Multi-agent information system for the assistance to urban mobility

Hayfa Zgaya, Slim Hammadi
Department of automatic and industrial computer science
LAGIS UMR 8146
Ecole Centrale de Lille
Cité scientifique-BP 48 59651 Villeneuve D'Ascq CEDEX
FRANCE

Abstract: - According to the exponential growth of services available on large distributed networks, transport customers require relevant, interactive and instantaneous information during their travels. Thus, we designed and implemented a multi-agent information system using a special kind of software agents: the Mobile Agents. This work belongs to the national French project VIATIC.MOBILITE from the industrial cluster I-trans[*], which is an initiative bringing together major French players in rail technology and innovative transport systems. The proposed system uses a two-level optimization approach based on metaheuristics. However, some network errors can occur during the mobile agents moving through the distant network nodes (bottleneck, failure, crash…). In this paper, we define a mobile agent negotiation process to reassign required services to available network nodes according to their current states in their correspondent final routes called Workplans. Consequently, we designed and implemented a flexible transport ontology which allows an easy handling of the terms and messages for negotiating.

Key-Words: - Multi-agent system, Mobile agents, Negotiation protocol, Ontology.

1 Introduction

According to the exponential growth of services available on large distributed networks, transport customers require relevant, interactive and instantaneous information during their travels. Unfortunately, distributed applications through wide networks are not easy to realize because of the limited aspect of bandwidth which remains restricted and also because of a high incidence of network errors (bottleneck, failure, crash…). Our goal is to properly access and share distributed data located through an Extended Transport Multimodal Network (ETMN). In this context, mobile technology[1, 2] can complement artificial intelligence because it can reduce considerably network traffic [3]. Giving the mobility character to a software agent will allow him to migrate towards any node on the network which can receive mobile entities. Nodes to be visited by a Mobile Agent (MA) correspond to his route called Workplan. Many researchers have long discussed the benefits of the MA paradigm and conclude that it might be efficient in some cases [4, 5]. In a recent work [6], we demonstrated that using the MA paradigm in a Transport Multimodal Information System (TMIS) in order to collect needed data, is widely beneficial than using classical paradigms such as the Client Server (CS) one, if we use an optimization approach. The verification was successful thanks to a two-level optimization approach [7, 8] using metaheuristics. However, some network errors can occur during the MAs moving through network nodes (bottleneck, failure, crash…). In this paper, we define a MA negotiation process to reassign required services to available network nodes, so we designed and implemented a flexible transport ontology which allows an easy handling of the terms and messages for negotiating. The remainder of this paper is organized as follows: the problem complexity and the correspondent general formulation are presented in the next section. The global architecture of the multi-agent system is proposed in section 3. The proposed negotiation protocol is specified in section 4, followed by the flexible transport ontology in section 5. Simulation results are given in section 6, implementing the negotiation protocol in case of some network errors scenarios. Conclusion and possible future works are addressed in last section.

2 Problem Formulation

The main concern of a TMIS is to satisfy users, respecting delays of responses (due dates) and minimizing their costs; this is a two-step optimization problem: firstly the assignment of an effective number of MAs to all existent network nodes. This assignment builds initial Workplans of MAs in order to explore, in an optimal manner, the ETMN entirely. The second step corresponds to the best assignment of a sub-set of the

[*] http://www.i-trans.org/

ETMN nodes to identified tasks, deducing final Workplans of MAs. The selected sub-set of nodes corresponds to the potential services providers to the identified tasks and a single identified task corresponds to an independent recognized sub-request which belongs to one or several requests formulated simultaneously by one or different customers. A single task can correspond to transport services (sub-route, well-known geographical zone…) or to related services (cultural event, weather forecast…). After the decomposition process, information providers (distant nodes), which propose services to the correspondent identified tasks, are recognized (fig.1). Finally, nodes must be assigned to tasks in order to satisfy all connected users. A user is satisfied if his request was answered rapidly with a reasonable cost.

