Agent-Based Autonomous Examination Systems

R. D. Gawali
Dept. of Computer Engineering
Lokmanya Tilak College of Engineering
Navi Mumbai (M.S.), India - 400614
gawalird@yahoo.com

B.B. Meshram
Dept. of Computer Technology
Veermata Jijabai Technological Institute (VJTI)
Mumbai (M.S.), India - 400019
bbmeshram@vjti.org.in

***Abstract* - Online Computerized Examination Systems is developed using software agent technology. Multi-Agent system containing main agent, mobile agent and stationery agent is developed to support the functionality of examination systems in Aglet environment. All these agents in the system communicate with each other to process the operations requested by the user. Authentication of the examinee for valid user name and password is done by the stationary agents at user authentication module and database with the help of mobile agent. After successful authentication and selection of subject by examinee, mobile agent collects questions, their alternatives and correct answer retrieved by stationary agent from the database. These questions are displayed to the examinee. As examinee goes on answering these questions, main agent stores the answer given by examinee in database and updates the score. When the examination gets over, the main agent processes the result and displays result.**

Keywords – agent; mobile agent; aglet

I. INTRODUCTION

Examination and evaluation are part of the competitive world today. Examinations are conducted to assess the knowledge, progress and the ability of the candidate to apply acquired knowledge to solve problems. Examinations can differ not only by their contents, but also by their purpose and the way they are conducted and evaluated. There are different methods used for conducting and evaluating examinations. The conventional paper-and-pencil test have a very tedious procedure, as one need to set question paper and conduct the examination at a particular center, which may or may not be convenient to everyone. Then there is need to collect the answer papers and assess all of them. This entire procedure is always time consuming. Apart from this, delays in declaration of the result and apprehension of unfair means in the examination make people to think that the judgment may not be always fair. Therefore the need of online examination arose.

Software agent technology provides better alternative for conducting online examinations. Online examinations can introduce more variations into the structure of the examination where examination can be offered at different time, different locations or even different tests to different candidates. Assessment and display of result may be possible in less time.

This paper describes multi-agent software system, which has been used for development of online examination systems. In this system, before start of examination, examinee needs to authenticate. After successful authentication, examinee selects the subject at whom he or she wishes to appear. This examination is scheduled for fixed time with fixed number of questions. These questions are retrieved randomly by examination system from database and displayed to examinee. Examinee can answer these questions by selecting one of the alternatives. The examination gets over either when examinee has answered all questions; examinee desires to terminate examination or end of allotted time period. Result of examination gets displayed immediately. Different software agents handle different operations such as retrieving questions, keeping track of time elapsed and assessment of answers in the system.

The paper is organized as below: section II describes the extensive literature survey made in order to complete this work. Section III deals with the proposed systems software architecture and its pseudo code. Section IV explains the result from the systems developed. Lastly we conclude with result in section V.

II. LITERATURE SURVEY

An agent is a computational entity, which acts on behalf of other entities in an autonomous fashion. Agents are independent pieces of software capable of acting autonomously in response to input from their environment. Agents can be of different abilities, but typically posses the required functionality to fulfill their design objectives. The concept of an agent originates from the area of Artificial Intelligence but has now gained more widespread acceptance in mainstream computer science.

Software agent is basically a software program designed to perform specific goal-driven functions and has the capability to communicate with other agents. A software agent is an expert in its function. It is aware of and can access the functioning capabilities of other agents. Each agent is able to ask for information and/or a decision from other and is also able to respond to events as they occur, modifying its behavior as required.

A software agent can halt itself, ship itself to another computer on the network, and continue execution at the new computer. An agent doesn't restart execution from the beginning at the new computer; it continues where it left off. For example, imagine an agent that increments a counter starting with zero. If that agent counts from zero to ten, then halts and ships itself to another computer, it will not start

978-1-4244-4711-4/09/$25.00 ©2009 IEEE

counting again at zero. It will continue counting starting with ten, because that was where it left off when it halted at its previous computer.

The number and type of application domains in which agent technologies are being applied to include workflow management, network management, air-traffic control, business process re-engineering, data mining, information retrieval/management, electronic commerce, education, personal digital assistants, scheduling/diary management [1].

