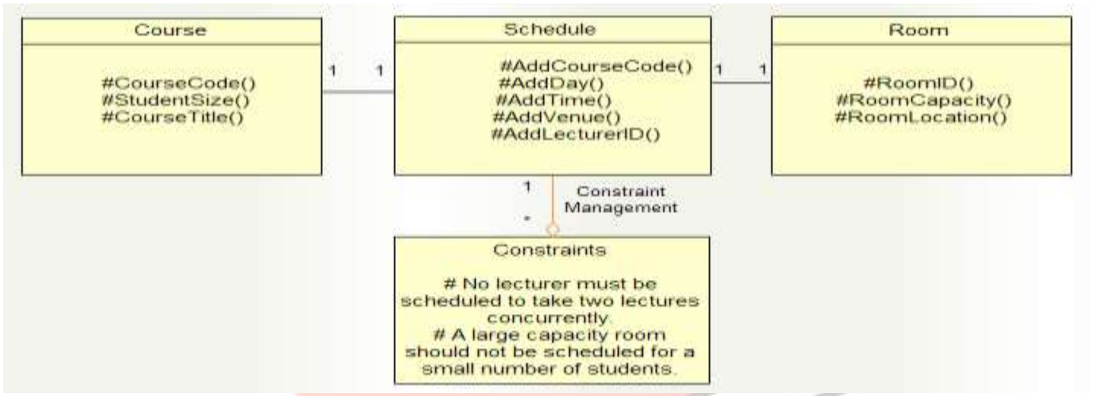


Figure 4: Class Diagram

The activity diagram is also given below;

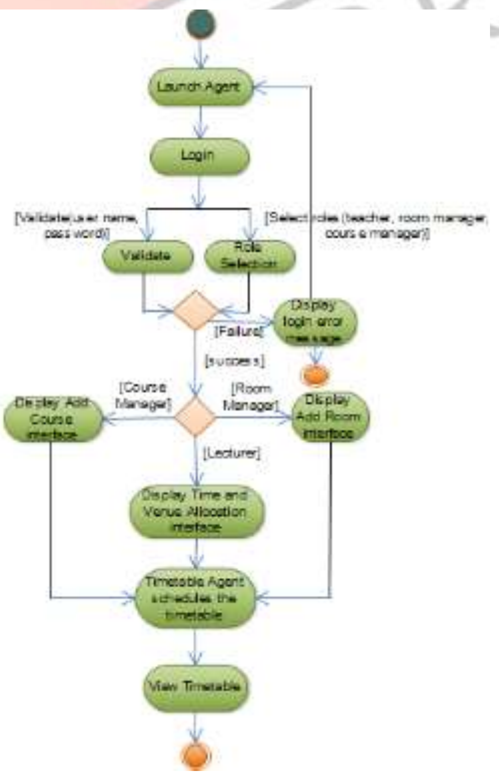
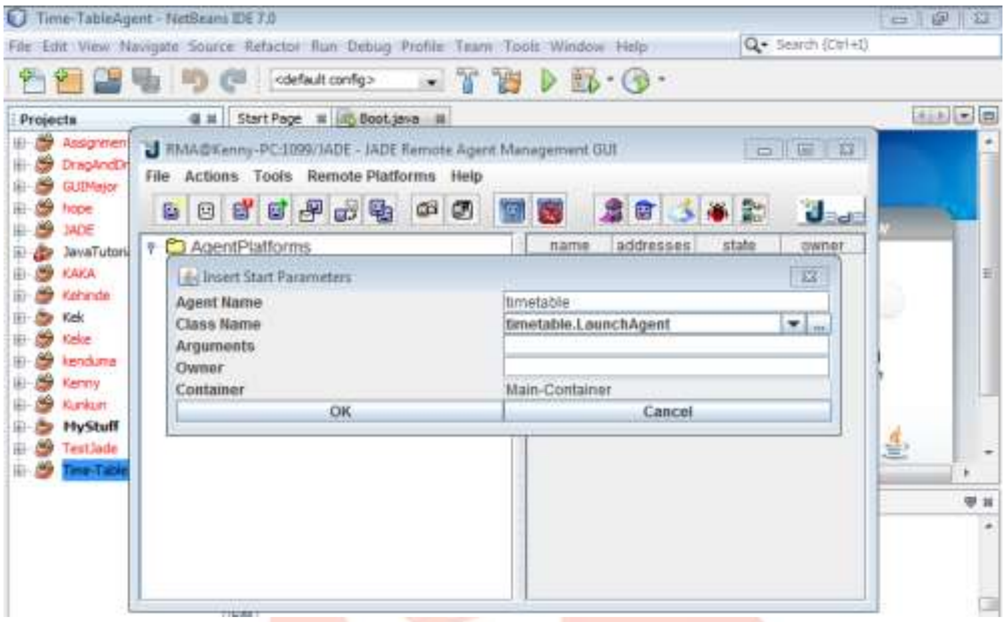

