E-DPSO Algorithm Design and Demonstration about Dynamic Selection and Merging Process of ac-Service Flow

Xiaona Xia
School of Computer Science, Qufu Normal University, Rizhao Shandong, 276826, China
Email: xiaxn@sina.com

Baoxiang Cao and Jiguo Yu
School of Computer Science, Qufu Normal University, Rizhao Shandong, 276826, China
Email: bxcao@126.com, jiguoyu@sina.com

***Abstract*— Based on the research and development about Web service, the business platform topology of ac-service flow is deepened from architecture-centric perspective. For Web service's dynamic composition and self-adaptation meeting user requirement, architecture-centric merging among flows is put forward, in order to get this goal, optimizing previous flow granularities' selection and composition. To satisfy flexible building and mapping of flows, the evolution algorithm E-DPSO based on discrete particle Swarm is designed. With dynamic character of service composition and the integrity of flow sequence, ac-service flow's implementation strategy is given and finished demonstration. ac-service flow is viewed as the business organization clue, the platform framework structure of service implementation process is built for planning and decision-making.**

***Index Terms*— Web Service; ac-Service Flow; Dynamic Service Selection; Service Flow Merging; E-DPSO Algorithm; Discrete Particle Swam Algorithm; Architecture-centric**

I. WEB SERVICES AND SERVICE FLOW

As a new type of Web application mode, Web service has seen significant development in recent years, its high interoperability, cross-platform and loose coupling is the basic technology of software architecture. How to composite all kinds of Web services ,and finish selecting and merging among service flows adaptively, in order to form new topology structure, satisfy the timely requirement for different users and achieve architecture goal with value-added difficult service structure, which is the development application requirement and R&D difficulty of platform[1].

Under normal circumstances, we call this model as service flow, that consists of multiple service granularities, every service granularity includes its own goal, implementation sequence, participation roles and collaboration services etc, but not establish specific service's instances. In the Web environment, how to select the service instances satisfying architecture goal and implementation dynamically , form a executable composting service sequences, and adjust service flow topology adaptively, which is the key implementation issues of architecture-centric business[2, 3].

II. RELATED WORK

BPEL4WS（Business Process Execution Language for Web Services）[4] and EFLOW[5] are the static and dynamic technology supporting for Web service composition, but the dynamic semantic evolution of flow has not been effectively demonstrated, and this is an effective supporting factors of adaptive service composition. About standard workflow's implementation, state calculus and formal characterization of dynamic semantics is the strong supplement about effective logical flow. Early, researchers construct service composition process by Petri[6] theory, with the help of corresponding composition mode, service granularities are associated and extended by Petri. In addition, the use of process algebra[7], Pi calculus[8] and OWL-S[9] describe the mapping relationship between service granularities and model topology rules to better achieve unambiguous understanding of flow's services.

On the autonomous Web service composition, reference[10] proposed ontology-based intention-behavior-realization mechanism, but, there is no demonstration for adaptive requirement logic about composition topology supported by semantic and dynamic description. Reference [11] demonstrated realizing action, execution, projection and planning for service composition sequences, and discussed and evolved the topology process for open architecture, this dynamic description logic provided effective way to the modeling, reasoning, service discovery and adaptive scheduling[12]. Meanwhile, reference [13] extended this kind of dynamic description, and combining the

Corresponding author: Xiaona Xia

doi:10.4304/jcp.7.7.1631-1638

researching results of Pratt about dynamic proposition logic, it proposed the dynamic extended description logic-EDDX(X) from the implementation process and result of service composition actions, and completes semantic redefinition and dynamic composition.

These researching methods are based on two ideas: QoS-based semantic description and QoS-based on computing process. The computing process based on QoS attributes can achieve the whole controlling of architecture-centric goal for local composition services[14]. Based on composition service theory and composition service QoS model. Reference [14] provided a new genetic algorithm by using matrix codes, and selected services from optimal process perspective.

PSO(Particle Swarm Optimization) is an evolutionary algorithm, which retains and uses the position and velocity information during evolving process. Based on the granularities' intention-oriented characters about goal, PSO achieves aggregation from individual particles to granularity group, so it is called intention variant[15] and migration, it improves the convergence speed of individual particles. PSO concept is simple, easy implementation, with less relevant parameters, it can effectively optimize tasks[16]. Reference [3] designed three kinds of velocities calculating operators and a kind of position evolution equation, its implementation shows that this algorithm can better solve whole convergence with high performance.