A. Agents

Agents are autonomous and social entities used to develop complex applications. The idea behind the concept of agent is the one of the active and autonomous module that can work cooperating and/or competing with other modules/agents and the surrounding environment. Agents have characteristics such as reactive, autonomous, goal-oriented, continuously running, communicative learning, mobile, flexible and social ability. Agents are autonomous because they decide where they will go and what they will do. They control their lifetimes. They can receive requests from external sources, such as other agents, but each individual agent decides whether or not to comply with external requests. Also, agents can decide to perform actions, such as travel across a network to a new computer, independent of any external request [1].

There are five basic types of agents according to their functionality [2],

(1) Simple reflex agent, which selects action on current percept,

(2) Model-based reflex agent, which keeps information about how the world works,

(3) Goal-based agent, which keeps track of world as well as set of goals,

(4) Utility-based agent, which uses model of the world along with a utility function. Then chooses the action that leads to the best-expected utility and

(5) Learning agent, which is responsible for making improvements by taking feedback.

Agents can be roughly subdivided into two types, stationary and mobile agents. Agents mostly tend to be mobile, but do not have to. Stationary agents execute only in the system space they are created in. If these agents need information that resides outside their own domain, other agents or communication mechanisms have to be called by them. Mobile agents are not limited to the system space they are created in. Mobile agents are free to travel among all possible hosts in a network [3].

B. Mobile Agents

Mobility of agent means it moves independently from one computer to another computer in the network and completes the task assigned by the user. Mobile agent is not bound to the system where it begins execution. It can move from one system to another within the network. It transports both its state and its code with it.

Mobile agent can have strong mobility where agent code, data and execution state gets migrated whereas in weak mobility only the agent code and data gets migrated. Since agents are often smaller than data they are going to work on, mobility can lead to a bandwidth saving. Mobility may help in solving host problems where agents migrate from a corrupted or halting host to another one in order to continue their computations [5].

C. Generic Mobile Agent Architecture

Agents cannot live themselves, but must rely on a specific environment. A mobile agent environment is a software system that is distributed over a network of heterogeneous computers. Its primary task is to provide an environment in which mobile agents can execute. It may also provide support services, and finally support for openness when accessing non-agent-based software environments. Agent which relate to the mobile agent environment itself, support services pertaining to the environments on which the mobile agent environment is built, services to support access to other mobile agent systems environment controls the agent life cycle, deciding when an agent must be started and stopped, must be destroyed; messages must be delivered to and so on.

Basic mobile agent architecture can be viewed as shown in fig. 1.

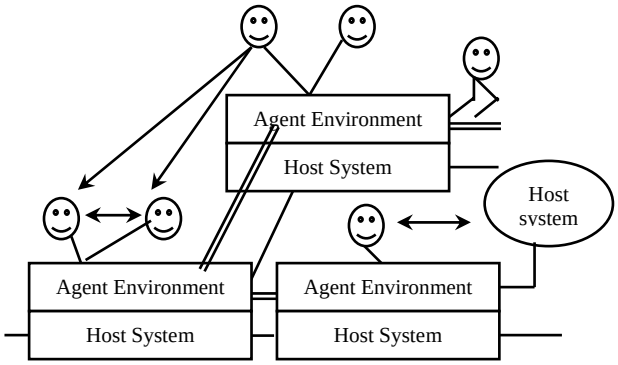

Figure 1. Mobile Agent Architecture

The mobile agent environment is built on top of a host system. An aglet requires a host Java application, an aglet host, to be running on a computer before it can visit that computer. When aglets travel across a network, they migrate from one aglet host to another. Each aglet host installs a security manager to enforce restrictions on the activities of un-trusted aglets. Hosts upload aglets through class loaders that know how to retrieve the class files and state of an aglet from a remote aglet host.

The smiley faces are mobile agents, which travel between mobile agent environments. They can communicate with each other either locally or remotely. Bi-directional arrows represent communication between mobile agents in the system. Finally communication can also takes place between a mobile agent and a host service [1].