Launch Agent
Login
[Validate(user name, pass word)]
[Select roles (teacher, room manager, course manager)]
Validate
Role Selection
Display login error message
[Failure]
[success]
[Course Manager]
[Room Manager]
Display Add Course interface
Display Add Room interface
[Lecturer]
Display Time and Venue Allocation interface
Timetable Agent schedules the timetable
View Timetable

Figure 5: Activity Diagram

III. RESULT AND DISCUSSION

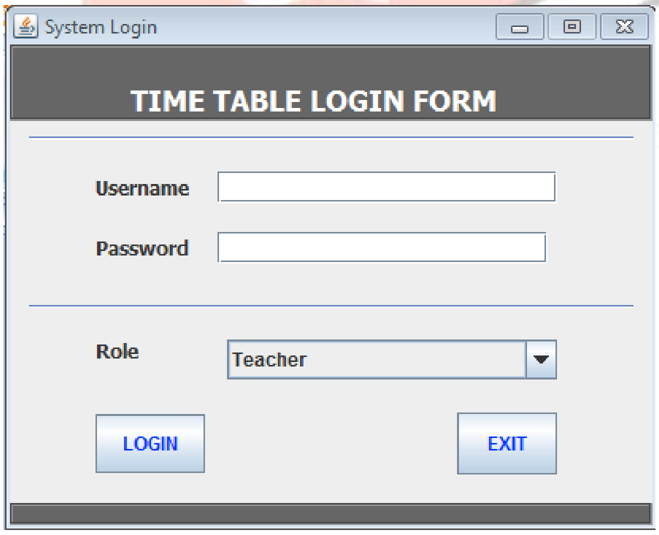
The proposed system involves using agent technology to solve the problem of course timetabling. The timetable agent launches the login form from which the users of the system: Course Manager, Room Manager and the Teacher or Lecturer select their own login role. The strength of this approach is to use a fundamental attribute of agents which is, autonomy. This autonomy is manifested in this work in the timetable agent. The add course form is used for adding courses and the add room form is used for adding venues. The information entered will be stored in a Microsoft Access database for scheduling the courses. The scheduling problem is solved using the schedule course form to ensure that constraints are satisfied. A feasible timetable is generated by the timetable agent.
This is the first page that enables you to start the new agent by specifying the agent name and the agent class.


Time-TableAgent - NetBeans IDE 7.0
RMA@Kenny-PC:1099/JADE - JADE Remote Agent Management GUI
AgentPlatforms
Insert Start Parameters
OK
Cancel

Agent Name: timetable
Class Name: timetable.LaunchAgent
Arguments:
Owner:
Container: Main-Container

Figure 6: Launch Agent Page

Once the timetable agent is launched, it launches the login form which is a form that enables the various users (lecturers, course manager and room manager) to enter their login details which include username and password as well as to allow them to select their individual roles.



Username:
Password:
Role: Teacher

Figure 7: System Login Form

Depending on the role selected that is, Teacher, Course Manager and Room Manager, different forms are launched defining what the various users can do. The forms are illustrated below;

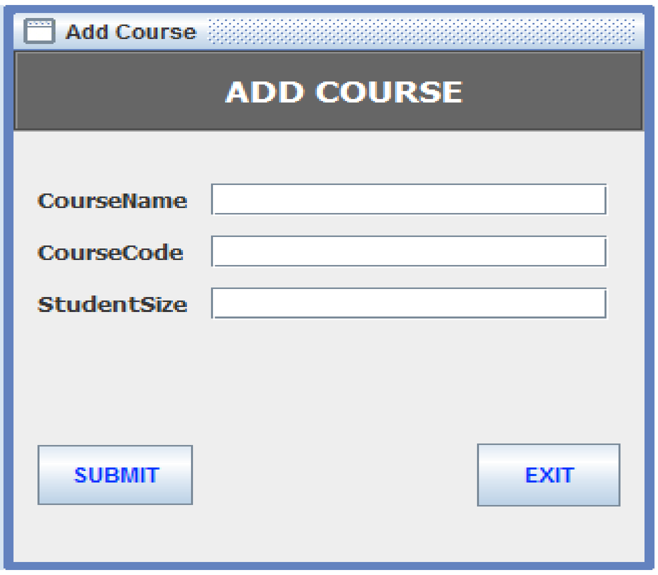


Figure 8: Add Course Form

The Add Course Form enables the course manager to add courses and their details.

