SWSCF: A Semantic-based Web Service Composition Framework

Jianqiang Hu
State Key Laboratory of Software Engineering, Wuhan University, Hubei, China
Department of Computer Science & Technology, Tsinghua University, Beijing, China
jianqianghu@keg.cs.tsinghua.edu.cn

Juanzi Li
Department of Computer Science & Technology, Tsinghua University, Beijing, China
ljz@keg.cs.tsinghua.edu.cn

Zhiyong Zeng
Software Faculty, Fujian Normal University, Fuzhou, China
billzeng2005@msn.com

***Abstract*—Web service composition is gaining a considerable momentum as an approach to the effective integration of distributed, heterogeneous, and autonomous application. Current composed service processes are mostly generated in the syntactic level and in a manual way. The limitation of the approach is that it does not allow business to dynamic change partners and services. In order to avoid this disadvantage, it is critical to map complex business from abstract process to executable flows according to application domain semantics and dynamic business requirements. In this paper, we first present a semantic-based Web service composition framework called SWSCF based on both semantic process recommendation and semantic matchmaking. Then, we analyze some key technologies including: (1) Introducing hierarchical activity mechanism for dynamic decomposition of business requirements to find a suitable semantic process template; (2) Searching and selecting service chain to match a semantic activity template which is used to specify an abstract activity; (3) Devising heterogeneous message transforming mechanism to eliminate the incompatible message types during generating data flow of an executable process. Furthermore, by comparison with some service composition frameworks, we prove SWSCF to be more flexible and effective.**

***Index Terms*—Hierarchical activity mechanism; searching and selecting service chain; heterogeneous message transforming mechanism; semantic-based Web service composition framework**

I. INTRODUCTION

Web services and related technologies promise to facilitate execution B2B ecommerce by integrating business application across networks like the Internet. In particular, the composition of Web services is gaining a considerable momentum as an approach to the effective integration of distributed, heterogeneous, and autonomous application. The key technologies in Web service composition include three portions: designing process, binding activities with concrete Web services and assigning data flow in the process. Current service processes are mostly generated in the syntactic level and in a manual way. The limitation of such a rigid approach is that is does not allow business to dynamically change partners and services [1]. Semantic Web services composition facilitates automated composition using Semantic Web technologies. Some semantic-automatic service composition frameworks such as presented in [1][2][3] are reasonable composition solutions, which allow users to use semantic process templates to capture the semantic requirements of the process, form executable processes according to the semantics of the activities in the templates, and recommend to users the process selected from a process repository.

Some problems still exist in current semantic-automatic service composition frameworks. In particular, application domain semantics and dynamic business requirements should be considered when complex business process is mapped from abstract process to executable flows. The user may be confused by the following questions:

1. How to reconstruct an abstract process with recommended semantic process in order to achieve flexible adaptation to the dynamic changes in business requirement?

In most of service composition frameworks, an abstract process, composed of several activities of specific functions linked with control flow, is designed by user. It can also been retrieved from the process repository by matching the business requirements with the process description. In the former situation, it needs that the user has to determine the function clarity of the activities. The user could not make use of good semantic process (e.g., semantic process templates) to construct an abstract process. In latter situation, it needs that the business requirement must conform exactly to the capability of the semantic process selected from process repository. Since the process cannot be divided into parts, the user has to overthrow overall abstract process even if a few changes take place in business requirement and re-design the whole process from the scratch.

2. How to search and select suitable services for binding and capturing semantic capabilities of each activity of an abstract process?

It is a challenging problem in searching and selecting the concrete Web services that not only capture the semantic capabilities of each activity but also best fit the overall composed business process. In most of composition frameworks, each activity is often specified by using Web service implementation or Web service interface. Semantic information may be ignored and thus it is not good for automatic Web services composition. Moreover, quality constraints are assigned to the overall process rather than to individual activity. Some quality criteria are common but not related to application domain.

3. How to establish explicit and compatible data flow linking between output parameters of one service to the input parameters of the other after obtaining control flow requirements?

Most of semantic-automatic service composition frameworks desire of coincident message types when establishing explicit data flow linking from the output message of one service to the input one another. Errors are often occurred when two message types are subtly heterogeneous, although both represent the same or similar semantics. The solutions in such frameworks have to give up alterative services to keep the process executing normally, which causes some more suitable concrete services are slipped off just because of the heterogeneity.

In order to avoid the above disadvantages, we present the solutions in SWSCF, Semantic-based Web Service Composition Framework. Some key features of SWSCF are: (1) Hierarchical activity mechanism is exploited to control business activity granularity which helps to select recommended semantic process. (2) Defining a semantic activity template as an atomic to capture semantics of an activity of service process, and searching and selecting a service chain to match a semantic activity template. (3) Heterogeneous message transforming mechanism is responsible for avoiding incompatible message types during generating data flow of an executable process.

