Using Self-Diagnosis to Adapt Organizational Structures *

Bryan Horling, Brett Benyo, and Victor Lesser
University of Massachusetts
Department of Computer Science
Amherst, MA 01003
{bhorling, bbenyo, lesser}@cs.umass.edu

ABSTRACT

The specific organization used by a multi-agent system is crucial for its effectiveness and efficiency. In dynamic environments, or when the objectives of the system shift, the organization must therefore be able to change as well. In this paper we propose using a general diagnosis engine to drive this process of adaptation, using the TÆMS modeling language as the primary representation of organizational information. Results from experiments employing such a system in the Producer-Consumer-Transporter domain are also presented.

Keywords: Organization and social structure, organization self-design.

1. OVERVIEW

As the sizes of multi-agent systems grow in the number of their participants, the *organization* of those agents will be increasingly important. In such an environment, an organization is used to limit the range of control decisions agents must make, which is a necessary component of scalable systems. Are agent agents arranged in clusters, a hierarchy, a graph, or some other type of organization? Are the agents' activities or behaviors driven solely by local concerns, or do external peers or managers have direct influence as well? Is communication between agents active, via messaging of some sort, or passive, using observations or engineered assumptions? These and other characteristics define the *organizational structure* of a multi-agent system - the rules which define the roles agents play and the manners in which they interact with other agents in the system.

Clearly the characteristics described above will have an impact on the efficiency and responsiveness of both large and small multi-agent systems. It should also be intuitively clear that the effectiveness of the organization is dependent on the agents, environment, and goals involved in the system. The problem then, is how to derive such a structure given a particular situation. The simplest option is to statically define the organization when the system is developed. This has the benefit of being a simple and direct solution, but can become impractical when the sets of agents and goals are large and diverse. Static solutions also suffer when elements of the multi-agent system are dynamic, since characteristics of the environment, organizational goals, or member agents may change such that the initial organization becomes inefficient. Members of the agent pool may become deactivated or compromised in some way, making it impossible for the system to function correctly, or other agents may not be used effectively when they are added. In this sense, the organization is a set of assumptions that the system works by. As these assumptions become invalid, the organization must be able to adapt to keep the system viable.

The term Organizational Self-Design (OSD) has been used previously [2] to describe the general technique of employing the members of a multi-agent system to generate or adapt their own organizational structures at runtime. Earlier work in this area tended to focus on adapting specific qualities of the organization, such as task allocation [9] or load balancing [6, 8]. Organizational structure generation has also been proposed as arising from local [6], global [2], and hybrid [11] perspectives. Each of these systems demonstrated specific techniques that worked well and efficiently in their respective environments, but they were not general solutions to the problem. In this paper we propose a more general approach, using diagnosis, to detect deficiencies in the organizational model and assist in the creation of solutions to those deficiencies; the eventual goal being to create a reusable organizational adaptation engine. We will show how a general diagnosis engine, coupled with a powerful representation of that organization, can be used to effect change in a wide range of characteristics from arbitrary perspectives.

To help make this notion of organizational adaptation more concrete, we will look at an example from the Producer, Consumer, Transporter (PCT) domain [4]. In this domain, there are conceptually three types of agents: producers, which generate resources; consumers, which use them; and transporters, which move resources from one place to another. In general, a producer and consumer may actually be different faces of a factory, which consumes some

* The effort represented in this paper has been sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement numbers F30602-99-2-0525 and F30602-97-1-0249, and by the Department of the Navy, Office of the Chief of Naval Research, under Grant No. N00014-97-1-0591. This material is also based upon work supported by the National Science Foundation under Grant No. IIS-9812755. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), Air Force Research Laboratory, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

AGENTS'01, May 28-June 1, 2001, Montréal, Quebec, Canada.

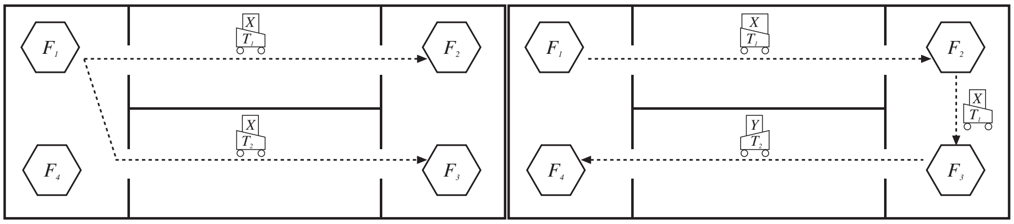

Figure 1: The initial (L) and revised (R) transporter organizations.

quantity of resources in order to produce others. There are several characteristics of this domain where alternatives exist for the factories and transporters - the choices made at these points by or for the PCT agents make up the organizational structure of the system. Examples of such characteristics include the types and quantities of resources a producer should generate, the set of potential sources a consumer should obtain required resources from, and what paths a transporter may choose to follow as it moves about.

