A Server-Based Multi-threaded System for Election Results Collation in Nigeria

N.C. Woods & T.E. Idowu
Department of Computer Science
University of Ibadan
Ibadan, Nigeria

ABSTRACT

In a distributed environment, data retrieval jobs deal with the collection of data from several and generally heterogeneous information sources. A mobile agent is an executing program that can migrate from machine to machine in a heterogeneous network. On each machine, the agent interacts with stationary service agents and other resources to accomplish its task. Consequently, mobile agents are mainly attractive in distributed information retrieval applications. On behalf of a user, the agent can search the resource locally by migrating to that location, thereby eliminating the transfer of intermediate results across the network and reducing the Network latency. This work is aimed at designing and implementing a server-based multi-threaded application that is capable of migrating from one host to the other in order to merge the results of an election from different locations. The current process of transferring data (the election results) from one location to another has inherent risks such as insecurity of data characterized by stealing or hijacking of the results, loss of data, manipulation of results, ballot stuffing, to mention a few. This work develops a system which can control and manage the distributed information retrieval processing in order to retrieve information from distributed database throughout a network. In this system, the mobile agents can migrate from a host node to various destinations, perform some data processing activities and send the relevant information back to the host. The Java-based mobile agent system, Aglets workbench, were fully deployed in order to develop the system.

Keywords: Mobile agent, Result collation, Election result, Aglets.

African Journal of Computing & ICT Reference Format:

1. INTRODUCTION

As digital data communication systems grow bigger, better and more affordable, their application areas also widen. For instance, it is possible for many people in the third world nations to engage in electronic transactions ranging from funds to data, with the help of their mobile phones and computers. It should be possible to also employ the data communication systems in governance. All nations of the world are moving towards being self-governing and this usually goes with elections. For some nations, the most tasking and challenging aspect of conducting a free and fair election is the collation of election results from various polling centres. In Nigeria and most third world counties, the current process of transferring data (the election results) from one location to another has inherent risks and challenges such as insecurity of data characterized by stealing or hijacking of the results, loss of data, manipulation of results, ballot box ‘stuffing’, difficult terrains, and in the extreme case it could lead to loss of lives.

The existing system of voting in Nigeria is the ‘traditional’ method; which is done by distributing ballot boxes to all the polling centres in the local government (LG) areas of the nation, where each eligible voter is expected to come and vote for the candidate of their choice. Thereafter, ballot papers are counted in each polling centres in the presence of voters and party representatives, then the total votes for each party is announced and recorded on paper. This result is then hand-carried to the collating centre for the LG and then from the LG to the state level and finally to the National level for the final collation. Figure 1 depicts this process.

Figure 1: Existing mode of election result collation in Nigeria
2. MOBILE AGENTS & THEIR INFRASTRUCTURE

2.1 Mobile Agents

A Mobile Agent is a type of software agent [9], with the features of autonomy, social ability, learning, and most importantly, mobility because mobility ensures efficiency, persistence and peer-to-peer communication [6, 10]. Agents could also be defined as a computational entity, which acts on behalf of others: it is autonomous, asynchronous and optionally intelligent with possible attributes of migration [11]. A Mobile Agent is also defined as a computer system, situated in some environments, that is capable of flexible autonomous action in order to meet its design objectives [12, 13].

The problem of transferring results from different LGs in the state to the state Independent National Electoral Committee (INEC) secretariat is still a major problem/setback when conducting election as it causes delay in collating, processing and announcing the election result. Evidently, the most tasking and challenging aspect of the processes involved in any election is the collation of results from various polling centres. There is a need to improve on the way this data (votes) is being collated, transmitted and processed.

With the availability and proper use of modern information and communication technology/applications, it is possible to transfer and collate the data without physically moving the data. In traditional Client-Server mode of communication, a good number of the information retrieval systems do not offer enough flexibility for distributed data repositories [1, 2]. There are many specifications in the traditional way; setting up a connection between the client and the server, sending a request to the database server and receiving the result from the server. If there are X servers in the network, the user has to start X network connections and send out X database queries. The network connection must be maintained all through the process. When there are more and more mobile devices, the bandwidth becomes limited and the devices cannot be online always.

