Validation of Agent Based Reconfiguration Scheme Using Modeling and Simulation Approach

K. Huang, S. K. Srivastava, & D. A. Cartes
Center for Advanced Power Systems
Florida State University
(huang@caps.fsu.edu, sanjeev@caps.fsu.edu, dave@caps.fsu.edu)

Keywords: Multi agent system, reconfiguration, modeling, simulation, distributed intelligence

Abstract

A popular distributed intelligence/control approach for research is the agent based application. Agent based solutions are being applied to diverse problems including reconfiguration of power systems. Although developing an efficient approach is important, also important is the validation of that approach and its realtime or near realtime performance. This paper discusses the importance of modeling and simulation towards verification of an agent based reconfiguration scheme for power systems. Results of two different modeling and simulation approaches are presented. In the first approach Virtual Test Bed software was used for validation of multi agent based reconfiguration algorithm. In the second case the same agents' performance was analyzed with processor-in-the-loop experiments with a real time digital simulator.

1. INTRODUCTION

Modeling and simulation is used for the analysis of a system to get a better understanding its expected outcomes and deficiencies, with respect to a specific integration and the systems aggregated performance. Models are created to understand, monitor, and control the behavior of a complex system. For a power system, the capability to reconfigure is important to its survivability and reliability. In recent years, distributed intelligence is applied to the reconfiguration of the power systems in order to avoid single point of failure, provide more robustness, scalability and flexibility to the power systems. Agent based reconfiguration is one of the most popular distributed intelligent application to a power system. In recent years, the multi agent system (MAS) technique is increasingly applied to solve such reconfiguration problems [1-5]. Not all MAS are the same. Some are truly decentralized and will be discussed later. On the other hand, some require a form of global dependency, centralized supervision, and thus are susceptible to single points of failure. In [1], a facilitator agent was required for restoration of the power system. The facilitator agent has the global information of the system, which makes the method centralized. In [2], the authors improved their method in [1]. However, in [2], facilitator agents with global information are still required. In [3], the coordinating agents with global information are required for the reconfiguration of the system. In [4], a truly decentralized MAS based reconfiguration method was proposed for radial shipboard power systems (SPS). However, the method put forward in [4] is only applied to a radial system reconfiguration. In [5] an improved decentralized method for avoiding information accumulation in an agent system for reconfiguration of a mesh structured power system is proposed.

In order to validate the reconfiguration algorithm and test MAS performance these approaches need to be tested and validated in a framework that captures their impact on the total integrated system. Modeling and simulation provides a low cost and low risk way of doing this. In this paper, an agent based system for power system reconfiguration is presented. Two modeling and simulation methods are discussed in order to test and validate the two different reconfiguration methodologies. This paper is organized as follows: in the next section, an agent based power system reconfiguration approach is presented. Then, two modeling and simulation approaches are proposed to test and validate the reconfiguration systems. Thereafter, the modeling and simulation results are discussed. Finally, some conclusions are given.

2. RECONFIGURATION OF A POWER SYSTEMS USING AGENTS

In this section, aspects of developed agent based reconfiguration methods that are common in both modeling and simulation approaches are presented.

2.1. Multi Agent System Structure

Figure 1 shows a MAS for power system reconfiguration. In Figure 1, each agent in the agent layer can send/receive information to/from one major electric component in the power system layer, such as generators, breakers, and loads. If two electric components have connection with each other, the corresponding agents are defined as neighboring agents. Hence, the topology of the MAS is similar to the topology of the power system. Agents exchange information with one another through the peer-to-peer connections in the upper

layer MAS; while power flows through the connection between electric components in the SPS. The agents in the MAS are restricted to communicate only with their neighboring agents. This constraint makes MAS work in a completely decentralized manner. No central agent is setup in the MAS and no agent in the MAS can get information from beyond its neighboring agents. Each agent makes control decision autonomously based on the information it receives from its neighboring agents and the corresponding electric component.

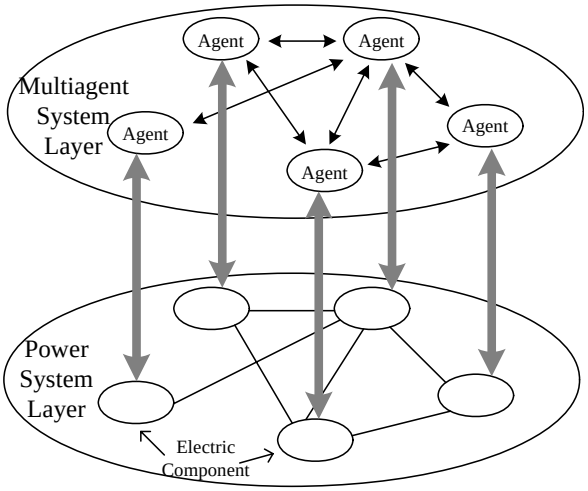


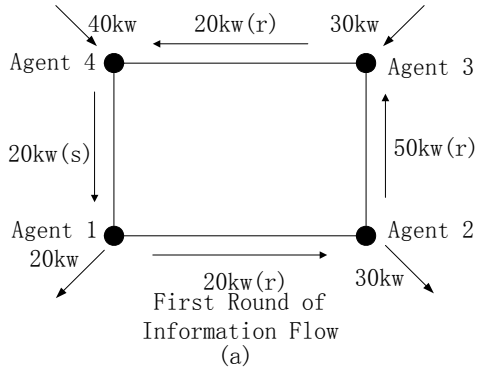
Figure 1. Interaction between agents and power system components