The remainder of this paper is organized as follows: Section II provides an over-view of SWSCF. Section III describes the hierarchical activity mechanism. Section IV defines the service chain and presents the service chain searching algorithm. Section V presents the heterogeneous message transformation mechanism. Section VI compares SWSCF with current service composition frameworks and Section VII draws some conclusions.

II. SEMATNIC-BASED WEB SERVICE COMPOSITION FRAMEWORK

SEWSIP (Semantic Enable Web Service Integration Platform) project, being developed at KEG Lab in Tsinghua University, aims to provide a platform integrating Web services based on Semantic Web. Semantic-based Web Service Composition Framework (abbr. to SWSCF) is the core component of SEWSIP [4].

As shown in Figure 1, the user can edit process or accept the recommended one from process repository in the SWT-based graphics user interface (called Service Process Editor) after business requirement has been decomposed by Hierarchical Activity Component. Hierarchical activity mechanism is exploited in Hierarchical Activity Component to control the activity clarity. It considers the business requirement as a root activity and allows the user to divide it into several child activities linked with the construct of flow according to the concept hierarchy of business requirement and domain ontology. Such dividing action can be applied in every child activity and finally generate an Action Tree with coarse-grained activities as upper nodes.

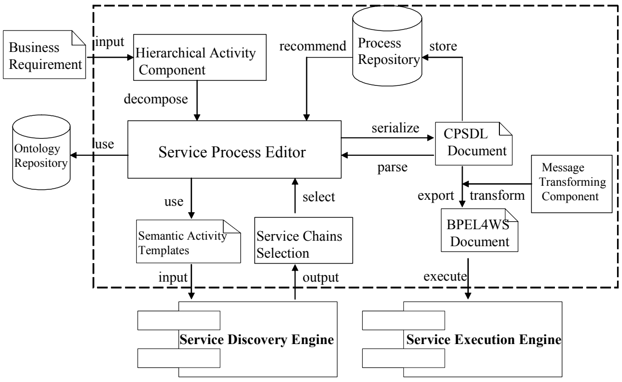

Figure 1. Semantic-based Web service composition framework

For each activity in the business requirement, Service Process Editor uses the semantic activity template to capture its semantic characteristics. Descriptions of semantic-annotated business requirement are constructed as a set of semantic activity templates, which send to the Service Discovery Engine as service query. Service chain, composed by one or more services, is treated as an atomic that might conform to a semantic activity template. Thus service chains be discovered and ranked by Service Discovery Engine and Service Chains Selection component. The user binds each activity with the best service chain and then constructs an abstract process. The finished service process can be serialized in CPSDL [5] (Composition Process Semantic-based Description Language) format and then be saved in the process repository for sharing and reusing latter to other process designers. In fact, CPSDL acts as the intermediates to serialize and conserve the designed process in the process repository. When the user needs to validate the process, he can convert its CPSDL file to BPEL4WS [6] and then deploy and execute it in Service Execution Engine, where Message Transforming Component is responsible for avoiding incompatible message types during generating data flow of an executable process. Aided by heterogeneous message transformation, Message Transforming Component gives the opportunity to more services to be bound with an activity even they have incompatible message types.

Deserving to be mentioned, the hierarchical activity mechanism, services chain searching and selection, and heterogeneous message transformation mechanism are added to make SWSCF more powerful and flexible than other semantic-automatic service frameworks.

III. HIERARCHICAL ACTIVITY MECHANSIM

In order to achieve flexible adaptation to the dynamic changes in business requirements, SWSCF uses the semantic process templates (serialized in CPSDL in Process Repository) to capture the semantic requirement of the business process similar to Meteor-S [1]. The semantic process template can act as a configurable module for common business processes maintaining the semantics of the participating activities, control flow, conditional branches and exposing it in an acceptable interface. However, in practice, it is very difficult to determine whether or not to find a suitable semantic process template from Process Repository.

Hierarchical activity mechanism provides the chance to decompose business requirement in a top-down manner and generate hierarchical business activities with different functional clarities. It helps the process designer to control the activities' function clarity in the business process. By this way, the business process can be divided into more thin-grained activities, and so that the process designer can make the best of the recommended process to construct an abstract process.

Definition 1 (**Hierarchical activities model**) Let BA be a set of the business activities, BA has the following characteristics: (1) If one business activity is root activity iff the value of its isRoot attribute is True; (2) Any business activity has parent activities except for the root activity; (3) There are not any circulating references in the set of the business activities.

Hierarchical activities model solves the above problem by constructing activity tree in terms of activity requirements' hierarchy. At first, the expected process is thought as a top activity. The process designer can either match the top activity in the process repository in terms of its semantic-description or discover and select suitable service chain (See Section IV) to bind with this activity. If both are failed, it means neither a similar existing process (i.e. semantic process template) can be found nor satisfied service chain can be bound with this top activity. Then, the process designer divides the top activity into several sub activities linked with the construct of flow by resolving the function requirement of the top activity as shown in Figure 2. Dividing action continues until (a) the function requirement of each leaf activity can be resolved; (b) the process designer construct the activity tree enough for him to generate the final abstract process; (c) the process designer aborts the process construction.

