Multi-Agent Based Model for Web Service Composition

Karima Belmabrouk, Fatima Bendella
Department of Computer Science
University of Science and Technology of Oran
Oran, Algeria

Maroua Bouzid
GREYC- Campus 2 - Sciences3
University of Caen, UNICAEN
Caen, France

***Abstract*—The evolution of the Internet and the competitiveness among companies were factors in the explosion of Web services. Web services are applications available on the Internet each performing a particular task. Web users often need to call different services to achieve a more complex task that can't be satisfied by a simple service. And users often prefer to have the best services responding to their requests. In this context, we should measure the Quality of Service (QoS) which is a very important aspect of Web services in order to offer to the user the best services.**

"How can we ensure the composition of different services to respond the user request" is the first problem that we contribute to resolve, proposing a multi-agent based model for the automatic planification of Web services. And "guarantee the required quality of the composite Web services" is a complex task regarding the unpredictable nature and dynamics of composite Web services, so our contribution to remedy to this problem consists of the use of two classes of quality attributes. The first one considers generic and the second contains specific attributes.

Keywords—Agents; Model; Quality of Service; Service composition; Web services

I. INTRODUCTION

Web services are applications available on the Internet performing each a special task, such as booking a flight ticket. However, the use of Web services is very limited because they perform only one special task; for example, to go on vacation, a person will find a Web service for each booking he wants to achieve (plane, train, and hotel).

To resolve this problem, Web services can be combined manually to form a single service; it is the composition of Web services.

There are several standards used specifically to describe, locate and compose Web services. We can mention for example: UDDI for Universal Description, Discovery and Integration, WSDL for Web Services Description Language, WSCI for Web Service Choreography Interface, WSFL for Web Service Flow Language and more recently WS-BPEL for Web Services Business Process Execution Language [3, 4, 9, 12 and 13].

The UDDI provides the discovery of services. In practice, it contains technical information and information on business processes such as the address for accessing Web services, but also much more contextual information such as the name of the responsible for their management, a brief description of their functionality or the name and the company's business branch on which they depend.

The present article consists of five sections; after a brief introduction, the second section is devoted to the presentation of the problem of the Web services composition. The third one concern exhibition of some research works made to remedy various problems related to the composition of Web services. We move then to the fourth section where we present the concept of quality, and some works based on the quality of Web services, issues in Web services composition.

In the fifth section, we present our multi-agent based model, specifying the detailed functional architecture of each agent proposed in the model.

The last part of this paper presents a conclusion about our proposed model, and we exhibit some perspectives of our future research works.

II. THE WEB SERVICES COMPOSITION

Web service composition refers to the process of creating a composite service, offering new functionality, from simple existing Web services. Several kinds of process ensure this operation like the dynamic process of discovery, the process of integration and the execution process of these services in a particular order to satisfy a definite need [7].

The composition of Web services has the objective of determining a combination of services based on a client or a user request. On this side, this operation appears as a single service and it will be transparent to the user, even if it represents the combination of several Web services.

III. STATE OF THE ART OF WEB SERVICES COMPOSITION

Various studies have been made to compose Web services like those presented in [1, 3, 4, 5, 7, 9, and 10]. A planning process can be used to find the correct order of Web services in the composition. As the number of services can be relatively important, it is necessary to add an optimization phase whose purpose is to provide the user with the best service compositions according to special criteria.

The proposed approach of A. Yachir et al. in [11] aims to build a plan for a composition of services after verifying its feasibility.

M. El Falou used different planning techniques to overcome some problems related to the composition of services in [5]. He proposed a distributed agent-based model,

developed architecture to respond to user requests, diagnosed online the status of implementation of the plan and repaired automatically his proposition in the case of fault detection, during the execution.

D. Pellier and H. Fiorino proposed in [9] an original architecture of automatic composition of Web services by planning techniques. They presented a fully distributed planning model in which agents were reasoning together to achieve a common goal predefined by the user, creating a global plan representing a possible composition of their services.

