AN INNOVATIVE MODEL FOR CONTROL OF COMPLEX DISTRIBUTED POWER NETWORKS

by

Syed Mohammed Alamdar Raza

A thesis submitted in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

National University of Sciences and Technology

2010

ACKNOWLEDGMENTS

I very humbly thank Allah Almighty for His blessings in giving me the wisdom, knowledge and understanding without which I would not have ventured into this research work. I seek Him further, to broaden my vision and enlighten me more in my career.

I express my deepest sense of appreciation for Dr Muhammad Akbar, the research supervisor of this thesis, for his supervision, guidance and encouragement in carrying out this study. My special thanks to Dr Farrukh Kamran, who provided me the understanding and inculcated interest in this entirely new but most prevailing field. Patronage of both of them was inevitable who were like beacon of guidance during entire course of my research. Also, my special thanks to the members of the guidance committee, namely Dr M N Jafri, and Dr Saeed Murtaza for their valuable suggestions. Their advice and critical review was indeed a great help in shaping this paper. I am extremely grateful to them for providing immensely useful guidance at all times during compilation of the thesis and conduct of research. I also thank **HEC** for financial support.

I would like to express my sincere thanks to Brig Ashraf, Col (R)Raja Iqbal, Col Naveed Khattak and Lt Col Attique who made wholehearted contributions to facilitate in compilation of this thesis.

Besides, I thank my teachers, friends and family members for their moral support and patience which enabled and encouraged me to complete this work.

To

My Children

Desaar, Kumail and Fizza

List of Figures

List of Tables

Chapter # 1

Introduction

1.1 Background

Shifting to a post-fossil energy setup is no small task. Merely the logistics side of the problem encompasses many challenging requirements such as replacing crude oil, coal and gas-fired power plants with environment friendly, renewable energy technologies. It is not only a matter of removing one and plugging in the other. Renewable energy sources often require huge spaces to produce useful amounts of power, and need to be located in areas most suitable to their generation needs (bright regions for solar, airy for turbines and the ocean for wave etc.) and hence be located in far flung areas. These may be supposed to provide power to limited and isolated pockets or may be connected to the main grid. In both the scenarios there are definite peculiar control requirements for local or distributed power system networks.

If the alternate energy resources are used in conjunction with the conventional power generation setups in main grids, control for distributed power networks will be required. A setup of diverse, scattered sources of power generation is generally called "distributed energy network" and it has defined advantages over the existing, mainly centralized infrastructure. Distributed energy can be more strong against an accident or attack on the power grid. Failure of a 5 MW wind turbine would be undesirable, yet it will not be as disastrous as suddenly taking a 1000 MW coal power plant off the grid. Distributed power also permits more resource flexibility. The more varied the resources used to produce electricity, the less likely are outages resulting from limited availability of one of them. This is particularly more important due to the varied nature of air and solar energy. Output from a particular wind or solar farm will rise and fall with local conditions,

but the overall availability of electric power from various locations and resources can still be consistent.

Resorting to distributed energy is currently more complicated than centralised power. Some of the contributing factors are managing the difficulties related to variable power generation, changing usage patterns, distances involved, enormous increase in nodes of network and multiplicity of sources. In addition to dealing with variable, dispersed inputs, distributed networks will have to allow efficient routing of power, with lesser idle or unused generators. These would allow an overall lower level of generation to support continued levels of operation. The distributed power system computerised networks will be a fundamental part of post-oil era powered with green and environment friendly resources. Moreover, like computer networks, a successful grid would like to have a setup, where the intelligence of the system can be placed close to all important locations of requirement where decisions are to be made, instead of centralized network intelligence.

Since power network is a large-scale system consisting of many subsystems, it is no longer the best solution to manage by using only the centralized control schemes or loosely decentralized control schemes. The reliability of the control system framework, which is highly coupled and interconnected with subsystems, decreases since a single failure can bring down the entire system. In order to overcome these problems, the control system should operate at a higher level of automation, flexibility, and robustness based on a new methodology which can work effectively in a large-scale distributed complex system. Recently, there has been a growing interest in distributed agent systems to deal successfully with the complexity and distributed problems in power systems. Each agent system can be assigned special functions to solve the

distributed problems. Moreover, in the multiagent system the agents can work to solve problems, which are beyond the capabilities or knowledge of an individual agent. There have been many applications of agent-oriented mechanisms in control and monitoring systems. Agent based system can help in solving the distributed control problems by allocating responsibilities to various locations. The multiagent system also helps in monitoring the condition of system and providing effective asset management by diagnostics and protection against faults. Both areas are built upon the multiagent system properties, such as proactive, reactive, and social properties, as well as other fundamental properties.

A major concern for the networked sensing and actuation of a large-scale system is the complexity, due to the number of components and their interaction patterns and communication delays. This complexity is raised when a control system is required to become intelligent by implementing a completely new variety of knowledge processing functions. The multiagent system paradigm, as the state-of-the-art software engineering concept provides a comprehensive and unifying framework for building large-scale control systems. However, a review of the technical literature reveals a lack of design methodologies for multiagent systems, though few software development platforms are available with limited features. This points to the need for a new system design methodology for the control of large scale systems such as distributed power networks. A distributed system control concept has, therefore, been presented here as an effective methodology to control a large-scale distributed power network. An organization for the system has been described as the foundation of the distributed control system. It has several functions that provide efficient ways to control locally, and to accommodate and overcome the complexity in the large-scale distributed systems.

1.2 Problem Statement and Objectives

Green power based and conventional generation setups and infrastructure networks, encompassing vast distances, with fully centralized control will warrant full duplex, high baud rate communication channels with features of duplicity and redundancy, especially if the systems are interactive. These setups are susceptible to frequent failures owing to the fact that malfunctioning of one part of the network will disseminate unpredictably in no time not even sparing the central control. Consequently the central control will suffer from the same trouble which, it primarily is expected to rectify.

The increasing complexity and interconnectedness of such systems poses fresh challenges for reliable and safe operation. Not a single entity can have control over distributed and highly interactive networks especially in real time. Moreover typical mathematical models that are employed for simulation and control are not able to properly handle the complexity and interconnection of more complex power networks. In most of the cases there are no procedures for understanding behaviour of the complex systems. Handling of disturbances in all these networks, along with avoiding of cascading effects through out and between networks requires a fundamental understanding of system's dynamics, and ability to exercise effective distributed control.

The challenge of managing distributed and interconnected systems therefore demands devising techniques capable of carrying out decentralized analysis of problems and intervening locally to fix and save the problems from propagating through distributed power networks, such as making use of multiagent systems.

An agent is a computer software program that is autonomous and situated in some distributed environment in order to meet its design objectives. Since the agents are faced

with different environments, they are designed differently for the given environment. In a large-scale distributed complex system, the agent's autonomous properties can reduce the complexity by reducing the coupling problems between the subsystems. Furthermore, the proactive, reactive, and robust properties can be well suited for applications in a dynamic and unreliable situation. Since the agent is situated in an environment that is a distributed power network, it needs a perception and effecter to act and react. First, the sensed raw data is processed and mapped into a scenario. Then an objective (a sub-goal) is initialized under the situation to achieve the main goal which is the optimal operation. After confirming the objective, the best plan is chosen for the objective (sub-goal) in the decision-making. Depending on the plan, an algorithm module is selected to launch it. Finally, the action made by the algorithm module effects through the effecter into the environment.

The objectives for this research are to propose Object, Use-case and State Machine models of a concept based upon agent system for control of distributed power system network. The proposed system should identify the requirement of different agents considering them as entities fully described with attributes and functions. The proposed model should specify interaction of different agents and events and should be able to serve as a basis for identification of requirements for control of distributed power networks.

The objectives also include simulation of a typical power system for establishing the requirement of distributed control thereby highlighting the advantages achievable through making in-situ decisions rather than relying on central control for retaining synchronism in generators and avoiding overloading of the system especially under increasing load conditions..

1.3 Approach

An agent system for multi faceted distributed generation setup has been proposed identifying role and requirements of each agent. The Object Model, treating various agents as Entity/ Object has been proposed showing their interaction/ interfacing. Each entity has been elaborated with its attributes. In addition, the services expected from each agent are highlighted. The Use-case Model is proposed showing assigned role of each agent. Each use-case has been elaborated with its purpose, data handling, stimuli and responses. A lay out of the sub-systems of each of the proposed main agents has also been included. An elaborate State Machine Model is proposed showing behavior of system in real time scenarios and identifying different states, events and interactions. Finally a power system is simulated using Matlab/ Simulink platform to establish requirements of a local agent decision for assuring stability by resorting to decisions best suited to the environment. Static and dynamic load flow analysis of the selected system for ascertaining instantaneous values of each bus voltage along with its phase angle, active power and reactive power at each bus has been carried out using Newton Raphson technique. Rotor angle and speed of generators are found at defined intervals in pre fault, post fault, and on fault conditions as per defined intervals in various scenarios of load increase utilizing swing equation. Remedial measures commensurate with nature of disturbance are taken by distributed control to save the system from going out of synchronism or becoming overloaded. Expected delayed responses by central control or no responses and consequences thereof including system going out of synchronism or becoming overloaded etc have been demonstrated.

1.4 Outline of the Thesis

The thesis is organized into eight chapters. Chapter 1 is introduction to the thesis and Chapter 2 is based upon a literature survey to account for developments made in control of

interconnected power systems and presents critical literature review. Different control aspects concerning the issue under discussion have been highlighted.

Chapter 3 reviews control of real time systems and multiagents with an exhaustive discussion on drawbacks of existing system in the wake of increased complexity and feasibility of utilising distributed systems for control of complex power networks, identifying dividends which can be accrued by utilising multiagents.

In Chapter 4, important features of requirement engineering have been taken up culminating into features of object modelling. It is followed by a discussion on object model of multiagent system proposed for control of distributed power network.

The model of system use which interacts dynamically with environment, as recommended by UML, being a scenario based technique for requirement elicitation has been covered in detail in chapter 5.

Chapter 6 describes concepts dealing with flow, interaction and sequencing of operations applied to the proposed state machine model of multiagent system for control of distributed network.

Chapter 7 includes the simulation of a power system, static and dynamic load flow analysis of the selected system, solution of relevant swing equation and demonstration of advantages of distributed control as compared to central control or no control.

Chapter 8 comprises of conclusion of the thesis describing the extent of work carried out and recommendations for future work.

Chapter # 2

Models for Control and Literature Survey

2.1 Introduction

In 1978, Sandell et al carried out a survey of a variety of alternates for decentralized control [1]. It was found that a fine amalgamation of professional assimilation and scrutiny may be used to define in a reasonable fashion an adhoc control mechanism in respect of a dynamic system. The authors reached at the conclusion that procedures were required that could present a design engineer with many acceptable control structure options for further consideration.

In this chapter a resume and scrutiny of the previous work carried out in the field of control and management of distributed and multi machine power systems are given. A brief review of various techniques of modeling a problem of control has also been included. The contents of this chapter have been taken from [1] and [30].

2.2 Control Methodologies

Here various types of concepts finding importance in overall control methodologies have been evaluated. The main task of control problems, models that can be employed, models formulating a problem, and agent architectures finding utility in handling these problems have been assigned due consideration.

2.2.1 Main Task of Control

From the control point of view, a system is to behave in a defined manner. This must fulfill an assignment, that can encompass approaching certain goals while ensuring that none of the constraints which are possible are contradicted.

Assignments can be identified by a person or some other object, or these may follow from certain characteristics of behavior of the system. Objectives can be either immediate goals or long term goals, e.g., to bring the system in a given state, or to maximize the long-term performance or to minimize the operation costs in long term. It is important that tasks should have one goal. These can possess many goals which can possibly be contradicting. In such scenario these are known as multi objective tasks.

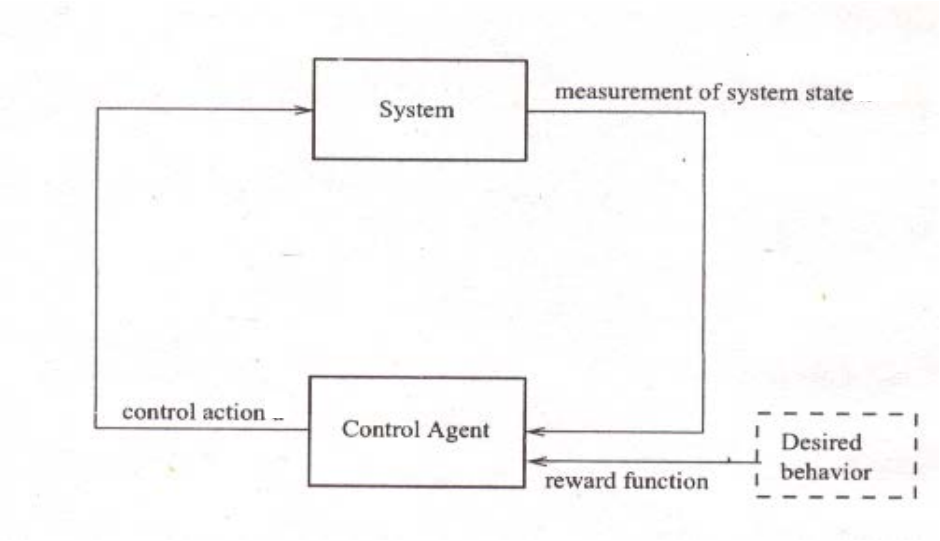

Figure 2.1 General Scheme for Controlling a System

By keeping in view the dynamics of the system and probable constraints placed on the actions, tasks to be carried out on the system are to be selected such that the requisite actions expected of the system are achieved. Actions that achieve the goal are found and process formulates Dynamic Control Problem (DCP). A standard DCP setting is shown in Figure 2.1. In order to determine an action such that system performance tends to approach the expected behavior as close as possible the control agent measures the state of the system.

The general DCP can be formulated as:

Find the actions such that the goal is achieved optimally
subject to

a model of the dynamic system
with defined inter actions and states.

This problem can be seen as an optimization problem, since the actions have to be chosen such that they achieve the goal in the best possible way. It is important that the goal remains independent of the definite model of the dynamic system. With a view to obtaining the goal it may be presumed that there exists a system model of the system under analysis. Ths model can be utilised to accurately identify the goal and to make a prediction about the system behavior given certain actions. A control problem model defines the exact control problem often based on the system model. The problem can be solved by agents using control problem model. Agent architecture provides the organization of agents which adapt some communication protocol.

2.2.2 System Models

State of the system, behavior of the physical system, actions on the system, and possible disturbances are described by a dynamic system model. There may be limits on possible actions and states in addition to system dynamics, which means the models are merely relevant in a given operating domain. The constraints can be due to limitations which are technical, regulatory and measures of safety etc. Models of system may change with time, which means, the structural parameters of the model are not bound to stay constant.

2.2.3 Control Problem Models

The overall goal may consist of one central goal, a variety of decentralized goals, or hierarchical goals commensurate with system model structure. While resorting to a central goal there is a wholesome goal for the entire system. When parts of system in a wholesome system hold their own exclusive goals, decentralized goals become relevant. Hierarchical goals become relevant when subsystems possess goals which (partially) overlie, or when goals for a system can

be ascertained on various levels of abstraction/ detail. The goals generally have a close association to the overall system.

Three varied classes of control problem models can be identified including, a central problem model consisting of a single DCP. The other model comprises of a decentralized problem of multiple smaller, independent, DCPs. If the smaller DCPs have no conflicting goals, the combination of the problems is equivalent to the overall DCP. However, if there are conflicting goals, the combination of the problems need not be equivalent. Lastly, a hierarchical problem model consists of a number of abstraction layers, in which higher layers contain more abstract DCPs, and lower layers more concrete DCPs. The higher layers depend on information from lower layers and vice versa.

2.2.4 Types of Agent Architectures

Solution of DCPs can be accomplished by using agents or controllers. Agents are softwares of problem solution that have many abilities including reasoning, learning, and communicating with others so as to find a solution to a given problem. Agents possess set of actions containing skills and they also have an information set containing their knowledge (including information from sensing and communicating)[2].

Architectures are used to organize agents, for example., via links of communication including, central agent architecture, having only single agent, a decentral agent architecture, having many agents possessing nil interaction with one another, and a hierarchical agent architecture, where there are various layers of agents. Higher layers may supervise and receive information from lower layers. Layers located at lower level can receive instructions from and offer information to upper layers. Agents situated on the similar level/ layer can be permitted to

interact directly with one another, as well as via the layers located at upper level. Figure 2.2 depicts a typical example of hierarchically structured agent architecture [3].

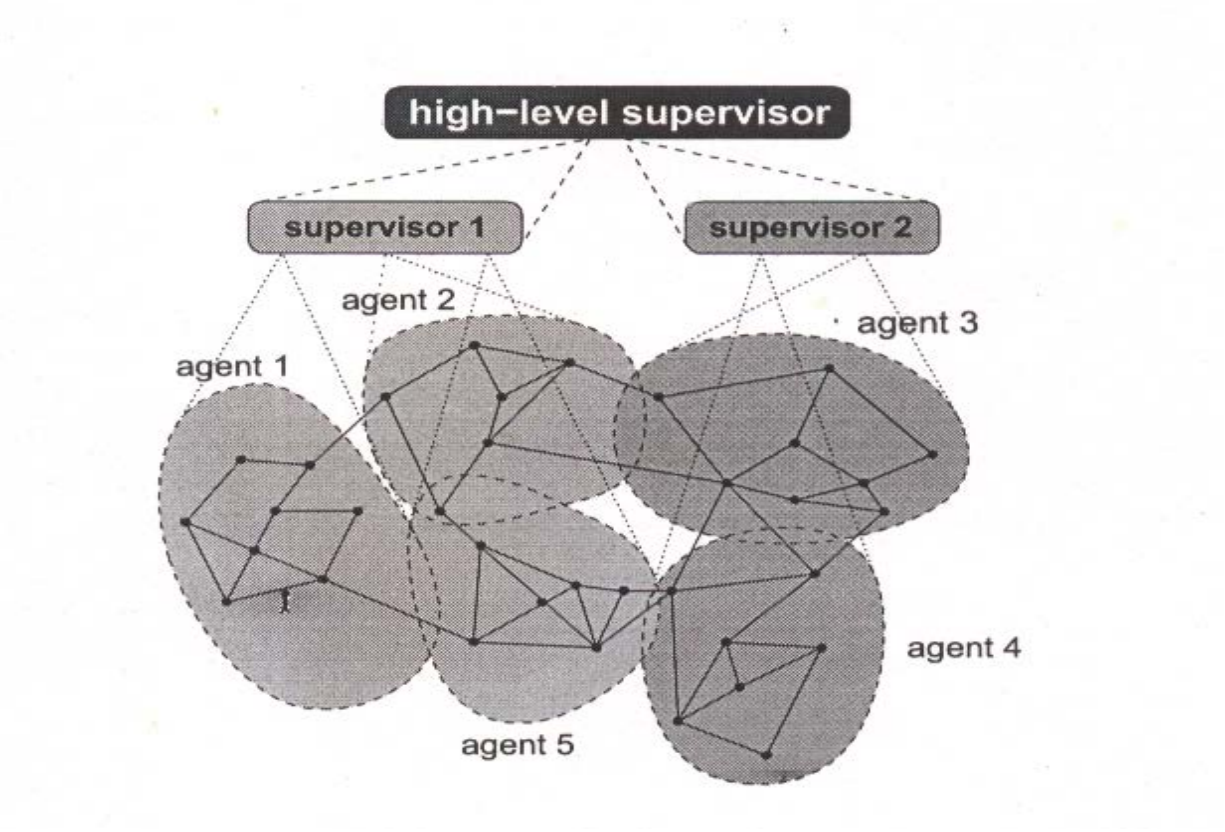

Figure 2.2: Hierarchically Structured Agent Architecture

Targets for levels located at lower level agents are defined by those located at upper level. Physical systems are handled by agents assigned the lowest level in the hierarchy. Besides structured in layers, Agents may also be categorized in groups within which they are allowed to interact with each other without seeking approval of any other entity. While finalising hierarchical architectures, it is not only essential to ascertain information that can be communicated, but also the order in which the information can be approached by agents, ie, communication protocol is the requirement.

2.3 Control of Distributed Power Systems and Networks

Many scientists have considered using prediction as part of a distributed control architecture. Few of the examples of these are architectures in which a single predictive controller is used as replacement of decentralized PID controllers [4], multiple different controllers are manually engineered as replacement of decentralized PID controllers [5-6], or prediction is used as supervisory layer in a cascaded setting [7-8]. The control architectures involved are typically engineered with insight in the specific application domain. Other architectures consider multiple subsystems that depend on one another, and that employ Model Prediction Control (MPC) in order to optimize system performance [9-21]. Methods described by them are general in nature, less application specific, and hence relevant to different scenarios. A comprehensive and decentralized control methodology is, therefore, a much desired option. Agent-based models provide understanding of large scale, behaviour of complex power infrastructures which is non linear for better control of disturbances mainly concerning cascading [3]. Assignment of different responsibilities to subsystem is the key to simpler independent bahaviour of individual components. In succeeding paragraphs, system model decomposition, control problem decomposition and problem assignment techniques etc as handled in different research papers will be discussed.

2.3.1 System Model Decomposition

Typically there are two ways in which a decentralized or hierarchical system model is formed, based on the way the overall system is considered. The central system model will be applicable. It means a centralized system model has to be constructed and then defragmented into many subsystems without affecting structural properties present in the model of system. This approach is known as top-down approach where a linear dynamic system is decomposed

analytically into a similar set of subsystems with inputs which are coupled[10,22]. The other is centralized system model considered implicitly. This typically involves modeling a subsystem and the relations with other subsystems directly ie the decomposition into subsystems is based on engineering understanding. In this case one resorts to a bottom-up approach and the subsystems are conceived without first bearing in mind a model for the overall system [9,11,16-17].

In general, a system can be subdivided by bearing in mind various time scales in a system and identifying weak couplings between subsystems [1]. No specific methods are available to resort to decomposition.

2.3.1.1 Decentralized Model Decomposition

In decentralized decomposition, all subsystems are independent of one another. However, many scientist use the word decentralized to address a group of subsystems that can communicate with one another. These subsystems are a special case of a hierarchical system model. Purely decentralized model decomposition is only possibly when two subsystems are completely independent of each other or when they are assumed to be independent of each other. The term decentralized should not be confused with the more general term distributed. The latter refers to systems consisting of subsystems in general, and not in particular to systems consisting of strictly independent subsystems.