D. *Aglets*

Aglets is a Java based mobile agent platform allowing a high portability of both the agents and the platform. Aglets includes both a complete Java mobile agent platform, with a stand-alone server called Tahiti, and a library that allows developers to build mobile agents and to embed the aglets technology for building mobile agents based applications. It is a Java agent which can autonomously and spontaneously move from one host to another carrying a piece of code with it. It can be programmed to execute at a remote host and show different behaviors at different hosts [5].

Aglets are possibly the most applicable mobile agent technology at the present moment in time.

Aglet architecture contains four basic elements aglet, proxy, context and identifier. Aglet is mobile agent that moves across the network. Proxy represents Aglet. Proxy shields the aglet from direct access to its public methods and it can hide the aglet's real location. Context is aglet's workplace. Stationary object provides a means for maintaining and managing running aglets in a uniform execution environment where the host system is secured against malicious aglets i.e. the Tahiti Server provides context to aglets created in it. Identifier is a globally unique and immutable throughout the lifetime of the aglet [4].

Aglet provides a graphical aglet viewer called Tahiti. Tahiti is an application program that runs as an agent server. It runs multiple servers on a single computer by assigning them different port numbers. Tahiti provides a user interface for monitoring, creating, dispatching, and disposing of agents and for setting the agent access privileges for the agent server. Most of the Tahiti operations are Aglet-dependent [6].

III. PROPOSED SYSTEMS

The application developed is an online examination, which is organized in different modules. These modules have different agents performing different tasks assigned to them.

A. *Systems Requirements*

Functional requirements in the proposed systems can be listed as,

- Authentication: only authorized user can appear the examination with valid username and password.
- Selection: examinee can select the subject and answer the question by selecting correct alternative.
- Timer: as examination is scheduled for fixed time, it will end when the time gets over irrespective of number of questions attempted.
- Ending test: allows the examinee to end the examination any time.

B. *Software Architecture*

Agents developed for online examination application belongs to the goal-based type agent category. Agents of this category keep track of current state as well as a set of goals it is trying to achieve and choose an action that will lead to the achievement of its goal.

The working of this goal-based type of agent [2] is as shown in fig. 2.

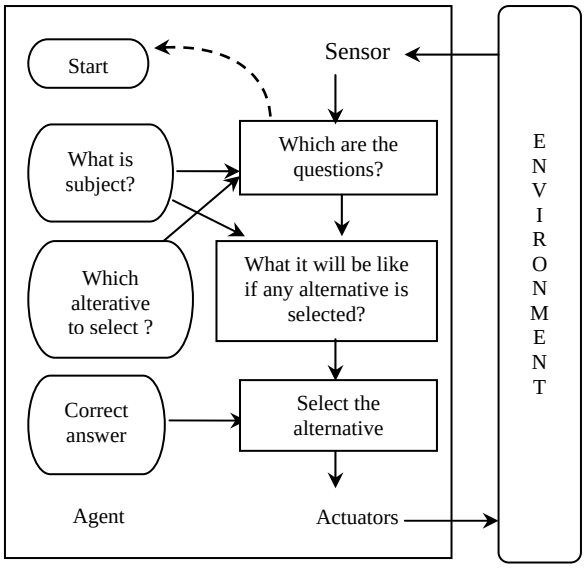

Figure 2. Goal-based Agent

The software architecture is built with three modules such as, user authentication module, examination module and result calculation module. Theses modules contain agents such as main agent, stationary agent and mobile agent to perform different roles. Main agent is responsible for conduction of examination where as stationary agent resides in the system where it is defined. Mobile agent carries data from on system to another.

User authentication module authenticates examinee with valid user name & password.

Examination module performs activities like selecting subject, retrieving questions for selected subject with its alternatives and correct answer. Retrieval of answer is task of finding answers that are relevant to examinee's questions. We have used probabilistic framework to retrieve the correct answer from the database [2].

In other words,

$$P (R = true \mid A, Q)$$

where A is the answer, Q is the question and R is the Boolean random variable.

By Bayes' rule, probability can be expressed as

$$P (r \mid A, Q) = P (A, Q \mid r) P (r) \mid P (A, Q)$$

where r denotes the value R = true.

This framework tries to find out correct answer (alternative) for a given question by using random variable as used in our Algorithm 2 (Examination Algorithm).

