International Journal of u- and e- Service, Science and Technology
Vol.8, No. 7 (2015), pp.337-346
http://dx.doi.org/10.14257/ijunesst.2015.8.7.34

Research on Dynamic Assembled Algorithm of Component based on Semantic Web Service

Luo Dingfu

Department of Computers, Guangdong Songshan Polytechnic College, Shaoguan 512126, China, sunfieldluo@163.com

Abstract

In development of service-oriented software systems based on component technology, dynamic assembling of service components is research issues, but many dynamic assembled algorithms based on keywords have some limits such as low successful rate and so on, it is necessary that research dynamic assembled algorithm. To analyze semantic web service and component technology, some correlative definitions of web service components are given by markup language of web service, the theory of domain ontology is cited, a dynamic assembled algorithm for components of semantic web service based on domain ontology is provided, the dynamic assembled ideal of service component is given according to semantic relationship, the algorithm is realized with pseudo codes. Finally, the feasibility and efficiency about algorithm are demonstrated by experiment and contrast analysis.

***Keywords**: service component, semantic web service, domain ontology, semantic relationship, algorithm*

1. Introduction

The method of service-oriented development based on component technologies which sustain industrialization of colony development is a good method for service-oriented systems [1, 2]. To analyze structure and semantics, services and components have inherent comparability, services are provided by components, service components which combine services and components are reasonable and practical software carrier for services [3]. Web service components which assemble heterogeneous components shield heterogeneity between components, a simple and only web service component can not meet requirements of application, a complicated task is composed of many web service components, when dynamic assembling of components is realized under uniform component model, components are furthest reused and service-oriented systems based on component are realized. The problem about assembling service components is a research issue in service-oriented architecture.

Many dynamic assembled algorithms are based on keyword matching [4-6], semantics of a service is not taken into account so as to limit successful rate of assembling. Some research achievements of semantic web are introduced into web service, in this way, automatic service discovery, transfer, combination, watch and comeback may be realized. In this paper, a dynamic assembled algorithm for component of semantic web service based on combine ontology technology is given.

2. Correlative Definition

As a category of ontology, domain ontology is a reused ontology in specific domain which provides concept definition, concept relation, occurred activities, primary theory and basal principle [7, 8].

ISSN: 2005-4246 IJUNESST

OWL is an ontology description language from W3C recommendatory standard, some definitions are as follows:

Definition 1. There are two concepts G_i and G_j, in domain ontology, if G_i and G_j have same semantic information, then G_i and G_j are called semantic equality, marked by $G_i \equiv G_j$; if G_i semantic information is involved in G_j semantic information, then marked by $G_i \supseteq G_j$.

Definition 2. There are two sets of concept SG_i and SG_j, if $\forall G_j \in SG_j, \exists G_i \in SG_i$, meet $G_i \equiv G_j$ or $G_i \supseteq G_j$,then marked by $SG_i \supseteq SG_j$; if $SG_i \supseteq SG_j$ and $SG_j \supseteq SG_i$, then marked by $SG_i \equiv SG_j$.

According to OWL-S which is an ontology description language of semantic web service based on OWL, some definitions are as follows:

Definition 3. A web service component is described with following expression: $WC_i(I_i, O_i)$, where WC_i is a name of the web services component, I_i is a input set of the web service component, O_i is a output set of the web service component.

Definition 4. A request component of web service is described with following expression: $WCR_k(I_k, O_k)$, where WCR_k is a name of the request component, I_k is a input set of the request component, O_k is a output set of the request component.

Definition 5. To two web service components $WC_i(I_i, O_i)$ and $WC_j(I_j, O_j)$, if $O_i \supseteq I_j$, then semantic association is called from WC_i to WC_j, marked by $WC_i \succ WC_j$, semantic relationship is marked by $D(WC_i, WC_j)$, where WC_i is called pre-service component about WC_j, WC_j is called subsequence services component about WC_i.