2.3.1.2 Hierarchical Model Decomposition

Hierarchical system model decomposition arises when coupled subsystems depend on each other. Higher levels in a hierarchy may be more abstract or may span a longer time period (e.g., they may have a lower communication, computation, or control rate). The coupling between subsystems can have different foundations. Sometimes the coupling is based on

physical variables and modeled explicitly. In this case control of a system is divided in different sections as subsystems. In each subsystem model the controls and state of a neighboring subsystem are taken into account. Multi-vehicle formation stabilization can also be considered where system models of the vehicles (including constraints) are uncoupled [23]. However, one layer higher, at a more abstract level, the state vectors of the subsystems are coupled due to constraints that make the vehicles drive in a formation. Similarly optimal dynamic routing of messages is considered in a store-and-forward packet switching network [18]. The nodes in the network are seen as subsystems with connections to neighboring subsystems. In particular, by reformulating the subsystem model they get rid of constraints. Yet at other times the coupling is more artificial and does not have a clear physical meaning. As an example, subsystem model is defined for each section in a distribution network and compatibility equations between subsystems are to be satisfied. The decentralized approach is based on an augmented Lagrangian formulation, where the flow balancing equations are dualized. In this formulation, the Lagrangian multipliers become the coupling variables [11,16].

2.3.2 Control Problem Decomposition

Generally the structure of the system model and the structure of the problem cannot be distinguished. A control problem is therefore associated with the goal of each sub system. It is believed that it is not always necessary to allocate a problem of control with each subsystem. One can opt to define goals over a number of subsystems, instead of separately for each subsystem Analytical decomposition of a centralized goal can ascertain goals for subsystems. It can be better understood with example of worst-case approach which is taken by defining for each subsystem, a subproblem of finding the Lagrangian multipliers that maximize the problem of finding the controls. The multipliers must minimize the augmented Lagrangian [11,16], and

an ad-hoc engineered subproblem goal specially where a goal is formulated for each subsystem [18].

Specific goals are not considered for the lowest layer at occasions [23]. However, a centralized goal is identified on the subsystems of the lowest level [10] at a level higher than it, or exchange of predictions on the bounds of agents' state trajectories [19-20]. This way the agents possess information regarding the trajectories that the subsystem of the other agents are likely to make. Through this prediction of behaviour of all other agents can be done while still keeping the control problem decomposition to subsystems.

2.3.3 Agent Design and Problem Assignment

The information set of the agents is generally assumed to contain sufficient information for solving the subproblems. The action sets of agents are assumed to be sufficient in this regard [11,16,18]. There are two options available in case an agent does not possess access to certain information that it needs directly . These options include: collect the data via communication, or devise mechanism to predict the information. In the proposed model of control of distributed power networks, each agent has been assigned the capability to send request for obtaining required information from any other agent. The available information will be passed on, after carrying out updation and as per data structure of the requesting agent [24].

Optimal partitioning of agents into groups has drawn much of the interest of researchers[12]. Removing the communication channel amongst agents (at least among agents in different sub-groups) is of essential value to a group of scientists [22]. By minimising communication between agents, they can work better at their own speeds. However this requirement must be weighed against the total cost of control actions. The authors have

suggested a formulation wherein this trade-off can be inconsequentially implemented by finding an arrangement assigning agents to groups. Sensitivity-based criterion has also been proposed for considering how information must be communicated to the groups [22]. In systems having grouped agents, the closed-loop control action sensitivity when compared with measurement of output is used as a criterion for deciding regarding a given output measurement which should be made available to a particular group of agents.

Specific agents are generally tasked to solve each control subproblem. Generally the designs for agent architectures are prepared being offline which do not change online [10]. The information that agents can exchange amongst each other is determined before hand such as by minimizing a communication cost, or objective function at system's overall level depending upon the requirement to be handled using artificial intelligence etc. The agent-based model covered in this thesis also proposes use of evolutionary softwares and genetic algorithms for prediction and objective function minimization. Each agent, however, has been assigned multiple tasks instead of increasing number of agents by allocating specific tasks to each agent [24].

2.3.4 Synchronous and Asynchronous Iterative Solutions by Agents

When decisions are to be taken among agents with conflicting goals, it is easier and realistic to reach at a solution through iterations [12]. Considering the decisions of its neighboring agents, every agent revises its decisions. Agents arrive at optimal decisions independently if the dynamic subsidiary problems are not coupled. However, if this is not the case, there are two methods to handle couplings: synchronously or asynchronously.

In the synchronous case precedence Constraints are placed as binding on the iterations in case of synchronous method. This necessitates that the faster agents wait for slower agents.

Serial synchronous methods and parallel synchronous methods can mutually be distinguished. In serial method one agent can take a step at a time whereas in the parallel case agents have to wait for all other agents to finish the step in hand before proceeding.

Asynchronous treatment preferred over the synchronous case because it permits all agents to run at their own speed, and hence the agents spend no time waiting for one another. This, however, is achieved at the cost of uncertainty in information, since agents may not know precisely regarding actions of other agents. Asynchronous iterations can be noticed in state machine model of Figure 6.1, where state of automated negotiations and risk management continuously revises the information it holds through iterations, on reconfiguration of network, provision of ancillary services at identified nodes and on receipt of results from evolutionary programming/optimization for minimum power transfer [25].

Parallel synchronization can be dealt in two-step algorithm. On adapting this approach every agent handles its subproblem first, using some values or parameters of the previous step. With the help of information from other agents these values/ parameters are optimized. In the next or second step this information is made available in which each agent exchanges information regarding its parameters with other agents [11,16]. In few cases agents also communicate and share their future plans to each other at the end of each optimization step [19-20,23]. Worst-case disturbances acquire priority because agents prefer to solve local min-max problems to optimize performance with respect to these. To acquire conservative solutions and predictions, parameterized feedback of state is introduced into the multiagent formulation of MPC. Agents' capabilities can be incorporated with a view to joining the trajectory formulaton with operational constraints and stabilization of dynamics of vehicles. This is done by adding a potential function reflecting the state information of a possibly moving obstacle or other agent to

the cost function [26]. Efforts to incorporate optimization features through synchronous or asynchronous iterations for revising/improving the information and decisions by individual agents have thus proved quite useful.

2.3.5 Choice of Action by Agents

Agents can opt to use various methods of physically choosing actions to perform at a given point of time or scenario. As far as a typical single-agent MPC is concerned, the agent opts to invoke the very first action of the sequence found as a result of solving the control problem assigned to it. While studying multiagent MPC, however, there are many available options since an agent may like to adopt suggestions from agents sitting in neighborhood regarding its actions. Agents can fully exercise option to select their actions and implement them such as, by merely opting the first action taken from the sequential list of optimal actions. A higher-layer agent can also be considered which collects data from various layers located at lower level to update parameters for each agent exclusively[11,16].

Actions of a particular agent may be chosen by other agents as well which means actions can also be shared [27]. Many other agents may also be permitted to utilise capabilities of a given agent. As an alternative, agents can opt actions to be performed democratically by allowing various agents to cast vote about action to take place. This concept is based upon the assumption that the majority knows what is right to do and hence it can prove fruitful [27]. Finally, agents may be allowed to trade their actions. In this case the agent having the bid which is highest is allowed to choose the action that an agent should act upon. It can be beneficial in scenarios where there is only a very limited resource available required to be shared.

2.3.6 Automatic Learning Features of Agents

Automatic learning is the latest concept which is likely to enhance the efficacy, broaden the domain of applied fields and strengthen the flexibility/ adaptability of protocols related with cooperation. As an example of applicability, learning may effectively be introduced for identification of parameters, or for enhancement in capacity of problem-solving and decision-making abilities. Learning imparts an agent the ability to predict about future actions of neighbors . Learning specifically yields dividends when agents work asynchronously.

There exists a possibility of on-line identification procedure jointly based on a MIMO parameterized model of the physical characteristics of the system and Kalman filtering [11]. In addition to this, a Kalman optimal estimator defined on the grounds of the on-line identified control can be employed with a view to making estimate of the state of the subsystems for which the subproblem of an agent is has been identified. Neural networks may also be allocated to the agents signifying nodes in a network [18]. Online computational requirements can be improved by training the neural networks off line. The models of the nodes can be allowed to rely on values acquired by neighboring nodes in future and each agent has to estimate these values [17].

2.3.7 Concepts of Artificial Intelligence

In practice, many nonlinear processes are approximated by reduced-order models, possibly linear, that are clearly related to the underlying process characteristics. However, these models may be valid only within certain specific operating ranges, and a different model may be required in the wake of changed operating conditions, or the control system should adopt the new system model parameters. The advent of Artificial Intelligence (AI) techniques, such as neural networks, has solved this problem to a great extent. The neural technology offers many more benefits in the area of nonlinear control problems, particularly when the system is

operating over the nonlinear operating range. The applications of neural networks in power system control are available [28-29].

A new control scheme to incorporate the nonconforming load problem was presented, in which an effort had been undertaken to develop algorithms capable of discriminating between non controllable short-term excursions and controllable long-term excursions [30]. Out of the two techniques described, one was developed using a neural network algorithm for pattern recognition of controllable signals, and the other technique was based on the detection of the controllable signal in the presence of a noisy random load using a random signal probability model. Test results reveal that neural network-based control implementation had a significant improvement over the modern control implementation. LFC system performance was evaluated with a nonlinear neural network controller using a generalized neural structure to yield better system dynamic performance than the individual neurons [31].

A four-area interconnected power system model with reheat nonlinearity effect of the steam turbine and upper and lower constraints for generation rate nonlinearity of hydro turbine was considered for the investigation [32]. It has been shown that the control problem can be viewed as a stochastic multistage decision-making problem or a Markov Chain control problem and have presented algorithms for designing control based on a reinforcement learning approach [29].

The fuzzy logic control concept departs significantly from traditional control theory, which is essentially based on mathematical models of the controlled process. Instead of deriving a controller via modeling the controlled process quantitatively and mathematically, the fuzzy control methodology tries to establish the controller directly from domain experts or operators who are controlling the process manually and successfully. Many studies exploiting the fuzzy

logic concept in AGC regulator design dealing with various system aspects have appeared in the literature [33–35].

Contribution considering the problem of decomposition of multivariable systems for the purpose of distributed fuzzy control was reported by Gegov [89]. The proposed decomposition method has reduced the number of interactive fuzzy relations among subsystems. The concept and development of AGC using Artificial Neutral Networks and fuzzy set theory to utilize the novel aspects of both in single hybrid AGC system design for power systems has also been mooted [36].

These days, Genetic Algorithm (GA) is the most popular and widely used algorithm of all the intelligent algorithms. GAs have been widely applied to solve complex nonlinear optimization problems in a number of engineering disciplines in general and in the area of control of power systems in particular [36-43].

A reinforced Genetic Algorithm has been proposed as an appropriate optimization method to tune the membership functions and rule sets for fuzzy gain scheduling of load frequency controllers of multi area power systems to improve the dynamic performance [38]. The proposed control scheme incorporates dead-band and generation-rate constraints also. Later, contrary to the trial-and error selection of the variable structure feedback gains, a genetic algorithm-based selection of feedback gains has been advocated for load frequency variable structure controller [39]. The selection scheme provides an optimal feedback gain selection in the variable structure control, and the test results show that not only the dynamic performance has been improved, but also, the control effort is dramatically reduced. A comprehensive study on control of an autonomous power system using combined intelligent techniques is also available [36].

A higher order robust dynamic performance is achieved with LFC designs based on GA and linear matrix inequalities [42]. The desired control parameters have been obtained by coordinating GA with linear matrix inequalitie control toolbox optimization. A new GA based fuzzy AGC scheme of a multi area thermal generating system is developed [43]. The scheme is capable of evaluating the fitness of GA optimization by selecting a function like “figure of merit,” which directly depends on transient performance characteristics like settling times, undershoots, overshoots, and time derivative of frequency.

2.4 Identification of Research Area

After going through the literature, it can be inferred that the authors have generally concentrated on single problem pertaining to control of distributed power networks. For example, research is available on transient stability using various techniques such as Lyaponov functions, fuzzy logic based power system stabilizer and nonlinear controllers with derivative adaptive technique. Similarly performance prediction of distributed power system based on small scale prototypes and a coordinated voltage control for power system voltage stability to control dissimilar control actions at geographically distant locations has been proposed. Moreover work on possible modeling techniques of complex systems is also available. However a relatively comprehensive and wholesome solution has been deliberated upon which can encompass functionally vital aspects of a control system of distributed power networks possessing features of evolution, self healing and autonomous decision making at different nodes thus managing real and reactive power along with allied ancillary services, reconfiguration on occurrence of fault and restoration on rectification subsequently, data handling and award of autonomy to various geographically scattered nodes for unhindered quality supply of electricity etc. Integration of different dimensions of the complex problem of controlling distributed power network of today’s

magnitude is certainly an ambitious demand requiring efforts and resources of large, qualified and experience consortia. These consortiums should comprise of experts from multi disciplinary fields with support of world community as identified by large number of researchers in the field. Stepping in the right direction and towards the objective is the only viable option.

With a view to providing a basis for further development a hierarchical conceptual model based upon agents having communication and interface capabilities has therefore been proposed [24-25,44-49]. Goals of each agent have been defined individually and iterations within states have been described for automated negotiations, reconfiguration and risk management based upon iterative evolutions. Actions of agents are autonomous yet each agent can initiate a request for overarching actions, assignments and data.

2.5 Summary

This chapter provides a critical overview of philosophies and work carried out in the field of control of power system networks and modeling of systems. Common aspects in the available research conducted in the relevant field have been identified. This has led to identification of certain groups and attributes at a rather non mathematical level. Recent developments such as control schemes based on concepts of neural networks, fuzzy logic and genetic algorithm have been accounted for. Emphasis has been given to categorizing various control strategies reported in literature that highlight their salient features.

Chapter # 3

Agent-Based Computing

3.1 Introduction

Control of Real Time System encompassing geographically distributed areas warrants conceiving innovative techniques to handle situations and scenarios. A promising solution is use of multiagent systems. In this chapter a review of Agent Based Systems is carried out with special emphasis on their properties and applications. In addition, a detailed discussion of using multiagent system in distributed power networks is included, highlighting dividends that will be accrued by utilizing multiagent systems for control of these networks. The contents of this chapter have been mainly taken from [68-70] and [72].

3.2 Open and Dynamic Computing Environments

The trend of software development is shifting from stand alone computers to distributed, open and dynamic systems presenting an environment made up of computers in infrastructures, in support systems and embedded in vast arrays of devices. These computers are networked with ability to interact dynamically forming new configurations to suit prevailing needs. However, the characteristics of dynamic and open environments in which systems of different origins, make, type and purpose must interact and operate effectively within rapidly changing circumstances and with exponentially increasing quantities of available information, suggest that improvements on the traditional computing models and paradigms are required [2].

In particular, the need for awarding autonomy, to enable components to respond to changing circumstances while trying to achieve overarching objectives without the need for user intervention is fundamental. Agent technologies have become the most valuable tool that is being used to tackle the problems associated with managing the complexities that arise.

Agents are viewed as autonomous, problem-solving computational entities capable of effective operation in dynamic and open environment. They are deployed in environments in which they interact, and possibly cooperate, with other agents (including both people and software) and may have conflicting aims. These are exactly the kinds of characteristics that are needed in the new computational environments. Agent-based systems have emerged over the past few years, from a convergence of technologies in distributed object systems and distributed artificial intelligence, and have seen rapid and dramatic growth both academically and commercially.

3.3 Object Technologies and Agents

The relation of agents to objects has caused difficulty in understanding what it is that makes agents distinct [2]. While object-orientation as a programming paradigm has achieved much success, and offers a valuable abstraction for the development of complex systems, agents provide a different and higher level of abstraction. Like objects that provide encapsulation of state and behavior, agents also encapsulate properties. However, objects are essentially passive in nature. They have no choice to interact, and are simply invoked by other objects to perform particular tasks or execute particular functionality. In contrast agents have the ability to decide for themselves to participate in computational activity, and to perform the desired operation. This is the fundamental distinction that marks out agents as distinct by virtue of their autonomy. It is this autonomy that is also responsible for providing the flexibility that is needed for open and dynamic environments.

In terms of modeling, objects are regarded as a valuable way to view the world, and an agent-based approach offers a much more natural representation of real-world systems in which

different individuals, interact according to their own agenda and priorities. They, then, can come together to achieve objectives that might not, or not as easily, be achieved by the individuals alone, but they do so when it is appropriate. When the goals of individual agents are closely aligned, and if they respond to requests made to them, and always provide information when queried, then the resulting systems may come close to an object-oriented system.

In short, agents can be distinguished from objects in that they are autonomous entities capable of having choice over their actions and interactions. However, they may be constructed using object technology. Moreover, agents typically run in their own thread of control, as opposed to standard object systems, which have one thread.

3.4 Concept of Agents

The concept of agents emerged due to the difficulties faced while attempting to solve problems without considering real external environment or the entity involved in that problem-solving process [2-3]. Thus, the solutions constructed to address these problems can be limited and inflexible in not properly handling real-world situations. In response, agents can be proposed as problem-solvers that are capable of flexible and effective operation in complex environments. This means that the agent receives input from its environment through sensors, and acts so as to affect that environment through effectors. Such a simple but powerful concept has been adopted by many branches of computing because of its usefulness and broad applicability.

A generally accepted definition of agents is the computational entities that are capable of exhibiting flexible behavior in dynamic and unpredictable environments [2]. Typically, agents are characterized along certain dimensions, rather than defined precisely. For example, a weak notion of agents involves autonomy or the ability to function without intervention, social ability

by which agents interact with other agents, reactiveness allowing agents to perceive and respond to a changing environment, and proactiveness through which agents behave in a goal-directed fashion [3]. As indicated, these characteristics are broadly accepted as representative of the key qualities that can be used to assess agenthood.

Several alternative characterizations with additional properties like mobility, ability to learn and the requirement that agents are rational and honest etcetera have also been mentioned in the literature.

In addition, agents are also required to be based around control architectures comprising mental components such as beliefs, desires, and motivations.

Several efforts have been made to explore the area in some depth, including encompassing agent frameworks [51-53] and agent taxonomies [53], which help in identifying the key features of agent systems and the characteristics of the different branches of the field. In attempting to distinguish agents from programs, Franklin and Graesser constructed an agent taxonomy aimed at identifying the key features of agent systems in relation to different branches of the field [53].

There is now a variety of different labels for agents ranging from the generic autonomous agents, software agents or intelligent agents to the more specific interface agents, virtual agents, information agents, mobile agents and so on [50,54-60]. The diverse range of applications for which agents are being used includes operating system interfaces, processing satellite imaging data, air-traffic control, business process management, electronic commerce, and computer games etcetera [61-66]. The richness of the agent metaphor that leads to such different uses of the term is both a strength and a weakness. Its strength lies in the fact that it can be applied in

many different ways in many situations for many different purposes. The weakness, however, is that the term agent is now used so frequently that it might be considered that there is no commonly accepted concept of an agent.

Etzioni and Weld summarize desirable agent characteristics as including autonomy, temporal continuity by which agents are not simply "one-shot" computations, believable personality in order to facilitate effective interaction, communication ability with other agents or people, adaptability to user preferences, and mobility, which allows agents to be transported across different machines and architectures [67]. They further characterize the first of these, autonomy, as requiring that agents are goal-oriented, collaborative in that they can modify the requests and clarify them, flexible in not having hard, scripted actions, and self-starting in that they can sense changes and decide when to take action.

3.5 Applications of Agents

Areas that find excessive application of agent systems include ambient intelligence and grid computing. Both these areas are related to the concept of controlling a vast and distributed network of power generation and transmission where a large number of generating units, distribution tie lines, transformers, sensors and actuators come together to handle, optimize and ensure unhindered power supply.

3.5.1 Ambient Intelligence

The ambient intelligence vision describes an environment of potentially thousands of embedded and mobile devices (or software artifacts) interacting to support user-centered goals and activity. This suggests a component-oriented view of the world in which the artifacts are independent and distributed. Autonomy, distribution, adaptation, responsiveness, and so on, are

the key characterizing features of these ambient intelligent artifacts, and in this sense they very strongly share the same characteristics as agents [68].

In particular, these ambient intelligence artifacts are likely to be function-specific and interact with numerous other ambient intelligence artifacts in the environment around them in order to achieve their goals. Interactions take place between pairs of artifacts (in one-to-one cooperation or competition), between groups of artifacts (in reaching consensus decisions), and between artifacts and the infrastructure resources that comprise their environments. Interactions like these enable the establishment of electronic institutions or virtual organizations, in which groups of agents come together to form coherent groups able to achieve some overarching goals.

3.5.2 Grid Computing

High-performance computing infrastructure, known as the Grid, for supporting large-scale distributed scientific systems has recently gained interest from several communities. It provides a computing infrastructure for supporting more general applications that involve large-scale information handling, knowledge management, and service provision.

It is natural to view large systems in terms of the services they offer, and consequently in terms of the entities providing or consuming services. Grid applications, in which typically many services may be involved, spread over a geographically distributed environment, which new services join and existing ones leave, very strongly suggest the use of agent-based computing. In particular, agents will need to collaborate and to form coalitions of agents with different capabilities in support of different organizations.

Initially aimed at high-performance computing, Grid computing is now being recognized as the future model for service-oriented environments, within and across enterprises [69].

Chapter # 4

Object Model of Multiagent Based Control of Distributed Power Networks

4.1 Introduction

Object modeling of the proposed multiagent system for control of complex and diverse distributed power system networks has been discussed in this Chapter. The chapter has been divided into two segments. First segment takes up relevant features of requirement engineering in a generic way, culminating into features of object modeling. The second segment comprises of the object model developed for innovative multiagent based control of complex and distributed power system network.

4.2 Requirement Engineering

Software engineers have to often solve immensely complex problems. Understanding nature of the problems can be very difficult, especially if the system is new. Consequently, it is difficult to establish exactly what the system should do. The descriptions of the services and constraints are the requirement for the system, which warrant finding out, analysing, documenting and checking the system [86]. The term requirement is not used throughout the software industry in a consistent way. A requirement is seen as a high-level, abstract statement of a service that the system should provide or a constraint on the system.

Requirements engineering process comprises of different levels of description which are user requirements to mean the high-level abstract requirements, and system requirements, that is, the detailed description of what the system should do. As well as these two levels of detail, a

more detailed description (a software design specification) may be produced to bridge the requirements engineering and design activities.

User requirements are statements, in a natural language plus diagrams, of what services, the system is expected to provide and the constraints under which it must operate. System requirements set out the system services and constraints in detail. The system requirements document, sometimes called a functional specification, should be precise. It may serve as a contract between the system buyer and software developer. A software design specification is an abstract description of the software design which is a basis for more detailed design and implementation. This specification adds further detail to the system requirements specification.