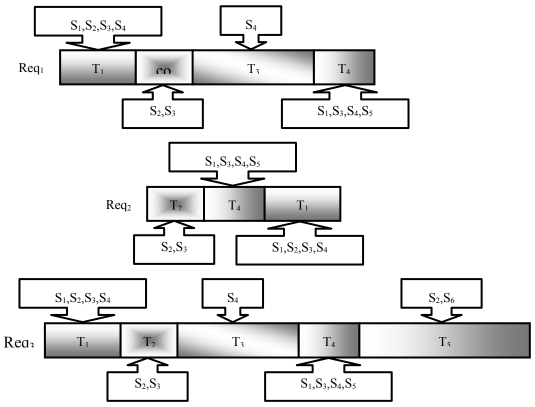

Fig.1. Nodes identification

The problem described previously is defined by:

- R requests, waiting for responses at the same instant t. The set of these requests is noted R_t,
- The set of independent I tasks, representing all available services on the ETMN, is noted $T=\{T_1,\ldots, T_I\}$,
- Each request $req_w \in R_t$ $(1 \leq w \leq R)$ is decomposed into a set of independent tasks, noted $I_{t,w}=\{T_{r_1},\ldots,T_{r_{n_j}}\}$ $(n_j \leq I$ and $I_{t,w} \subseteq T)$,
- The set of independent I' tasks $(I' \leq I)$, composing globally R_t, is noted I'_t $(I'_t \subseteq T$ and $\bigcup_{w=1}^{R} I_{t,w} = I'_t)$,
- Each request req_w has a due date d_w initially known, an ending date D_w and a total cost C_w,
- The realization of each task $T_i \in T$ requires a resource, or node, selected from a set of J registered nodes in the ETMN, noted $S=\{S_1,\ldots, S_J\}$,
- The set of J' nodes $(J' \leq J)$, selected from S to perform I'_t, is noted S' $(S' \subseteq S)$,
- There is a predefined set of processing time; for a given node S_j and a given task T_i, the processing time of Ti using the resources of S_j, is defined and noted $P_{i,j}$,
- There is a predefined set of information cost; for a given node S_j and a given task T_i, the cost of the information to collect from S_j, corresponding to the service referenced by Ti, is defined and noted $Co_{i,j}$,
- The size of collected data to ensure a service is defined; for a given node S_j and a given task T_i, the data size is defined and noted $Q_{i,j}$,
- We have partial flexibility; the realisation of each task Ti requires a node selected from a set of nodes which propose the same service performing the task Ti, with different cost, processing time and data size.

The three characteristics described above, namely $(P_{i,j};Co_{i,j};Q_{i,j})$, represent successively the first, second and last term of each element of what we call a *service table*. In order to situate the complexity of our problem, an analogy was performed between the problem described above and the Flexible Job Shop Problem (FJSP): In our problem, we manage the similarities of requests in order to avoid the same data research. Besides, we have to assign the servers to tasks as well as to assign MAs to remote nodes (servers), taking into account the network state. Therefore our problem presents more difficulty than the FJSP which has been shown to be NP-hard. In addition, the distributed character of our system and the requirement to cooperate different autonomous static and mobile entities confirm our choice of a multi-agent architecture for our system.

3 The multi-agent system

To resolve the problem described previously, we propose a system based on the coordination of five kinds of software agents (fig.2):

- Interface Agent (IA): this agent interacts with the user of the system allowing him to choose appropriate form of response to his demand so he manages the request and then displays the correspondent result. Therefore, when a user accesses to the TMIS, an agent IA deals with the formulation of his request and then sends it to an available identifier agent. This one relates to the same platform to which several users can be simultaneously connected, thus he can receive several requests formulated at the same time,
- Identifier agent (IdA): this agent manages the decomposition of the requests which were formulated through a same short period of time ε[*](ε-simultaneous requests). The decomposition process generates a set of sub-requests corresponding, for example, to sub-routes or to well-known geographical zones. Sub-requests are elementary independent tasks to be performed by the available set of distributed nodes (information providers) through the ETMN. Each

[*] Fixed by the programmer

node must login to the system registering all proposed services. A service corresponds to the response to a defined task with fixed cost, processing time and data size. Therefore, an agent IdA decomposes the set of existing simultaneous requests into a set of independent tasks, recognizing possible similarities in order to avoid a redundant search. The decomposition process occurs during the identification of information providers. Finally, the agent IdA transmits cyclically all generated data to available scheduler agents. These ones must optimize the selection of providers, taking into account some system constraints,

- Scheduler Agent (SA): several nodes may propose the same service with different cost, processing time and data size. The agent SA has to assign nodes to tasks, minimizing total cost and total processing time to respect due dates (data constraint). Selected set of nodes corresponds to the sequence of nodes which build the Workplans (routes) of the collector agents. An agent SA has firstly to optimize the number of collector agents before assigning nodes to tasks.
- Intelligent Collector agent (ICA): an agent ICA is a mobile software agent who can move intelligently from a node to another through a network in order to collect needed data and finally returns to his home node, noted H. This special kind of agent is composed of data, code and a state and has an intelligent behaviour. Collected data should not exceed a capacity threshold in order to avoid the overloading so the agent SA has to take into account this aspect when assigning nodes to tasks. When they come back to the system, the agents ICA must transmit collected data to available fusion agents,
- Fusion Agent (FA): the agents FA have to fusion correctly collected data in order to compose responses to simultaneous requests. The fusion procedure needs information on behalf of IdA and SA agents and progresses according to the collected data availability. Each new answer component must be complementary to the already merged ones.