The objective of D. B. Claro [3 and 4] was to compose automatically and optimally Web services dedicated to produce cost estimates using planning services to detect which belong to the composition. She proposed a framework made up of several phases that can negotiate services to automatically provide users with optimum results with generic optimization criteria.

However, users can have their evaluation criteria or even contradict the generic one, which has been the work of P. Albers and O. Licchelli in [1]. They utilized the SPOC prototype presented in [3 and 4] and they used the user profiles in order to improve the performance of this system, and they presented the most adequate solutions to a particular user. Researchers validated this approach using an example of building construction based on the ontology of French public market.

HSN is a model of composition that permits correcting and validating a service made before his execution proposed in [2]. The editor allows the progressive-composition followed by verification of time constraints. That means that the user could locate and correct any temporal conflicts in its specification and he can export the RSH file for future use.

The work presented by N. Temglit et al. in [10] proposed a model for the composition of semantic Web services. In this model the concepts handled by Web services application domain are represented semantically, i.e., operations and static concepts used to describe the properties of Web services.

The convergence between Web services and Semantic Web aims to increase the expressiveness of descriptions and make more efficient management, discovery, composition and invocation of services through a communication protocol and a directory service UDDI (Universal Description, Discovery, and Integration). This protocol allows a "supplier" to register his service and a "consumer" to find the right service.

Researchers proposed different levels of abstraction to the “operation” concept to allow a progressive-access to concrete-services. Thus, two levels of composition at different granularities (abstract and concrete) may be generated which will reuse plans already constructed to meet similar needs, even with modified preferences.

IV. Some Work on the Quality of Service

The term QoS refers to the ability to provide a service (such as a communication medium) according to the requirements of response time and bandwidth.

Many types of research corresponding to the quality of Web services have been published in recent years [14, 15, 16 and 17]. They revolve around different areas of research such as:

- How to define a variety of factors of service quality?
- How to specify QoS runtime information based on their volatility and complexity?
- How to match user needs with existing services regarding quality?
- How to set user preferences for Web services?
- How to perform the classification of similar services on user preferences?
- What are the factors that predict the quality of service in some environmental conditions that involve dependencies between QoS parameters and relationships with contextual factors?

N. Kokash and V. D’Andrea proposed a method based on risks to the assessment of Quality of Service [6]. Their approach allows a simple choice of Web services with consideration of several service quality factors. They studied the risk factors to measure the impact of failures of atomic services in the composition of services. The major disadvantage of this metric is that different compositions require redundant recalculate risks, making the computational approach less effective than methods based on the evaluation of QoS for well-defined attributes.

A new approach, to monitoring the composition of services, has also been proposed by A. Yachir et al. [11], in addition to that of the composition of services, to take into account the dynamic and stochastic aspects of the ubiquitous environment. This second approach takes into account the context of use and the quality of services available. Three mechanisms are the basis of the proposed method, namely the Bayesian learning for the invocation of services, automatic discovery of services and the automatic and dynamic reorganization plan composition of services. Authors of these approaches tested them in various application scenarios. The results have shown their feasibility and their usefulness in robotics ubiquitous.

SPOC, which is a system proposed by D. Claro in [3 and 4], consists of four phases:

- Discovery service that helps to identify services from an ontology UDDIO services;
- Planning that allows you to organize the tasks to be performed;
- Execution that sends a request to each service to get information such as cost, duration, etc.; and
- Optimization whose goal is to provide the user with some compositions optimized according to predefined quality criteria such as cost, price, revenue and reputation, and that using the genetic algorithm NSGA-II.

The proposed approach in [8] is to change, dynamically, the path of execution of a composite Web service when it is necessary to meet the QoS requirements. A prototype has been proposed to evaluate the proposed QoS monitoring and adaptation architecture on all for minimizing the response time of composite Web services.