III. DYNAMIC SERVICE'S SELECTION OF SERVICE FLOW

3.1 ac-Service Flow

On the ac-service flow, this thesis concerns the following points: architecture-centric constraint goal mechanism, service granularities, logical topology based on service composition, improving the adaptive ability for service composition, achieving the unification. For the completion of the selection, this thesis inherits the corresponding model logic of business, rules, roles and task composition modes in workflow, evolves the service flow topology, the generating process of ac-service flow is showed in Fig.1, and needs four parts:

(1) ac-Goal. It is the media constructor(architecture Decision-making mechanism) and combined of the three sub structures: Web Service Logic, Business Requirement and QoS Parameter. Architecture decision-making analyzes the above three part data and gets ac-Goal, this constructor is the organic thinking structure that service is the basic decision-making granularities.

(2) Service Topology Rules. Service topology is a set of underlying logic relationships, but also the support structure of decision-making basis. If there is topology to satisfy ac-Goal, and it is accepted candidate relationship. If not, architecture accepts decision-making mechanism to build new service granularities' applying, architecture-centric platform finishes generating new services' topology and expanding service topology sets.

(3) Sub-Goal Collection. The implementation process of ac-Goal needs to divided goal and forms sub-goal set, furthermore, architecture achieves the parallel sequences of ac-service flow about goal.

(4) Flow Topology Rules. This is a set of underlying topology rules, based on service composition topology and sub goals, it builds service flow to meet ac-Goal. Service flow topology includes two parts: service granularities and service relationships.

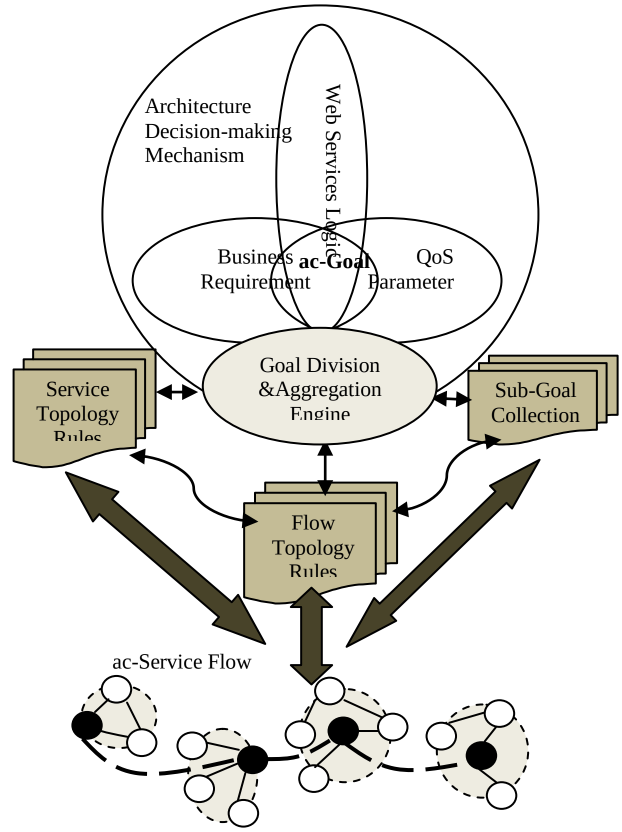

Figure 1. ac-Service Flow's Topology Sequence

3.2 The Decision-making and Selection of Service

About he constraint rules of requirements of Service Topology Rules, Sub-Goal Collection, Flow Topology Rules and QoS properties, about Service Topology Rules, Sub-Goal Collection and Flow Topology Rules, ac-Goal calculates every quality property with Formula 1. Before the calculation, it is needed to the data of quality properties. This thesis uses the method in reference[16], formula 1 can dispose negative attributes(greater value, lower quality, such as response time), formula 2 is used to dispose positive attributes(greater value, more high quality, such as availability). Among them, and stand for former and latter unified value of some quality properties, and stand for the maximum and minimum values of this quality properties respectively.

$$v_i = \begin{cases} \dfrac{q_{\max} - q_i}{q_{\max} - q_{\min}} & q_{\max} \neq q_{\min} \\ 1.当 q_{\max} = q_{\min} \end{cases} \quad \text{formula 1}$$

$$v_i = \begin{cases} \dfrac{q_i - q_{\max}}{q_{\max} - q_{\min}} & q_{\max} \neq q_{\min} \\ 1.当 q_{\max} = q_{\min} \end{cases} \quad \text{formula 2}$$

IV. ARCHITECTURE-CENTRIC MERGING LOGIC OF SERVICE FLOW

Architecture-centric goal achieves flow's monitoring and information's capturing, service flow is consistent with the traditional workflow structure, showed in Fig. 2.