4.3 Object Models

An object-oriented approach to the whole software development is now commonly used, particularly for interactive system development [86-88]. This means expressing the systems requirements using an object model.

Object models developed during requirements analysis may be used to represent both system data and its processing. In this respect, they combine some of the uses of data-flow and semantic data models. They are also useful for showing how entities in the system may be classified and composed of other entities.

An object class is an abstraction over a set of objects which identifies common attributes and the services or operations which are provided by each object. Objects are executable entities with the attributes and services of the object class. Objects are instantiations of the object class and many different objects may be created from a class. Generally, the models are developed using analysis focus on object classes and their relationships.

Models of systems which are developed during requirements analysis should model real-world entities using object classes. Various types of object models can be produced showing how object classes are related to each other, how objects are aggregated to form other objects, how objects interact with other objects and so on. All of these can add to understanding of a system which is being specified.

The analysis process for identifying objects and object classes is recognized as one of the most difficult areas of object-oriented development. Object identification is basically the same for analysis and design.

An object class in UML is represented as a vertically oriented rectangle with three sections. Name of the object class is mentioned in top section, class attributes in the middle section and the operations associated with the object class in the lower section of the rectangle.

4.4 Inheritance Models

Object-oriented modeling involves identifying the classes of object which are important in the domain being studied. These are then organized into taxonomy [86,88]. Taxonomy is a classification scheme showing relation of object class to other classes through common attributes and services.

To display this taxonomy, the classes are organized into an inheritance hierarchy where most general object classes are presented on the top of the hierarchy. More specialized objects inherit their attributes and services. These specialized objects may have their own attributes and services.

4.5 Object Aggregation

In addition to acquiring attributes and services through an inheritance relationship with other objects, some objects are made up of other objects. That is, an object is an aggregate of a

set of other objects. Considering the model described in section 4.8, the hierarchy of main agents for control of distributed networks, namely, Failure Supervision Agent and Main Managements Agent are an example of object aggregation.

Objects communicate by requesting services (calling methods) from other objects and, if necessary, by exchanging the information required for service provision. Copies of information needed to execute the service and the results of service execution are passed as parameters.

In some distributed systems, object communication is implemented directly as mutual text message exchange. The receiving object parses the message, identifies the service and the associated data and carries out the requested service. However, when the objects coexist in the same program, method calls are implemented in the same way as procedure or function calls in a language. When service requests are implemented in this way, communication between objects is synchronous. That is, the calling object waits for the service request to be completed. However, if the objects are implemented as “concurrent processes or threads” the object communication may be asynchronous. The calling object may continue its operation while the requested service is executing.

4.6 Concurrent Objects

Conceptually, an object requests a service from another object by sending a ‘service request’ message to that object. There is no requirement for serial execution where one object waits for completion of a requested service. Consequently, the general model of object interaction allows objects to execute concurrently as parallel processes. These objects may execute on the same computer or as distributed objects on different machines.

In practice, most object-oriented programming languages by default have a serial execution model where requests for object services are implemented in the same way as function calls.

4.7 An Object-Oriented Design Process

The general process used for object-oriented design has a number of stages [86-88]. The first stage is to understand and define the context and the modes of use of the system. After this, design of the system architecture is carried out, followed by identifying principal objects in the system. Then design model is developed and object interfaces are high lighted.

In fact, all of these activities are interleaved and influence each other. Objects are identified and the interfaces fully or partially specified as the architecture of the system is defined. As object models are produced, the individual object definitions may be refined and this may mean changes to the system architecture.

4.8 Object Model for Control of Power Networks

Object model for control of distributed power system networks comprises of ten agents in two strings, namely Failure Supervision Agent and Main Management Agent [24,46]. The hierarchy of each alongwith relevant attributes has been included in Figure 4.1. Inheritance of attributes of Failure Prediction Agent and Occurrence Identification Agent with Failure Supervision Agent can be ascertained from the model. Similarly Status Monitoring, Power Generation, Stability and Reconfiguration Agents inherit attributes from Main Management Agent.

The agents in the object model are shown in Figure 4.1. After a thorough scrutiny of essentially required activities for effective control of distributed power system networks, a comprehensive list of functions was prepared. The problems handled in various research studies and surveyed literature provided an outline of vital activities necessary for enforcing adequate control of a self healing network. Based upon object model fundamentals, these functions were grouped into performance viable classes of objects which formed the basis of devising a multiagent based mechanism. “Identification” attribute associated with each agent describes the component/gadget or sub system for which the data is being handled by it. The information is extracted through a number of sensors located throughout the network and associated with different components, bus bars, transformers, and generators. The decisions are made autonomously and implemented through actuators packaged with sub systems of the network. The system is implemented through micro controller based mechanism.

The agents shown in Figure 4.1 can be divided into two broad categories, the reactive agents and the deliberative agents. The reactive agents perform programmed remedial measures for situations warranting instant response. Their objective is autonomous and hence should be able to implement fast control. The deliberative agents not only represent the system's current status but also modify the plans if they do not match the real world model. The goals of the deliberative agents are robustness and automatic remedial measures. These agents also carry out co-ordination and use heuristic knowledge to identify triggering event from reactive agent that are urgent, important and / or resource consuming.

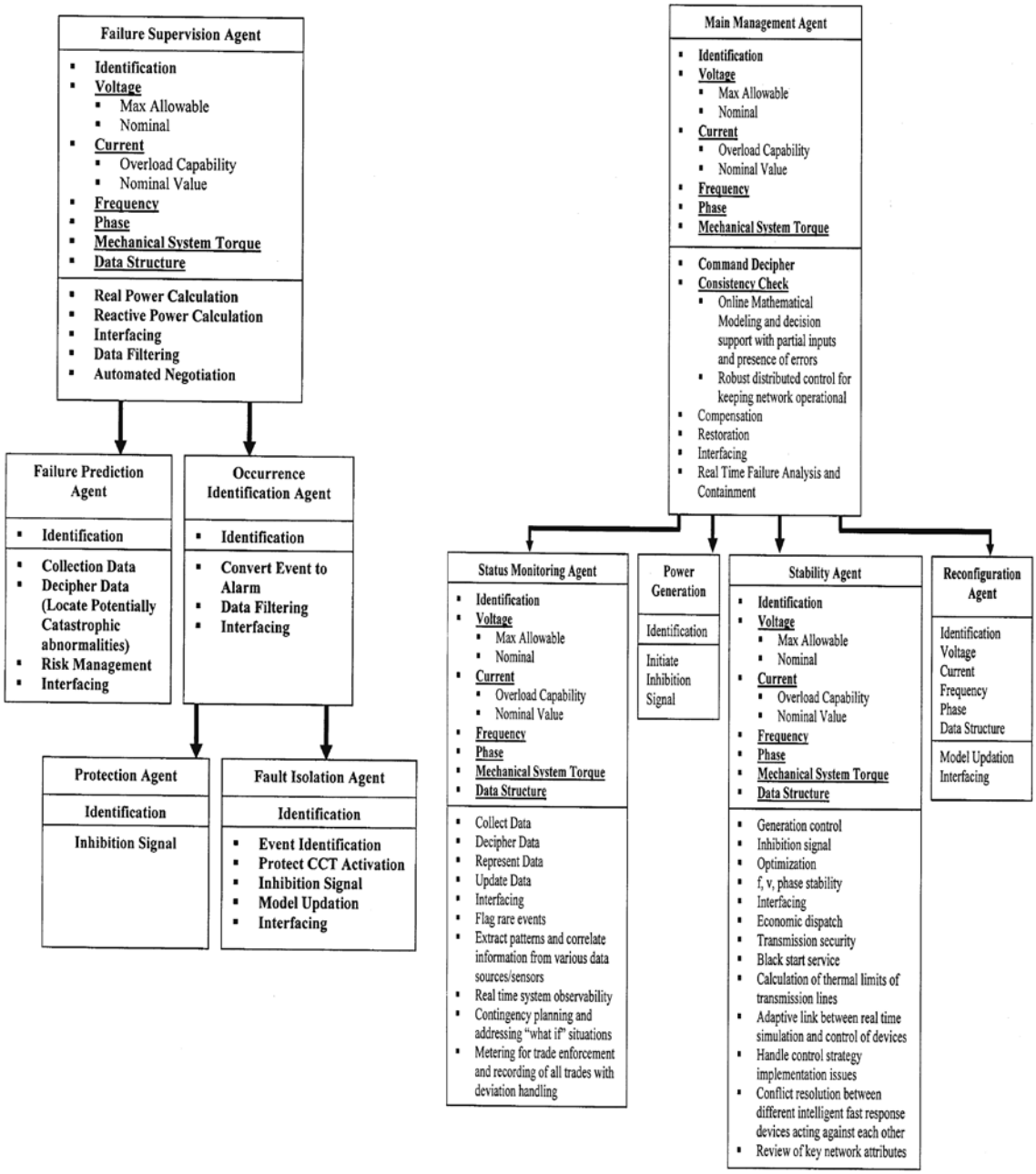

Figure 4.1: Object Model of Agent System

The robustness of the software infrastructure is preserved by using analytically redundant software architecture with two complementary entities (agents) on each job, wherein a simple and highly reliable core agent that guarantees minimal essential services and a complex entity

that provides many desirable features are suggested. The reliable core will function in spite of a failure in the complex agent.

The stochastic events arising from the dynamics of the power network drive the co-ordination between distributed agents. An event driven real time communication architecture will assemble relevant distributed agents into task driven teams and will provide the teams with timely and consistent information to carry out coordinated actions.

The attribute of Identification shown in each object is used to identify the modality (e.g. generator, load, transmission line, or compensator) the agent is associated with.

Interface service of each object provides interface functions for agents to access the knowledge base of each other and also for agent communication (knowledge exchange). This is an essential service and hence should form part of each agent because in a distributed system of greater magnitude and dimensions, communication from one or several agents will be interrupted from time to time. Under these unfavorable conditions, the remaining agents should be robust enough to make decisions that do not jeopardize grid stability or reliability. Fault tolerant agents that can make decisions in the absence of data from one of their adjacent agents are, thus, required. Since the primary role of different agents differs from each other, required data from a particular agent may be missing, but it can be inferred from load flow and voltage information from neighboring agents. This arrangement, therefore, facilitates collaborative fault detection.

A separate agent for compensation can be developed to provide an array of services such as reactive power generation, power flow control, harmonic compensation, voltage regulation, or dynamic control over frequency and voltage. It can, however, also be made part of any other agent with relevant features. In our case it has been mde part of Main Management Agent.

Failure prediction agent will constantly analyze the scenario of functioning so as to have a logical foresight, through look-ahead simulations for predicting failure events in power system network.

One of the major tasks of occurrence identification agent is to convert events into alarm signals to trigger the fault isolation/ relevant remedial sub agents. It will also carry out data filtering and routine interfacing of data.

Since the quantum of data to be handled is expected to be enormous because of continuous real time incoming of parameters, the scrutiny/ filtering of data as per requirement of operation, coordination/ display (if applicable) etc is mandatory [25].

Protection agent and power generation control agent are simple ones, performing as an option in case of failure of complex agents i.e. Fault isolation agent and Stability agent. They are therefore providing redundancy to the system.

Designated softwares such as Failure Supervision and Main Management Agents will read/ compute voltage magnitudes, phase angles and transmission line power flows for a network under steady state operating conditions. Other results such as transformer tap settings and generator reactive power outputs can also be computed. As an analyzing support, high-speed printers should be employed to print out the complete solution in tabular form for analysis by the planning engineer/ R&D setups. Main Management Agent will also carryout consistency checks and decision support with partial inputs and presence of errors. Provision of compensation services, restoration of network and real time failure analysis are also some of its designated tasks.

Stability agent helps the network under disturbance conditions such as sudden loss of a generator/ transmission line, sudden load increase and decrease, short circuits and switching operations. This agent utilizes power flow equations and machine dynamic equations to compute the angular swings of machines during disturbances. The agent also computes critical times for network faults and allows the evolutionary programme/ expert system to determine the effects of various machine parameters, network modifications and control schemes.

Status monitoring agent will carry out real time system observability, contingency planning, flagging of rare events and metering for track enforcement etc. Separate short circuit programmes will form part of Fault Isolation Agent to compute three phase and line to ground faults in power system networks in order to select circuit breakers for fault interruption, select relays to detect faults and control circuit breakers and determine relay settings. Short circuit currents is computed for each relay and circuit breaker location for various system-operating conditions such as lines or generating units out of service, in order to determine minimum and maximum fault currents via Failure Prediction Agent.

4.9 Summary

This chapter presented an object model based on multiagent systems for control of distributed power networks, with a brief description of each agent. Essential features of a typical object model were also deliberated upon.

Chapter # 5

Use-case Model of Multiagent Based Control of Distributed Power Networks

5.1 Introduction

Use-case Model is a dynamic model that describes system interaction with the environment. The approach recommended by UML is to develop a use-case model where each use-case represents an interaction or service in the system [87]. The use-case description helps to identify objects and operations in the system. Use-case model, therefore, is a scenario-based technique for requirement elicitation. It is a fundamental feature of UML notation for describing object oriented system models. Use-case Model of the agent system based control of distributed power networks has been described in this chapter.

5.2 Scenario-based Elicitation

People usually find it easier to relate to real-life examples rather than abstract descriptions. They understand and criticize a scenario of how they might interact with a software system. Scenarios are particularly useful for adding detail to an outline requirements description. These are descriptions of example interaction sessions. Each scenario covers one or a few possible interactions. Different types of scenarios are developed which provide information at different levels of detail about the system.

The scenario starts with an outline of the interaction and, during elicitation, details are added to create a complete description of the interaction. A scenario may include a system state

description at the beginning of the scenario, a description of the normal flow of events in the scenario, a description of undesired eventualities and remedies, information about other concurrent activities and a description of the state of the system after completion of the scenario.

Scenario-based elicitation is carried out informally where the “Requirements Engineer” works with stakeholders to identify scenarios and to capture details of these scenarios. Alternatively, a more structured approach of event scenarios or use-cases may be used.

5.3 Use-cases

Use-cases are a scenario-based technique for requirements elicitation. It was first introduced in the Objectory method [86-88]. These have now become a fundamental feature of UML notation for describing object-oriented system models. A use-case identifies the actors involved in an interaction and names the type of interaction.

5.4 System Context and Models of Use

It is essential to develop an understanding of the relationships between the software being conceived and its external environment. Developing this understanding helps to decide methods for providing the required system functionality and ways to structure the system so that it can communicate effectively with its environment.

The system context and system use model represent two complementary models of the relationships between a system and its environment. System context is a static model that describes other systems in the environment and the system use model is a dynamic model that describes its interaction with environment. A standard representation of use-case model is shown in Figure 5.1 [86].

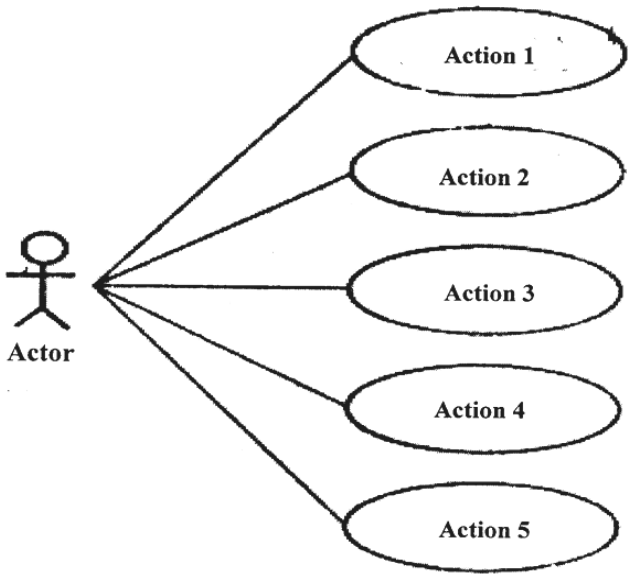

Figure 5.1: Standard Representation of Use-cases

The context model of a system is represented using associations where, essentially, a block diagram of the overall system architecture is produced. This is expanded by representing a sub-system model using UML packages shown in Figure 5.2 [86].

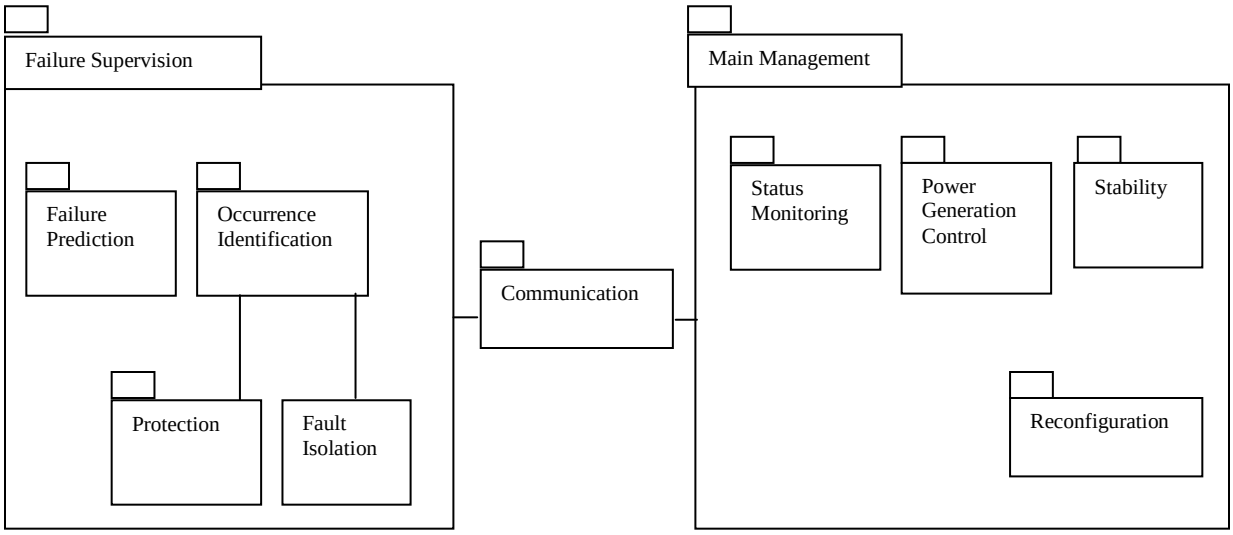

Figure 5.2: Agent System/ Sub Systems

The modeling of interactions of a system with its environment should use an abstract approach that does not include too much detail of these interactions. The approach proposed in UML is to develop a use-case model where each use-case represents an interaction with the system. In use-case models every possible interaction is named in an ellipse and the external entity involved in the interaction is represented by a stick figure.

Any technique for describing use-cases is valid as long as the description is short and easily understandable. It is necessary to develop descriptions for each use-case shown in the model. The use-case description helps to identify objects and operations in the system.

5.5 Object Identification

In practice object identification is actually concerned with identifying object classes. The design is described in terms of these classes. Inevitably, the object classes initially identified are to be refined and revisited as deeper understanding of the design process is developed [86-88].

Many proposals have been made to identify object classes. Grammatical analysis of a natural language description of a system is used for this purpose. In this case objects and attributes are nouns and operations or services are verb. This approach has been used in the HOOD method for object-oriented design widely used in European aerospace industry [87]. Another possibility is to use tangible entities (things) in the application domain such as aircraft, roles such as manager, events such as request, interactions such as meetings, locations such as offices and organizational units such as companies etc. Or yet another approach is to use a behavioural approach in which the designer first understands overall behaviour of the system. Various behaviours are assigned to different parts of the system and an understanding of actors initiating and participating in these behaviours is derived. Participants who play significant roles are recognized as objects.

Finally a scenario-based analysis is used where various scenarios of system use are identified and analysed in turn. As each scenario is analysed, the team responsible for the analysis must identify the required objects, attributes and operations. A method of analysis where analysis and designers take on the role of objects is effective in supporting this scenario-based approach.

In practice, different sources of knowledge are used to discover objects and object classes. Objects and operations initially identified from the informal system description are a starting point for the design. Further information from the application domain knowledge or scenario analysis is then used to refine and extend the initial objects. This information is collected from requirements documents, discussions with users and from an analysis of existing system.

5.6 Use-cases of Agents

Layout of sub systems proposed in Figure 5.2 has been considered for proceeding onwards. Use-cases of each agent are described in this section [24,25]. Each use-case has been described under headings of Data, Stimulus and Response. Data refers to the handling of data by a particular use-case, stimulus is the event when a particular use-case has to perform and response is the expected outcome of that use-case.

5.6.1 Failure Supervision Agent (Actor)

It's identified use-cases are Collect Data, Real Power Calculation, Reactive Power Calculation, Interfacing, Automated Negotiation and Data Filtering/ Manipulation. These use-cases are briefly described in this section, following the prescribed pattern and stylized form of description [86].

5.6.1.1 Data Collection

This use-case collects real time data at defined intervals, remains active throughout and provides information to the requesting agent. Description of the use-case is shown in Table 5.1.

Table: 5.1 Data Collection

Use-case	Collect Data
Data	It collects real time data of voltage, frequency, current, and phase etc from sensors/instruments of subsystems in each data collection interval which keeps on updating itself, at a twenty second interval. The data handled also includes maximum allowable and nominal voltage, current overload capability with nominal value and mechanical system torque.
Stimulus	It is active all the time to receive digital data from I/O card and Modems through sensors.
Response	It provides data to various agents/ Actors.

5.6.1.2 Real Power Calculation

This use-case is meant for calculation of instantaneous real value of power at different nodes of distributed network. Description as per format has been included in Table 5.2.

Table: 5.2 Real Power Calculation

Use-case	Real Power Calculation
Data	For a polyphase circuit of M Phases, each ones instantaneous power is calculated as: $p(t) = v(t)\, i(t)$ and the total instantaneous power is the sum of active powers of each individual phase: $p(t) = \Sigma\, v_i(t)\ i_i(t) \quad i=1 \text{ to } M$
Stimulus	It calculates active power in real time and is available for use by another actor or service on request.
Response	Calculated active power is sent on receipt of request.

5.6.1.3 Reactive Power Calculation

It calculates Volt-Ampere Reactive for different nodes of the network. This information is primarily used for management of ancillary services and compensation. Relevant aspects of use-case are given in Table 5.3.

Table: 5.3 Reactive Power Calculation

Use-case	Reactive Power Calculation
Data	Average power 'P' over a cycle is calculated. Then rms values of voltage and current are calculated to find apparent power 'S'. Reactive power is then calculated as under: $Q = \sqrt{(S^2 - P^2)}$
Stimulus	Real time values are available on request.
Response	Values are sent on receipt of request.

