8th International Conference on Power Electronics - ECCE Asia
May 30-June 3, 2011, The Shilla Jeju, Korea

Multi-Agent System for Optimization of Microgrids

Foo Y.S. Eddy[1] and Gooi H.B.[1]

[1] School of Electrical & Electronic Engineering, Nanyang Technological University, Singapore

Abstract-- **Microgrids are low voltage networks usually located at the consumer end of the distribution system. It typically consists of consumer loads, energy storage and small generation systems and is capable of islanding to protect itself against grid supply interruption. With the increased awareness of clean energy power systems, renewable technologies such as solar PV, wind turbines and fuel cells are gradually emerging within the power system network, and control and management for these equipment are necessary to ensure the stable operation of microgrids. However, existing centralized control systems are unable to handle the large number of renewable components and thus, a decentralized control scheme called Multi-Agent System (MAS) is introduced to manage these components. The implementation of distributed control will include JADE as the platform for agent communications as well as developing customizable agents for specific microgrid requirements such as ancillary services, power trading and negotiation and network security.**

Index Terms--**Distributed intelligent control, microgrid, multi-agent system, power market.**

I. Introduction

With the increased awareness of global warming, carbon emissions and escalating fuel prices, many research activities have focused on incorporating clean and renewable energies e.g. solar, wind, hydro and fuel cells into the electrical networks thus in an effort to encourage use of renewable technologies at the consumer's level, incentives are also awarded to households which install photovoltaics (PVs) or small wind turbines [1]. By encouraging the installation of such technologies, it can be observed that the supply of electricity is gradually shifting away from centralized generation to a more distributed, plug and play form of generation. Existing electrical networks will also see a transition from passive to a more active type of network which give rise to challenges in managing power flow and stability issues within the network [2].

Due to the anticipated increase in penetration of Distributed Energy Resources (DERs) at the microgrid level, control and management are necessary to ensure smooth and stable operation of microgrids. However, traditional centralized intelligence approaches proved to be inadequate to cope with the increasing growth of DERs due to the lack of flexibility and extensibility [3]. Moreover, centralized control was initially designed to handle large generation units. With numerous DERs appearing in the power network, it is difficult or nearly impossible to control the entire system by a single central controller adopting a tops-down approach [4, 5]. If such a controller is implemented, it would require increased cost for communication infrastructure and introduce added complexity in the centralized control supervisor. Although there are disadvantages adopting the central control scheme for microgrid applications, centralized control can offer broader observability of microgrid operations and greater knowledge for making control decisions as part of its tradeoffs [6].

To overcome this problem, Multi-Agent Systems (MAS) was proposed as an alternative to centralized control because it can effectively handle multi-objective and multi-constraint complex systems by decomposing complex tasks to several simpler tasks to accomplish its goals [7]. MAS is a form of decentralized control as it uses distributed intelligence by employing software entities or agents to communicate, negotiate and optimize microgrid operations. As opposed to centralized approach, MAS uses a bottoms-up approach to manage and optimize microgrid operations so that communications and complexity of microgrids are kept to minimal. In the following sections, we propose a framework and methodology on developing customized agents for power trading and power management within a microgrid and uses Java Agent Development (JADE) toolkit as a platform to implement the proposed control scheme.

In this paper, Section II will briefly introduce an overview of the Microgrid Energy Management System (MG-EMS) located at Laboratory for Clean Energy Research (LaCER), Nanyang Technological University (NTU). It will present our approach to implement MAS in the NTU microgrid prototype. Section III will discuss the MAS framework, concept, control architecture, agent theory and standards for agents. Section IV will discuss the proposed methodology. It will briefly discuss optimization objectives and provide some description on JADE working environment. Section V will discuss the functions of our customized agents and display some simulation results from JADE. Section VI will conclude by providing a summary of research tasks accomplished and also provide some directions for our future research work.

II. Overview of NTU Microgrid

The NTU microgrid prototype was developed in LaCER at the School of Electrical and Electronic

This research work has been supported by A*STAR under the Microgrid Energy Management System Project (Grant no. 072 133 0038).

978-1-61284-957-7/11/$26.00 ©2011 IEEE

Engineering (EEE). Within the microgrid, several electrical components ranging from programmable AC sources to renewable sources such as solar PVs, wind turbines, fuel cells, and energy storage in the form of battery banks are used to supply electrical energy to the 0.4-kV low voltage network. A web based MG-EMS server system is also used to manage these components as well as to optimize its operations [8]. MG-EMS itself consists of Supervisory Control and Data Acquisition (SCADA), Load Forecasting (LF), Unit Commitment (UC), State Estimator (SE), Optimal Power Flow (OPF) and Automatic Generation Control (AGC). Figure 1 provides an overview of how DERs are connected and controlled within the microgrid.

In terms of control architecture, the NTU microgrid can be viewed as adopting a centralized control scheme. All the control decisions and optimization are done on the central MG-EMS server by sensing real time measurements from the DERs and energy storage through the use of smart meters and intelligent energy sensors. After processing the measured data, control pulses are sent back to the DER converters and loads for execution of control and demand side management respectively.