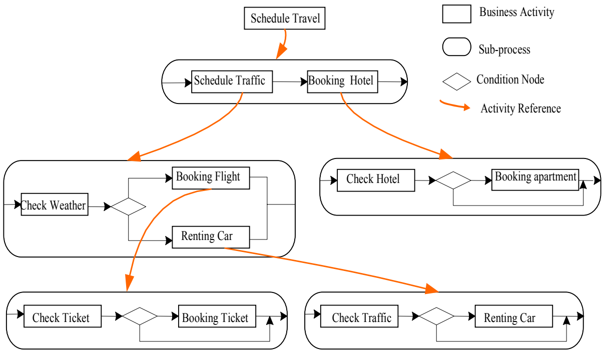

Figure 2. One example of hierarchical decomposition

This mechanism brings the following merits to SWSCF framework:

- Process designer can choose to employ those concrete services with suitable clarity because activities with various clarities are all listed in the activity tree.
- Process can be adjusted dynamically by the process designer since he can know immediately about which activity can be well bound. Furthermore, the process designer is easy to construct the process if some changes take place in business requirements.
- Process can be aborted in time when the process designer finds that it is impossible to bind any concrete services for a key activity nor can this activity be divided any more.

IV. SERVICE CHAINS SEARCH AND SELECTION

In order to capture the semantic of application domain, the requirements for an activity should be given using its semantic characteristics. We specify each activity of an abstract process using a semantic activity template, i.e., the activity requirements are given as the semantic of the inputs/outputs/preconditions/effects (IOPEs). The functional semantics (IOPEs) of an activity are represented using ontological concepts. Those suitable services that might conform to these semantic characteristics should be discovered and ranked. Unfortunately in some cases, single service that matches these semantic characteristics could not be precisely found. Several services should be composed to a service chain to meet the semantic requirement of an activity. Besides, with the ever increasing number of services similar in functionality being made available on the Internet, quality requirements should be considered to rank the resulting service chains.

A. Definition of service chain

Semantic matchmaking is the most critical step in the process of constructing service chains. Semantic characteristics (in terms of IOPEs) should be taken into consideration. Let one service s be a triple: $s=\{s.In, s.Out, s.Pre, s.Effect\}$, where s.In, $s.Out$, $s.Pre$ and $s.Effect$ are the semantic set of inputs/outputs /preconditions/effects, respectively. Let A and A' be the corresponding semantic set of Web service s and s', for example, $A=s.In$ and $A'=s'.In$, and then there are three types of matches as follows [7]:

Definition 2 If A and A' are equivalent, either equal literally, or equal by renaming the variables, or equal logically obtained by logic inference; formally, *Exact*(A, A'). If A is sub-concept of A'; formally, *PlugIn*(A, A'). Otherwise, the match of A and A' is fail.

Exact(A, A') match and *PlugIn*(A, A') match depend on logical inference which can gain high precision. Assume "<•,•>" is the match operation, we can define:

$$< A, A' >= \begin{cases} True \quad if\ Exact(A, A') \quad or \quad PlugIn(A, A') \\ \\ False \qquad\qquad others \end{cases} . \qquad (1)$$

Definition 3 (**Service chain**) Let s_1, s_2, …,s_n be n Web services. "$\bigcup$" is the union operation of set, and "<•,•>" be the match operation. If

$$<s_i.Out,\ s_{i+1}.In >=Ture \qquad \forall\, i \in \{1,2,...,\ i-1\} \quad (2)$$

$$< ((\bigcup_{j=1}^{i} s_j.Effect) \bigcup s_i.\Pr e), s_{i+1}.\Pr e > =Ture$$

$$\forall\, i \in \{1,\ 2,\ \dots,\ n\text{-}1\};\ j \in \{1,2,\dots,i\} \qquad (3)$$

Then s1, s2, …, sn can be composed be a service chain.

For example, Suppose

s_1: $s_1.In=\{In_1,\ In_2,\ In_3\}$, $s_1.Out=\{Out_1,\ Out_2,\ Out_3\}$, $s_1.Pre=\{Pre_1\}$, $s_1.Effect=\{E_1\}$; (4)

s_2: $s_2.In=\{Out_1,\ Out_2\}$, $s_2.Out=\{Out_4\}$, $s_2.Pre =\{E_1\}$, $s_2.Effect=\{E_2\}$; (5)

s_3: $s_3.In=\{Out_4\}$, $s_3.Out=\{Out_5\}$, $s_3.Pre=\{Pre_1,\ Pre_2\}$, $s_3.Effect=\{E_3\}$. (6)

By Definition 3, we can easily deduce that s_1, s_2 and s_3 can be composed a service chain.

