A Cooperative Multi-Agent Approach in Support of Learning Object Recommendations

PHAEDRA MOHAMMED
Department of Mathematics and Computer Science
The University of the West Indies
St. Augustine, Trinidad and Tobago

Abstract: Autonomous agents have captivated researchers from many fields for a long time with the main research contributions originating from Artificial Intelligence, Human-Computer Interaction, and Concurrent Object-Based Systems. In general, agents have been perceived as being able to perform a variety of tasks on the internet, and within industrial and commercial applications as result of their situated, self-directed, goal-oriented behaviour. As a result, there is need for communication amongst agents since they periodically have to exchange knowledge and results in order to accomplish individual objectives. Furthermore, coordination of the activities and tasks being performed by agents has to be explicitly controlled. This paper describes the types of agents which are commonly deployed in systems and explains the importance of cooperation and interoperation in light of the tasks which these agents typically perform. The paper discusses the cooperative synchronization of the agents in multi-agent system called MARS which recommends online educational materials to users based on their learning interests.

Key-Words: - Agent, communication, coordination, decision-making, learning objects, recommender systems

1 Introduction

Intelligent software agents have captivated researchers from many fields for a long time with the main contributions originating from Artificial Intelligence, Human-Computer Interaction, and concurrent object-based systems [1]. Despite this level of activity, the formal definition of an agent is still not very clear however there is a general consensus regarding certain characteristics common to all agents namely situatedness, autonomy, and flexibility [1].

The situatedness of an agent means that the agent is able to perceive and interact with its environment by receiving stimuli and causing changes to the overall state of the environment. Autonomy signifies that an agent acts without the deliberate involvement of humans or other agents. The agent is able to control its internal state and actions and is essentially self-regulating and independent. Flexibility denotes the goal-oriented behaviour exhibited by an agent and its ability to interact with other agents in order to achieve certain goals. Huhns and Stevens [2] support these explanations (given by Jennings, Sycara, and Wooldridge [1]) with their interpretation of agents as having the ability to perceive, reason, act, and communicate.

In general, agents have been perceived as being able to perform a variety of tasks on the internet, and within industrial and commercial applications as result of their situated, self-directed, goal-oriented behaviour. One particularly common use of agents is for information management activities such as information gathering and filtering, and as personal assistants. Here, agents perform tasks such as retrieving information from the internet (search engines), filtering and sorting relevant information, and automating user tasks. Agents therefore offer ways of helping users achieve their goals, and they assist in decision making by controlling the complexity of information presented to users.

During the execution of many of these tasks, agents often operate in environments which are made up of more than one agent. These multi-agent environments facilitate distributed problem-solving via an agent network which may contain just a few interacting agents or hundreds of interacting agents. As a result, there is need for communication amongst agents since they periodically have to exchange knowledge and results in order to accomplish individual objectives. Furthermore, coordination of the activities and tasks being performed by agents has to be explicitly controlled and monitored in order to achieve overall system goals.

The remainder of this paper is organised as follows. Section 2 examines four general categories of software agent functionality which are commonly deployed in web-based systems. Section 3 discusses the importance of coordination, communication and cooperation among these types of agents. Section 4 describes the pragmatics of a multi-agent recommendation system called MARS which shows how these types of agents have been used to retrieve online educational materials using communication and coordination techniques. Section 5 describes the results achieved using MARS to recommend educational content to students and the paper concludes in Section 6.

2 Agent Functionality

Software agents are envisioned to perform tasks for users, collect, filter and process information, and interact with other agents. The agents that handle these operations are divided into four main categories: problem solving, user centric, control and translation agents [4]. Problem solving agents usually have expert information specific to a domain that is used to realize some goal and sometimes use algorithms or rule based approaches in achieving their functionality. These agents are commonly found in information filtering and gathering applications. Generally, most of the decision making for a domain takes place within problem solving agents.