Definition 6. To a request component of web service $WCR_k(I_k, O_k)$ and a web service component $WC_i(I_i, O_i)$, if $O_k \supseteq I_j$, then semantic association is called from WCR_k to WC_i, marked by $WCR_k \succ WC_i$, semantic relationship is marked by $D(WCR_k, WC_i)$, where WC_i is called subsequence services component about WCR_k; if $O_i \supseteq O_k$, then semantic association is called from WC_i to WCR_k, marked by $WC_i \succ WCR_k$, semantic relationship is marked by $D(WC_i, WCR_k)$,where WC_i is called pre-service component about WCR_k.

The calculation of semantic relationship of service components is translated into that of opposite parameter similarity of ontology description; the definition of similarity of any two concepts in ontology repository is given as follows:

Definition 7. Computational function between any two concepts in ontology repository $S: G_1 \times G_2 \rightarrow R+$. Where G_1 and G_2 are two concepts in ontology repository G, $S(G_1, G_2)$ is a positive real number which measure , approximate degree of G_1 and G_2, $S(G_1, G_2) \in [0,1]$, meet following two conditions:

1) $\forall g \in G$, $S(g, g) = 1$

2) $\forall g_1, g_2 \in G$, $S(g_1, g_2) = S(g_2, g_1)$

According to the definition of similarity of two concepts, a calculation formula of similarity based on semantic distance is given:

$$S(G_1, G_2) = 1/\exp(d(G_1, G_2)) \quad (1)$$

Where $d(G_1, G_2)$ express semantic distance between G_1 and G_2 [9][10].

If two concept sets $SG_i(G_1^{'}, G_2^{'}, \cdots, G_m^{'})$ and $SG_j(G_1^{''}, G_2^{''}, \cdots, G_n^{''})$ meet $SG_i \supseteq SG_j$, then $SD(SG_i, SG_j)$ express semantic similarity between the two concept sets, the calculation formula as follows:

$$SD(SG_i, SG_j) = \frac{1}{n}\left[\sum_{j=1}^{n} \max_{i=1}^{m} (S(G_i^{'}, G_j^{''}))\right] \quad (2)$$

Where $SD(SG_i, SG_j)$ value range is [0, 1].

If two service components $WC_i(I_i, O_i)$ and $WC_j(I_j, O_j)$ meet $WC_i \succ WC_j$, then according to the calculation formula (2) , the calculation formula of semantic relationship is given from $WC_i(I_i, O_i)$ to $WC_j(I_j, O_j)$:

$$D(WC_i, WC_j) = SD(O_i, I_j) \tag{3}$$

Similarly, the calculation formula of semantic relationship is given from $WCR_k(I_k, O_k)$ to $WC_i(I_i, O_i)$:

$$D(WCR_k, WC_i) = SD(I_k, I_i) \tag{4}$$

the calculation formula of semantic relationship is given from $WC_i(I_i, O_i)$ to $WCR_k(I_k, O_k)$:

$$D(WC_i, WCR_k) = SD(O_i, O_k) \tag{5}$$

Definition 8. A service component assembling is a sequence $WC_1, WC_2, \cdots, WC_n$ which meets a request component of service $WCR_k(I_k, O_k)$, the sequence meets following three conditions:

1) $WCR_k \succ WC_i$

2) $\forall$ WC_i, WC_{i+1} meets $WC_i \succ WC_{i+1}$

3) $WC_n \succ WCR_k$

Definition 9. The satisfaction of service component assembling is a degree which the service component assembling meet a request component of service, marked by SAT , to a request component of servic $WCR_k(I_k, O_k)$, the calculation formula of SAT is as follows:

$$SAT = D(WCR_k, WC_1) \times \prod_{i=1}^{n-1} D(WC_i, WC_{i+1}) \times D(WC_n, WCR_k) \tag{6}$$

3. Algorithm Implementation

Based on domain ontology, firstly, the subsequence service components of a request component of service must be found and then the subsequence service components of the service components must be found too, this process is repeated until a subsequence service component of the service components is the request component of service, service component assembling is successful. But a service component has many subsequence service components, these service components of semantic association are expressed as a directed graph which is called assembled graph of semantic web service component.

