MULTI AGENT SYSTEMS AS A METHOD TO OPTIMIZE AN OBJECT PLACEMENT PROBLEM

Chris Venteicher

323 Frost Drive

Williams Bay, WI 53191

chris@venteicher.org

Introduction

Background, Motivation and Purpose

The primary purpose of this work was to develop practical knowledge about multi-agent systems. A number of questions exist in regard to the use and advantages of agent based systems. For example: When is it appropriate to use agents? How do you implement the agents? What should be communicated (and how do you do it)?

The work described in this paper seeks to answer these questions while improving the solution to an existing problem. The Author has previously developed software to search for the proper placement of an object in a 3 dimensional scene to reproduce a target image when the scene is rendered. This paper will examine how agents can be applied to the search process to increase the efficiency.

Overview of remaining sections

To this end the following paragraphs will start with a description of the existing search problem. An effort will then be made to examine agents and how they can best be applied to the problem.

A review of existing work will be presented. The review will cover materials related to the basics of agent implementations including the structure and choice of messages and the available agent platforms. Some more detailed documents related to constraint satisfaction problems and agent based searches are also discussed.

Based on the accumulated knowledge an agent search implementation will be proposed and examined. This will include details of the implementation as well as insight into the architectural choices made.

The results will be analyzed. This will include an analysis of traces showing the agent activity, numerical results showing the increase in efficiency associated with the agents and where the work should be focused in the future.

Existing Problem

Implementing a better solution to an existing problem was a good way to achieve the experience gathering goal of this work. Previously a system was developed to support searches for the proper placement of an object in 3 dimensional space to reproduce a target image. Given a target image and a 3D model of an object shown in the image, the system attempts to render a duplicate image that matches the target. To render a match the system must find the exact location of the object in 3D space that results in a match.

Finding the correct object placement is a difficult problem. The system algorithmically determines a new location to try, positions the object and renders a test image. The test image is compared against the target (original) image and a fitness values is computed.

In all likelihood an exact match was not produced and the system must continue to search for the proper location. Given the fitness of the last location a new location to test will be determined and another trial will be conducted. The system will continue to test new locations until the correct location is found.

Testing a location is an expensive task. Rendering the test image and determining the fitness of the result are computationally intensive. A maximum of approximately 100 tests can be done per second when a simple object and small target image are used.

Why is this search task interesting? Because it is being considered as part of the larger problem of how to compress video streams using the Mpeg4 standard (MPEG 4 Standard, 1998). The MPEG 4 standard allows objects in a video clip to be rendered on the fly at the receiver. To take advantage of this the encoder needs to identify the object and its exact location at every frame.

Related Work

General Principles

This was completely new work and nearly everything about the implementation was an open question. As a result it was appropriate to consult the literature to address a number of relevant questions.

For example: What is the advantage(s) of an agent based system and is it appropriate to this task? What type of agents should be created and what messages should they communicate? How should the agents be implemented?

"Agents are computer systems capable of flexible autonomous action in dynamic unpredictable, typically multi-agent domains" (Agent Link Home Page, 2002).

The scope of agent research is large and there are multiple definitions in the literature of what constitutes an agent. But there are a number of simple observations that can be made about software agents.

The first is that agents are distinct entities. Other agents exist concurrently, but each agent has its own distinct history and current state.

The second observation is that agents communicate with messages. Message based communication is key to how they operate. And the use and nature of the messages is a key difference between agent architectures and other ways of doing software (Labrou, 1999).

Agent Communication

Labrou, Finin and Peng provide a thoughtful discussion of agent communication languages (Labrou, 1999). In their paper they describe messages exchanged between agents as "speech acts." A "speech act" is an action and as such is intended to bring about a change in the world (Woodridge, 1995).

Labrou, Finin and Peng report that agent communication languages vary but share a common structure. In general the languages have a content layer, a message layer and a communications layer. The communication layer contains low level communication features like source and destination. The message layer specifies the speech act and is limited to the speech acts available to the language. The content layer, (which is a payload to the agent language) represents the material the speech act refers too. Table 1 lists common speech acts implemented by the agent platforms that were studied as part of this work.

Table 1 - Speech Acts Commonly Included In Agent Packages

REQUEST
AGREE
REJECT
INFORM
QUERY
CFP
PROPOSE
ACCEPT-PROPOSAL
REJECT-PROPOSAL
FAILURE
NOT-UNDERSTOOD

Constraint Satisfaction Problems

One of the fundamental varieties of agent problems that are examined is the constraint satisfaction problem (Yokoo 1996). The constraint satisfaction problem is typically defined as a set of variables and a set of constraints relating those variables. A solution to a CSP problem is a set of variable values that satisfies all of the constraints.

The problem of properly locating an object in space to reproduce an image can be mapped to the CSP framework. Consider one variable for each axis in space where the object could be randomly placed. A 3 dimensional placement problem could name the variables x, y and z. The search in the placement problem the constraint is location that duplicates the original image when the object is rendered. In terms of CSP constraints there is a value of the variables that puts the object in a position that matches the original image.

CSP solution search is generally NP-complete. But research indicates that some CSP problems are harder than others (Crawford, 1996). Research indicates that real CSP problems fall in to three classes: under-constrained, critically-constrained and over-constrained.

Crawford reports that under-constrained problems have a small number of constraints and are generally easy to solve because there are a large number of solutions. Over-constrained problems have a large number of constraints and are generally easy to solve because an algorithm can quickly converge on a solution. Critically-constrained problems have enough constraints that there are only a small number of solutions.

Likewise, critically-constrained problems don't have enough solutions to allow an algorithm to quickly converge on a solution. Critically-constrained problems are generally much harder than the other types because extensive searching must be done to find the solution.

What constraint type is the object placement problem? In normal images there is one global fitness maxima and it is relatively distinct from other solutions. In fact for normal images most positions will be of low fitness and fitness will increase dramatically and incrementally as the test object gets close to the correct position.