Once the examinee completes the examination, result module calculates the marks scored by the examinee for correct alternatives selected.

Interaction among modules is as shown in fig. 3.

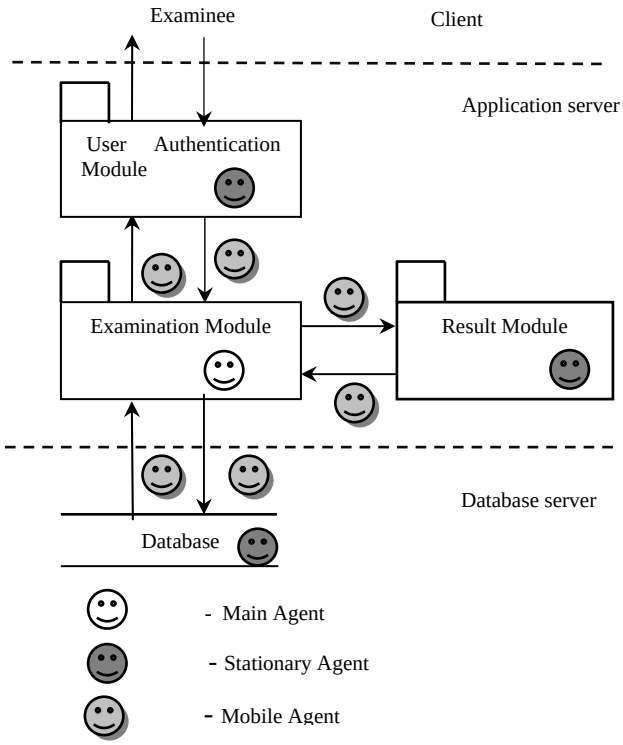

Figure 3. Software Architecture

Agents in examination systems are developed using Aglet 2.0.2 version [6] on Windows XP environment. JDK 1.5 is used for development of Java code for agents, as Java is excellent for designing and programming mobile agents. Microsoft Access is used as back end in the system for database.

The interaction between different agents in modules is as shown in fig. 4.

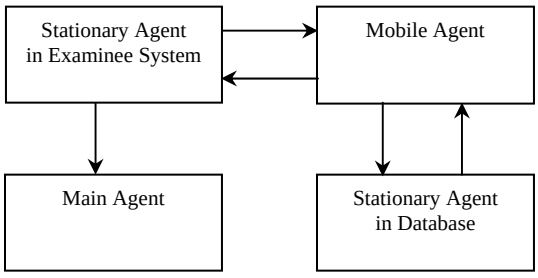

Figure 4. Interaction between Agents

Examination system starts working with authentication as examinee enters the user name & password. The stationary agent at examinee system passes the information about examinee such as user name and password to the mobile agent. The mobile agent submits this information to the stationary agent in the database. The stationary agent verifies this information with the details available in the database. The result of verification is handed over to the mobile agent. The mobile agent returns back to the examinee system and submit information about authentication to the stationary agent. If authentication is successful, the examinee selects the subject for the examination. For unsuccessful authentication, examinee needs to re-enter username and password. The stationary agent collects the information about subject selected and gives it to the mobile agent. The mobile agent submits it to the stationary agent at the database. The stationary agent processes the request and retrieves the questions with alternatives and correct answer. This retrieved data is handed over to the mobile agent. The mobile agent takes back the whole thing to the examinee system & submits it to the stationary agent. Stationary agent gives this information to main agent who is actually responsible for conducting examination. When the examination gets over, the stationary agent calculates the score for correct alternatives selected & displays result.

C. Database Schema Design

1) Student (Username, Password)
where

User name is user name of the examinee,

Password is password of examinee

2) Subject (Sub-ID, Sub-Name)
where

Sub-ID is ID of each subject,

Sub-Name is Name of subject.

3) Questions(Que-ID**,**Que, Alt1, Alt2, Alt3, Alt4, Answer)
where

Que-ID is ID of each question,

Que is the question,

Alt1 is the 1st alternative of particular question,

Alt2 is the 2nd alternative of particular question,

Alt3 is the 3rd alternative of particular question,

Alt4 is the 4th alternative of particular question,

Answer is correct answer of given question

4) Result **(**Username, Score)
where

Username is user name of examinee,

Score is marks secured by examinee

D. Proposed Algorithms

Algorithms for user authentication module, examination module and result calculation module in proposed systems are explained below.