In this example, consumers F_2 and F_3, shown in Figure 1, require some amount of resource X. In the initial organization, each is being supplied with X by producer F_1. X is then supplied to F_2 and F_3 by transporters T_1 and T_2, respectively, each of which operates at 50% capacity. Factory F_4 is initially idle, but at some future point in time it obtains production request, which requires resource Y to be satisfied. Fortunately, F_3 produces Y, but in the initial organization, no additional transporters are available to deliver the needed goods. With a diagnostic system in place, the transporters could determine that their initial organization, while functional under the initial conditions, included under-loaded transporters and was therefore potentially suboptimal. Instead of using two transporters running at 50% capacity, just one at 100% capacity could satisfy the original requirements for X expressed by F_2 and F_3, albeit at a slight time penalty because of the extra stop. Thus, if instantiated, the revised organization would leave T_2 free to perform the transportation required by F_4. More quantitative results from this domain will also be covered in section 4. Related work, using diagnosis to learn coordination rules in an intelligent home scenario, can also be seen in [5].

Figure 2 shows at a high level how we propose organizational design can be situated and integrated in an agent. In this architecture, critical components within the agent, such as those responsible for problem solving, negotiation and scheduling, obtain the vast majority of their information from an organizational design layer. This layer abstracts and filters elements of the operating environment in a manner consistent with the agent's role in the organization. The abstraction is composed of one or more information sources, such as TÆMS structures or MQ values [12], capable of encoding the various aspects of the organization. TÆMS a task and interaction modeling language, will be discussed in detail in a later section. MQ (motivational quantity) values, which give the agent a more powerful way to reason about the utilities of its tasks, will not be covered in this paper. To permit adaptation, the organizational design layer is maintained by a diagnostic subsystem, which attempts to repair faults and inefficiencies by adjusting elements of the organizational structure. This diagnostics process can itself be driven by a number of sources, including observations of the environment, conditions monitored within the agent, and discourse with other agents. The direct effects of these diagnoses typically take place within a relatively small group of agents, so one can think of this technique as being a search process for the correct organization through local adaptation. The organization will go through a set of distribution adaptations, each involving a series of local adaptations by individual agents.

Going back to the previous PCT example, we can see how this technique would work in practice. The initial organizational structure would be encoded in TÆMS structures in both the transporters and factories. They would indicate such characteristics as what goals the factories and transporters should work towards and how they could be accomplished. Initially, the organization would be unconstrained, permitting the type of interactions seen in Figure 1L. Diagnosis running on the transporters or factories would determine that while the transporter loads were well balanced in the initial state, the arrangement was not necessarily the most efficient use of their abilities. F_2 could use this information to add a constraint to it's local organizational representation, indicating that it should use T_1 for its transportation needs. Later, when F_4 requests the use of a transporter, T_2 will then be available.

In the next section we will give more details about our view of the actual knowledge used by an agent to represent the organizational information that makes up the abstraction layer shown in Figure 2. Following this, we will cover our diagnostic system, shown in the middle of this same figure, and how it is integrated into and used by our agents. In section 4 results from an experiment in the PCT domain will be covered, and in section 5 we will present our conclusions.

2. ORGANIZATIONAL KNOWLEDGE

As mentioned in the previous section, the range of information that comprises an organizational structure can be quite broad. It is our opinion that there is no single, comprehensive set of characteristics that might make up *the* definition of an organizational structure. Instead, the set is dependent more on what alternatives are possible within a particular multi-agent system and which of those alternatives can have an impact on the system's behavior and effectiveness. Given that, we will present in this section our organizational representation, called TÆMS (Task Analysis, Environmental Modeling, and Simulation), which is flexible enough to model a wide range of organizational characteristics.