In general the object placement problem is over-constrained. There must be a search to find a location that is in the general area of the target. But once there starts to be object overlap the search can use relative fitness to converge on the target. In general the object placement problem is not of the intractable critically-constrained type.

But with this said some object / test image combinations will be critically-constrained. There may be other objects that look like the target but are not. There may be multiple instances of the target object within the image. In some cases the system will simply not converge.

It is important that the agents be able to handle these critically-constrained situations. The agents should not converge to quickly on what may be local minima. The agents should give up when it will take to long to find a good solution.

Agent based searches, Distributed plans

Existing work indicates an appropriate agent/task partition for CSP searches (Yokoo, 1996). Generally agents are allocated a part of the larger search space to search. The agents communicate information about their success or failure to the other agents in the group. The system adapts the search to the results obtained from the agents.

An equivalent architecture was adopted for this work. Multiple agents will conduct simultaneous searches over the possible positions for the object. The system will adapt based on the success or failure of the individual agents.

There are a number of communication methodologies that can be used to determine how the system adapts. To attempt to understand them it is valuable to consider the problem in human terms.

Consider as an example a search for a lost animal or child in a forest by a group of people. It is obvious the searchers want to split up and relay information to help each other out. But how do they get started? One option is for the group to get together and negotiate a plan. This is referred to as distributed planning. Another option is for an authoritative figure to take charge and allocate tasks to the participants. This is centralized planning.

After the planning is done the work starts. The workers will search independently and relay the result. If the planning is distributed the group receives the results and adapts the plan by negotiation.

There is a body of research that considers this type of distribute adaptation in terms of negotiated search between competing agents (Durfee, 1996). Although distributed search with distributed planning and control is possible and interesting the search can be managed by a central authority.

It is very possible an agent manages the workers. All the workers provide results directly to the manager and the manager adapts the search based on what it learns. The manager worker approach has merits both because it is simpler and because the workers will be contending for a shared resource.

Agents using shared resources

As was mentioned before rendering and evaluating images is a complex, resource-limited task. For the project a single shared render/evaluator entity will be used. And this shared resource will be the bottleneck in the system.

In the literature this is called a distributed constrained search (Durfee, 1996). A market-oriented strategy is appropriate to manage the contention for the resource.

The implemented system will use a simple mechanism to prioritize use of the resource by the searching agents. More sophisticated mechanism, perhaps based on contract nets or competition between agents, would be advantageous but is not possible for this implementation.

Agent platforms

A large number of public domain platforms exist. The Bee-gent, Zeus, FipaOs and Jade packages were examined (Bee-gent, 2001)(Zeus, 2001)(FipaOs, 2001)(Jade, 2001). Some packages are the result of university-based research and some are the result of corporate research. Most platforms are Java based. The platforms differ but in general are complex and appear to have a steep learning curve.

Most packages provide a graphical component for visualizing the messages passed between agents many provide editors to assist in defining the agents and the message contents. As is the case with most Unix GUI implementation these GUI interfaces look to be of questionable utility.

The Zeus, FipaOs, Jade and Bee-gent implementations were downloaded and examined. Examples were configured and tested. The UI's were started and used to manipulate the examples. The code in the examples was examined.

All the platforms provide the implementation of message passing between the agents. The platforms each provide a definition of the message that can be exchanged. That definition is likely extensible. Most center around supporting the FIPA definition of the message content.

Many of the platforms use messages based around a dynamic or self-defining message structure based on KQML, XML etc. Platforms based on the standardized FIPA communication protocol look the most appropriate to take advantage of the ability of agents to interact with dissimilar entities.

The conclusion was reached the UI editors would likely not prevent the user from needing to completely learn the underlying system. There were various degrees of complexity associated with each but it was clear mastering the complexity would be very time consuming. Based on the perceptions of a step learning curve for both the agent package and Java the decision was made to avoid the packages for the initial implementation of this work. However, close examination of the packages identified the functionality needed from any implementation.

Description of experimental work and results

Using the knowledge obtained from related work an agent based searcher was developed. This section will describe both the architecture and implementation of the system.

Agent entities

Three types of agents were created: shared resource, manager and worker.

A Single manager agent was created to control the search. The manager creates workers to perform the actual search. And, a single shared resource was created to handle the rendering/fitness evaluation needs of the workers.

The shared resource agent is fundamentally different than the other two agent types. This agent represents the interface to the code that does the rendering and image delta operations. The shared resource was made an agent because it simplifies the implementation not because it is a proactive agent. Making the shared resource an agent allows the worker agents to communicate with the entire world through a single message based interface.

It should be noted that the shared resource is serialized. Rendering 3-D images and calculating deltas between those images takes time. And, the work can't be done in parallel (on the test system.) The shared resource is the bottleneck in system. The worker agents are in competition for the resource. And the manager controls the allocation of the resource in addition to the task allocation.

Communication (Primitives and Constructs)

Table 2 details what intents are supported for each agent type, who they can be communicated to, the speech act used to convey the intent and the associated parameter payloads.

Table 2 – Supported Agent Intentions

Agent	Intent	Destination	Speech Act	Message Payload
Manager	Request Worker To Start Task	Worker	Request	Starting Position, Starting Priority, Shared Resource Credits
	Request Worker To Abort Task	Worker	Request	
	Request Status from Worker	Worker	Request	
	Request Worker Priority Change	Worker	Request	New Priority
Worker	Request Shared Resource To Start Task	Shared Resource	Request	Position to test, Worker's Priority
	Inform Manager Search Task Done	Manager	Inform	Ending Position, Ending Fitness value
	Agree To Search Task	Manager	Agree	
	Refuse Search Task	Manager	Refuse	
	Agree To Abort Search Task	Manager	Agree	
	Refuse To Abort Search Task	Manager	Refuse	
Shared Resource	Inform Worker Render Task Done	Worker	Inform	Resulting fitness
	Agree To Render Task	Worker	Agree	
	Refuse Render Task	Worker	Refuse	

Operation begins after the shared resource and manager agents are instantiated. The manager proceeds to create a predefined number of worker agents.

