Computing and Informatics, Vol. 23, 2004, 157–177

ONTOLOGIES IN A MULTI-AGENT SYSTEM FOR AUTOMATED SCHEDULING

Evelio J. Gonzalez, Alberto F. Hamilton
Lorenzo Moreno, Roberto L. Marichal
Jonay Toledo

Grupo de Computadoras y Control
Dept. of Fundamental and Experimental Physics, Electronics and Systems
Av. Astrofisico Francisco Sanchez, S/N, CP 38207, University of La Laguna
Tenerife, Canary Islands, Spain
e-mail: ejgonzal@ull.es

Manuscript received 28 April 2004; revised 6 September 2004
Communicated by Ladislav Hluchý

Abstract. Multi-Agent Systems (MAS) have been successfully used in a wide range of applications such as robotics and e-commerce, and in particular in planning and scheduling. The aim of this paper is to present the interesting features that the use of ontologies in MAS offers. As an example, the development of a MAS for automated planning and scheduling in a University Research Group Scenario is shown in this paper. In this scenario, researchers are frequently proposed to attend internal meetings about several subjects such as lessons planning or research evaluations. Scheduling and negotiating meeting details such as time and location becomes highly complicated as the number of intended attendees increases. Moreover, there are usually conflicts about the use of some common resources such as portable computers or projectors.

As can be seen, the scheduling problem that the MAS solves is very easy. So having solved it is not what is important about this paper. In contrast, what is important is the potential which a scheduler can schedule for the items whose description, for example, is on the web, and can read on it (without knowing a priori) the logic of how the scheduling can be done.

Keywords: Multi-agent systems, ontologies, scheduling

1 INTRODUCTION

Multi-Agent Systems (MAS) [22, 32] have been shown as an effective tool in a wide range of applications such as robotics, e-commerce or even military transportation [24]. They are used in order to solve diverse real world problems that are inherently distributed in nature. Scheduling does not represent a new problem within the MAS field. Herewith the authors refer to the work of Kautz [23], Sandip Sen [29, 30] as well as the Electric Elves [2, 28], although the work presented in this paper has another objective.

In this context, a MAS for automated planning and scheduling in a University Research Group Scenario is presented in this paper. In particular, it is applied to the Grupo of Computadoras y Control (CyC). This is a research group that is developing its work at the University of La Laguna (Canary Islands, Spain). It consists of 2 full professors, 6 associate professors, 12 assistant professors, 3 students and 4 technicians. They are physically located in different work places, even in separate buildings, and they use computers with different operating systems. These members are involved in both academic and research activities. That is why many internal meetings are frequently proposed on a variety of subjects. Unfortunately, sometimes these meetings become highly difficult to organize, because it is nearly impossible to find a convenient time for all the intended attendees.

Another source of conflict is the availability of the common group resources such as several portable computers and a projector. If two or more users require one of these resources, it is not easy to decide which user has the higher priority. In this scenario, authors have decided to implement a MAS for automated planning and scheduling. This MAS should help group members to find the best possible time frames to perform a meeting and to designate the use of the portable computers and of the projector.

This MAS, called MASPlan, allows researchers to schedule meetings and make common resources available. The authors consider that this problem is very interesting due to the following:

- there is a large number of agents involved in the MAS architecture
- the tasks of scheduling meetings and negotiating resources are significantly different. The fact of each user giving his own strategy to his corresponding agent improves the system, and makes it different from a mere distribution problem
- agents from several users can be simultaneously involved in a negotiation. In the same way a specific agent can be engaged in several negotiations at the same time.

The scheduling problem that the MAS solves is very easy. It could even seem that employing MAS and ontologies is a bit like using a sledge-hammer on a thumb-tack. Having solved that scheduling problem is not what is really important about this

paper, although it proposes some new algorithms and variations over other well-known tactics. In that respect, MASPlan shows the ability to treat with algorithms based on different tactics such as votation, buying-selling or a more complex one.

An ontology is a set of classes, relations, functions, etc. that represents knowledge of a particular domain. Having *a shared ontology that all the agents utilize in their inter-agent communication is critical to successful communication because a shared ontology provides the common format in which express data and knowledge* [8]. There are several ontology languages such as KIF [18] or OKBC [3]. Moreover, the authors have preferred those languages known as "markup languages" [19]. These languages offer several advantages (source files are compact and portable, easy to learn and use, flexible). However, the last generation of these languages (RDF [6], DAML+OIL [27] and OWL [26]) gives computers an extra small degree of autonomy that can help them do more useful work for people.

> Systems may be able to provide all sorts of additional services and responses beyond the requirements of the standard but a certain basic set of conclusions will always be required [27].

In this context, what is important in this paper is showing the potential that a scheduler, through using ontologies, can schedule items whose description, for example, is on the web, and can read on it (without knowing a priori) the logic of how to do the scheduling. Ontologies and MAS are clearly on the wave of the future in this respect. In MASPlan, DAML+OIL, a semantic markup language for Web resources is used as the MAS ontology language. Although many OWL editors and converters from DAML+OIL have been developed, a fully OWL parser (that is, that deals with ontologies written in OWL Full, the most complete version of OWL) is missed. Therefore, the authors have preferred to develop the MAS presented in this paper using DAML+OIL tools. However, all the references about DAML+OIL in the rest of this paper can be translated to OWL in a easy way [25].

MASPlan proposes a unique formula for the use of a strongly standardized multi-agent architecture (FIPA) as well as an Ontology agent based on a highly expressive markup language.

The rest of this paper deals with a brief description of the state-of-the-art about the use of MAS in scheduling together with FIPA, the main standard used in the development of the MAS. After these descriptions, the authors describe the MASPlan Agent Framework together with the designed ontology (one of the most important aspects in this paper), algorithms that have been used with respect to meeting scheduling and resource negotiation and a system evaluation. Finally, conclusions and future work are reported.

2 STATE-OF-THE-ART

As indicated above, scheduling does not represent a new problem within the MAS field. In this section, the authors will describe the state-of-the-art briefly. In particular, the work of Kautz, Sandip Sen as well as the Electric Elves will be analyzed.