2.2. Redundant Information Accumulation

Since the agents are restricted to communicate only with their neighbors, the redundant information accumulation (RIA) problem may happen in the MAS when there is a loop in the system [5]. The RIA problem is like the positive feedback loop in the information flow among agents and makes the information flow unstable.

The reconfiguration of the power system is controlled by the agents in the MAS. The agents make decision based on the information flow in the agent system. In an MAS applied to a ring bus power system, the RIA problem may happen and lead to incorrect information flow in the system [5]. Consider a simple ring structured power system with four generators and four loads. In this system, the generators are connected in a ring structure, and each load is connected to one of the generators. The corresponding generator agents (Agent1, Agent2, Agent3, and Agent4) of the MAS are shown in Figure 2. In the Figure 2 the load agents have been omitted in order to simplify the information flow. Assume that Agent1 has 20kw power supply and the load agent that connects to Agent1 has 40kw power request. Then the power summation of these two agents is 20kw request (shown by the arrow pointing outwards). This summation is termed as effective capacity agnet1. Similarly the effective capacities of Agent2, Agent3 and Agent4 are 30kw request, 30kw supply, and 40kw supply respectively.

In a system as shown in Figure 2, the information flow may feedback to an agent where it originated. In this paper, power supply and power request are simplified as s and r respectively. In Figure 2(a), assume the information flow starts from agent 1. On the first round of information flow, agent 1 sends a message of 20kw request, i.e. 20kw(r), to agent 2. Agent 2 gets the message of 20kw(r), and adds it with the 30kw(r) capacity of itself. Then it sends a message of 50kw(r) to agent 3, and so on. On the second round of information flow, after agent 1 gets the message of 20kw(s) from agent 4, it will also add it with the 20kw(r) capacity of itself, and the result is 0kw(r). Then it will send a request of 0kw(r) to agent 2 as shown in Figure 2(b). An additional 20kw(s) power has been added into the request as compared with the request in the previous round of information flow shown in Figure 2(a). This 20kw(s) will be added to each message in this round, and will accumulate in each round of information flow. This redundant information accumulation has a similar effect to that of a positive feedback loop, leading to an unstable information flow.



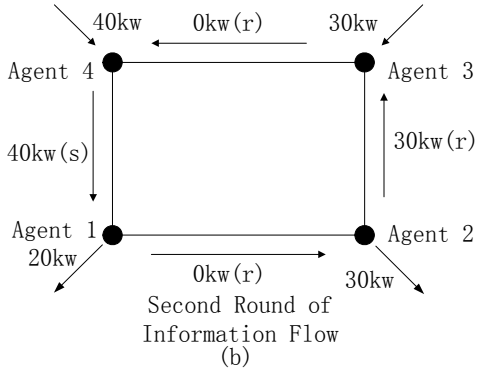


Figure 2. Redundant information accumulation

Since the RIA problem is due to the loop structures in the MAS, the solution to RIA problem is to break the loop structures in the MAS by disconnection the communication between certain agents. For example, if the communication between Agent2 and Agent3 is disconnected, the RIA problem can be avoided in the system. However, in a distributed agent system, no agent has the global view. No agent has preset information about the existence of the loops in the system and where to break the loop in the system by disconnecting the communication between agents. The agents need to exchange information with each other in order to find the break point and avoid the RIA problem in the agent system. Hence an algorithm for avoiding the RIA problem is developed that detects a ring of mesh structure in the system and then

accordingly generates a tree structure. More details of this method can be found in [5].

2.3. Reconfiguration Methodology

Once the RIA problem is avoided the next step is to initiate the reconfiguration algorithm. The agents in the MAS sense or monitor the environment (power system parameters) and then based on events happening in the power system they will make decisions to take appropriate reconfiguration action. At this stage it is important to point out the importance of modeling approach used for modeling the agents. It is the modeling approach that place restrictions on how a reconfiguration algorithm will be implemented. The modeling approach will also limit the scope of actions for an agent. An agent can be modeled with a very broad scope of actions but then this approach can have a negative effect on the performance of MAS and complexity of reconfiguration problem. Conversely a simplified agent modeling approach can result in situations in which agents are not equipped with functionalities to handle a complex scenario. Hence an optimized approach for modeling of agents is required such that all or most of the (reconfiguration) possibilities are covered. The modeling approach will also define the inputs and outputs of an agent. This requirement will stem from the required reconfiguration objectives. Based on these requirements a reconfiguration approach is then designed.