The merging logic of service flow can be made from these four basic models, while QoS's calculation of merging is a QoS calculation method about participation flow, every sf-QoS of flow can be gotten based ib traditional service composition's QoS formula[17, 18], here is not repeated and sf-QoS of service flow is as the basic value to be calculated.

Sf-QoS can be reflected on functionality and non-functional properties about architecture-centric service platform, including:

（1）Response Time sf-T: it is the length of time that service flow completes requested services about sub-goals.

（2）Implementation Cost sf-C: It is the produced consumption of service flow driven sub-goals.

（3）Availability sf-A: It is the probability of successful integration of gotten service flow about sub-goals.

（4）Right sf-R: It is the whole evaluation formed by service flow's transactions in a certain period.

（5）Goal Matching sf-M: It is the matching degree of service flow's implementation goal about sub-goals, which determines the extent of the flow's deployment.

Supposing sf is the service flow after having merged, cs is the service flow involved in merging flow, QoS values of sf and cs are defined as:

sf-QoS={sf-T, sf-C, sf-A, sf-R, sf-M}

cs-QoS={cs-T, cs-C, cs-A, cs-R, cs-M}

Merging forms about different service flows is corresponding to different conversion processes.

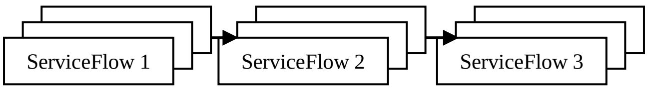

Fig 2. (1) Sequence Merging

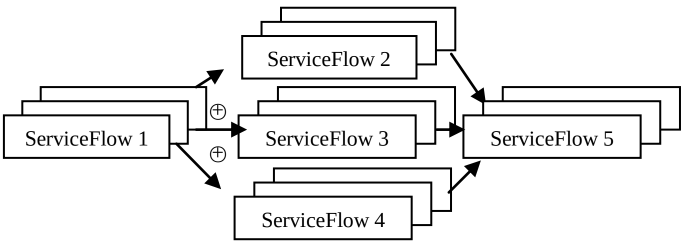

Fig. 2 (2) Selection Merging

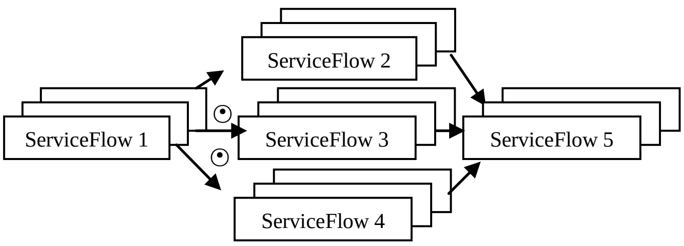

Fig.2 (3) Concurrence Merging

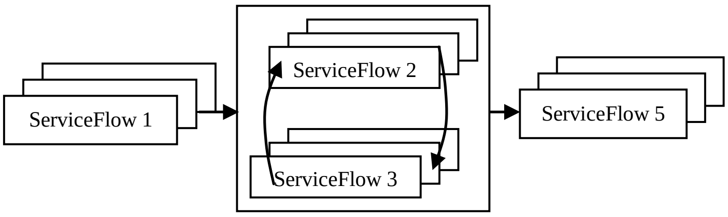

Fig.2 (4) Circle Merging

Figure 2. Basic Merging Style of Service Flow

TABLE 1 CONVERSION PROCESS OF SERVICE FLOW'S MERGING

Sequence Merging	Selection Merging
$sf-T=\sum_{i=1}^{n} cs-T_i$ $sf-C=\sum_{i=1}^{n} cs-C_i$ $sf-A=\prod_{i=1}^{n} cs-A_i$ $sf-R=\frac{\sum_{i=1}^{n} cs-R_i}{n}$ $sf-M=\prod_{i=1}^{n} cs-M_i$	$sf-T=\sum_{i=1}^{n} P_i \times cs-T_i$ $sf-C=\sum_{i=1}^{n} P_i \times cs-C_i$ $sf-A=\prod_{i=1}^{n} P_i \times cs-A_i$ $sf-R=\frac{\sum_{i=1}^{n} P_i \times cs-R_i}{n}$ $sf-M=\prod_{i=1}^{n} P_i \times cs-M_i$
Concurrence Merging	Circle Merging
$sf-T=Max\{cs-T_i\}$ $sf-C=\sum_{i-1}^{n} cs-C_i$ $sf-A=\prod_{i=1}^{n} cs-A_i$ $sf-R=\frac{\sum_{i=1}^{n} cs-R_i}{n}$ $sf-M=\prod_{i=1}^{n} cs-M_i$	$sf-T=k \times cs-T_i$ $sf-C=k \times cs-C_i$ $sf-A=\prod_{i=1}^{n} cs-A_i$ $sf-R=(cs-R_i)^k$ $sf-M=\prod_{i=1}^{n} cs-M_i$ $sf-M=\prod_{i=1}^{n} (cs-M_i)^k$

Note: P_i is the selected probability about branch in merging structure of service flow, is the circle scheduling numbers of participation service sub-flows.

According to different properties of merging and QoS property calculating process in TABLE 1, the comprehensive quality of service flow will be derived.

Dynamical scheduling algorithm of service flow's merging is described as:

DynamicServiceFlowMergeSchedule（）