2.1 TÆMS

The primary representation of the organizational structure is done with the domain-independent TÆMS task modeling language [3] (see Figure 3 for a simple example). A TÆMS task structure is essentially a goal decomposition tree, where leaf nodes represent executable primitive methods and internal task nodes provide a hierarchical organization. Root level tasks (those with no supertasks) are known as task groups, and conceptually represent high level goals that might be achieved. Associated with each task is a quality-accumulation function (QAF), which indicates how the quality of the task is calculated from that of its subtasks. Associated with each method is a distribution-based description of its expected quality, cost and duration measures. Together, the probabilistic method de-

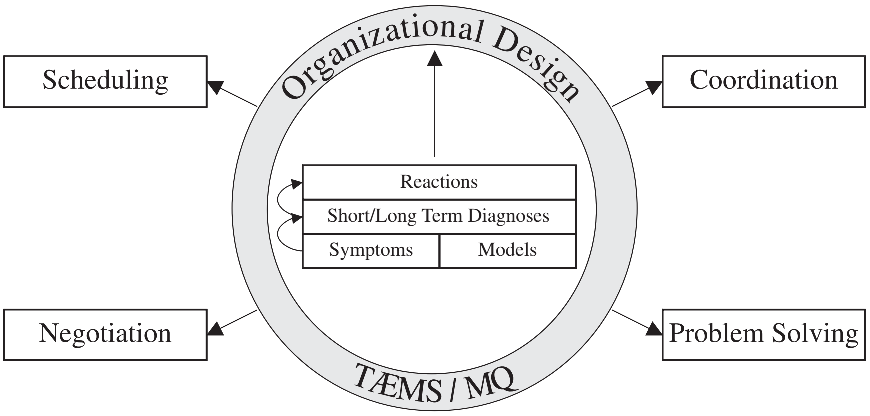

Figure 2: Role of organizational knowledge within an agent.

scriptions and QAFs allow a scheduler to effectively reason about the traits and tradeoffs of a wide range of possible schedules. A third type of element, interrelationships, which arise between internal tasks, methods and resources can be used to indicate a wide range of interactions, such as enables, facilitates, hinders and produces (e.g. performing a task will *enable* the execution of another, or a task will *produce* some amount of resource, as seen in Figure 3). Interrelationships may also span task structures between agents, and tasks and methods performed by remote agents may be represented locally. Combined, the capabilities give developers using TÆMS the flexibility to model a wide range of traits, from low-level performance characteristics of a single action to a high-level representation of the system's control hierarchy.

TÆMS task structures are typically used to encode the different mechanisms for achieving a goal, and the constraints and tradeoffs associated with each potential plan. They are also used to describe both the potential capabilities of an agent and the subset of those capabilities it should employ given its place in the organization. To do this, each agent will have two different versions, called *views*, of its local task structure: *subjective* and *conditioned*. The subjective view contains what the agent believes to be the complete model of its local execution alternatives[1]. The conditioned view is a copy of the subjective which has gone through a process of *conditioning* - it may contain task, method or interrelationship deletions, modifications or insertions. The conditioned view is normally used for plan construction, so these modifications indirectly allow the problem solver performing the conditioning process to focus the attention of the scheduling and coordination mechanisms. As we will see below, the conditioned view can also represent the instantiation of the role assigned to it by the organizational structure.

2.2 Task and Goal Representation

Since the general purpose of TÆMS is to facilitate plan generation, it is well suited for representing the different task alternatives available to an agent in an organization. In an agent's subjective view we can represent (or dynamically generate) structures describing each of the high level goals the agent can achieve. Each of these structures would in turn describe the various alternate ways that a particular goal might be achieved. The subjective view would then be, in this light, a complete description of all the possible roles an agent might be assigned to, and how the agent might act to satisfy that role.

Within a particular organizational structure, however, an agent will typically (but not necessarily) be working toward just a single or limited set of goals. Thus, in the conditioned view there will be a single task group representing that goal. The subtree underneath that task group might be further pruned to reflect other decisions within the organization. For example, in the subjective view there might be two possible ways to complete a task, one local solution and another using a remote contract, whereas an organizational constraint could remove the remote option from the conditioned view. So, using this representation we can encode all the tasks a particular agent might be working on, and also the specific task(s) they have chosen or been assigned. These techniques are used in the experiment shown in section 4 to control the path selection done by transporter agents.

2.3 Specifying Interactions

As mentioned above, TÆMS allows the agent to represent tasks and methods that other agents may perform. This capability allows TÆMS to model potential interactions between agents very effectively. Consider the case where agent C_1 requires resource X as part of its manufacturing process, as seen in Figure 3. Here, C_1 has a method Get-Materials, which consumes some amount of resource X. In the subjective view we can see that C_1 knows of three other agents that can produce X for it: P_1, P_2, and P_3, each of which is represented by a shaded, nonlocal method that has a produces interrelationship to X. In the conditioned view only P_2 is represented, which indicates that in this organization, C_1 should obtain X from P_2. A less restrictive organization might allow C_1 to choose probabilistically from either P_1 or P_2, which could be represented by adding P_1's produces interrelationship to the conditioned view. In this new model, the local scheduler would select from the two each time the resource is needed, based on the characteristics that differentiate the two produces interrelationships. This type of probabilistic usage relationship will be discussed further in the example in section 4.