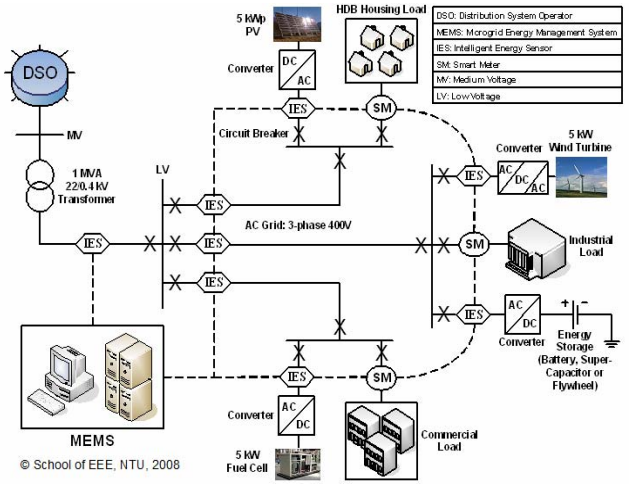

Fig. 1. Overview of NTU microgrid

However, in the following sections, we propose another control scheme for the controlling of DERs using JADE. This control scheme is entirely coded in Java programming language and is capable of either operating as a standalone entity or working together with third party softwares such as MATLAB and LabVIEW. The main reason for adopting JADE is because JADE is an Operating System (OS) independent platform which means that the coded algorithm can be implemented on any computer as long as the Java Virtual Machine (JVM) is installed. In the case of LabVIEW, programmers may have to re-code their control functions from MATLAB/SIMULINK into LabVIEW compatible program code which introduces compatibility issues. JADE removes the hassle of converting the coded algorithm across different softwares and makes it portable across various types of OS thus making it very convenient and flexible.

III. Multi-Agent System Framework

A. MAS Concept

Adopting Multi-Agent Systems (MAS) approach to solve optimization problem and control issues are evident in [9-11]. The main objective of MAS is to decompose and distribute large complex tasks into simpler and manageable ones. In the context of microgrids, data and information are decentralized implying that failure of any control component within the microgrid will not threaten the operation of the entire microgrid [12]. MAS distributed control also has inherent properties such as its scalability and openness which makes microgrid control very flexible.

In terms of implementation costs, it is still relatively cheaper to implement MAS distributed control because the amount of information exchanged is minimal. This means cheaper communications infrastructure and computer network resulting in lower implementation costs. Figure 2 shows the two different types of control and illustrates the main difference in control architecture between central and hybrid MAS distributed control.

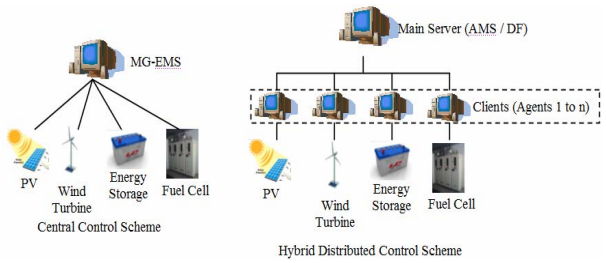

Fig. 2. Central and hybrid distributed control hierarchy

The central control scheme has direct control interface with the DERs and energy storage and all the computations are done centrally on the main server. This means that equipment in a central control hierarchy always wait for control instructions from the main server before performing certain functions. Usually, centralized control is being used when all the DERs are under the same owner or the power pool arrangement so that that some form of cooperative strategy is achieved.

In a hybrid distributed control, responsibilities and tasks are assigned to the clients which directly control the equipment while the main server provides basic essential communication services such as Agent Management System (AMS) and Directory Facilitator (DF) for the agents residing within clients to communicate and negotiate with each other [13]. Furthermore, the main server in a hybrid distributed control can also provide monitoring and ancillary services depending on the requirements and specifications of the microgrid. Sometimes, responsibilities assumed by the main server and clients may change depending on how the microgrid is managed [14]. Typically, a hybrid distributed control scheme is applied to the microgrid where all the connected DERs are owned by different owners. This is because DER owners may have different objectives therefore instead of cooperating with each other, a local competitive microgrid market is formed to manage these DERs having different objectives [15].

B. Autonomous Agents

It is important to understand and appreciate the underlying main principles of agent theory as they drive the main building block for MAS applications [16, 17]. MAS itself is composed of many agents interacting with one another. Each agent has its own allocated tasks and objectives. One important characteristic of such agents is the ability to have self autonomy by exhibiting intelligence. This means that it can make decisions for the assigned equipment without any intervention from a central controller. Considering a microgrid with many Distributed Energy Resources (DERs), it is advantageous to have this characteristic so that control operations will not be delayed if any DER fails to respond. Other important key attributes of agents include:

Social ability - Agents are able to communicate with one another to achieve their objectives. This is because agents have partial or no knowledge of the environment i.e. the microgrid in this context. Hence by communicating with other agents, it is able to continuously update itself with relevant information.