Kautz et al. [23] have developed an agent system that assists users in a range of daily, mundane activities. The key feature of the framework is the use of personalized agents, one for each individual user, called "userbots" that mediate communication between users and task-specific agents via the user preferred mode.

Sandip Sen [29, 30] deals with the problem of efficiently automating the process of scheduling meetings between employees in an organization. His approach is a distributed MAS, where each employee in the organization is provided with an automated (computational) meeting scheduling agent. This agent negotiates with the agents corresponding to the other users to schedule the meeting, protecting the privacy of its associated user while following other preferences of this individual. The meeting scheduling agent uses a calendar manager software to manipulate the user's calendar, and the email system to communicate messages with other meeting scheduling agents.

Finally, Electric Elves [2, 28] applies agent technology in service of the day-to-day activities of the Intelligent Systems Division of the University of Southern California Information Sciences Institute. The system is applied the problem of meeting planning with participants outside the organization where some of the necessary information about participants is not known in advance. Electric Elves reschedule meetings when someone is delayed, order food for meetings for someone working late or identify speakers for research meetings. Each person in the project is assigned his/her own personal proxy agent, which represents this person to the agent system. This agent keeps track of a project member's current location using several different information sources (calendar, GPS device, ...). When a proxy agent notices that someone is not attending a scheduled meeting or that they are too far away to make it to it in time, it springs into action.

The systems described above have included a reasoning mechanism in their development: identification trees, criterion functions... Nevertheless, although these mechanisms are functional, they are not able to offer the advantages that the use of ontologies does. In the University Research Group Scenario, for example, from the statements "Marta Sigut is an Assistant Professor" and "Assistant Professor is a subclass of User", a system conforming to an efficient ontology language should conclude that "Marta Sigut is a User". Therefore, if the system is asked for the list of users in the scenario, "Marta Sigut" should be included. However, this type of conclussions are difficult to obtain by other reasoning mechanisms.

3 FIPA

In this section the authors describe briefly the FIPA specifications, used in the development of MASPlan. These specifications have become a strong standard in MAS development and it involves not only agent language specifications but also agent management, conversations, etc. The use of this standard involves more robustness as MAS mentioned above are communicated via email or KQML [11].

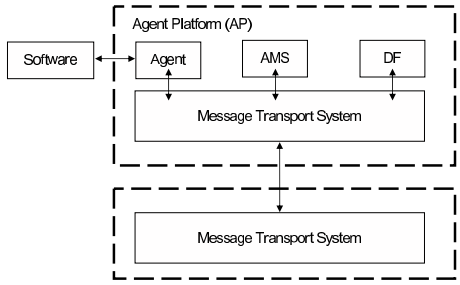


Fig. 1. FIPA Agent Management Reference Model

FIPA (Foundation for Intelligent Physical Agents) is an organization whose purpose is promoting the development of generic agent specifications [12]. Its agent management reference model (see Figure 1) provides the normative framework where FIPA agents exist and operate. The Directory Facilitator (DF) provides yellow pages services to agents that query it to find out services offered by other agents. On the other hand, the Agent Management System (AMS) offers white pages services and maintains a directory, which contains transport addresses for agents registered in the Agent Platform (AP). The Message Transport Service (MTS) is the default communication method between agents on different APs.

FIPA-OS [15], is an Open Source agent framework from research at Nortel Network's Harlow Laboratories that implements the FIPA specifications about agent interoperability. There are other Java-based implementations of FIPA specifications such as JADE [20] and ZEUS [34]. However, FIPA-OS has been chosen in this work due to

> its well-placed use of Java interfaces to separate agent subsystems, translation of incoming messages and system occurrences into events for internal processing, an isolated scheduling policy for task execution, use of a conversation object to enforce protocols and hold messages, and a task generation tool for constructing tasks from protocol definitions [17]

and because its use is more intuitive than JADE and ZEUS ones.

In this work, an Ontology Agent (OA) is included. An OA is an agent that provides access to one or more ontology servers. It also provides ontology services to an agent community, so the identification of a shared ontology for communication between two agents is facilitated. For example, agents can query the instances of the users that are included in the proposed scenario [13].

4 MASPLAN AGENT FRAMEWORK

The authors have treated the Group Scenario through the agent framework described below. This framework, whose scheme is shown in Figure 2, is composed of 6 different types of agents.

User Agent (UA): This agent is an end-user interface, that shows the schedule to its related user and allows it to ask for a meeting or a resource. When it occurs, this agent tries to locate its negotiator agent and communicates what user needs. Once the negotiator has finished its work, the user agent receives the result and shows it to the user. It also provides the user with a training mechanism in order to allow the Negotiating Agent to anticipate whether the user is inclined at some time to modify his agenda. Since the user-friendly interface (see Figure 3), the user can have a look at its own agenda, propose a meeting or request a resource.

Negotiator Agent (NA): The implementation of the meeting and resource negotiation algorithm is applied via this agent. When it is asked by its related user agent for a meeting negotiation, it looks in the DF for the negotiator agents of the rest of the intended attendees. Then, the negotiation process begins. Alternatively, in the case of a resource negotiation, it looks for the resource agent. Once the negotiation is over, it requests the Mail Agent to send an email to the user and it communicates the result of the negotiation to the user agent.

Ontology Agent (OA): This is one of the key considerations in this phase of the work which differs from the ones cited in Section 2. This agent supplies group ontology information to the rest of the agents when required. For example, it provides the complete list of users or the available instances of resources. The definition of an external ontology provides numerous general advantages: it permits the consultation with regard to concepts, the updating and use of ontologies and it eliminates the need to program the entire ontology in every agent, hence reducing required resources. In Section 5, some particular advantages are described.

Resource Agent (RA): This agent is invoked when a resource negotiation occurs. Firstly, it asks the Ontology Agent for the instances of the selected resource type. For example, there are several instances of the resource class Portable_Computer. Then it manages all the changes in the resource agenda (written in DAML+OIL too) such as a user abandon.

Mail Agent (MA): When an agenda change is confirmed, the Mail Agent is requested by the respective negotiator agent to send an email to the user via the mail software. A fact that was also considered was sending a SMS message to each user's mobile telephone, but this consideration was discarded in the system due to security reasons.