User centric agents are typically concerned with user interaction and are portrayed using graphical user interfaces, animated personas, and interface tools. These agents focus on engaging users and sometimes gather information about the user's goals and preferences through observation of the user's activities and they normally facilitate the collection of input data from users. Control agents often need to interact with problem solving agents when seeking to fulfill the requests of users since they are not inherent problem solvers.

Control agents are the third type, and they coordinate other agents. These agents are regularly found in multi-agent systems and help the resident agents to function properly by directing their processes or providing assistance. Control agents also solve problems however they are domain independent agents since they are primarily concerned with facilitating the proper flow of information through a multi-agent system. These agents also direct the synchronization of the interdependent activities of the resident agents.

Translation agents are the last type and they allow heterogeneous systems to exchange and share information. These agents are especially useful on the web since they are used to bridge multi-agent systems which have incompatible knowledge formats by masking the conversion of information exchanged. Communication between different agents is fruitful only when both are able to understand each other; hence translation agents would facilitate the necessary knowledge conversions by acting as intermediaries.

The individual capabilities of problem solving, user centric, control and translation agents are not sufficient enough to perform the tasks of a software agent. Some degree of distributed processing is required and this in turn necessitates inter-agent communication, coordination, and cooperation.

3 Coordination and Communication

Coordination among agents may be of two types: cooperative and negotiation [2]. For the purposes of this paper, only cooperative coordination is considered. Whenever agents perform activities in a shared environment with the intention of working towards a shared goal there is a need to exploit the beneficial interactions between them [1]. This requires decomposing the problems which must be solved in order to achieve the goal into sub-problems. Agents that take part in cooperative coordination tackle these sub-problems individually and as groups. These agents can be problem solvers, user centric, or translators; they each synthesis and contribute answers towards to the end goal. Consequently their activities need to be syncronised by the control agents so that unnecessary conflicts are prevented and avoided [3].

Communication supports the interactions between the agents after the coordination has been worked out, and it is needed for agents to understand each other and be understood as well [2]. Agent Communication Languages (ACLs) have been used in the past to tackle the issues of agent communication by providing agents with a means of exchanging knowledge and information and these transactions are normally carried out using objects or sending messages. Hendler [4] elaborates on the nature of an ACL by explaining that ACLs have performative (speech act) levels and service levels. The performative levels of an ACL clarify the intention of messages, describe a desired state, and represent an agent's requirements for a particular task. The service levels allow agents to advertise

their capabilities and find agents that provide particular services.

4 Pedagogical Agents

Autonomous agents are called pedagogical agents when built for supporting learning processes. Pedagogical agents are still agents and so they must address many of the issues that agents deal with in online environments. Concerns such as having robust behaviour in unpredictable environments, coordinating their behaviours with other agents, performing arbitration between alternate actions, and responding to environmental stimuli need to be addressed by pedagogical agents as well [5].

Devedžić [6] explains that pedagogical agents help users who can be either learners or teachers. For both types of users, such agents are considered to be personal agents [7, 8]. In the case of learners, personal agents facilitate collaborative or individualized learning by monitoring and engaging the students in a learning process. Teachers are also assisted by personal agents but on the authoring side. These agents help with the identification, integration and assembly of learning material, and give teachers feedback on the performance of students in courses. Hence, personal agents are in fact a type of user centric agent as shown in Figure 1. Another type of pedagogical agent is the task agent [8] which aids personal agents in fulfilling some task. So, task agents typically coincide with the problem solvers, control and translation agents on the Internet; this is also shown in Figure 1.