Reactivity - Agents are programmed such that they can respond to any changes in the environment without much delay. For example, if any DER agent within the microgrid is disconnected or offline, other agents will be notified and subsequently, make minor changes in their algorithms to ensure that the microgrid continues to operate smoothly.

Pro-activeness - Agents exhibit goal-directed behaviors in order to accomplish their objectives. This is because agents alone cannot achieve their objectives unless they take the initiative to interact with other agents. Therefore, instead of being passive, agents are considered active entities so that their goals can be achieved.

Reliability - Agents are not allowed to intentionally provide false or misleading information that can potentially corrupt the integrity of the information exchanged.

Mobility - Agents are able to migrate from one host platform to another without making much changes to the existing system. This is particularly useful when a client computer is scheduled for maintenance because the agent can migrate to another client without causing any downtime for the equipment it is currently controlling.

C. FIPA Specifications

In terms of inter-agent communications, the Foundation for Intelligent Physical Agent (FIPA) standard is being used [18, 19]. FIPA is an international non-profit association of companies and organizations producing specifications for agent technologies. This set of standards is formed to achieve a high level of interoperability in complex systems.

In FIPA specifications, three essential roles were identified to describe the reference model of an agent platform. These roles are necessary for managing the platform as well as describing the agent management content language and ontology. The first role requires AMS to be set up so that supervisory control over the agent platform can be done. It is also required to maintain a directory of residing agents and handle their life cycle. The Agent Communication Channel (ACC) is the next role where it is the default basic communication method offering reliable, orderly and accurate message routing services. The last role is requiring a Directory Facilitator (DF) agent to be present for providing yellow page services to agents within the platform.

Other FIPA specifications include Agent Communication Language (ACL). It is a language used by agents to exchange messages. Since agents can be created by different developers, the FIPA-ACL standard is used so that agents running on different platforms are able to understand and interpret messages. This will also prevent any ambiguity and confusion arising from agent communications since a standard set of message template is laid out following the FIPA-ACL specifications.

Some FIPA specifications not covered in this paper include agent-software integration, agent mobility, agent security, ontology service and human-agent communication. More details regarding these specifications can be found in [20].

IV. Proposed Methodology

A. Optimization Objectives

One of the main objectives of microgrid is to maintain power balances within the network while minimizing operating costs and maximizing consumer benefits. The power equation in (1) describes the economic dispatch problem subjected to constraints:

$$\min \sum_{i=1}^{n} C_i(P_{gen,i}) \tag{1}$$

$$\text{Subject to } \sum_{i=1}^{n} P_{gen,i} = \sum_{j=1}^{m} P_{load,j} + \sum P_{loss}$$

$$P_{gen,i}^{\min} \le P_{gen,i} \le P_{gen,i}^{\max}$$

where C_i is the cost function of the ith generator; $P_{gen,i}$ is the active power generated by the ith generator in kW; $P_{load,j}$ is the active power demand of the jth load in kW; and P_{loss} is the total active distribution power loss.

Assuming that the generators use fuel to produce electricity, the typical cost function for each generator under economic dispatch is given as:

$$C_i(P_{gen,i}) = aP_{gen,i}^{\ 2} + bP_{gen,i} + cP_{gen,i} \tag{2}$$

where a, b and c are constants which have pre-determined values. However, when determining the cost function for DGs such as PVs and wind turbines, the equation for their respective cost function will be different from (2). This is because renewable DGs have to incorporate intermittent supply of natural resource, return on investment costs, utilization factor and maintenance costs into the cost function which is outside the scope of this paper and will not be covered in this paper.

The above optimization problem can be solved by a few techniques such as sequential quadratic programming, genetic algorithm programming and LaGrangian technique. Computed optimization results

will be in the form of scheduling dispatch for the respective generators. This optimization problem is relatively easy to solve for a small system but becomes increasingly more challenging when the value of i, the number of DERs, is large. In order to control many DERs, JADE will provide another intuitive way of how the mircogrid operations can be optimized based on incremental costs ($/kWh). One major difference to note between economic dispatch and JADE is how incremental costs are used in solving the optimization problem. The principle behind economic dispatch basis is that all generations have the same incremental costs while JADE consolidates different incremental costs from DERs and tries to match supply with load demand pricing.

B. JADE Architecture

There are currently many research activities involved in agent development and a number of agent construction tools are used such as JADE, Zeus, Voyager, Excalibur Agent, FIPA-OS, AgentTool, AgentBuilder, Grasshopper-2 and RETSINA [21] to realize this implementation. Among these available toolkits, JADE was selected for this research because JADE complies with FIPA standards and provides a middleware where programmers can zoom straight into designing agents and their behaviors without having to fully understand how JADE actually runs in the background [22]. JADE is also a Java framework for agent development through the use of object oriented abstractions. Therefore, JADE can provide a convenient distributed platform for users to focus on developing agents for control and monitoring of power balance during microgrid operation.