Rule Agent (RA): This agent provides the system with the ability of learning. It is consulted whenever there is any agenda change in order to organize a meeting. In this case, the NA will consult with this agent in order to determine whether or not the user is supposed to agree to the proposed agenda change. The answer will depend on the previously shown user behavior (whether through real cases or using the described training module) by employing identification trees (ID3) [33], due the high amount of involved variables.

For operative purposes, the user agenda is divided into half hour segments.

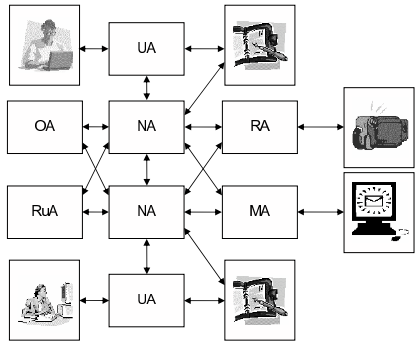


Fig. 2. MASPlan agent framework

5 ONTOLOGY DESIGN

For ontology design, there are many possible ontology languages. As stated above, the authors have used DAML+OIL as ontology language, because it provides a basic infrastructure that allows a machine to make some sorts of simple inferences that human beings do [27] and because this language has become a standard in ontology representations. At this moment a new, improved and more expressive markup language, known as OWL, is being developed. This language includes some useful definitions such as symmetric and inverse functional (MIRAR) functions. Nevertheless DAML+OIL is sufficiently expressive for carrying out this work.

The inclusion of DAML+OIL in this project has implied the use of an ontology development tool which can manage DAML+OIL ontologies and a DAML+OIL parser. In this work, the authors have chosen OilEd v.3.4 [1] as the development tool and Jena v.1.4.0 [21] as the DAML+OIL parser.

At first sight, classes, properties, instances and axioms described below could seem too detailed just to schedule meetings and share resources. Nevertheless, these terms can help audience not specialized in this area to discover the possibilities that ontologies offer when implementing algorithms more complex than those shown in this paper.

FIPA supplies a Personal Assistant Ontology (FIPA-PA), including terms such as “meeting-description”, “meet”, “schedule” and “participate” [14]. MASPlan ontology is based on that FIPA-PA, so the implemented ontology includes those functions.

Apart from these concepts, the authors have included other ones in the implemented ontology. Firstly, there is a class hierarchy for MAS users that can be seen in Figure 4. This hierarchy is completed with some DAML+OIL instances about each Group user and information about its email addresses (information that will be managed by the Mail Agent). For example, the following code represents that the user J. A. Mendez is an Associate Professor (technically, JA_Mendez is an instance of Associate_Professor class).

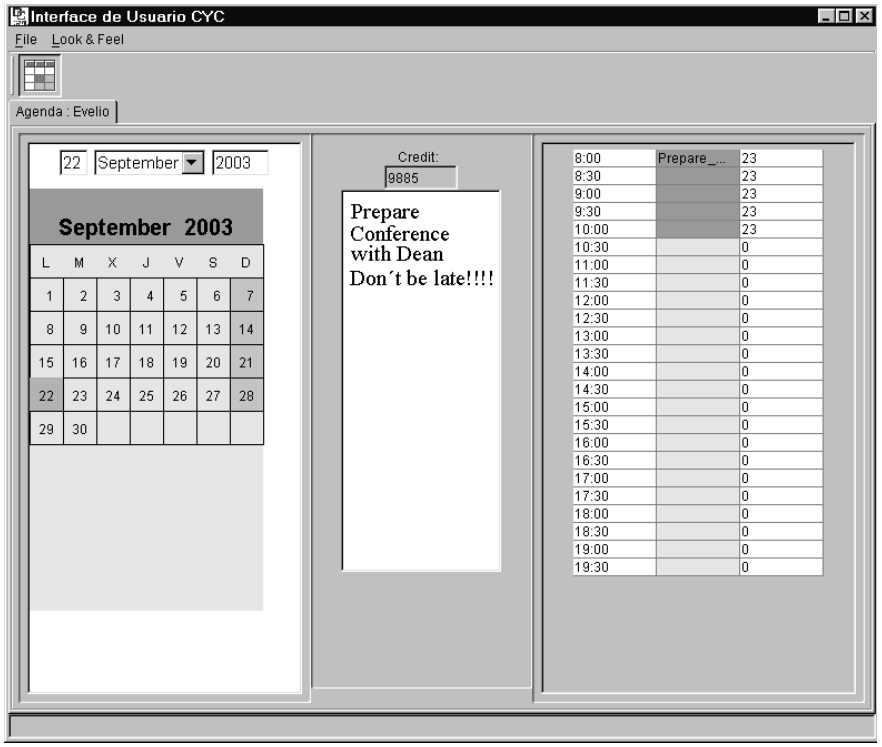


Fig. 3. User agent interface

```
<rdf:Description rdf:about=‘‘#JA_Mendez’’>
  <rdf:type>
    <daml:Class rdf:about=‘‘#Associate_Professor’’/>
  </rdf:type>
</rdf:Description>
```

These classes and instances allow negotiator and user agents to access the full list of available users and their categories. With this information, for example, an agent could negotiate a meeting with the restriction that a technician should assist to it.

In the same way, the authors have implemented a hierarchy for the group resources. In this sense, classes Projector and Portable_Computer have been included. This way, negotiators can easily locate the group resources.

A third block in the ontology consists of a list of activities which can be assigned to a time frame by a user, for example, conferences, preparing_lessons or meetings. The authors have included a class Priority_Activity which is the superclass of those

classes that represent a priority activity for the user such as conferences or attending lessons. The authors are currently working on including this information in future algorithms design and implementation.