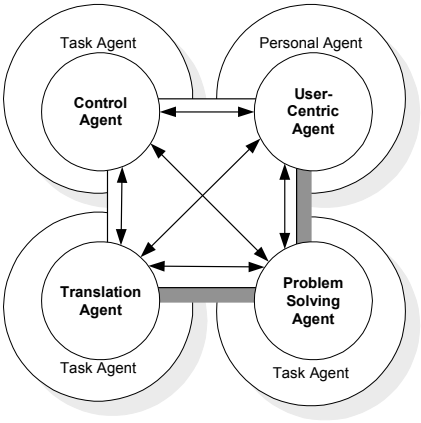

Fig.1 Correlation between general purpose agents and pedagogical agents

Educational research depends heavily on the recommendation of appropriate learning material to users in relation to the concepts that they are interested in or the goals that they need to accomplish. Based on the inherently distributed nature of learning materials on the Internet, a framework which takes advantage of the distributed computing in agency fits well into this context. Figure 1 illustrates how these agents are related as general purpose agents, and as pedagogical agents.

5 The MARS system

MARS [9] is a recommendation system which interacts with users, such as learners, seeking educational material on a particular topic or course creators searching for learning resources related to a specific academic discipline. Learning resources or learning objects* are recommended by MARS based on how well they match the topics/content requested by the user.

Several architectural elements make up MARS and these include the multi-agent system, recommendation rules, ontological mapping and merging functions, and metadata analyzers. These elements were implemented in MARS using Java-based tools. As a result, their integration into one complete system was seamless. Only the communication and coordination mechanisms of the multi-agent system of MARS will be discussed in this paper. However, the details of the additional components may be examined at length in [9].

5.1 MARS agents

The MARS system is made up of three agents called *LOMetadataHandler* (control agent), *SearchAgent* (user centric agent) and *ReasoningAgent* (problem solving agent). Agency was used in the design because it provides a means of user interaction and more importantly, it delivers the intelligent reasoning capabilities needed when making learning object recommendations. The three MARS agents were implemented using the JADE [10] which complies with the FIPA standard for intelligent multi-agent systems.

The *SearchAgent* monitors and interacts with the user and hence a GUI is its primary tool. The *SearchAgent* collects the user's request data and initiates the recommendation process by submitting this information to the MARS control agent. In addition, the *SearchAgent* presents the user with the learning object recommendations made displayed as a prioritized list in the GUI. *LOMetadataHandler* is in charge of overseeing the operations of the other MARS agents, and for providing them with assistance. By designing the *LOMetadataHandler* as

a *control agent*, the coordination of the information flow between the two other MARS agents is easily managed and the sequential processing of learning objects flows smoothly. *LOMetadataHandler* assists these agents by acting as a communication bridge between them which collects and forwards user requests, metadata details, and recommendation results to and from these MARS agents as appropriate. The *ReasoningAgent* produces the learning object recommendations for the users, and hence it is the source of the intelligence in the MARS system.This agent requires information from both the *LOMetadataHandler* and *SearchAgent.*

5.2 Communication

One challenge in building a multi-agent system is 'what to communicate and when to communicate' [1]. To address the concern of what to communicate, MARS was designed such that the three resident agents communicate specific information via customized Agent Communication Language (ACL) messages called *RecInfo* messages which conform to an internal message template defined by a content language. With respect to when to communicate, a cooperative approach [1] is adopted in MARS so that the agents act asynchronously by following a coherent plan for pursuing the goal of making recommendations

Figure 2 illustrates the source and destinations of the *RecInfo* messages used in MARS. These messages convey several pieces of information that are essential during the recommendation process and the contents are described in Table 1

Table 1 Contents of a RecInfo Message

Name of Variable	Description of Variable
keywords	Stores a collection of the keywords entered by the user as strings
intent	Declares the intention of a message
userMetadata	Stores the string representation of the URI of the ontology selected by a user on which to base the search
LO_Metadata	Stores the string representation of the URI of the domain metadata of a learning object
LO_location	Stores the string representation of the URI of a learning object as supplied by a learning object agent / repository
score	Stores the numeric assessment of how well a learning object's context matches the context of a user's request

A class called RecInfo (abbreviation for Recommendation Information) was created which allowed these pieces of information to be amalgamated into one Java object which was embedded in ACL messages. In fulfilling the JADE requirements for agent communication, the vocabulary of the RecInfo class was described in an ontology modeled by a RecInfoOntology class. This class is a translation vocabulary that is used by the Agent Communication Channel (ACC) which encodes and decodes the RecInfo objects when they are sent in the ACL messages. Hence, the information that is used in MARS for making a recommendation was aptly described by simply defining the template and vocabulary for the object to be filled into the message body.

