UDC 519.68:519.86
Original scientific paper
Received: 20.09.2003.

Distributed mobile agent model for multi-area power system on-line economic load dispatch

Kannan Nithiyananthan and **V. Ramachandran**
Center for Professional Development Education, College of Engineering, Guindy Anna University,
Chennai 600025, INDIA
e-mail: knithiyananthan@hotmail.com

SUMMARY

The main objective of this paper is to construct a mobile agent model in a distributed environment through which the Economic Load Dispatch (ELD) solutions of multi-area power systems can be monitored and controlled. In the proposed model for ELD, a mobile agent is created which enables the mobility of the economic dispatch executable code to the neighboring power system client host. A single server (main container) /multi client (sub container) based architecture enables the power system client's to access the economic load dispatch mobile agent at it's own framework with their respective data and to get the economic load dispatch solutions. A distributed agent model has been implemented in such a way that for every specific period of time, the ELD agent migrates from one power system client to another to obtain the system data; the optimized economic load dispatch solutions with a power loss have been sent back to the respective clients in a heterogeneous environment. The ELD agent creates a new thread of control for every client's request and hence a complete distributed environment is exploited.

Key words: *Economic Load Dispatch, multi-area power system, mobile agent model, client-server architecture.*

1. INTRODUCTION

The economic dispatch problem of power systems is to determine the optimal combination of power outputs for all-generating units which minimizes the total fuel cost while satisfying the constraints. The power system economic load dispatch solution obtained through a conventional client-server architecture is complicated, the memory management is difficult, the source code is bulky and the exception-handling mechanism is not so easy. In the conventional power system operation and control, it is assumed that the information required for monitoring and controlling power systems is centrally available and all computations are to be performed sequentially at a single location [1]. With respect to sequential computation, the server has to be loaded every time at each client's request and the time taken to deliver the economic load dispatch (ELD) solution is also comparatively high [2, 3, 4].

This paper outlines a new methodology for solving the economic load dispatch in a distributed environment. The mobile agent based client-server architecture overcomes the difficulties associated with a sequential computation and it can be easily implemented.

The mobile agent framework provides a number of advantages including the saving of network bandwidth and increasing the overall performance by allowing the ELD agent to process the power system data in the client machine itself. The proposed agent framework supports the asynchronous processing that makes ELD agent to fulfill a given task without the need to have a permanent connection from the client to the main container. The proposed architecture makes the economic dispatch application more fault tolerant where the network failure can influence only the migration of an ELD agent as the rest of the process is then performed locally on the same node.

2. DISTRIBUTED AGENT MODEL FOR AN AUTOMATED ECONOMIC LOAD DISPATCH SOLUTION

In the present paper, a distributed environment has been set up using the Java Agent Development Framework, see Ref. [5], based mobile agent framework to estimate and to monitor the economic load dispatch solutions for different sub-systems of an integrated power system. Each sub system has been considered as a power system client and hence a multi power system clients - single ELD agent model is implemented. In this model the tie line power flow for each area is assumed to be constant and the tie line power flow is treated as a load for each sub system. These power system clients are interconnected in a network where the ELD agent is always on the move as shown in Figure 1. A client computer basically effects the distributed power system monitoring through a GUI for every specific period of time and frequently exchanges data with the ELD agent. The ELD agent performs the economic load dispatch computation and then distributes the results. Chronologically, the ELD mobile agent should start the process first in the main container (*hostl*), so that it can take the initiative to set up a connection link. The mobile agent object obtains the necessary data from the container and responds back to them with the economic dispatch solutions. Then it migrates to the next host and it serves the economic dispatch executable code. This total process of migration of the ELD agent from one host to another is repeated for every specific time interval. The transaction of the ELD agent from different host takes place several times, so the possibilities of the occurrence of errors and security for the power system data may be high. Hence it must be handled properly and it takes care of a JADE distributed environment automatically.