Service chain may convey the directed sequences of multiple services aggregation in a semantic activity template. Moreover, it constitutes a reasonable foundation of the purpose for automatic service composition.

B. Service chain searching algorithm

Figure 3 presents the algorithm for searching service chain in Service Discovery Engine which hosts services with semantic descriptions. In fact, there may be multiple service chains that match a semantic activity template (i.e., *SAT*={*SAT.In*, *SAT.Out*, *SAT.Pre*, *SAT.Effect*}) in Service Discovery Engine. Multiple service chains, including a common source service, can be merged into one DAG (Directed Acyclic Graph) structure diagram including directed trees. The question of searching service chain can be formulated as recursive constructing a direct tree from Service Discovery Engine.

In order to control the time complexity, we use parameter n to restrict the degree of recursion and parameter serviceSet0 to avoid the acyclic of DAG diagram. At First, the matches, which are based on *SAT.In* and *SAT.Pre*, act as filters that eliminate the obvious non-matched services. Then, whether or not both <*s.Out*, *SAT.Out*> and <*.Effect*, *SAT.Effect* > are True is key to the success in constructing a direct service. If <*s.Out*, *STA.Out*> or <*Effect*, *STA.Effect*> is False, the algorithm will demonstrate recursion until *n* descends to 0. Finally, the algorithm returns the list of directed trees which are easily to be transformed to service chains.

```
TreeList SearchServiceTrees (serviceSet, SAT, serviceSet0, n)
  //serviceSet is the set of all services in registry,
    SAT={SAT.In, SAT.Out, SAT.Pre, SAT.Effect} is
    the semantic activity template, serviceSet0 is the
    matched services set, n is the searching degree
{  serviceInSet←getServiceIn(serviceSet, STA.In,
                                    serviceSet0)
  //Get the set of services matched STA.In from serviceSet-serviceSet0
  serviceInPreList←getServiceInPre(serviceInSet,
                                           STA.Pre)
  //Get the set of services matched STA.Pre from servicInSet
  TreeList←new TreeList;
  For service s in serviceInPreList do
  {  tree←new Tree(s)
     //Creat a directed tree and its Root is s
     Effect←CombineEffect(s.Effect, STA.Pre)
     //Get s.Effect ∪ STA.Pre
     If  <s.Out, SA.out>=Ture and <Effect,
           STA.Effect>=True
       { tree.add(s);
         TreeList.add (tree)}
     Else If (n>1)
       { serviceSet0←serviceSet0 ∪ {s}
         STA←{s.Out, STA.Out, Φ, effect}
         TreeList←SearchServiceTrees(serviceSet,
                    STA, serviceSet0, n-1)
         tree.add(TreeList)
         }
  }
  Return TreeList;
  }
```

Figure 3. Service Chain Searching Algorithm

C. Quality driven service chain selection algorithm

In order to enable quality-driven service chain selection, a quality-based service chain selection model is presented. This model supports customizing criteria related to application domain dynamically. It computes the weight of each evaluation criteria (for example, Qos criteria) by formulating as Multiple Attribute Decision Making (MADM) problem.

Definition 4 (**Quality-based Service chain selection model**) $DQos=\{\ A\ ,\ Dm\ ,\ CS\ \}$, where, (1) $A=[a_{ij}]_{n\times m}$, represents the decision matrix of evaluation criteria and their formulas, $a_{ij}=g(q_i, s_j)$ is a numeric value where $s_j \in S$ at $q_i \in Qos$, here Qos model $Qos=\{\ q_1,\ q_2,\dots,\ q_m\ \}$ represents the set of Qos criteria related to application domain, and $S=\{\ s_1,\ s_2,\ \dots,\ s_n\}$ represents the service chain set of specific s in service chain providing the information of Qos criteria. (2) $Dm = \{dm_1, dm_2, ..., dm_u\}$, represents the set of decision modes which are used for active computing weight of Qos criteria according to domain knowledge. (3) $CS = \{cs_1, cs_2, ..., cs_v\}$, represents the set of constraints on decision criteria. For example, users give the threshold point on different Qos criteria.

According to the approaches to determining weights, the modes for selecting service chain are summarized as four types: (1) *subjective weight mode*, (2) *single weight mode*, (3) *objective weight model*, (4) *subjective-objective weight mode.* Users can apply one of them to select appropriate service chains from a sorted set (abbreviated as $sort(s)$). In the followings, we will give the definitions of $sort(s)$ according to these four decision modes respectively.