L. Zeng et al. proposed AgFlow, a prototype of Web services composition with the support of quality characterized by [12]:

- A model of service quality to assess the overall quality of Web services.
- Two alternative approches to service selection for the execution of composite services.

AgFlow is a platform that provides tools for the definition of ontology services; the specification of composite services using state diagrams; and the connection of the service tasks corresponding to a composite service.

AgFlow platform was used to validate the feasibility and benefit of the proposed approaches. In particular, the greatest composite services incorporating a big number of candidate services components have been created and used to conduct experiments with very encouraging results: the computational cost of the planning phase of a composite service with 80 tasks is to about 8 seconds in a static environment and about 190 seconds in a dynamic environment. The comprehensive planning approach leads to a better quality of service, particularly to reduce costs and execution times.

L. Li et al. Presented in [14] a selection approach based on multi-dimension QoS which introduced multi-dimension QoS to describe the QoS properties of Web service composition and computes the QoS of every dimension of Web service composition. The most important result according to this research is the ability to extract the Web services composition which can best respond to the non-functional constraints from the Web service compositions that can satisfy functional requirement of users.

C. Jatoth and G.R. Gangadharan proposed in [15] two fitness metrics using. First, they choose the best fit services for composition by their local fitness value based on the combination of various QoS factors and prioritization. Then, they use a global fitness value to minimize the computational complexity, around service composition. The experiment results indicated the importance of the proposed approach towards scalable and robust QoS-aware service composition. They also plan to focus their future work on another approach based on fuzzy logic in order to evaluate the local fitness value metric.

Z. Zheng et al. conducted evaluations on user-observed QoS of Web services from distributed locations. Service users invoked great number of Web services under heterogeneous environments on real-world Web services. They presented comprehensive experimental results, and released reusable data sets [16].

W. Khowfa et al. used A Multi-Criteria Decision Making (MCDM) and Quality of Service (QoS) attributes in [17] to obtain the appropriate non-functional service from similar service's pool available in the cloud.

V. Our Multi-Agent Architecture for Planning the Composition of Web Services

Our solution to remedy to the problem of user satisfaction constraints, when their requests need composition of different Web services, is to propose a fully distributed model in which various kinds of agents complete their tasks, jointly, to achieve a common goal preset by the user. Those agents create a global plan to define a possible composition of their services. We will describe our detailed proposition in the following sections.

A. Basic work

We can represent the set of available Web services by a graph whose nodes represent the input and output attributes as well as Web services. And the arcs between these nodes correspond to be links between input attributes and a Web service or connections between a Web service and its output attributes i.e. the results produced after the execution of the Web service. A Web services composition corresponds to a sub-graph of the graph of available Web services.

M. El Falou proposed two algorithms, in this context [5]. The basic idea is to extract the best locally with the use of a global heuristic based on a local evaluation of the best local plan to achieve the goal, and also on an estimation of the globally distributed heuristic that guides the agent to choose its best local plan, ensuring the completeness and the optimality of the algorithm.

To calculate the global heuristic, each service which joins the group of Web services agents must calculate any links between its local services as well as all relationships with all other services.

An agent calculates the global heuristic associated with a service once for each query, back chaining the goal state to the initial state. The global heuristic always involves estimating a distributed distance of a local state q_i^k of the agent A_i to the goal state, taking into consideration the best local plans of the others.

Initially, agent A_i calculates, locally, the distance from q_i^k to the nearest node to the goal node. This distance $(q_i^k, q_i^{k\prime})$, where $q_i^{k\prime} = \pi_i[q_i^k]$ is the result of execution best local partial plan πi at the local state q_i^k. Then, each other agent A_i can calculate its local distances distance $(q_{\propto_j}^l, q_{\propto_j}^{l\prime})$ between all pairs of states $q_{\propto_j}^l \; and \; q_{\propto_j}^{l\prime}$ of all the local states.