Input: subServiceFlowList; ac-Goal; wholeServiceFlowTopology;

Input: ServiceCarrierList[][]//*One-dimensional* *address stands for its owner service sub-flows, two dimensional address stands for service's implementation loads.*

```
Output: CompositeServiceFlowTopology[];
        subServiceFlowRelationship[][];
Begin
(1)  sSFL_length=length(subServiceFlowList);
     //Get the number of service sub-flows.
(2)  wholeServiceFlowTopology=Topology(ac-Goal); //Get the whole flow's topology
     requirement by goal.
(3)  CompositeServiceFlowTopology[]=null;
(4)  subServiceFlowRelationship[]=null; //Step (2) and (3) initialize service flow's
     structure that is completed to merge.
(5)  j=1;
(6)  while i<= sSFL_length do
(7)   begin
(8)     If (Match(subServiceFlowList[i], ac-Goal));
     // Matching sub-flows with ac-Goal.
(9)      Begin
     CompositeServiceFlowTopology[j]=subServiceFlowList[i]; j+=1;
     subServiceFlowRelationship[i][j]=Topology(subServiceFlowList[i]);end
   //Topology() is the topology relationship linking attributes, it can get the
   topology sequence of ubServiceFlowList[i].
(10) i+=1;
(11) End
(12) WholeServiceFLow_QoS=null;
(13) m=1; n=1;
(14) while m<=i do
(15) while n<=j do
(16)  Begin
(17)   Learning(wholeServiceFlowTopology,
     subServiceFlowRelationship[i][j]);// Capturing the logical relationships of
     different sub-flows topology about whole flows.
(18)  Case(WholeServiceFlowTopology)
(19)   Case Sequence Merging  Calculate QoS
     WholeServiceFLow_QoS +=QoS break
(20)   Case Selection Merging Calculate OoS
     WholeServiceFLow_QoS +=QoS break
(21)   Case Concurrence Merging Calculate QoS
     WholeServiceFLow_QoS +=QoS break
(22) Case Circle Merging Calculate QoS
     WholeServiceFLow_QoS +=QoS break
(23) end
(24) Output(CompositeServiceFlowTopology[j])
(25) Output(subServiceFlowRelationship[i][j])
end
```

IV. ARCHITECTURE-CENTRIC MERGING LOGIC OF SERVICE FLOW

PSO (Particle Swarm Optimization)[19] is a population-based adaptive searching method proposed by Kennedy and Eberhart in 1995 [3]. Inspired by climate cluster activities of birds and fish, replaces the natural selection mechanism with organizing social behavior, collaboration among individual population achieves capturing optimal solution [20].

Reference[4] inherits and develops the QoS normalization method of reference [21], the optimal QoS calculation of flow is also a iterative QoS calculation results about optimal service composition, the value of its sector in [0, 1] are positive indicators（also with a negative index）of the QoS values.

Supposing the cs-QoS indicator of merging service flow i (assuming sf-QoS corresponds to m cs-QoS), the result after normalization is $cs-Q_i$, then fitness function of service flow is defined as $sf=\sum_{i=1}^{m} cs-w_i \times cs-Q_i$, $cs-w_i$ is the weight indicator corresponding to cs-QoS of i merging sub flows, and satisfying $\sum_{i=1}^{m} cs-w_i =1$.

