MULTI - AGENT SYSTEMS FOR CONTROL, DIAGNOSTIC AND MONITORING

I. Valova, J. Zaprianov
Institute of Control and Systems Research
Bulgarian Academy of Sciences
Acad. G. Bonchev str., P.O. Box 79, Sofia 1113, Bulgaria

Abstract

The theory of multi-agent systems arises from the trend of developing better Intelligent systems and of distributing computing more and more.

Many important computing applications such as, process control, robotic, communications networks will benefit from using Multi Agent System (MAS) approach.

In this article we present a model for using intelligent agents to assist operators to recognize potential problem quickly enough and have been able to assist in preventing a wide range of real-life failure cases. Although the described method is too general it is flexible and applied into the real-time process industry. This paper discusses also Knowledge Broker and Dynamic Knowledge DataBase. The term Knowledge Broker as a particular kind of intelligent computer broker that deals in knowledge, in the way of keeping, querying, distributing it or communicating it, whether as a primary or secondary function.

Our treatment in this paper will be twofold; one an outline of the issue's underlying theoretical framework within the field of Artificial Intelligence (AI), another a look software for practical implementations.

This is idea for using intelligent agents to assist operators to recognize potential problem quickly enough and has been able to assist in preventing a wide range of real – life failure cases.

1 INTRODUCTION

Intelligent agents and intelligent systems are widely used by AI (artificial intelligence) community and in other disciplines, but there isn't a common definition of them. MAS usually contain a subset of the following functionality:

- Knowledge and problem-solving systems are able to process knowledge to solve problems through what looks like reasoning
- Intelligent planning systems advise on actions and set up or help to establish the sequence of event that form a plan.
- Learning systems start with information about a domain and the extend the knowledge in some way, sometimes through inference.

2 THE ARCHITECTURE OF SUGGESTED MAS

Fig.1 shows architecture of MAS, which consists of three layers: the agents layer, the intelligent knowledge broker (IKB) layer, and the Dynamic Knowledge Data Base (DKDB) layer.

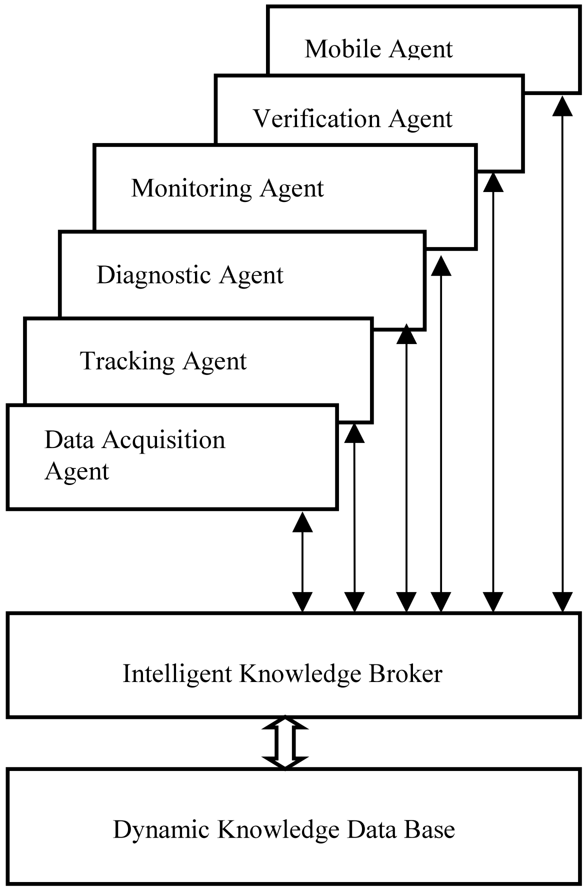

Fig.1 Relationship of the various agents to the IKB and the DKDB

The agent layer is essentially an application layer consisting of all agents.

The IKB layer manages communication between the agents and the DKDB.

And the DKDB layer stores knowledge used by all the agents and the IKB.