Other interrelationship types might inform the agent that another agent's actions could enable, disable, facilitate or hinder local execution. Assuming the agent needs to interact, explicitly or not,

[1]There is also an omniscient *objective* view, inaccessible to agents, which defines the real execution alternatives. In simulation, one can engineer differences between the objective and subjective views to create scenarios where the agent's expectations are not met.

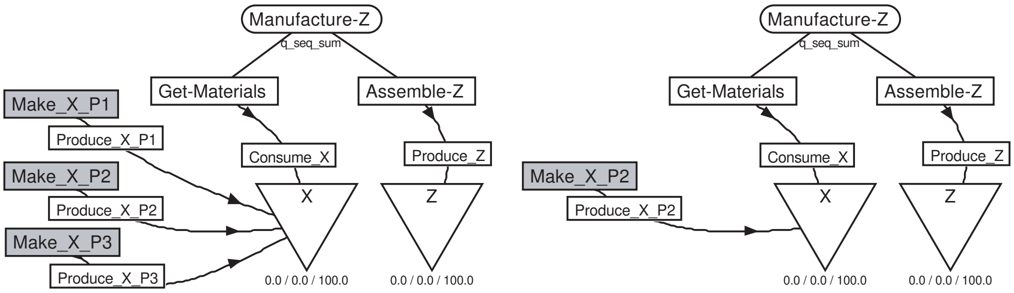

Figure 3: Subjective (left) and conditioned (right) views of C_1's task structure.

with those remote agents to exploit these interrelationships, they then indicate a point of potential coordination. For instance, if a remote agent's method enables one at the local agent (i.e. there is an enables interrelationship between them), the local agent must ensure that the remote method has successfully completed before it can succeed at its local method. This implies that some sort of coordination must take place to cause the correct ordering of events. Thus, an agent using this type of model can succinctly encode what sort of coordination is needed (based on the interrelationship type), with what other agents it should take place, and given a schedule of execution, when it should occur.

2.4 Other Organizational Details

Data concerning particular agents, existing commitments, and execution schedules are also stored within TÆMS models. Inevitably, however, there are some details particular to a given organization that do not directly fit into this representation. For these situations, all elements in a TÆMS model can be associated with an arbitrary set of attributes, where one could specify such things as preferred communication medium, optimal load measurements, or interaction history with a particular agent. Also stored here are performance characteristics, such as result thresholds and tolerances and expected frequency statistics, which the diagnosis subsystem can use to help identify potential failures.

With this information, we can now return to the questions posed in the overview section. The arrangement of agents can be expressed and derived locally by using the complete structure and owning agent tags of tasks and interrelationships. Commitments can exhibit potential influences on agent activities, or by explicitly modeling the task of obtaining goals from remote agents. Interrelationships can denote communication alternatives among agents, and their presence in the conditioned view determines if they should be explicit or implied. Our subjective view represents all the possible roles and responsibilities the agent may hold in the organization, while the conditioned view indicates its currently assigned position. The organizational search space is therefore specified by the range of possible conditioned structures. To adapt its role in the organization, the agent must develop an appropriate mapping from the subjective to conditioned. The TÆMS knowledge representation thus serves as a reasonable representation of the organizational structure; the task now is to use diagnosis to find the appropriate mapping.

3. THE DIAGNOSTIC SUBSYSTEM

Figure 4 shows the architecture of the diagnostics subsystem we currently employ. It uses a blackboard-based design, separating the process into three distinct layers: symptoms, diagnoses, and reactions. This type of system offers several advantages. It promotes a clear chain of reasoning, since the diagnoses supporting a given reaction can easily be identified, as can the symptoms that support a particular diagnosis. Each layer is also subdivided by time, so a history of activity on each level is readily accessible. The blackboard layers also clearly define the separation of responsibilities. This modularity allows any of the layers to be accessed at any time, enabling arbitrary components or even remote agents to asynchronously use and add to elements on the blackboard. The different layers of the blackboard, and the components which make use of them (excepting the effect monitor), will be discussed below. In our current systems, each agent uses this subsystem to perform local diagnosis, although it is quite feasible that in other systems a specialized "diagnosis" agents would be responsible for monitoring small groups of their peers.