In general, communication in JADE is carried out in three different ways depending on the source and destination of the Agent Communication Language (ACL) messages. These ACL messages are modeled in JADE as ACLMessage objects that conform to the FIPA specification for message sending in multi-agent systems. For inter-platform communication where the participating agents reside on the same container in the same platform, simple event signalling is used and there is no need for message translation since the ACLMessage object is sent as an Event object. When communication occurs between two agents that reside on the same platform but in different containers, Java Remote Method Invocation (RMI) is used so that the ACLMessage is translated and sent using RMI. For this research, the MARS agents reside on the same agent container so the ACLMessage objects were sent by simple event signalling.

5.3 Coordination

Each agent is designed to handle specific types of messages and messages with specific intentions. By deliberately fashioning the *RecInfo* message and allowing the agents to communicate solely in this way, the agents are able to reason about the state of their collective coordination. For example, when the *intent* of a *RecInfo* message is set to 'complete', the *SearchAgent* is able to understand that no more recommendation messages are to be expected and when the *intent* of a *RecInfo* message is set to 'reasoning', the *ReasoningAgent* appreciates that the information in that message is meant to be used for reasoning.

This setup addresses the challenge of agent coordination as described by Jennings, Sycara, and Wooldridge [1] where messaging/communication

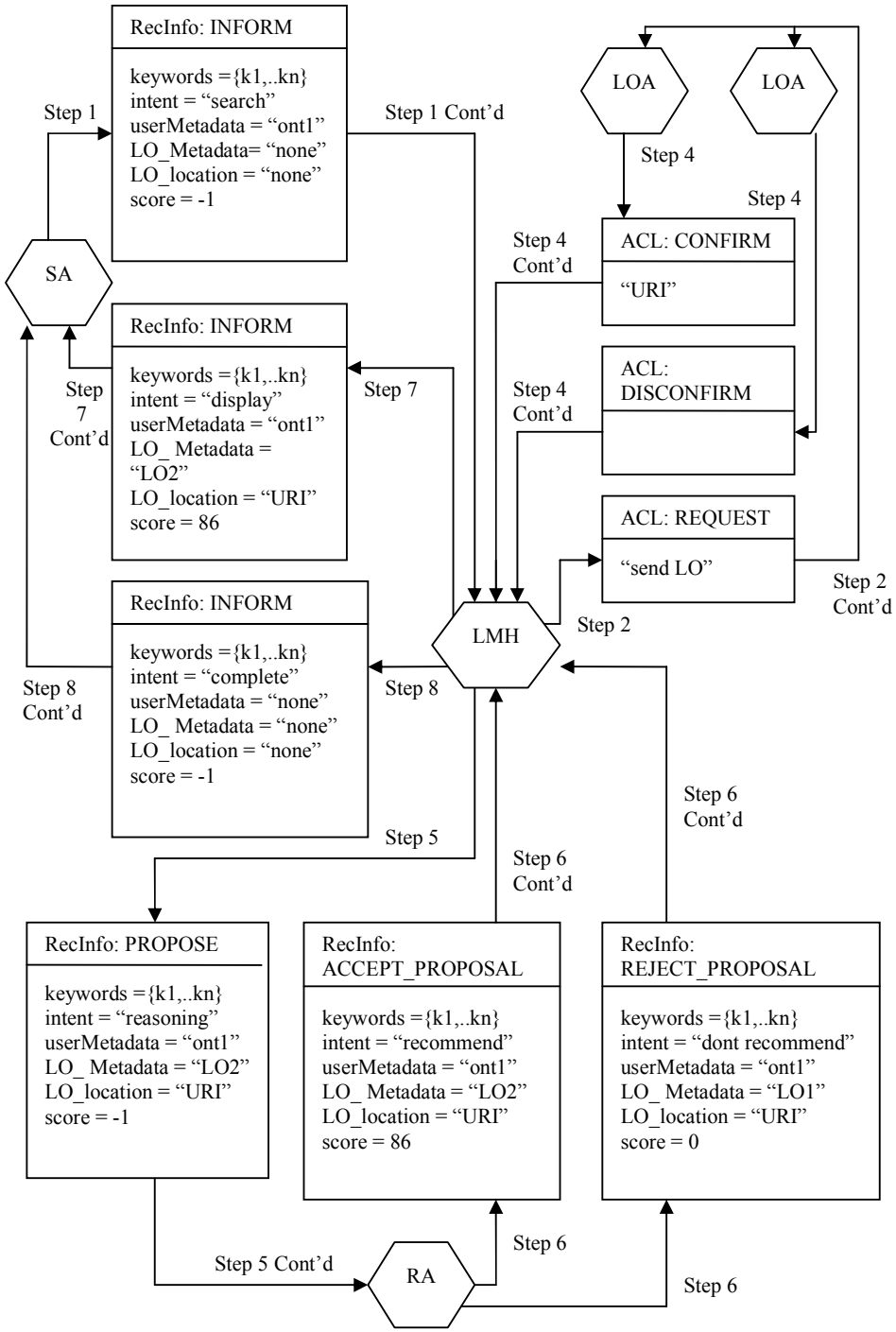

Fig. 2 Cooperative coordination in MARS

designs can address more than one concern in multi-agent systems. Furthermore, the use of the '*intent'* variable in a *RecInfo* message strengthens the decisions to be made by an agent since the use of performatives alone can be ambiguous as in the case of the *SearchAgent* where two types of INFORM messages can be received.