To respond to tasks, needed data is available through the ETMN and their collect corresponds to the jobs of ICA agents. Therefore, the SA agent must optimize the assignments of nodes to tasks, minimizing total cost and processing time to respect due dates. To this assignment problem, we propose a two-level optimization solution, expressing the complex behaviour of an agent SA, which was already studied and implemented in previous works [7, 8]. The first level aims to find an effective number m of ICA agents, building their initial Workplans in order to explore, in an optimal manner, the ETMN completely [7]. The second level represents the data flow optimization corresponding to the nodes selection to increase the number of satisfied users [8].

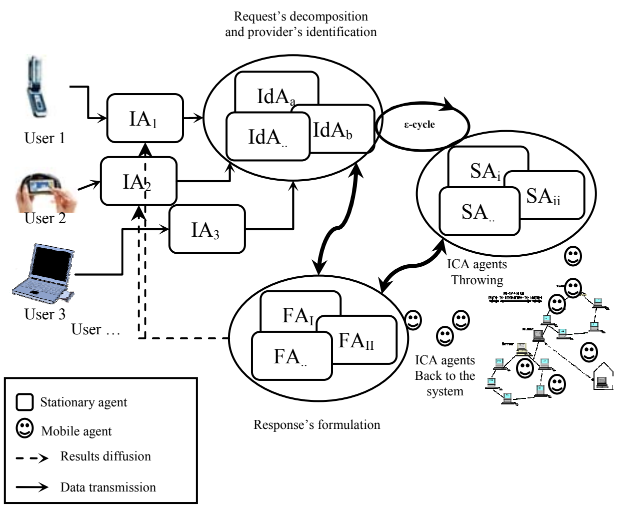

Fig.2. System architecture

A transport user is satisfied if and only if he has a full answer to his request with a minimum cost and respecting the due date. This last step deduces final Workplans of ICA agents from initial ones, using evolutionary algorithms so we designed an efficient coding for the chromosome (the solution) respecting the problem constraints. A possible solution is an instance of a flexible representation of the chromosome, called Flexible Tasks Assignment Representation (FeTAR). The chromosome is represented by a matrix CH(I'×J') where rows represent independent tasks (services), composing globally simultaneous requests and columns represent recognized distributed nodes (providers). Each element of the matrix specifies the assignment of a node S_j to the task T_i as follows:

$$CH[i, j] = \begin{cases} 1: \text{if } S_j \text{ is assigned to } T_i \\ *: \text{if } S_j \text{ may be assigned to } T_i \\ X: \text{if } S_j \text{ cannot be assigned to } T_i \end{cases}$$

An example of a generated FeTAR instance with I'=8 and J'=10 can be illustrated as follows:

CH	S_1	S_{13}	S_{24}	S_{55}	S_{68}	S_{70}	S_{71}	S_{78}	S_{79}	S_{93}
T_8	*	*	*	*	1	*	*	*	*	*
T_{12}	*	*	*	*	x	*	*	1	*	*
T_{18}	*	1	*	*	x	*	*	*	*	*
T_{22}	*	*	*	*	x	*	1	*	*	*
T_{35}	x	*	1	x	x	x	*	*	x	x
T_{51}	x	x	*	*	x	x	x	1	*	*
T_{52}	*	*	*	1	x	x	x	x	*	*
T_{58}	*	*	*	1	x	x	*	*	*	*

The deduction of final Workplans of ICA agents may reduce their initial number m; indeed, when all nodes composing the initial Workplan of an agent ICAk are not selected through the second level optimization approach, this one is removed. Therefore, the number of active ICA agents decreases to $m' \leq m$.

In this paper, we are interested into the interaction between SA agents and ICA agents, especially in case of some network disturbances. In that case, these two kinds of agents have to negotiate the reassignment of tasks which still need providers. The negotiation process will depend on the current positions of ICA agents and also on their priorities, preferences and constraints. Moreover, the SA agents have to assure moderate cost and processing time for the reassignments during the negotiation process.