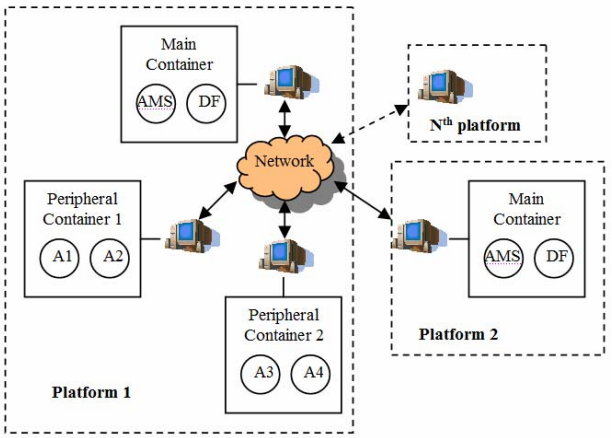

Fig. 3. JADE control architecture

JADE can be described as a distributed platform consisting of main containers and peripherals containers. In Figure 3, it can be seen that when a platform is created, the main container is always the first container to be initialized in JADE. AMS and DF agents are also automatically created once the main container is initialized. Once the main container is initialized, peripheral containers can be set up on other computers which contain other agents. Thus, a distributed platform is created. When another platform intends to join existing network, another main container is initialized on the new host together with the predefined agents (AMS and DF) and other customized agents from Table 1 (e.g. Generator Agent, Load Agent, Control Agent, etc). The implementation process illustrates that JADE is an open system which supports Plug and Play capabilities and is also scalable without much modification to the control scheme. This is particularly desirable in a microgrid because new DGs can be connected conveniently without having to reconfigure and modify existing control architecture.

C. JADE Operating Environment

There are two ways to execute, manage and monitor the activities of distributed agents. First method is by executing the commands from windows command prompt shell which only displays simulation results in text while the other method allows users to view JADE working environment graphically by calling the remote management agent (rma). The rma is a default Graphical User Interface (GUI) agent used by JADE to graphically display how agents are categorized. In Figure 4, there are other features within rma GUI such as displaying the DF catalog of registered agents in the network and the sniffer agent which displays the message flow between agents.

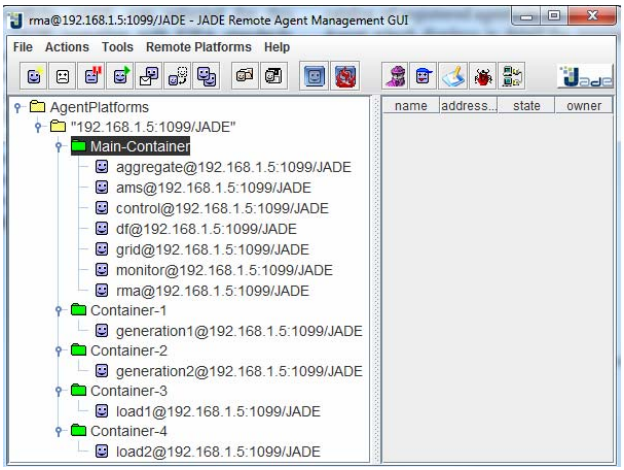

Fig. 4. Snapshot of JADE working environment

Another important aspect of JADE is agent communication. The ACL message template used by JADE is shown in Figure 5 to illustrate how agents communicate with one another. Among the parameters within the template, only certain basic parameters such as receivers, communicative act, content and conversation id are required in order for agents to perform basic interaction. Under the receivers section, programmers are required to state which agent should receive the message. To further specify which agent should receive the outgoing message, programmers are also required to fill up the conversation id section. These two parameters are needed so that only intended agents with specified conversation id and agent class are able to receive messages and other agents under the same agent class will not receive these messages.

The communicative act parameter specifies which performative the message belongs to so that the sent messages can be easily traced when the sniffer agent is

launched in JADE. Some of these performatives include Call for Proposal (CFP), inform, request, query and refuse. The content parameter is where all the message contents are inserted. When the message is deciphered by the recipient, the message content is the only source of information agents can acquire to update their knowledge database.

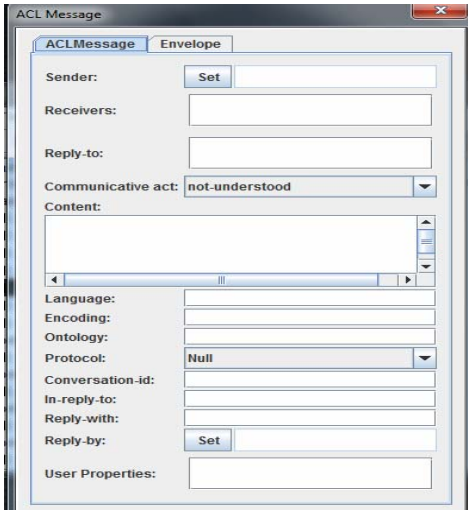

Fig. 5. ACL message template

V. Simulation Studies

A. Scenario

To test the functionality of MAS using JADE, a simple microgrid topology is being set up as shown in Figure 6.