Figure 2 shows the sequence of communication and coordination events which take place in MARS during the recommendation process. The abbreviations for the agents *SearchAgent*, *LOMetadataHandler*, and *ReasoningAgent* are SA, LOM, and RA respectively and these are represented by hexagon symbols in the diagram. When the *SearchAgent* collects a user's request data and the search template that was selected, it formulates a *RecInfo* message with these search criteria and sets the *intent* of the message to 'search' as shown in Figure 2. This message is sent to the *LOMetadataHandler* which processes the message and performs a variety of tasks. The first task involves the brokering of candidate learning objects by polling a series of learning object agents (represented in Figure 2 as LOA) with simple ACL messages. These messages have a REQUEST performative and they state 'send LO' in the message body. Any agent that possesses one or more learning objects responds positively with an ACL message that has a CONFIRM performative and the URI of one learning object. Those that cannot satisfy the request respond with empty messages that have a DISCONFIRM performative.

Upon receiving an ACL message with a CONFIRM performative, the *LOMetadataHandler* extracts the location (URI) of the learning object's metadata, adds this URI and the search criteria received from the *SearchAgent* into a new *RecInfo* message with the *intent* set to 'reasoning', and sends the message to the *ReasoningAgent*. The *ReasoningAgent* processes this message and uses the information in the message's content in the weighting system which produces a positive score or a score of zero. A positive score implies that a recommendation should be made for that learning object and a *RecInfo* message with an ACCEPT_PROPOSAL performative is sent to the *LOMetadataHandler* with the score; a score of zero indicates an inappropriate learning object and a *RecInfo* message with a REJECT_PROPOSAL performative is sent accordingly.

The *LOMetadataHandler*, collects all of the messages that it receives from the *ReasoningAgent* and compares them to its list of candidate learning objects in order to ensure that all the recommendations, good and bad, were made and received. Only the positive recommendations are filtered out and sent to the *SearchAgent* in *RecInfo* messages containing the learning object's URI and its recommendation score. The *SearchAgent* collects and displays the recommendations as options to the user. The onus lies with the user at this point to select which one(s) he/she would like to explore. It is important to note here that the communication is not strictly sequential as shown in Figure 2 since messages are stored in the agents' message queues for processing.