Example 1. To a request component of service $WCR_k(I_k, O_k)$, if there are following service components: $WC_1(I_1, O_1), WC_2(I_2, O_2), WC_3(I_3, O_3), WC_4(I_4, O_4), WC_5(I_5, O_5)$, where $I_k \equiv I_1 \equiv I_3$, $O_1 \equiv O_2$, $O_3 \equiv O_4 \equiv O_5$, the assembled graph of semantic web service component is illustrated in Figure 1.

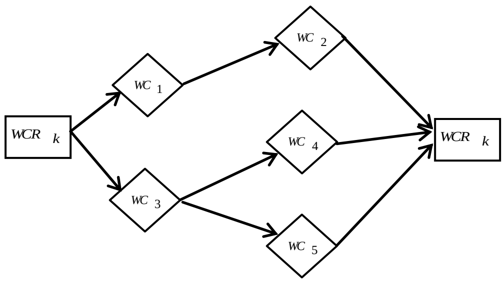

Figure 1. The Assembled Graph Of Service Components

In Figure 1, two service components which are conjoint with a directed edge are semantic association, a start node is a pre-service component of an end node, an end node is subsequence service component of a start node. A rectangle express a request component of service, a rhombus express a service component. In assembled graph of semantic web service component which is corresponding to a request component of service, each path from start point to end point is a service component assembling which meet the request of service component.

It is optimal combination found with the highest efficiency to achieve dynamic assembling of service components. The basic ideal of assembled graph of semantic web service component is optimized: if a service component has many subsequence service components, then according to relationship, these service components are sorted, those of big relationship is sorted in front, those of small relationship is sorted in back, the assembled graph of semantic web service component is formed based on relationship sort of subsequence service components.

Example 2. In Example 1, if $D(WCR_k, WC_3) \geq D(WCR_k, WC_1)$, $D(WC_3, WC_5) \geq D(WC_3, WC_4)$ then the assembled graph of semantic web service component which is optimized is illustrated in Figure 2.

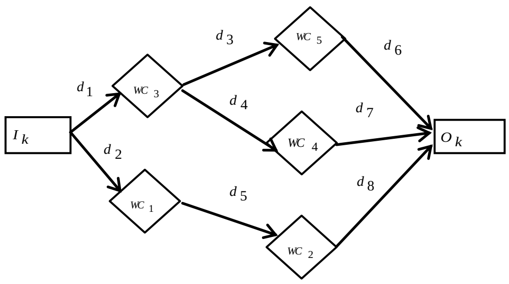

Figure 1. The Assembled Graph of Service Components is Optimized

In Figure 2, the values $d_1, d_2, \cdots, d_8$ express relationship, because of $D(WCR_k, WC_3) \geq D(WCR_k, WC_1)$, $WC_3(I_3, O_3)$ is located above $WC_1(I_1, O_1)$. In the assembled graph, all subsequence service components are sorted by relationship.

The quality and efficiency of service component assembling are controlled by threshold of least assembled satisfaction of service component. The assembled algorithm of service component is described as follows:

There are n web service components $WC_1, WC_2, \cdots, WC_n$, the threshold of least assembled satisfaction of service component is δ , the queue of service component assembling is marked by LWC , request components of service are marked by $WCR_k(I_k, O_k)$, the whole algorithm is described with pseudo codes as follows:

$LWC = NULL$
$WC_i = GetMaxD(WCR_k)$
for i=1 to n
 $addTail(LWC, WC_i)$
 $NeighborWC[WC_i] = GetNextNeighbor[WC_i]$
 $WC_i(I_i, O_i) = GetTail(LWC)$
 If $WC_i \succ WCR_k$ then
  $CS = SAT$
 else
  $$CS = D(WCR_k, WC_1) \times \prod_{j=1}^{i-1} D(WC_j, WC_{j+1})$$
 end if
 If $CS > \delta$ then