Group CyC is responsible for three labs called A, B and C. Each lab has a person in charge. This allows for the systems to dispatch an email message when one of the labs is being used as a meeting place or a concrete user is preparing some procedures in it. As an example, the following DAML+OIL code indicates that J. L. Sanchez is in charge of Lab B.

```
<rdf:Description rdf:about=‘‘#Lab_B’’>
  <rdf:type>
    <daml:Class rdf:about=‘‘#Laboratory’’/>
  </rdf:type>
  <ns0:responsible rdf:resource=‘‘#JL_Sanchez’’/>
</rdf:Description>
```

The ontology also reflects the user agenda division into time frames. In this context, classes and properties such as Time_Frame, Hour, Related_Activity, Score or Involved_Users have been included.

Finally, the ontology contains concepts related to scheduling and negotiations such as Send_Email, Cancel_Meeting or Negotiate_Meeting.

The ontology consistency has been tested by FaCT [7], DAML+OIL Ontology Checker [4] and DAML Validator [5] reasoners.

As stated above, ontologies (specially those implemented in a markup language such as DAML+OIL) offer the potential that a scheduler, through using them, can schedule items whose description, for example, is on the web, and can read on it (without knowing a priori) the logic of how to do the scheduling. That is why general classes, properties and axioms have been included for this fact, for example, with regard to arithmetical operations. These definitions, as can be seen below, are used to describe the implemented algorithms. An equivalent system (although it does not take so much advantage of using ontologies) may consist of including the bytecodes (as the multiagent system is implemented in Java) of the algorithms and/or the involved agents directly (or through a link) in the ontology. So a NA could serialize/deserialize the desired objects when it is necessary.

6 MEETING SCHEDULING

Although this paper focuses on the use of MAS and ontologies, it will propose in this section and in the following one some new algorithms and variations over other well-known negotiation tactics. This way, MASPlan shows the ability to treat with algorithms based on different tactics such as votation, buying-selling or a more complex one.

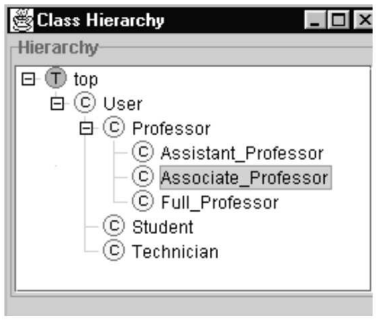


Fig. 4. MAS Users class hierarchy

When a user wants to negotiate a meeting in a time frame of index i and length L, its negotiation agent should calculate the best possible time frame. An example of negotiation technique is shown in this section. Concretely, the described algorithm minimizes a scoring function that weights some undesirable factors: distance to the desired period, "damage" to the other users and length difference. In this scenario, the scoring function is constructed as follows.

Firstly, the agent determines a distance factor for all the time frames in the selected day. This factor represents the distance (in time frames) from a time frame of index n to the desired time frames as it is shown in Equation (1), where $f = i + L - 1$.

$$D_n = \begin{cases} 0 & \text{if } i \leq n \leq f \\ n - f & \text{if } n > f \\ i - n & \text{if } n < i \end{cases} \tag{1}$$

As can be seen, this factor is equal to 0 (that is, no penalization) for those time frames desired by the initiator.

Once the agent has calculated this factor, it asks the negotiation agents of the desired meeting attendees for their score about the time frames of the initially desired meeting day. This way the initiator agent can assign a value J_n to each time frame of index n following Equation (2):

$$J_n = k\ D_n + \frac{1}{P} \sum_{i=1}^{N} P_{i,n} \tag{2}$$

where N is the number of intended attendees excluding the negotiation initiator, P is the score (in arbitrary units) that the initiator assigns to the meeting (for each time frame) and $P_{i,n}$ is the score given by the intended attendee i to the time frame of index n. On the other hand k is a weight for the distance factor. This factor is selected by each user using its UA. A recommended value for this factor is

$$k = \frac{N}{number\,of\,useful\,time\,frames\,in\,a\,day} \tag{3}$$

that weights the number of involved agents (like the second adder in Equation (2)) and tends to normalize the distance factor.

The second addend in Equation (2) introduces a voting mechanism which is reflecting the advantages of the Borda count [31] (Pareto-efficient, Condorcet loser, symmetry). Nevertheless, the preferred option is not the one which represents the higher point value rather the one which offers the least negative impact to the users involved.

Finally, the agent calculates the score function for a hypothetical meeting of length d and starting at the time of index n following the Equation 4, for $d = 1, 2, \ldots, L$. This score function adds a length difference factor to the summing of the affected time frames.

$$J_{n,d} = \sum_{i=n}^{n+d-1} J_i + \frac{L-d}{L} \tag{4}$$

The initiator negotiation agent selects the parameters that minimize this score function, and communicates this result to the other negotiation agents. If there is nothing wrong, all the negotiation agents will change their respective agenda files.

This negotiation process is shown in detail in Table 1 for a meeting negotiation example. The initiator wants to have a meeting with three users (called Nico, Leo and Jose) at 9:30 (time frame of index 4) and duration = 5 (2 h 30 min). In this example $k = 0.23$ (empirical). It is noted that none of the time frames are available for all the intended attendees, but the meeting is considered very important (the assigned score by the initiator is 100). After all the calculations described above, the initiator finds two configurations that minimize the score function ($J_{5,2}$ and $J_{6,1}$). When this occurs, the negotiation agent selects the longest one. Therefore, it proposes a meeting to be held at 10:00 and duration 2 (1 h).