At that time, the traditional client-server approach would be cumbersome to run. A popular current solution to resolve these problems is a distributed and flexible mobile agent-based architecture [2, 3, 4]. Using Mobile Agents (MA), it is possible to retrieve data from multiple information systems that exist in a distributed environment [5, 6, 7, 8]. This work is aimed at designing and implementing a server-based multi-threaded application, with the aide of mobile agents, which is capable of migrating from one host to the other in order to collate the result of any elections held in Nigeria.

The paper outline is organised as follows: Section 2: discusses Mobile Agents and their infrastructure; Section 3 sheds light on the proposed system; Section 4 discusses the Implementation of the system and Section 5 concludes the paper.

The three key defining concepts are situatedness, autonomy, and flexibility. The authors explained situatedness to means that the agent receives sensory input from its environment and its actions affect the environment, while flexibility indicated that an agent is responsive, pro-active, and social. Some other authors [14] say that a mobile agent is a process that can transport its state from one environment to another, with its data undamaged, and still perform appropriately in the new environment. Mobile agents can decide when and where to move and this movement evolved from RPC (Remote Procedure Call) methods [10].

Briefly, some of the attributes that define an agent were identified as autonomy, communication ability, reactivity, mobility and pro-activity [12]. These will ensure that an agent can (1) operate without direct intervention of human users or without having to depend on human users creation and deployment., (2) interact with other agents or human users to collaborate, negotiate, or coordinate in order to define and perform its tasks, (3) perceive its environment and responds to cues automatically without delay and having to consult its human user, (4) move across operating environments and perform tasks remotely, (5) reason, plan and execute tasks by taking the initiative without prompting, respectively.

Mobile agents are active in that they can choose to migrate between computers at any time during their execution. Mobile agents are sometimes identified by the kind of job they are employed to perform. In this notion, they can be classified as: collaborative agents, interface agents, smart agents, information agents, reactive agents and hybrid agents [10, 15]. Other classifications also exist.

2.1.1 Key Characteristics of Mobile Agents

1. Migration

Mobility is the characteristic that allows agents to move from one network node to another; however migration determines how this transfer is achieved [16]. Although a MA is essentially an executing process, the governing factor that distinguishes it from a normal process is the fact that not all of its instructions have to be executed on the same node. With MAs, it is the agent that decides when to move and the underlying infrastructure must support and execute this request. Enforced migration can be imposed upon an agent in extreme circumstances, such as, if the agent attempts to perform a forbidden action. The two approaches to moving agents between nodes are [16]: State-oriented: This system allows an agent to move at any point in their execution. When the command is initiated, the current state of the agent is encapsulated and transferred across the network to the receiving network node. Once received, execution of the agent resumes from where it stopped. And Stateless: where the agents can move at any point in the code, but restart their execution from the beginning of their code rather than the point of migration.

Vol 7. No. 5 – December, 2014
African Journal of Computing & ICT

© 2014 Afr J Comp & ICT – All Rights Reserved - ISSN 2006-1781
www.ajocict.net
2. Data Acquisition
To acquire the information needed to achieve their goals, MAs usually interrogate their local environment [16]. This information needs to be filtered locally by the agents before it is either stored with the agent or forwarded to some receiving destination (such as the original network node of the agent).

3. Route Determination
Once the agent has finished with a network node, it must make a decision of where to move to next. The acquisition of data leading to this decision can be derived by one of three methods: Predetermination, Dynamic determination, Hybrid determination [16].

4. Communication
It is fundamental that MAs should have the ability to communicate. The parties that a mobile agent wishes to converse with include Local environment, other agents as well as Users. There are two methods for agent communication to take place [16, 17].

- **Networked-oriented:** Agents communicate through some network-based mechanism, such as message passing. This means that the communicating parties do not have to be residing on the same node or on the same network.
- **Node-oriented:** Agents communicate through local inter-process communication mechanism, such as files, shared memory or anonymous pipes. This means that the communicating parties must be currently executing on the same network node.