Authors have developed two different reconfigurations approaches that are discussed in next sections.

3. AGENT BASED MODELLING AND SIMULATION USING VIRTUAL TEST BED

In this section, a Virtual Test Bed (VTB) based modeling and simulation is introduced to test and validate the agent based reconfiguration approach proposed in the previous section.

3.1. Simulation in Virtual Test Bed (VTB)

Virtual Test Bed is a simulation package developed by University of South Carolina. The VTB is software for prototyping of large-scale, multi-technical dynamic systems [6]. It allows proof-testing of new designs prior to hardware construction. the Virtual Test Bed provides capabilities to import dynamic models from a variety of environments (while enforcing data, signal, or natural coupling laws), to connect finite element models with lumped-element dynamic models, to import structural models that describe the physical properties of the system, and to create and drive advanced visualizations from within the simulation environment. In VTB, components such as power source, breaker, motor, are interconnected to form a schematic, which is a visual description of the plant. To run an outside controller in parallel with the simulation, a.k.a. processors in the loop (PIL), the primary step is to establish communication channels between the simulation software and the remote processors (in our case, software agents) controlling the simulated plant. In other words the requirement is to implement a mechanism which injects and receives signals to and from the simulation.

3.2. Agent Modeling

As stated in section 2.1, agents are modeled to represent (or interface with) a major component of a power system. This modeling approach/rule is easy to follow and clearly defines what kind of agents can exist in the MAS. If an agent has a connection with another agent, the connection is defined as an interface of the agent. An agent may have more than one interface, since it can connect to more than one agent. The agent in Figure 3 has four interfaces. Each interface has different variables such as contract, output cost_load curve, input cost_load curve, nvload_in, and nvload_out, as shown in Figure 3 [4]. These variables are direct outcome of selected modeling approach and reconfiguration algorithm requirements. In our later discussions, the power that flows through an interface refers to the power that flows through the connection in the power system layer that the interface represents.

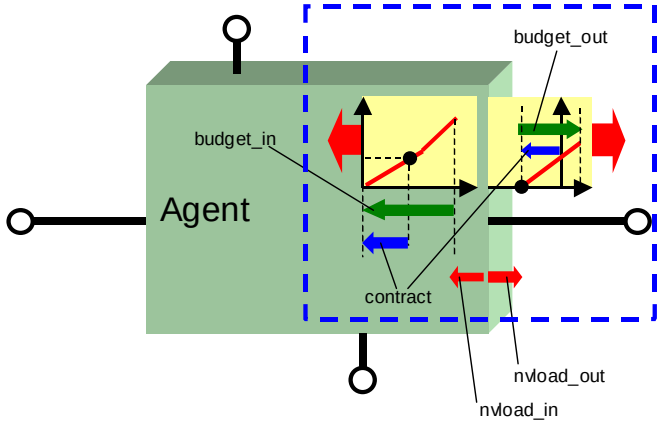


Figure 3. Agent model with four interfaces

The contract is the exact power that flows through this interface. The input/output cost_load curve provides the budget_in/budget_out and per unit power cost of the interface. The budget_in provides the available power that could be received through the interface, while budget _out indicates the maximum power that could be transferred to other agents through this interface. The nvload_in indicates the maximum power that could be received via this interface if all the non vital loads in the system are shed, while nvload_out indicates the maximum power that could be transferred to the other agents through this interface if all the non vital loads in the system are shed. Agents also have behaviors, such as UpdateBehavior, CostBehavior, and RequestBehavior. UpdateBehavior resides in all agents and generates the output cost_load curve and nvload_out for each interface [4].

The agents were implemented on iPAQs. The iPAQ is a Pocket PC (PPC) developed by Hewlett Packard [7]. The iPAQ has integrated WiFi, which provides wireless connections for iPAQ. The communication protocol for WiFi is IEEE 802.11b, which supports wireless communication up to 11Mbit/s. A part of agent code is written in Java using JacORB [8], which is a free implementation of the OMG's Common Object Request Broker Architecture (CORBA) standard [9]. LEAP [10] library combined with JADE [11] was used as the agent development framework to develop the remaining part of agent code. A component in VTB, written using Microsoft Visual C++, is dynamically linked by VTB when that component is inserted into a simulation.

3.3. Reconfiguration Approach

The agents exchange the variables and implement behaviors, mentioned above, to implement the market reasoning based reconfiguration algorithm [4].

For an agent that represents a generator, it has only one interface because a generator is generally connected to a circuit breaker. The budget_out of the interface is the capacity of the generator; nvload_out is 0; and the output cost curve is a curve with slope equal to the per unit power cost of the generator.

For an agent that represents a load, it also has only one interface because a load is generally connected to a cable, which is generally connected to a breaker. The budget_out of the interface is 0. The nvload_out is 0 if it is a vital load, otherwise it is the power currently consumed by this load. Because a load cannot provide output power, the cost for power output is infinite. The corresponding output cost curve is a vertical line of infinite slope.

