Multi-Agent Organizations for
Real-Time Operations

Sarosh Talukdar, V.C. Ramesh, Richard Quadrel, Richard Christie

EDRC 18-35-92

MULTI-AGENT ORGANIZATIONS FOR REAL-TIME OPERATIONS*

Sarosh Talukdar
V. C. Ramesh
Richard Quadrel*
Engineering Design Research Center, Carnegie Mellon University, Pittsburgh, PA

Richard Christie
Department of Electrical Engineering, University of Washington, Seattle, WA

ABSTRACT
The real-time operations of electric power networks are subject to two sets of forces. The first, including deregulation movements and growing environmental concerns, is acting to increase the complexity of operations. The second, including new computer technologies and emerging knowledge-based agents, provides some means for handling additional complexity. This paper argues that organizational changes will have to be made before the second set of forces can be applied to effectively counter the first. To make this argument, the paper presents a framework for discussing organizational structures. Then it reviews the structures of the two generations of computer-based, multi-agent systems that have been developed for operations. It points out that these structures are well-suited to the algorithmic tasks involved in operations but not to the knowledge-based tasks. The paper concludes with some suggestions for research into alternative structures.

1. INTRODUCTION
Real-time operations--the task of running an electric power network--has three notable characteristics. First, it is complex. Its solution requires the combined efforts of many different types of agents, some human, others computer-based. Second, it is important. The state of a country's economy depends on the quality of its electric service, which in turn, depends on how well its utilities run their networks. Third, its complexity is increasing rapidly. New technologies and socio-economic forces are acting to continually increase its size and add uncertainties.

This paper examines the computer-based, multi-agent systems that are used in real-time operations. The examination will be conducted using a taxonomy (Fig. 1.1) that describes a multi-agent system in terms of its *structure* and *behavior* which are themselves composed of the attributes described below:

---

* This work has been supported in part by the Engineering Design Research Center, a National Science Foundation Engineering Research Center.

* Dr. Quadrel is now with Battelle, Pacific Northwest Laboratories, Richland, WA.

organization: a collection of agents and mechanisms for their interaction,
computing environment: hardware and software for building and using organizations. The environment includes networks of computers, their operating systems and tools that support distributed processing.
openness: the ease with which the structure and behavior of a system can be changed,
performance: how well the systems does its task when all its agents are working normally.
fault tolerance: how well the system does its task when some of its agents are behaving abnormally.

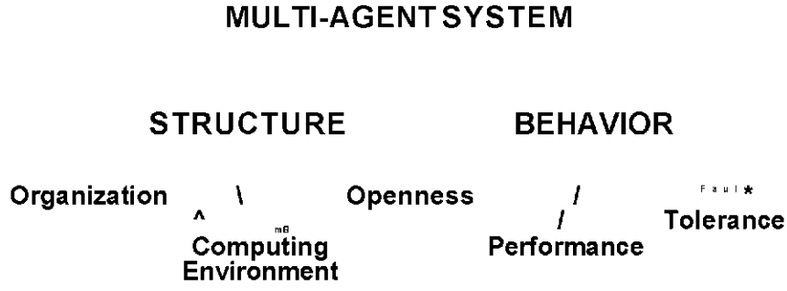

Figure 1.1 Taxonomy of a multi-agent system

The three behavioral attributes (performance, fault tolerance and openness) are conflicting; improvements in one must usually be paid for with reductions of another. In the past, designers chose to emphasize performance and fault tolerance at the expense of openness. As a result, system upgrades have been difficult to make. The trend now is toward more open systems. There is also a trend toward the use of some knowledge-based agents (past systems relied exclusively on numeric agents). The remainder of the paper examines these trends from an organizational viewpoint. The reason for taking this viewpoint is that organizations are as important in determining behavior as computing environments, but have received far less attention.

The remainder of the paper is arranged as follows. Section 2 presents some terms and ideas that are useful in discussing organizations and openness. Section 3 reviews the real-time operating task and traces the development of multi-agent systems from their closed and numeric beginnings to emerging systems that are more open and are starting to accept knowledge-based agents. Section 4 is speculative; it presents our view of some profitable directions for future research. Section 5 summarizes the main ideas of the preceding sections.

2. BACKGROUND

A computer-based, multi-agent system contains both agents and stores. The agents are the embodiment of procedural information. They determine how data is transformed, specifically, how input information is mapped to output information. The stores or databases are repositories for declarative information. A multi-agent system can be built around a single store. However, multiple stores often make more sense: just as complex

problems require too much procedural information to conveniently fit into a single monolithic agent, so also, they require too much declarative information to fit into a single store.

The two structural components of a multi-agent system are its organization and its computing environment. The organization determines what agents and stores will be included, how they will be connected and how they will interact in working on a task. The computing environment provides the means for implementing the organization.

Over the last two decades, computing environments have been growing in capability at an exceedingly rapid rate. Organizations, which are just as important in determining behavior, have been growing much more slowly, and therefore, will be given greater attention here.

2.1 Organization Space

In this part of the paper we will develop a graph-like model of organizations that will allow us to think of each organization as a point in a space with three dimensions: *architecture*, *coupling* and *planning* (Fig 2.1).

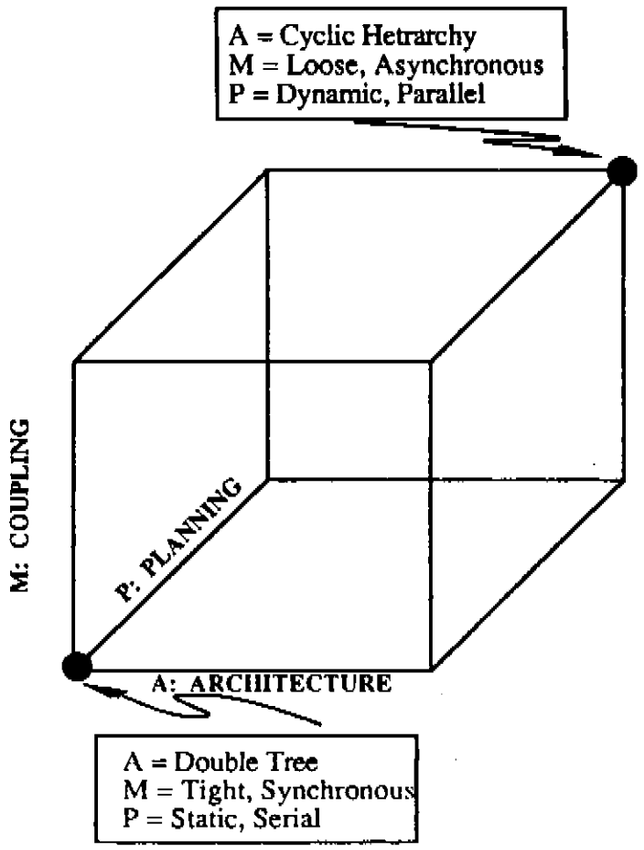

Figure 2.1 Organization space

2.1.1 Architecture

By *architecture* we mean the topology of an organization: the arrangement of its agents, stores and interconnections. These connections serve two purposes. The first is to represent the flow of authority in the organization from supervising agents to supervised agents. The second is for the flow of data. How can these flows be represented?

Standard *organizational charts* have been used by business managers for many years to illustrate the flow of authority among managers and workers. Other techniques have been developed to visualize the architecture of computer-based organizations, such as Petri nets [1], higraphs [2], Communicating Sequential Processing (CSP) [3], and Calculus of Communicating Systems (CCS) [4]. A comparative review of these and some other schemes can be found in [5].

We prefer a combination of organizational charts and higraphs. The former provides a simple way for representing authority flows; the latter, by adding Venn diagrams to graph formalisms, provides an elegant means for visualizing data flows. We call the combination a *TAO* (task-aspect-operator) *graph* [6].

A TAO graph contains two types of nodes (rectangular and round) and two types of directed hyper arcs (solid and broken). The rectangular nodes represent stores or databases. Each store can be thought of as a set of *aspects*. An aspect is a partial description, view or model of any artifact of interest. For example, the topology of a power network is one of its aspects, a table of its line impedances is another. Sets of aspects can be aggregated or disaggregated using the union and intersection operations of set theory and the visual formalisms of Venn diagrams.

The round nodes in a TAO graph represent agents. Each agent can be thought of as an operator that transforms aspects from a set of input stores into aspects of a set of output stores. Solid arcs are used to show the paths along which data flows; broken arcs are used to show the paths along which authority flows. As a simple illustration, Figure 2.2 shows an agent *a* that maps entries from the union of stores X and Z and the intersection of stores W and Y into entries in stores K, L, and M. Agent *b* supervises agent *a*, telling it when to perform the mapping and what elements of X and Z to use as inputs.