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Nico	100	100	20	20	10	0	0	30	10	20	25
Leo	10	50	30	10	0	10	10	35	10	50	0
Jose	0	0	10	10	10	0	50	0	20	35	10
$k * D_n$	0.69	0.46	0.23	0	0	0	0	0	0.23	0.46	0.69
J_n	1.79	1.69	0.83	0.40	0.20	0.10	0.60	0.65	0.63	1.51	1.04
$J_{n,5}$	5.18	3.49	2.13	1.95	2.18	3.49	4.43	X	X	X	X
$J_{n,4}$	5.18	3.59	1.73	1.50	1.75	2.18	3.59	4.03	X	X	X
$J_{n,3}$	4.98	3.59	1.83	1.10	1.30	1.75	2.28	3.19	3.85	X	X
$J_{n,2}$	4.35	3.39	1.83	1.20	**0.90**	1.30	1.85	1.88	2.74	3.15	X
$J_{n,1}$	2.59	2.76	1.63	1.20	1.00	**0.90**	1.40	1.45	1.43	2.31	1.84

Table 1. Meeting negotiation process

With regard to the ontology, it is able to express this (or other) algorithm. This way, agents can negotiate among them without knowing their own strategy a priori. As an example, the following code represents the length difference factor included in Equation (4).

```
<ns0:lengthDifferenceFactor>
  <ns0:division>
    <ns0:dividend>
      <ns0:subtraction>
        <ns0:minuend>
          <ns0:desiredLength/>
        </ns0:minuend>
        <ns0:subtrahend>
          <ns0:length/>
        </ns0:subtrahend>
      </ns0:subtraction>
    </ns0:dividend>
    <ns0:divisor>
      <ns0:length/>
    </ns0:divisor>
  </ns0:division>
</ns0:lengthDifferenceFactor>
```

As can be seen, using ontologies, strategies can be easily modified without recompiling any code. Moreover, agents can be reused on-line following diverse negotiation techniques.

With regard to the concrete algorithm described in this section, a user could act as a system dictator via a dominating strategy of assigning an artificially inflated value to all the agenda time frames. This fact would not be acceptable from a common benefit point of view. This possibility could be mitigated by assigning each user a maximum monthly point credit in order that the sum total of points utilized would not exceed this credit. This maximum credit could be increased for exceptional circumstances (conferences...). Figure 5 shows the simplified message flow between the involved agents.

The structure of the NA permits the easy implementation of other negotiating techniques. For example, giving the negotiation up if the value of the score function is greater than a reference value, or negotiating the meeting for another date. This is an important advantage for using MAS and ontologies: a new negotiating technique can be implemented without recompiling any code, for example reading it on the web. Each NA could even follow its own technique. This way, users could not cheat by manipulating their request to get the outcome they want as they do not know how the system will work.

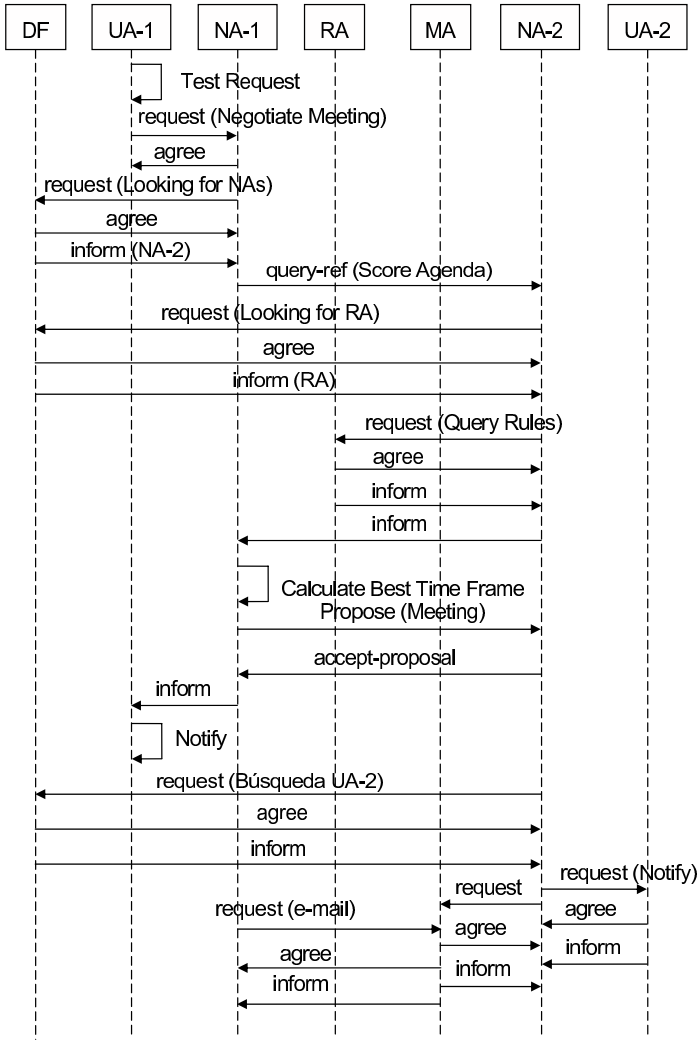


Fig. 5. Simplified message flow in MASPlan meeting negotiation

This algorithm differs from other systems [30, 29] as a proposed meeting always takes place. This way, social welfare is supposed to be increased as meetings are held to benefit the Group.

7 RESOURCE NEGOTIATION

As was indicated above, another source of conflict is the management of common group resources. When two or more users require one of these resources, it is not easy

to decide which user has the higher priority. For this negotiation, the proposed MAS implements time and behavior dependent tactics [10, 9]. The proposed method could seem rather artificial, but it illustrates the possibilities of using MAS and ontologies in scheduling and planning. The use of the ontologies allows to define other more realistic negotiation methods and/or formulas than the described below.

To illustrate this resource negotiation, consider the following example. A user B desires to use the projector within 5 days, but when its negotiation agent tries to reserve the projector, it realizes that the resource is reserved at that moment by a user S. Then B negotiation agent starts a resource negotiation, trying to "buy" the projector from "seller" S. Both agents initially determine their respective acceptable region for this negotiation $[\min_j^i, \max_j^i]$ $(j = B, S)$. In the case of "buyer" B its region limits are based on the proposed offer by its user and the remaining monthly credit. In contrast, in the case of the "seller" the main factor is the price previously "paid" when it reserved the resource. As S is not expected to be very interested in "selling" the projector (he also needs the projector, although he understands that B's projector use could have higher priority in the group) his region limits should represent a great value (for example, if S had paid a score of 100, an acceptable region could be $[\min_S = 125, \max_S = 175]$, which guarantees at least a 25% benefit).

Once the agents determine their respective regions, they can negotiate directly, informing their respective offers to the other negotiation agent. As is stated above, these values are calculated following time and behavior tactics.