5.1 Basic Operator of E-DPSO

Definition1 Optimal Replacement of Merging Flow

Sub flow cs_i ($1\leqslant i \leqslant m$, m is the number of sub flows that participates service flow' merging), $s-cs_{ij}$ is the selected service granularity of the pnd generation about cs_i , if the selected service granularity of cs_i is $s-cs_{ik}$ in $(p+1)$nd generation, then $s-cs_{ij} \rightarrow s-cs_{ik}$ is one replacement from pnd to $(p+1)$nd generation in service flow's selection and composition, and $s-cs_{ij}$ is named as replaced service granularity, $s-cs_{ik}$ is named as replacing service granularity, it will complete one optimal replacement process of service composition, denoted by $Service\_substitute(cs_i^{p+1}) = s-cs_{ik}$. From this definition, we are not difficult to draw, if $s-cs_{ik} \equiv s-cs_{ij}$ ($s-cs_{ik}$ and $s-cs_{ij}$ is completely equivalent), this sub flow implements empty replacement operation.

Similarly, here we define the sub flow's replacement operation of merging service flow sf . Supposing the ind merging sub goal is $sf-subGoal_i$ ($1\leqslant i \leqslant n$, n is the sub goals' number formed by sf), cs_{ij} is the selected merging sub flow of $sf-subGoal_i$ in q generation, if the selected merging sub flow of $sf-subGoal_i$ in $(q+1)$nd generation is cs_{ik} , this process $cs_{ij} \xrightarrow{sf} cs_{ik}$ is merging sub flows' replacement process from qnd to $(q+1)$nd generation about sf, cs_{ij} is called as the replaced merging sub flow, cs_{ik} is the replacing sub flow, denoted by $flow\_substitute(sf^{q+1}) = cs_{ik}$. Of course, the sub flow's replacement also exist empty operation.

Definition 2 Optimal Replacement Order of Merging flow

The implementation cluster composed of more service compositions or flow merging replacement is called as flow's replacement order, its elements' topology order of cluster have the actual significance about ac-Goal, during the replacement of every generation about participation granularities, velocity is one flow's replacement order.

Definition3 α Optimization Replacement of Merging Flow

α Optimization Replacement of merging sub flow cs_i ($1 \leqslant i \leqslant m$, m is the sub flows' number of participation service flows' merging) means to accept $sf - subGoal_i$'s adjustment, optimally select by α probability, and complete composition services' replacement. Such as, cs_i and optimal replacement $s-cs_{ij} \rightarrow s-cs_{ik}$, then α Optimization Replacement ($s-cs_{ij} \xrightarrow{sf-subGoal_i} s-cs_{ik}$) exists several optimization replacement forms:

if α =0, then service composition process of sub flow cs_i does not need service granularities' replacement, it meets for $Service\_substitute(cs_i) = s-cs_{ij}$. For merging service flow sf, $flow\_substitute(sf)\underline{\underline{sf-subGoal_i}}cs_{ij}$;

If α =1, cs_i will select service granularity $s-cs_{ik}$, then $Service\_substitute(cs_i) = s-cs_{ik}$. For merging service flow sf, $flow\_substitute(sf)\underline{\underline{sf-subGoal_i}}cs_{ik}$;

If 0< α <1, uniform distribution on a randomly generated number $sr \sim U$ (0, 1), if $sr \leqslant \alpha$,and $Service\_substitute(cs_i) = s-cs_{ik}$, $flow\_substitute(sf)\underline{\underline{sf-subGoal_i}}cs_{ik}$; otherwise, $Service\_substitute(cs_i) = s-cs_{ij}$,and sympathy $flow\_substitute(sf)\underline{\underline{sf-subGoal_i}}cs_{ij}$.

Definition 4 Optimal $sMax$ Operation of Merging Sub flow.

cs_i ($1 \leqslant i \leqslant m$, m is the sub flows' number of participation service flows' merging) has the following replacement operation with coefficient $\rho Service\_substitute(cs_i)$ and $\eta Service\_substitute(cs_i)$, $\oplus$ is the operator of $sMax$, then $sMax$ is expressed as:

$$sMax\ (\rho Service\_substitute(cs_i) \oplus \eta Service\_substitute(cs_i)) = \begin{cases} \rho Service\_substitute(cs_i), \rho \geq \eta \\ \eta Service\_substitute(cs_i), \rho < \eta \end{cases}$$

For the architecture platform, merging service flow is the uniform flow about ac-Goal, there is not $sMax$ operation.