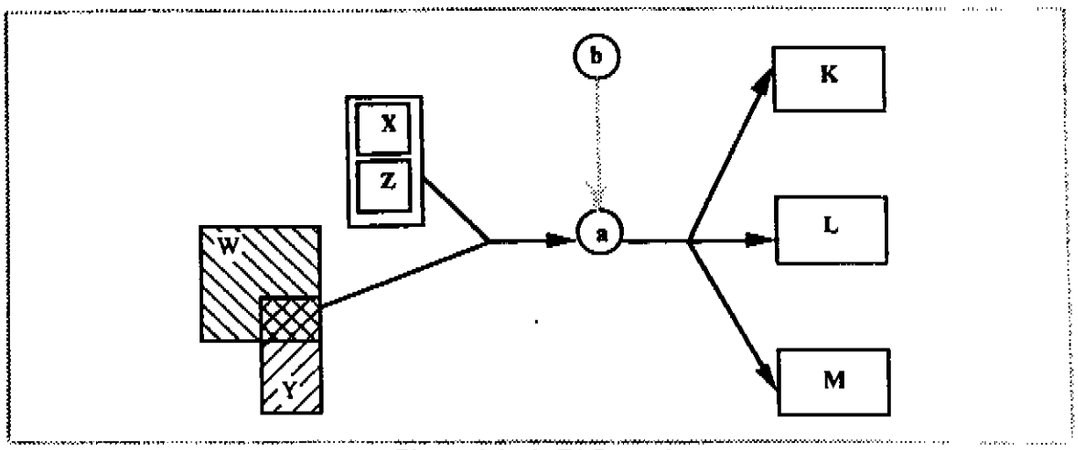

Figure 2.2 A TAO graph

Thus, a TAO graph consists of two subgraphs, one representing the paths of data flow and containing all the solid arcs and all the nodes, the other representing the paths of authority flow and containing all the broken arcs and all the agents. Each of these subgraphs can take on a number of forms that include trees (T), lattices (L), cyclic graphs (C) and null graphs (N). The null graph for data flow is uninteresting - it represents a trivial case where no computations can take place. However, in the case of authority flow, the null graph denotes a "hetrarchy": an arrangement without bosses where all the agents have the same status.

Let A = [x,y] be a double whose first element, x, denotes the form of the information flow, and whose second element, y, denotes the form of the authority flow. The twelve nontrivial possibilities for A range from [T,T] which we will call a *double tree*, to [C,N] which we will call a *cyclic hetrarchy*. In the double tree, all but one of the agents is supervised. In a cyclic hetrarchy, none of the agents is supervised; they are all autonomous. Illustrations of these two extremes and some of the intermediate forms are given in Fig. 2.3.

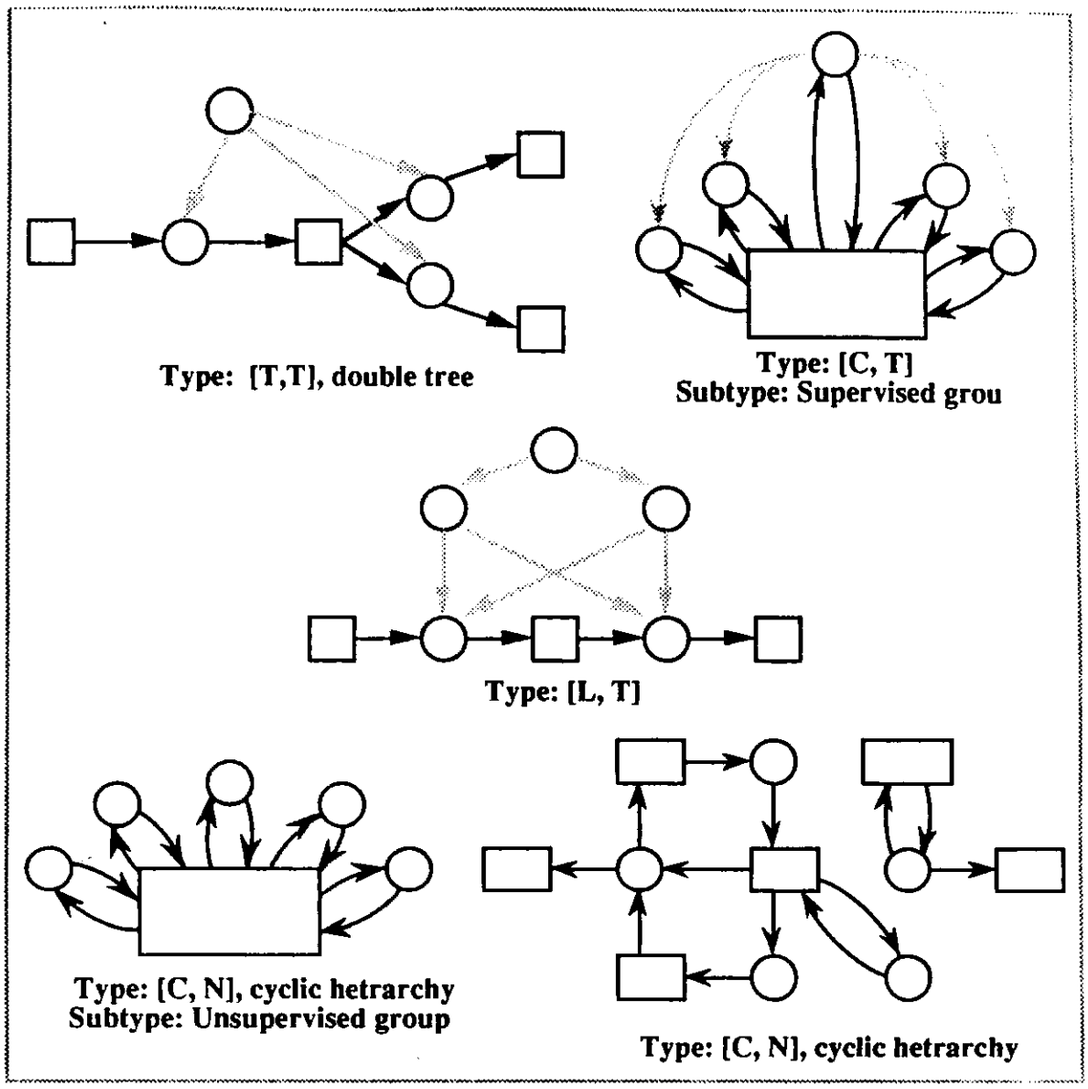

Figure 2.3 Some Architectures

2.1.2 Coupling

Whereas architecture locates the links among agents and stores, *coupling* is concerned with the natures of these links. As such, coupling is a subset of what has often been called "communications" in discussions of distributed problem solving [7]. Communication can be defined by the answers to the following questions [7, 8, 9, 10]: Who speaks? To whom? What is said? When? With what duration and frequency? By what decision procedure? Through what medium?

All of these are organizational issues except the last, which is concerned with how communications are actually realized and therefore belongs to the computing environment. Who speaks to whom is determined by the locations of the data links, that is, by architecture. The remaining issues, particularly how they affect the activities of agents,

come under the heading of *coupling*.

Coupling is said to be *loose* if agents spend the bulk of their time computing and only a small fraction of their time communicating with one another [11]; otherwise it is *tight*. If communications must occur at prespecified points in the computational process, coupling is said to be *synchronous*; however, if agents are allowed to communicate opportunistically and at their convenience, then coupling is said to be *asynchronous*. Naturally, one would prefer coupling to be loose and asynchronous so agents can spend most of their time computing rather than communicating or waiting for communications.

2.1.3 Task planning and execution

A task can be thought of as a transformation of given information into desired information. In TAO graph terms, the given information corresponds to given aspects, each in a given input-store. The desired information corresponds to unknown aspects, each in a given output or goal-store. Thus, the conditions necessary for a system to be able to perform a task are: first, the system must contain the appropriate input and goal stores, and second, there must be at least one path in the data flow (through solid arcs) linking these stores.

The complexity of a task has at least two dimensions: size and uncertainty. Size refers to the amount of computation that must be done in journeying from the input to the goal-stores, and has been extensively discussed in the computer science literature, see [12], for instance. Uncertainty refers to deficits in the information needed to plan the journey from input to goal-stores [13]. Loosely speaking, uncertainty refers to the things that can go wrong (called contingencies) in the journey, such as the failure of a numerical agent to converge, the lack of accurate predictions for crucial exogeneous variables, or the loss of some important piece of information. (We will refer to these computational contingencies as *C-Type contingencies* to distinguish them from another type of contingency that will be mentioned later.)

The purpose of task planning is to decompose complex tasks into simpler subtasks, to assign these subtasks to agents, and to schedule their execution. The processes involved can be viewed along three dimensions. The first deals with the amount of feedback involved, the second, with the type of cooperation among agents, and the third, with the amount of parallelism in agent activity.

With respect to feedback, planning processes can be divided into two categories: *static or open loop* and *dynamic or closed* loop. In the static case, plans are completed and frozen before the subtasks are executed. In the dynamic case, the planning and execution activities are allowed to overlap; results from the partial completion of tasks are fed back to make adjustments and corrections in the plans. Dynamic planning can further be divided into centralized and decentralized. The former concentrates planning decisions in a few supervisory agents, the latter distributes the planning over the entire set of agents.

The type of cooperation among agents [7, 14, 15, 16] can range from antagonistic (agents deliberately impede one another) through neutral (agents may commit errors but do not deliberately impede one another) to synergistic (the capabilities of the system as a whole is

greater than the sum of its parts). If intent is disregarded, then the antagonistic-to-neutral spectrum can be folded into the neutral-to-synergistic spectrum by treating the actions of antagonistic agents as errors or faults.

Finally, the range of possibilities in scheduling and executing subtasks extends from serial execution through slightly overlapping subtasks to fully concurrent execution .

2.2 Observations on Organizations

• Static planning offers only two alternatives, neither attractive, for handling the large numbers of C-Type contingencies that result from highly uncertain tasks. The alternatives are: anticipate the contingencies and make plans for all of them in advance, or make plans for only some of them, leaving the others uncovered. In contrast, dynamic planning allows responses to contingencies to be developed "on the fly." In dynamic planning one need consider only imminent or actually occurring contingencies--usually a very small fraction of all possible contingencies. In other words, highly uncertain tasks call for dynamic planning.