We ensure the propagation of these local distances, from the local state q_i^k to neighboring agents, by using the dependencies between the agents i.e. a link appears between a service of an agent $A_{\propto_j}$ and a service of another agent $A_{\propto_{j+1}}$, when the first agent produces effects that are pre-conditions of the last one. And this is to obtain a global heuristic measure that can be defined by:

$$distance(q_i^k, but) = distance\left(q_i^k, q_i^{k'}\right) + \sum_{\alpha_j \in I/i} distance\ (q_{\propto_j}^l, q_{\propto_j}^{l\prime})$$ where:

$q^{l\prime}_{\propto_j} \equiv q^{l}_{\propto_{j+1}}.i$.e. the final state of the $A_{\propto_j}$ has a direct external link to the original state of the agent $A_{\propto_{j+1}}$;

I: the index set of agents {1... n}.

The best local plan π_i of an agent A_i is one that reaches a state $q_i^{k\prime}$ such that:

$\pi_i[q_i^k]=q_i^{k'}$ and $q_i^{k'} = \arg\min Distance\,(q_i^k, but)$.

Upon receipt of a query R = (init, goal) by an agent, and once the global heuristic calculated, the extraction of the plan is achieved by a forward strategy from the initial state to the goal state.

At initialization, the system proceeds to color all executable services in the init state.

Then, for each neighbor service *Sn* of a colored service *Sc*, *Sn* is also colored if and only if all its incoming arcs out colored services. By doing this, we get the executable sub-graph of services; where each service in this graph is an accessible and executable service.

To response to user's exigencies of composite Web services, the system must consider all specified features, to extract the mains criteria to which they must respond.

Such a system should consider the fact that the QoS may have multiple dimensions, and that the QoS of the composite services is, essentially, determined in function of QoS of all services used to compose the global Web service.

For example, the user may be required to minimize execution time while satisfying some constraints regarding price and reliability, while another user may give more importance to price at runtime. QoS approach to service composition is, therefore, essential that maximizes the QoS performances of composite services, taking into account the constraints and preferences set by the user.

UDDI directory can locate on the network the desired Web service. This step is very important because it allows access to directories of potential users of Web services.

B. Our contribution

We were based on the various descriptions of qualification present in the UDDI for selecting the best Web services that may belong to a composite service. This task will be assigned to each agent when choosing its best locally, taking into account the different qualifiers specified by the client, when running the application.

1) Formal definition of Web services

E is an environment consisting of the following:

SW: is a set of Web services SWi,

SW = {SW1, SW2... SWn} = {SWi / i = 1...n}.

SWi: is a Web service described by a set of input parameters Pei and output parameters Psi,

SWi = (Pei, Psi).

Pei: is a set of input parameters of the Web service SWi,

Pei = {pei1, pei2... peim} such that m is the number of input parameters (m = Card (Pei)),

Pei = {pei1, pei2... peim} = {peij / j = 1...m}.

Psi: is a set of output parameters of the Web service SWi,

Psi = {psi1, psi2... psik} such that k is the number of output parameters (k = Card (Psi)),

Psi = {psi1, psi2 … psik} = {psij/j=1...k}.

2) Formal definition of semantic links

LS: is a set of semantic connections between Web services. We can represent LS by a square matrix SWxSW.

$$LS = \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}$$

LSij: is a set of semantic connections between Web services SWi and SWj defined as follows:

$$ls(SWi, SWj) = \begin{cases} 1 \text{ if and only if } \exists psit \in Psi \text{ and } psit \in Pej \\ 0 \text{ if } \forall\, pejt \in Pej, \text{and } \forall\, psit' \in Psi \rightarrow pejt \neq psit' \end{cases}$$

f_{lsij}: a function that measures the intensity of the semantic link between two Web services SWi and SWj, defined as follows:

$$f_{lsij} = \frac{Card(PSi')}{Card(PEj)} \;/\; Psi' = \{psit \in Psi \text{ and } \forall t, psit \in Pej\}$$

Where:

$f_{lsij} = 1$ if and only if $Card(PSi') = Card(Psi)$ and

$Card(Psi) = Card(Pej)$.