5.6.1.4 Interfacing for Data Structures

This use-case is mainly responsible for ensuring compatibility of data structures of transferred information in a distributed network of hybrid nature. Stylised form of use-case is shown in Table 5.4.

Table: 5.4 Interfacing for Data Structures

Use-case	Interfacing
Data	Instantaneous basic parameters and the calculated values of active/ non active power in the form of data structure mutually agreed upon and acceptable to other agents/ actors and services are handled.
Stimulus	Request from other systems serves as stimulus.
Response	The desired data is sent in the form (data structure) acceptable to the requesting agent.

5.6.1.5 Automated Negotiation

This use-case either manages provision of reactive power at nodes with adequate reactive power, or else the request for reconfiguration of networks is sent. Relevant description is included in Table 5.5.

Table: 5.5 Automated Negotiation

Use-case	Automated Negotiation
Data	Real time data from sensors/ calculated values from other use-cases are obtained by sending requests on all identified nodes on the system network.
Stimulus	Request to reconfigure/ supply various quantities at nodes in need of ancillary services prior to arriving at serious situations is sent.

Response	Response is provision of ancillary services or approaching relevant agent for reconfiguration.

5.6.1.6 Data Filtering/ Manipulation

This use-case handles collection, updation and analysis of data with source traceability. The use-case has been described in Table 5.6.

Table: 5.6 Data Filtering/ Manipulation

Use-case	Data Filtering/ Manipulation
Data	All real time parameters are handled with source tag.
Stimulus	Request for deciphering, analyzing, integrating, generating, gathering, representing, updating and elimination of data acts as stimulus.
Response	Provision of the desired results of the processed data/ updating is the response.

5.6.2 Failure Prediction Agent

Its proposed use-cases are Collect Data, Risk Management and Interfacing. The description of the use-case not defined earlier is “Risk Management”. This use-case handles the scenario developed after the outcome of automated negotiations with support from evolutionary programme and local expert system. Description as per format is included in Table 5.7.

Table: 5.7 Risk Management

Use-case	Risk Management
Data	Results of Automated Negotiation service of Failure Supervision Agents are received as data to be handled by this use-case.
Stimulus	Request from Failure Supervision Agents services to manage the situation in a given scenario serves as stimulus.
Response	Response is to handle the new context and update the evolutionary process of genetic algorithms in collaboration with local expert system advice.

5.6.3 Occurrence Identification Agent

The proposed use-cases are Convert Event to Alarm, Data Filtering and Interfacing.

The description of new use-case with title, “Covert Event to Alarm” will be given now. This use-case is responsible for identifying situation codes generated by different agents

and subsequently generate code of the best response for action by acuators, in addition to initiation of alert signals for crew, as described in Table 5.8.

Table: 5.8 Conversion of Event to Alarm

Use-case	Convert Event to Alarm
Data	Coded identification for recognition of a particular event (disturbance or malfunctioning) and code of identification for the best response as per evolutionary process of genetic algorithms in collaboration with local expert system is handled as data.
Stimulus	Signal from Risk Management and Automated Negotiation regarding occurrence of a particular event/ events is regarded as stimulus.
Response	It generates signal from/for relevant actuators for remedial measures. It also initiates visual/ audio indication to maintenance crew, specific to nature of event for ease of identification by the crew.

5.6.4 Fault Isolation Agent

The use-cases are Event Identification, Circuit Protection, Inhibition Signal Generation, Model Updating and Interfacing. The description of these use-cases is provided next.

5.6.4.1 Circuit Protection

This use-case is assigned the task of identifying from pre-defined events as per code generated by different use-cases and activate necessary circuit protection mechanisms accordingly, as per information contained in Table 5.9.

Table: 5.9 Circuit Protection

Use-case	Circuit Protection
Data	Data code for identifying particular event/ events.
Stimulus	It activates on receipt of signal/ data from Event Identification and Convert Event to Alarm.
Response	Activation of relevant circuit protection mechanism is the response.

5.6.4.2 Fault Isolation Inhibition Signal Generation

If circuit protection mechanism fails to prove effective, it invokes this use-case for emergency shutdown of identified portion of network and facilitates passage of this information to various agents. The use-case has been described in Table 5.10.

Table: 5.10 Fault Isolation Inhibition Signal Generation

Use-case	Inhibition Signal Generation
Data	It deals with data code meant for circuit stoppage
Stimulus	Request for circuit protection regarding emergency shut off of a part or whole of the identified circuit as per coded data is the stimulus.
Response	Response is emergency shut off and passage of information to this effect to Model Updating service, Stability Agent and Reconfiguration Agent.

5.6.4.3 Model Updating

Information regarding inhibition or resumption of circuit is used by this use-case for constant and iterative updation of network model, as described in Table 5.11.

Table: 5.11 Model Updating of Fault Isolation

Use-case	Model Updating
Data	Information code regarding shut off or resumption of whole or part of a circuit is the relevant data for this use-case.
Stimulus	Intimation from Inhibition Signal Generator regarding shut off or resumption acts as stimulus.
Response	It carries out model updating and informs reconfiguration agent accordingly.

5.6.5 Protection Agent

It provides redundancy/ duality to Fault Isolation Agent and comprises of a simple algorithm to handle emergency possessing the singular use-case of circuit inhibition, as shown in Table 5.12.

Table: 5.12 Initiate Inhibition Signal for Protection Agent

Use-case	Initiate Inhibition Signal

Data	Data handled by this is the output of occurrence Identification Agent.
Stimulus	It stimulates on receipt of signal regarding failure of Fault Isolation Agent and handing over the control to agent providing redundancy.
Response	Its response is to shut off indicated malfunctioning circuit in emergency.

5.6.6 Main Management Agent

The suggested use-cases are Command Decipher, Consistency Check, Compensation, Restoration and Interfacing. These will be described in next paragraphs.

5.6.6.1 Command Decipher

This use-case receives input from agents of Failure Supervision and assesses effect on network by identifying potentially calostrophic abnormalities, as shown in Table 5.13.

Table: 5.13 Command Decipher

Use-case	Command Decipher
Data	It handles filtered Data from Failure Supervision Agent.
Stimulus	It is stimulated on receipt of data from Failure Supervision Agent Hierarchy that includes all agents of the hierarchy.
Response	It is expected to establish/ assess effect of command on network.

5.6.6.2 Consistency Check

After updating/resumption of network configuration, various built in tests are conducted by this use-case for assurance of expected performance. The use-case has been described in Table 5.14.

Table: 5.14 Consistency Check

Use-case	Consistency Check
Data	Details of recent change in the model/ configuration of the system network from any of the relevant use-cases is the received data.
Stimulus	This receipt of information regarding system modification serves as stimulus for the use-case.

Response	It conducts the built in tests so as to ensure ability of the modified setup to meet the designated objectives of optimum power supply within specified constraints. Results are sent to Fault Supervision Agent, occurrence identification agent and reconfiguration agent.

5.6.6.3 Provision of Compensation

Request for ancillary services as initiated by any node on network through Stability or Fault Supervision Agents is entertained by this use-case, as shown in Table 5.15.

Table: 5.15 Provision of Compensation

Use-case	Compensation
Data	Phase angles, voltages and reactance are the handled data.
Stimulus	Request for ancillary services provision by optimization service of Stability Agent / Fault Supervision Agent serves as stimulus.
Response	Reactive power supply, harmonic compensation and dynamic control over frequency, voltage and power flow are the expected responses from use-case of compensation.

5.6.6.4 Restoration of Network

When the networks returns back to normal operation, all reconfigurations are to be set aside for optimum performance. This use-case implements restoration through actuators, as shown in Table 5.16.

Table: 5.16 Restoration of Network

Use-case	Restoration
Data	Data from Fault Isolation Agent or Reconfiguration agent is received.
Stimulus	Information regarding normalization of performance parameters of the inhibited circuit serves as stimulus.
Response	It initiates signals to actuators for reverting back to a setup closer to original system configuration as far as possible.

5.6.7 Stability Agent

Its Use-cases are Generation Control, Inhibition Signal Generation, Optimization, f, v,Φ Stability and Interfacing

5.6.7.1 Generation Control

This use-case is assigned the prime responsibility of exercising control over generation under disturbance conditions, in collaboration with a number of use-cases of different agents, as described in Table 5.17.

Table: 5.17 Generation Control

Use-case	Generation Control
Data	Variables of power flow equations and machine dynamics equations are the parameters required by this use-case.
Stimulus	Disturbance conditions such as loss of generator/ transmission line, load fluctuations, short circuits, switching operations act as stimulus for it.
Response	Its response is calculation of new machine parameters such as angular swings and initiation of instructions to actuators for implementation of remedial measures in collaboration with Failure Supervision, Reconfiguration Agents and local expert system advice as per evolutionary process.

5.6.7.2 Optimization for Stability

Optimization of cost of production, power transfer and optimized network configuration is done by this use-case. Relevant description is included in Table 5.18.

Table: 5.18 Optimization for Stability

Use-case	Optimization
Data	All real time data as required for calculations to achieve optimization is handled by it.
Stimulus	Being an important use-case, it conducts optimization interactions all the time.
Response	It carries out cost optimization and formulation of optimized network updation proposals for maximum power transfer. Initiation of signals for shutting down or resumption of generation (full or partial) as per load flow and power requirements of the load is also done by it to maintain stability.

5.6.7.3 f, v,Φ Stability

This use-case compares the real time data of parameters of power flow equations etc with reference/nominal values and on identifying deviations it resorts to bring the values back within limits, as shown in Table 5.19.

Table: 5.19 f, v,Φ Stability

Use-case	f, v,Φ Stability
Data	Data handled by this use-case includes variables of power flow equations and machine dynamics equations, and error signals after comparison with reference values.
Stimulus	It does not require stimulus and remains active throughout.
Response	On exceeding the error signals from reference value, it resorts to respective control mechanisms as adapted.

5.6.7.4 Inhibition Signal Generation for Stability

From point of view of stability, at times it becomes necessary to shut off various portions of network in emergency. This use-case implements this requirement on receipt of information from automated negotiation. Use-case has been described in Table 5.20.

Table: 5.20 Inhibition Signal Generation for Stability

Use-case	Inhibition Signal Generation
Data	Output of Occurrence Identification Agent.
Stimulus	Situation code forwarded by Automated Negotiation serves as stimulus for this use-case.
Response	It generates circuit inhibition signal for emergency protection or circuit inhibition and forwards relevant information to generation control for calculation of new parameters.

5.6.8 Reconfiguration Agent

This use-case is designated the responsibility of physical implementation of updated configuration of network on request from Fault Isolation Agent or Power Generation

Control. The use-cases are Model Updation and Interfacing. Model Updation use-case has been described in Table 5.21

Table: 5.21 Model Updation of Reconfiguration Agent

Use-case	Model Updation
Data	Active and inhibited parts of the network are identified.
Stimulus	Inhibition Signal generated by Fault Isolation Agent or Power Generation Control Agent act as stimulus for this use-case.
Response	It instructs actuators/ circuit breakers for reconfiguration of system and issues instructions to Restoration Service of Main Planning Agent

5.6.9 Power Generation Control Agent

It provides redundancy to Stability Agent and assumes control only on failure of complex algorithms of Stability Agent. Its only use-case is Initiate Inhibition Signal, as described in Table 5.22.

Table: 5.22 Initiate Inhibition Signal of Power Generation Control

Use-case	Initiate Inhibition Signal.
Data	Output of Occurrence Identification Agent intimates failure of main agent.
Stimulus	Receipt of signal regarding failure of Stability Agent, therefore, acts as stimulus for this use-case.
Response	It shuts off the indicated malfunctioning circuit in emergency as response.

5.7 Summary

Important use-cases of the agent system proposed for control of complex and distributed power system methods have been described. Stimulus, responses and data handling for each use-case has been included. A layout of sub systems has also been proposed. Besides, few essential aspects of use-case models and their utilization have also been described. These use-cases will form basis for development of state machine model of the proposed multiagent system for control of distributed power network presented in Chapter 6.

Chapter # 6

State Machine Model of Multiagent Based Power Networks

6.1 Introduction

A system is best understood by first examining its static structure, that is, the structure of its objects and their relationships to each other at a single moment in time. Then changes to the objects and their relationships over time are examined. Aspects of a system that are time dependent are visible in the dynamic model only. Control of a system describes the sequences of operations that occur in response to external stimuli.

This chapter describes flow of control, interactions, and sequencing of operations in a system of concurrently-active objects. The major dynamic modeling concepts are events, representing external stimuli, and states, representing values of objects. A detailed state machine diagram of the proposed system showing events and states is also included in this chapter.

6.2 Real-Time System Modeling

Real-time systems are required to respond to events occurring at regular or irregular intervals [87]. These events (or stimuli) may cause the system to move to a different state. For this reason, state machine modeling, may be used to describe a real-time system.

A state model assumes that, at any time, the system is in one of a number of possible states. When a stimulus is received, this may cause a transition to another state. A system controlling a valve may move from a state ‘Valve open’ to a state ‘Valve closed’ when an operator command (the stimulus) is received.

The rounded rectangles represent system states and the arrowed labels represent stimuli which force a transition from one state to another. The names chosen in the state machine diagram are descriptive and the associated information indicates actions taken by the system actuators or information that is displayed. Operation of the system is traced by reading the model in the direction of arrows.

State machine models are a good, language-independent way of representing the design of a real-time system. For this reason, they are an integral part of real-time system design methods.

6.3 Events and States

In Chapter 4, the object model described possible patterns of objects, attributes, and links that exist in a system. The attribute values and links held by an object are treated as state. The objects stimulate each other, resulting in a series of changes to their states. An individual stimulus from one object to another is an event. The response to an event depends on the state of the object receiving it,and can include a change of state or the sending of another event to the original sender or to a third object. Pattern of events, states, and state transitions for a given class are represented as a state diagram [86]. A state diagram is a network of states and events, just as an object diagram is a network of classes and relationships. The dynamic model consists of state diagrams, with important dynamic behavior, and shows the pattern of activity for an entire system. Each state machine executes concurrently and can change state independently. The state diagrams for various classes combine into a single dynamic model via shared events.

6.4 Controlling Operations

A behavioral description of an object must specify what the object does in response to events. Operations attached to states or transitions are performed in response to the corresponding states or events.

An activity is an operation that takes time to complete. An activity is associated with a state. Activities include continuous operations, such as displaying a picture on a television screen, as well as sequential operations that terminate by themselves after an interval of time, such as closing a valve or performing a computation. A state may control a continuous activity, (such as raising an alarm indicating a malfunction) that persists until an event terminates it by causing a transition from the state. The notation “do: A” within a state box indicates that activity A starts on entry to the state and stops on exit. A state may also control sequential activity, such as moving an actuator, which progresses until it completes or until it is interrupted by an event that terminates it prematurely. The same notation “do: A” indicates that sequential activity A begins on entry to the state and stops when complete. If an event causes a transition from the state before the activity is complete, then the activity is terminated prematurely.

6.5 State Machine Model of the Proposed System

Behavioural models are used to describe overall behaviour of the system [24,25]. These include data flow models which model the data processing in the system and the state machine models which model the system reactions to events but a strict demarcation between these models is seldom possible because of an ever existing overlap.

Keeping the above in view, state machine diagram highlighting major events being handled by various agents, and the offered services depicted as states pertaining to multiagent system of software for control of distributed power networks is shown in Figure 6.1. Since the diagram contains two super states also, it will be pertinent to describe these in some detail.

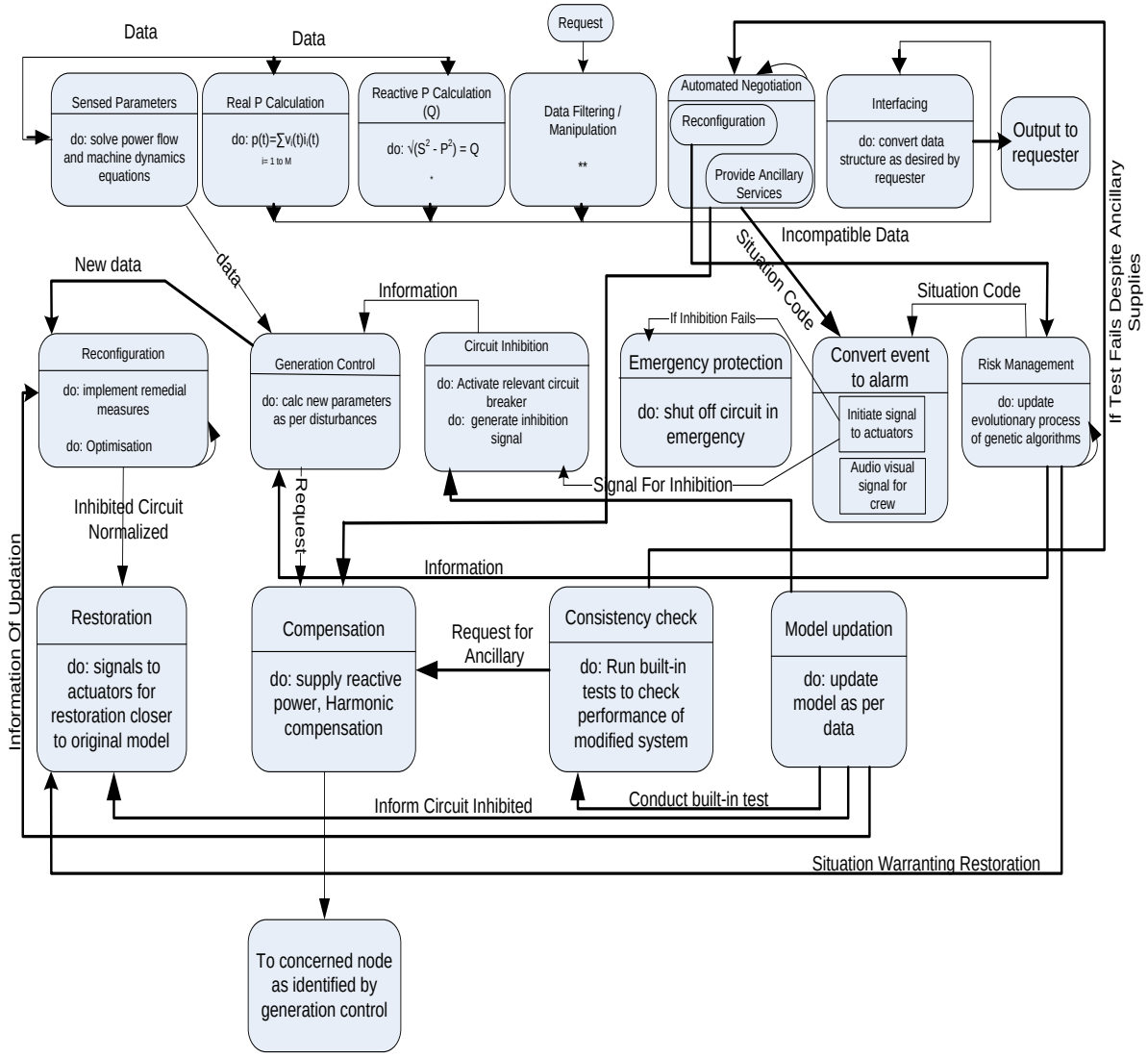

Table: 6.1 State Machine Model

6.5.1 Super State of Reactive Power Calculation

Super state of Reactive Power calculation of Figure 6.1 is shown in Figure 6.2 with allied sub states [25]. Average power of one cycle of the waveform is calculated by integrating the power contained in cycle between 0 to 2π using the formula:

$$P = \frac{1}{T}\int_{0}^{2\pi} v(t)i(t)dt \qquad 6.1$$

After calculating value of 'P' from equation 6.1, the next step is to calculate RMS value of current and RMS value of voltage using the formula:

$$I_{RMS} = \sqrt{\frac{1}{T_2 - T_1}\int_{T_1}^{T_2} \left[I_P Sin(wt)^2\right] dt} \qquad 6.2$$

$$V_{RMS} = \frac{V_P}{\sqrt{2}} \qquad 6.3$$

Apparent power denoted by 'S' is then calculated using the formula:

$$S = I^2 Z \qquad 6.4$$

or

$$S = \frac{E^2}{Z} \qquad 6.5$$

Finally the reactive power is calculated through expression

$$Q = \sqrt{(S^2 - P^2)} \qquad 6.6$$

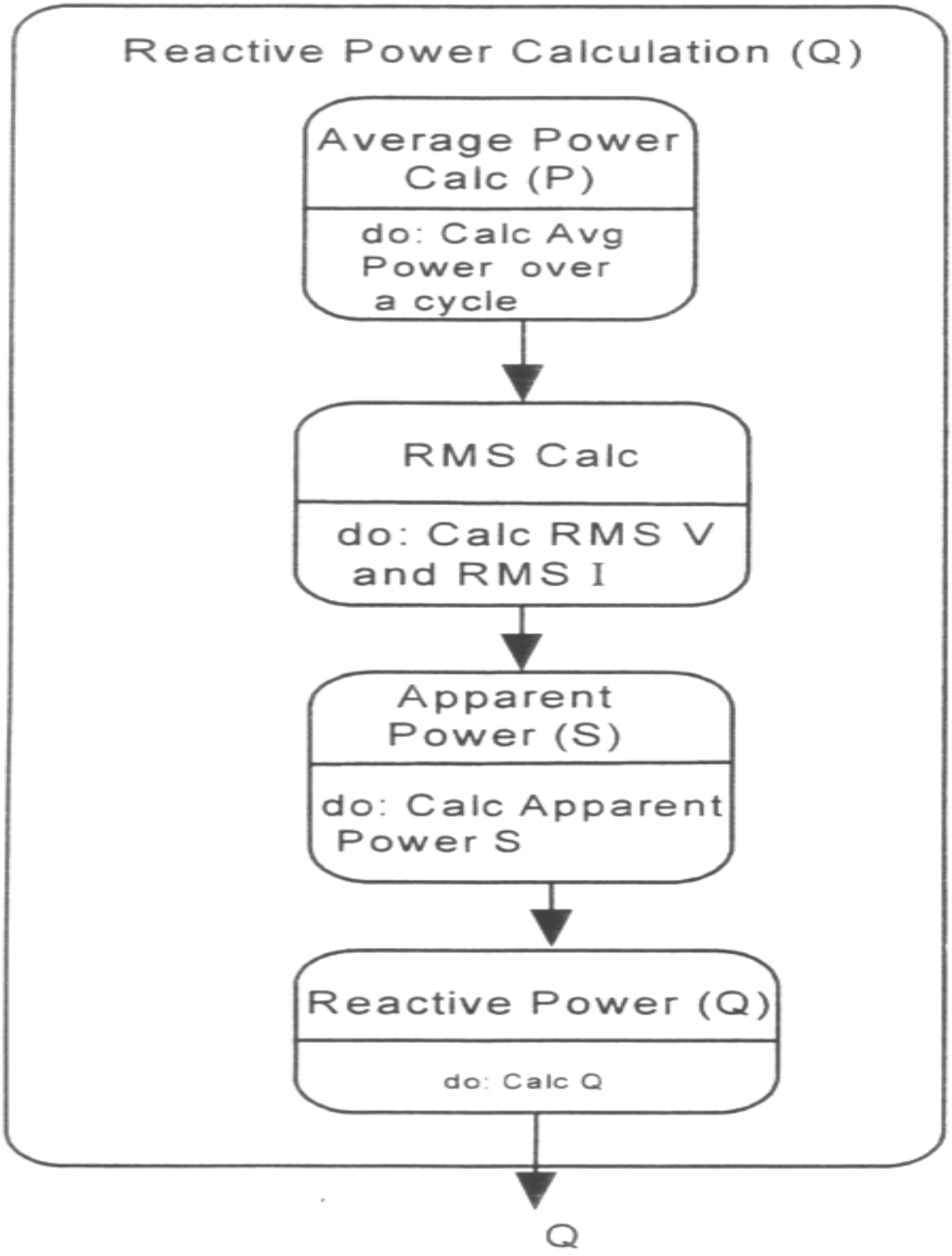

Figure 6.2: Reactive Power Calculation

6.5.2 Super State of Data Filtering and Manipulation

Super State of Data Filtering and Manipulation has been described in substate model of Figure 6.3. All real time / interactive systems have to provide some way of presenting information to sub systems, other systems or users. The data is to be generated in the desired pattern, deciphered from codes, integrated with different sources of information analysed in a logical fashion, and finally represented, discarding the data which has outlived its utility.