If $WC_i \succ WCR_k$ then return LWC
else
$DelTail(LWC)$
$WC_i(I_i, O_i) = GetTail(LWC)$
end if
If $LWC = NULL$ then Return Null
If $NeighborWC[WC_i] = NULL$ then
$DelTail(LWC)$
$WC_i(I_i, O_i) = GetTail(LWC)$
else
$DelTail(LWC)$
$AddTail(LWC, NeighborWC[WC_i])$
loop
end if
$WC_i = GetMaxD(WC_i)$
end for

Where $NeighborWC$ is a pointer array, its each element points a service component, the function $GetMaxD(WC_i)$ return a service component which semantic relationship with WC_i is maximal in all subsequence service components, the function $GetNextNeighbor[WC_i]$ return next subsequence service component relative to WC_i, the function $AddTail(Q, e)$ append e to Q, the function $GetTail(Q)$ return tail element of queue Q, the function $DelTail(Q)$ delete tail element of queue Q.

Example 3. in Figure 2, if d_1 =0.9 , d_2 =0.8 , d_3 =0.6 , d_4 =1.0 , d_5 =0.9 , d_6 =0.8 , d_7 =0.9 , d_8 =0.7, δ =0.8, then the result of applied the algorithm is illustrated in Figure 3.

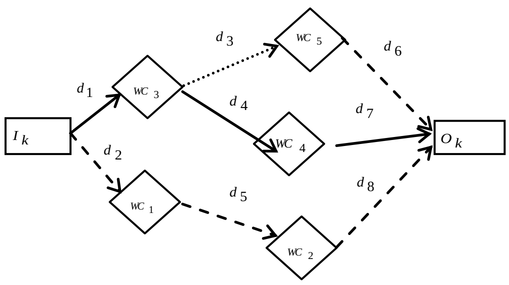

Figure 2. Service Components Assembling

In Figure 3, real line express service component assembling which the algorithm return; long broken line express the path has been traveled, but the path can not meet the request; dot broken line express the path can not be traveled.

In the paper, the assembled algorithm for service components is only travel part paths in the assembled graph of semantic web service component, when the path meet request is found, the result is returned so that ensure find assembled queue as possible as well in best times.

4. Experiment and Contrast Analysis

The experiment compare the algorithm to traditional algorithm that based on keywords [11, 12], the data of experiment are random and simulate the relationship of web service components. Seven data set of testing are created which total numbers of service components are respectively 200, 500, 700, 1000, 1200, 1600, 2100.

Firstly, eight request components of service are randomly given from data set that total numbers of service components is 2100; In the paper, the assembled satisfaction of service component is a criterion measure the quality of service component assembling. The result is illustrated in Table 1, the first column show service request components, and the second column show the assembled satisfaction of service component based on keywords algorithm, and the third column show the assembled satisfaction of service component in the paper.

Table 1. The Quality of Service Component Assembling

service request components	The algorithm in the paper	The algorithm based on keywords
1	0.78	-
2	0.86	1.00
3	0.91	-
4	0.95	-
5	1.00	-
6	0.96	1.00
7	0.87	-
8	0.92	-

The assembled satisfaction of service component is maximal (*i.e.,* 1) using algorithm based on keywords, but the successful rate of service component assembling is very low, service component assembling can not be implemented in many requests.

Secondly, the thresholds of least assembled satisfaction of service components are respectively 0.6, 0.7, 0.8, 0.9, 1.0 in data set that total numbers of service components is 2100, to each threshold, service component assembling are implemented to 100 random service request components by the algorithm in the paper, the successful rate of service component assembling and the average time of assembling are illustrated in Figure 4 and Figure 5.