When an agent receives budget_out and nvload_out from its neighboring agent through an interface, it makes the budget_in and nvload_in of the interface equal to the received budget_out and nvload_out respectively. When an agent receives contract from its neighboring agent through an interface, it makes the contract of the interface equal to the opposite of the received contract.

An agent that neither represents a generator nor a load generates budget_out and nvload_out of interface i by combining all the budget_in and nvload_in from all the other interfaces except interface i on the agent.

Defining x^+ as $\max(x,0)$, and x^- as $\min(x,0)$, the market variable *budget_out* of interface i is computed using (1).

$$budget\_out_i = \sum_{j \neq i} (budget\_in_j - contract_j^+)^+ - contract_i^- \quad (1)$$

The nvload_out variable is computed using (2):

$$nvload\_out_i = \sum_{j \neq i} nvload\_in_j \quad (2)$$

3.4. CORBA Based Simulation Interface

In order to interface agents with simulation in VTB a general model to provide signal I/O to a client program written in various languages is needed. One example is a model that exports/imports signals to/from a Java application which can be used to integrate the agent system with the power system simulation in VTB. With this goal in mind authors have developed a CORBA based object in VTB, which allows for two way interaction with a remote program [12]. This can either be an agent or a program simply relaying information from a device designed to mimic the behavior of a component of the power system.

The Object Request Broker (ORB) is a distributed service that implements the request to the remote object. It locates the remote object on the network, communicates the request to the object, and waits for the results. When the object is available, the ORB communicates those results back to the client. In order for an agent to control a component in a VTB simulation, it has to first establish a connection with that component. The details of connection process are given in [12]. Once all the agents have established a connection with appropriate VTB components, the agents will start the SPS reconfiguration process.

Figure 4 presents the developed interface architecture between VTB simulation and MAS. The CORBA interface consists of several objects. Each object is coupled to a component in VTB. The agents in MAS establish connection with a specific component in VTB via appropriate CORBA objects. Through these objects, agents can send and receive signals from the components in VTB simulation. An agent connects to a CORBA object via wireless communication (shown by bold arrows in Figure 4). The dotted arrows in Figure 4 show the inter agent communication.

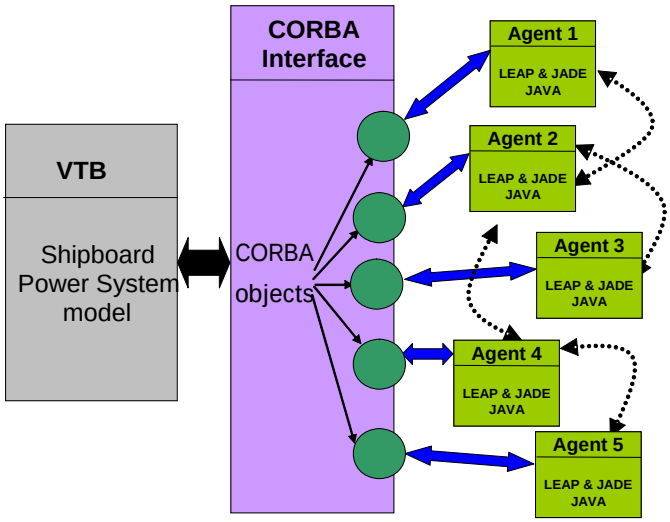


Figure 4. CORBA interface between VTB and MAS

3.5. Simulation Results for Power Systems in VTB

In order to test and validate the developed algorithm, a power system was simulated in VTB. Figure 5 shows a test power system simulated in VTB. It consists of 4 generators, 8 loads (6 PQ loads, 2 induction motors), 15 CBs, and 4 buses. All these components are controlled by individual agents. Therefore, the MAS consists of 31 agents. Of these 31 agents, 6 agents reside on HP iPAQs and the remaining agents were on a single computer.

A test case is presented to demonstrate the communication mechanism between MAS and VTB simulation. Initially in the power system in VTB, all the breakers except breaker0, breaker1, and breaker3 are closed and both motors are stopped and voltage levels at bus0 and bus1 are 0. Assume that all the loads except motor0 are vital, and the operation of the two motors requires the same amount of power. Also, assume that the combined output of all the sources, except Source0, which is out of service for maintenance, can support all the vital loads, i.e., all the PQ loads and motor1. But when Source0 is also turned on then all the loads can be satisfied.