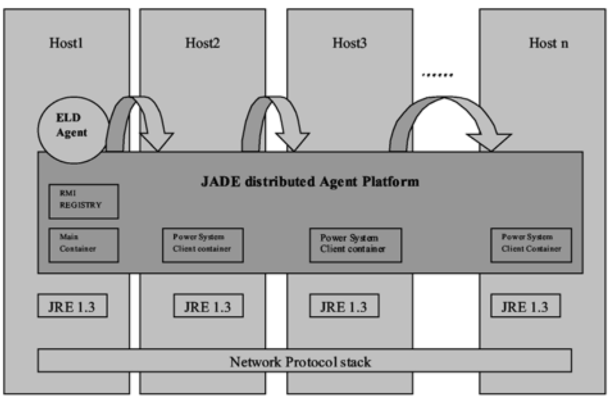

Fig. 1 The ELD mobile agent based architecture for the economic load dispatch

2.1 Mobile agent flow model [2]

In this proposed mobile agent model, each neighboring power system is considered as a client container and the ELD mobile agent is made to migrate from one container to another as shown in Figure 2. At first instance the mobile agent calls one of its environmental methods requesting its migration to the new environment *Env2* and the current environment creates a network connection with the new environment. Once the new environment accepts the network connection from the current environment then the current environment uses an ObjectOutputStream to serialize the mobile agent to a stream of bytes and sends the stream to the new environment. At another instance the new environment calls the readObject method of an ObjectOutputStream so it can reconstruct the ELD mobile agent from the byte of the stream. After this instance the new environment sends the acknowledgement to the current environment that it had received the agent. Since the instance of the mobile agent in the new environment is not yet active, the current environment tells the instance of the mobile agent to shutdown and it sends a confirmation to the new environment that the old instance had shutdown. Once this shutdown status confirmation is received, the new environment allows the mobile agent instance to be active. This collaboration of work makes the mobile agent to migrate to a new environment.

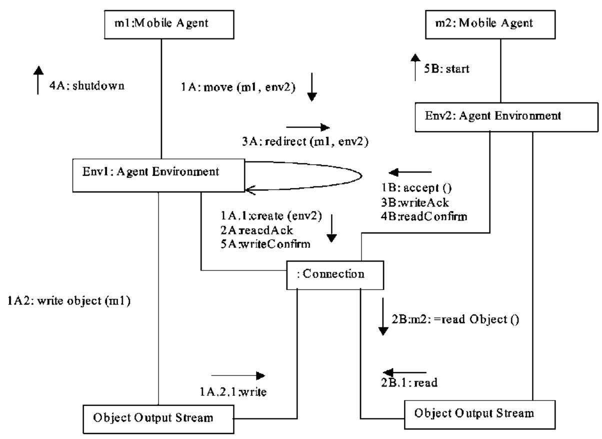

Fig. 2 The ELD mobile agent migration

2.2 Mobile agent paradigm

The major components of the mobile agent architecture are an agent manager, a security manager, an application manager and a directory manager as shown in Figure 3. All the mentioned major components have individual responsibilities to perform. The agent manager receives the ELD agent for execution on the local host and also sends agents to remote power system clients and before the transportation of the agent, the agent manager serializes the agent and its state. It then passes the serialized form to its counterpart on the destination host. In a highly reliable architecture it actually passes the agent to the reliability manager, which ensures that the agent manager on the remote host receives the agent. Upon receipt of an agent, the agent manager reconstructs the agent and the objects it references and then creates its execution context. The Security manger authenticates the agent before it is allowed to execute. Thereafter, the Java virtual machine automatically invokes the security manager to authorize any operations using the system resources. The ELD mobile agents may use the directory manager to identify the location of the next neighboring host. An arriving mobile agent accesses to resident servers such as database servers through this gateway. The ELD Logic encapsulates the agent behavior and the logic of economic dispatch application. This architecture prescribes that an agent is a composite Java object that includes mobility, persistence and can communicate with other hosts. During its life cycle a mobile agent receives various kinds of events in response to its actions. For instance, if it moves to another host, a mobility event occurs just before and after the migration and the corresponding call - back mechanism is invoked. In this way, each event gives to the agent the opportunity to determine how to react. A programmer implements a mobile agent filling its call - back methods such as before Dispatch and after Arrival as appropriate.