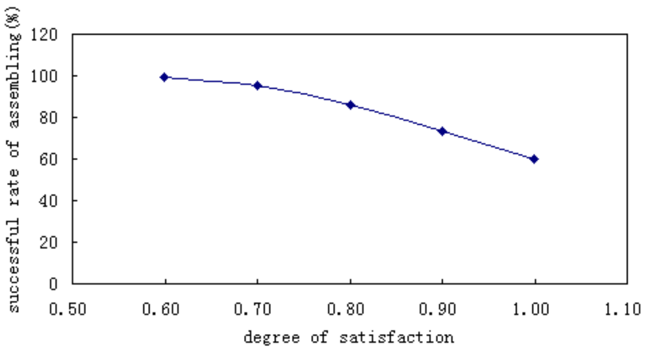

Figure 3. The Relation of the Satisfaction and the Successful Rate

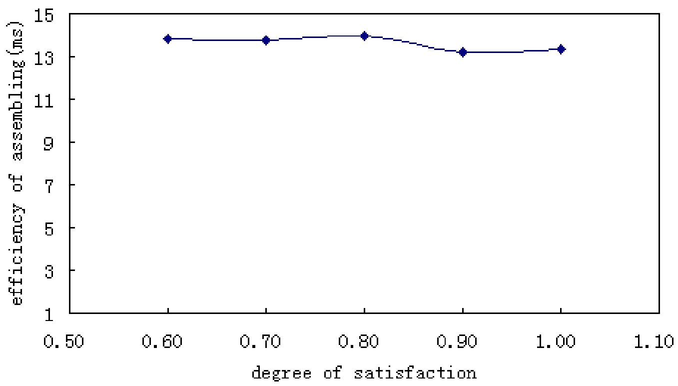

Figure 4. The Relation of the Satisfaction and the Efficiency

According to Figure 4 and Figure 5, the thresholds of least assembled satisfaction of service components do not firsthand influence the assembled efficiency of the algorithm, but the successful rate of service component assembling can be influenced. What the thresholds of least assembled satisfaction of service components are bigger indicates the quality of service component assembling is higher. In this way, service component assembling is few; the successful rate of service component assembling is lower. When the thresholds of least assembled satisfaction of service components are minimal (*i.e.,* 1), semantics is equal and the successful rate of service component assembling is higher than that of the algorithm based on keywords.

The algorithm based on keywords can not consider semantics; the scale of assembling is small, the efficiency is high and the extensibility is good, but the successful rate of service component assembling is low.

The algorithm in the paper is sustained by domain ontology, the successful rate of assembling is far higher than that of the algorithm based on keywords, only part paths are scanned in the assembled graph of semantic web services component, when the service component that meet requests is found, the assembling is stopped so that decrease many unnecessary traveling paths, the efficiency of algorithm is very high and is rarely influence by the total numbers of service components.

5. Related Work

Without considering the semantic Web technology conditions, by industry wide to accept the BPEL4WS language is used for Web service composition the scheme described. Because the BPEL4WS is based on Workflow Description language, which can be compared the characterizations of the Web service composition scheme intuitively modeled as finite state automata. Accordingly, through model checking, reachability points Analysis techniques for generating Web service composition scheme or the existing Analyze and test the Web service composition scheme.

After the introduction of the semantic Web technology, workflow based method for the same can be extended to solve the semantic Web service composition problem. Literature [13] for semantic Web service input and output parameters investigation, makes the semantic concept referring to each parameter as a proposition, so that each atom Web service is modeled as a set of propositional Horn clause. At the same time, users can output parameters and input parameters to provide the desired modeling of life

Problems in the Horn clause of facts and goals, which will serve to combined question. The question is converted to logical reasoning questions about the Horn clause in this medium. On the foundation by the Petri network to the Horn clause set for modeling, will help to the Web service composition problem for invariant technique solution. The method based on workflow, is a common characteristic of each state described as a propositional formula holds in the current state of the set; Because the semantic information provided by ontologies based on description logics, this Class methods can only put each concept in description logic as an atomic proposition to include the relationship between modeling concepts for the existence of atomic propositions the relationship between the implication (*e.g.,* literature [13] using Horn sentence rules to characterize the equivalence relationship exists between each semantic concept obviously). This approach has a high solving efficiency. But the limitation is unable to make full use of all kinds of ontology in semantic association.