(1) *Subjective Weight Mode*

Suppose the weights are determined according to the relative preferences of users, then service broker selects Web services with optimal $w_j^* \in w = \{ \sum_{j=1}^{n} w_j = 1$, $w_j \geq 0$,j=1, 2, …, n}, denotes as:

Definition 5 Let w_j be a weight reflecting the relative importance of the values of criterion q_j, where w_j^* is determined by Weighted Least Square Method [8] (Equation (7)). Let a user give his pairwise comparison matrix $D = [d_{ij}]_{n \times n}$ on the criteria set Qos according to his preference, with elements of matrix D satisfying Equation (8), where d_{kj} denotes the relative weight of the criterion q_k with respect to the criterion q_j , then $sort(s)$ is the sorted set of Web services in a descending (" $\rhd$ ") order, i.e., $s_i \rhd s_j$ if and only if $B_i w^* \geq B_j w^*$, where $B_i = (b_{i1}, b_{i2}, ..., b_{in})$, $w^* = (w_1^*, w_2^*, ..., w_n^*)'$, $s_i, s_j \in sort(s)$, *i, j*=1, 2, …, *m*.

$$\begin{cases} \min f_1 = \sum_{k=1}^{n} \sum_{j=1}^{n} (d_{kj} w_j - w_k)^2 \\ subject \quad to \quad \sum_{j=1}^{n} w_j = 1 \\ w_j \geq 0 \quad j = 1, 2, ..., n \end{cases} \tag{7}$$

$$d_{kj} > 0 , \; d_{jk} = 1 / d_{kj} , \; d_{kk} = 1 \quad i, k=1,2,\dots,n \tag{8}$$

(2) *Single Weight Mode*

It is a special case of subjective weight model. Suppose the most criterion q_k ($w_k^* = 1$) is given by a user, then the service broker selects Web services from $sort(s)$, which is sorted in a descending ("$\rhd$") order, i.e., $s_i \rhd s_j$ if and only if $b_{ik} \geq b_{jk}$, $s_i, s_j \in sort(s)$, *i, j =1, 2, ..., m*.

- *Objective Weight Mode*

Suppose the weights are determined by solving mathematical models automatically without any users' preference, including the entropy method [9], and multiple objective programming, etc., then service broker selects Web services with optimal $w_j^* \in w = \{ \sum_{j=1}^{n} w_j = 1, \; w_j \geq 0$,j=1,2,…,n}, denotes as:

Definition 6 Let w_j be a weight reflecting the relative importance of the values of criterion q_j, where w_j^* can be obtained by solving Equation (9), where $b_j^* = \max \{b_{1j}, b_{2j}, \dots, b_{mj}\}$ is the 'ideal' value of the criterion q_j in decision matrix B , the objective function f_2 can be interpreted as minimization of the deviation between the 'ideal' value of alternatives and the ranking value of each alternative, then $sort(s)$ is the sorted set of Web services in a descending (" $\rhd$ ") order, i.e., $s_i \rhd s_j$ if and only if $B_i w^* \geq B_j w^*$, where $B_i = (b_{i1}, b_{i2}, ..., b_{in})$, $w^* = (w_1^*, w_2^*, ..., w_n^*)'$, $s_i, s_j \in sort(s)$, *i,j*=1, 2, …, *m*.

$$\begin{cases} \min f_2 = \sum_{i=1}^{m} \sum_{j=1}^{n} (b_j^* - b_{ij})^2 w_j^2 \\ subject \quad to \quad \sum_{j=1}^{n} w_j = 1 \\ w_j \geq 0 \quad j = 1, 2, ..., n \end{cases} \tag{9}$$

Either subjective weight mode or objective weight mode has its advantages and disadvantages. The weights determined by subjective weight mode reflect the subjective judgment of users, which makes the marks of alternatives of the MADM problem have more arbitrary factors. The objective weight mode determines weights through mathematical calculation, which neglects subjective judgment information of users. Since neither of two approaches is perfect, an integrated mode may be more desirable for the determination of criterion weights.

(4) *Subjective-objective Weight Mode*

Let the weights be determined by both subjective consideration and objective impact, then service broker selects Web services with optimal

$w_j^* \in w = \{\sum_{j=1}^{n} w_j = 1,\ w_j \geq 0$, j=1, 2, …, n}, denotes as:

Definition 7 Let w_j be a weight reflecting the relative importance of the values of criterion q_j, where w_j^* can be obtained by solving Equation (10), then $sort(s)$ is the sorted set of Web services in a descending ("$\triangleright$") order, i.e., $s_i \triangleright s_j$ if and only if $B_i w^* \geq B_j w^*$, where $B_i = (b_{i1}, b_{i2}, \ldots, b_{in})$, $w^* = (w_1^*, w_2^*, \ldots, w_n^*)'$, $s_i, s_j \in sort(s)$, i, j=1, 2, …, m.