• Architectures with tree-like data flows are inappropriate for dynamic planning. The reason is that trees provide no more than one path to each goal store. If this path fails, the only thing that dynamic planning can do is decide to retry it. To take better advantage of dynamic planning one needs multiple paths so the best one can be chosen during execution.

• If planning is centralized in a few supervisors, then these supervisors must be modified to handle the new C-Type contingencies and to take advantage of the new functionality brought by the addition of any new agent. These modifications can, and usually do, take a large amount of effort. In other words, supervisors are often the bottleneck in expanding the agent set.

2.3 Openness

How does one define a system to be "open" or "closed?" Loosely speaking, a system is open if its structure and thereby, its behavior, can be easily changed. For greater precision, one must look to the different ways in which these changes can be made. Fig. 2.1 and the material above indicate that the organizational structure can be changed along three principal dimensions by varying:

- the set of agents,
- the set of stores and representations,
- the flows of authority and information,
- the process of decomposing tasks into subtasks,
- the assignment and scheduling of subtasks, and
- the communication mechanisms that come under the heading of coupling.

Similarly, the structure of the computing environment can be changed along the dimensions of its space of alternatives by varying the computers used, the network that connects them, their operating systems, etc. It follows that a perfectly open system is one whose structure can easily be changed along every dimension of the organization space and computing

environment. However, most definitions of openness allow for far less structural variation. For instance, Hewitt defines an open system as one whose agents are simple; whose planning process is decentralized, able to handle inconsistent information and allows for concurrency; whose communications are asynchronous; and whose behavior is fault tolerant [17, 18]. The underlying assumption in the selection of any such subset of structures is that it maps into the set of all desirable behaviors. While we believe that this assumption is largely true for the subset prescribed by Hewitt, it is certain that there are a significant number of engineering tasks whose evolution will call for a trajectory that lies outside this or any other small subset of all possible organization and distributed computing alternatives. As another example, the ISO (International Standards Organization) has taken a prescriptive approach by defining a standard for the interconnection of computer systems, referred to as the *Open Systems Interconnection* (OSI) [19]. This standard divides systems into seven layers and seeks to make changes easy by preventing the effects of any change from propagating beyond the layer in which the change is made. As such, this initiative deals only with the distributed computing environment and does not consider organizational issues.

In the power system community, the definition of an open system is still evolving. Originally, when systems for real-time operations were proprietary, an open system was thought to be one where complete design information--specifications, drawings and source code--were public knowledge. Later, the term "open" came to mean a system that subscribed to a set of widespread and generally accepted standards, such as UNIX and TCP/IP. Recently, the term has come to mean "modular interoperability." The vision is to define standard interfaces for each type of hardware and software module so that they become "upwardly compatible." This approach will accommodate the immediate need of utilities to replace existing modules with better versions as they become available but does not look far enough ahead to where more drastic structural changes will be necessary.

To be of more than passing interest, the definition of an open system should be neither confined to a small subset of structural alternatives nor limited to prescriptions based on currently available technologies. We propose that openness be represented by a vector, in which each element corresponds to one structural dimension. Such a vector would allow one to characterize systems that are open along some dimensions and closed along others. For instance, one could have a system that is open to new agents and computers but closed to new stores and planning processes.

3. REAL TIME OPERATIONS

The purpose of an electric utility is, of course, to deliver electric energy to its customers. Bulk shipments are carried over high voltage transmission networks while final deliveries are made over low voltage distribution networks. Decisions on how to run the transmission network -- how much energy should be produced in the utility's own generators, how much should be bought from other producers, and over which routes the energy should flow -- are made in process control centers called EMSs (Energy Management Systems). These centers collect data and implement decisions in real time through sensors and actuators that are distributed over the transmission network and generating stations. The decisions themselves are made by a collection of human and computer based agents in the

EMS. In this section, we review the operating task of the EMS and describe its evolution.

3.1 Normal Behavior and N-type Contingencies

The behavior of a power network is characterized by three quantities: first, the configuration or topology of the network, second, the values of a set of exogenous variables most of which represent randomly changing customer demands, and third, the value of a state vector whose elements are the network's node voltages, line flows and control setpoints.

The space of all possible behaviors can be divided into two regions: *normal* and *abnormal*. In the normal region, essentially all of the network's elements are energized. The state vector satisfies a set of constraints reflecting equipment ratings, prescribed safety margins and prevailing standards for customer quality-of-service. The network is in a quasi-steady state, meaning that the state vector changes continually, but at so slow a rate that steady state models can be used to calculate its value. Networks spend most of their time in such normal regions. Their rare excursions into abnormal regions usually result from sudden changes in configuration caused by equipment failures. (Henceforth, we will refer to such configurational changes as *N-type contingencies* to distinguish them from the computational or C-type contingencies mentioned earlier.) The occurrence of an N-type contingency precipitates dynamics that can overstress equipment causing further outages and failures. The actual outcome depends on both the initiating contingency and the incumbent state of the network. In the worst case, the cascade of dynamics and outages leads to a total blackout. Recovery from a blackout or even the outage of a large portion of a network is a tedious process that can take hours and sometimes even days.

3.2 A Taxonomy of EMS Tasks

The overall task to be performed by an EMS is to steer the associated power network along a trajectory that remains within the normal region and keeps operating costs low. This task is complex in terms of both size and uncertainty. The size of the task is reflected in the thousands of nodes, thousands of decision variables (many of them discrete), and tens of thousands of constraints of the typical power network. The main source of uncertainty is the set of N-type contingencies. If one assumes, as many utilities do, that the probability of two pieces of equipment failing simultaneously is negligible, then the set of N-type contingencies has a few thousand entries (one for each major piece of equipment). However, many of the crises of the past have been precipitated by "double failures." Typically, a protective device, such as a relay, fails but the failure goes undetected till the relay is called on to respond to some other failure. The inclusion of such "double failures" swells the set of N-type contingencies considerably.

The overall EMS task can be decomposed into the following major subtasks:

1. Data acquisition: collecting measurements from sensors distributed over the power network.
2. Topology Analysis: determining the current configuration of the transmission network.

3. State Estimation: determining the current value of the state vector.
4. Forecasting: predicting customer demands.
5. Security Assessment: identifying a subset of critical N-type contingencies and assessing their impacts.
6. Transaction Planning: negotiating the amounts, times and prices of energy exchanges with other utilities.
7. Generator Control: deciding when generators should be started, at what levels they should be operated, and when they should be stopped.
8. Network Control: selecting breaker positions as well as values for all the other discrete variables, such as transformer tap settings, in the transmission network.

Each of these subtasks can be further decomposed into the following categories:

• *algorithmic task* : a problem that is well posed in the sense that it can be formulated in precise mathematical terms and moreover, at least one algorithm is available for solving most of its practical instances. Power flow calculations are an example. These calculations are undertaken to determine the state vector given the network's configuration and node injections. The fast decoupled algorithm works on most cases of interest in operations.

•*knowledge-based task* : a problem that is often ill posed and whose solution can be obtained by the application of existing knowledge. This knowledge may occur in either explicit or tacit forms. By explicit knowledge we mean information that has been encoded in some permanent and accessible medium, such as a manual or a computer program. By tacit knowledge we mean information that exists only in the minds of some human experts.

•*team task* : a problem whose solution requires the combined efforts of several algorithmic and/or knowledge-based approaches. Optimal power flows are an example. These calculations are undertaken to minimize some cost function while satisfying a number of network and customer constraints. There are many applicable algorithms, including interior point methods, successive linear programming, and successive quadratic programming. None provides consistently good solutions or is robust enough to handle the range of problem instances that is of interest in operations. But under the control of a competent numerical analyst, combinations of algorithms can often be devised to extract the desired solutions.

• *insoluble task* : a problem that cannot be solved today.

3.3 EMS Evolution

EMS evolution has progressed through two generations that we will describe in tern is of the taxonomy of Fig. 1.1. Later we will discuss the conceptual design of a third generation.

3.3.1 The First Generation

First generation EMSs were developed and installed from the '60s to the mid '80s. Most existing EMSs are of this type.

Cnmputinf Environment

The earliest EMSs used a single mainframe specialized for power system requirements. This configuration was almost immediately replaced by one with dual redundancy - the complete duplication of the processing hardware for greater fault tolerance. Most of the first generation EMSs still in use arc dual-redundant. A second configuration change was driven by the need to increase performance without replacing the mainframe computers. Additional processors were added to deal with easily separable tasks. In "front-ended" designs, a number of mini or micro processors each handle part of the communications load, passing data to the large main processor. In "back-ended" designs, one or two large processors, often of the same type as the main processor, perform the numerically intense task of security analysis. The specific form and number of additional processors are set at design time and vary from one installation to another. Figure 3.1 shows a typical dual-redundant front- and back-ended configuration.

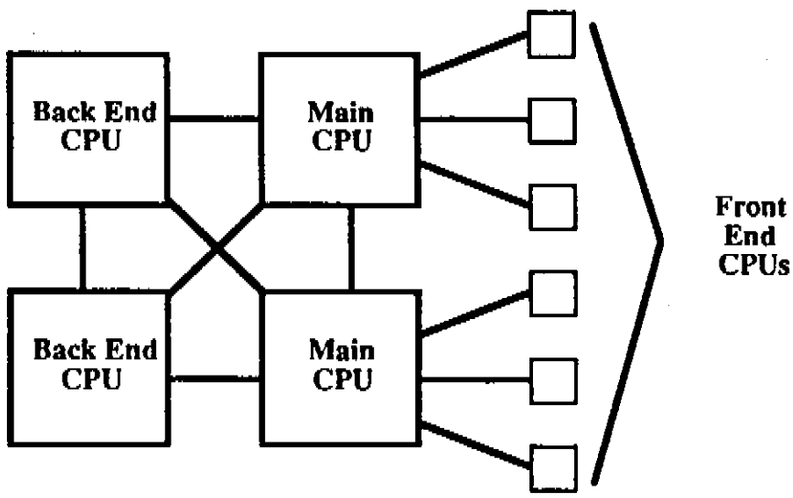

Figure 3.1 A typical front and back-ended EMS configuration.