The information presentation may simply be a direct representation of input information as a text file, or it may present it graphically. Information presentation should be kept separate from information itself. It is not true that the designer always knows the best way to present the data, and hence object structures should not be “hard-wire” presentation operations. By separating the data presentation system from data, the representations can be changed without having to change the underlying computational system. When the object state changes, all places where this data is involved are automatically notified and updated to reflect the change. Since the Failure Prediction Agent (object that maintains data information) does not always know specific data structure of the requesting object, interfacing of data for making it compatible with data structure of the requester is required. The output of super state of data filtering and manipulation therefore, goes to state of interfacing and then to the requester as per the desired format.

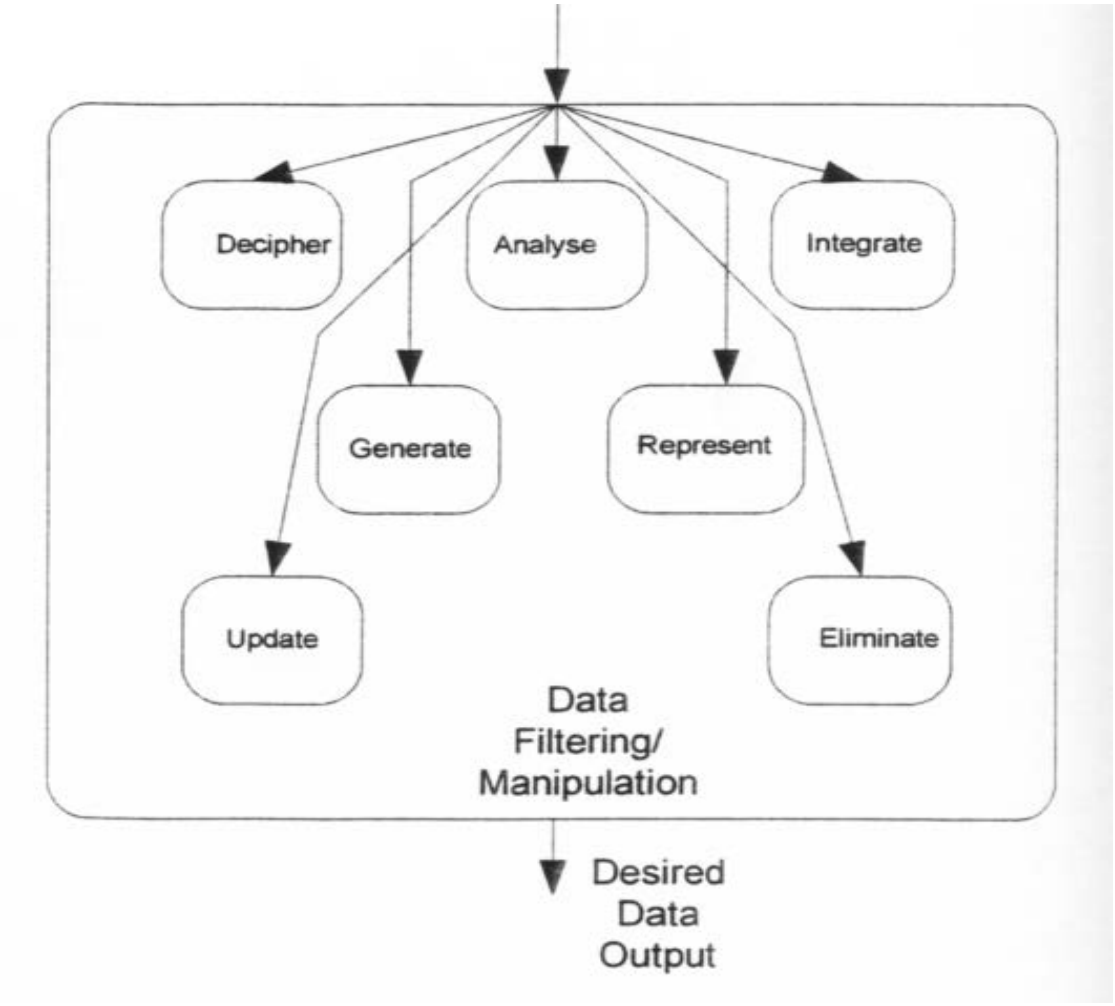

Figure 6.3: Data Filtering/ Manipulation

6.5.3 Description of State Machine Model

State machine model depicted in Figure 6.1 suggests that there are a number of events, activities/actions, states and transitions involved in each scenario, or sequence of events. Iterations of cycles commencing at each state concurrently or otherwise, will continue till natural or forced termination/interruption is implemented. Selected sequences will be described here for convenience of understanding a rather complex diagram. The diagram, however, has to be consulted in conjunction with object and use-case models already described in preceding chapters.

Processed data obtained from sensors is utilized for solving power flow and machine dynamic equations and real and reactive power etc. If an instantaneous value of various parameters is required at an identified node, bus, generator or transformer etc the request will be forwarded to data filtering/manipulation service. The requisite information with time tag will be provided to interfacing service available to all the objects, where the data structure will be made compatible for ready consumption of the requester, keeping in view the diverse nature of objects and machines forming part of the network.

6.5.3.1 Description of Main Loop of State Machine Model

Looking at the diagram for main loop reveals that, as a consequence of conducting a built-in test for performance and consistency check of a modified system, the result is sent to automated negotiation for considering reconfiguration, or provision of ancillary services. A situation code commensurate with the arising situation will be directed towards “Convert Event to Alarm” where audio visual signal for crew will be initiated. Concurrently the identified actuators will be mobilized for remedial action if required. An inhibition signal will be sent to circuit inhibition service to activate relevant circuit breaker. Information regarding inhibited circuit will be dispatched to generation control where new parameters will be calculated in consonance with the observed disturbance. New data will be sent to circuit inhibition service to activate relevant circuit breaker. This data will be sent to reconfiguration service to implement remedial measures and launch algorithms for fresh optimization. On the other hand the data will be sent to Risk Management for updating evolutionary process of genetic algorithms. The output of Risk Management will be available to generation control and circuit restoration. When situation normalizes, the restoration will signal the actuator for restoration of network as close to original model as possible. Outputs of consistency check and automated negotiation will

supervise provision of ancillary services and compensation etc to concerned node in the complex network as identified by generation control.

6.5.3.2 Loop for Provision of Compensation

Request for provision of compensation or ancillary services will be initiated from either automated negotiation, generation control, or consistency check. The provision of services will however only be from compensation state. Automated negotiation, risk management and reconfiguration states remain active throughout and keep on performing the assigned tasks iteratively, as shown by arrows initiating and terminating on themselves in Figure 6.1.

6.5.3.3 Loop for Generation Control

Generation control commences operation on receipt of information from Risk Management or Circuit Inhibition. However, it can be invoked directly, as a result of abnormal parameters of power flow or machine dynamics equations. Each time the generation control comes in operation, new parameters are calculated and this information is passed on to reconfiguration state. Reconfiguration state, then verifies the information as received from model updation and implements remedial measures as necessary. If the inhibited signal has ceased, circuit restoration is ordered which is implemented through actuators. Restoration state however always consults the results of latest iterations of genetic algorithms/evolutionary programmes before making the decision.

6.5.3.4 Failure of Ancillary Services

If despite provision of ancillary services, built in tests conducted by consistency check still generate unsatisfactory results, this information is passed on to automated negotiation. Here the reconfiguration is again considered and proposal is forwarded to Risk Management.

The evolutionary process is again provided with new data and the best response to new situation is worked out. This response is sent to generation control. At this moment, the latest information from convert event to alarm, initiation of signals to actuators and circuit inhibition details are again considered and new data is transferred to reconfiguration state for implementation and optimization. The situation continues till Risk Management generates restoration signal to reconfiguration. After this circuit restoration close to original configuration is ordered and implemented through actuators at restoration state.

6.6 Summary

The dynamic model represents control information; the sequences of events, states, and operations that occur within a system of objects. Like the object model, the dynamic model is a pattern that specifies the allowable scenarios that may occur. The notation for the dynamic model represents a compromise between simplicity and expressiveness; there are some meaningful constraints that cannot be expressed by the notations. The state machine model of the proposed system has been covered in this chapter which should be read in conjunction with object and use-case models of the distributed power system networks.

Chapter # 7

Simulated System and Results

7.1 Introduction

This chapter handles the details of simulation of a power system in matlab/ simulink for ascertaining performance under increasing load conditions when controlled by a central control or no control and when controlled locally. Remedial actions taken by local control by being proactive to avert overloading of the generators has also been demonstrated. The details of load flow calculations used for the simulation adapting Newton Raphson Method for finding out bus voltages, phase angles, active power, and reactive power at each instant during various contingencies and normal operation have also been included. Similarly the details of swing equations used for finding rotor angle and speed of generators at each instant have also been described. At the end the results of the simulations have been described under various load conditions experienced by the selected system.

7.2 System for Simulation

With a view to validate the efficacy and advantages of distributed control a three bus system as shown in figure 7.1 has been selected and simulated in matlab/ simulink. The system comprises of synchronous machines employed as generators, transformers and shunt capacitance for maintaining voltage of 1 pu at the bus.

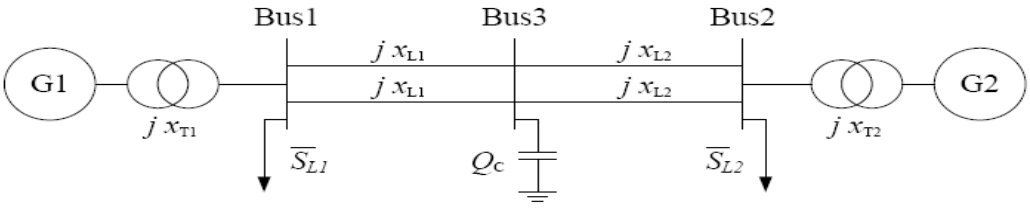

Figure 7.1: System for Simulation

Implementation of generators and buses is shown in figure 7.2 and that of lines and loads has been displayed in figure 7.3. carried out in Matlab.

```
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Bus Data ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
mp=2   %Static load model coefficients
mq=2
NA=0;
Qc=0; % shunt reactive power for compensation intially zero
PM1=1.5
PM2=0
PM3=0
BUSDATA=[
% 1 2 3 4 5 6 7 8 9 10
% Node Type Pgen Qgen Pload Qload YL Ysh  V  Angle
   1    2    1.5   NA  0.5    0   0   0  1.0  0
   2    1     0    0   2.0    0   0   0  1.0  0
   3    3     0    0    0     Qc  0   0  1.0  0];
% Type=1 means slack-bus, Type=2 means PU-bus, Type=3 means PQ-bus
```

Figure 7.2: Bus and Generator Data of System

```
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Line Data ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
XL2=0.4;
XL1=XL2+(1.5/250);
LINEDATA = [
% Line from    to      R       X       B
   1    1      3       0       XL1/2   0 %Equivalent impedance
   2    2      3       0       XL2/2   0]; %Equivalent impedance
Gbus=find(BUSDATA(:,2)==1 | BUSDATA(:,2)==2); %generator buses
ng=length(Gbus); %number of generator buses
nbus=length(BUSDATA(:,1)); %number of buses
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Reactances ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Xd1_prim=0.15; %generator 1 internal reactance
Xd2_prim=0.15; %generator 2 internal reactance

Xt1=0.1; %transformer 1 reactance
Xt2=Xt1; %transformer 2 reactance
%~~~~~~~~~~~~~~~~~~~~~~~~~~~~ Loads ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
SL1=0.5+0j;
SL2=2+0j;
```

Figure 7.3 Line and Load Data of Selected System

7.2.1 Rotor Angle Stability

It refers to the ability of synchronous machines of an interconnected power system to remain in synchronism after being subjected to a disturbance. Instability that may result occurs in the form of increasing angular swings and hence increasing difference of rotor angles between the two or more generators.

When the two generators stay in synchronism with each other the system remains stable and ready to sustain any disturbance faced at later stage. If any of the generators goes out of synchronism, oscillations start which result in cascaded failures which keep on increasing till disseminated in entire power system. To avert the situation it is best to make arrangements to keep the generators in synchronism at all times by reduction in load, elimination of fault or increasing loadability of the system. The load flow analysis and swing equation parameters provide values of each bus voltage along with angle, active and reactive power and speed as well

as rotor angle with respect to dq axis. If difference of rotor angles in dq frame is converging with respect to time, the system does not go out of synchronism. If the value of difference of angles between generators keeps on increasing, system is bound to go out of synchronism [90]. The chances of system going out of synchronism are increased on sudden enhancement of load on any of the buses. A prompt remedial action taken on immediately sensing the problem saves the system from loosing synchronism. At times a proactive approach by sensing the problem before it actually happens saves many contingencies and normalized performance of power systems is assured. The decision to introduce series compensation to increase loadability of the lines by reducing overall line reactance can be taken [90]. It may also prove prudent to render the line with fault or enhanced load out of circuit before the circuit breaker nominal values are surpassed with a view to save the system from a probable collapse. In any case the reaction time available for resorting to remedial measures remains critical.

As already described in detail in chapter 3, central control faces many difficulties in taking timely action to ALL the situations due to time lagged communication of data on serial lines, channel speed and bottlenecks developed at the receiving site, non availability of direct transmission path, digital processing delays, transitions to and from fibres, data rate and volume of thousands of measurements, requirement of immense processing power, limitations of on line analysis tools, FIFO mechanisms of Ethernet protocols with difficulty in assigning priority to emergent communications etc.

Distributed control proves to be quite effective in certain situations where delayed actions taken by central control, or no action taken at all results in disturbed systems.

7.2.2 Dynamic Modeling of Selected System

The variables to be calculated for solving the power system shown in figure 7.1 include voltages and phase angles on each bus (Uk and ɵk), rotor angle and speed (δk and ωk) of each machine along with active and reactive power Pk and Qk on each bus. elevant equations for solving the system are of the form as mentioned below [91]:

$$\left.\begin{aligned} \dot{\delta}_k &= \omega_k \\ \dot{\omega}_k &= \frac{1}{M_k}\left[P_{mk} - \frac{E'_{qk}U_k}{X'_{dk}}\sin(\delta_k - \theta_k)\right] \end{aligned}\right\} \quad 7.1$$

$$\left.\begin{aligned} P_k &= \sum_{l=1}^{N} B_{kl}U_k U_l \sin(\theta_k - \theta_l) \\ Q_k &= -\sum_{l=1}^{N} B_{kl}U_k U_l \cos(\theta_k - \theta_l) \end{aligned}\right\} \quad 7.2$$

And

$$\left.\begin{aligned} P_k + P_{LK} &= 0 \\ Q_k + Q_{LK} &= 0 \end{aligned}\right\} \quad 7.3$$

Where M denotes mutual inductance, U the voltage, X' the transient reactance, B the susceptance and P_{LK} and Q_{LK} the active and reactive powers of the load at bus k.

If 'x' variables are those which require solution of differential equations and 'y' variables are the ones which are ascertained by solving algebraic equations then:

$$\left.\begin{aligned} x &= [\delta_1 .....\delta_n, \omega_1 .....\omega_n]^T \\ y &= [\theta_1 .....\theta_N, U_1 .....U_N]^T \end{aligned}\right\} \quad 7.4$$

Equations 7.1 and 7.3 can be rewritten as:

$$\left.\begin{aligned} \dot{x} &= f(x,y) \\ 0 &= g(x,y) \end{aligned}\right\} \qquad 7.5$$

The above is a set of differential-algebraic equations which describe the dynamics of multi machine system. The differential equation describes the dynamics of generator and contribution of dynamic loads is also included. The algebraic equations consist of network equations based on Kirchoff's current law ie sum of all currents and hence powers flowing into bus k must be zero.

7.2.3 Load Flow Calculations in the Power Network

The technique for determining all bus voltages in a network is called Load Flow. If voltage magnitudes and voltage angles at all buses are known, the system is completely determined. In a network, power can be generated and consumed at many different locations. All locations where a line ends is called bus or node. At bus k expression for complex power in per unit system can be written as:

$$\overline{S}_{Gk} - \overline{S}_{Dk} = \sum_{j=1}^{N} \overline{S}_{kj} \qquad 7.6$$

Where

$\overline{S}_{Gk} = P_{Gk} + jQ_{Dk}$ = Complex power by the generator

$\overline{S}_{DK} = P_{DK} + jQ_{Gk}$ = Load consumed complex power

$\overline{S}_{kj} = P_{kj} + jQ_{kj}$ = Transmitted power to bus j

The power balance at the bus according to equation 7.6 must hold both for active and for reactive part of the expression. By using P_{GDK} and Q_{GDK} as notation for net generation of active and reactive power at bus 'k' respectively, the expressions 7.7 and 7.8 hold.

$$P_{GDk} = P_{Gk} - P_{Dk} = \sum_{j=1}^{N} P_{kj} \qquad 7.7$$

$$Q_{GDk} = Q_{Gk} - Q_{Dk} = \sum_{j=1}^{N} Q_{kj} \qquad 7.8$$

The above equations imply that for every bus in the system, the power balance must hold for both active and reactive power.

At each bus, four variables are of interest ie net generation of active power P_{GDK}, net generation of reactive power Q_{GDK}, voltage magnitude U_k and voltage phase angle Q_k. This means that total number of variables for a system are mx4 where "m" is number of buses. The voltage phase angles must be given as an angle in relation to reference angle. This reduces number of system variables to (mx4)-1.

If there are three buses in a power system, six equations can be made using equations 7.7 and 7.8. However a total of 12 variables (P, Q, U & θ for each bus) will be required to be known. Excluding reference angle, 11 variables will be required to be known. Since only 6 equations are held, another 5 equations will be required, or else 5x quantities must be known to be able to calculate remaining six variables. Depending on what quantities are known at a certain bus, the buses are modeled in three different ways. PQ or the load bus is the one for which net generated power P_{GDK} and Q_{GDK} are assumed to be known. In the system under consideration depicted in figure 7.1, bus 3 has been taken as PQ bus. For PU bus or the generator bus, the net active power generation P_{GDK} as well as the voltage magnitude Uk are assumed to be known. In the power system under study, bus 1 is taken as PU bus. Only one bus in the entire system is selected as slack or reference bus. The voltage angle is chosen as a reference angle θ_k (often 0^0) and the voltage angle is assumed to be known. Un known quantities are the net generation of both active and reactive power. Bus 2 in the system is slack bus.

If the values of U and ɵ are known, active and reactive power flow between all buses can be found using equations 7.9 and 7.10.

$$P_{kj} = \frac{U_k^{\ 2}}{Z^{\ 2}} R + \frac{U_k U_j}{Z^2}\left[X\ Sin\theta_{kj} - R\,Cos\theta_{kj} \right] \qquad 7.9$$

$$Q_{kj} = -BU_k^{\ 2} + \frac{U_k^{\ 2}}{Z^2} \times - \frac{U_k U_j}{Z^2}\left[R\,Sin\theta_{kj} + \times Cos\,\theta_{kj} \right] \qquad 7.10$$

The above equations are nonlinear since the expressions for power flow on a line includes squared voltages as well as trigonometric expressions. The system can therefore be solved by using Newton-Raphson Method.