The worker agents are given a task allocation and credits to use with the shared resource agent. The workers proceed with their tasks until completion. The workers periodically run out of credits and must request more from the manager.

The manager monitors the progress of the worker agents. Based on the relative progress of the agents the manager can alter the percentage of the shared resource each agent receives. The manager could also task or re-task agents to match the new information but for simplicity does not currently do so.

SIMULATION OF RENDERING TASK

To generate these results only modification of task priorities to limit and control use of the shared resource was evaluated. The search space was not partitioned and each worker was assigned the entire space. However, the search algorithm used by the agents was chosen to minimize duplication of effort among the agents.

There are many opportunities to extend this work to give the agents more control over the search process. However, control over task priorities does allow for a demonstrative example. The manager will clearly be able to learn from the progress of the individual agents and direct resources at the agents that are showing the best results.

It should be noted that the shared resource is serial. Because the agents are not making progress in parallel, focusing the shared resource on a particular agent is functionally similar to directing multiple agents to the same area.

Another simplification was made to facilitate the example. The shared resource agent was only implemented to operate on one variable rather than 6. In other words, a 1 dimensional placement problem was considered. In the original problem X, Y, Z position and X, Y, Z-axis rotation were variable.

For this work only the X position was allowed to change. This task is equivalent to correctly placing a line segment of fixed length within a finite range. The test evaluated here is significantly simpler than the full test but sufficient to demonstrate the feedback and optimization by the agents. Future work will need to test the full dimensional search.

Software implementation

The project was implemented using a simplified agent framework. Originally the simplified framework was envisioned as a quick way to describe the operation in C++ before moving to a pre-existing Java based agent framework. However, the C++ framework proved to be easy to implement and incorporated the fundamental principles from the other packages. The complexity of the pre-existing packages (as well as an unfamiliarity with Java) was a strong motivator to keep the simple implementation.

Figure 1 describes the software implementation. In this code agents are represented as C++ classes. Each unique type of agent (shared Resource, manager or worker) was specified as a specialized form of a Meta-agent class. The specialized classes encapsulated functionality unique to each agent type. However, all agents implemented an event handler to receive events.

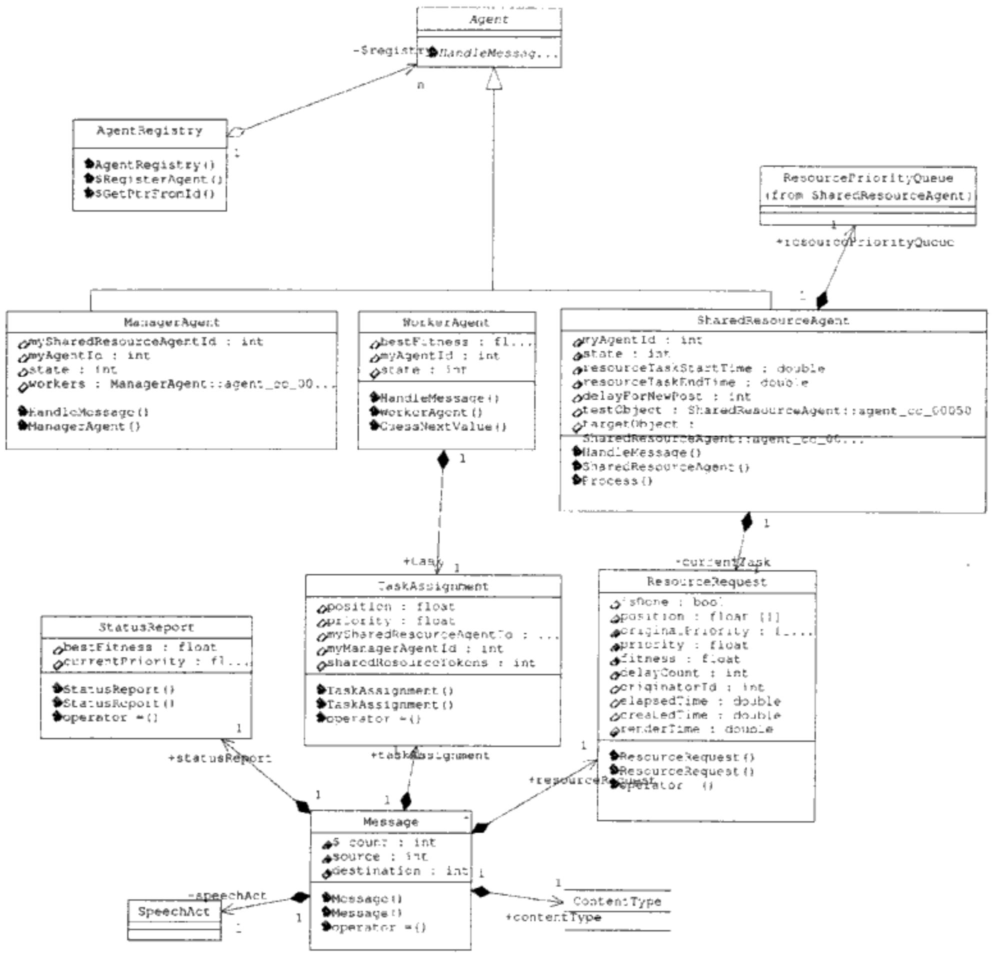
The main function of the code establishes the static agents in the system. The static agents are the shared resource agent and the manager agent. All agents created by the project are registered in the agent registry. Each new instantiation has a unique ID to facilitate the routing of messages.

A message queue is implemented in STL to buffer the messages between the agents. After the main function creates the two static agents a loop is entered. Within that loop messages are sequentially pulled from the STL message buffer and routed to the destination encoded in the message. Each agent implements a message handler function that is called to by the main function to deliver the message.