Organization

Architecture : In first generation EMSs, each task is performed by one agent, and only by that agent. For example, all user interface screens are generated by one program. A complete system typically has about fifty agents. Most of the agents are activated periodically by a scheduler, a simple supervisory agent. Some are activated by other agents under certain conditions, and the remainder are organized into pipelines, or chains of agents that operate in sequence. Each agent in the pipeline activates the next agent when its own task is complete. The first agent in the pipeline is usually initiated by the scheduler. The authority flow graph is therefore a pure tree with a large branching factor at the first node (the scheduler), and low branching factors thereafter (the pipelines).

Data is centralized in a single store and the data flow graph is fundamentally a tree. Exceptions appear when agents store state information, reading and writing the same data items, adding some cyclic features to the tree, or when a data item can be set by se\ cral

agents, making that portion of the data flow graph a lattice. These exceptions are minor, however, and the first generation EMS architecture is best described as a double tree ([T,T]).

The complete TAO graph for such a system is too large to show here. A representative portion (the security assessment function) is shown in Figure 3.2.

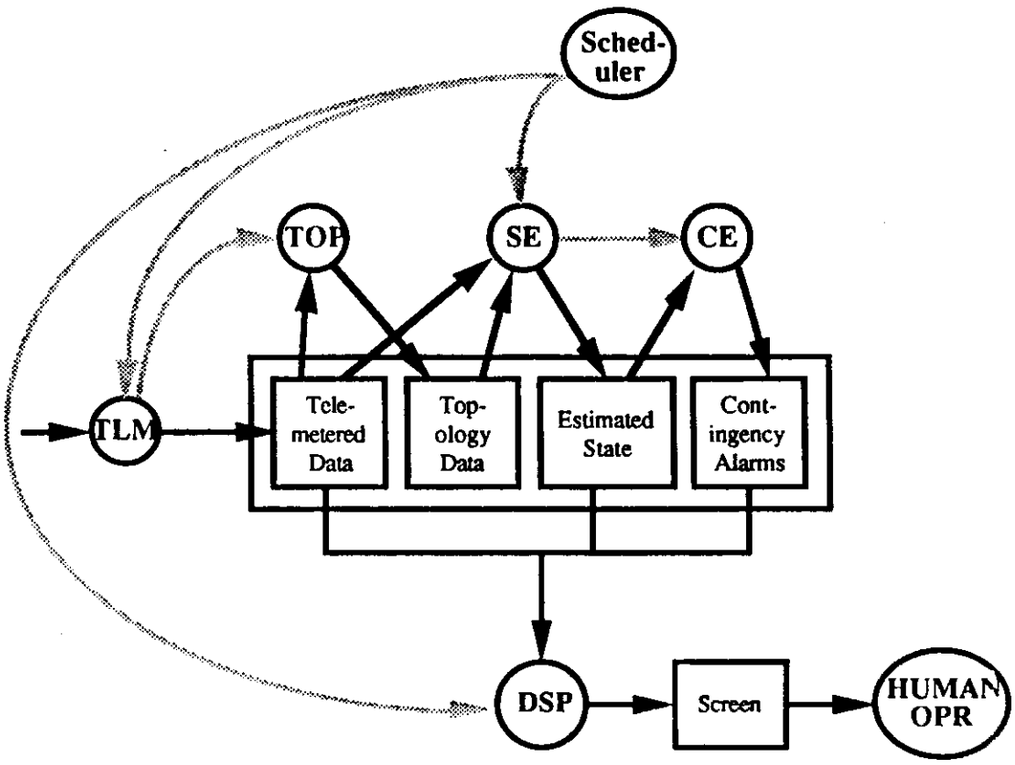

Figure 3.2 A partial TAO graph of security assessment in a first generation EMS. The Telemetry agent (TLM) acquires data from the real world and stores it. Certain data changes cause the Topology agent (TOP) to process network topology and store it. The State Estimation agent (SE) runs periodically to translate whatever telemetered and topology data is present into a state estimate of the power system. The new state estimate then triggers the Contingency Evaluation (CE) agent, which identifies contingencies that it thinks could cause problems and stores these problems as contingency alarms. The Display agent (DSP) periodically translates data from any of a number of stores into a display screen store that a human (HUMAN OPR) can examine and use to make decisions.

Coupling : Coupling between agents in the EMS is tight and synchronous. Coupling is implemented primarily by message passing. It is highly restrictive. What can be said, indeed, what must be said, and who says it to whom are strictly fixed. The times when things can be said, their duration, frequency and transmission medium are more flexible, but still carefully controlled at design time. Also, the data acquisition task (acquiring data from remote sensors) involves coping with metering inaccuracies and communication delays. Much of the work done in state estimation is to compensate for these errors in data

acquisition.

Planning : Task planning is static and serial. The agent that performs any specific task and the control that initiates it are determined at design time. The form of the control rarely extends past periodic or event driven initiation, and almost always occurs without regard to either the status of the other agents in the EMS, or to the status of the power system.

Openness

First generation EMSs have a "closed" computing environment. The hardware and systems software (the operating system, input/output interfaces etc.) are proprietary to the vendor and often incompatible with systems from other vendors. The reasons for the adoption of such an environment were two-fold. One was that such systems were all that the computer industry had to offer till the late 1970s. Another was the real time nature of EMS operations. Real time systems impose hard deadlines on the completion of tasks, and vendors were often forced to develop special hardware and systems software to ensure these deadlines were met. Vendors also had no motivation to adopt universal standards and sell compatible systems. Understandably, they preferred to sell an entire EMS, using the attractive features of their offering to leverage the entire package and discourage the inclusion of third party products. Many of these forces are still at work today.

Besides the difficulties with proprietary hardware and software, first generation environments were expensive to upgrade. Hardware could only be replaced in large pieces. The high cost involved made utilities postpone EMS upgrades as long as they could, and they were therefore unable to profit from the rapid improvements occurring in computer technology. Another problem was that purchasers were left stranded when vendors went out of business. Finally, the specialized operating systems, user interfaces, and other system software did not facilitate maintenance.

First generation EMSs were also "closed" from the organizational point of view. The double tree architecture made it very difficult to upgrade power system software applications. Every time agents had to be added, deleted or modified, the scheduler or supervisor and a number of other agents that interacted with the affected agent also needed modification due to the close coupling between the agents and to the static task planning.

Performance

A fundamental measure of the performance of EMSs is the quality, reliability and economy of electric power service in the systems they control. By this measure, first generation EMS performance is superb. Remarkably few service failures have occurred, and recovery has been rapid. The cost of electric power has been driven primarily by fuel prices, and not by delivery costs. EMSs are believed to permit deferment of large capital cost projects such as new generation or transmission facilities.

In terms of the major subtasks mentioned in Section 3.2, first generation EMSs do a good job in data acquisition, topology analysis and state estimation, and a satisfactory job, limited by input data accuracy, in load forecasting. They also perform significant parts of the transaction planning, security assessment and generation control tasks. However, on

these last three tasks, they perform well only while the system is in the normal region. Also, they use numeric algorithms that are greedy, that is, algorithms that are unable to sacrifice a small gain in the short term for a much bigger gain in the long term.

EMS performance of all tasks degrades substantially during abnormal situations, because the time scale of events is usually much shorter than EMS cycle times. During dynamics, human agents in control centers with first generation EMSs get very little assistance.

Fault Tolerance

The dual-redundant hardware configuration of the first generation EMS has provided excellent hardware fault tolerance, with availability rates in the very high 90 percent range routinely specified and achieved. Fault tolerance for agents is, in contrast, almost non-existent. The organizational structure requires each agent to successfully complete its task. Agents that fail often result in a complete system reset. Such failures are weeded out during system build time for normal operation, but are likely to appear unexpectedly during abnormal operation.

3.3.2 Second Generation EMSs

Second generation EMSs are those developed and installed over the last five years. The major (and perhaps, the only) changes from the first generation EMSs are in the computing environment.

Computing Environment

The hardware organization of the second generation EMS is distributed, based on the concept of many processors performing different tasks [20]. These systems consist of multiple processors that are the nodes of a local area network. For each processor there is at least one other processor that can take over the first processor's tasks should it fail. Since the system can operate with any one processor out of service, this is termed *N-1 redundancy*.

Nodes interact in a uniform way with the network, and no one node is the master. Most nodes have homogeneous hardware, using identical processors and other components, but heterogeneous functionality, with different hardware assigned to support the node's specific task, such as external communications line connections, user interface hardware or disk drives. In some cases, specific tasks are implemented on heterogeneous processors, usually for computational power. There are many more processors than in the first generation EMSs, and each does a smaller portion of the overall task.

The processing nodes themselves, and the network, are often, but not always, implemented with industry standard, general purpose hardware, such as workstations and local area networks.

Organization

Changes in the computing environment have not resulted in a drastic change in the organizational schemes employed in the second generation EMSs.

Architecture : Several agents are now employed to do the same task that was done by a single agent in the first generation. This implies more modularity in the organization with more disaggregation of agents. The form of the authority graph now has a few lattice structures, but it is still fundamentally a tree. The system is now physically a multi-store system, with stores distributed among the processors, but they are made to appear to the agents as though they are one central store, with a similar store structure and similar store-agent interactions. The data flow graph remains practically unaltered.