6 Evaluation

The evaluation of the system's performance was carried out by generating random user requests and examining the learning material recommended and checking whether there was any information that was appropriate for the user's keywords in the learning material. In addition, the quantity of information related to the request was examined in cases where more than one recommendation was made for a request. The material recommended in many cases provided useful information which was related to the keywords. For example, in one scenario, Java programs were recommended which were specifically about Queues and Stacks in Computer Science to a user who requested material on the topics "Java" and "Integers". Although the recommended learning materials were not meant for a lesson on how to use integers in Java, they were still useful to the user since they illustrated how to declare, instantiate, and use integers in the Java language

By assigning scores, a user was given a prioritized list of learning objects starting with the ones that were the most appropriate for his/her request followed by those of lesser relevance. Rather than having to figure out where to start looking, the user could simply follow the ranking in the list. This was observed in a scenario where two learning objects were recommended with one being ranked significantly higher than the other. Upon examination of the material in both learning objects, the one with the higher score did in fact have more material specifically describing the keywords in relation to each other. The other one also had material linking the keywords but went into far less detail. So, the MARS agents provided helpful recommendations by ranking both sets of material for the user and assisting in the filtering process. The documentation of the test scenarios may be examined further in [9].

7 Conclusion

The research work done using the MARS system has shown that when systems are built with agents that perform problem solving, interact with users, and direct the syncronisation of agent activities, beneficial interactions are maximized which achieve the system's goals. A practical implementation of how coordination may be achieved among cooperative agents was presented together with a description of how messages which model the actions that an agent takes have been used to facilitate communication.

References:

[1] Jennings, N.R., K. Sycara, and M. Wooldridge, A Roadmap of Agent Research and Development, *Autonomous Agents and Multi-Agent Systems*, 110109, Volume I, pp.286-407.

[2] Huhns, M.N., and L.M. Stephens, Multiagent Systems and Societies of Agents, In *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, Gerhard Weiss, editor, MIT Press, Cambridge, MA, 1101010.

[3] Wooldridge, M., An Introduction to MultiAgent Systems, Wiley, 2002.

[4] Hendler, J., Making sense out of agents, *IEEE Intelligent Systems*, 1101010, Volume 15, Issue 2, pp. 42-48.

[5]Johnson L., E. Shaw, and R. Ganeshan, Pedagogical Agents on the Web, In Proc. Third International Conference on Autonomous Agents, ACM Press, 1101010, pp. 294-2100.

[6] V. Devedžić, Education and the Semantic Web, *International Journal of Artificial Intelligence in Education*, 2006, Volume 15, pp. 410-76.

[7] Vassileva, J., G. McCalla, and J. Greer, Multi-agent multi-user modeling, *User Modeling and User-Adapted Interaction,* Special Issue on User Modelling and Intelligent Agents, 2004, Volume 14, Issue 1, pp. 1810-210.

[8] Lin F., P. Holt, S. Leung, M. Hogeboom, and Y. Cao, A multi-agent and service-oriented architecture for developing integrated and intelligent web-based education systems, In Proc. International Workshop on Semantic Web Technologies for E-leanring, 40 August- 4 September, 2005, Maceios-Alagoas, Brazil, 2005, pp. 11-20.

[9] Mohammed, P., MARS: A rule based system for recommending learning objects in a given context, MPhil Thesis, Department of Mathematics and Computer Science, The University of the West Indies, St.Augustine, Trinidad and Tobago.

[10] Bellifemine, F., G. Caire, and D. Greenwood, Developing Multi-Agent Systems with JADE, Wiley, 2008

Footnotes:

*A learning object is a digital learning resource that facilitates a single learning objective and which may be reused in a different context. Web pages, audio files, images, Flash animations, PowerPoint presentations, Adobe Acrobat files etc, are all considered to be learning objects when intended for a specific pedagogical goal.