The lowest level of the blackboard contains symptoms, elements that contain observations about such things as the environment, agent activities and commitments. Two classes of components currently generate symptoms: *observers* and *modelers*. Observers work by simply monitoring different aspects of the agent, and generating symptoms when appropriate. Modelers take a more proactive approach by building or learning models, and then using these models as a basis for comparison, an approach similar to that used in conventional model-based diagnosis. As models are updated, or predictions derived from the models fail, appropriate symptoms describing these instances are noted on the blackboard. We have experimented with modelers that learn interrelationships in TÆMS objects [7] and others that predict environmental resource usage.

Diagnosis is a well-researched field, with many different methods and techniques already available to the system designer. Our goal was to use a technique that offered great flexibility in the information it could use and the diagnoses it could generate, without sacrificing subject scope or domain independence. It is not clear from the outset, however, that any single diagnostic technique (e.g., model-based, symptom-directed, collaborative) is suitable for the entire range of faults exhibited by multi-agent systems. It was therefore desirable to use a system or framework capable of incorporating different diagnostic techniques. In such an architecture we can make use of a variety of different methods, given the types of failures they best address, and the performance characteristics they exhibit (e.g. convergence time, scalability, efficiency, etc.).

Expanding on work first researched in [10], we chose to organize our diagnostic process using a causal model. The causal model is a directed, acyclic graph that organizes a set of diagnosis nodes. Figure 5 shows an example of such a graph; more examples of graphs addressing broader topics can be found in [1]. A more applied model used in the PCT domain can also be seen in Figure 6. Each node in the causal model corresponds to a particular diagnosis, with varying levels of precision and complexity. As a node produces a diagnosis, the causal model can determine what other, more detailed, diagnoses may further categorize the problem. Within the

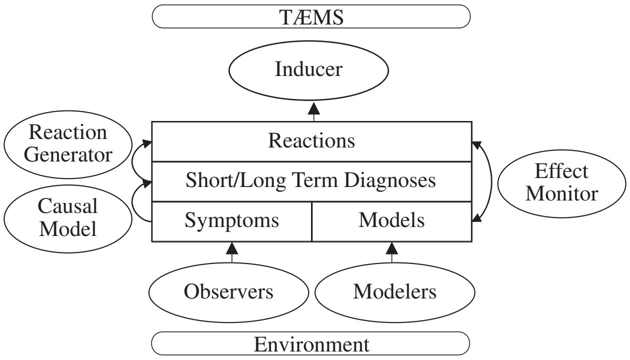

Figure 4: High-level architecture of the diagnostics subsystem.

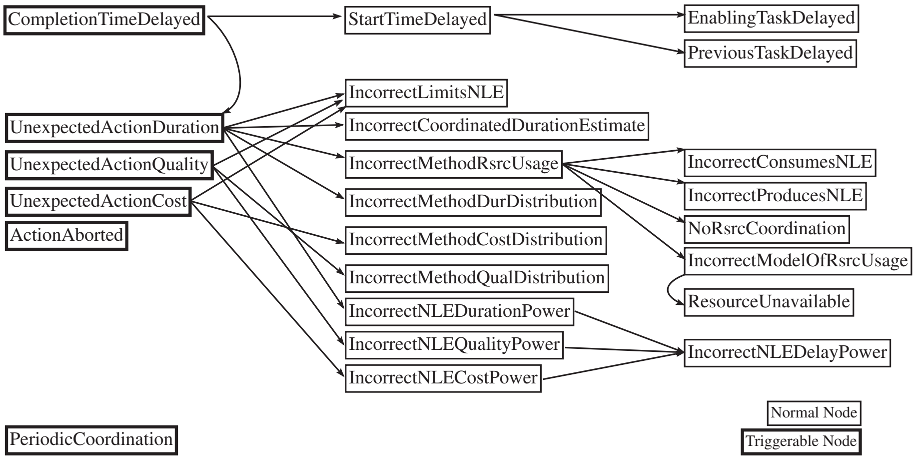

Figure 5: Causal model for diagnosing action- and coordination-based faults.

diagnosis system, the causal model then acts as a sort of road map, allowing diagnosis to progress from easily detectable symptoms to more precise diagnostic hypotheses as needed. A more advanced technique can also use the same structure to help validate diagnoses, by using backward chaining through the branches to determine the state of other potentially related diagnostic nodes.