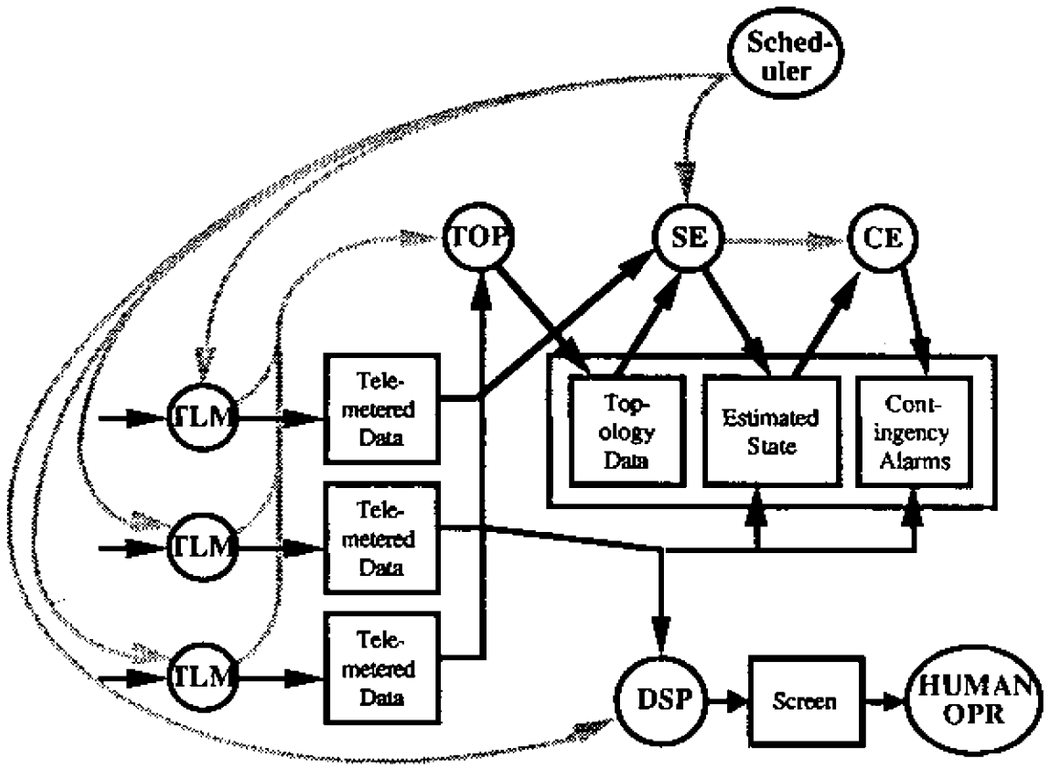

Figure 3.3 A TAO graph of security assessment in a second generation EMS. There are now several TLM agents, each with its own store, and the TOP and SE agents acquire data from each of them. The chain (SE, CE) is unchanged because all of these agents are implemented in one processor with a central store for data used in that processor. The DSP agent appears unchanged, but in the complete TAO diagram there would be a number of DSP agents, where in a first generation EMS there would be only one.

Coupling : Coupling among agents remains tight. There is more interaction among agents as a natural consequence of their distribution in the hardware, but the coupling mechanisms -- primarily message passing -- are unchanged and remain synchronous.

Planning : Task planning remains static and serial. Although multiple agents perform similar tasks, they are not generally interchangeable. Their parts of the task -- primary and backup -- are assigned at design time, and are not altered during execution.

Openness

In these systems the computing environment is essentially "open." It is much easier (compared to first generation systems) to add new hardware, and to incrementally improve the processing capacity of a part of the environment without drastically affecting the rest. Many of these systems also adhere to existing operating system and networking standards developed by the computer industry, such as Unix and TCP/IP (Ethernet). Their use of general purpose processors increases the amount of off-the-shelf systems software (like compilers, debuggers, and user interface packages) available. These systems are more fault tolerant to hardware failures since they employ more processors. They also provide better performance of some services than the first generation systems, primarily due to improvements in computer technology (faster processors, for example).

The open computing environment eases the hardware and some of the system software difficulties when modifying existing agents, but the close coupling and increased complexity present some counterbalancing factors and the essentially unchanged organization of the system remains restrictive. The net effect appears to be an improvement in ease of modification, but not a complete or satisfactory solution. While the number of agents has increased, their problem solving behavior (predominantly numerical) remains unaltered. As will be described in section 3.4, there has been some experimentation with a few knowledge-based agents; but most of the installed EMSs remain based on numerical agents.

Performance

Second generation EMSs have not been in service long enough to evaluate performance measures based on power system operation. Certainly no major power system problems due to second generation EMSs have arisen during this short time.

The scope of tasks performed by software agents in second generation EMSs has remained unchanged, and many of the agents themselves are unchanged, so the evaluation and limitations of their performance is also unchanged from that of the first generation. Human operators are still responsible for major portions of the network control, transaction planning, generation control and security assessment tasks. The same greedy algorithms are still employed, and EMS performance has not been extended to cope with abnormal events.

Fault Tolerance

Hardware fault tolerance in the second generation is similar or superior to that of the first but is achieved at a significantly lower cost; dual redundancy requires one half of the hardware in standby, whereas N-1 redundancy requires only 1/Nth of the hardware to stand idle.

Tolerance of agent faults remains non-existent, as would be expected from the unchanged organization. Perhaps the only improvement is that agent failures restart only the affected processor, not the entire system.

3.3.3 Observations

Functionally, the first and second generation EMSs are closed. They perform the algorithmic tasks well (data acquisition, data processing, and analysis in the normal operating state), but have not been designed to handle knowledge-based or team tasks. As a result, they can offer little prescriptive help, especially in abnormal operating states.

Both first and second generation EMSs have essentially the same organizational structure (double tree architecture, tight coupling among agents and static planning). This puts them near the lower left corner of the organization space (Fig. 2.1). Organizationally, they are closed; the addition of new functionality (i.e, new agents, new stores and new planning processes) is difficult.

The computing environment of the second generation is quite open, especially to new computers, but less so to new operating systems and network protocols.

Tolerance to hardware faults is more than adequate in both generations. Tolerance to agent faults and errors (C-Type contingencies), however, is nonexistent.

3.4 Trends and Needs

The future of real-time operations will be determined by two trends: the first, making the operating task more complex; the second, providing solutions for the additional complexity. Any gap between these trends will translate into a decrease in the quality of electric service. Some of the constituents of the first trend are:

- Customer demand is growing faster than network capacity. As a result, systems will have be operated closer to the limits of their operating envelopes. Behavior in these border regions is not well understood. However, it is certain to be more complex. For instance, the effects of N-Type contingencies will probably spread much further and influence more distant utilities than they now do.
- New technologies, such as FACTS (Flexible AC Transmission Systems), will further complicate power system behavior.
- The unbundling of services and the increase in competitive pressures being brought about by deregulation will force profound changes in operating practice. For instance, the familiar notions of costs and losses will probably have to be replaced by the unfamiliar and more complex notions of revenues and profits as operating objectives. The result will be an increase in the number of ill posed tasks.
- NUGs (Non-Utility Generators), load management technologies, and other artifacts of deregulation are proliferating. As a result the numbers of decision variables, constraints, and random exogeneous variables will increase.
- Environmental concerns and constraints are increasing in importance and number.

The net effect will be an increase in both the size and uncertainty of the real time operating task. Human agents cannot be expected to carry the entire burden. Instead, the computer

based system will have to carry more of it. This will involve new functions. In particular, it would seem that the computer based system will have to take more responsibility for prescription and develop abilities to better handle abnormal circumstances. How do these needs match with ongoing developments? The main constituents of these developments and their trends are:

Distributed Computing

The components of distributed computing environments are growing in capability at rates that approach several percent per month. This includes not just hardware but also operating systems [21, 22] and higher level tools [23, 24].

Standards and Industrial Initiatives

Utilities have long sought to obtain "modular interoperability" -- the ability to replace any existing agent with a new one that performs the same function in a better way [25, 26, 27]. In order to satisfy this desire, an IEEE task force is working to identify standards for communications, database interfaces and user interfaces. The Electric Power Research Institute (EPRI), in the first phase of a related effort called the " Integrated Utilities Communications Project" is designing a "Utility Communications Architecture" (UCA), based on the OSI, and Database Access Integration Services (DAIS) [28, 29]. Such efforts, if successful, will provide modular interoperability, making it much easier to improve the performance of existing functions, but not necessarily to add new ones.

Better Numerical Algorithms

Numerical algorithms for EMS tasks have been evolving over several decades and will continue to do so; existing algorithms will be continually refined; and new algorithms will appear periodically. For many tasks there are now a multiplicity of algorithms. For instance, the optimization of continuous generator and network variables can be done by either successive linear programming or successive quadratic programming. Since no algorithm is without its weaknesses, it would be desirable to combine algorithms into groups where they can cooperate in finding solutions to common problems. Though such cooperative arrangements are beginning to be developed [30], no existing EMS organization has the capabilities needed to implement them.

Knowledge-Based Agents

Many operating tasks are ill posed and require knowledge-based approaches. Some of the knowledge is explicit and expressed in manuals. However, much of it is tacit and stored only in the minds of certain humans. The last ten years have seen large efforts to build expert systems that capture extant or freshly produced knowledge and apply it to tasks that are numerically intractable or ill posed [31]. A good survey of knowledge-based applications in the power area can be found in [32].