7.2.4 Application of Newton Raphson Method to Selected Power System

The aim is to determine voltages at all buses in the system by applying Newton Raphson method. Consider a power system with N buses with all variables in P.U.

$$\left.\begin{aligned} P_k &= U_k \sum_{j=1}^{N} U_j \left[G_{kj} Cos(\theta_{kj}) + B_{k\,j} Sin(\theta_{kj}) \right] \\ Q_k &= U_k \sum_{j=1}^{N} U_j \left[G_{kj}\ Sin\ (\theta_{kj}) - B_{kj} Cos\ (\theta_{k\,j}) \right] \end{aligned}\right\} \qquad 7.11$$

Further more

$$\left.\begin{aligned} P_k &= \sum_{j=1}^{N} P_{kj} \\ Q_k &= \sum_{j=1}^{N} Q_{kj} \end{aligned}\right\} \qquad 7.12$$

and

$$\left.\begin{aligned} P_k - P_{GDK} = 0 \\ Q_k - Q_{GDK} = 0 \end{aligned}\right\} \qquad 7.13$$

For using standard Newton Raphson Method assume:

$$x=\begin{bmatrix}\theta\\U\end{bmatrix}=\begin{bmatrix}\theta_1\\\vdots\\\theta_N\\U_1\\\vdots\\U_N\end{bmatrix},\ f(\theta,U)=\begin{bmatrix}f_p(\theta,U)\\\vdots\\f_Q(\theta,U)\end{bmatrix}=\begin{bmatrix}P_1\\\vdots\\P_N\\Q_1\\\vdots\\Q_N\end{bmatrix},\ b=\begin{bmatrix}b_P\\b_Q\end{bmatrix}\begin{bmatrix}P_{DGD1}\\\vdots\\P_{GDN}\\Q_{DG!}\\\vdots\\Q_{GDN}\end{bmatrix} \qquad 7.14$$

It is also known that

$$\Delta g[x^{(i)}]=b-f[x^{(i)}]=-g[x^{(i)}] \qquad 7.15$$

Now from equations 7.14 and 7.15

$\Delta P_k = P_{GDk} - P_k$
$\Delta Q_k = Q_{GDK} - Q_k$

Based upon the Jacobian Matrix of Newton Raphson method

$$JAC=\begin{bmatrix}\dfrac{\partial f_P(\theta,U)}{\partial\theta} & \dfrac{\partial f_P(\theta,U)}{\partial U}\\ \dfrac{\partial f_Q(\theta,U)}{\partial\theta} & \dfrac{\partial f_Q(\theta,U)}{\partial U}\end{bmatrix}=\begin{bmatrix}H & N'\\ J & L'\end{bmatrix} \qquad 7.16$$

The entries of the these matrics are given by

$$H_{kj}=\frac{\partial Pk}{\partial\theta j} \qquad N'_{kj}=\frac{\partial Pk}{\partial Uj}$$

$$J_{kj}=\frac{\partial Qk}{\partial\theta j} \qquad L'_{kj}=\frac{\partial Qk}{\partial Uj}$$

Now since

$$JAC\left[x^{(i)}\right]\Delta\chi^{(i)}=0-g[\chi^{(i)}]=\Delta g[\chi^{(i)}] \qquad 7.17$$

$JAC\left[x^{(i)}\right]=\begin{bmatrix}H & N'\\ J & L'\end{bmatrix}$ and

$$\Delta x^{(i)} = \begin{bmatrix} \Delta & \theta \\ \Delta & U \end{bmatrix} \text{ and}$$

g(x) = ΔP_k and ΔQ_k

Putting all above in 7.17 one gets

$$\begin{bmatrix} H & N' \\ J & L' \end{bmatrix} \begin{bmatrix} \Delta\theta \\ \Delta U \end{bmatrix} = \begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} \qquad 7.18$$

To simplify entries of matrices N' and L' these matrices are multiplied with U, and equation 7.18 can be written as:

$$\begin{bmatrix} H & N \\ J & L \end{bmatrix} \begin{bmatrix} \Delta\theta \\ \frac{\Delta U}{U} \end{bmatrix} = \begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} \qquad 7.19$$

Now from equation 7.11, and taking partial differentiation as required, besides multiplying with U_j, the values of N'_{kj} and L'_{kj} :

Where K≠j ie not on same bus

$$\left.\begin{aligned} H_{kj} &= \frac{\partial P_k}{\partial \theta_j} = U_k U_j \left[G_{kj}\, Sin(\theta_{kj}) - B_{kj} Cos(\theta_{kj})\right] \\ N_{kj} &= U_j N'_{kj} = U_j \frac{\partial P_k}{\partial U_j} = U_k U_j \left[G_{kj} Cos\ (\theta_{bj}) + B_{kj} Sin\ (\theta_{kj})\right] \\ J_{kj} &= \frac{\partial Q_k}{\partial \theta_j} = -U_k U_j \left[G_{kj}\, Cos\ (\theta_{kj}) + B_{kj} Sin\ (\theta_{kj})\right] \\ L_{lk} &= U_j L'_{kj} = U_j \frac{\partial Q_k}{\partial U_j} = U_k U_j \left[G_{kj}\ Sin\ (\theta_{kj}) - B_{kj} Cos\ (\theta_{kj})\right] \end{aligned}\right\} \qquad 7.20$$

and for k=j i.e, on the same bus

$$H_{kj} = \frac{\partial P_k}{\partial \theta_k} = -Q_k - B_{kk} U_k^{\,2}$$

$$N_{kk} = U_k \frac{\partial P_k}{\partial U_k} = P_k + G_{KK} U_k^{\,2}$$

$$J_{kk} = \frac{\partial Q_k}{\partial \theta_k} = P_k - G_{kk} U_k^{\,2}$$

$$L_{kk} = U_k \frac{\partial Q_k}{\partial U_k} = Q_k - B_{kk} U_k^{\,2}$$

7.21

Now based upon equation 7.19

$$\begin{bmatrix} \Delta Q \\ \frac{\Delta U}{U} \end{bmatrix} = \begin{bmatrix} H & N \\ J & L \end{bmatrix}^{-1} \begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} \qquad 7.22$$

And hence values of U and ϴ are obtained as implemented in software.

Finally U and ϴ will be updated as:-

$$\theta_k = \theta_k + \Delta \theta_k$$

$$\mathrm{U_k} = \mathrm{U_k}\left[1 + \frac{\Delta \mathrm{U_k}}{\mathrm{U_k}}\right]$$

7.23

The relevant portion of M File for load flow is shown in figure 7.4.

```
    for m=1:buses
    for n=1:buses
        PP(m,n)=VOLT(m)*VOLT(n)*(G(m,n)*cos(ANG(m)-ANG(n))+B(m,n)*sin(ANG(m)-ANG(n))); % active power transferred from bus m to bus n
        QQ(m,n)=VOLT(m)*VOLT(n)*(G(m,n)*sin(ANG(m)-ANG(n))-B(m,n)*cos(ANG(m)-ANG(n)));
    % Implementation of eq set 6.17
    end


end
% end
P=sum (PP')% vector containging power transferred from each bus  as a column
Q=sum (QQ')
DelP=PGD-P([bus1,bus3]) % not calculated for Slack Bus %implementation of equation 6.19

for m=1:buses
 for  n=1:buses  % so as to make jacobian , equation set 6.26 and 6.27
    if m==n
        H(m,m)=-Q(m)-B(m,n)*VOLT(m)*VOLT(m);
    else
        H(m,n)=VOLT(m)*VOLT(n)*(G(m,n)*sin(ANG(m)-ANG(n))-B(m,n)*cos(ANG(m)-ANG(n)));
    end
end
end
H(bus2,:)=[];
H(:,bus2)=[];
Jac= [H]
Inv=inv(Jac);
CH=Inv*[DelP]'; % equation 6.28 implementation
deltheta = CH(1:2); % equation 6.29 implementation

iter = iter + 1

ANG([bus1 bus3])=ANG([bus1 bus3])+deltheta' % final angles on each bus
end
```

Figure 7.4: Load Flow of Power System

7.2.5 Swing Equation for the System

When the rotor is rotated by the turbine, a rotating magnetic flux is produced in the air gap. The magneto motive force (mmf) produced by the field current in the field winding combines with the mmf produced by currents in the stator winding. The result and flux across the air gap between stator and rotor, induces voltages in each phase of armature winging. These induced voltages have the same magnitudes, but at phase shifted by 120 electrical degrees.

The resultant flux also provides the electromagnetic torque (or the electrical output torque denoted by ‘Te measured in Nm) between stator and rotor. This electromagnetic torque (developed in the generator when it delivers power) opposes torque of turbine (other mechanical input torque denoted by ‘Tm’ also measured in Nm).

One of the most important factors in studying stability of a power system is to motion of rotor. The dynamics of this motion is described by a set of differential equation based on Newton's second law:

$$J\frac{d^2B_m}{dt^2} = J\frac{dw_m}{dt} = T_m - T_e \qquad 7.24$$

Where B_m= Mechanical rotor angle which defines the instantaneous position of rotor d-axis with respect to a stationary reference frame and

$B_m = W_{mt} + B_{mo}$

Where Wm= mechanical speed or angular speed of rotor and B_{mo} is the initial position of rotor i.e, at t=0 with respect to reference axis.

J= Total Moment of inertia of turbine, shaft and generator measured in Kgm^2.

Wms = Rotor mechanical synchronous speed=$2\Pi n_o$ where n_o is revolutions per minute.

Multiplying swing equation 7.24 with Wm, the obtained expression is.

$$W_m J \frac{Dwn}{dt} = P_m - P_e \qquad 7.25$$

Where

$P_m = W_m T_m$ which is mechanical power input given in watts

$P_e = W_m T_e$ which is three phase electrical power output given in watts.

Next, the inertia constant H of the generator is defined by

$$H = \frac{Wks}{Sng} = \frac{0.5Jw^2ms}{Sng} \qquad 7.26$$

Where

Wks= Total kinetic energy stored in the generator in steady state (or stored KE in the rotating inertia at speed Wms)

Sng= Generator rated volt Amperes

Typically H ranges between 1 and 6 seconds, depending on the size and type of generator. The inertia constant states how many seconds it will take to bring the generator from synchronous speed to stand still if rated power is extracted from it while no mechanical power in supplied by the turbine

Substituting value of "J" from equation 7.26 in equation 7.25,

$$\mathrm{W_m} = \frac{2H\ Sng}{\mathrm{W^2ms}} \frac{dw_m}{dt} = P_m - P_e \qquad 7.27$$

Now consider equations as under:-

$$\mathrm{W_g} = \frac{\mathrm{p}}{2} w_m \qquad 7.28$$

$$\frac{\mathrm{d}\beta}{\mathrm{dt}} = w_g \qquad 7.29$$

$$\delta = (\mathrm{wg} - \omega \mathrm{s})\ \mathrm{t} + \mathrm{Co} \qquad 7.30$$

$$\frac{d\delta}{dt} = \delta^{\cdot} = (wg - ws) = w \qquad 7.31$$

$$\frac{d^2\delta}{dt^2} = \delta^{\cdot\cdot} = \frac{d}{dt}(wg - ws) = Wg^{\cdot} = W^{\cdot} \qquad 7.32$$

$$\frac{w_g}{w_s} = \frac{p/2\ w_m}{p/2 w_{m2}} = \frac{w_m}{w_s} \qquad 7.33$$

Where

p= number of poles

Co= Constant (in steady state, S=Co: Wg=Ws)

Using equation 7.28 to 7.32 , we can write equation 7.27 as:

$$2\frac{HSng}{w_s} = \frac{dwg}{dt} = \frac{w_s}{w_g}(P_m - P_e) \cong P - P_e \qquad 7.34$$

$$(\text{Assuming}\frac{w_s}{w_g} \cong 1)$$

Dividing both sides of equation 7.27 with P base it results into:

$$\mathrm{M}\frac{d_{wg}}{dt} = \frac{P_m - P_e}{P_{base}} = Pm^{pu} - Pe^{pu} \qquad 7.35$$

Where

$$\mathrm{M}\frac{2H}{w_s}\frac{Sng}{P_{base}} \quad and \quad Pe^{pu} = \mathrm{Re}al\left[\bar{E}_q \bar{I}^*\right]$$

Next apply equation 7.30 and 7.31

$$\left.\begin{aligned} \dot{\delta} &= (w\,\mathrm{g} - w\,\mathrm{s}) = w \\ \dot{w} &= \frac{1}{M}(Pm^{pu} - Pe^{pu}) \end{aligned}\right\} \qquad 7.36$$

Furthermore the contribution of damping power provided by damper windings may be included in the swing equation 7.36. The damper power is denoted by

$P_D = D(w_g - w_s)$

Where D is a small positive constant. Thus equation 7.36 can be re written as:

$$\left.\begin{aligned} \dot{\delta} &= w \\ \dot{w} &= \frac{1}{M}(P_m^{pu} - P_e^{pu} - D_w) \end{aligned}\right\} \qquad 7.37$$

Here “pu” superscript is omitted which means all values are in pu. Further more in steady state $w_g = w_s$ i.e, generator rotates with a speed equal to nominal speed.

If valves of H, M, D are known one can solve for δ and w and value of J will not be required.

7.2.6 Scenarios for Simulation

The system is either uncontrolled, controlled with the help of a central control or it can be controlled with the help of local agents. Similarly, either the load is increased to a limit where a series compensator of 0.78 pu helps to maintain synchronism of the two generator system. If the load increases further, then load shedding at Bus 1 is resorted to. All these contingencies have been covered under three scenarios which are Pre fault scenario, On fault scenarios and Post fault scenario.

System is solved for load flow and initial values of state variables and algebraic variables i.e, voltages on each bus and their phase angles, Active and reactive power at each bus, EMF of each generator, rotation angles of generator and speeds are calculated. .

A fault has been introduced at 0.5 sec and depending upon nature of fault, new values of parameters are calculated and difference of rotation angles of both the generators have been plotted against time in each scenario, This has been done under separate headings of SCADA control and Agent control. It is observed that if the fault is cleared using local agents within 130 m seconds, the generators remain in synchronism. Similarly if system is uncontrolled or SCADA control, where response is received after 136 m sec, the system goes out of synchronism.

At each time Interval as selected for Pre fault, On fault and post fault scenarios, the equations are solved using Matlab “ode 45” function by file titled “f_xy” and state variables of swing equation i.e, δ and ω for both the generators are returned. Similarly at each time interval, algebraic equations are solved using Matlab function “f solve” and file titled “g_xy” . This returns values of algebraic variables that include voltages of buses, angles of voltages, and active/reactive power at each bus.

7.2.7 Simulation Results

The simulation in matlab for the system described in figure 7.1 reveals that if it runs in pre fault or normal conditions, the generators stay in synchronization as shown in figure 7.5. The value acquired for the difference of angles depends upon values of M_k, emf of generator E'_{qk}, bus voltage U_k, transient reactance x'_{dk} and the relationship between dq axis reference frame and network reference frame as given by equation 7.4.

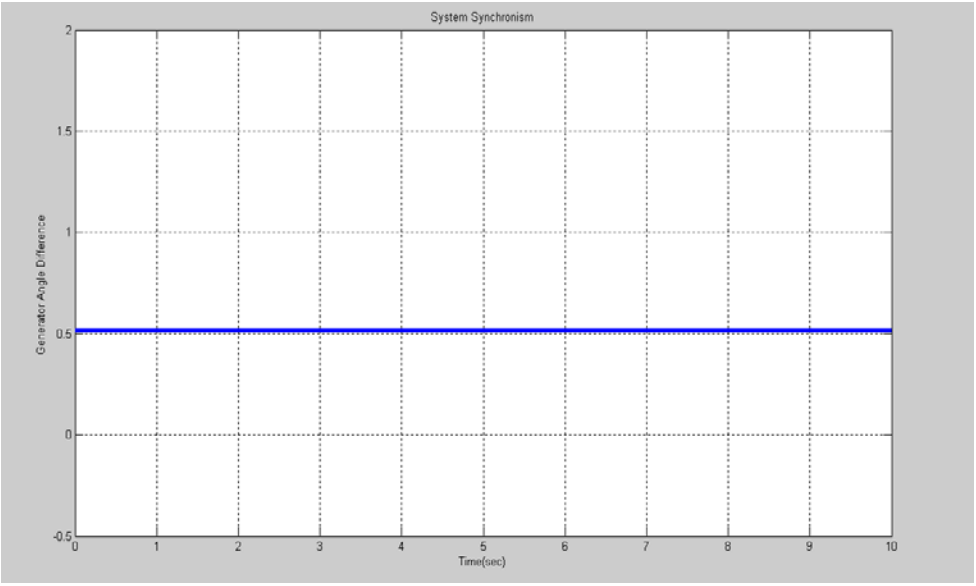

Figure 7.5: System in Synchronism

On enhancement of load on the system bus 2 if the distributed intelligent agent is not available and on non availability of any remedial action, or delayed response by a central control surpassing time in milliseconds, the generators go out of synchronization as shown in the simulation results of Figure 7.6.

However, if a response of introducing series compensation immediately on sensing the enhanced load within the permissible limits of series compensator capacitance is taken, the system regains synchronism. The simulation result is shown in Figure 7.7. The maximum series compensation limit for this simulation is 0.76 pu. If the load enhancement is such that the

requirement surpasses the capacity of series compensator, then the solution to this will be dropping of load to retain system stability. The simulated result of a case where no remedial action was taken on enhancement of too much of load beyond handling capability of series compensator is shown in figure 7.8. Figure 7.15 on the other hand demonstrates the decision taken by distributed control to drop the load from line.

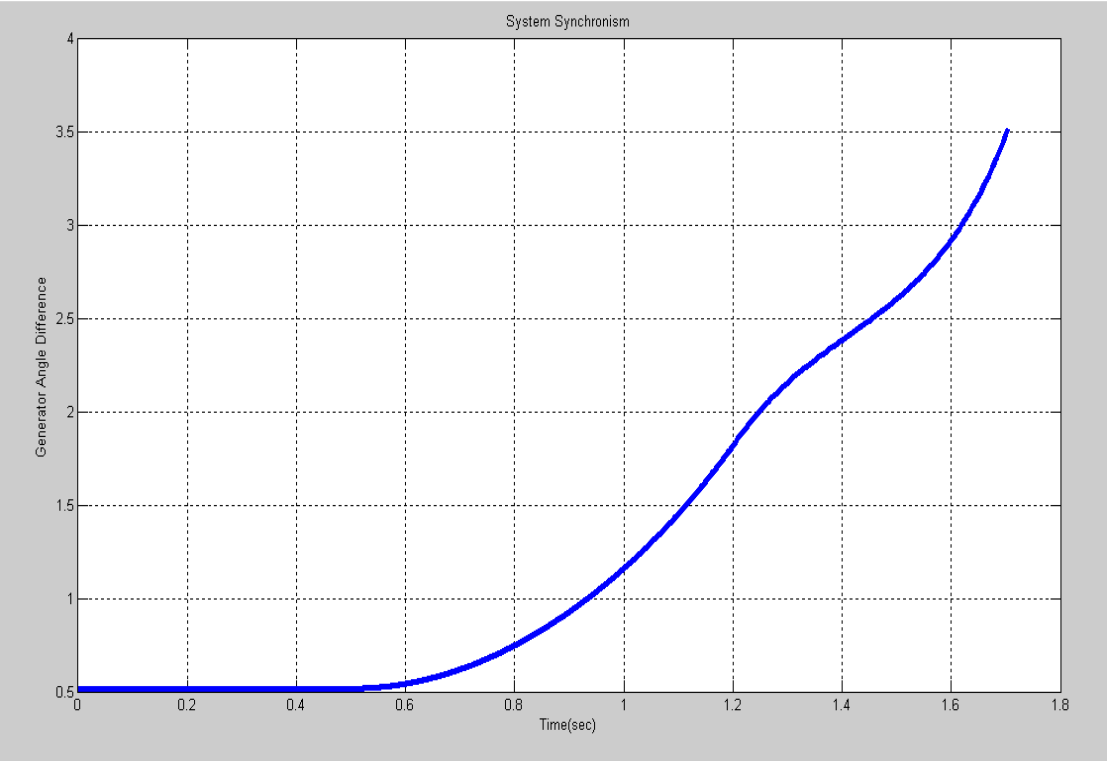

Figure 7.6: Delay in Remedial Action on Increase in Load

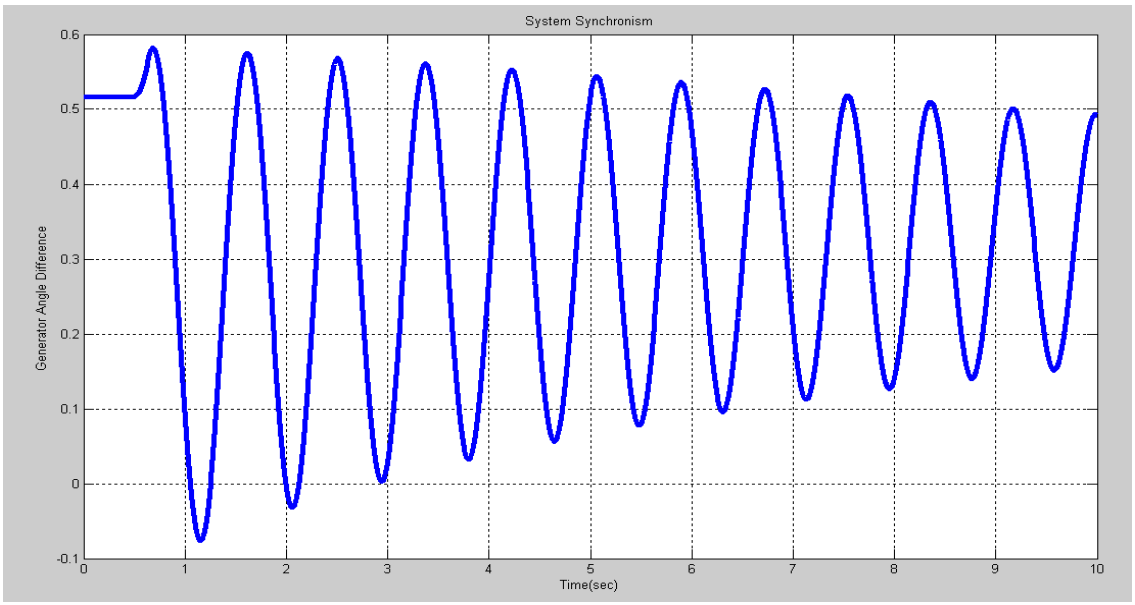

Figure 7.7: Achievement of Synchronism with Series Compensation

Relatively quicker synchronism is achieved if the load is dropped as compared to the one achieved from incorporation of series compensation shown in Figure 7.7. The simulation result is shown in figure 7.9.