It is worth mentioning that nodes in the causal model do not necessarily produce single-shot diagnoses. Some nodes, such as UnexpectedActionDuration, simply produce a diagnosis and stop. Others, such as PeriodicCoordination, can produce a diagnosis and monitor it over time to determine if conditions change or more evidence is found. Thus, a node could pose an initial diagnostic hypothesis when confronted with a particular situation. Since it only has limited evidence (presumably one data point), the confidence on that diagnosis would be low. The node can persist, however, and either passively watch for related evidence, or actively gather new information that either contradicts or corroborates the initial diagnosis. Furthermore, since other diagnoses or reactions may be based on that initial diagnosis, a change may also affect their confidence, causing a ripple effect throughout the blackboard as the original diagnosis accumulates new information.

The reactions level contains descriptions of the potential solutions to diagnoses found on the previous level. In some sense, then, these reactions are the effectors of organizational change. As diagnoses are hypothesized, and their confidence reaches a certain threshold, the reaction generator will pose solutions to those diagnoses. For instance, if the causal model determines that insufficient resources were available for a particular action because their usage was not coordinated over, a potential reaction would modify the conditioned view of the agent's TÆMS model so that coordination would take place for that action in the future. A different reaction for that problem might remove the offending method from the view altogether. Similarly, if a diagnosis determined that an agent's actions were predictably periodic, a reaction could set up default commitments to reduce the need for explicit communication during each of those cycles. Like diagnoses, reactions can also be long-lived, providing incremental change in response to updated diagnoses or to slowly test new organizational changes.

Organizational changes for higher level characteristics work the same way. For instance, in Figure 3, a consumer's choice of producers limited is by the organization. A reaction could implement this change by removing the methods and interrelationships that describe those extra producers from the conditioned view. In the initial PCT organizations seen in Figure 1, a reaction would modify the conditioned view of F_3 to indicate it should use T_1. When this change is made, T_2 would be free to accept the transportation request from F_4.

Similar methods can drive more large scale reorganizations, although additional safeguards should be present to protect against the likely larger cost of failure. In these cases, local reactions can directly implement sophisticated reorganization techniques like those seen in [6, 8, 11, 9], or they can direct the local agent controller or problem solver to do so. For instance, local diagnosis could first determine that the control hierarchy for the current organization is inefficient or overwhelmed, because one or more high-level managing units was unable to cope with its workload. Because resolving such an issue can result in an interruption of service, this diagnosis would first cause a more detailed view of the situation to be analyzed, by evaluating different metrics, analyzing trends, or gathering additional evidence from remote sources. If this more advanced diagnosis also determines a problem exists, a reaction can be generated which prompts a more sophisticated direct, distributed search for a more appropriate organizational structure.

The task of selecting from among several potential reactions lies with the inducer. Our current inducer simply instantiates any reaction it sees on the blackboard. In future versions this component would be more complex, able to differentiate between reactions, analyze the potential benefits and drawbacks of each, and determine the best reaction given the agent's current context and prior history.

4. EXPERIMENTAL RESULTS

A specific system using the architecture outlined in the previous sections has been implemented and tested using scenarios from the PCT domain. In this section, we will outline one of those experiments, examining the effects of organizational changes in a small, eight member multi-agent system.

In this scenario, there are four factories and four transporters operating in the environment shown in Figure 1. As shown in that figure, there are also four "doorways", or potential points of contention along the lengthwise transporter routes. These doorways only allow one transporter through at a time, which transporters must be aware of as they select their routes. The objective for transporters is then to deliver their cargo on time, given the potential vagaries of factory production and the need to avoid collisions on travel pathways. Factories in the environment have different production capabilities and resource requirements, summarized in Table 1, and they must also select one or more transporters to deliver materials to them. Each factory is capable of producing both a simple resource, one that requires no external elements to build, and a complex resource, which requires other resources to produce. F_4 can also produce an even more complex resource Q, which is the combination of four other resources.

Factory	Simple	Complex
F_1	$\emptyset \to A$	$B + C \to X$
F_2	$\emptyset \to C$	$B + D \to Y$
F_3	$\emptyset \to D$	$A + C \to W$
F_4	$\emptyset \to B$	$A + D \to Z$ $A + C + X + Y \to Q$

Table 1: Production rules for factories in PCT example.