Of the eight EMS tasks mentioned earlier, topology analysis, state estimation and security analysis have received the most attention [33, 34, 35]. Alarm processing and fault diagnosis are the two areas where knowledge-based approaches have been most extensively researched. The goal of alarm processing is to provide operators with a concise summary of alarms by suppressing redundant alarms [36] and prioritizing alarms [37, 38].

Knowledge-based systems have also been developed for identifying the type and location of system faults [39, 40, 41, 42, 43]. Steady state security assessment [44, 45, 46, 47] is another area where knowledge-based systems are being developed for providing on line assistance. One of the earliest knowledge-based systems for power systems operations was developed in the area of restoration (generation and network control) [48]. Since then, quite a few other systems have been developed for assisting operators in quickly restoring the power systems after an outage [49, 50, 51]. Some work is also being done in using knowledge-based assistance for transaction planning, real time corrective control, and load forecasting [35].

The result of all this work is a host of knowledge-based agents, many of which could be of great use in real time operations. However, the actual penetration of knowledge-based agents into EMSs is vanishingly small; the three main reasons are all related to existing EMS organizations.

First, the organizations of EMSs are relatively closed to new types of agents; a great deal of effort is usually required to insert them. This is especially true for the first generation EMSs that constitute the majority of installed systems, and hence the majority of target systems for knowledge-based agents. Second generation EMSs present fewer, but still significant, installation difficulties.

Second, EMS organizations use static planning and hence are unsuited to dealing with errors made by agents (C-Type contingencies). Therefore, the organizations dictate a very conservative approach to new agents; these agents must be extensively tested and verified before being put on line, a process that can take years, and is particularly difficult for knowledge-based agents.

Third, many of the tasks addressed by knowledge-based approaches are of the team variety; they seem to require the combined efforts of several algorithmic and knowledge-based approaches. Existing organizations have no capabilities for coordinating the efforts of several agents in solving a single and common task.

To summarize, the net effect of the above developmental trends is likely to be a continual improvement in the performance of existing functions. However, increasing task complexity, calls for the incorporation of new functions to better handle the knowledge-based and team components of operations. These new functions will remain difficult to incorporate until existing organizations are changed to make them:

- more open to new types of agents, particularly, knowledge-based agents that have not, and perhaps cannot, be completely verified;
- better able to mount coordinated efforts in which several algorithmic and knowledge-based agents cooperate, helping one another in solving a common task.

These needs would seem to require organizations that are capable of dynamic planning, which in turn, will require architectures with higher connectivity (more loops and alternate

paths) than the trees and pipelines of existing EMSs.

4. IDEAS FOR BETTER ORGANIZATIONS

What organizations should future generations of EMSs use? Unfortunately, there are no systematic approaches for developing an answer. In fact, there are no systematic tools for either the synthesis or analysis of organizations. Instead, the organization-builder must rely on intuition and ideas borrowed from wherever they can be found. Two well known sources of ideas are distributed artificial intelligence and human organizational theory [15, 52, 53, 54, 55]. Recently, a third source has come to light-insect societies and related animal organizations, such as flocks of birds and schools of fish. In the remainder of this section we will examine three cases, one from each source, that seem to lead from the region in organization space (Fig. 2.1) currently occupied by EMSs to a better region.

4.1 Blackboards

In essence, a blackboard is a block of global memory that is shared by a number of programs. These programs communicate by posting messages on, or reading messages from the blackboard. Access is controlled by a supervisory program. Hearsay-II, a package for speech recognition [56], first referred to the shared memory as a "blackboard" and used it for dynamic planning. Since then, blackboards have proliferated in artificial intelligence and engineering applications [57, 58, 59].

Organization

Architecturally, a blackboard is a supervised group with a central store (Fig. 2.3). Access to the store is controlled by the supervisor. In all other respects, the agents are autonomous. Every time an agent is given access to the store, the agent modifies the store's contents. The supervisor's job is to determine a sequence of accesses that will transform the store's contents from an initial state into a goal state. The supervisor determines this sequence opportunistically. In other words, planning is dynamic and centralized in the supervisor. Note that a blackboard has the same architecture as an EMS (Fig 3.2). The differences are in the blackboard's use of agents with more autonomy, and in its use of dynamic instead of static planning.

Pehavior

Blackboards, because of their use of dynamic planning, are good at handling uncertain tasks. In principle, the set of agents can be made large enough to cover every contingency. Since agents are invoked (given access to the store) only as needed, the presence of an excess of agents need not slow down the completion of a task appreciably. Blackboards are also fairly open to new agents. To add an agent, one need merely provide it with the means to read from, and write to, the store. However, for the new agent to be useful, the supervisor must know when to invoke it. This centralization of planning in the supervisor is a serious bottleneck. Another bottleneck results from the use of a global store; the insertion of new representations into this store can be difficult.

4.2 Scientific Communities

Humans have been experimenting with, and thinking about, organizations ever since they first began to live and hunt in groups. The results cover most of organization space (Fig.

2.1). Among the best known are the multi-layered hierarchies that apply thousands of humans to solve exceedingly complex problems, such as the design of the space shuttle. However, such hierarchies are difficult to replicate in software; several layers of supervisors only compound the bottleneck created by the single supervisor in a blackboard system.

Kornfeld and Hewitt have pointed out another type of human organization-scientific communities — that contains no supervisors and is an interesting metaphor for software systems [60].

Organization

One can think of a scientific community as a cyclic hetrarchy (Fig 2.3) in which the agents are autonomous groups of scientists and the stores contain the types of data (observations and theories) that are relevant to their area of interest. Planning in a scientific community is dynamic (scientists react to the latest available data), tends to be decentralized (each group determines its own course of action) and data driven (to be objective, scientists must follow the evidence wherever it leads). Groups work in parallel and the community tolerates and indeed, thrives on, diversity of opinion [60].

Behavior

Kornfeld and Hewitt note: "That scientific communities are successful at generating and deciding between alternative explanations for phenomena is indisputable. Scientific progress, looked at globally and with a time scale of many decades, seems coherent and purposeful. Looked at locally, this is anything but true. At any one time many conflicting theories may purport to explain the same phenomena. Occasionally what may ultimately turn out to be the wrong party to a dispute will gain temporary popularity; though the fields themselves seem to grow in depth and power over the long haul" [60]. In other words, scientific communities perform well and are fault tolerant. We may add that they are also open to new agents; often the desire to participate is all that is required for a new agent to be included in the community.

4.3 Insect Societies

Social insects—ants, termites, certain bees and certain wasps—have evolved organizations by which they perform quite complex engineering tasks. These organizations are in many respects, similar to those of scientific communities. The main difference is that they do not need agents as intelligent as scientists. Instead, they use agents (ants, termites, etc.) with learning, reasoning and communication skills so modest as to be within easy reach of computers.

Organization

Like scientific communities, insects use hetrarchical architectures. Certainly, there is a "queen", but her function is strictly reproductive: she does not lead the other colony members, nor does she issue orders to her workers. Although different "castes" exist within the colony (e.g., drones, soldiers, workers, queens), there is no hierarchical relationship among them [61]. Rather, insects act as autonomous agents.

Like scientific communities, social insects use dynamic planning processes that are decentralized, rely heavily on concurrency and allow for diversity of opinion. As Wilson notes [62],

> "The individual member of a large colony cannot possibly perceive the actions of more than a minute fraction of its nestmates; nor can it monitor the physiological condition of the colony as a whole. Yet everything balances out, a fact that keeps drawing the mind back to Maeterlinck's poetic question about the termite colony: 'What is it that governs here, issues orders, foresees the future..?' " [63]

Diversity of opinion can result in a certain degree of anarchy: for instance, during ant colony migrations, "many workers exit the nest carrying eggs, larvae and pupae in their mandibles, while other workers are busy carrying them back again. Still other workers run back and forth carrying nothing at all." In spite of this seemingly anarchistic behavior, tasks are ultimately accomplished by the emergence of a consensus or "global intent," as illustrated in the cooperative nest building of the weaver ant.

> "The weaver ant is an arboreal species whose nests are constructed by living leaves, bound together by larval silk. The construction of the nest requires the cooperation of many workers - some to hold the leaves in place while others spin the silk to bind them. When workers first attempt to fold a leaf they spread over its surface and pull up on the edge wherever they can get a grip. One part is turned more easily than the others, and the initial success draws other workers who add their effort, abandoning the rest of the leaf margin" [64].

It would seem that the initial diversity of opinion in how to tackle a task leads to a better overall solution.

Like scientific communities, the planning processes of insects seem to be largely data driven. For instance, the construction of a nest proceeds without the benefit of a blueprint to show what the finished result should be, even when the process requires several worker-lifetimes. Pierre-Paul Grassé suggests that such accomplishments are not due to memory transfer or some long-term communication; instead, "it is the product of work previously accomplished, rather than direct communication among nestmates, that induces the insects to perform additional labor. Even if the work force is constantly renewed, the nest structure already completed determines, by its location, its height, its shape, and probably also its odor, what further work will be done" [65].

Of course, the big organizational difference between insect societies and scientific communities is in their members; insects cannot reason, learn or communicate as scientists do. Insects can remember the location of important landmarks (such as their nests), and both bees and ants have been trained to walk through relatively complex mazes based on color and light cues [66]. Once learned, the memory of a behavior or location can persist for up to 14 days. Nevertheless, insects have little or no ability to generalize their knowledge or apply it to new situations. A trained ant, when challenged to run the same maze backwards, treats this as an entirely new and different problem [67].