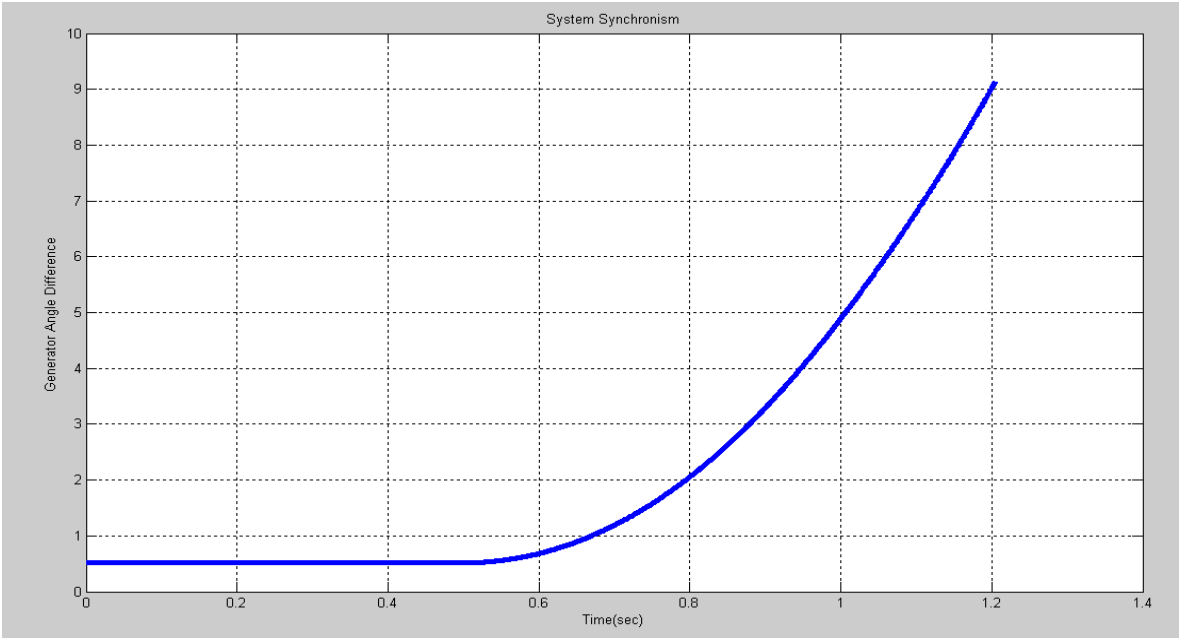

Figure 7.8: Enhancement of Load Beyond Capacity of Series Compensator

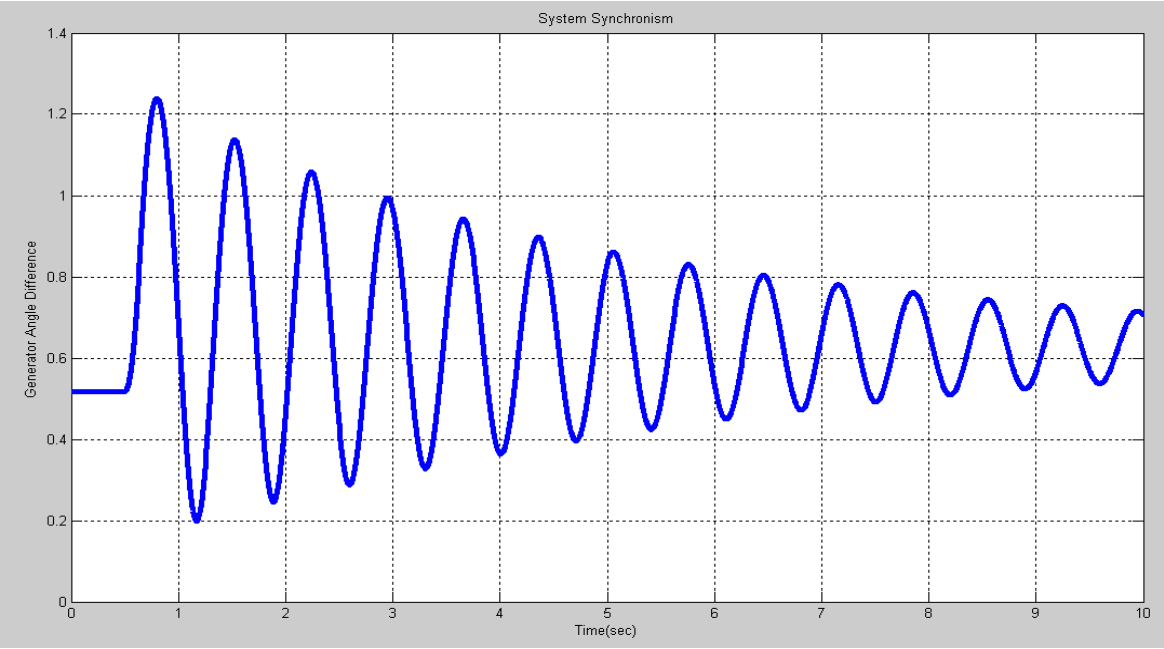

Figure 7.9: Synchronism on Dropping of Load from Line

7.2.8 Proactive Response by Agents with Simulation Results

The three bus multiple generation system with shunt capacitance has been simulated using Simulink platform as depicted in Figure 7.10. [92].

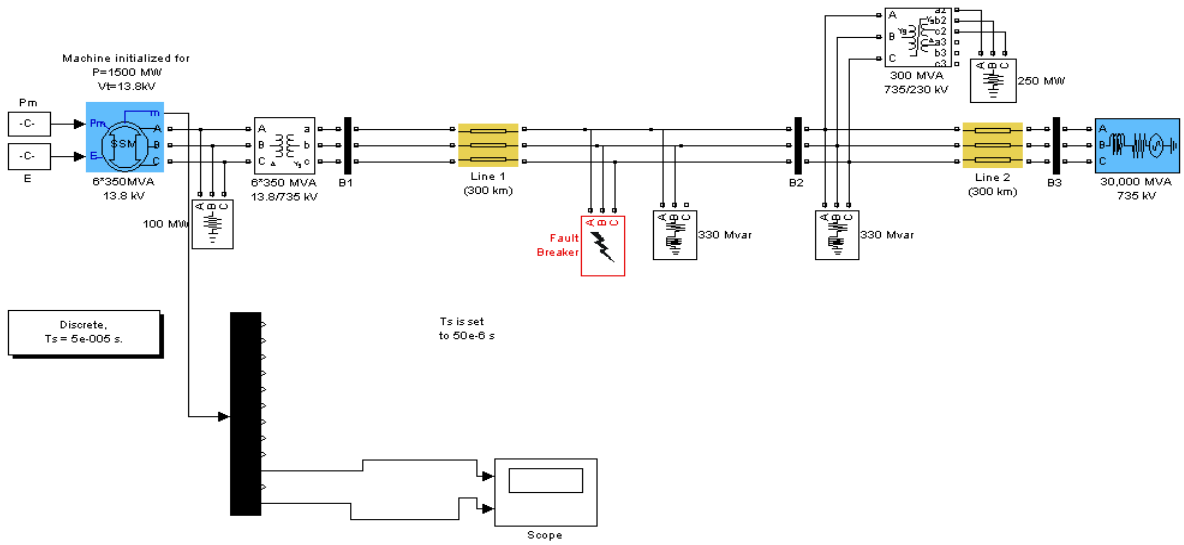

Figure 7.10: Three Bus Multiple Generator Model

In absence of fault and as per nominal parameters the system performs around 0.6 to 0.7 pu of the electrical power (Pe) on the synchronous generator. When the three phase fault occurs on the line as shown, the value of Pe increases to around 0.93 pu showing that any other disturbance will force the system to be unstable. The value of Pe on fault is shown in Figure 7.11.

If the system possesses an intelligent agent, it will immediately sense the problem and introduce series compensation to reduce the loading of generator thus saving it from any other disturbance which will certainly cause the system to become unstable. The series compensated model is shown in Figure 7.12. The reduced value of Pe is shown in Figure 7.13.

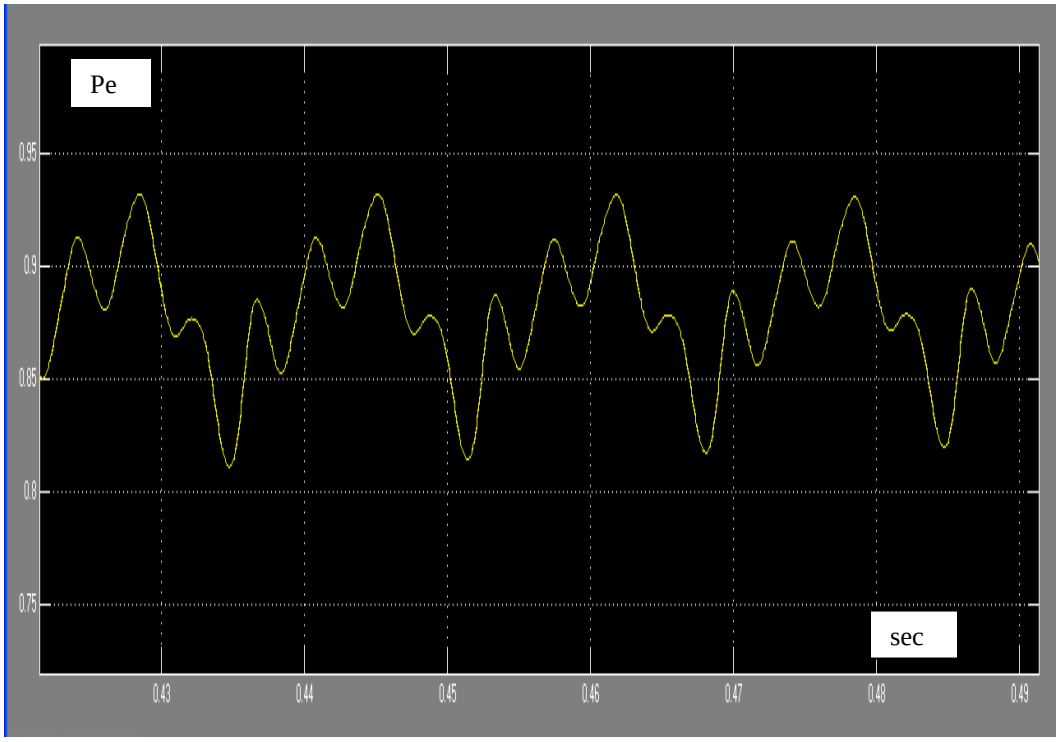

Figure 7.11 Excessive Loading of Synchronous Generator on Fault

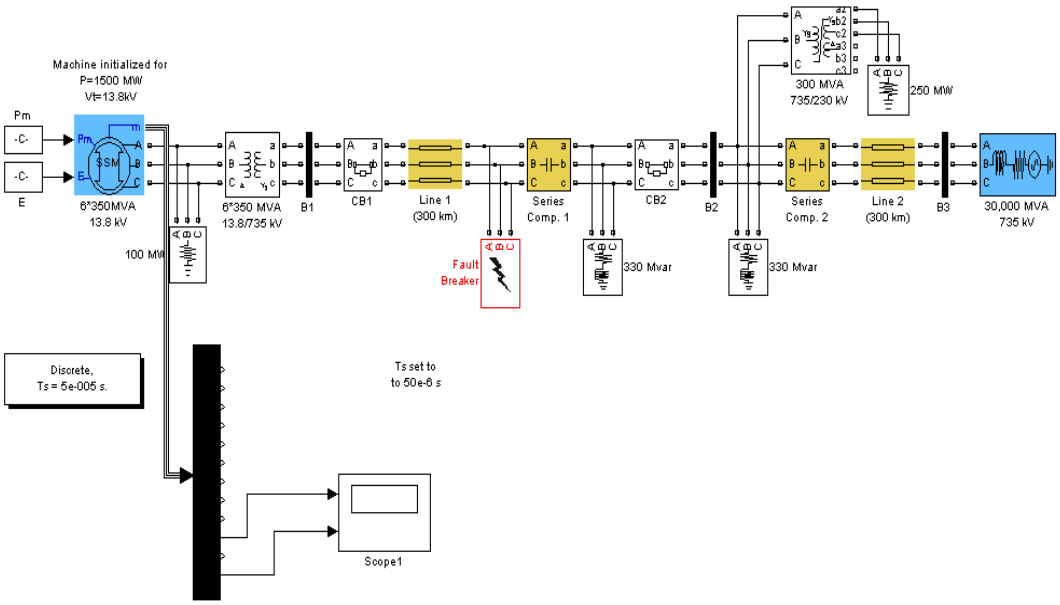

Figure 7.12: Series Compensation Incorporation by Agent

If the load increases beyond the capacity of series compensation, the agent controlled network will opt to drop the load by opening relevant circuit breaker to retain stability before rated values of circuit breaker are approached. In Figure 7.14, the load has been doubled and agent on sensing the enhanced load beyond the capacity of series compensator, opened the circuit breaker reducing the loading of generator to a value of Pe around 0.56 pu as shown in Figure 7.15.

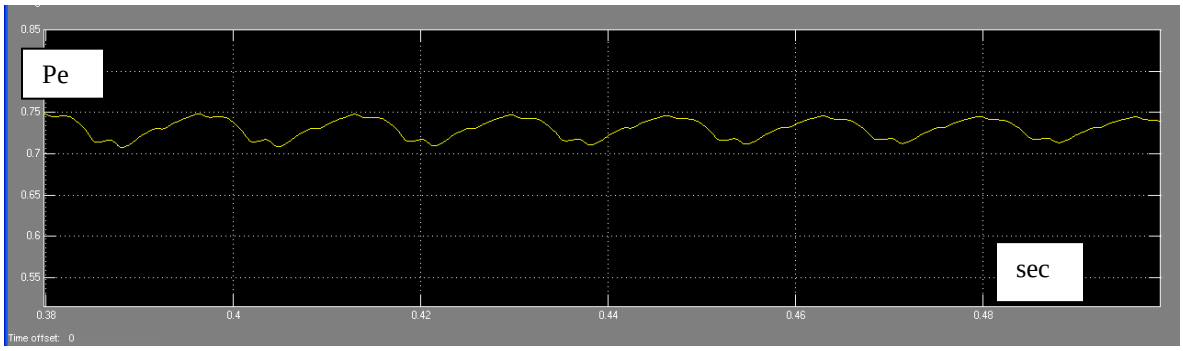

Figure 7.13: Reduced Loading on Fault with Compensation

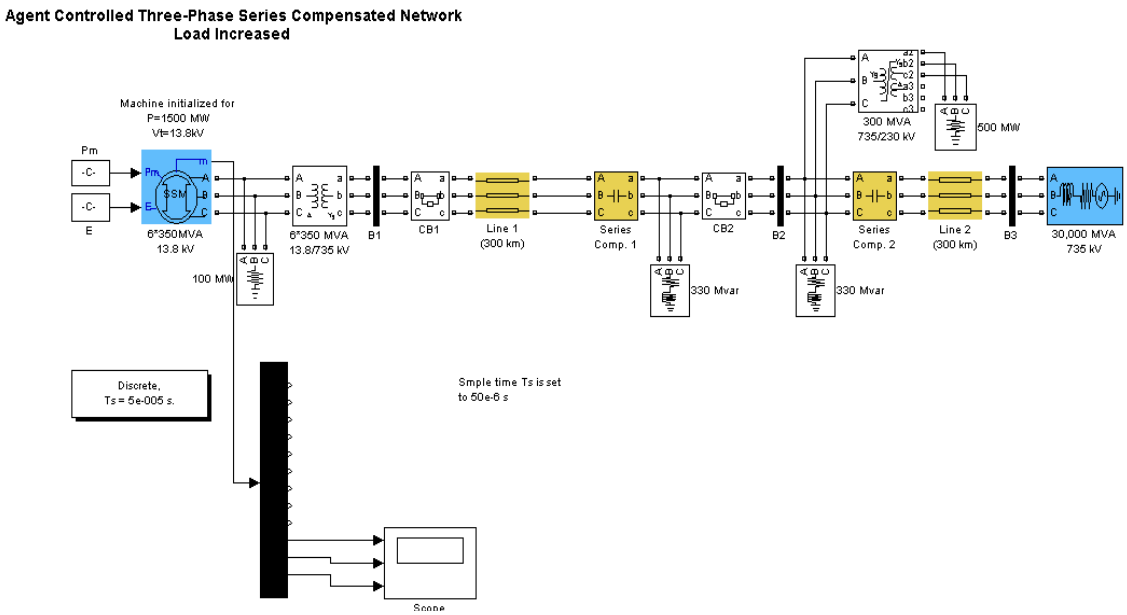

Figure 7.14: Line Opening at Double the Rated Load

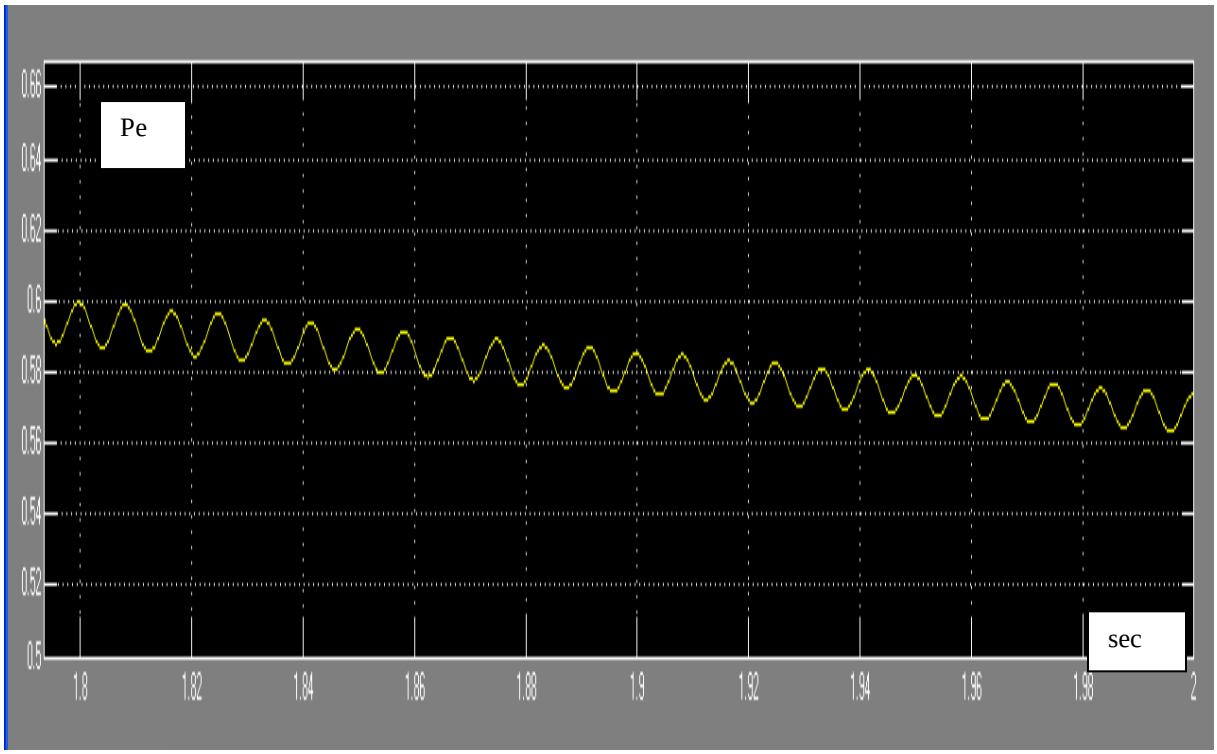

Figure 7.15: Reduction in Loading on Opening of Line

7.3 Summary

The results shown in preceding paragraphs indicate that system lost synchronism when remedial measures were delayed as expected in a central control. Similar situation is likely to crop up when no control is exercised as in the case of most of the networks, especially those found in developing countries. Proactive approach by sensing overloading of the system resulted in taking appropriate decision of introducing series compensation to enhance loadability, or elimination of line with fault or overload to ensure smooth performance of rest of the network.

Chapter # 8

Conclusion and Recommendations

8.1 Overview

Today's complex electric power distribution systems (EPDS) in data centres, require sophisticated control techniques to support all aspects of operation including failures. In many critical applications, they need to maintain minimal operating conditions under failure conditions. When a high degree of survivability is desired the effects of failures must be mitigated and control must be maintained in survivable scenarios. To achieve this goal it is necessary to make a series of decisions and control actions. The fault has to be detected, its source identified and its magnitude estimated. Depending on the nature of the failure, actions are to be taken that compensate it. After this, a new functional network topology is chosen and EPDS has to be reconfigured. All these decisions must be made by a self-reconfigurable control system that incorporates not only simple regulatory loops and the supervisory control logic but also a set of components that detect, isolate, and manage faults, in coordination with the control functions. The goal is to increase survivability, eliminate human mistakes, make intelligent reconfiguration decisions more quickly, reduce manpower required to perform the functions, and provide electric power to critical loads through the surviving system. In fact, it is envisioned that in future, the EPDS will have distributed intelligence for rapid self-reconfiguration under fault scenarios. In order to achieve distributed intelligence in the EPDS, state-of-the-art software techniques must be used. As a sub-field of Distributed Artificial Intelligence, Agent Systems are a promising technology to fulfill these requirements. An agent is an abstraction, a logical model that describes software that “acts on behalf of” a user or other program; that software agent has

the authority to decide if and when an action is appropriated, also it has the autonomy to activate itself or it can be invoked by a task. Keeping in view the expected dividends as highlighted, an innovative conceptual model based upon distributed system has been proposed for control of power networks.

In this chapter summary of work and the goals achieved has been included. Limitations and recommended future work in the identified domains has also been discussed.

8.2 Achievements

After identifying the problem and establishing that resorting to employment of a distributed control system for effective, autonomous and proactive control will yield dividends mandatory for handling multifarious issues of existing and future power distribution networks, vitally important assignments concerning control of a complex and distributed network were identified. Agent based approach offered a much more natural representation of real world systems where various locations can make decisions according to their own agenda and priorities. Each individual location can achieve overarching objectives that are not achievable by the individuals alone. When the goals of individual agents are closely aligned, and they are completely benevolent and veracious so that they always respond to requests made to them, and always provide information when queried, then the resulting system comes very close to resembling an object oriented system. Realising this, a detailed object model was prepared for agent based system assigning different groups of tasks to individual agents in the hierarchical and concurrent model thus prepared. Every agent was treated as an entity with identified attributes and functions. This was followed by a use-case model where every function was treated as a distinct use-case with defined stimuli, data handling and responses to various recipients in the multiagent system. Taking this model as a basis, a state machine model was

prepared focusing on the design of logical and physical architectures, but independent of physical constraints and non functional requirements. An effort to incorporate scheduling approach and identification of terminal objects has been made. Requirement of distributed control has been established by simulating a power system thereby analyzing the performance under distributed as well as central or no control

8.3 Limitations

Despite an effort to enhance the scope of the proposed model through incorporating a large number of activities compulsory for optimum functioning of control for power network, there are still domains which have not been included. An enhancement in identified functionalities is therefore not only possible for this model but also for any other model that will include more functions and activities, such as modeling of extreme interdependencies in system inputs, harmonic interference of large population of semi conductor based electronic equipment alongwith harmonic cancellation, thermal and electric energy storage/energy conservation and CO_2 emission control, minimization of transmission losses and natural gas cogeneration etc.

Non functional requirements of the system have also not been taken up in detail such as timings, dependability and security. Difference in reading period of different sensors has been ignored and timing attributes have not been associated with objects. These may be required by some structured design methods which are targeted towards real time systems. There are however design methods that do not impose a computational model that will ensure that effective timing analysis of the final system is undertaken. In real time system design and software development method life cycle, timing problem is not catered for, in requirement specification, architecture design, detailed design and coding phases and is identified in testing or deployment phase.