4 The negotiation process

Some perturbations can occur through the network when ICA agents are following their correspondent final Workplans, according to the generated FeTAR instance. In this case, the ICA agents have to avoid unavailable nodes in their remained final Workplans. In addition, they have to change their itineraries in order to take into account the cancelled tasks which still need assignment because of the perturbations. Therefore, a new assignment process has to occur to find suitable new available providers. To do this, we have to benefit of active ICA agents who are still travelling through the network and to exploit new ones otherwise. So ICA agents have to interact with SA agents in order to find suitable solution to the current situation. Thus, we propose a negotiation process inspired from the well-known contract net protocol [10] between ICA agents who represent the participants of the negotiation and SA agents who are the initiators. In our proposed solution, we allow a partial agreement of the proposed contract (a FeTAR instance) from each ICA agent, to be confirmed partially or totally by the initiator of the negotiation (SA agent). A renegotiation process is necessary while there are still tasks which need to be reassigned. The purpose of this solution is to allow the ICA agents to cooperate and coordinate their actions in order to find globally near-optimal robust schedules according to their priorities, preferences and constraints which depend on their current positions in their correspondent Workplans. Through the negotiation process tours, SA agents must assure reasonable total cost and time.

4.1 Initiators

An initiator of a negotiation is a SA agent who never knows the exact position of each travelling ICA agent. However, he knows all initial Workplans schemes and the final assignments of the servers (final effective Workplans). SA agent does not need to wait for all answers to take a decision, since he can accept a subset of responses to take pressing sub-decisions; urgent actions must be taken according to the current positions of ICA agents. Consequently, SA agent can take decisions every short period of time. In that case, he must update the set of services which need to be reassigned by providers through the confirmation step. After that, he has to propose a new contract according to the updated services set and to the different capabilities of the participants of the negotiation. We suppose that errors on the network are identified before that an ICA agent leaves one functioning node toward a crashed one.

4.2 Participants

A participant of a negotiation is an ICA agent who never knows anything about the other participants of the same negotiation process. Obviously, he knows his own initial Workplan scheme and his final assignments of servers (final effective Workplan). In addition, each ICA agent has his own priorities, preferences and constraints which are dynamic, depending on the network state and on his current position in the already defined final Workplan. Constraints of an ICA agent express tasks which he can’t perform or servers he can’t visit because they cause problems (overloading, time consuming, high latency…). Priorities express servers where the ICA agent prefers visit because they are already programmed in his remained final Workplan. Finally, preferences express servers which are already programmed in the remained initial Workplan and not in the final one. Each time an ICA agent receives a new contract, he analyzes it to make a decision (refusal or total/partial acceptance).

4.3 Proposed protocol

A protocol defines the language used by agents to exchange information. The proposed negotiation protocol (fig.3) is characterized by successive messages exchanges between initiators which are the agents who initiate a negotiation (SA agents) and participants of the negotiation (ICA agents). We designed our protocol so that a negotiation process can occur between several initiators and participants; it can be, for example, the case of simultaneous requests overlapping, but it is not the purpose of this paper. Presently, we describe a negotiation protocol between a unique initiator and several participants. Negotiation always begins with the creation of a contract by the initiator agent, proposing it to active participants. The first contract corresponds to final Workplans which were already optimized thanks to our two-level optimization approach [7, 8]. A renegotiation means a round of modification request for a contract which "a part" has not been accepted the round before. In what follows, we detail the different exchanged messages between initiators and participants.

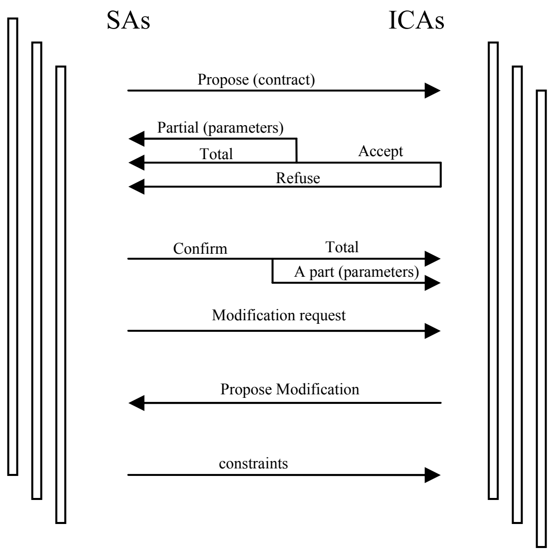

Fig.3. Proposed protocol

4.3.1 Proposition of the contract

The contract message is a proposition of a new organization (the first contract) or reorganization of final Workplans to achieve tasks. If the execution of some services was cancelled because of some network perturbations, it is indeed the case of reorganization. This will be done by reassigning one more time servers to these tasks which represent the set of the Dynamic Reassigned Tasks (DRT). The initiator sends an individual contract to each active ICA$_k$ agent who proposes the *contract-reception* service:

<SA$_i$, ICA$_k$, *contract-reception*, propose, ∂, fipa-sl, MASOntology, f> with $\partial = \partial_1$ if it acts of the first contract and $\partial = \partial_2$ otherwise:

$\partial_1 \equiv$ Workplan (Owner : ICA$_k$ Initial : $i_1,...,i_{k_i}$ Final : $f_1,...,f_{k_f}$)

$\partial_2 \equiv$ FinalWk (Owner : ICA$_k$ Final : $f_1,...,f_{k_f}$)

With $i_1,...,i_{k_i}$ represent references of nodes which belong to the initial Workplan of the ICA agent k (ICA$_k$) and $f_1,...,f_{k_f}$ represent references of nodes which belong to the final Workplan of the same agent. Thus we have $k_i \leq k_f$. In what follow the third field in an agent message (parag.5.1), corresponding to the service, will be null because the conversation will be identified thanks to the last field *f* to shape a conversation.

4.3.2 Response to the contract

When a participant receives the proposed contract, he studies it and answers by:

- A total acceptance if he agrees to coordinate all tasks chosen by the initiator, included in its remaining trip (remained final Workplan), according to its current position,

<ICA$_k$, SA$_i$, Ø, accept-proposal, ∂, fipa-sl, Ø, f>

- A partial acceptance if he agrees to coordinate a sub-set of the tasks selected by the initiator, included in its remaining trip (remained final Workplan), according to its current position, the partial-accept-proposal message content expresses the references of cancelled tasks and those of non available servers (the reason of the non total-acceptance):

<ICA$_k$, SA$_i$, Ø , *partial-accept-proposal*, ∂, fipa-sl, *ICANegotiationOntology*, f> with $\partial \equiv$ (tasks: $c_1,...,c_n$ nodes : $r_1,...,r_m$)

- A refusal if he does not agree with any task in the proposed contract, the *refusal* message content expresses the references of non available servers (the reason of the refusal):

<ICA$_k$, SA$_i$, refuse, ∂, fipa-sl, Ø, f> with $\partial \equiv (r_1,...,r_m)$

The initiator does not wait for all answers because he must act rapidly, so he just waits for some answers for a very short period of time to make a decision; this feature is expressed in the last field f of an agent message, through the *reply-by* facet (parag. 5.1).

4.3.3 Confirmation

An initiator has to confirm independently the agreed part of each contract k proposed to an agent ICA$_k$ who represents an autonomous participant of the negotiation, the confirmation can be:

- Total if the initiator agrees with the total response to the previous proposed contract: <ICA$_k$, SA$_i$, Ø, confirm, Ø, fipa-sl, Ø, f>
- Partial if the initiator agrees with a partial response to the previous proposed contract, the *partial-confirm-proposal* message content expresses the references of agreed tasks:

<ICA$_k$, SA$_i$, Ø, *partial-confirm-proposal*, ∂, fipa-sl, Ø, f> with $\partial \equiv (g_1,...,g_p)$

4.3.4 Modification request

If the DRT table is not yet empty, the initiator asks participants to propose a new distribution of services assignments which are canceled, the *request-modification* message content expresses the DRT table: <SA$_i$, ICA$_k$, Ø, *request-modification*, ∂, fipa-sl, MANegotiationOntology, f> with $\partial \equiv$ (DRT)

4.3.5 Modification proposition

In a previous work [11], we designed a reassignment procedure strategy of servers to tasks, taking into account not only the dynamic positions of ICAs in their Workplans, but also their constraints, priorities and preferences, according to their respective current positions. Constraints of an ICA agent express tasks which he can not perform or servers he can not visit because they cause problems (overloading, time consuming, high latency…). Priorities express servers where the ICA agent prefers visit because they are

already programmed in his final Workplan. Finally, preferences express servers which are already programmed in the initial Workplan and not in the final one. The *proposition* message content expresses for each participant k the new proposition of his remained Workplan according to his current state:
< ICA_k, SA_i, Ø, propose, ∂, fipa-sl, MASOntology, f>
with $\partial \equiv$ FinalWk (
Owner : ICA_k Final : $f_1,...,f_{k_f}$) Where $f_1,...,f_{k_f}$ represent references of nodes which belong to the final Workplan of the agent ICA_k.

4.3.6 Cancel
To avoid indefinite waiting for answers or for modifications, the initiator agent must take a decision at the end of a fixed period of time, illustrated by the last field of an agent message (parag. 5.1). Therefore he cancels the contract if there is no solution (lack of resources, no available provider…) or he creates new ICA agents to execute the current contract: < SA_i, Ø, ICA_k, Ø, ∂, fipa-sl, Ø, f>

The complex interactions features of our system exceed the limits of the traditional negotiation systems which impose several restrictions on the type and format of negotiation messages. Therefore, we designed and implemented a flexible transport ontology which allows an easy handling of the terms and messages for negotiating.