2.1 The agent layer

Each agent performs a separate function:

The data acquisition agent receives real – time data from all current process frame by frame .The IKB then passes the frame to all the other agents so that they understand everything the data acquisition agent know.

The tracking agent continuously updates the data links between the agent system and the actual state of the system.

On the basis of the real-time data, the simulation model in the tracking agent produces a single event object for each frame. The event object contains simulated values for how the system should be operating at all times. The tracking agent then passes the event object through the IKB to the monitoring agent, which infers conclusions from the available information and returns a set of symbolic description of the state of the system to the diagnostic agent.

The monitoring agent analyzes the values then produces description of the system's behavior.

The diagnostic agent takes the output from the monitoring agent and attempts to generate a qualitative causal explanation that will eventually by useful to the human operators. The diagnostic agent is usually in the waiting mode, but upon receiving information from the monitoring agent, the diagnostic agent starts its diagnosing process. When it reaches one or more diagnoses, the diagnostic agent sends the appropriate information to the knowledge broker.

Verification agent operates a faster then real – time numerical, model – driven simulator to measure the correlation of the diagnostic agent's output against the simulator's ideal values. When the verification agent receives any diagnosis, it verifies the diagnosis and sends out its judgment on the diagnosis. The judgment reaches the diagnostic agent through the IKB. If the verification agent rejects a diagnosis, the diagnostic agent will try to find the reason for rejection and attempt to diagnose in other ways.

The human – computer interface agent display the status to the operators and serves as the user interface.

Mobile agent is the most common intelligent agent. It is able to work with computer networks as large as Internet itself. The mobile agent may transport from computer to computer during execution and may carry accumulated knowledge and data with them.

2.2 Intelligent Knowledge Broker layer

In order to reach our goal we have to know more about exactly what kind of intelligence we are after. What is intelligent behavior in brokers?

At this point we are well into the realm of AI. (Which we understand as the research field occupied with the construction of artifacts with one or more intelligent features.) It is therefore necessary to inquire as to how terms like "intelligence" and "intelligent broker" may be used in this field. Also, the concept of *communication* is of some importance. So, let us look at a few alternatives for a definition.

One possible viewpoint is that any kind of program can be said to be an intelligent broker if and only if it has some sort of communicative competence as well as some sort of intelligence. Since most AI programs already can be said to communicate with their users in one way or other, there are two ways to go with this view: Either to define all of these programs as intelligent brokers, or to define the term of "communication" in this context as the exchange of information between agents and DKDB over a network, and accordingly reward the term only to network-communicating AI programs

Another distinct possibility is to view an intelligent broker as an entity operating on the *Knowledge Level*, which is a conceptual level different from, and higher than, the *Symbol Level*. The traditional way is to interpret a computer program as a symbolic machine, i.e. as an entity on the Symbol Level. But in accordance with the Knowledge Level view we may define the term "intelligent broker" as representing a higher-level entity instead, an entity of a kind that possesses knowledge, goals, constraints, and methods to follow in order to reach its goals. There is also a *principle of rationality* associated with this level, which dictates that an broker will always use its knowledge in a way that ensures the achievement of its goals - provided the agents and DKDB with the knowledge needed.

Keeping in mind these various difficulties with regards to a definition for this paper, we select as the best alternative this fairly simple, straightforward descriptive definition:

An *intelligent broker* is a computer program describable on the Knowledge Level which has a behavior that can reasonably be called intelligent, including the ability to communicate intelligently.

On the other hand, such a descriptive definition as the above is too general for our specific purposes. We are particularly interested in the communication of knowledge between agents and DKDB. We need a more specific definition to describe what we have in mind.

What does "communication of knowledge" really mean? Communication takes place by brokers sending data to agents and DKDB. But how does the data become information for an broker, and how does the information turn into knowledge?

In the context of an broker (as a computational system) in a decision-making process, the terms of data, information and knowledge may be defined like this:

Data are syntactic entities - patterns with no meaning; they are input to an interpretation process; ie. to the initial step of decision making.