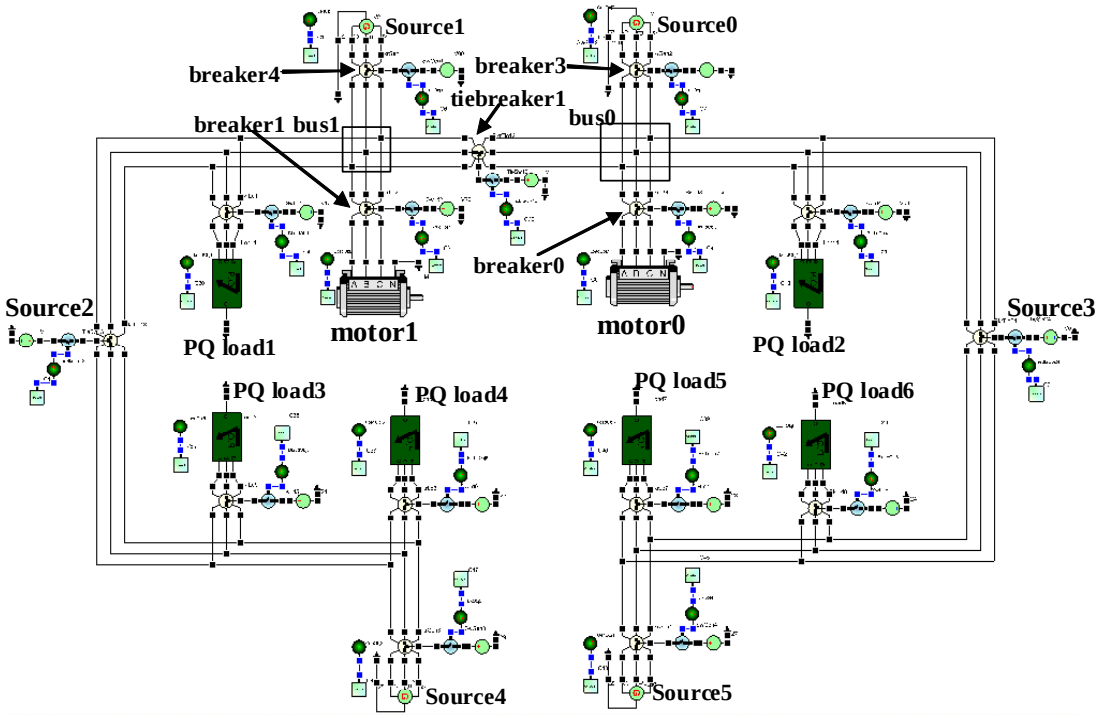


Figure 5. Power system simulation in VTB

In this test case, agent controlling motor0 marks motor0 as a non vital load, and tries to turn on motor0. For simplicity, an agent controlling a component X in VTB is referred as X_agent. The motor0_agent then tries to reserve resource in the agent system. A reservation message is sent from motor0_agent → breaker0_agent → bus0_agent → tiebreaker1_agent → bus1_agent → breaker4_agent → Source1_agent. The Source1_agent grants this request and the inform message is sent to motor0_agent via the same path. On receiving the inform message, the breaker0_agent looks up its associated CORBA object in VTB from the CORBA Naming service. When it is found, the object can be referenced as an internal object inside the agent. The object then sets the breaker0 in VTB simulation to the closed position. Now a path is established and the motor0 begins spinning.

Sometime after, motor1_agent is used to turn on motor1. Similar resource reservation message is sent up to bus_agent1. On receiving this request, the bus1_agent notices the sources have no extra capacity, but the branch on breaker0 has a non vital load. Thus a reallocation message is sent following the path bus1_agent → tiebreaker1_agent → bus0_agent → breaker0_agent → motor0_agent, to shed the non vital load. On receiving the reallocation message, breaker0_agent calls the remote CORBA object to set the VTB breaker position to open. This causes the breaker0 to open, and motor0 voltage to become 0 volts. Motor0 stops as the result of the reallocation message. Once the load on the breaker0 branch is shed, the bus1_agent grants the request from motor1_agent and reply the inform message following the path - bus1_agent → breaker1_agent → motor1_agent.

On receiving the inform message, the breaker1_agent calls its corresponding remote CORBA object to close the breaker1 in the VTB simulation. Then motor1 voltage rises and motor1 starts spinning. The motor speeds of the processes are recorded by VTB and are shown in Figure 6. From this figure it can be seen that at time 321 sec. speed of Motor0 (solid line) starts to decrease and as soon as this motor stops completely, Motor1 gets turned on its speed (dotted line) starts to increase.

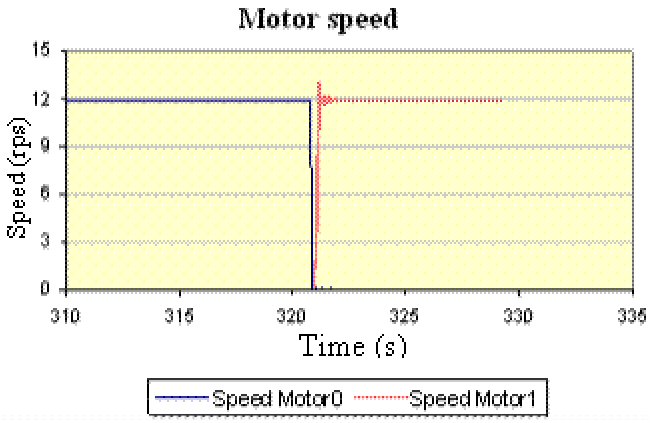


Figure 6. Speed of motors during load shedding event

4. AGENT BASED MODELLING AND SIMULATION USING RTDS

In this section, a real time digital simulator (RTDS) based simulation platform is proposed to test and validate the reconfiguration approach proposed in section 2. The real time nature will allow the validation not only of the reconfiguration algorithm. It will allow the observation the impact of communication bandwidth and latency and distributed processor system integration on the MAS performance.