$$\begin{cases} \min f_1 = \sum_{k=1}^{n} \sum_{j=1}^{n} (d_{kj} w_j - w_k)^2 \\ \min f_2 = \sum_{i=1}^{m} \sum_{j=1}^{n} (b_j^* - b_{ij})^2 w_j^2 \\ subject \quad to \quad \sum_{j=1}^{n} w_j = 1 \\ w_j \geq 0 \quad j = 1, 2, \ldots, n \end{cases} \tag{10}$$

In order to obtain $w_j^* \in w$ in model 10, a linear weighted summation method in multiple objective programming analyses is used to derive a solution to Equation (11)

$$\begin{cases} \min f_3 = \alpha \sum_{k=1}^{n} \sum_{j=1}^{n} (d_{kj} w_j - w_k)^2 + \beta \sum_{i=1}^{m} \sum_{j=1}^{n} (b_j^* - b_{ij})^2 w_j^2 \\ subject \quad to \quad \sum_{j=1}^{n} w_j = 1 \\ w_j \geq 0 \quad j = 1, 2, \ldots, n \end{cases} \tag{11}$$

where α , β denotes the relative importance of the *subjective weight* mode and the *objective weight* mode respectively [11], α , β has

$$\alpha + \beta = 1,\ 0 < \alpha, \beta < 1 \tag{12}$$

To solve Equation (11), ignore the nonnegative constraint ($w_j \geq 0$, j=1,2,…, n) and set up the following function

$$L = \alpha \sum_{k=1}^{n} \sum_{j=1}^{n} (d_{kj} w_j - w_k)^2 + \beta \sum_{i=1}^{m} \sum_{j=1}^{n} (b_j^* - b_{ij})^2 w_j^2 + 2\lambda(\sum_{j=1}^{n} w_j - 1) \tag{13}$$

where λ is the Lagrange multiplier. Let $\frac{\partial L}{\partial w_h}$, h=1, 2, …, n., then

$$\alpha[\sum_{k=1}^{n} (d_{kh} w_h - w_k) d_{kh} - \sum_{j=1}^{n} (d_{hj} w_j - w_h)] + \beta \sum_{i=1}^{n} (b_h^* - b_{ih})^2 w_h + \lambda = 0 \tag{14}$$

Together with the constraint $\sum_{j=1}^{n} w_j = 1$ in the model 11, the n+1 equations can written in a compact form

$$\begin{bmatrix} Z & e \\ e^T & 0 \end{bmatrix} \begin{bmatrix} w \\ \lambda \end{bmatrix} = \begin{bmatrix} O \\ 1 \end{bmatrix} \tag{15}$$

where $w = (\ w_1\ ,\ w_2\ ,\ldots,\ w_n\ )'$, $Z = [z_{ij}]_{n \times n}$, $e = (1,\ 1, \ldots,\ 1\ )'$, $O = (0,\ 0, \ldots, 0\ )'$. The elements in matrix Z are

$$z_{ij} = -\alpha(d_{ij} + d_{ji}) \quad i \neq j; i, j = 1, 2, \ldots, n$$

$$z_{ii} = \alpha(\sum_{k=1}^{n} d_{ki}^2 + n - 2) + \beta \sum_{k=1}^{m} (b_i^* - b_{ki})^2 \quad i = 1, 2, \ldots, n$$

$$. \ \lambda = -\frac{1}{e' Z^{-1} e} 1$$

where w^* is the weight vector determined by the subjective-objective weight mode.

V. HETEROGENEOUS MESSAGE TRANSFORMING

After designing the abstract process (CPSDL document), an executable process should be generated. This involves finding services pertinent to each activity in the process, retrieving their WSDL [11] file, and extracting relevant information, establishing data flow of service process. After obtaining control flow requirements between two services from each activity in an abstract process, it is critical to establish explicit and compatible data flow linking between output parameters of one service to the input parameters of the other.

However, it is common that the type of the request message and the input type of the invoked service are not compatible. As shown in Figure 4, the request message comes from the output of Service A and contains all the information for invoking Service B, but with the different element order compared with the input type of Service B. Most of Web service composition systems avoid such situation by requiring absolute match between the two message types. In SWSCF, we present heterogeneous message transformation mechanism to solve this problem.