In time dependent tactics, the predominant factors used are time t and the time limit for the negotiation $t^j_{\max}$ $(j = B, S)$. MASPlan implements this tactic, choosing an exponential function. So, the time dependent offer OT communicated by negotiation agent B at step t is:

$$OT_B(t) = \min_B + (\max_B - \min_B)\left(1 - e^{-\frac{t}{t^B_{\max}}}\right) \tag{5}$$

In the case of the seller, it is assumed that the convergence on a mutually acceptable solution was slower, since the agent is not very keen on selling the resource. So in the seller offer time dependent function OT_S it is necessary to include an egoism factor E_S that represents the seller resistance to agree to the sale. Empirically an $E_S = 3$ reproduces desired agent behaviors.

$$OT_S(t) = \min_S + (\max_S - \min_S)e^{-\frac{t}{E_S t^B_{\max}}} \tag{6}$$

In the proposed example, $t^B_{\max} = 5$ days. Obviously, it is not a good idea that the agents negotiate for 5 days. So MASPlan considers $t^B_{\max} = 5$ steps, that is, $t = t + 1$, after that the buyer makes a new offer. In this way the user would be penalized if he does not order the equipment and services for example on a timely basis.

On the other hand, the behavior dependent tactics compute the next offer based on the previous attitude of the other negotiating agent. In this sense, MASPlan has implemented a variation of the Random Absolute Tit-For-Tat function.

$$OB_B(t+1) = OB_B(t) + \min\left(R(1)\left(O_s(t-1) - O_s(t)\right), K_B\, O_B(t)\right) \tag{7}$$

where $R(1)$ is a function that generates a random value in the interval $[0, 1]$ and $K_B O_B(t)$ represents the maximum variation that the OB_B can reach. This way, a value $K_B = 0.2$ represents a maximum possible variation of 20 %. This tactic is applicable when $t > 2$.

Finally, the offer is a weighted-random summing of both functions.

$$O_j(t) = w_{T,j}(t) OT_j(t) + \left(1 - w_{T,j}(t)\right) OB_j(t) \quad j = B, S \tag{8}$$

where

$$w_{T,j}(t) = w_{\min,j} + \left(1 - w_{\min,j}\right) R(1) \tag{9}$$

and $w_{\min,j}$ is the minimum value of the time dependent weight. The authors have taken $w_{\min,j} = 0.8$. This way, time dependent factor is rated at least four times as important as the behavior dependent one.

Two negotiation examples can be seen in Figure 6. In the first one, both negotiator agents reach a mutually acceptable solution (the buyer's offer is greater than the last seller's one). In contrast, the second negotiation fails, because the buyer does not reach the seller's offer before negotiation time is over. Initially this negotiation time is $t^B_{\max}$, but the authors have included an experimental corrector factor in the MAS implementation. So, negotiation occurs whilst $t < 1.25 * t^B_{\max}$. In the event that the negotiation is successful, a termination/acquisition protocol will be established via the RMA. The seller would receive a point compensation in his monthly credit value.

One of the main fields to be treated in future works is facilitating the user definition of agent profiles. This way, a user could select for example the agent's egoism factor or his time or behavior related weights.

As stated above, the algorithm can be implemented using the designed ontology.

8 SYSTEM EVALUATION

The MASPlan system has been evaluated within the CyC group. This evaluation was based on two principal aspects: its reliability as well as its success in arriving at the optimum result. With regard to the first aspect, reliability, the multi-agent system has been challenged by handling efficiently a high flow of intricate requests and demands. We would herewith point out that the system was tested via the involvement of 27 UAs, each initiating an independent negotiating protocol. This represented an exceptionally high processing load for the CyC server (at the same time as offering a range of other services to the group). In a 10-minute time span it successfully and satisfactorily concluded the respective negotiations. With regard to the second aspect, the system was severely tested by a variety of diverse situations based on user need priority. Additionally, as a consequence, feedback was received

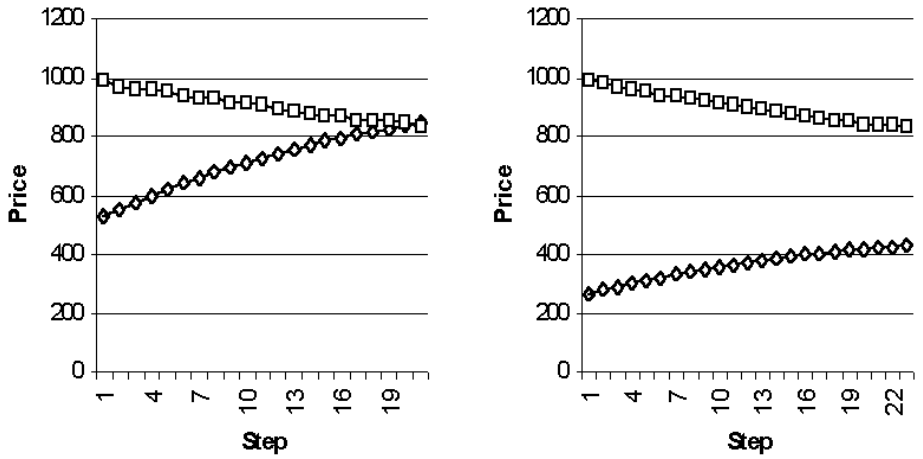


Fig. 6. Resource negotiation examples

from the users relating to elements of the problem solving process which were not contemplated in the initial system design.

A problem in the system evaluation is giving clear criteria of quality of its use. Solutions given by the described algorithms are intuitively in conformity with the desired behavior, but it is necessary to answer the question of how is this system, apart from the described above software improvements, better than the traditional method of "phoning" and "emailing". In this context, number of scheduling operations has been taken as evaluation criterion. A *scheduling operation* is defined as an individual act oriented to scheduling a meeting or reserving a resource. A study of MASPlan evolution is shown in Figure 7. The number of scheduling operations in the Group has been registered for 6 months, using the "phoning-emailing" method in the first 3 months and using MASPlan in the latter ones. As can be seen, MASPlan has approximately involved a 50 per cent reduction which gives an idea about the good part about using the system. Moreover, as the Rule Agent learns more about the desires of the users, corrections (and therefore the number of scheduling operations) tends to decrease.