4.1. Simulation in RTDS

The VTB cannot provide real time simulation for the power system, so the VTB simulation is only used for validating the reconfiguration algorithm. A real time simulation for the reconfiguration of the power system is setup on a real time digital simulator (RTDS) [13]. The RTDS is a high speed, real time test system that can be used for control system testing and general power system simulation. The RTDS simulator uses the parallel processing hardware technology to achieve its performance. The RTDS simulator employs an advanced and easy to use graphical user interface – the RSCAD software [13]. This software is the user's main interface with the RTDS hardware. The RSCAD is composed of several modules designed to allow the users to perform all of the necessary steps to prepare and run a simulation and to analyze simulation output. With RSCAD, it is very easy to setup a power system model with the graphical module in the libraries of RSCAD.

4.2. Agent Modeling

In this case also agents are modeled such that they represent (or interacted with) major power system components. But, unlike market reasoning based variables, used in previous agent based algorithm, these agents exchange variable like – agent id, net power, root agent id, etc. One major digression from the previous agent modeling approach is the variable net power. This variable is actually a global variable (unlike other variable, which are local) and its determination and calculation is a complex process [5]. This variable is also an outcome of agent modeling approach and reconfiguration requirement of avoiding RIA problem.

The agents in this simulation platform are also implemented similar to section 3.2. However, since the no CORBA based interface was used, so no JacORB part of code was required. Instead a JAVA code to communicate with the FPGA interface (discussed in section 4.4) was developed.

4.3. Reconfiguration Approach

The authors of this paper developed a decentralized reconfiguration approach for power system. The reconfiguration approach without RIA problem (refer section 2.2) includes three steps:

Step 1: Initialization. Agents in the MAS exchange information with their neighboring agents. The information for exchanging includes the agent ID, the real power and reactive power of the component that the agent represents, etc. Each agent in the system has a unique ID number. Each agent has updated information of its neighboring agents.

Step 2: Tree structure generation. In this procedure, the agents break the existing rings in the agent system in order to avoid the RIA problem in later communication. The agents generate a tree structure in the system by communicating with each other. When the tree structure is generated, the agent with smallest id number becomes the root agent in the system. After the tree structure in the system is generated, the root agent will get the net power of system for reconfiguration. The *net power* is defined as $\sum_i S_i - \sum_j D_j$, where S_i is the power supply of the component that is represented by the i^{th} agent; D_j is the power demand of the component that is represented by the j^{th} agent [5].

Step 3: Reconfiguration. Due to an event in the power system, e.g. faults, the status of the components or/and the net power in power system may change. The corresponding agents will then receive the updated status and/or net power of the components and forward this updated information to their neighbor agents. Agents will also monitor the system to make sure that all constraint requirements (voltage, current, frequency, total generation - total load, etc.) for the system are met. In case if any constraint is violated then they will make appropriate reconfiguration decision based on the current status of the component they represent and the information received from the neighboring agents.

4.4. FPGA Interface

The agents on the iPAQs need to setup connection with the power system in the RTDS, in order to receive/send information from components. The iPAQ has a serial port for communication. The RTDS has its own I/O modules that can be used for the input and output data. However, the RTDS and iPAQs cannot be directly connected because of the different voltage levels and communication protocols. An FPGA (Field Programmable Gate Array) interface between RTDS and the iPAQs was developed. The FPGA module consistently scans the components of the power system that runs in the RTDS, and stores the received parameters of the components in the FPGA memory. Thereafter, the agents on the iPAQs can read the updated parameter from FPGA memory to get the current status of the components. When an agent needs to send a control command to the component, it sends a command to the FPGA module. The FPGA module consistently forwards the received control command to the corresponding component of power system in RTDS

4.5. Simulation Results

In this section, results of two reconfiguration cases are simulated to demonstrate the effectiveness of agent based reconfiguration methodology. Figure 7 shows a power system with mesh structure simulated in RTDS. The power system has two generator and two loads. Assume that the Generator1 and Generator2 in Figure 7 have power capacity of 20kw and 10kw respectively, and the Motor1 and Motor2 have power consumption of 15kw and 6.5kw respectively. Motor1 is designated as a vital load, which has to be supplied in the system all the time. Motor2 is designated as a nonvital load, which can be shed when the power system is short of power supply. The net power in this case is 20kW + 10kW – 15 kW – 6.5kW = 8.5kW.