This equation is evident, if all output parameters psik of a Web service SWi match exactly with all input parameters pejt of another Web Service SWj.

$f_{lsij} = 0$ if and only if $Card(Psi') = 0$.

That means that we can't find any output parameter psi of a Web service SWi in Pej, the set of input parameters of a Web service SWj.

3) Formal definition of the criteria of the quality of service

In the same environment defined above, we can integrate the following:

- C: is a set of vectors quality parameters Ci corresponding to different Web services SWi.
- Ci: is a vector of quality parameters, Ci = (ci1, ci2... cik) Such that k is the dimension of vector Ci.

Relative to a given domain D, it is possible to classify the attributes of service quality by generic or specific attributes. Such as generic attributes are independent of domains to which they belong and the specific attributes are related to the Web services associated with domain to which they depend on.

The model we propose is composed of two classes of quality attributes:

- CGEN: is the set of generic attributes

- CSPE: is the class of specific attributes

In the case of generic attributes CGEN, we can distinguish between measurable attributes and non-measurable ones. So we extend the notation of those parameters as follows:

- c_i^m: is a measurable attribute of a generic Web service SWi.
- c_i^{nm}: is a generic non measurable parameter of a Web service SWi.

Therefore, c_{ij}^m the jth measurable generic attribute for Web service SWi and c_{ik}^{nm} the kth non measurable generic one for the same Web service SWi.

We note that most of the measurable attributes are described using performance parameters like:

- The flow: representing the number of service requests during a time interval.
- The response time: indicates the time required to complete a request for a Web service.
- Reliability: this reflects the ability of a service to perform his duties properly.
- Scalability: representing the service's ability to handle the largest number of operations or transactions during a given period, while maintaining the same performance.
- Robustness: this is the probability that a service can respond correctly to invalid input messages, incomplete ones or with conflicts.
- Availability: the possibility of availability of a service.

There are QoS attributes that are not measurable, but are useful for Web services such as:

- The cost or price of performance: it is the price payable by a Web service client for service.
- The reputation is a measure of the credibility of the service. It depends on the user experience.
- The security is a grouping of a set of qualities including confidentiality, message encryption, and access control.

The question that we can ask about our model is the following: How can we measure a non-measurable generic attribute c_i^{nm}?

Within the class of specific attributes CSPE, it is clear that since these kinds of parameters can only be set following the domain belonging of a SWi Web service, then it is impossible to enumerate or classify these elements with the same way. Therefore, any particular parameter, in our model can be represented as follows:

- c_i^s : a specific attribute on a Web service SWi.

4) Description of our model:

a) General presentation

We propose an automatic architecture for Web services composition based on agents, in order to achieve a distributed planning leading to the creation of a composite Web service seen as a plan defining precedence relationships and causality between basic services and ensuring number of conditions.

Our model is to make a client request, covering a set of parameters and requiring a certain number of constraints that will be considered by an intelligent agent of cognitive type, named «UserAgent», and which will be responsible for managing the preferences of the user, who form his beliefs. He will be responsible also for the synthesis of these constraints, in order to classify them by type.

Another agent used in our model is the «ServiceAgent». A set of them will be, automatically, generated for the purpose of search services that can respond to the request of the user, to filter these services and ordering them according to their relevance by matching each service implicated with an autonomous agent capable of discovering the various offers related to the requested service to choose the best offer and to invoke the service, if necessary.

The third agent in our architecture is «PlanAgent». It's responsible for the Web services composition involved in our process, trying to propose an optimal planning, in terms of time of execution of the overall plan and taking into account the specific constraints imposed by the client.

Each «ServiceAgent» will collaborate with «UserAgent» and «PlanAgent» to propose a global plan to response to the user query, as shown in Fig. 1.