In the initial phase of the scenario, the goal of each factory is to produce seven of each type of complex resource by time 700. After time 700, the objective shifts so that the system as a whole should produce as much Q as possible by time 1200. To provide further context, the round trip duration from F_1 to F_3 is around forty time units (barring path contention), and resource production can take five time units. Two organizational characteristics have alternatives as part of this scenario - the transporter selected for a particular transportation task, which is decided by the consuming factory, and the path the transporter selects to perform that task. Each consumer then chooses one of the four available transporters to satisfy its delivery needs. Each transporter has a choice of two different viable paths for any given delivery task. This latter selection is implemented probabilistically, so a given transporter might have a 70% chance of selecting path A, and a 30% of path B. Three runs were performed, the first employed an arbitrary static organization, the second used diagnosis with only task allocation nodes from the causal model shown in Figure 6, and the third used the entire causal model, which added path selection diagnosis to the second trial. The objectives behind most of the nodes in the model should be intuitive: DeadlineMissed fires when action's deadline has not been achieved, TransporterOverloaded is true when a transporter's task load is disproportionate to those of it's peers, and OverCoordination determines when excessive coordination activity has been detected, as would occur when a particular route is highly contended. Organizational change occurs under two circumstances. A TransporterOverloaded diagnosis will result in the transporter attempting to shift some of its current delivery tasks to alternate transporters. This will also induce change in the consumer's organizational model such that future requests will go to the alternate transporter. An OverCoordination diagnosis associated with the route coordination protocol (which prevents transporters from colliding along a common path) will cause the transporter to adjust its local route selection probabilities.

Table 2 shows the results from the experiments; average delay is the average amount of time from when a factory begins gathering materials for resource production to when the resource is completed. As shown in the table, the results from the static organization are quite poor, except those for resource X, which benefitted from using relatively underloaded transporters in the organization. The delays during production of Q are particularly bad, being more than three times those in other runs. Clearly this performance is dependent on the organization that was initially chosen, but more important to this discussion is the fact that an initial poor organization was greatly improved with the addition of diagnostic-based adaptation, as shown by the results from the second and third runs.

	Average Delay				
Adaptation	W	X	Y	Z	Q
None	261.9	94.5	253.6	252.0	422.0
CM (Tasks)	97.8	88.4	90.3	98.4	118.9
CM (Full)	93.0	90.0	92.5	96.5	99.6

Table 2: Results from three trials in the PCT scenario.

In the results from the second run we see that the average production delay for each resource was reduced by nearly two-thirds in most cases. These gains were obtained by using load statistics, which can be generated from the transporter's conditional TÆMS view, to more efficiently allocate transporters to the various tasks available. Reallocation was performed by adding or removing interrelationships from a factory's conditional view, which constrained the set of transporters the factory could potentially use. Initially, the consumers chose from all transporters available in the environment, which was quite inefficient because transporters working on long-haul (diagonal) runs were selected as often as those on shorter runs. Through incremental changes to their conditional views, reacting to transporter performance and load, consumers in the second trial settled into an organization where more lightly-loaded transporters were selected more frequently, producing a more efficient allocation. The allocation for the initial phase settled around time 240, after four task reassignments. When the second phase started, after time 700, additional task reallocations took place every 100 pulses or so until the system completed.

With the introduction of path selection diagnosis in the third run the delays dropped again, especially that of resource Q, which due to its larger component set has the most potential for conflicting transporter routes. Diagnosis relevant to path selection was performed by the TooManyConflicts and OverCoordination nodes, which determined if a transporter encountered excessive conflicts

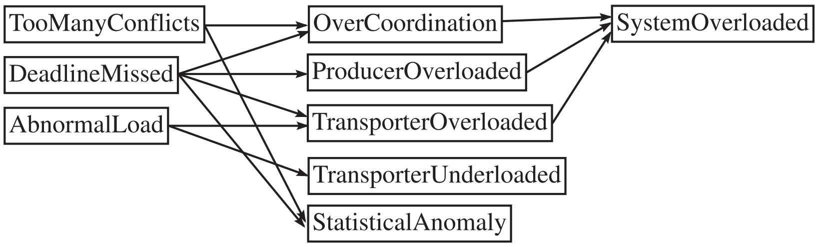

Figure 6: Causal model structure used in the PCT scenario.

when coordinating with other transporters over route usage. In this trial, the transporters' options regarding path selection were improved by constraining them in such a way to reduce the possibility of conflict. This was also implemented through local incremental change, this time by the transporters themselves, as they experimented with varying path probabilities (the chance that a particular route will be chosen) until one was found which incurred few conflicts. By lowering the potential for conflicts, the path probabilities reduced the overhead spent on both control decisions and coordination, which left more time for the actual act of transporting. Interestingly, despite similar final results, the organizational changes with both techniques available were very different than those of the previous trial. Periodic task reallocations were done every 100 pulses or so until time 800. Additionally, two to three path probability changes were made before time 200 for each of the transporters, and one or two more after time 700. These differences are caused by the fact that the adaptations were performed autonomously by individual agents in response to different efficiency metrics. Because no central authority governs the organizational changes, it is probable that the agents will adapt differently to the different metrics, but eventually settle on a similar result. Therefore, the various states the organization as a whole will go through towards this result will vary depending on the type of diagnosis being performed.