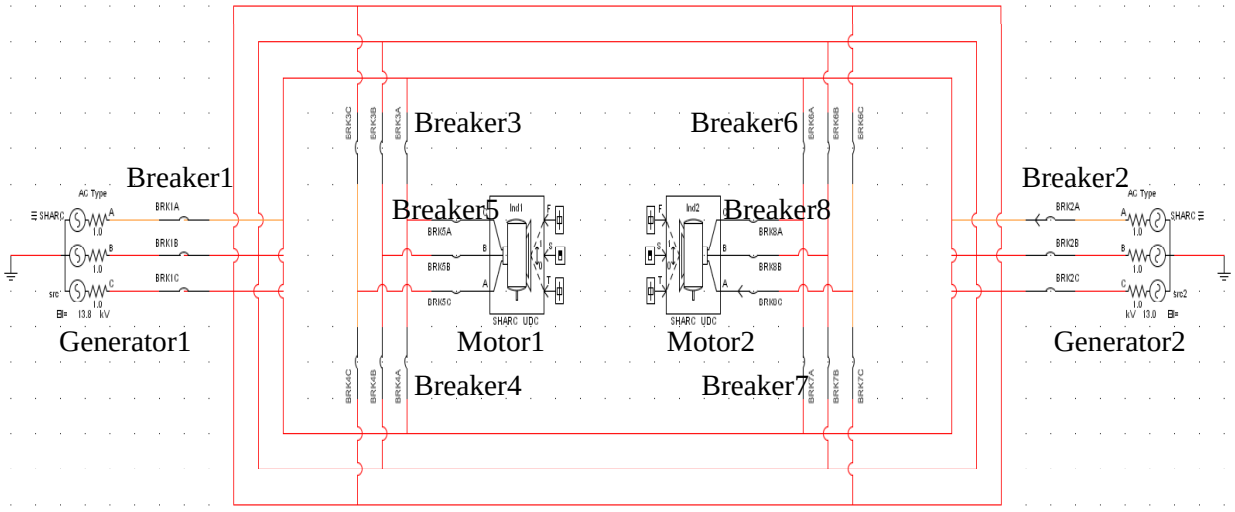
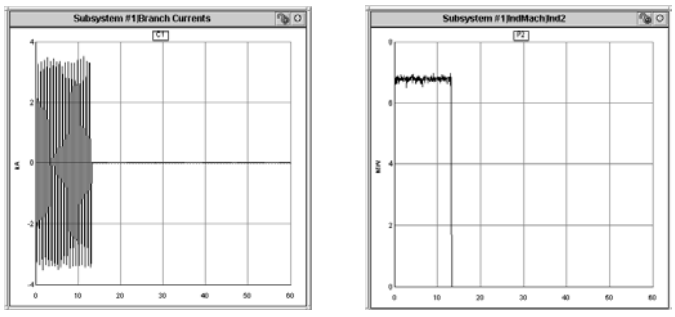


Figure 7. Power system in RTDS

4.5.1. Case 1: Load Shedding

Since the present net power of the system is 8.5kw, which is bigger than 0, all loads in the system can be supplied. Assume that a fault happens at Generator2. The Generator2 is disconnected from the system, the net power of the system becomes –1.5kw. The negative net power of the system mean some loads has to been disconnected from the system. Since Motor1 is a vital load and Motor2 is a nonvital load, the agent of the Motor2 detects the negative net power of the system and sends a request to the agent of Breaker8. When the agent of Breaker8 receives the request, it opens the Breaker8 to disconnect Motor2 from the system. When Motor2 is disconnected from the system, the net capacity becomes 5kw. Figure 8 shows the simulation result in RTDS. Figure 8 (a) and Figure 8 (b) show the current and power of Motor2 respectively.

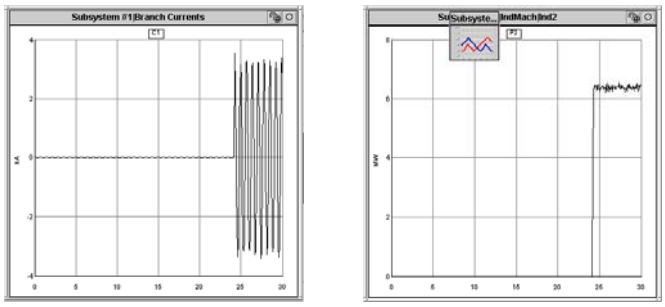


(a) (b)

Figure 8. Power and current of Load2 in load shedding

4.5.2. Case 2: Load Recovery

Assume that the fault on Generator2 is cleared, and Generator2 is reconnected into the system. The net power of the system becomes 25kw. The agent of Motor2 detects the new net power of the system. Because the new net power of the system is bigger than the power demand of Motor2, Motor is can be reconnected into the system. The agent of Motor2 sends a request to the agent of Breaker8. When the agent of Breaker8 receives the request, it closes the Breaker8 to reconnect Motor2 into the system. Figure 9 shows the simulation of the load recovery in RTDS. Figure 9 (a) and Figure 9 (b) show the current and the power of the Motor2.