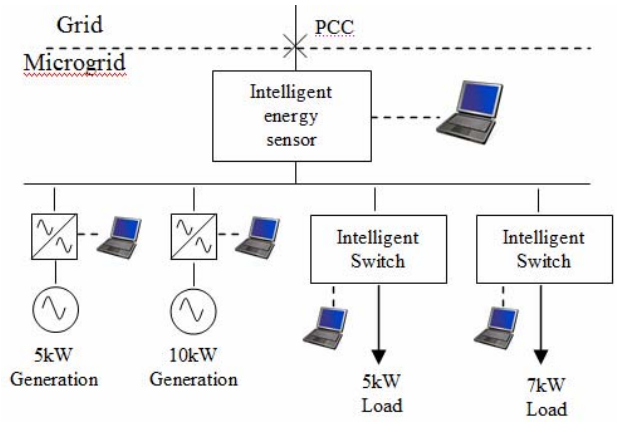

Fig. 6 Microgrid simulation testbed

In this microgrid, there are two generation units rated 5kW and 10kW and two load units rated 5kW and 7kW respectively. The generation units typically represent renewable technologies such as PVs and wind turbines and are connected to the main busbar through a converter which is controlled directly by a computer with the corresponding agents programmed. The load units are connected to the main busbar through an intelligent switch which is also connected to a computer containing the necessary agents. There is also an intelligent energy sensor at the Point of Common Coupling (PCC) where the main computer server is connected to it. All communications between computers are done through TCP/IP. JADE also configures the communication process such that all PCs are using the same port and host. However in this paper, all the agents are located in the same computer for simulation because of simplicity and also to illustrate the functionality of MAS.

In this simulation study, six different customized agents were developed following FIPA-ACL specifications. The names and objectives of each agent are summarized in table 1. The location of where the agents reside in should also be noted. For GA and LA agents, these agents should be located directly to the units and loads where they are supposed to control. For example, GA agent should be attached directly to generation source and LA agent should directly control the loads. The remaining agents AA, MA, CA and GrA can be located in any of the available computers but usually they are consolidated and reside in the main server computer.

TABLE I
List Of Agents And Objectives

Agent Names	Objectives
Generation Agent (GA)	Controls microsource power setting and allow owners to set selling price for trading
Load Agent (LA)	Allows customer to specify the amount of power to purchase and performs trading on behalf of customers
Monitor Agent (MA)	Request updated information on the status of GA and LA and display them on GUI
Aggregator Agent (AA)	Request GA and LA for data and consolidate total supply and total demand
Control Agent (CA)	Responsible for performing power exchange between the microgrid and main grid
Grid Agent (GrA)	Collects real time Grid pricing and informs other agents about connection status of Grid

The microgrid operations for supplying and buying power is being planned at every interval. The duration of the interval is selected at every hour in this study. During each interval, the agents in Table 1 will begin executing their designed objectives. The trading and negotiation process will continue until all generation and load units are matched. Any net surplus or deficit in power within the microgrid will be managed by CA. At the end of each time interval, the simulation results are displayed in the command prompt window.

B. Agent Description and GUI

Besides programming agent intelligence, every agent developed by the programmer also has a GUI agent attached to it. The role of the GUI agent is to graphically display important status of the agent it is attached to and also allows users to perform different functions via the GUI panel. In the remaining part of this section, each of the GUI agents will be displayed and a description of the agents internal process is being discussed to provide an idea of how agents operate together with its GUI.

A snapshot of the GA agent GUI is being shown in Figure 7. In this GUI, there are two parameters that can be changed by the user while the other parameters are to display updated information. GA users can specify how much energy they want to supply to the microgrid power market while stating its selling price. Once all information is inserted, the user can click on the refresh

button to refresh the status of the agent which displays the agent identifier, power available for sale, selling price, grid buying and selling price and the connection status between the microgrid and the main grid.

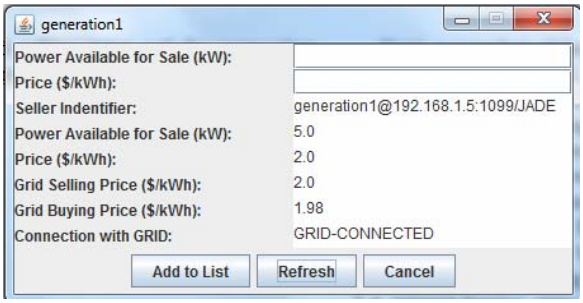

Fig. 7 Snapshot of Generation Agent GUI

Next, the main internal thread processes of the GA agent are described to give a glimpse of how the agent operates. The initialization for all agents begins with registering itself with the Directory Facilitor (DF) by providing it with a set of service descriptions such as setName and setType. For GA, setName is selected as generation and setType is chosen to be power selling.

Once GA is registered in DF, other agents are able to see it in the yellow pages. GA will then execute a set of custom behaviours defined by the programmer. These set of agent behaviours are classified as cyclic because they are constantly waiting for new incoming messages from other agents. For GA, the set of behaviours defined are offer request and purchase order, update aggregate and monitor agent, receive grid price and status. The offer request and purchase order behaviours are to wait for proposal messages from LA and inform successful LA agents about their offer. The update aggregate and monitor agent behaviours are reactive because once GA has processed an incoming message it will automatically update the respective agents on its status. The receiving grid price and status behaviour is to wait for GrA to inform GA on any changes regarding current grid prices and connection status so that GA users can state their selling prices competitively.