5 Flexible transport ontology
We aim to define a proper vocabulary to the whole proposed multi-agent system (section 3) in order to automate the different kind of exchanges between agents. Therefore, we propose extensible ontologies (fig. 4) which can adapt to all possible kind of interactions. Therefore, in order to perform a proper semantic checks on a given agent expression, it is necessary to classify all possible elements in the domain of discourse. Thus, we have to distinguish between predicates and terms. This classification is derived from the Agent Communication Language (ACL) defined in FIPA that requires the content of each ACLMessage to have a proper semantics according to the performative of the ACLMessage [9]. In this paper, we derive our different edges of ontologies from a basic one (level 0) which already defines fundamental features. Thus, in order to keep a flexible ontology aspect, we start our derivations with a Generic Ontology (level 1) where we define the concept Element which is able to represent any element in any target logistic field: transport, hospital… Each element has a unique reference which represents it in the global ETMN and an order number for the management. In this paper, we focus on the transport field (level 2) represented by the TransportOntology where we define the Task, Server, Request, Service and ServiceTable concepts and also the "Provides" and the "Available provider" predicates. We let a flexible choice of modeling (level 3), here we adopt the multi-agent system modeling so we designed the MASOntology where we define the Workplan concept, the "Performs" agent action, the "available agent", the "IsInitialOf" and finally the "IsFinalOf" predicates. Through our proposed multi-agent approach we used a negotiation strategy (level 4) for which we designed the ICANegotiationOntology which defines a special vocabulary to the negotiation of agents ICA with agents SA: "PartialConfirm" and "PartialAccept" agent actions and "IsPriorityOf", "IsPreferenceOf" and "isConstraintOf" predicates. For the implementation of our whole system (agent behaviours, communication, interactions…), we already used Java Agent DEvelopment framework (JADE). It is a middleware which allow a flexible implementation of multi-agents systems and offers an efficient transport of ACL messages for agent communications which complies with FIPA specifications. Jade offer the "yellow pages" service which allows agents to publish one or more services they provide so that other agents dynamically find them and successfully exploit the proposed "yellow pages" services at a given point in time. Besides, this middleware includes a proficient support for content languages and ontologies, that's why we are implementing our semantic hierarchy of ontologies with JADE framework.

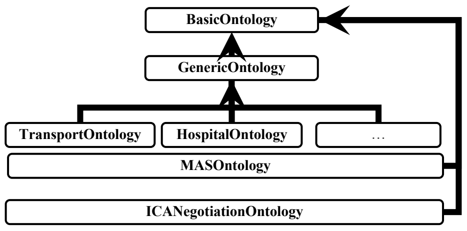

Fig.4. Ontology packages

5.1 The agent message
We opt the following structure for an agent message exchange: <sender, receivers, service, perform, content, content-lang, ontology, f> with:
- sender: the sender of the message,
- receiver: the list of receivers, they represent the recipients of the message,
- service: the "yellow-pages" service proposed by the receiver of the message,
- perform: the performative which expresses the communicative intention,
- content: the information included in the message,
- content-lang: the content language which represents the used syntax to express the content,

- the ontology: the vocabulary of the symbols used in the content and their meaning,
- f = <f1, f2, f3, f4, f5> represents some fields used to control several concurrent conversations and also to specify timeouts for receiving a reply. In this paper, we don't assign this field but we just explain it for a best comprehension of message exchanges:

f1. reply-to A: the recipient of the message reply is the agent A,

f2. conversation-id ide: a conversation identifier which may be fixed by the sender of the message in order to identify the ongoing sequence of communicative acts, that together form a conversation,

f3. reply with exp: identify the reply to the current message with the expression exp,

f4. in-reply-to exp: to denote that this message is a reply to an earlier action of which the reply was denoted by exp (f3) ,

f5. reply-by d: time and/or date expression which indicates the latest time by which the sending agent would like to receive a reply.

5.2 Predicates

Predicates are expressions that say something about the status of the world and can be true or false. For the negotiation process, we need also some predicates, already defined in super-classes ontologies.