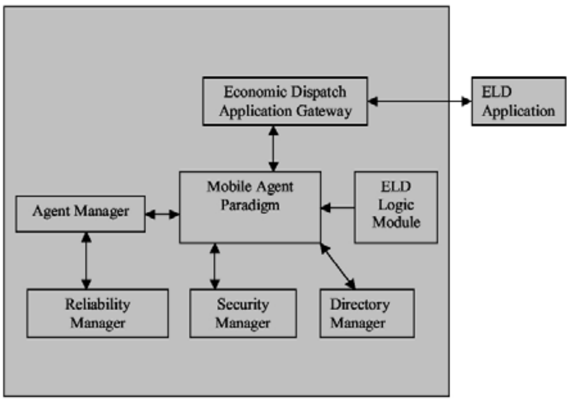

Fig. 3 The ELD mobile agent paradigm

3. ELD AGENT LIFE CYCLE

An ELD mobile agent can be in one of several states, according to agent Platform Life Cycle in FIPA specification; these are represented by some constants in the Agent class in the JADE environment as shown in Figure 4. The states are: (a) AP_INITIATED: the ELD agent object is built but has not registered itself yet with the Agent Management System, it has neither a name nor an address and cannot communicate with other agents; (b) AP_ACTIVE: the ELD agent object is registered with the AMS, has a regular name and address and can access all the various JADE features; (c) AP_SUSPENDED: the ELD agent object is currently stopped. Its internal thread is suspended and no agent behavior is being executed; (d) AP_WAITING: the Agent object is blocked waiting for the message. Its internal thread sleeps on a Java monitor and it will wake up when a certain condition is met (typically when a message arrives); (e) AP_DELETED: the ELD agent is definitely dead. The internal thread has terminated its execution and the Agent is no more registered with the AMS; (f) AP_TRANSIT: a mobile agent enters this state while it is migrating to the new location. The system continues to buffer messages which will then be sent to their new location; (g) AP_GONE: JADE internally uses this state when a mobile agent has migrated to a new location and has a stable state.

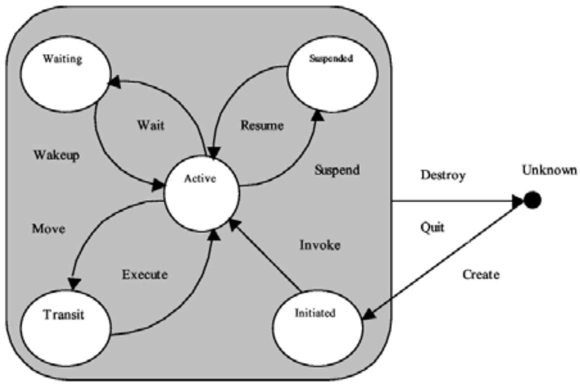

Fig. 4 Mobile agent life cycle

4. RESULTS

The above distributed mobile agent model has been implemented in Windows NT based HP workstations connected into an Ethernet LAN. The results are shown in a GUI as given in Figure 5.

The above GUI shows the economic load dispatch solution for a specific 3-generator bus power system client. In this GUI, once the start button event is triggered the ELD agent will receive the power system data and will start the computation of the economic dispatch solution. Once the economic dispatch solution is displayed through the GUI move button the event has be triggered by selecting the container or environment in the available locations window where the ELD mobile agent has to move. Using this approach, different power system clients can monitor continuous updated optimized economic load dispatch solutions at regular time intervals.