1) *Algorithm 1: User Authentication Algorithm*

input:

username, username of examinee

password, password of examinee

output:

authentication of examinee

selection of subject

function user_authentication ( )

1. Get username and password of examinee

2.Mobile agent sends received username, password to database for authentication

msg.setArg ("username",username);

msg.setArg ("password",password);

3. Stationary agent at database retrieves username, password from database

String query = Select * from Student where username = "username" and password="password"

4. if (username, password received from mobile agent = username, password retrieved from database)

then

examinee is registered user

else

examinee is not registered user and

display error massage - login is failed

5. if (login is successful and

subject selected = subject from choice provided)

go to examination ( )

else

select subject from choice provided

2) *Algorithm 2: Examination Algorithm*

input:

subject, subject selected by examinee

questions, questions for subject selected

alternatives, alternatives to be selected as answer

timer, to keep track of time elapsed

question_count, counter to count questions

output:

alternatives selected as answer by examinee

function examination ( )

1. If (examinee chooses to start the examination)

then

establish connectivity with database

2. If (database connectivity is established)

then

stationary agent at database retrieves questions randomly from database for the subject selected

String query= Select * from Questions where Sub_id = "selSubject";

int rnd = rand.nextInt(noofrecords);

randVector.addElement(""+rnd);

else

no retrieval of questions and alternatives

3. if (all questions are retrieved)

then

set timer = zero

set question_count = one

else

retrieve remaining questions

4. do

start incrementing timer (timer = timer + 1)

main agent makes questions and alternatives available to examinee one by one

while (question_count <= total questions

or timer <= allotted time)

5. Stop answering questions

call calculate_result ( )

3) *Algorithm 3: Result Calculation Algorithm*

input:

question_count, counter to count questions

alternatives, alternatives selected by examinee

correct answer ,correct alternative of question

score, marks awarded for correct answers

output:

displays username

total no. of questions answered

score of the examinee

function calculate_result ( )

1. while (timer <= allotted time)

set flag = false

set score = zero

2. for (i = 1, i <= question_count , next question)

3. if (alternative selected = correct answer)

then

set flag = true;

else

no change in flag status

4. if (flag is set to true)

then

increment score by 1 (score = score + 1)

else

no change in score

IV. RESULTS OF EXAMINATION SYSTEMS

Examination systems will start after applying proper settings for aglets and database connectivity. Examinee enters user name and password through interface provided for authentication. Examinee selects the subject from the list of subjects provided. Authentication and selection of subject can be done through screen as shown in fig. 5.

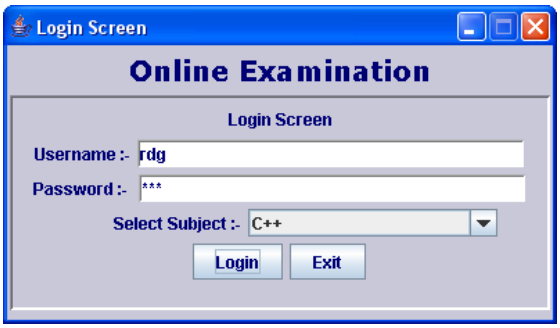

Figure 5. Authentication and Selection

If login is successful i.e. authentication of examinee and selection of subject is completed and verified by the systems, the systems will start as shown in fig. 6.

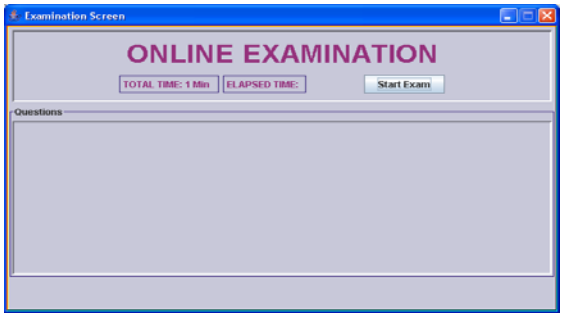

Figure 6. Start of Examination

Examinee can start examination by clicking on "Start Exam" button.