Collaboration between several «ServiceAgent» can take place to try to provide partial plans corresponding to a part of the user query, representing a response to a partial composite service. In this case, it's possible to destroy all «ServiceAgent» implicated in the creation of the partial plan provided by fusion of local solutions and the birth of a new «ServiceAgent» with assigning roles of all agents destroyed. This agent will be able to collaborate with «UserAgent» and «PlanAgent», to search the optimal global plan.

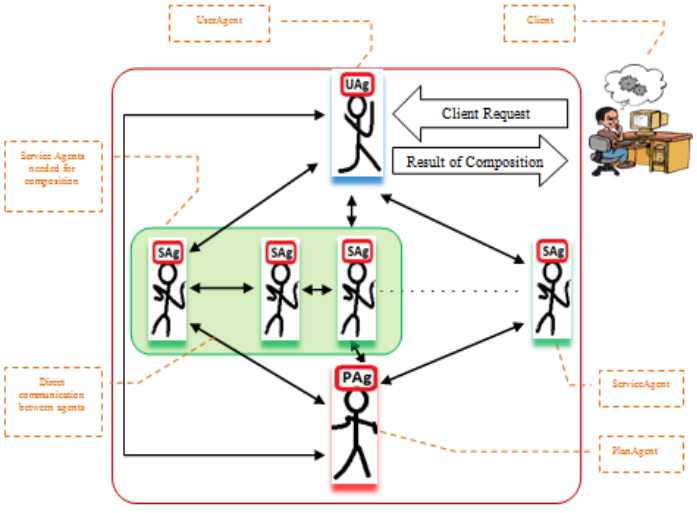

Fig. 1. Agent-based Architecture for Automatic Web services composition

b) The different agent's functional architecture

THE «USERAGENT» ARCHITECTURE:

The «UserAgent» is responsible for processing the user request before forwarding it to the «PlanAgent» will try to find a solution to it. Its primary role is to analyze the application to derive all the parameters related to the purpose and the list of constraints imposed by the user. The «UserAgent» consists of two main modules (see Fig. 2).

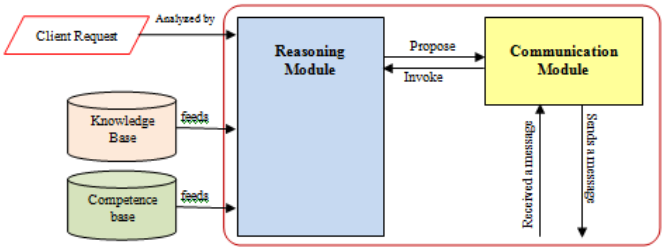

Fig. 2. The «UserAgent» functional architecture

a) The reasoning module: it is the party responsible for the analysis of the user query to determine the parameter list and the input constraints. Based on all knowledge acquired by the agent and his skills, he is able to reason, and to extract the intensions and desires of the user.

b) The communication module: it allows you to manage the communication and interaction with other agents «UserAgent».

THE «PLANAGENT» ARCHITECTURE:

The «PlanAgent» must be able to reason about the task he has to perform and be able to communicate with other agents («UserAgent», «ServiceAgent»). It is initialized with the semantic description of the user to achieve the goal. Its role is to coordinate the work of different agents «ServiceAgent» type to reach the goal.

The «PlanAgent» consists of four modules (see Fig. 3).

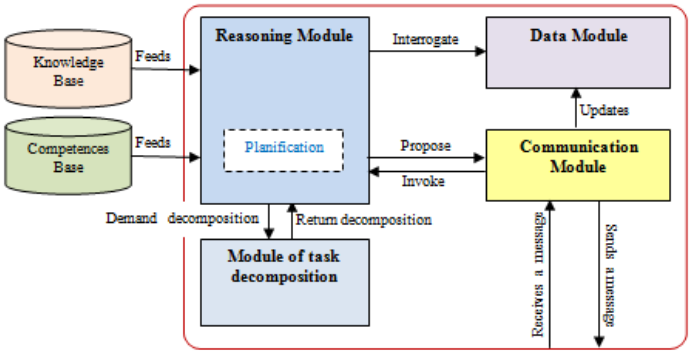

Fig. 3. The «PlanAgent» functional architecture

a) reasoning module: it is the party responsible for defining the behavior of the agent on its interactions with other agents and with the environment, consisting primarily of three essential elements which are:

- The data transmitted by the user, through its request or by other agents.
- Own and skills.
- Knowledge.

b) The data module: which supports the information provided by other agents?

c) The decomposition task module: useful for the decomposition of a major task in elementary sub-tasks where the «PlanAgent» doesn't find a solution to the problem posed by the client.

d) The communication module: it manages the communication and interaction with other agents, and ensures the exchange of messages between them.

THE «SERVICEAGENT» ARCHITECTURE

When the «PlanAgent» cannot find a solution to the query raised by «UserAgent», it invokes The «ServiceAgent. The «PlanAgent», therefore, proceed to the decomposition of the task he could not accomplish to subtasks, and ask agents like «ServiceAgent» to find partial solutions to various elementary tasks. The main role of a «ServiceAgent» is to find a partial plan that meets the client's request, taking into account constraints. It's possible to have talks with a «UserAgent» to negotiate on its preferences. Two essential modules can describe its architecture (see Fig. 4).

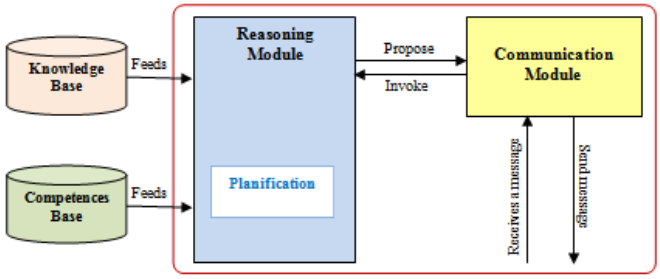

Fig. 4. The «ServiceAgent» functional architecture

a) The reasoning module: it is the party responsible for the search for a partial solution proposed by «PlanAgent» order to contribute to solving part of the problem. The reasoning will be based on the knowledge and skills «ServiceAgent».

b) The communication module: it allows you to manage communication and interaction «ServiceAgent» with another agent which can be either:

- «UserAgent» for the negotiations on the constraints posed.
- Another «ServiceAgent» to collaborate with him in order to get closer to the overall plan, merging their partial plans.
- The «PlanAgent» to provide a partial solution on the stain that has been assigned.

VI. CONCLUSION AND PERSPECTIVES

In this paper, we proposed a multi-agent based model to solve the problem of Web service composition. Our contribution consists of the description of the detailed

functional architecture of each kind of agent used in our prototype.

We suggest to clients to employ two kinds of constraints during the execution of their requests, thus ensuring a better quality of service. A formal presentation of their criteria was introduced to facilitate the evaluation of the optimal plan either in the case of a partial plan or in the case of global one. To validate our model, simulation work is in progress under the «NetLogo» platform where we try to decompose a complex planning problem "P" into N elementary sub-problems "Pi" to find the optimal solution.

An agent type «PlanAgent» is responsible for:

1) Analysis of the query of the «UserAgent»

2) The search of a «ServiceAgent» able to find a solution to the query Q.

3) Presentation of the optimal solution found.

4) If we can't find, the «PlanAgent» is responsible for decomposing the problem Q, into a set of problems Qj that we can resolve, separately, by «ServiceAgent»

5) «PlanAgent» will then be responsible for the composition of partial solutions proposed by the various «ServiceAgent» involved, answering the initial query Q where different strategies in the field of artificial intelligence may be adopted, in this case, among other scheduling algorithms.

Our simulation environment can contain a set of N agents of type «ServiceAgent», as each of them will be responsible for seeking a global or partial solution to the problem Q.

REFERENCES

[1] P. Albers, O. Licchelli. 2007. «Composition de Services Web Personnalisés», Inteligence Artificielle et Web Intelligence, Grenoble.