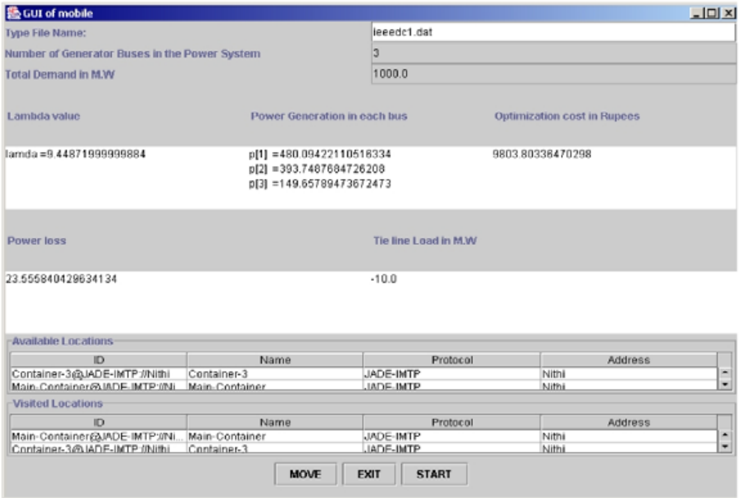

Fig. 5 GUI with economic load dispatch solution

5. CONCLUSION

An effective distributed agent model has been developed to monitor the economic load dispatch of the multi-area power systems. It has been tried out in overcoming the overheads associated with the sequential power system economic load dispatch computation through this model. Although, the client-server architecture for the economic load dispatch solution is very well established, the value of this study lies in the fact that it emphasizes a unique methodology based on the mobile agent model to serve a large number of clients in a distributed power system environment, across various JADE platforms based on the communication between virtual machines. A practical implementation of this approach suggested in this paper was assessed based on 3, 6, 9 and 13 bus sample systems. Accordingly, the proposed model can be implemented for a large power systems network spread over geographically apart.

6. REFERENCES

[1] A.J. Wood and B.F. Woollenberg *Power Generation, Operation and Control*, John Wiley & Sons, New York, 1984.

[2] G. Bandyopandhyay, I. Senguptha and T.N. Saha, Use of client-server model in power system load flow computation, IE (I) *Journal-Electrical*, Vol. 79, pp. 199-203, 1999.

[3] D.G. Hart, D. Uy, J. Northcote-Green, C. Laplace and D. Novosel, Automated solutions for distribution feeders, *IEEE Computer Applications in Power*, pp. 25-30, 2000.

[4] B. Qiu and H.B. Gooi, Web based SCADA display systems (WSDS) for access via internet, *IEEE Transactions on Power Systems*, Vol. 15, No. 2, pp. 681-686, 2000.

[5] F. Bellifimine, G. Caire, T. Trucco and G. Rimassa, JADE Programmer's Guide.

[6] MarkGrand, *Java Enterprise Design Patterns*, Vol. III, Wiley computer publishing, 2002.

DISTRIBUCIJSKI MODEL MOBILNOG AGENSA ZA VIŠESTRUKI/VIŠENAMJENSKI ENERGETSKI SUSTAV ON-LINE OTPREMANJA EKONOMSKOG OPTEREĆENJA

SAŽETAK

Glavni cilj ovog rada je razviti model mobilnog agensa u distribuiranom okruženju pomoću kojeg se mogu pratiti i kontrolirati rješenja za ekonomičnu otpremu tereta (ELD) u višenamjenskom energetskom sustavu. Kod predloženog modela za ELD, stvara se se mobilni agens koji omogućava pokretljivost izvršnog koda ekonomične otpreme u susjedni energetski sustav klijenta domaćina. Arhitektura koja se temelji na jednom serveru (glavni kontejner) / za više klijenata (pod-kontejner) omogućava klijentu energetskog sustava pristup mobilnom agensu za ekonomičnu otpremu tereta u vlastitom okviru s odgovarajućim podacima kako be se dobila rješenja za ekonomičnu otpremu tereta. Distribucijski model agensa (model distribucijskog agensa) primjenjuje se tako da se za svako posebno vremensko razdoblje ELD agens seli od jednog klijenta energetskog sustava do drugog kako bi dobili podatke o sustavu; optimalna rješenja za ekonomsko otpremanje opterećenja s gubitkom energije šalju se natrag do odgovarajućih klijenata u heterogenom okruženju. ELD agens stvara novu mrežu kontrole na zahtjev svakog klijenta te se stoga može koristiti kompletno distribucijsko okruženje.

Ključne riječi: *ELD, višenamjenski energetski sustav, model mobilnog agensa, klijent/server arhitektura.*