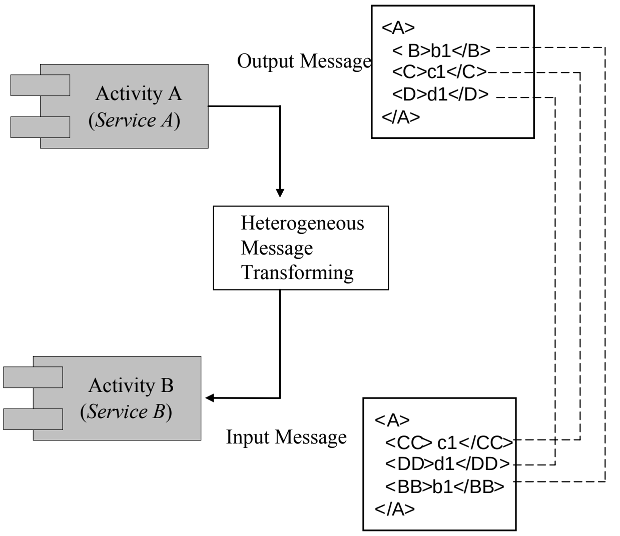

Figure 4. Heterogeneous message transforming mechanism

A XML transforming component [12] is developed in SWSCF for transformation between XML documents based on their conformed schemas. It is wrapped and deployed as a Web service in the Internet so that it can be invoked by other services. When transforming CPSDL [5] document to BPEL4WS, Message Transforming component is invoked to convert the output message of Service A into the input message of Service B such in Figure 4.

The component automatically generates mapping rules between the two message types. The rules can also be generated by users in the visual user interface of this component. These rules contain the mapping relationships from the elements in the request message to the ones in the input message, and they can be auto-generated by computing the similarity of elements based on the schema of the two messages. The similarity of elements is measured with Define 8. XSLT [13] file is generated from these rules and saved in the location with a distinct URL. When executing the service process in process engine, the transformation service accepts the request message and the URL of the generated XSLT from the process engine, and returns the message as response that can be accept as the input message of the next invoked service.

Definition 8 Let $name_1 =< w_{11}, w_{12}, ..., w_{1m} >$ and $name_2 =< w_{21}, w_{22}, ..., w_{1n} >$ be the names of element e_1 and e_2 respectively, where w_{1i} , w_{2j} ($i = 1,2,...,m$; $j = 1,2,...,n$) are words belongs to these two elements. Function (2) is used to calculate the similarity distance between $name_1$ and $name_2$, and Function (3) is used to calculate the word similar distance between word w_{1i} and w_{2j} , where s_{1i} , s_{2j} represents the semantic set of w_{1i} and w_{2j} respectively, s is the common parent Node of s_{1i} and s_{2j} [14].

$$NameSim(name_1, name_2) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} WordSim(w_{1i}, w_{2j})}{m \times n} \qquad (16)$$

$$WordSim(w_{1i}, w_{2j}) = \frac{2 \times \log p(s)}{\log p(s_{1i}) + \log p(s_{2j})} \qquad (17)$$

What's worth paying attention to is that heterogeneous message transformation mechanism does not depend on SWSCF. It can also be applied in other Web service composition systems so that more concrete services can be employed into the process although their message types are subtly different with what the user requires.

VI. COMPARISIONS WITH RELATED WORK

SWSCF is a framework that supports hierarchical activity mechanism, service chain searching and selection, and heterogeneous message transformation mechanism. To the best of our knowledge, these key technologies are not supported neither among traditional workflow management systems (such as MQ Workflow [15] and InConcert [16]), nor among recent Web services composition systems (such as Meteor-S [1], SAHARA [2] and SWORD [17]).

Meteor-S [1] is a framework aiming to support the complete lifecycle of semantic web processes, which is similar with SWSCF. It extends Web Service Industry Standard, WSDL and UDDI [18], bringing semantics into Web service description and discovery. Both Meteor-S and SWSCF share the common idea that using ontology in the activity description that allows much richer semantic description of activity requirements and more effective way of locating services to carry out the activities in the executable Web process. Nevertheless, Meteor-S also assumes that users have the capability to design the process containing the activities with precise clarity and that users can establish the explicit data flow linking between output parameters of one service to the input parameters of the other. Moreover, Meteor-S specifies each activity of an abstract process using Web service interface or a semantic activity template but single service matches its semantic characteristics often could not be precisely found.

SAHARA [2] is a comprehensive architecture for the creation, placement, and management of services for composition across independent providers. Its goal is to enable end-to-end service composition with desirable, predictable and enforceable properties spanning multiple potentially distrusting service providers. It classifies component services and composed services into a layered hierarchy from IP Network layer to End-User Application, which is not the same concept with the activity hierarchy in SWSCF.

SWORD [17], a set of tools for the composition of Web services, gives a rule-base composition model. In SWORD, a service is represented by a rule that expresses that given certain inputs, the service is capable of producing particular outputs. A rule-based expert system is then used to automatically determine whether a desired composite service can be realized using existing services.

It works well for information-providing services but it is not effective to generate business-to-business process or workflow process.

VII. CONCLUSIONS

In this paper, we propose a semantic-based Web services composition framework SWSCF which exhibits flexibility to integrate services in terms of application domain semantics and dynamic business requirements. The major contribution of the paper are as follows: (1) Introducing hierarchical activity mechanism to implement dynamic decomposition of business requirements which facilitates to find suitable semantic process templates; (2) Giving service chain searching algorithm to match a semantic activity template and domain-adaptive service chain selection model which guarantee the best assignation of an abstract activity; (3) Devising heterogeneous message transforming mechanism which eliminates the incompatible message types during generating data flow of an executable process. These good mechanism are been implemented in SEWSIP prototype.