Whenever any GUI is closed by the user, the agent automatically deregisters itself from DF and terminates itself from JADE.

A snapshot of the Load Agent (LA) GUI is being shown in Figure 8. The LA agent is typically treated as a buyer because it searches the DF yellow pages for generation agents offering power supply services.

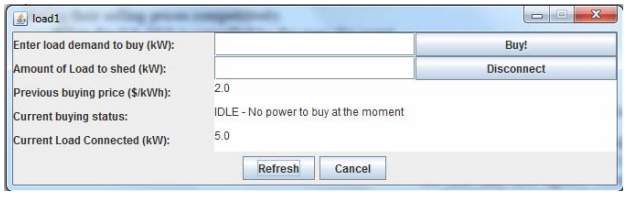

Fig. 8 Snapshot of Load Agent GUI

In this GUI, users are able to key in the amount of the power they want to buy from the power market or upstream network and specify any amount of load they want to disconnect. The previous buying price is also shown which state the price that was finalized during the previous round of trading. The buying status refers to the current buying status which can tell users whether LA is still trading or has already completed its trading. The current load connected parameter displays the current loading of that particular LA with respect to the microgrid.

LA first registers itself with DF by providing load and power buyer as input parameters to setName and setType respectively. The next step is to execute a series of different behaviours. Whenever LA receives an order from GUI to buy power from the market or upstream network, it executes a requesting trade behaviour which calls for proposals from all GAs and submit a final confirmation to the GA which offered the best price. If there are no GA in the DF yellow pages, LA will execute its requesting trade behaviour every 20 seconds until its objectives are met or the interval ends with CA buying power from the grid to meet its load demand. Once power is bought, LA will automatically execute the update aggregate and monitor agent behaviour to inform the respective agents on its current status.

The Monitor Agent (MA) GUI is being shown in Figure 9. This is a simple GUI displaying the status of all GA and LA agents which are currently actively trading in the microgrid power market or upstream network.

Fig. 9 Snapshot of Monitor Agent GUI

The first two columns in the GUI show the GA identifier and status respectively while the last two columns are reserved for LA identifier and status. The GA status column reflects the current selling price and available generation and the LA status column shows the amount of load connected currently.

MA will begin initialization by providing monitor and microgrid monitoring services as inputs to setName and setType for service description to DF yellow pages. The next step is to execute behaviours for MA to function. The first behaviour is receiving GA status which only receives messages sent by the GA agent. It waits for any messages sent by GA and stores the information which is then displayed by the GUI. The receiving loading status is another behaviour which waits for messages from the LA agent informing MA to update its database whenever there are any load changes.

In Figure 10, a snapshot of the Aggregate Agent (AA) GUI is being shown. In this GUI, it displays four important information regarding the power flow within the microgrid. The AA GUI keeps track of the total net power within the microgrid which can be seen under the total untraded power parameter. The agent also aggregates the total load demand within the microgrid and is shown under the total load demand parameter. The current grid buying and selling prices are also reflected in

the GUI to update users on current grid pricing.

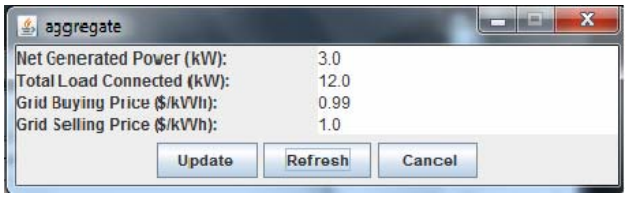

Fig. 10 Snapshot of Aggregate Agent GUI

The initialization input parameter for AA are aggregation and microgrid aggregation service for setName and setType respectively. The next step in the agent's lifecycle is to execute its behaviour. Upon invoking the update status on AA GUI, the agent will request for available generation from GA and loading status from LA respectively. Receive GA status and receive LA status behaviours will then wait for replies from GA and LA. Once all information are received, AA will compute and display available generation and loading status in the GUI.

A snapshot of the Grid Agent (GrA) GUI is shown in Figure 11. This GUI allows users to change grid selling price while the grid buying price is automatically computed by the agent. Users are also allowed to toggle the connection status of the grid to study the impacts of islanding on agents. After all values are configured, the GUI will display the grid pricing and connection status to inform users that the agent is in operation.

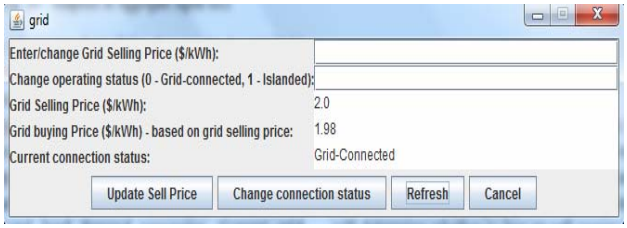

Fig. 11 Snapshot of Grid Agent GUI

The initialization phase begins with GrA registering Grid Agent and grid status as inputs to setName and setType respectively. The main set of behaviour for GrA is receiving request from CA. This behaviour will wait for messages from CA to determine whether the microgrid needs to buy energy or sell net surplus of energy to the grid during grid connected mode. In islanding mode, GrA is unable to process messages from CA thus CA will have to implement other strategies to manage the microgrid. Other behaviours such as broadcasting grid prices and grid connection status are invoked through this GUI.