Information is interpreted data -data with meaning; it is the output from data interpretation as well as the input to, and output from, the knowledge-based process of decision making.

Knowledge is learned information - information incorporated in an broker's reasoning resources, and made ready for active use within a decision making process; it is the output of a learning process.

In light of this we choose to understand the "communication of knowledge" as the process by which each agent formulates its knowledge into information and sends it as data to DKDB.

On this basis we choose to concentrate on the type of broker described by the following normative definition:

A *knowledge broker* is a type of intelligent broker that deals in knowledge, in the way of keeping, querying, distributing it or communicating it, whether as a primary or secondary function.

2.3 Dynamic Knowledge Database layer

DKDB layer stores knowledge used by all the agents and the knowledge broker. Whenever an agent needs access to the DKDB, the IKB uses distributed method calls to retrieve sharable knowledge from the DKDB.

Therefore, the real remaining question for us here is a practical one: Can future agents and IKB be relied upon to use knowledge bases that are *sufficiently* large for real applications, and can they combine their specific and general knowledge in a *sufficiently* fruitful way to be of any real benefit to human users?

Representing specific knowledge is not a problem either. Such representations have been in use for a long time, for instance in conventional database technology. And although there is a rather fundamental conceptual boundary between general and specific knowledge, they are not at all distinct. The one rely on the other, so to speak, and if one can represent knowledge about specific objects, one can just as well represent knowledge about general ones. (Consider the numbers in mathematics, for example.) There may still be insurmountable practical difficulties in representing all general (or specific) knowledge. But actually representing some of it is no problem at all.

The concept of agent-based software raises a number of important questions:
What is an appropriate agent communication language?
How do we build agents capable of communicating in this language? What communication architectures are conducive to cooperation?

Unfortunately, problems arise when it becomes necessary for programs that use one language to interoperate with programs that use a different language. Agent-based software engineering attacks these problems by mandating a universal communication language, one in which inconsistencies and arbitrary notational variations are eliminated. There are two popular approaches to the design of such a language – the procedural approach and the declarative approach.

The procedural approach is based on the idea that communication can be best modeled as the exchange of procedural directives. Scripting languages, such as TCL, Apple Event and Telescript, are based on this approach. They allow programs to transmit not only individual commands but entire programs, thus implementing delayed or persistent goals of various sorts. They are also directly and efficiently executable. But, there are disadvantages to purely procedural languages. For one, devising procedures sometimes requires information about the recipient that may not be available to the sender. Second, procedures are unidirectional.

The declarative approach is based on the idea that communication can be best modeled as the exchange of declarative statements (definition, assumption, among others). To be maximally useful, a declarative language must be sufficiently expressive to communicate widely varying sorts of information including procedures. At the same time, the language must be reasonably compact. It must ensure that communication is possible without excessive growth over specialized languages. As an exploration of this approach to communication, researchers in the ARPA Knowledge Sharing Effort have defined the components of an agent communication language (ACL) that satisfies these needs [1].

ACL can best be thought of as consisting of three parts-its vocabulary, and an inner language called KIF (Knowledge Interchange Format), and an outer language called KQML (Knowledge Query and Manipulation Language). An ACL massage is a KQML expression in which the “arguments” are terms or sentences in a KIF formed from words in the ACL vocabulary.

To support low-level communication, a communication API can be used. It can be viewed as a library of C, Java routines that allows a user to send KQML message string via TCP/IP, HTTP, IPX/SPX, NETBIOS protocol stacks.

In the near future we are going to use intercommunication approaches as the Object Management Group's Common Object Request Broker Architecture (OMG/CORBA), MS/DCOM that are often promulgated as solutions to the agent communication problems. The primary concern of these technologies is to ensure that applications can exchange data structures and invoke remote methods across disparate platforms [2],[3].

3.Conclusion

Although many questions remain to be solved, we believe the introduction of agent technology will be an important step towards achieving this destination.

REFERENCES

[1] http://www.csee.umbc.edu/agents/kqml/
[2] http://www.omg.org
[3] http://www.microsoft.com