In the loop the main function periodically sends a message to all the agents listed in the agent registry. The message sent by the main loop indicates a time slice and wakes all agents up to do any pending processing they may have to do.

Figure 1 - Software Implementation UMT Diagram

agent



File: /home/cven/agentOut.mdl Sat Jan 19 11:54:10 2002 Class Diagram: Class / Reverse Engineered
Page 1

When the manager agent receives its first time slice there are no worker agents in existence. The manager instantiates a worker agent. The worker agent automatically registers with the agent registry so it can be identified and can receive time slices in the future.

Each time the manager receives a time slice it will instantiate another worker agent until the number of workers reaches a predefined amount. This interface supports the idea of a manager dynamically adding or removing agents based on changing needs. But that capability was not explored for this work.

Manager, Worker and Task Allocation

The manager allocates worker agents a task. The task consists of a description of the bounds of the search space and the initial position of the test object in that space.

It is possible for the manager to define a task with a search space that is only a portion of the global search space but that capability was not experimented with for this work. Rather it seemed sufficient to randomly determine the initial position of the test object and search by making random steps starting from the initial position. The search is not as efficient because there can be overlap among the agents. However, because of the random initialization of position and random movements the overlap should be limited.

Shared resource

Once allocated a task, workers contend with each other for use of the shared resource. The shared resource agent encapsulates the shared resource. The worker agents communicate directly with the shared resource agent to evaluate new positions for their test objects. It is by requesting tests and receiving results that the workers direct their searches.

Rendering a test image from a test object and algorithmically comparing the result to an original image is time consuming. This project uses 1 dimensional search space. As a result it takes very little time to do the rendering. But the purpose of this work is to address a problem where rendering is the system bottleneck. To be true to this purpose the shared resource agent waits a random amount of time to return the results to the worker. The wait is equivalent to what a full multi-dimensional implementation would normally experience as the time required to render and evaluate a test.

Because of the induced time delay the shared resource is in fact the bottleneck in this system as would be expected. As a result the agents will be contending for the use of the resource.

The shared resource agent will receive requests from the workers and place them in a separate STL priority queue. When the shared resource agent is free it will pull the highest priority request off of the queue and begin processing it.

The workers resource requests contain a priority value assigned by the manager. The shared resource agent will select the highest priority requests from the queue first before considering order in which they are received.

This scheme works well for giving some worker agents priority over the others. However it does allow the high priority agents to completely dominate the resource. In the interest of fairness there is a mechanism that will increment the priority of all queued requests each time a request is de-queued As a result even the requests from the lowest priority worker will get serviced after spending a maximum amount of time on the shared resource agents queue. It should also be noted that worker agents only make one shared resource request at a time.

Learning and Adaptation by Manager

The manager agent will periodically request a status update from each worker. The worker receives the request and is mandated to send the current position and fitness of the test object back to the manager.

The manager records the position and fitness. If the worker has found a position of suitable fitness the manager can deactivate all of the workers. Otherwise the manager ranks the performance of this agent against the other agents. Based on the new rankings the manager sends a priority update message to all the workers requiring a priority change.

The aggressiveness with which the manager queries status from the workers and updates the priorities is configurable. By altering the rate we can demonstrate the relative effectiveness of this capability.

ANALYSIS OF RESULTS

Multiple types of results were obtained from this project. The primary result was the accumulation of knowledge. This includes knowledge of when, where, how and why to implement agent systems.

In addition, specific results were obtained from the agent-based searcher. The searcher produces three forms of output. That output can be used to analyze the operation of the system.

The first form of output from the searcher describes the messages that are being sent between the agents. This output is text dumped to standard out by the program. When the text is piped through the program "msc" (msc, 2001). A nicely format message sequence chart is generated.

The second form of output consists of files of the form "agentX.out" where "X" is the id of the agent. These files contain records that capture the results of each shared resource request the workers make. To be more specific, each line in this file describes each search attempt in terms of when it was completed, how fit the rendered image was and what priority the request had (i.e. the same as the workers priority) when it entered the shared resource agent's task queue.

The third form of output from the program is a real time trace of the activities of one or more of the agents. The search program instantiates an xmgrace (xmgrace, 2001) window at startup and updates the window with new data as the program runs. This form of output was very valuable while debugging and configuring the system.

Table 3 presents a message sequence chart that shows the interaction between one worker, manager and shared resource agent. This table is taken from the larger table presented in the appendix that shows the simultaneous interaction of multiple agents

Table 3 – Manager, Worker, Shared Resource Interaction

Message		Manager	Agent2	Agent3	Agent4	Resource
REQUEST_TASK_START		------>				
AGREE_TASK_START		<------				
REQUEST_TASK_START			-------+	-------+	------>	
AGREE_TASK_START			<------+	-------+	-------	
INFORM_TASK_DONE			<------+	-------+	-------	
... repeat ...						
REQUEST_TASK_START			-------+	-------+	------>	
AGREE_TASK_START			<------+	-------+	-------	
INFORM_TASK_DONE			<------+	-------+	-------	
INFORM_TASK_DONE		<------				
REQUEST_STATUS		------>				
AGREE_STATUS		<------				
INFORM_STATUS		<------				
REQUEST_PRIORITY_CHANGE		------>				
AGREE_PRIORITY_CHANGE		<------				
REQUEST_TASK_START		------>				
AGREE_TASK_START		<------				
REQUEST_TASK_START			-------+	-------+	------>	
AGREE_TASK_START			<------+	-------+	-------	
INFORM_TASK_DONE			<------+	-------+	-------	
... repeat ...						

Table 3 shows a speech act based dialog between the agents. Concatenating the speech act with the subject of the act forms the message entries. Although speech acts are independent of the subject during normal operation they are concatenated when output to facilitate the generation of the message sequence chart.

Per the previous discussion it can be seen that the manager will assign search tasks to the worker. The worker will make use of the shared resource to evaluate the positions in the search. And the shared resource will inform the manager of the fitness of the specified position.