The semantic Web technology can play a function and it needs to do: First, from the ontology to extract all contained semantic concept relationship that includes the relation is not necessarily in the service composition process all departments used.

Method for program synthesis and other software engineering are studied to apply in the semantic Web service composition. The literature [14] OWL - S (previously referred to DAML_S) semantic Web services modeling depicts the grounds of Propositional Linear Logic named LL axiom, will be represented by a LL user demand modeling theorem, and then convert the Web service composition problem into linear logic problem of proving theorems in LL, which in the process of modeling the main test. Considering the input and output parameters of semantic Web service: each semantic concept parameters cited as one of the original linear logic LL proposition, and the inclusion relation exists between the original modeling concepts proposition and the implication relation. The method based on the workflow of the party method has the same limitations: it can't make full use of various body semantic relations, and the need for prior reasoning. Aiming at these limitations, this paper in research proposed the use of first-order linear logic for semantic Web services into the prospect for modeling, but there is no related papers published.

Method of action theory and intelligent planning and other artificial intelligence for semantic Web service composition provides appropriate theories and methods. The literature [15] application of the situation calculus of OWL - S / DAML_S characterizations of the atom web service modeling, where the Web service's input and the former conditions for the situation calculus modeling precondition action axiom, will generate Web service output and execution of Web services and effect after construction die for the situation calculus of successor state axioms. Based on the work of the literature on [16], further application of extended Golog language on the O IV I, contains no control structure of Split and Split+ S / DAML-S combination of Web services Join to describe Web service, which will combine problem into a programming problem of Golog language literature support [17]. Atomic Web Services Modeling for planning system in SHOP2 operation, controlling structure of Split and Split+Join will not contain Web service composition modeling method for SHOP2, and with the SHOP2 implementation of semantic web service composition.

The document [18] application thing calculus for modeling OWL S characterizations of the Web service, the Web service composition problem into the event calculus planning based on the question. The question, then with the help of the prover problem solving abductive theorem is given. Providing a logical basis for appropriate action theory for semantic web services, but in general, the work still has two limitations. First of all, SHOP2 programming tool adopts closed world in planning process circle hypothesis, cannot reflect the description logic used in the open world assumption advantage; secondly, the situation calculus, the event calculus action theory based on the predicate logic of first order or higher order logic, have little ability to express strong; but at the same time, because of the first-order predicate logic could eventually lead to service composition algorithms cannot be terminated.

The literature in the description logic [19] proposed the establishment above the action subsequently, the literature theory [20 21] will apply the theory to the semantic action Semantic Web service modeling and reasoning and on this basis Web service composition. This action theory is used in description logic non circulation ring of TBox on static domain knowledge by description logic description; The ABox must meet before executing the action on the state of the world, condition and action in the implementation of the uncertainty component.

6. Conclusions

According to the experiment and contrast analysis, the algorithm in the paper has some advantages: the extensibility is good; the successful rate of assembling is high; when the quality of service component assembling is ensured, the efficiency of algorithm is high.

Acknowledgments

I would like to acknowledge and extend my heartfelt gratitude to associate Professor Yang Yu, for his vital encouragement, generous assistance and invaluable advice, all of which have been of inestimable worth to the completion of my thesis.