With regard to the reception of the system, there was a positive reaction to the system by all the users, although a few of them have preferred the more traditional procedure for the organization of meetings and supplemental equipment requests.

9 CONCLUSIONS AND FUTURE WORK

In this paper, the authors have presented a multi-agent system for planning and scheduling in a University Research Group Scenario. In particular, it has been applied to the Grupo Computadoras y Control (CyC) of the University of La Laguna

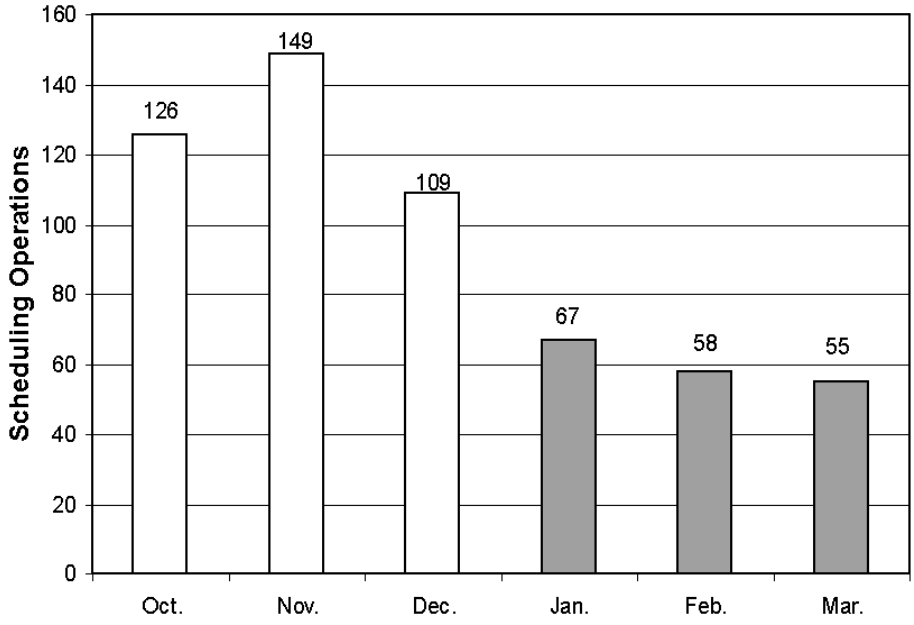


Fig. 7. Evolution of scheduling operations

(Tenerife, Canary Islands, Spain). This system called MASPlan attempts to automate meeting negotiation in an intelligent way amongst several researchers and to manage group resources (mainly portable computers and projectors).

For this purpose, it develops FIPA specifications for agent management and communications. The system is implemented using FIPA-OS tool, an Open Source agent framework from research at Nortel Network's Harlow Laboratories. It has been chosen in this work for its well-placed use of Java interfaces to separate agent subsystems, translation of incoming messages and system occurrences into events for internal processing, an isolated scheduling policy for task execution, and because its use is more intuitive than the given by other tools (JADE and ZEUS), although FIPA-OS does not provide any finite state machine support.

As an ontology agent has been included in MAS Agent Framework, an ontology language has to be chosen. There are many possible ontology languages available such as OIL and RDFS. However, the authors have shown a preference for DAML+OIL because it provides a basic infrastructure that allows a machine to make some sorts of simple inferences that human beings do and because this language has become a standard in ontology representations. What is important in this paper is showing the potential that a scheduler, through using ontologies, can schedule items whose description, for example, is on the web, and can read on it (without knowing a priori) the logic of how to do the scheduling. Ontologies and MAS are clearly on the wave of the future in this respect. In MASPlan,

DAML+OIL, a semantic markup language for Web resources is used as the MAS ontology language.

Once the ontology has been designed, the authors have implemented, as an example of the possibilities that MAS and ontologies offer, an algorithm for meeting negotiation and time and behavior dependent tactics with some modifications such as the inclusion of an egoism factor, for resource negotiation.

One of the main fields to be treated in future works is facilitating the user definition of agent's profiles. This way, users could choose for example the egoism factor or time or behavior dependent weights. The authors are currently working on including some features that are based on the ontology design, for example, algorithms that contain information about user and activity hierarchies.

The authors strongly recommend the use of MAS and ontologies in order to develop applications such as those analyzed in this paper.

REFERENCES

[1] Bechhofer, S.—Horrocks, I.—Goble, C.—Stevens, R.: OilEd. A Reasonable Ontology Editor for the Semantic Web. Lecture Notes in Computer Science, Vol. 2174, 2001, pp. 376–385.

[2] Chalupsky, H.—Gil, Y.—Knoblock, C. A.—Lerman, K.—Oh, J.—Pynadath, D. V.—Russ, T. A.—Tambe, M.: Electric Elves. Applying Agent Technology to Support Human Organizations. Proceedings of the Thirteenth Innovative Applications of Artificial Intelligence Conference (IAAI-01), pp. 51–58, AAAI Press, 2001.

[3] Chaudhri, V. K.—Farquhar, A.—Fikes, R.—Karp, P. D.—Rice, J. P.: Open Knowledge Base Connectivity 2.0. Knowledge Systems Laboratory, January, 1998.

[4] DAML+OIL Ontology Checker web site. Avalaible on: `http://potato.cs.man.ac.uk/oil/checker`.

[5] DAML Validator web site. Avalaible on: `http://daml.org/validator`.

[6] Decker, S.—Mitra, P.—Melnik, S.: Framework for the Semantic Web: An RDF Tutorial. IEEE Internet Computing, Vol. 4, 2000, No. 6, pp. 68–73.

[7] FaCT web site. Avalaible on: `http://www.cs.man.ac.uk/~horrocks/FaCT/`.

[8] Falasconi, S.—Lanzola, G.—Stefanelli, M.: Using Ontologies in Multi-Agent Systems. Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop, 1996.

[9] Faratin, P.—Jennings, N. R.—Buckle, P.—Sierra, C.: Automated Negotiation for Provisioning Virtual Private Networks using FIPA-Compliant Agents. Proc. 5th Int. Conf. on the Practical Application of Intelligent Agents and Multi- Agent Systems (PAAM-2000), Manchester, UK, pp. 185–202, 2000.