As part of the task allocation the manager specifies how many search attempts the worker should make before it notifies the manager that the task is complete. When the worker has expended all of its attempts it will inform the manager the task is done. The manager will then provide a new task to the worker if a fit enough solution has not been found.

In the current implementation the new task assigned to the worker will instruct the worker to continue from where it left off on the previous task. The manager will pass the final position of the last task as the first position of the new task. Every assigned task is given the entire range of the search space in this implementation. Subsequent task assignments also do not manipulate the task search space. Future work should investigate using modifications in both the initial position and search space bounds to improve efficiency.

The manager does actively control the priority assigned to the workers while they perform their task assignments. Periodically the manager will request status reports from individual workers. The workers are obliged to reply with the fitness result of their last search. The manager will use this information to rank the workers and assign priorities based on the results.

Currently the manager requests a status report from a worker after it has completed a fixed number of tasks. Typically the program is configured to have the worker complete between two and five task before a status report is requested.

Once the manager receives the status report and recalculates worker priorities all workers whose priorities have changed are notified by a priority change request. The workers must submit the new priority on all subsequent requests to the shared resource manager. Table 3 shows the status and priority exchange for a single agent. The message sequence chart in the appendix shows the priorities of multiple agents being altered in response to a status request.

Priority changes represent the mechanism this implementation uses to dynamically alter behavior at run time. Dynamic optimization represents perhaps the key reason why multi agent systems are useful. Agents are a layer between the jobs and the person or thing that wants the job done and as such can dynamically direct resources to get the job done as efficiently as possible.

Search Performance

The following graph shows the relative progress of the worker agents during a typical run. As can be seen from the graph each worker starts out with about the same level of success in its search efforts. Keep in mind each worker was initialized to start at a random point in the search space.

Figure 2

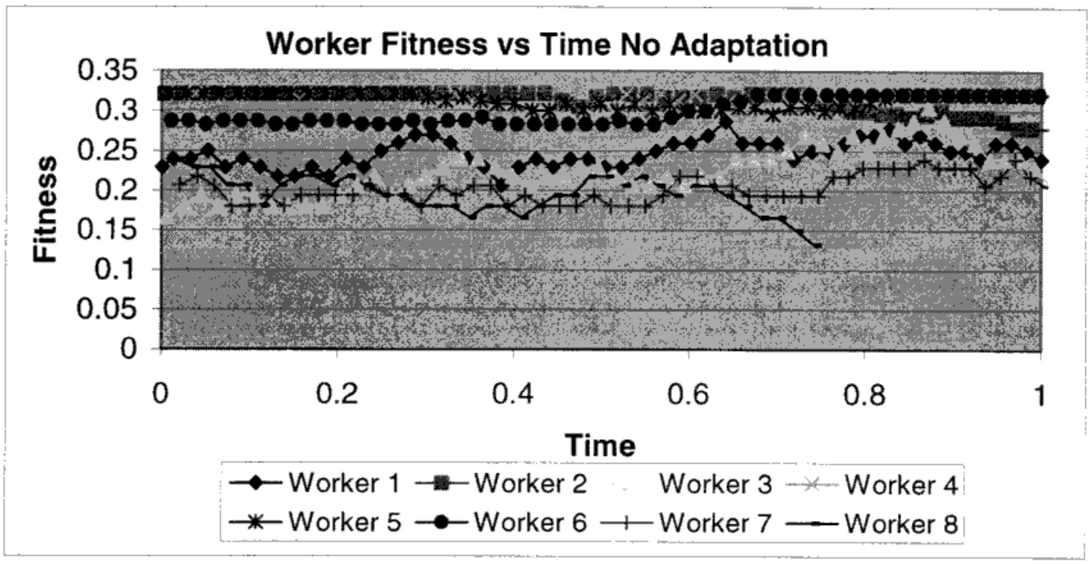
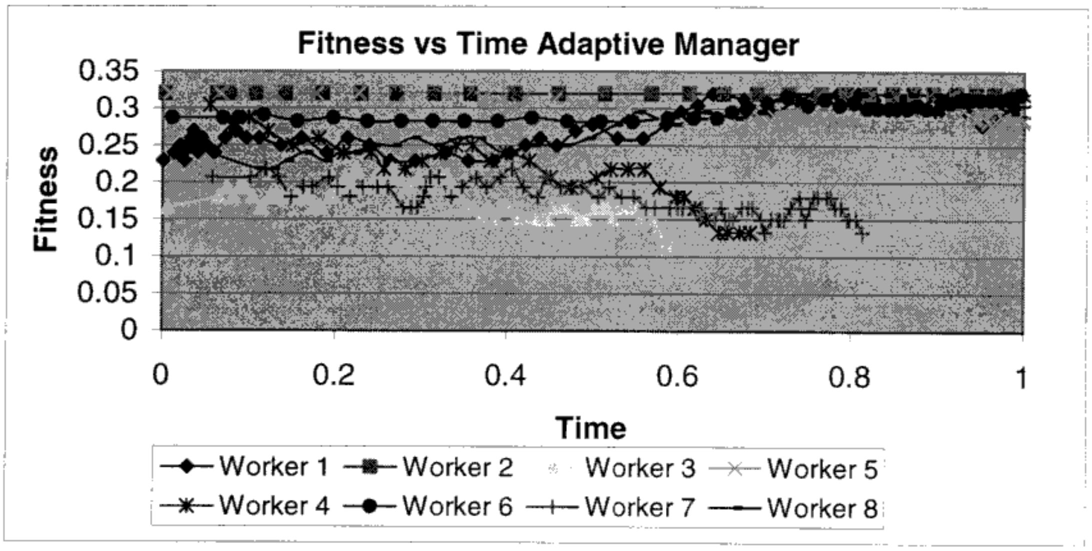


Figure 3



The workers were free to move in constrained random steps after initialization. The graph shows that as one of the workers makes contact with the target object it's success is noted. The worker makes contact after its random steps bring the workers test object over some part of the target object. When this happens the manager agent takes notice of the increased fitness associated with the worker and it's task.