References

[1] O. Zimmermann, P. Krogdahl and C. Gee, "Elements of Service-oriented Analysis and Design," IBM: Technical Report, **(2004)**.
[2] M. Burstein and C. Bussler, "A Semantic Web Services Architecture," IEEE Internet Computing, vol. 9, **(2005)** May, pp. 72-81.
[3] C. Ardagna, E. Damiani and P. Samarati, "A Web Service Architecture for Enforcing Access Control Policies," Electronic Notes in the Oretical Computer Science, vol. 142, **(2006)** January, pp. 47-62.
[4] B. Benatallah, Q. Sheng and M. Dumas, "The Self-Serv Environment for Web Services Composition," IEEE Internet Computing, vol. 7, **(2003)** January, pp. 40-48.
[5] A. Sheila, Mellraith and L. Davidmartin, "Bring Semantics to Web Services," IEEE Intelligent System, vol.102, **(2009)** January, pp. 90-93.
[6] X. Jianjn, Z. Qian and i. Juanzi, "Modeling and Implementation of Unified Semantic Web Platform," Web Intelligence, vol. 33, **(2004)** July, pp. 603-606.
[7] A. Madche and V. Zacharias, "Clustering ontology-based metadata in the semantic web," In Proceedings of 6th European Conference on Principles and Practice of Knowledge Discovery in Database (PKDD), **(2002)**, pp. 348-360.
[8] A. Budanitsky and G. Hirst, "Evaluating wordnet-based measures of lexical semantic relatedness," Computational Linguistics, vol. 32, **(2006)** January, pp. 13-47.
[9] F. Giunchiglia, M. Yatskevich and P . Shvaiko, "Semantic Matching ∶ A1gorithms and Implementation," Journal on Data Semantics IX, Springer Berlin / Heidelberg, **(2007)**, pp. 1-38.
[10] M. Paolucci and T. Kawanmura, "Semantic Matching of Web Services Capabilities," In Proc the International Semantic Web Conference, Sardinia. Italy, **(2002)**, pp. 523-531.
[11] R. Ponneknatis and A. Fox, "A Developer Toolkit for Web Service Composition," In Proceedings of International World Wide Web Conference, Honolulu. Hawaii. USA, vol. 8, **(2002)**, pp. 83-107.
[12] Z. Stojanovic, "A Method for Component-based and Service-oriented Software Systems Engineering," Delft University of Technology, vol. 10, **(2005)** May, pp. 457-460.
[13] T. Xian-Fei, J. Chang-Jun, D. Zhi-Jun and W. Cheng, "A Petri net-based semantic Web service automatic composition method", Journal of Software, vol. 18, no. 12, **(2007)**, pp. 2991-3000.
[14] R. Jing-Hai, P. Kungas and M. Matskin, "Composition of semantic Web services using linear logic theorem proving", Information Systems, vol. 31, no. 4-5, **(2006)**, pp. 340-3.
[15] S. Narayanan and S. A. McIlraith, "Simulation, verification and automated composition of Web services", Proceedings of the 11th International World Wide Web Conference. New York, **(2002)**, pp. 77-88.
[16] S. A. McIlraith and T. C. Son, "Adapting Golog for composition of semantic Web services", Fensel D et al eds, Proceeding of the 8th Intemational Conference on Principles of Knowledge Representation and Reasoning Massachusetts Morgan Kaufmann, **(2002)**, pp. 482-496.
[17] W. Dan, B. Parsia, E. Sirin, *et al.*, "Automating DAML-S web services composition using SHOP2", Proceedings of the 2nd International Semantic Web Conference Berlin Springe-Verlag, **(2003)**, pp. 195-210.
[18] E. K. Ozorhan, E. K. Kuban and N. K. Cicekli, "Automated composition of Web services with the abductive event calculus", Information Sciences, vol. 180, no. 19, **(2010)**, pp. 3589-3613.
[19] F. Baader, C. Lutz, M. Milicic, *et a1.*, "Integrating description logics and action formalisms ∶ First results", M. Veloso, S. Kambhampati, eds. Proceedings of the 12th National Conference on Artificial Intelligence Menlo Park AAAI Press/The MIT Press, **(2005)**, pp. 572-577.
[20] F. Baader, C. Lutz, M. Milicic, *et a1.*, "A description logic based approach to reasoning about Web services", Vasiliu L, ed. Proceedings of the WWW 2005 Workshop on Web Service Semantics Towards Dynamic Business Integration. Chiba ∶ ACM Press, **(2005)**, pp. 636-647.
[21] M. Milicic, "Complexity of planning inaction formalisms base on description logics", Dershowitz N, Voronkov A, eds. Proceedings of the 14th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning Berlin Springer-Verlag, **(2007)**, pp. 408-422.