[10] Faratin, P.—Sierra, C.—Jennings, N. R.: Negotiation Decision Functions for Autonomous Agents. Int. Journal of Robotics and Autonomous Systems, Vol. 24, 1998, Nos. 3–4, pp. 159–182.

[11] Finin, T.—Labrou, Y.: Tutorial on Agent Communication Languages. First International Sympopsium on Agent Systems and Applications and the Third International Symposium on Mobile Agents, ASA/MA'99, October, 1999.

[12] FIPA Abstract Architecture Specification. Technical Report, SC00001L. Foundation for Intelligent Physical Agents. December, 2002.

[13] FIPA Ontology Service Specification. Technical Report, XC00086D. Foundation for Intelligent Physical Agents. August, 2001.

[14] FIPA Personal Assistant Specification, Technical Report, XC00083B. Foundation for Intelligent Physical Agents. August, 2001.

[15] FIPA-OS web site. Avalaible on: `http://www.emorphia.com/research/about.htm`.

[16] FIPA web site. Avalaible on: `http://www.fipa.org`.

[17] Fonseca, S. P.—Griss, M. L.—Letsinger, R.: Agent Behavior Architectures. A MAS Framework Comparison. Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems, Bologna, Italy, 2002.

[18] Genesereth, M. R.—Fikes, R. E.: Knowledge Interchange Format, Version 3.0 Reference Manual. Technical Report Logic-92-1. Stanford, CA, USA. 1992.

[19] Guidelines for Electronic Text Encoding and Interchange (TEI P3). Edited by C. M. Sperberg-McQueen and Lou Burnard, Text Encoding Initiative. Chicago, Oxford 1990.

[20] JADE web site. Avalaible on: `http://jade.cselt.it`.

[21] Jena 2 Manual. HP Labs Semantic Web Research, August 2003.

[22] Jennings, N. R.—Sycara, K.—Wooldridge, M.: A Roadmap of Agent Research and Development. Autonomous Agents and Multi-Agent Systems Journal, Vol. 1, 1998, No. 1, pp. 7–38.

[23] Kautz, H. A.—Selman, B.—Coen, M.—Ketchpel, S.: An Experiment in the Design of Software Agents, Software Agents — Papers from the 1994 Spring Symposium (Technical Report {SS}-94-03), AAAI Press, 1994.

[24] Montana, D.—Herrero, J.—Vidaver, G.—Bidwell, G.: A Multiagent Society for Military Transportation Scheduling, Journal of Scheduling, Vol. 3, 2000, No. 4.

[25] OWL Converter web site. Avalaible on: `http://mindswap.org/2002/owl.html`.

[26] OWL Web Ontology Language Overview. Technical Report. World Wide Web Consortium, March 2003.

[27] Pearse, A.: Why use DAML? Whitepaper published on-line on 10 April 2002. URL: `http://www.daml.org/2002/04/why.html`.

[28] Pynadath, D.—Tambe, M.—Arens, Y.—Chalupsky, H.: Electric Elves. Immersing an Agent Organization in a Human Organization. Proceedings of the AAAI Fall Symposium on Socially Intelligent Agents — the human in the loop, 2000.

[29] Sen, S.: Developing an automated distributed meeting scheduler, IEEE Expert, Vol. 12, 1997, No. 4, pp. 41–45.

[30] Sen, S.—Haynes, T.—Arora, N.: Satisfying User Preferences While Negotiating Meetings. International Journal of Human-Computer Studies, Vol. 47, pp. 407–427.

[31] Straffin, P. D.: Topics in the Theory of Voting. Birkhauser, 1990.

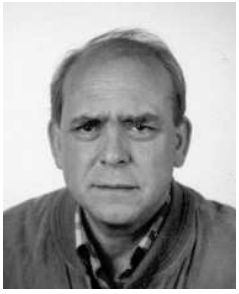
[32] Weiss, G.: Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. Cambridge, MA: MIT Press, 1999.

[33] Winston, P. H.: Artificial Intelligence. Addison-Wesley, 1994.

[34] ZEUS web site. Avalaible on: `http://193.113.209.147/projects/agents/zeus/index.htm`.

Evelio Gonzalez received the MS degree in applied physics in 1998 and his PhD degree in computer science in 2004 from the University of La Laguna, Tenerife, Spain. From 1998 to 2001, he was a research student in the Department of Applied Physics, Electronics and Systems at the same university. Currently, he works as assistant professor in the University of La Laguna. His areas of interest include simulation, digital control, computer architecture, artificial intelligence and intelligent agents.

Alberto F. Hamilton received his MS degree in physics in 1991 from the University Complutense of Madrid and his PhD degree in computer science in 1995 from the University of La Laguna. He is associate professor in the Applied Physics, Electronic and Systems Department at the University of La Laguna, Canary Islands, Spain from 1997. His current research interests are system control, robotics and intelligent agents.



Lorenzo Moreno received the M.S. and Ph.D. degrees from the Universidad Complutense de Madrid, Spain, in 1973 and 1977, respectively. From 1977 to 1979, he was an associate professor in the Department of Computer and Control Engineering, University de Bilbao, Spain. From 1979 to 1988, he was an associate professor in the Department of Computer Science, Universidad Autónoma de Barcelona, Spain. He is presently full professor in the Applied Physics, Electronics and Systems Department at the University de La Laguna, Tenerife, Spain. His areas of interest include computer architecture and control.

Roberto Marichal received the MS degree in applied physics in 1996 and his PhD degree in Computer Science in 2003 from the University of La Laguna, Tenerife, Spain. From 1996 to 2000, he was a research student in the Department of Applied Physics, Electronics and Systems at the same university. Currently, he works as assistant professor in the University de La Laguna. His areas of interest include neural networks and nonlinear systems.

Jonay Toledo received the MS degree in computer science in 2001 and in electronic engineering in 2002 from the University of La Laguna from the University of La Laguna, Tenerife, Spain. Currently, he is a research student in the Department of Applied Physics, Electronics and Systems at the same university. His areas of interest include neural networks and nonlinear systems.