The graph shows shared resource allocation among the worker agents. As the manager agent takes note of the success being experienced by one of the workers, resources are directed towards that worker. As the prized worker's search better positions the test object it gets more and more resources. In other words the successful worker agent gets to make many more trials than the unsuccessful full agents.

Another graph shows a run where the manager does not adjust resource allocation based on performance. As can be seen in this graph the agents take longer to find the solution. More importantly the agents take longer to find the first acceptable solution.

CONCLUSION

This work demonstrated the addition of an agent layer to a pre-existing domain specific search problem. The resulting system provided a good illustration of the issues associated with agent implementations. The example also demonstrated how adding the agent layer could result in more efficient and dynamic solutions.

The project implemented the messaging structure and the messages communicated by the agents. The developed code could be used as a basis for extending functionality to include more complicated interactions, or the project could be switch to an existing agent package.

The results demonstrate how agents can be added as a layer to control a pre-existing product. The example presented here simulates the operation of the underlying searcher and operates on only one dimension of a multidimensional search space. However results indicate that the agent implementation improved performance on the limited domain. It is reasonable to expect the results could be extended to the actual target system implementing the full search space.

A good deal of time was spent examining existing software packages before the decision was made to implement a simpler solution. The knowledge will be valuable if this work is extended. The C++ implementation could also be used to quickly prototype other agent implementations.

REFERENCES

Agent Link Home Page, (n.d.), Retrieved January 10, 2002, from http://www.agentlink.org ISO/IDE. (1998). International Standard 14496-1 (MPEG 4-Generic Coding of Audio-Visual Objects).

Labrou, Y., Finin, T. and Peng, Y. (1999). The current landscape of Agent Communication Languages, IEEE Intelligent Systems, vol. 14, no. 2

Woodridge, M. and Jennings, N.R. (1995). Intelligent Agents: Theories and Practice, Knowledge Engineering Review, vol. 10, no. 2

Agent Communication Language (ACL). (n.d.). Retrieved January 10, 2002, from http://www2.toshiba.co.jp/beegent/acl/acl.htm

Yokoo. M., and Ishida, T. (1996). Search Algorithms for Agents. In G. Weiss (Eds.), Multi-Agent Systems A Modern Approach to Distributed Artificial Intelligence (pp. 165-200)

Durfee. E., (1996). Distributed Problem Solving and Planning In G. Weiss (Eds.), Multi-Agent Systems A Modern Approach to Distributed Artificial Intelligence (pp. 165-200)

Wellman, M.P. (1993). A Market-Oriented Programming Environment and its Application to Distributed Multicommodity Flow Problems, Journal of Artificial Intelligence Research

Tutorial for Bee-gent. (n.d.). Retrieved January 10, 2002, from http://www2.toshiba.co.jp/beegent/tutorial/tindex.htm

The ZEUS Technical Manual. (n.d.). Retrieved January 10, 2002, from http://www.btexact.com/projects/agents/zeus/library/zeus-tm-doc.zip

Jade Programmers Guide (n.d.). Retrieved January 10, 2002, from http://sharon.cselt.it/projects/jade/jadeDoc.zip

FIPA-OS Developers' Guide, (n.d.), Retrieved January 10, 2002, from http://fipa-os.sourceforge.net/docs/Developers_Guide.pdf

Grace Graphing Program Home Page, (n.d.), Retrieved January 10, 2002, from http://plasma-gate.weizmann.ac.il/Grace/index.htm

MSC - Message Sequence Chart Generator Home Page, (n.d.), Retrieved January 10, 2002, from http://www.users.bigpond.com/rubensr/msc/index.htm