[2] A. Belkheir, S. Bouyakoub. 2009. «A Hierarchical Model for Web Services Composition», International Journal of Web Services Practices, Vol. 4 No.1 (2009), pp.44-50.

[3] D. B. Claro. 2006. «SPOC–Un canevas pour la composition automatique de services Web dédiés à la réalisation de devis», Thèse de doctorat soutenue le 6 Octobre 2006, Spécialité Informatique; Ecole Doctorale d'Angers, N°d'ordre 767, Université d'Angers.

[4] D. B. Claro, P.Albers, J. Hao. 2006. «Web Services composition», in J. Cardoso, A. Sheth, Semantic Web Services, Processes and Applications, Chapter 8, Springer, p. 195-225.

[5] M. El Falou. 2010. «Contributions à la composition dynamique de services fondée sur des techniques de planification et diagnostic multi-agents», Thèse de doctorat soutenue le 2 juin 2010, Université de Caen/ Basse-Normandie, U.F.R. Sciences, Ecole doctorale SIMEM.

[6] N. Kokash and V. D'Andrea. 2006. «Evaluating Quality of Web Services: A Risk-driven Approach»; Technical Report#DIT-06-099.

[7] B. Medjahed, A. Bouguettaya, and A. K. Elmagarmid. 2003. «Composing Web services on the Semantic Web». The VLDB Journal, 12(4).

[8] M. Qiao, F. Khendek, A. Serhani, R. Dssouli and R. Glitho. 2009. «An Architecture for Automatic QoS Adaptation for Composite Web Services»; International Journal of Web Services Practices, Vol.4, No.1 (2009), pp.18-27.

[9] D. Pellier, H. Fiorino. 2009. «Un modèle de composition automatique et distribuée de services Web par planification», RSTI-RIA-23/2009. Intelligence artificielle et Web intelligence, pages13 à 46.

[10] N. Temglit, H. Aliane, M. Ahmed Nacer. 2008. «Un modèle de composition des services Web sémantiques»; Volume 11-Numéro spécial CARI2008 -Marc Kokou Assogba.

[11] A. Yachir, Y. Amirat, A. Chibani, K. Tari. 2009. «Approche basée sur la qualité de service pour la composition automatique des services en robotique ubiquitaire»; Laboratoire Images, Signaux et Systèmes Intelligents–LISSI (EA3956)–Université Paris Est.

[12] L. Zeng, B.Benatallah, M.Dumas, J. Kalagnanam, and Q. Z. Sheng. 2003. «Quality Driven Web Services Composition». In Proceedings of the 12th international conference on World Wide Web (WWW), Budapest, Hungary. ACMPress, May2003.

[13] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam and H. Chang. 2004. «Qos-aware middleware for Web services composition». In IEEE Transaction on Software Engineering.

[14] L. Li, M. Rong and G. Zhang. 2013. «A Web Service Composition Selection Approach based on Multi-Dimension QoS». The 8th International Conference on Computer Science & Education (ICCSE 2013) April 26-28, 2013. Colombo, Sri Lanka.

[15] C. Jatoth and G.R. Gangadharan. 2015. «Fitness Metrics for QoS-AwareWeb Service Composition Using Metaheuristics». Springer International Publishing Switzerland 2015, R. Neves-Silva et al. (eds.), Intelligent Decision Technologies, Smart Innovation, Systems and Technologies 39, DOI 10.1007/978-3-319-19857-6_24.

[16] Z. Zheng, Y. Zhang, and M. R. Lyu. 2014. «Investigating QoS of Real-World Web Services, IEEE Transactions on Services Computing». Vol. 7, No. 1, January-March 2014.

[17] W. Khowfa, O. Silasai and C. Kaewpruksapimon. 2015. «QoS Based Service Selection in Cloud Environment: A Review». Int. J. Advance Soft Compu. Appl, Vol. 7, No. 3, November 2015 ISSN 2074-8523.