Insect communication is accomplished primarily through transfer of chemical signals, which are inherently of low information content (i.e., different chemicals are secreted to indicate warning, receptivity, hostility, etc.) When these signals are emitted, they are almost always broadcast to the colony as a whole, rather than directed toward one particular individual. Even so, the chemical signals can only be received by those individuals who are in proximity to detect them, and only for a relatively short duration of time (before the signal evaporates) [68].

Behavior

Despite the limitations of their members, societies of insects are able to perform quite complex tasks in the construction, maintenance and defense of their nests, and in the acquisition of their food. The "design" of the *Polistine* wasp's nest, for example, is a remarkable achievement: it is invariably well-situated, structurally stable, defensible and of adequate dimension and functionality. The wasps frequently renovate and expand the nest as the size and the needs of the colony change [69]. In another example, swarms of *Eciton burchelli* (better known as "army ants") have been observed to construct "living bridges" of their own bodies in order to cross over crevices or tree limbs. A number of workers will hook their tarsal claws together to form a chain of bodies over which the rest of the swarm can cross [70]. Such feats suggest that the behavior/intelligence of the colony as a whole is greater than the sum of its parts [62].

Insect societies are extremely fault tolerant. They continue to perform their functions even when large percentages of their members have been killed and reach successful conclusions to tasks even when members are working at cross purposes. For instance, Lindauer points out that "in order to obtain pieces of wax for cells of their own, honeybee workers regularly tear away walls that are in the process of being constructed by other nestmates" [71].

Insect societies are extremely open to changes in their agent sets readily allowing for expansions, contractions and even total replacements. For instance, some nest constructions require many worker-lifetimes to complete and the initial builders cannot communicate instructions to subsequent generations (they are dead before the new brood hatches). Yet each new addition is built in a proper relationship with the previous parts.

4.4 Summary and Remarks

1. Blackboard organizations are characterized by:
- semi-autonomous agents,
- supervised group architectures,
- centralized, dynamic planning, and
- serial task execution.

2. Scientific communities use the same organization as insect societies. This organization is characterized by:
- autonomous agents,
- cyclic-hetrarchic architectures,

- decentralized, dynamic planning that allows for a diversity of opinion but encourages the emergence of a consensus, and
- concurrent task execution.

We will call an organization with these features an *asynchronous team*, or *A-Team*.

3. Both blackboards and A-Teams can handle task uncertainty and are open to new agents, though A-Teams are more open.

4. Organizationally, blackboards lie somewhere between second generation EMSs and A-Teams. (EMSs lie in the near, lower, left part of organization space (Fig 2.1), blackboards are somewhere in the middle, and A-Teams are close to the upper, right edge.) Therefore, it should be easier to adapt blackboards for use in EMSs than A-Teams. Nevertheless, A-Teams deserve serious consideration because of their greater openness and also because any organization that can serve agents as diverse as scientists and insects must have great range and versatility.

5. Social insects demonstrate that agents with very modest reasoning, learning and communication capabilities can, if there are enough of them and they have the right organization, perform fairly complex engineering tasks. Can this lesson be applied to the development of software systems? Only preliminary results are available but they are promising. A-Teams whose agents are numerical algorithms encapsulated in rule-based programs have been built for the solution of nonlinear algebraic equations [72, 30]. The encapsulations are very simple and provide the agents with minimal communication and reasoning capabilities. Nevertheless, the system outperforms conventional approaches [30]. Similarly promising results have been obtained with A-Teams for the design of high rise buildings [5]. The investigation of A-Teams for EMS applications is beginning [73, 74]. The area is security assessment and more specifically, the maintenance of event trees. An event tree is a directed graph whose root node represents the current state of a power system, whose other nodes represent states that could occur in the future and whose arcs represent the events that would cause transitions from one state to another. Since there are very many of these states and events, a complete event tree is extremely large. To be useful, it must be pruned so only small and interesting portions need be calculated. Also, the calculations need to be continually updated as the current state of the power system changes. An experimental A-Team has been developed for both pruning and updating the tree [73]. This team contains two types of agents. The first maintains a small set of important events (N-type contingencies) for study. The second calculates the states that would result from these events. Agents of both types work concurrently and asynchronously. At present they are able to provide a view of possible behaviors that extends three contingencies into the future. Work on adding agents that would suggest corrective actions for undesirable behaviors is underway.

5. CONCLUSIONS

In real-time operations, as in other engineering areas, there is a gap between the complexity of the problems we need to solve and the complexity of the problems we can solve. Perceptions of quality-of-service depend in large measure on the direction and rate of change of this gap.

The factors that are acting to increase the complexity of real-time operations include rising levels of utility interaction, deregulation movements, the proliferation of non-utility generators and growing environmental concerns. Keeping pace with the effects of these factors will require the computer-based multi-agent systems in EMSs to acquire new functionality, particularly, for ill posed problems and abnormal operating states.

The multi-agent system of EMSs are now in their second generation. The first generation was closed in all respects--changes of any sort were difficult to make. The second has opened-up the distributed computing environment, reducing the effort required to upgrade and expand the networks of computers that are used. The power industry is now working to develop standards for communications and interfaces. These standards will make it possible to replace any existing agent with a new one that performs the same function in a better way. However, the addition of agents that perform new functions will remain difficult until organizational changes are made.

Existing organizations have tree-like architectures and use static task planning processes that have remained essentially unchanged through two generations of EMS evolution. These organizations are well suited to algorithmic tasks but not to knowledge-based tasks and especially not to tasks that require the cooperative efforts of algorithmic and knowledge-based agents. To make them better able to handle these latter two types of tasks will require structures that allow for dynamic planning.

There are no systematic methods for designing good organizations from scratch. Instead, the organization-designer must rely heavily on finding and adapting appropriate cases to his or her purposes. The paper lists three such cases. The first is blackboards which have been widely used in artificial intelligence and other engineering areas. Blackboards allow for dynamic planning and are more open to new types of agents than tree-like architectures. They are also fairly close to existing EMS organizations and therefore, should not require a large amount of re-engineering. The other two examples are scientific communities and insect societies. Both appear to use the same sort of organization. We call this organization an A-Team. Among its characteristics are: autonomous agents, a hetrarchic (leaderless) architecture that is exceedingly open to new agents, decentralized dynamic planning, and highly parallel activity that encourages diversity at the beginning of a task and the emergence of a consensus by the end. A-Teams are still in the very early stages of investigation but we feel that any organization that can serve the needs of agents as diverse as scientists and insects deserves serious consideration for the organization of software.

6.REFERENCES

[1] J.L Peterson, "Petri Nets," *Computing Surveys*, vol. 9(3), pp.223-252, Sep. 1977.

[2] D. Harel, "On Visual Formalisms," *Communications of the ACM*, Vol 31, No. 5, pp. 514-520, May 1988.

[3] C.A.R. Hoare, "Communicating sequential processes," *Communications of the ACM*, Vol 21, No. 8, pp.666-667, August 1978.

[4] R.Milner, "A calculus of communicating systems" *Lecture notes in Computer Science*, Vol 92, Springer-Verlag, New York, 1980.

[5] Rich Quadrel, "Asynchronous Design Environments: Architecture & Behavior", Ph.D. Thesis, Department of Architecture, Carnegie Mellon University, September 1991.

[6] S.N.Talukdar and S.J.Fenves, "Towards a Framework for Concurrent Design", in *Knowledge-Based Systems in Engineering Design*, D.Sriram,Ed., 1989.

[7] Keith S. Decker, "Distributed Problem-Solving Techniques: A Survey", *IEEE Transactions on Systems, Man, and Cybernetics*, September/October 1987.

[8] R.G.Smith, "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver", *IEEE Transactions on Computers*, vol. c-29, no. 12, pp. 1104-1113, December 1980.

[9] S.B.Shimanoff, "Coordinating group interaction via communication rules", in *Small Group Communication*, L.A.S.R.S.Cathcart, Ed., Dubuque, IA: Brown 1984.

[10] E.Chang, "Participant systems", in *Proc. 1985 Distributed Artificial Intelligence Workshop*, Dec. 1985.

[11] R.Davis and R.G.Smith, "Negotiation as a metaphor for distributed problem solving", *Artificial Intelligence*, Vol. 20 pp. 63-109, 1983.

[12] M.R.Garey and D.S.Johnson, *Computers and Intractability - A guide to the theory of NP-completeness*, W.H.Freeman and Company, 1978.

[13] J. Galbraith, *Designing Complex Organizations*, Addison-Wesley Publishing Co., 1975

[14] M.R.Genesereth, M.L.Ginsberg, and J.S.Rosenschein, "Cooperation without communication", in *Proc. 5th Nat. Conf. Artificial Intelligence*, Philadelphia, PA, Aug. 1986, pp. 51-57.

[15] J.M.Shafritz and J.S.Ott, *Classics of Organization Theory*, Second Edition, Dorsey Press, 1987.

[16] R.G.Smith and R.Davis, "Frameworks for cooperation in distributed problem solving", *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-11, pp. 61-70, Jan. 1981.

[17] C.Hewitt, "The challenge of open systems", *BYTE*, April 1985, pp. 223-242.

[18] C.Hewitt, "Offices are Open Systems", *Readings in Distributed Artificial Intelligence*, A.H.Bond and L.Gasser (eds)., Morgan Kaufmann, 1988.

[19] M.T.Rose, *The Open Book - A practical perspective on OSI*, Prentice Hall, 1990.

[20] M.Kunugi, M.Yohda, S.Tamura, H.Watanabe and T.Hasegawa, "Performance Validation of a Functionally Distributed Energy Management Architecture", *Proceedings of the 1991 Power Industry Computer Application Conference*, Baltimore, MD May 7-10 1991, pp.193-199.