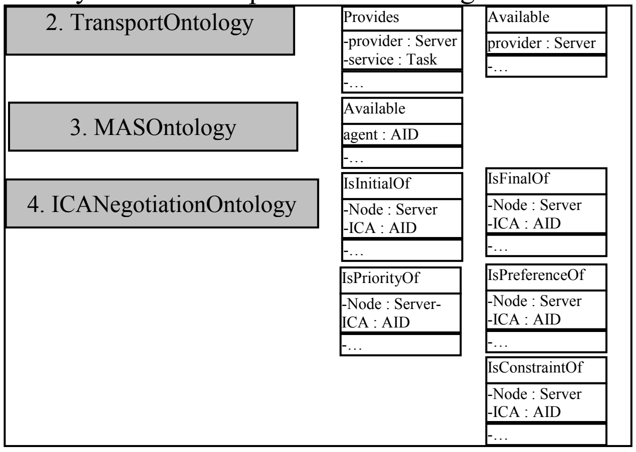

Fig.5. Ontology predicates

5.3 Terms

Terms are expressions identifying entities (abstract or concrete) that "exist" in the world and that agents talk and reason about. We distinguish in our design: Concepts and Agent actions:

- Concepts: expressions that indicate entities with a complex structure that can be defined in terms of slots, examples:

(Element: Ref 14 : Order 2),
(task: Ref 2 : Order 15 : providers 2 12 15 : nbProviders: 3),

- AgentActions: special concepts that indicate actions that can be performed by some agents, examples:

(Performs (node: Ref 2 : Order 5: Services 2 8 6 : nbServices 3) (task: Ref 5 : Order 5 : providers 2 6 : nbProviders 2)),

Using the described flexible transport ontologies, we describe in next paragraph the adopted interactions between ICA agents and SA agents through a negotiation process.

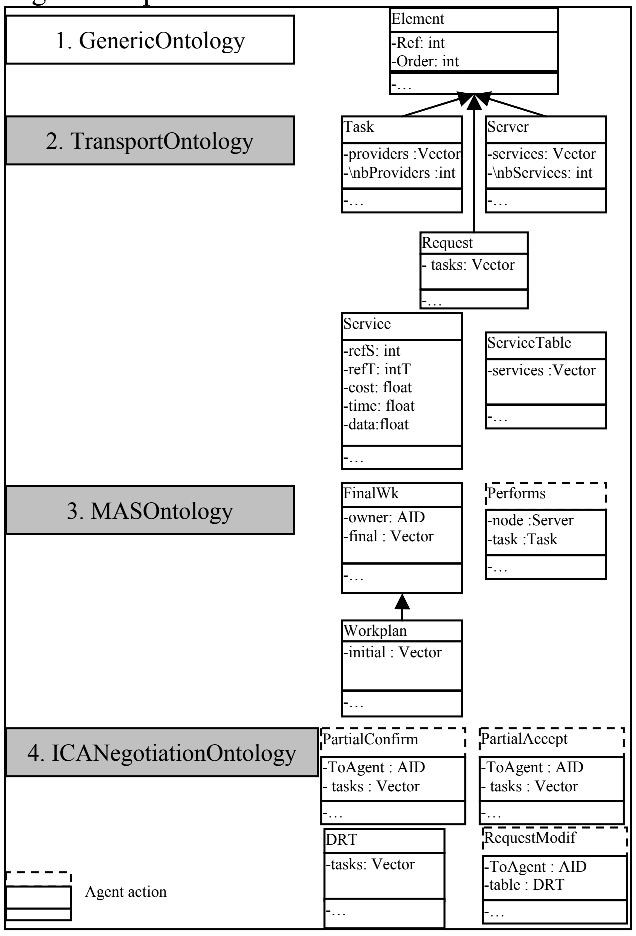

Fig.6. Ontology terms

6 Simulation

We are developing our system, with JADE platform (Java Agent DEvelopment framework). JADE is a middleware which permits a flexible implementation of multi-agent systems; it offers an efficient transport of ACL (Agent Communication Language) messages for agents communications which complies with FIPA specifications. JADE is written in java language, supports mobility, evolves rapidly and until there, it is the only existent multi-agent platform which tolerates web services integration. We use a JADE graphical tool which sniffs message exchange between agents. This tool is useful to debug a conversation between agents. On the left side window of the Sniffer graphic tool (fig.7), we can see available servers containers on the network, where ICA agents can move in order to collect data according to the adopted contract model.