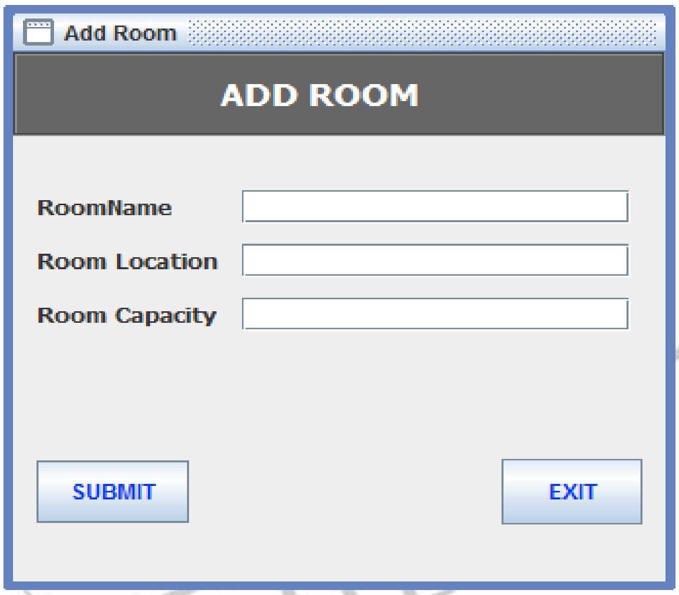
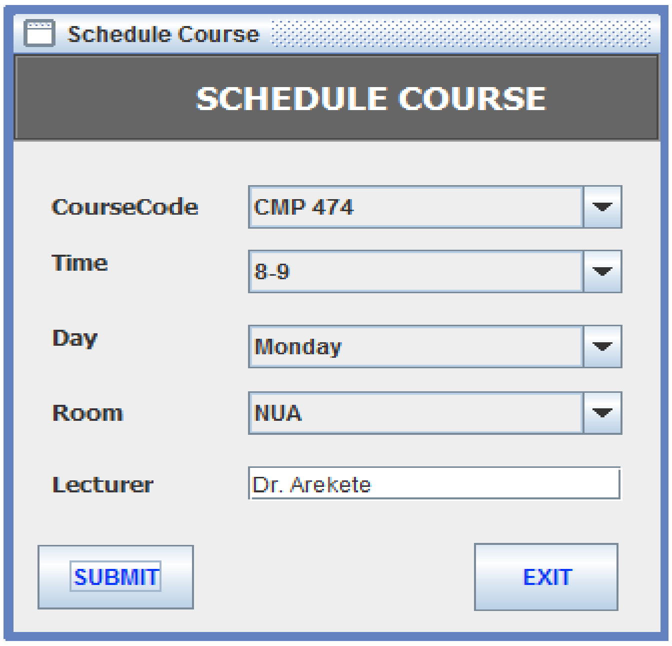


Figure 9: Add Room Form

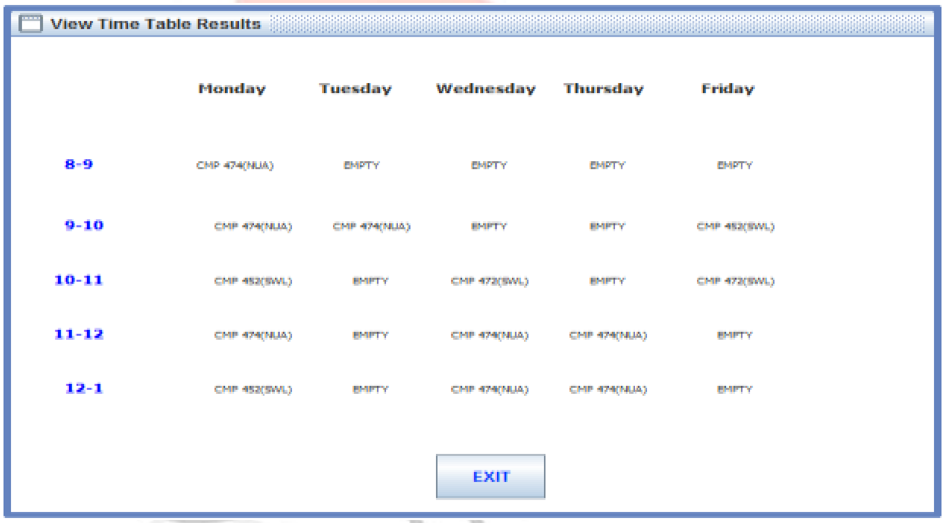
This Add Room Form enables the room manager to add rooms and their details.