APPENDIX

Figure 4

```
                    Agent99 Manager Agent2  Agent3  Agent4  Agent5  Agent6  Agent7  Agent8  Agent9  Agent10 Shared Resource
                    |       |       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START  |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_TASK_START    |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START  |       |-------+------>|       |       |       |       |       |       |       |       |
REQUEST_TASK_START  |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START    |       |<------+-------|       |       |       |       |       |       |       |       |
AGREE_TASK_START    |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START  |       |-------+-------+------>|       |       |       |       |       |       |       |
REQUEST_TASK_START  |       |       |       |-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START    |       |<------+-------+-------|       |       |       |       |       |       |       |
AGREE_TASK_START    |       |       |       |<------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START  |       |-------+-------+-------+------>|       |       |       |       |       |       |
REQUEST_TASK_START  |       |       |       |       |-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START    |       |<------+-------+-------+-------|       |       |       |       |       |       |
AGREE_TASK_START    |       |       |       |       |<------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START  |       |-------+-------+-------+-------+------>|       |       |       |       |       |
REQUEST_TASK_START  |       |       |       |       |       |-------+-------+-------+-------+-------+------>|
AGREE_TASK_START    |       |<------+-------+-------+-------+-------|       |       |       |       |       |
AGREE_TASK_START    |       |       |       |       |       |<------+-------+-------+-------+-------+-------|
REQUEST_TASK_START  |       |-------+-------+-------+-------+-------+------>|       |       |       |       |
REQUEST_TASK_START  |       |       |       |       |       |       |-------+-------+-------+-------+------>|
AGREE_TASK_START    |       |<------+-------+-------+-------+-------+-------|       |       |       |       |
AGREE_TASK_START    |       |       |       |       |       |       |<------+-------+-------+-------+-------|
INFORM_TASK_DONE    |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START  |       |-------+-------+-------+-------+-------+-------+------>|       |       |       |
REQUEST_TASK_START  |       |       |       |       |       |       |       |-------+-------+-------+------>|
AGREE_TASK_START    |       |<------+-------+-------+-------+-------+-------+-------|       |       |       |
AGREE_TASK_START    |       |       |       |       |       |       |       |<------+-------+-------+-------|
REQUEST_TASK_START  |       |-------+-------+-------+-------+-------+-------+-------+------>|       |       |
REQUEST_TASK_START  |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
REQUEST_TASK_START  |       |       |       |       |       |       |       |       |-------+-------+------>|
AGREE_TASK_START    |       |<------+-------+-------+-------+-------+-------+-------+-------|       |       |
AGREE_TASK_START    |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
AGREE_TASK_START    |       |       |       |       |       |       |       |       |<------+-------+-------|
REQUEST_TASK_START  |       |       |       |       |       |       |       |       |       |-------+------>|
AGREE_TASK_START    |       |       |       |       |       |       |       |       |       |<------+-------|
INFORM_TASK_DONE    |       |       |       |<------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START  |       |       |       |-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START    |       |       |       |<------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE    |       |       |       |       |<------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START  |       |       |       |       |-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START    |       |       |       |       |<------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE    |       |       |       |       |       |       |<------+-------+-------+-------+-------|
REQUEST_TASK_START  |       |       |       |       |       |       |-------+-------+-------+-------+------>|
AGREE_TASK_START    |       |       |       |       |       |       |<------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |       |       |       |       |       |<------+-------+-------+-------|
REQUEST_TASK_START      |       |       |       |       |       |       |       |-------+-------+-------+------>|
AGREE_TASK_START        |       |       |       |       |       |       |       |<------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_STATUS          |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_STATUS            |       |<------|       |       |       |       |       |       |       |       |       |
INFORM_STATUS           |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE |       |------>|       |       |       |       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE |       |-------+------>|       |       |       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE |       |-------+-------+------>|       |       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE |       |-------+-------+-------+------>|       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE |       |-------+-------+-------+-------+------>|       |       |       |       |       |
REQUEST_PRIORITY_CHANGE |       |-------+-------+-------+-------+-------+------>|       |       |       |       |
REQUEST_PRIORITY_CHANGE |       |-------+-------+-------+-------+-------+-------+------>|       |       |       |
REQUEST_PRIORITY_CHANGE |       |-------+-------+-------+-------+-------+-------+-------+------>|       |       |
AGREE_PRIORITY_CHANGE   |       |<------|       |       |       |       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE   |       |<------+-------|       |       |       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE   |       |<------+-------+-------|       |       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE   |       |<------+-------+-------+-------|       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE   |       |<------+-------+-------+-------+-------|       |       |       |       |       |
AGREE_PRIORITY_CHANGE   |       |<------+-------+-------+-------+-------+-------|       |       |       |       |
AGREE_PRIORITY_CHANGE   |       |<------+-------+-------+-------+-------+-------+-------|       |       |       |
AGREE_PRIORITY_CHANGE   |       |<------+-------+-------+-------+-------+-------+-------+-------|       |       |
REQUEST_TASK_START      |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_TASK_START        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_TASK_START        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_TASK_START        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_STATUS          |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_STATUS            |       |<------|       |       |       |       |       |       |       |       |       |
INFORM_STATUS           |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_TASK_START        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_TASK_START        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_TASK_START        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_STATUS          |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_STATUS            |       |<------|       |       |       |       |       |       |       |       |       |
INFORM_STATUS           |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_TASK_START        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |------>|       |       |       |       |       |       |       |       |       |
AGREE_TASK_START        |       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |       |       |       |       |       |       |       |<------+-------|
REQUEST_TASK_START     |       |       |       |       |       |       |       |       |-------+------>|
AGREE_TASK_START       |       |       |       |       |       |       |       |       |<------+-------|
INFORM_TASK_DONE       |       |       |       |       |<------+-------+-------+-------+-------+-------|
REQUEST_TASK_START     |       |       |       |       |-------+-------+-------+-------+-------+------>|
AGREE_TASK_START       |       |       |       |       |<------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |       |       |       |       |       |       |<------+-------+-------|
REQUEST_TASK_START     |       |       |       |       |       |       |       |-------+-------+------>|
AGREE_TASK_START       |       |       |       |       |       |       |       |<------+-------+-------|
INFORM_TASK_DONE       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START     |------>|       |       |       |       |       |       |       |       |       |
AGREE_TASK_START       |<------|       |       |       |       |       |       |       |       |       |
REQUEST_TASK_START     |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |       |<------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |<------+-------|       |       |       |       |       |       |       |       |
REQUEST_STATUS         |-------+------>|       |       |       |       |       |       |       |       |
AGREE_STATUS           |<------+-------|       |       |       |       |       |       |       |       |
INFORM_STATUS          |<------+-------|       |       |       |       |       |       |       |       |
REQUEST_TASK_START     |-------+------>|       |       |       |       |       |       |       |       |