8.4 Future Work

Proposed areas for future work include incorporation of genetic algorithms/ fuzzy logic in tandem with agent systems for optimisation of generation scheduling, economic despatch and self healing of networks.

A research into enhanced number of layers of hierarchy of agents by using concept of nesting is another area of exploring the possibility of furthering the performance of control structures.

Automation of assigning sub problems to various agents is also a possibility. Agents can be made to possess the capability of negotiating extensively amongst themselves as to how best they can distribute responsibilities at a given instance, or swap the functionalities with a view to furthering the efficiency and robust control of distributed power networks.

Conditions for convergence to optimal solutions of individual agents for solving overall problem may be worked out to fit in the overall structure proposed in models included in the thesis.

Generation of annotated behaviour models from end user scenarios will simplify the requirement elicitation process by developing a tool for automation of generating desired behaviours. This work can further be enhanced to interactively produce additional scenarios that the end user can classify as positive or negative behaviours from the point of view of desired behaviours.

After having detailed use-case, object and state machine modeling, either a multiagent system development platform can be used to simulate the functioning, or coding in an object

oriented programming language can commence in tandem with hardware design for different modules of system.

8.5 Challenge of Implementation

Challenge of implementing this system requires a balanced, risk-managed, and cost-effective approach to investments and use of technology that can make a sizable difference in mitigating the risk. Revolutionary developments in both information technology (such as Artificial Intelligence, System Dynamics, Network/Complexity Theory and Mechatronics) and materials science (such as Nanotechnology, Microfabrication and Advance Materials) promise significant improvement in the security, reliability, efficiency, and cost effectiveness of power distribution infrastructures [89]. Steps taken now can ensure that this critical infrastructure continues to support population growth and economic growth without environmental harm. Considering their impact, regulatory agencies should be able to induce electricity producers to plan and fund the process, which might be the most efficient way to get it into operation.

Simply replicating the existing system through expansion or replacement will not only be technically inadequate to meet the changing demands for power, but will also produce a significantly higher price tag for electricity. Through the transformative technologies, the countries can put in place a 21st century power system capable of eliminating critical vulnerabilities while meeting intensified consumer demands.

A phased approach to system implementation will probably be a prudent approach. For example, equipment purchased, should emphasize switchgears, regulators, transformers, controls, and monitoring equipment that can be easily integrated with automated transmission

and distribution systems. Long-term plans for equipment upgrades should also address system integration considerations.

Energy policy, technological development and innovation require long-term commitments as well as sustained and patient investments in innovation, technology creation, and development of human capital. Keeping in view economic, societal, and quality-of-life issues and the pivotal role of the electricity infrastructure, a self healing grid is essential. Every node in the power network of the future will be awake, responsive, adaptive, price-smart, real-time, flexible, and interconnected, which essentially is the prime focus of the work covered in the thesis.

8.6 Summary

Scenario based use-case and state machine models are recognised as effective means for requirement elicitation and documentation. They support an informal, narrative and concrete style of description that focuses on the dynamic aspects of software-environment interaction. A scenario is a temporal sequence of interactions among system components, that is, software-to-be together with its environment. This has to be made up of active components conceived as agents that control system behaviours.

References

[1]. Sandell, jr., P. Varaiya, M. Athans, and M.G. “Safonov. Survey of decentralized control methods for large scale systems”. *IEEE Transactions on Automatic Control*, 23(2):108–128, April 1978.

[2]. Michael Luck, Ronald Ashri, Mark D’ Inverno, “Agent Based Software Development” ISBN 1-58053-605-0, 2004, Artech House Inc.

[3]. [10] Wooldridge, M. J., and N. R. Jennings, "Intelligent Agents: Theory and Practice," Knowledge Engineering Review, Vol. 10, No. 2,1995, pp.115-152.

[4]. D. Pomerleau, A. Pomerleau, D. Hodouin, and ´E. Poulin. A procedure for the design and evaluation of decentralised and model-based predictive multivariable controllers for a pellet cooling process. *Computers and ChemicalEngineering*, 27:217–233, 2003.

[5]. R. Irizarry-Rivera and W. D. Seider. Model-predictive control of the Czochralski crystallization process. Part I. Conduction-dominated melt. *Journal of Crystal Growth*, 178:593–611, 1997.

[6]. S. Ochs, S. Engell, and A. Draeger. Decentralized vs. model predictive control of an industrial glass tube manufacturing process. *Proceedings of the 1998 IEEE Conference on Control Applications*, pages 16–20, Trieste, Italy, 1998.

[7]. R.N. Silva, L.M. Rato, J.M. Lemos, and F. Coito. Cascade control of a distributed collector solar field. *International Journal of Process Control*, 7(2):111–117, 1997.

[8]. F.D. Vargas-Villamil and D.E. Rivera. Multilayer optimization and scheduling using model predictive control: Application to reentrant semiconductor manufacturing lines. *Computers and Chemical Engineering*, 24:2009–2021, 2000.

[9]. M.W. Braun, D.E. Rivera, M.E. Flores, W.M. Carlyle, and K.G. Kempf. A model predictive control framework for robust management of multiproduct, multi-echelon demand networks. *Annual Reviews in Control*, 27:229–245, 2003.

[10]. M.R. Katebi and M.A. Johnson. Predictive control design for large-scale systems. *Automatica*, 33(3):421–425, 1997.

[11]. G. Georges. Decentralized adaptive control for a water distribution system. *Proceedings of the 3rd IEEE Conference on Control Applications*, pages 1411–1416, Glasgow, UK, 1999.

[12]. E. Camponogara, D. Jia, B.H. Krogh, and S. Talukdar. Distributed model predictive control. *IEEE Control Systems Magazine*, 1:44–52, February 2002.

[13]. M. Aicardi, G. Casalino, R. Minciardi, and R. Zoppoli. On the existence of stationary optimal receding-horizon strategies for dynamic teams with common past information structures. IEEE Transactions on Automatic Control, 37:1767–1771, November 1992.

[14]. [60] L. Acar. Some examples for the decentralized receding horizon control. Proceedings of the 31st IEEE Conference on Decision and Control, pages 1356–1359, Tucson, Arizona, 1992.

[15]. S. Sawadogo, R.M. Faye, P.O. Malaterre, and F. Mora-Camino. Decentralized predictive controller for delivery canals. Proceedings of the 1998 IEEE International on Systems, Man, and Cybernetics, pages 3380–3884, San Diego, California, 1998.

[16]. H. El Fawal, D. Georges, and G. Bornard. Optimal control of complex irrigation systems via decomposition-coordination and the use of augmented lagrangian. Proceedings of the 1998 IEEE International on Systems, Man, and Cybernetics, pages 3874–3879, San Diego, California, 1998.

[17]. M. Gomez, J. Rodellar, F. Vea, J. Mantecon, and J. Cardona. Decentralized predictive control of multireach canals. Proceedings of the 1998 IEEE International on Systems, Man, and Cybernetics, pages 3885–3890, San Diego, California, 1998.

[18]. M. Baglietto, T. Parisini, and R. Zoppoli. Neural approximators and team theory for dynamic routing: A receding-horizon approach. Proceedings of the 38th IEEE Conference on Decision and Control, pages 3283–3288, Phoenix, Arizona, 1999.

[19]. D. Jia and B. Krogh. Min-max feedback model predictive control for distributed control with communication. Proceedings of the 2002 American Control Conference, pages 4507–4512, Anchorage, Alaska, May 2002.

[20]. D. Jia and B.H. Krogh. Distributed model predictive control. Proceedings of the 2001 American Control Conference, pages 2767–2772, Arlington, Virginia, June 2001.

[21]. W.B. Dunbar and R.M. Murray. Model predictive control of coordinated multi-vehicle formations. Proceedings of the 41st IEEE Conference on Decision and Control, pages 4631–4636, Las Vegas, Nevada, December 2002.

[22]. N. Motee and B. Sayyar-Rodsari. Optimal partitioning in distributed model predictive control. Proceedings of the 2003 American Control Conference, Denver, Colorado, June 2003.

[23]. W.B. Dunbar and R.M. Murray. Distributed receding horizon control with application to multi-vehicle formation stabilization. Technical report, Division of Engineering and Applied Science, California Institute of Technology, Pasadena, California, 2004.

[24]. Syed M Alamdar Raza, Muhammad Akbar, Farrukh Kamran, M N Jafri, "An Innovative Conceptual Model of Multiagent System for Control of Distributed Power System Networks including Green Power Resources" Journal Energy Exploration and Exploitation(ISSN: 0144-5987)· Volume 24 · Number 4+5 · 2006 pp. 297–312.

[25]. Syed M Alamdar Raza, Muhammad Akbar, Farrukh Kamran, M N Jafri, "State Machine Model of Evolutionary Programs of Multiagent Systems for Control of Distributed Renewable Energy Networks" WSEAS Transactions on Power Systems ISSN(1790-5060), Issue 10, Volume 1, October 2006.

[26]. D.H. Shim, H.J. Kim, and S. Sastry. Decentralized reflective model predictive control of multiple flying robots in dynamic environment. Proceedings of the 42nd IEEE Conference on Decision and Control, 2003.

[27]. E. Camponogara. Controlling Networks with Collaborative Nets. PhD thesis, ECE Department, CarnegieMellon University, Pittsburgh, Pennsylvania, August 2000.

[28]. B. Franoise, Y. Magid, andW. Bernard, "Application of neural networks to load-frequency control in power systems," Neural Netw., vol. 7, no. 1, pp. 183–194, 1994.

[29]. T. P. I. Ahamed, P. S. N. Rao, and P. S. Sastry, "A reinforcement learning approach to automatic generation control," Elect. Power Syst. Res., vol. 63, pp. 9–26, Aug. 2002.

[30]. L. D. Douglas, T. A. Green, and R. A. Kramer, "New approaches to the AGC nonconforming load problem," IEEE Trans. Power Syst., vol. 9, no. 2, pp. 619–628, May 1994.

[31]. D. K. Chaturvedi, P. S. Satsangi, and P. K. Kalra, “Load frequency control: A generalized neural network approach,” Elect. Power Energy Syst., vol. 21, no. 6, pp. 405–415, Aug. 1999.

[32]. H. L. Zeynelgil, A. Demiroren, and N. S. Sengor, “The application of ANN technique to automatic generation control for multi-area power system,” Elect. Power Energy Syst., vol. 24, no. 5, pp. 345–354, Jun. 2002.

[33]. C. S. Indulkar and B. Raj, “Application of fuzzy controller to automatic generation control,” Elect. Machines Power Syst., vol. 23, no. 2, pp. 209–220, Mar.–Apr. 1995.

[34]. A. E. Gegov and P. M. Frank, “Decomposition of multivariable systems for distributed fuzzy control [power system load frequency control],” Fuzzy Sets Syst., vol. 73, no. 3, pp. 329–340, Aug. 1995.

[35]. J. Talaq and F. Al-Basri, “Adaptive fuzzy gain scheduling for load-frequency control,” IEEE Trans. Power Syst., vol. 14, no. 1, pp. 145–150, Feb. 1999.

[36]. Y. L. Karnavas and D. P. Papadopoulos, “AGC for autonomous power system using combined intelligent techniques,” Elect. Power Syst. Res., vol. 62, no. 3, pp. 225–239, Jul. 2002.

[37]. Y. L. Abdel-Magid and M. M. Dawoud, “Optimal AGC tuning with genetic algorithms,” Elect. Power Syst. Res., vol. 38, no. 3, pp. 231–238, Sep. 1996.

[38]. C. S. Chang, W. Fu, and F. Wen, “Load frequency control using genetic- algorithm based fuzzy gain scheduling of PI controllers,” Elect. Machines Power Syst., vol. 26, no. 1, pp. 39–52, Jan. 1998.

[39]. Z. M. Al-Hamouz and H. N. Al-Duwaish, “A new load frequency variable structure controller using genetic algorithms,” Elect. Power Syst. Res., vol. 55, no. 1, pp. 1–6, Jul. 2000.

[40]. A. Abdennour, “Adaptive optimal gain scheduling for the load frequency control problem,” Elect. Power Compon. Syst., vol. 30, no. 1, pp. 45–56, Jan. 2002.

[41]. S. K. Aditya and D. Das, “Design of load frequency controllers using genetic algorithm for two area interconnected hydro power system,” Elect. Power Compon. Syst., vol. 31, no. 1, pp. 81–94, Jan. 2003.

[42]. D. Rerkpreedapong, A. Hasanovic, and A. Feliachi, “Robust load frequency control using genetic algorithms and linear matrix inequalities,” IEEE Trans. Power Syst., vol. 18, no. 2, pp. 855–861, May 2003.

[43]. S. P. Ghoshal, “Application of GA/GA-SA based fuzzy automatic generation control of a multi-area thermal generating system,” Elect. Power Syst. Res., vol. 70, no. 2, pp. 115–127, Jul. 2004.

[44]. Syed M Alamdar Raza, Muhammad Akbar, Farrukh Kamran, “An abstract approach for control of complex distributed power system networks” International Journal of Information and Systems Sciences ISSN (1708-296X), Issue 1 Vol 2007.

[45]. Syed M Alamdar Raza, Muhammad Akbar, Farrukh Kamran, " Use-Case Model of Genetic Algorithms of Agents for Control of Distributed Power System Networks" , Scientific Khyber Journal(ISSN: 1017-3471), Vol 19, No 1 2006.

[46]. Syed M Alamdar Raza, Muhammad Akbar, Farrukh Kamran, "Object Model of Genetic Algorithms of Agents for Control of Distributed Renewable Energy Resources", IEEE PES 2005 Conference and Exposition in Africa Durban, South Africa, 11-15 July 2005, pp 282-285.

[47]. Syed M Alamdar Raza, Muhammad Akbar, Farrukh Kamran, " Use-Case, State Machine and Event Loop Model of Genetic Algorithms of Agents for Control of Distributed Power System Networks" , SOAS 05 Workshop at University of Paisley , Glasgow, UK, 11-13 December 2005.

[48]. Syed M Alamdar Raza, Farrukh Kamran, Muhammad Akbar, " Dynamic and Scenario Based Elicitation of Genetic Algorithms of Agents for Control of Distributed Power System Networks and Renewable Energy Resources, IEEE ICM 2005 Conference, Islamabad, 13-15 December 2005,pp 148-154.

[49]. Syed M Alamdar Raza, Farrukh Kamran, Muhammad Akbar, " State and Event Loop Model of Evolutionary Programs of Multiagent Systems for Control of Distributed Renewable Energy Networks" , IEEE INMIC 05 Conference Karachi, 23-25 Dec 2005,pp 34-39.

[50]. Wooldridge, M. J., and N. R. Jennings, "Intelligent Agents: Theory and Practice," Knowledge Engineering Review, Vol. 10, No. 2,1995, pp.115-152.

[51]. d'lnvemo, M., M. Luck, and UKMAS Contributors, "Multiagent Systems Research into the 21st Century," Knowledge Engineering Review, Vol. 16, No. 3, 2001, pp. 271-275.

[52]. Luck, M., and M. d'lnvemo, "A Formal Framework for Agency and Autonomy," Proceedings of the First International Conference on Multiagent Systems, Menio Park, CA: AAAI Press / MIT Press, 1995, pp. 254-260.

[53]. Franklin, S., and A. Graesser, "Is It an Agent, or Just a Program? A Taxonomy for Autonomous Agents," Intelligent Agents III — Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, Lecture Notes in Artificial Intelligence, 1. P. Mtiller, M. Wooldridge, and N. Jennings, (eds.). Volume 1193 of LNCS, Springer, 1997,pp.21-35.

[54]. Johnson, W. L., and B. Hayes-Roth, (eds.). Proceedings of the First International Conference on Autonomous Agents, New York: ACM Press, 1997.

[55]. Genesereth, M. R., and S. P. Ketchpel, "Software Agents," Communications of the ACM, Vol. 37, No. 7, 1994, pp. 48-53.

[56]. Lashkari, Y., M. Metral, and P. Maes, "Collaborative Interface Agents," Proceedings of the Twelfth National Conference on Artificial Intelligence, 1994, pp. 444-449.

[57]. Aylett, R., and M. Luck, "Applying Artificial Intelligence to Virtual Reality: Intelligent Virtual Environments," Applied Artificial Intelligence, Vol. 14, No. 1, 2000, pp. 3-32.

[58]. Kuokka, D., and L. Harada, "Matchmaking for Information Agents," Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, 1995, pp. 672-679.

[59]. Chess, D., et al., "Itinerant Agents for Mobile Computing," IEEE Personal Communications, Vol. 2, No. 5, 1995, pp. 34-49.

[60]. Wong, D., N. Paciorek, and D. Moore, "Java-Based Mobile Agents," Communications of the ACM, Vol. 42, No. 3, 1999, pp. 92-102.

[61]. Etzioni, O., et al., "The Softbot Approach to OS Interfaces," IEEE Software, Vol. 12, No. 4, 1995, pp. 42-51.

[62]. Toomey, C., and W. Mark, "Satellite Image Dissemination Via Software Agents," IEEE Expert, Vol. 10, No. 5, 1995, pp. 44-51.

[63]. Kinny, D., M. Georgeff, and A. Rao, "A Methodology and Modeling Technique for Systems of BDI Agents," Agents Breaking Away: Proceedings of the Seventh European Workshop on Modeling Autonomous Agents in a Multiagent World, Lecture Notes in Artificial Intelligence 1038, Y. Demazeau and J.-P. Miller (eds.), New York: Springer, 1996, pp. 56-71.

[64]. Jennings, N. R., et al., "Agent-Based Business Process Management," International Journal of Cooperative Information Systems, Vol. 5, Nos. 2 & 3, 1996, pp. 105-130.

[65]. Guttman,R. H.,A.G.Moukas, and P.Maes, "Agent-Mediated Electronic Commerce: A Survey," Knowledge Engineering Review, Vol. 13, No. 2, 1998, pp. 147-159.

[66]. Grand, S., and D. Cliff, "Creatures: Entertainment Software Agents with Artificial Life," Autonomous Agents and Multiagent Systems, Vol. 1, No. 1,1998, pp. 39-57.

[67]. Etzioni, O., and D. Weld, "Intelligent Agents on the Internet: Fact, Fiction, and Forecast," IEEE Expert, Vol. 10, No. 4,1995, pp.44-49.

[68]. M. Amin, “National Infrastructures as Complex Interactive networks” in “Automation, Control and Complexity” Samad and Wryrach (Eds), John Wiley and Sons, 1999.

[69]. M.Amin, John Stringer, “The Electric Power Grid: Today and Tomorrow” in “Harnessing Materials for Energy, Vol 33, No.4, pp 399-407, April 2008.

[70]. M. Amin, “Automation Control and Complexity. New Developments and Directions,” Samad and Weyrach (Eds), John Willey and Sons, 1999.

[71]. Eickstedt, Benjamin, “Adaptive Control of Heterogeneous Marine Sensor Platforms in Autonomous Sensor Network” IEEE Intelligent Robot and Systems Oct 2006 pp 5514-5521.

[72]. M. Amin “Towards Self Healing Energy Infrastructure System”, IEEE Computer Applications in Power, 2001, pp 20-28.

[73]. Gehrlce and and Bindner, “Building a Test Platform for agents in Power System Control” in IEEE Intelligent System Applications to Power Systems, November 2007, pp 5-8.

[74]. Kulwora Wanichpoong, Goodman, “Optimal Area Control of AC Railway System via PWM Traction Drives” IET Journal of Electric Power Applications, Volume 152 Issue 1 January 2005, pp 33-40.

[75]. Hai Jang Zhuang and Shen, “Distributed Medium Access Control” IEEE Journal of Wireless Communications Volume 14, Issue 3, June 2007 pp 25-31.

[76]. JinJun Liu, Xiaogang, “Stability Margin Monitoring for DC Distributed Power Systems via Perturbation Approaches” IEEE Transactions on Power Electronics Volume 18, Issue 6, Nov 2003 pp 1254-1261.

[77]. Ladoux P, Postigione, “A Comparative Study of AC/DC Converters for High Power DC are Furnace” IEEE Transactions on Industrial Electronics, Volume 52, Issue 3 June 2005 pp 747-757.

[78]. Heydt G.T, “Power Quality Engineering,” IEEE Power Engineering Review Volume 21, Issue 9, September 2001 pp5-7.

[79]. Heur and Daggle, official report on “Recent Reliability Issues and System Events” prepared by US Department of Energy, Dec 1999.

[80]. Chi, Lin and Shin, “Distributed Optimal Power Flow with Discrete Control Variables of Large Distributed Power Systems” IEEE Transactions on Power System, Volume 23, Issue 3, Aug 2008.

[81]. Abe, Hirokawa, “Optimal Intermediate Bus Capacitance for System Stability on Distributed Power Architecture” IEEE Power Electronics and Motion Control Conference, September 2008 pp 393-399.

[82]. Abe, Hirokawa, “Stability Design Consideration for on Board Distributed Power System Consisting of Full Regulated Bus Converter and POLs” IEEE Power Electronics Specialist Conference, June 2006 pp 1-5.

[83]. Abe, Hirokava, “Output Impedance Design Consideration of three Control Schemes for Bus Converter in On Board Distributed Power System” IEEE Power Electronics and Drive Systems November 2007 pp-1199-1204.

[84]. http://go.worldbank.org/DUCPVCOG20

[85]. Aylett, R., and M. Luck, "Applying Artificial Intelligence to Virtual Reality: Intelligent Virtual Environments," Applied Artificial Intelligence, Vol. 14, No. 1, 2000, pp. 3-32.

[86]. Ian Somerville, “Software Engineering’ 6th edition, Addison Wesley, August 2000.

[87]. James Reinbaugh, Michael Blaha, “Object Oriented Modeling and Design”, 6th edition, Prentice Hall Inc, April 2000.

[88]. Rober Fautina, “Practical Software Process Improvement,” Artech House Inc, 2005.

[89]. Ronan Doherty, Hugh Outhred and Mark O Malley, “Establishing the Role that Wind Generation may have in Future Generation Portfolios”, IEEE Transactions on Power Systems, Vol.21, No.3, August 2006.

[90] Mehrdad Ghandhari, “ Dynamic Analysis of Power Systems” Part I and Part II, Royal Institute of Technology Sweden 2007.

[91] Lennart Soder, “ Static Analysis of Power Systems”, Royal Institute of Technology Sweden 2004.

[92] Blockset Tutorials of Simpower Systems concerning series compensation demonstrations.