[21] S.J.Mullender, G.V.Rossum, et al, "Amoeba: A Distributed Operating System for the 1990s", *Computer*, May 1990, pp. 44-53.

[22] M.Satyanarayanan et al., "The ITC Distributed File System: Principles and Design", *Proceedings of the 10th ACM Symposium on Operating System Principles*, pp. 35-50, December 1985.

[23] S.Ahuja, N.Carriero, and D.Gelernter, "Linda and Friends", *Computer*, Vol. 19, No.8, Aug. 1986, pp. 26-34.

[24] E.Cardozo, "DPSK: A Kernel for Distributed Problem Solving", Ph.D. Thesis, ECE department, Carnegie-Mellon University, January 1987.

[25] H.Daniels and N.Mayur, "More than mainframes", *IEEE Spectrum*, vol. 22, pp. 54-61. Aug. 1985.

[26] J.S.Horton and D.P.Gross, "Computer Configurations", *Proceedings of the IEEE*, vol. 75, no. 12, December 1987.

[27] D.J.Gaushell and H.T.Darlington, "Supervisory Control and Data Acquisition", *Proceedings of the IEEE*, vol. 75, no. 12, December 1987.

[28] "Building a framework for integrated communications", *EPRI Journal*, July/August 1988.

[29] G.Paula, "Scada/EMS: New applications, lower costs, open system", *Electrical World*, July 1991.

[30] P.S.de Souza and S.N.Talukdar, "Genetic Algorithms in Asynchronous Teams", *Proceedings of the Fourth International Conference on Genetic Algorithms*", Morgan Kaufmann, Los Altos, CA, July 1991.

[31] Elaine Rich, *Artificial Intelligence*, McGraw-Hill, 1983.

[32] C.C.Liu, "Knowledge-Based systems in Power Systems: Applications and Development Methods", *Trans. IEE of Japan*, Vol. 110-B, No.4, Apr. 1990.

[33] Proceedings of the first symposium on Expert Systems Applications to Power Systems, Sweden, 1988. (ESAPS 88)

[34] Proceedings of the second symposium on Expert Systems Applications to Power Systems, Seattle, 1990. (ESAPS 90)

[35] Proceedings of the third symposium on Expert Systems Applications to Power Systems, Japan, 1991.(ESAPS 91)

[36] B.Wollenberg, "Feasibility study for an energy management system intelligent alarm processor", *IEEE Transactions on Power Systems*, May 1986, pp. 241-247.

[37] R.Bijoch, S.Harris and T.Volkmann, "Intelligent alarm processor at Northern States Power", *Proc. 1988 ESAPS*, pp. 79-83.

[38] A.Shoop, S.Silverman and B.Ramesh, "Consolidated Edison System Operation Computer Control Center Alarm Advisor", *Proc. 1988 ESAPS*, pp. 84-88.

[39] S.N.Talukdar and E.Cardozo, "Artificial Intelligence Technologies for Power System Operations", Report EL-4323, Electric Power Research Institute, Jan. 1986.

[40] E.Cardozo and S.N.Talukdar, "A Distributed Expert System for Fault Diagnosis", *IEEE Trans. on Power Systems*, May 1988, pp. 641-646.

[41] K.Komai, T.Sakaguchi and S.Takeda, "Power System Fault Diagnosis with an Expert System enhanced by the general problem solving method", *Proc. IASTED High Technology in the Power Industry*, August 20-22, 1986.

[42] C.Fukui and J.Kawakami, "An Expert System for Fault Section Estimation using Information from Protective Relays and Circuit Breakers", *IEEE Trans. on Power Delivery*, October 1986, pp. 83-87.

[43] K.Tomsovic, C.C.Liu et al., "An Expert System as a Dispatchers' aid for the isolation of line section faults", *IEEE Trans. on Power Delivery*, July 1987, pp. 736-743.

[44] R.D.Christie, S.N.Talukdar and J.C.Nixon, "CQR: A Hybrid Expert System", *Proceedings of Power Industry Computer Applications conference*, May 1990, pp. 267-273.

[45] D.J.Sobajic and Y.H.Pao, "An artificial intelligence system for power system contingency screening", *IEEE Trans. on Power Systems*, May 1988, pp. 647-653.

[46] T.Sannes and O.B.Fosso, "Security assessment and emergency control using an object-oriented interface", *Proc. 1988 ESAPS*, pp. 8.1-8.8.

[47] R.Rios-Zalapa and B.J.Cory, "An expert system for security assessment in operational planning of power systems", *Proc. 1988 ESAPS*, pp. 17.28-17.35.

[48] T.Sakaguchi and K.Matsumoto, "Development of a knowledge-based system for power system restoration", *IEEE Trans. on Power Apparatus and Systems*, Feb. 1983, pp. 320-329.

[49] Y.Kojima, S.Warashina et al., "The development of power system restoration method for a bulk power system by applying knowledge engineering techniques", *IEEE Trans. on Power Systems*, Aug. 1989, pp. 1228-1235.

[50] C.C.Liu, S.J.Lee and S.S.Venkata, "An expert system operational aid for restoration and loss reduction of distribution systems", *IEEE Trans. on Power Systems*, May 1988, pp. 619-626.

[51] N.Kakimoto, M.Emoto and M.Hayashi, "An application of artificial intelligence to restoration of bulk power systems", *Proc. 1988 ESAPS*, pp. 8.17-8.23.

[52] J.G.March and H.A.Simon, *Organizations*, Wiley, New York, 1958.

[53] T.W.Malone and S.A.Smith, "Trade-offs in designing organizations: Implications for new forms of human organizations and computer systems", Tech Report, CISR-WP-112, MIT Center for Information Systems Research, Cambridge, MA, Mar. 1984.

[54] H.A.Simon, "The Design of Large Computing Systems as an Organizational Problem", *Organisatiewetenschap en Prakitijk*, 1976.

[55] M.S.Fox, "An organizational view of distributed systems", *IEEE Trans. Sys., Man, Cybern.*, vol. SMC-11, pp. 70-80, Jan. 1981.

[56] L.D.Erman, F.Hayes-Roth, V.R.Lesser, and D.R.Reddy, "The Hearsay-II speech-understanding system: Integrating knowledge to resolve uncertainty", *Computing Surveys*, ACM, vol. 12, pp. 213-253, June 1980.

[57] H.Penny Nii, "Blackboard Systems: The blackboard model of problem solving and the evolution of blackboard architectures", *The AI Magazine*, Summer 1986.

[58] H.Penny Nii, "Blackboard Systems: blackboard application systems, blackboard systems from a knowledge engineering perspective", *The AI Magazine*, August 1986.

[59] M.D.Rychener, *Expert Systems for Engineering Design*, Academic Press, 1988.

[60] W.A.Kornfeld and C.E.Hewitt, "The scientific community metaphor", *IEEE Trans. Sys., Man, Cybern.*, vol. SMC-11, pp. 24-33, Jan. 1981.

[61] C. Emery, "Le polymorphisme des fourmis et la castration alimentaire," *Compte Rendu des Séances du Troisième Congrès International de Zoologie*, Leyde, pp. 395-410, 1895.

[62] E.O.Wilson, *The Insect Societies, Belnap Press of Harvard Univ.*: Cambridge, MA, 1971.

[63] M. Maeterlinck, *The life of the ant*, (English tr. by B. Miall), George Allen & Unwin, Ltd.:London, 1930.

[64] J.H. Sudd, "How insects work in groups," *Discovery, London*, pp. 15-19, Jun. 1963.

[65] P.P. Grassé, "Nouvelle expériences sur le termite de Müller *(macrotermes mülleri)* et considérations sur la théorie de la stigmergie," *Insectes Sociaux*, vol. 14(1), pp. 73-102, 1967.

[66] H. Kalmus, "Vorversuche über die Orientierung der Biene im Stock," *Zeitschrift für Vergleichende Physiologie*, vol. 24(2), pp. 166-187, 1937.

[67] B.A. Weiss and T.C. Schneirla, "Inter-situational transfer in the ant *Formica schaufussi* as tested in a two-phase single choice-point maze," *Behaviour*, vol. 28(3-4), pp. 269-279, 1967.

[68] E.O. Wilson, "Chemical communication among workers of the fire ant *Solenopsis saevissima*," *Animal Behaviour*, vol. 10(1-2), pp. 134-164, 1962.

[69] M.J.W. Eberhard, "The social biology of polistine wasps," *Miscellaneous Publications, Museum of Zoology, Univ. of Michigan*, vol. 140, pp. 1-101, 1969.

[70] C.W. Rettenmeyer, "Behavioral studies of army ants," *Kansas Univ. Sci. Bulletin*, vol. 44(9), pp. 281-465, 1963.

[71] M. Lindauer, "Ein Beitrag zur Frage der Arbeitsteilung im Bienenstaat," *Zeitschrift für Vergleichende Physiologie*, vol. 34(4), pp. 299-345, 1952.

[72] S.N.Talukdar and P.S.de Souza, "Asynchronous Teams", presented at the Second SIAM Conference on Linear Alegbra: Signals, Systems and Control, San Francisco, CA, Nov. 1990.

[73] S.N.Talukdar, V.C.Ramesh, J.C.Nixon, "A Distributed System of Control Specialists in Real-Time Operations", *Proceedings of the third symposium on Expert Systems Applications to Power Systems*, Japan, April 1991.

[74] V.C.Ramesh, Rich Quadrel, P.S.de Souza, and S.N.Talukdar, "Asynchronous Teams: An Organizational Model for Distributed Problem Solving", presented at the 1991 AIChE Summer National Meeting, Pittsburgh, August 1991.