The Control Agent (CA) GUI is shown in Figure 12. In this GUI, it displays the total microgrid power generation and loading as well as computing the net microgrid power. Once the net microgrid power is computed, the agent will determine whether to buy or sell power to the grid as shown in the GUI. The grid connection status is also displayed in the GUI to verify mode of operation.

The initialization step is to provide control and microgrid control management as inputs to setName and setType.

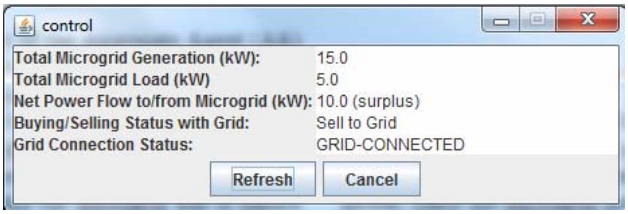

Fig. 12 Snapshot of Control Agent GUI

There are two main behaviours for CA. The first is receiving information from aggregate agent behaviour. This behaviour waits for AA to send periodic updated information so that CA can process this information and make a trading decision with GrA. The second behaviour is initiating power exchange with GrA every hour. Once CA has decided whether to buy or sell power to grid, this behaviour is executed to perform trading with grid. After settling with GrA, CA will announce to all GA and LA to reset their power settings to zero in preparation for the next trading interval.

C. Simulation Results

One round of trading interval was simulated and the results are shown in Figure 13 and Figure 14. In Figure 13, it shows the sniffer agent (default agent in JADE) tracking all message exchanges between the agents. Each arrow in the diagram represents a message sent. Users can check where and who the message originated from by tracing the number of incoming and outgoing arrows for that particular agent.

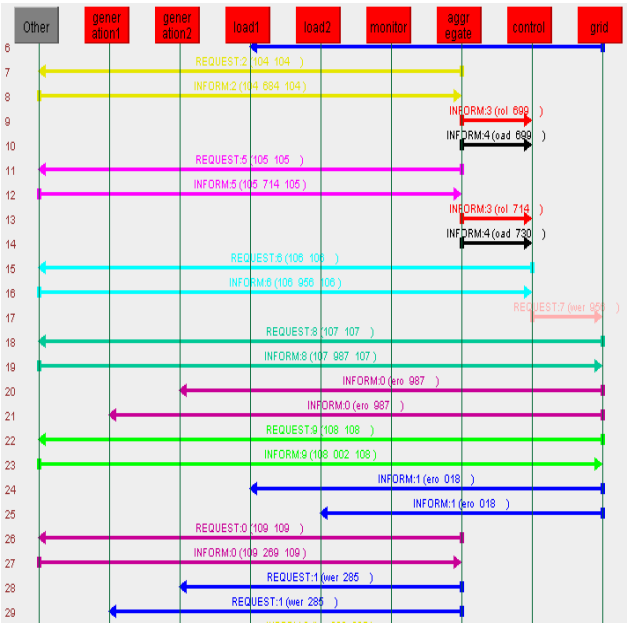

Fig. 13 Snapshot of Sniffer Agent and agent message exchange

The command prompt window in Figure 14 shows some of the major activities the agents perform during the trading interval. After trading and negotiation are completed, the final result is being reported by CA which tells users how much power is being traded with the grid and GrA announcing the end of the current trading interval.

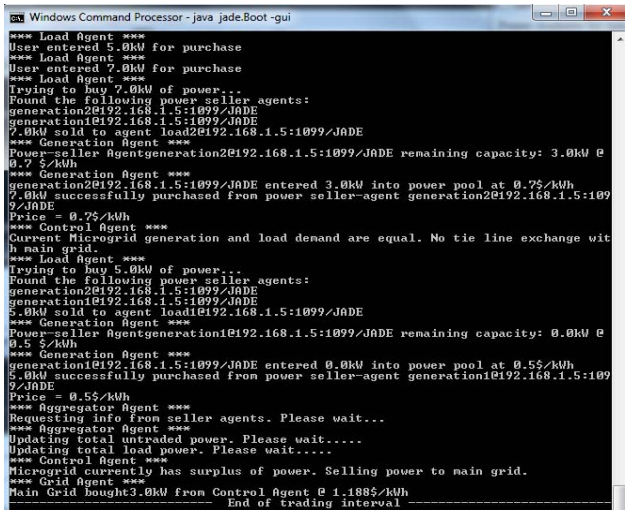

Fig. 14 Output window from command prompt

VI. Conclusions

This paper has presented an overview of the NTU microgrid architecture and has shown how the MAS based control scheme can benefit the microgrid. In this simulation study, we have successfully developed customizable agents which can interact and make informed decisions based on the information received through message exchanges. By adopting similar agent design approach, additional agents can be introduced which makes this form of decentralized control modular.