(a) (b)

Figure 9. Power and current of Load2 in load recovery

5. IMPORTANCE OF MODELING AND SIMULATION

Fig. 10 shows the life cycle of the modeling and simulation the authors currently use. First, the real system is simplified into the system model by using the analytical methods. After the system model is set up, the model is interpreted into computer language for simulation. The simulation results can, on one hand, provide information to modify the system model in order to make the model precise enough to represent the real system. One the other hand, the

simulation results can also be used to validate the real system that needs to be tested.

The results of the modeling and simulation efforts, presented in previous sections, are intended to be used for future planning and guidance of agent based reconfiguration approach. These efforts represent an example of how effective modeling and simulation can address the real world problems and its solutions. It also provides insight into the problem management and solution improvement issues.

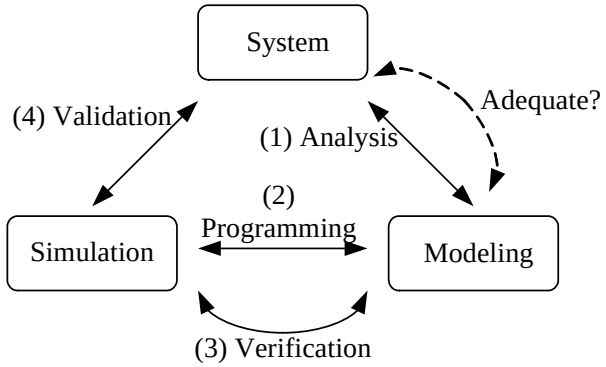


Fig. 10 Life cycle of modeling and simulation

Through the various simulation results, some of which were presented in previous sections, the effectiveness of the developed agent based reconfiguration schemes were verified. These results also pointed out the loopholes or bugs in the reconfiguration algorithm that were later corrected. Proper care given during the selection of modeling approach, later resulted in ease of testing and conducting various experiments in small period of time. Without proper modeling and simulation it would have been impossible to verify the reconfiguration algorithm and MAS performance and hence no further progress would have been made in this area. Also, it should be mentioned that modeling and simulation provide a low cost and low risk approach for verification and performance evaluation. Modeling and simulation is an important step towards practical realization of agent based reconfiguration scheme.

6. CONCLUSIONS

Modeling and simulation is an important way for researching and understanding a system. A model is the simplified representation of the actual system. A simulation always refers to a computational imitation of the operation of the mode. In the research work of the power system, modeling and simulation is always required to test and validate control strategies for power system. The modeling and simulation methodology can provide more efficiency and lower the cost.

Software simulation and processor-in-the-loop simulation are both important ways for power system simulation.. In this paper, the VTB is used as a software simulation tool for power system simulation, while RTDS is used as a hardware simulation tool for power system simulation. The agents system for power system reconfiguration is simulated on iPAQs. The power systems simulated on VTB and RTDS can communicate with agents on iPAQ through certain interface. The simulation results show that these two modeling and simulation method work effectively for validating the agent based reconfiguration approach for the power systems.

REFERENCE:

[1] T. Nagata, H. Sasaki, “A Multi-agent Approach to Power System Restoration”, IEEE Transaction on Power System, vol. 17, no. 2, May 2002, pp. 457-462

[2] T. Nagata, H. Fujita, H. Sasaki, “Decentralized Approach to Normal Operations for Power System Network”, Proceedings of the 13th International Conference on Intelligent Systems Application to Power Systems, Nov. 2005, pp. 407-412

[3] S. K. Srivastava, H. Xiao, and K. L Butler-Purry, “Multi-Agent System for Automated Service Restoration of Shipboard Power Systems”, 15th International Conference on Computer Applications in Industry and Engineering, San Diego, CA, 7-9 November 2002

[4] L. Sun and D. A. Cartes, “Reconfiguration of Shipboard Radial Power System using Intelligent Agents”, ASNE Electric Machine Technique Symposium, Jan. 2004

[5] K. Huang, S. K. Srivastava, D. A. Cartes, “Solving the Information Accumulation Problem in Mesh Structured Agent System”, IEEE Transaction on Power Systems, vol. 22, no. 1, pp. 493-495, Feb. 2007

[6] http://vtb.engr.sc.edu/

[7] Hewlett-Packard Development Company, L.P. http://www.hp.com/country/us/en/prodserv/handheld.html

[8] JacORB website, http://www.jacorb.org/

[9] Object Management Group website, http://www.omg.org/

[10] Lightweight Extensible Agent Platform (LEAP), http://leap.crm-paris.com/

[11] Java Agent Development Framework (JADE), http://sharon.cselt.it/projects/jade/

[12] L. Sun, G. Morejon, and D. A. Cartes, “Integrating Virtual Test Bed and JADE Agent platform Using CORBA for Reconfiguration of Shipboard Power Systems,” IEEE Power Engineering Society General Meeting, 18-22 Jun 2006, Montreal, QC, Canada.

[13] http://www.rtds.com/

[14] Insignia Solutions, Inc http://www.insignia.com/

[15] Object Management Group website, http://www.omg.org/