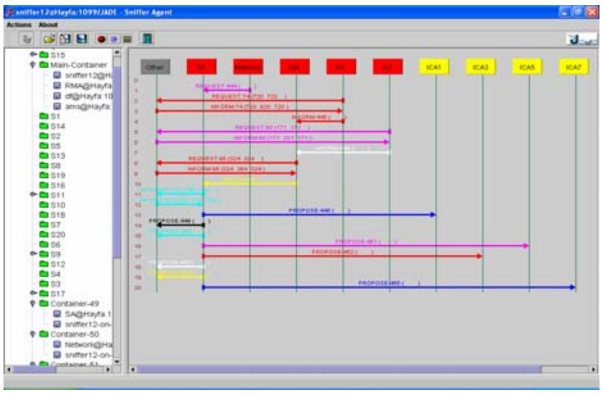

Fig.7. Message exchange between agents

F6, F7 and F9 in fig.8 below represent three different generated FeTAR instances assignments for the same network error scenario (S_{21},S_{59},S_{96}), where the number of agreed assigned tasks was respectively 6, 7 and 9 in the priorities of ICA agents. Thanks to our proposed negotiation process, cancelled services find new available providers through an agreement between static scheduler agents and mobile collector agents of the system so the correspondent transport users are satisfied in spite of some network perturbations.

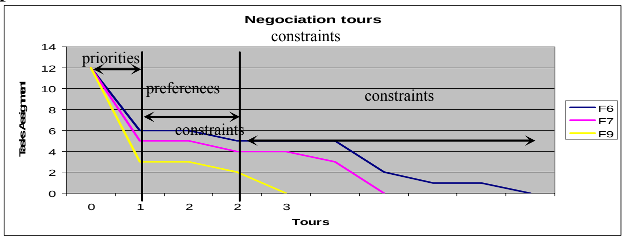

Fig.8. Negotiation tours simulation

7 Conclusion and prospects

In this paper, we proposed a negotiation protocol between static decision maker agents and mobile intelligent agents. Indeed, the recourse to the ontologies was essential for the complex interaction which required a special vocabulary to perform a proper semantic checks on a given agent expressions. In a future work, we aim to manage the interactions between several initiators and participants in different negotiation processes in case, for example, of simultaneous requests overlapping. Thus, the control of several concurrent conversations is indispensable. Besides, the extension or the generalization of the proposed protocol is conceivable.

References:

[1] V.A. Pharm and A. Karmouch. Mobile Software Agents: an overview. *IEEE Communication Magazine*, University of Ottawa, Ontario, (July 1998), 26-37.

[2] W.Theilmann and K.Rothermel. Efficient Dissemination of Mobile Agents. *In Proceedings of the 19th IEEE Int. Conf. on Distributed Computing Systems Workshop*, IEEE Computer Society. (May 1999), 9-14.

[3] A. Carzaniga, G. P. Picco and G. Vigna. Designing distributed applications with mobile code paradigms. *In Proceedings of the 19th Int'l Conf. on Software Engineering*, (July 1997), 22-32.

[4] G.P.Picco and M. Baldi. Evaluating the Tradeoffs of Mobile Code Design Paradigms in Network Management Applications. *In Proceedings of the 20th IEEE Int'l Conf. on Software Engineering (ICSE'97)* (Kyoto, Japan April 1998). In R. Kemmerer and K. Futatsugi, 146-155.

[5] D.P. Buse, J.Q. Feng and Q.H. Wu. Mobile Agents for Data Analysis In Industrial Automation Systems. *In Proceedings of the IEEE/WIC international conference on intelligent Agent Technology (IAT'03)* (Halifax, Canada, October 13-16, 2003), 60-66.

[6] H. Zgaya, Slim Hammadi. Assignment and Integration of Distributed Transport Services in Agent-Based Architecture. *In Proceedings of the IEEE/WIC international conference on Intelligent Agent Technology (IAT'06)* (Hong Kong, China, December 18-22, 2006).

[7] H.Zgaya, S.Hammadi, K.Ghédira. Workplan Mobile Agent for the Transport Network Application. (IMACS'2005) (Paris, France, July 11-15, 2005).

[8] H.Zgaya, S.Hammadi, K.Ghédira. Evolutionary method to optimize Workplan mobile agent for the transport network application. (IEEE SMC'2005) (Hawaii, USA, October 10-12, 2005).

[9] G. Caire, D. Cabanillas. *Jade Tutorial Applications-defined Content languages And Ontologies*. JADE Tutorial (Nov. 2004).

[10] Reid G. Smith. The Contract Net Protocol: highlevel communication and control in a distributed problem solver. IEEE Transactions on computers, C-29(12):1104–1113, (Dec. 1980).

[11] H. ZGAYA, Slim HAMMADI. Dynamic Approach to Reassign Tasks when Servers Breakdown in a Multi-Modal Information System. *In Proceedings of the 2006 IMACS Multiconference on Computational Engineering in Systems Applications (CESA'2006)* (Beijing, China, October 4-6, 2006). Tsinghua University Press, Beijing, China, 4, 1 (Oct. 2006) 985-990.