Communication can also take place in two basic forms: **Synchronously**, where, the communicating parties must arrange for a time to communicate and must be synchronized before data can be transferred. This is typically used for situations where the data is important and needs to be confirmed. While in **Asynchronous form**, the communicating parties can communicate with each other at will: the data is received as the receiving party checks for it. This form is generally used for transferring informative data. [16, 18].

2.1.2 The life cycle of a Mobile Agent
Irrespective of what kind of MA, their life cycle is similar. An MA consists of the program code and the program execution state, that includes the current values of variables, next instruction to be executed among other parameters [1, 19]. The agent initially resides on the home machine, from where it is then dispatched to execute on a remote computer called a host machine. When an MA is dispatched its entire code and execution state is transferred to the host machine. The host provides a suitable execution environment and resources (CPU, memory, etc.) for the MA to perform its task. After completing its task on the host, the mobile agent migrates to another host to execute and this continues until the mobile agent returns to its home machine after completing execution on the last machine in its itinerary. In summary we can say that a mobile agent experiences the following events in its life cycle [9, 10, 19]:

- **Creation:** a brand new agent is born and its state is initialized.
- **Dispatch:** an agent travels to a new host.
- **Cloning:** a twin agent is born and the current state of the original is duplicated in the clone.
- **Deactivation:** an agent is put to sleep and its state is stored on a disk of the host.
- **Activation:** a deactivated agent is brought back to life and its state is restored from disk.
- **Retraction:** an agent is brought back from a remote host along with its state to the home machine.
- **Disposal:** an agent is terminated and its state is lost forever.

2.1.3 Benefits of Mobile Agent
The use of mobile agents has several advantages and following are some the areas that may benefit from appropriate use of mobile agents [20, 21, 22, 23, 24, 25].

1. **Reduction of network traffic/ load:** software codes are often smaller than data that it processes. Therefore, the transfer of mobile agents to the sources of data creates less traffic than transferring the data and this in turn reduces the network bandwidth consumption.

2. **Space savings:** Because a mobile agent resides only on one node at a time, while carrying its functionality with it, the agent does not have to be duplicated, thereby saving space.

3. **Support for heterogeneous environments:** If the framework is in place, agents can target any system because MAs are separated from their host by, mobility framework.
4. **Robustness and fault tolerance**: If a distributed system starts to malfunction, then mobile agents can be used to increase availability of certain services in the concerned areas. E.g. the density of fault detecting or repairing agents can be increased. In addition, fault-tolerance prevents a partial or complete loss of the agent.

5. **Easy software upgrades**: An MA can be exchanged virtually at will. In contrast, swapping functionality of servers is complicated, especially if we want to maintain the appropriate level of quality of service (QoS).

6. **Asynchronous and Autonomous Execution**: Once a mobile agent is dispatched from the home machine, the home machine can disconnect from the network. The mobile agent then executes autonomously without the intervention of the home machine. The home machine can reconnect at a later time and collect the agent and its data. This allows it to overcome network latency.

7. **Freeing the user to log out or migrate since the agent’s life is independent of the user’s session.**

2.2 **The Mobile Agent Infrastructure**

The development of MA infrastructure as shown in figure 3 provides a framework for code mobility. Every network component (NC) contains a Mobile Code Daemon (MCD) running within a Java Virtual Machine (JVM). The MCD provides a number of services that facilitates the execution of mobile agents, which include: a Mobile Code Manager (MCM) that manages the life cycle of a mobile agent, a Migration Facilitator (MF), to transport mobile agents between NCs, a Communication Facilitator for collaboration between local and remote mobile agents, and an interface called the Virtual Managed Component (VMC) which provides for mobile agents accessing the NC’s managed objects and resources in a controlled and secure way. The VMC is responsible for management of the mobile agents’ access rights and the allocation of resources to that agent [26, 27].