5. CONCLUSION

Generating an effective organizational structure for a multi-agent system is a crucial part of making them efficient, especially for large systems where global control is impractical. Adapting these organizations at runtime therefore becomes important when the environment, goals, or participants are liable to change. Several specific techniques have been offered by previous work in this area; we propose a more general solution to the problem by organizing such activity under the umbrella of diagnosis. A general diagnostic engine such as that shown in this paper is capable of detecting and diagnosing a variety of faults and inefficiencies, which can be used to drive organizational change. The organization itself is represented using models, such as TÆMS structures, which abstract the relevant portions of agents' capabilities and interactions in a way that facilitates both its use by agent control components and its adaptation by diagnosis. In this architecture, the methods driving change, and the characteristics affected by adaptation, can then be simplified to general techniques updating a domain independent representation, which can be reused from one system to the next.

A number of issues remain to be researched in this area. How efficient are the resulting organizations? How long does it take to discover a problem, and then to converge on a viable solution? Can one guarantee that the reactions will do no harm, and avoid oscillations? Our use of a blackboard structure, which can be searched for historical reactions, can help avoid these pitfalls. How much of the diagnostic engine can be domain independent, and how are reaction values and thresholds calculated. Of these these latter issues, can some sort of learning technique be used to automate value selection? We hope to address these areas in future work.

6. REFERENCES

[1] Ana L.C. Bazzan, Victor Lesser, and Ping Xuan. Adapting an Organization Design through Domain-Independent Diagnosis. Comp Sci Technical Report TR-98-014, University of Massachusetts at Amherst, February 1998.

[2] Daniel D. Corkill and Victor R. Lesser. The use of meta-level control for coordination in a distributed problem solving network. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, pages 748–755, Karlsruhe, Germany, August 1983.

[3] Keith S. Decker and Victor R. Lesser. Quantitative modeling of complex environments. *International Journal of Intelligent Systems in Accounting, Finance, and Management*, 2(4):215–234, December 1993. Special issue on "Mathematical and Computational Models of Organizations: Models and Characteristics of Agent Behavior".

[4] Edmund H. Durfee and Thomas A. Montgomery. Coordination as distributed search in a hierarchical behavior space. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6):1363–1378, 1991.

[5] Bryan Horling and Victor Lesser. Using diagnosis to learn contextual coordination rules. In *in Proceedings of the AAAI-99 Workshop on Reasoning in Context for AI Applications, a version also available as UMASS CS Tech Report TR99-15*. AAAI, 1999.

[6] Toru Ishida, Makoto Yokoo, and Les Gasser. An organizational approach to adaptive production systems. In *National Conference on Artificial Intelligence (AAAI-90)*, pages 52–58, 1990.

[7] D. Jensen, M. Atighetchi, R. Vincent, and V. Lesser. Learning quantitative knowledge for multiagent coordination. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, Orlando, FL, July 1999. AAAI.

[8] Onn Shehory, Katia Sycara, Prasad Chalasani, and Somesh Jha. Agent cloning: An approach to agent mobility and resource allocation. *IEEE Communications*, 36(7):58–67, July 1998.

[9] Y. So and E. H. Durfee. Modeling and designing computational organizations. In *Working Notes of the AAAI Spring Symposium on Computational Organization Design*, 1994.

[10] T. Sugawara and V. Lesser. Learning control rules for coordination. In *Multi-Agent and Cooperative Computation '93*, pages 121–136, 1993.

[11] Roy Turner and Elise Turner. Organization and reorganization of autonomous oceanographic sampling networks. In *Proceedings of the IEEE Interlational Conference on Robotics and Automation*, 1998.

[12] Thomas Wagner and Victor Lesser. Relating quantified motivations for organizationally situated agents. In N.R. Jennings and Y. Lespérance, editors, *Intelligent Agents VI — Proceedings of the Sixth International Workshop on Agent Theories, Architectures, and Languages (ATAL-99)*, Lecture Notes in Artificial Intelligence. Springer-Verlag, Berlin, 2000.