Examination will start by the display of questions for selected subject where one question will be displayed at a time on screen as shown in fig. 7.

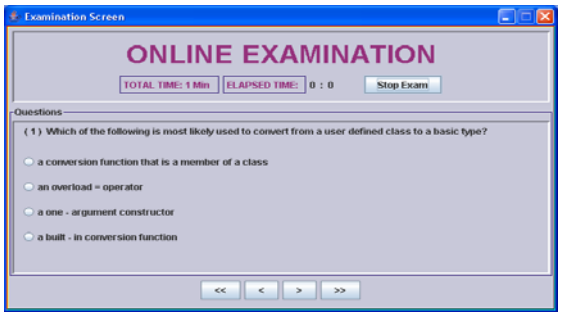

Figure 7. Display of Questions

Examinee can answer current question by selecting any one of the alternatives provided with the answer, can go to next question or can go to previous question.

After answering all questions provided, examination systems will ask whether examinee wants to stop the examination as shown in fig. 8.

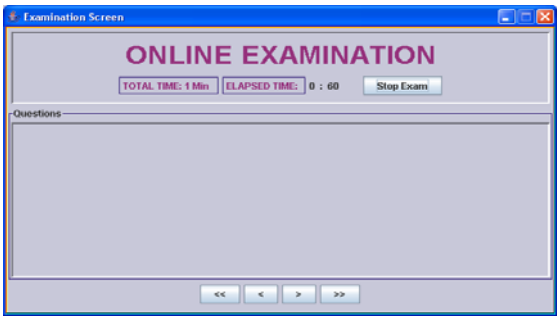

Figure 8. To stop Examination

The examination gets over either when examinee has answered all questions; examinee desires to terminate examination or end of allotted time period.

As soon as the examination gets over, result of examination will be displayed to the examinee as shown in fig. 9.

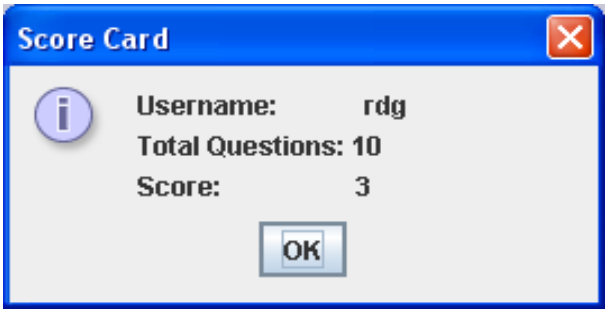

Figure 9. Scorecard of Examinee

V. CONCLUSION

The proposed and implemented system in Java works properly in Aglet and Windows XP environment. Agent technology allows conducting of examination in parallel for more students. Issues such as transaction processing and concurrency control are taken care of in this system. The system is well tested and yields the good results.

Our model, which uses multi agent systems, provides a more flexible paradigm against existing client-server implementations. Our system supports better utilization of network bandwidth, improves candidate's response time avoiding network delays, and thus increases network performance. Agents in this model exhibit properties such as goal-oriented, reactive, autonomy, asynchronous operation, communicative and mobility and thus it acts as a goal-based agent.

For authentication of examinee biometric security can be used. Information retrieval model such as vector space model can be applied in this implementation if more than two alternatives are correct answers for the given question.

REFERENCES

[1] Bill Venners,"The Architecture of Aglets, The Inner Workings of an Agent Technology", Java World, April 1997

[2] Stuart Russell and Peter Norvig, "Artificial Intelligence- A Modern Approach ", Pearson Education, Second Edition, 2003

[3] Danny B. Lange, Mitsuru Oshima, "Programming and Deploying Java Mobile Agents with Aglets", Addison Wesley 1998

[4] Danny B. Lange and Mitsuru Oshima, " Mobile agents with Java: The Aglet API", World Wide Web (1998), Springer Netherlands, pages 111–121,Volume 1, Number 3 / September, 1998

[5] http://en.wikipedia.org/wiki/Aglets

[6] http://www.trl.ibm.com/aglets