The future research direction will be to integrate JADE with MATLAB/SIMULINK for software verification before performing hardware verification on the NTU microgrid. We will also be looking to implement the embedded algorithm into the smart meters in NTU microgrid. As such, more effort is required to successfully interface the software with the hardware setup.

Acknowledgment

The authors would like to acknowledge the generous fund support provided by A*STAR under the Microgrid Energy Management System Project (Grant no. 072 133 0038).

References

[1] L. Xuan and S. Bin, "Microgrids - an integration of renewable energy technologies," in *Electricity Distribution, 2008. CICED 2008. China International Conference on*, 2008, pp. 1-7.

[2] (2009). *Distributed generation and Microgrid concept. Chapter 1*. Available: www.knovel.com

[3] J. Zeng, *et al.*, "An agent-based approach to renewable energy management in eco-building," in *Sustainable Energy Technologies, 2008. ICSET 2008. IEEE International Conference on*, 2008, pp. 46-50.

[4] J. Zhang, *et al.*, "The application of multi agent system in microgrid coordination control," in *Sustainable Power Generation and Supply, 2009. SUPERGEN '09. International Conference on*, 2009, pp. 1-6.

[5] S. Suryanarayanan, *et al.*, "A conceptual framework of a hierarchically networked agent-based microgrid architecture," in *Transmission and Distribution Conference and Exposition, 2010 IEEE PES*, 2010, pp. 1-5.

[6] C. M. Colson and M. H. Nehrir, "A review of challenges to real-time power management of microgrids," in *Power & Energy Society General Meeting, 2009. PES '09. IEEE*, 2009, pp. 1-8.

[7] W. Caisheng, *et al.*, "From hybrid energy systems to microgrids: Hybridization techniques, configuration, and control," in *Power and Energy Society General Meeting, 2010 IEEE*, 2010, pp. 1-4.

[8] V. Q. N. Cheah Peng Huat; Siow Lip Kian; Liang Hong Zhu, Nyugen Dinh Duc, Gooi Hoay Beng. *Microgrid Energy Management System (MEMS)*. Available: http://digital.ni.com/worldwide/singapore.nsf/web/all/7462C54674636753862577C000366EEF

[9] T. Logenthiran, *et al.*, "Multi-agent system for energy resource scheduling of integrated microgrids in a distributed system," *Electric Power Systems Research,* vol. 81, pp. 138-148, 2011.

[10] Z. Xiao, *et al.* (2010, Hierarchical MAS Based Control Strategy for Microgrid. *2,* 1622-1638.

[11] A. L. Dimeas and N. D. Hatziargyriou, "Operation of a Multiagent System for Microgrid Control," *Power Systems, IEEE Transactions on,* vol. 20, pp. 1447-1455, 2005.

[12] L. M. Tolbert, *et al.*, "Scalable multi-agent system for real-time electric power management," in *Power Engineering Society Summer Meeting, 2001. IEEE*, 2001, pp. 1676-1679 vol.3.

[13] *Java Agent Development Framework (JADE)*. Available: http://jade.tilab.com/

[14] A. L. Dimeas and N. D. Hatziargyriou, "A MAS architecture for microgrids control," in *Intelligent Systems Application to Power Systems, 2005. Proceedings of the 13th International Conference on*, 2005, p. 5 pp.

[15] E. Kaegi, *et al.*, "Decentralized Unit Commitment and Dispatch for the Distribution Systems using Intelligent Agents Approach," presented at the 16th PSCC, Glasgow, Scotland, 2008.

[16] D. Chun-Xia, *et al.*, "Multi-Agent Based Control Framework for Microgrids," in *Power and Energy Engineering Conference, 2009. APPEEC 2009. Asia-Pacific*, 2009, pp. 1-4.

[17] S. D. J. McArthur, *et al.*, "Multi-Agent Systems for Power Engineering Applications Part I: Concepts, Approaches, and Technical Challenges," *Power Systems, IEEE Transactions on,* vol. 22, pp. 1743-1752, 2007.

[18] S. D. J. McArthur, *et al.*, "Multi-Agent Systems for Power Engineering Applications Part II: Technologies, Standards, and Tools for Building Multi-agent Systems," *Power Systems, IEEE Transactions on,* vol. 22, pp. 1753-1759, 2007.

[19] S. D. J. McArthur, *et al.*, "Building multi-agent systems for power engineering applications," in *Power Engineering Society General Meeting, 2006. IEEE*, 2006, p. 7 pp.

[20] F. Bellifemine, *et al.*, "Developing Multi-agent Systems with JADE," ed. Italy, 2001.

[21] C. Mangina, "Review of Software Products for Multi-Agent Systems," A. I. U. Ltd, Ed., ed, 2002.

[22] X. Qun, *et al.*, "The control of Distributed Generation System using Multi-Agent System," in *Electronics and Information Engineering (ICEIE), 2010 International Conference On*, 2010, pp. V1-30-V1-33.