AGREE_TASK_START       |<------+-------|       |       |       |       |       |       |       |       |
REQUEST_TASK_START     |       |       |-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START       |       |       |<------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |       |       |<------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |<------+-------+-------|       |       |       |       |       |       |       |
REQUEST_STATUS         |-------+-------+------>|       |       |       |       |       |       |       |
AGREE_STATUS           |<------+-------+-------|       |       |       |       |       |       |       |
INFORM_STATUS          |<------+-------+-------|       |       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE|-------+-------+------>|       |       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE|------>|       |       |       |       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE|-------+------>|       |       |       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE  |<------+-------+-------|       |       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE  |<------|       |       |       |       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE  |<------+-------|       |       |       |       |       |       |       |       |
REQUEST_TASK_START     |-------+-------+------>|       |       |       |       |       |       |       |
AGREE_TASK_START       |<------+-------+-------|       |       |       |       |       |       |       |
REQUEST_TASK_START     |       |       |       |-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START       |       |       |       |<------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |       |       |       |       |<------+-------+-------+-------+-------|
INFORM_TASK_DONE       |<------+-------+-------+-------+-------|       |       |       |       |       |
REQUEST_STATUS         |-------+-------+-------+-------+------>|       |       |       |       |       |
AGREE_STATUS           |<------+-------+-------+-------+-------|       |       |       |       |       |
INFORM_STATUS          |<------+-------+-------+-------+-------|       |       |       |       |       |
REQUEST_PRIORITY_CHANGE|-------+-------+-------+-------+------>|       |       |       |       |       |
REQUEST_PRIORITY_CHANGE|-------+-------+-------+------>|       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE  |<------+-------+-------+-------+-------|       |       |       |       |       |
AGREE_PRIORITY_CHANGE  |<------+-------+-------+-------|       |       |       |       |       |       |
REQUEST_TASK_START     |-------+-------+-------+-------+------>|       |       |       |       |       |
AGREE_TASK_START       |<------+-------+-------+-------+-------|       |       |       |       |       |
REQUEST_TASK_START     |       |       |       |       |       |       |-------+-------+-------+-------+------>|
AGREE_TASK_START       |       |       |       |       |       |       |<------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |       |       |       |       |       |       |<------+-------+-------+-------|
INFORM_TASK_DONE       |       |<------+-------+-------+-------+-------+-------|       |       |       |       |
REQUEST_STATUS         |       |-------+-------+-------+-------+-------+------>|       |       |       |       |
AGREE_STATUS           |       |<------+-------+-------+-------+-------+-------|       |       |       |       |
INFORM_STATUS          |       |<------+-------+-------+-------+-------+-------|       |       |       |       |
REQUEST_PRIORITY_CHANGE|       |-------+-------+-------+-------+-------+------>|       |       |       |       |
REQUEST_PRIORITY_CHANGE|       |-------+------>|       |       |       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE|       |-------+-------+-------+-------+------>|       |       |       |       |       |
REQUEST_PRIORITY_CHANGE|       |-------+-------+-------+------>|       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE  |       |<------+-------+-------+-------+-------+-------|       |       |       |       |
AGREE_PRIORITY_CHANGE  |       |<------+-------|       |       |       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE  |       |<------+-------+-------+-------+-------|       |       |       |       |       |
AGREE_PRIORITY_CHANGE  |       |<------+-------+-------+-------|       |       |       |       |       |       |
REQUEST_TASK_START     |       |-------+-------+-------+-------+-------+------>|       |       |       |       |
AGREE_TASK_START       |       |<------+-------+-------+-------+-------+-------|       |       |       |       |
REQUEST_TASK_START     |       |       |       |       |       |       |       |-------+-------+-------+------>|
AGREE_TASK_START       |       |       |       |       |       |       |       |<------+-------+-------+-------|
INFORM_TASK_DONE       |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START     |       |       |-------+-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START       |       |       |<------+-------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |       |       |       |<------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START     |       |       |       |       |-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START       |       |       |       |       |<------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |       |       |<------+-------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START     |       |       |       |-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START       |       |       |       |<------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |       |       |       |       |<------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |<------+-------+-------+-------|       |       |       |       |       |       |
REQUEST_STATUS         |       |-------+-------+-------+------>|       |       |       |       |       |       |
AGREE_STATUS           |       |<------+-------+-------+-------|       |       |       |       |       |       |
INFORM_STATUS          |       |<------+-------+-------+-------|       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE|       |-------+-------+-------+------>|       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE|       |-------+-------+-------+-------+-------+------>|       |       |       |       |
REQUEST_PRIORITY_CHANGE|       |-------+------>|       |       |       |       |       |       |       |       |
REQUEST_PRIORITY_CHANGE|       |-------+-------+-------+-------+------>|       |       |       |       |       |
AGREE_PRIORITY_CHANGE  |       |<------+-------+-------+-------|       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE  |       |<------+-------+-------+-------+-------+-------|       |       |       |       |
AGREE_PRIORITY_CHANGE  |       |<------+-------|       |       |       |       |       |       |       |       |
AGREE_PRIORITY_CHANGE  |       |<------+-------+-------+-------+-------|       |       |       |       |       |
REQUEST_TASK_START     |       |-------+-------+-------+------>|       |       |       |       |       |       |
AGREE_TASK_START       |       |<------+-------+-------+-------|       |       |       |       |       |       |
REQUEST_TASK_START     |       |       |       |       |       |-------+-------+-------+-------+-------+------>|
AGREE_TASK_START       |       |       |       |       |       |<------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |       |       |       |<------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE       |       |<------+-------+-------|       |       |       |       |       |       |       |
REQUEST_TASK_START     |       |-------+-------+------>|       |       |       |       |       |       |       |
AGREE_TASK_START       |       |<------+-------+-------|       |       |       |       |       |       |       |
REQUEST_TASK_START     |       |       |       |       |-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |       |<------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |<------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |<------|       |       |       |       |       |       |       |       |
REQUEST_STATUS          |       |------>|       |       |       |       |       |       |       |       |
AGREE_STATUS            |       |<------|       |       |       |       |       |       |       |       |
INFORM_STATUS           |       |<------|       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |------>|       |       |       |       |       |       |       |       |
AGREE_TASK_START        |       |<------|       |       |       |       |       |       |       |       |
REQUEST_TASK_START      |       |       |-------+-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |<------+-------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |       |       |       |       |<------+-------+-------+-------|
REQUEST_TASK_START      |       |       |       |       |       |       |-------+-------+-------+------>|
AGREE_TASK_START        |       |       |       |       |       |       |<------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |       |<------+-------+-------+-------+-------+-------+-------|
REQUEST_TASK_START      |       |       |       |-------+-------+-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |       |<------+-------+-------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |       |       |       |<------+-------+-------+-------+-------|
REQUEST_TASK_START      |       |       |       |       |       |-------+-------+-------+-------+------>|
AGREE_TASK_START        |       |       |       |       |       |<------+-------+-------+-------+-------|
INFORM_TASK_DONE        |       |       |       |<------+-------+-------+-------+-------+-------+-------|
```