3. **THE PROPOSED SYSTEM**

We propose a system using the above MA infrastructure that has host computers on a network, where each host is to capture the result of the election conducted in a polling station (PS). The system will also have a host/server computer located in each local government area (LGM) of a State as well as a server at the State level (SHM) of the Electoral Office. After voting in each polling station, the result of that polling station is entered into the host computer for that polling station and then stored on a database resident on the host. We propose a three-level home machine system whereby the LG area servers are ultimately the host machines, with respect to the State home machine (SHM). These SHM are also host machine, with respect to the Central server at the INEC Abuja secretariat (CSM). Figure 4 shows a sub-section of this structure.

4. **SYSTEM IMPLEMENTATION**

Each node (PS, LGM, SHM, CSM) will have a database to which all results will be logged in and read from, ultimately by the agent. The master computer dispatches several clones of the same agents to all the computers on the network to gather the results. Figure 5 shows the diagrammatic model of the proposed system and interaction among the different components. The processes involved are listed below:
i. Given number of nodes as \( m \) and a master node as \( p \)

ii. At a given time \( T_1 \) (on completion of voting exercise), the master node \( p \) creates \( m \) clones of aglets \( a \) and dispatches each clone to the various nodes.

iii. On arrival at the nodes, the aglets query the database for the results and build a serializable object that contains all the information needed from the node.

iv. After a defined time \( T_2 \), the master node tries retracting the aglets from their context. It can only retract aglets instances that are done querying and compiling the results.

v. After all compilation is done, a rebuild process that involves showing the results of the poll is displayed based on the information obtained from each aglet.

vi. If an aglets finishes before time \( T_2 \), the aglets is made to sleep (deactivated).

Figure 5: The diagrammatic model of the proposed system and interaction among different components

To this effect, the following was developed:

1. The polling software has a backend database, implemented with a MySQL.
2. The polling software will be deployed on computers that will be connected to the INEC voting network.
3. The computers will be aglet enabled.
4. The Main computer / master will have the Tahiti server running on it and will have the ability to create, clone, dispatch and retract aglets at any given time \( T_2 \).

Dispatching an Agent: The agents are deployed /dispatched to another node, with the use of the Aglet Transfer Protocol (ATP) (see figure 4) connection. More than one instance of aglet Tahiti server can be run on a machine, in which each of the instance represents a node with its port number and can be dispatched using local host.

Visited Nodes/ results: After time \( T_2 \) a window displays a list of all the nodes visited by the Aglet (agent), it can also display the results from all the visited nodes, at the click of a button. Figure 6 shows an instance of the result from a remote host which is displayed on the command prompt of the remote host with its port number (6000). The data displayed are in this order: state_id, lga_id, party_id, post_id, unit_id, ward_id, total votes, date/time uploaded.

Figure 6: An instance of the result from a remote host
The result is stored in the table “Results” on the remote location database. This is the result that will be picked up by the agent and then returned to the Home server database.

**Result Compilation:** After the agents are retracted, the results gathered are added to the home server database table “accum_result”. Figure 7 displays this action and the results are pooled from the result table of each of the remote host. While figure 8 shows the MySQL view of the accumulated results from the various nodes visited.

**Election results:** After all the agents have been retracted and their data gathered, the pool results can then be displayed as shown in figure 9. This is the window that displays the total result of all nodes for each party. The result can only be viewed on the main server/home node, after the result has been summarized by the agent.
Strength of the Proposed System

The proposed system has the following strength compared to the existing system, thus:

1. Ability to retrieve information remotely
2. Ability to retrieve and manage information under limited network bandwidth and latency
3. Provides simplicity and flexibility when conducting elections compared to the existing system.
4. Ability to process results on site and returning the summary of the result.

5. CONCLUSION

In conclusion, there is the urgent need to imbibe the mobile agent technology in the process of election result collation, which was designed to facilitate easy retrieval and collation of election results in a distributed environment like Nigeria. This work was developed to ease the process of accessing and retrieving information from various remote machines with limited network bandwidth and latency. The remote hosts are deployed with agents execution environments which listen to the incoming signals. This work would incorporate more flexibility and accessibility to information in a distributed computing environment. If implemented, the Nigerian electoral process would be smoother, faster and safer, especially the aspect of result collation.

REFERENCES