ACKNOWLEDGMENT

This research project was supported by the National Basic Research Program of China (973) under Grant No. 2007CB310803, This work was supported by China Postdoctoral Science Foundation under the grant No.20070410061; Supported by State Key Laboratory of Software Engineering(SKLSE20080704).

REFERENCES

[1] K. Sivashanmugam, J. A. Miller, A. P. Sheth, K. Verma, "Framework for Semantic Web Process Composition," *Technical Report*, LSDIS Lab, Computer Science Dept., UGA, 2005.

[2] B. Raman, S. Agarwal, Y. Chen, and M. Caesar, etc., "The SAHARA Model for Service Composition across Multiple Providers", In *Proceedings of International Conference on Pervasive Computing*, LNCS 2414, Zürich: Springer-Verlag, pp.1-14, 2002.

[3] E. Sirin, J. Hendler, and B. Parsia, "Semi-automatic Composition of Web Services using Semantic Description," In *Proceedings of the ICEIS-2003 Workshop on "Web services: Modeling, Architecture and Infrastructure"*, Angers, France, 2003.

[4] W. Yang, J. Li, and K. Wang, "Interactive Service Composition in SEWSIP," In *Proceedings of the 2005 IEEE International Workshop on Service-oriented System Engineering (SOSE' 05)*, Beijing, China, 2005.

[5] W. Yang, J. Li, B. Xu and K. Wang, "Composition Process Semantic-based Description Language (CPSDL) specification," Available from: *http://keg.cs.tsinghua.edu.cn/sewsip/cpsdl/*

[6] IBM Corporation, "Business Process Execution Language for Web Services," Version 1.1. Available from: *ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf*, 2003.

[7] L. Li, L. Horrocks, "A Software Framework for matchmaking Based on Semantic Web Technology", In *Proceedings of World Wide Web (WWW' 03)*, Budapest, Hungary, 2003.

[8] A.T.W. Chu, R.E. Kalaba, K. Spingarn, "A Comparison of Two Methods for Determining the Weights of Belonging to Fuzzy Sets", *Journal of Optimization Theory and Application*, Vol. 27, pp.531-538, 1979.

[9] C. L. Hwang, and K. Yoon, "Multiple Attribute Decision Making", Springer-Verlag, Berlin Heidelberg New York, 1981.

[10] J. Ma, Z. Fan, L. Huang, "A Subjective and Objective Integrated Approach to Determine Attribute Weights", *European Journal of Operational Research*, Vol.112, No. 2, pp. 397-404.

[11] W3C, "Web Service Description Language (WSDL)," Version 1.1, (2001). Available from: *http://www.w3.org/TR/wsdl*

[12] W., J. Li, and K. Wang, "Relation-Tree based XML Transforming", *Chinese Journal of Computer Science*, pp.114-117, China, 2004.

[13] W3C, "XSL Transformations (XSLT)," Version 1.0, Available from: *http://www.w3.org/TR/xslt*, 1999.

[14] F. Christiane, editor, "WordNet: An Electronic Lexical Database," MIT Press, Cambridge, Massachusetts, USA pp.23-46, 1998.

[15] IBM Corporation, "MQ Services Workflow-Concepts and Architecture", GH12-6285, USA, 1998.

[16] R. Marshak, "In Concert Workflow," *Workgroup Computing Report*, Patricia Seybold Group, America, 1997.

[17] P. Shankar, F. Armando, "SWORD: A Developer Toolkit for Web Service Composition," In *Proceedings of the 11th International World Wide Web Conference*, Honolulu, 2002.

[18] UDDI org, "UDDI Spec TC," Version 3.0.2, 2004. Available from: *http://www.uddi.org/pubs/uddi_v3.htm*

Jianqiang Hu was born in Jinhua, China, in 1971. He received his Ph.D. degree in computer software and theory from National University of Defense Technology, China, in 2005, and his M.Sc. and B.Sc. degrees in computational mathematics from Fuzhou University and Zhejiang Normal University, China, in 1998 and 1995, respectively. He is currently a post-doc research fellow in Knowledge Engineering Group at the Department of Computer Science and Technology, Tsinghua University, China. His current research interests include distributed systems integration, service composition, semantic-based requirements engineering.

Juanzi Li was born in shanxi, China, in 1964. She received her Ph.D. degree from the Department Computer Science and Technology, Tsinghua University in 2000. She is now a professor of Tsinghua University. Her Current research interests focus on XML information processing, semantic web and web services.

Zhiyong Zeng was born in Longnan, Jiangxi, China, in 1966. He received the Ph.D degree in school of computer from Xidian University, Xi'an, in 2006. He is currently adjunct professor in faculty of software, Fujian normal university, China. His current research interests includes digital image processing and pattern recognition.