CourseCode	CMP 474
Time	8-9
Day	Monday
Room	NUA
Lecturer	Dr. Arekete

Figure 10: Schedule Course Form

This Schedule Course Form enables the lecturers to schedule their courses by entering their preferred time and venue as well as other details.



	Monday	Tuesday	Wednesday	Thursday	Friday
8-9	CMP 474(NUA)	EMPTY	EMPTY	EMPTY	EMPTY
9-10	CMP 474(NUA)	CMP 474(NUA)	EMPTY	EMPTY	CMP 452(SWL)
10-11	CMP 452(SWL)	EMPTY	CMP 472(SWL)	EMPTY	CMP 472(SWL)
11-12	CMP 474(NUA)	EMPTY	CMP 474(NUA)	CMP 474(NUA)	EMPTY
12-1	CMP 452(SWL)	EMPTY	CMP 474(NUA)	CMP 474(NUA)	EMPTY

Figure 11: View Time Table Results Form

This View Time Table Results Form enables the users to view the time table.

IV. CONCLUSION AND FUTURE WORK

An agent based system for timetabling has been developed in this work. An Object Oriented Design using Unified Modeling Language diagrams such as Use Case diagrams, Class diagram, Sequence diagram and Activity diagram was adopted. The programming was done using Java while JADE was adopted as the mobile agent platform. A simple time table was generated. For further research, we recommend the following; the agent system should be examined to determine its behavior under increased and more complex soft requirements, agent technology can be adopted for other systems such as network monitoring among others.

REFERENCES

[1] P. Hoberman, "Time Table", Available at: www.perryhoberman.com/page32/index.html [Accessed at October 2012].

[2] Y. Yang, R. Paranjape, L Benedicenti and N. Reed "A Mobile Agent System for University Course Timetabling," Indian International Conference on Artificial Intelligence (IICA-05), India, pp. 20-22, 2005.

[3] A. Hertz and V. Robert, "Constructing a course schedule by solving a series of assignment type problems," European Journal of Operational Research, 108: 585-603, 1998.

[4] M.A.S. El-mohamed, G. Fox and P. Coddington, "A comparison of annealing techniques for academic course scheduling," The 2nd International Conference for the Practice and Theory of Automated Timetabling, PATAT'97: 92-112, 1997.

[5] C.K. Chan, H.B. Gooi and M.H. Lim, "A Co-evolutionary Algorithm approach to a university timetable system," In Proceedings of the 2002 Congress on Evolutionary Computation (CEC '02), 2: 1946-1951, 2002.

[6] K.O. Babaagba and S.A. Arekete, "A Review of Agent-Based University Course Timetabling Systems," International Journal of Engineering Development and Research, vol. 5(2), 2017.

[7] P. De Causmaecker, P. Demeester, Y. Lu and G. Vanden Berghe, "Agent technology for timetabling." In Proceedings of the 4th international conference on practice and theory of automated timetabling (pp. 215–220), 2002.

[8] Y. Yang, R Paranjape and L. Benedicenti, "An agent based general solution model for the course timetabling problem," Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, May 08-12, 2006, Hakodate, Japan [doi>10.1145/1160633.1160901].

[9] S. A. Arekete, O.C. Akinyokun, O. Olabode and B.K. Alese, "Design of a Mobile Agent for Monitoring Users Activities". Computer Engineering and Intelligent Systems. 2013; 4(2): 33-48. Available from: www.iiste.org.

[10] S.S. Manvi and P. Venkataram, "An Agent Based Synchronization Scheme for Multimedia Applications," International Journal of Systems and Software, Vol. 79, No. 5, pp. 701-713, 2006.

[11] R.Y. Fung and T. Chen,"A multiagent supply chain planning and coordination architecture." The International Journal of Advanced Manufacturing Technology, 25(7), 811-819, 2005.

[12] Z.Z. Shi and N.N. Zheng, "Progress and challenge of artificial intelligence," Journal of computer science and technology, 21(5), pp.810-822, 2006.

[13] K.O. Babaagba and S.A. Arekete, "Multi-Agent System for Time table scheduling and Management," A work in fulfillment of a Bachelors Degree in Computer Science, Redeemer's University, Ede, Osun State, 2013.

[14] D. Poole,"Artificial Intelligence-Foundations of Computational Agents," Available at: http://artint.info/html/ArtInt_35.html [Accessed at October 2012].

[15] J.M. Tim, "Artificial Intelligence-A Systems Approach, Hingham," Massachussets New Delhi,Infinity Science Press LLC, 2008.

[16] D. Zhang, W. Zorn, "Developing network management applications in an application-oriented way using mobile agent," Computer Networks, 0169-7552/98/, pp. 1551–1557, 1998.