Definition 5 Optimal $sRank$ Operation of Merging Sub Flow

cs_i ($1 \leqslant i \leqslant m$, m is the sub flows' number of participation service flows' merging) has the following replacement operation with coefficient $\rho Service\_substitute(cs_i)$ and $\eta Service\_substitute(cs_i)$, $\otimes$ is the operator of $sRank$, then $sRank$ is expressed as:

$$sRank\ (\rho Service\_substitute(cs_i) \otimes \eta Service\_substitute(cs_i)) = \begin{cases} \rho Service\_substitute(cs_i), \rho \geq \eta \\ \eta Service\_substitute(cs_i), \rho < \eta \end{cases}$$

$sRank$ belongs only to merging sub flow.

Definition 6 Optimal $sSeq$ Operation of Merging Sub Flow.

cs_i ($1 \leqslant i \leqslant m$, m is the sub flows' number of participation service flows' merging) has the following replacement operation with coefficient $\rho Service\_substitute(cs_i)$ and $\eta Service\_substitute(cs_i)$, $\odot$ is the operator of $sSeq$, then $sSeq$ is expressed as:

$$sSeq\ (\rho Service\_substitute(cs_i) \odot \eta Service\_substitute(cs_i)) = \begin{cases} \rho Service\_substitute(cs_i), \rho \geq \eta \\ \eta Service\_substitute(cs_i), \rho < \eta \end{cases}$$

$sSeq$ belongs only to merging sub flow.

5.2 Inertia Weight s^{ϖ} of E-DPSO

About the adjustment optimization of service composition and flows' merging, E-DPSO needs to give ϖ the corresponding analysis for service flow, here defines as s^{ϖ}, the main goal of s^{ϖ} dynamically adjusts its own value according to ac-Goal and the controlling condition of service flow's building, in order that the participation services can get the larger weight value when there is more difference between actual values and goal's realization. During implementing, service flow accepts the adjustment of ac-Goal, when actual weight values is closer to the goal, it may reduce

${}_s\varpi$ automatically, that can promote service granularities or merging sub service flow to complete determining the best location in a smaller scope. The specific analytical formula of ${}_s\varpi$ is:

$$ {}_s\varpi = \begin{cases} \dfrac{\sum|{}_s value_i - {}_s a_i|}{\sum|{}_s value_{i-\max} - {}_s a_i|} \times (cs-QoS, sf-QoS) + 0.8 \\ \dfrac{\sum|{}_s value_i - {}_s a_i|}{\sum|{}_s value_i - {}_s a_i|} \times (cs-QoS, sf-QoS) \le 0.6 \\ 1.4, Else \end{cases} $$

Formula (1)

In the analytical weight Formula (1) of E-DPSO, $cs-QoS$ of service composition and $sf-QoS$ of merging sub service flows complete convergence, QoS controlled by ac-Goal ensures ${}_s\varpi$ faster close to goal's adjustment, according to the different operation of service composition and flow merging, QoS values' replacing selection has two styles: $cs-QoS$ and $sf-QoS$. ${}_s value_i$ is gotten during current services' selection and sub flows' merging controlled by goal; ${}_s a_i$ is the controlling requirement value of ac-Goal by i order; ${}_s value_{i-\max}$ is the possible maximum value. The Definition of ${}_s\varpi$ ensures its value is close to the interval distribution of [0.8, 1.4].

5.3 Optimal Expectation Criteria of E-DPSO

The optimal expectation of E-DPSO: if the ratio p meeting for ac-Goal in architecture services has gotten the higher matching value, ac-Goal optimizes services group. The service flow i with nnd participation sub flows, if existing q sub flows ($0 \leqslant q \leqslant n$), every sub flow sf_d is empty replacement in q, $flow\_substitute(sf_{id}) \underline{\underline{ac-Goal}} sf-is_d$, $sf-is_d$ is a service instance selected from ac-Goal service topology sequences in random, when expected optimizing granularities of services group, optimal granularities of service flows complete optimal position's matching. Similarly, about sub service flows' merging, if sub service flow j with m participation granularities has s services ($0 \leqslant s \leqslant m$), every service cs_c in s is also empty replacement, then $service\_substitute(cs_{jc}) \underline{\underline{ac-Goal}} cs-is_c$.

p, q, s and the flow weight of ac-Goal directly affect the optimal result of E-DPSO. If the value of p, q and s is too small to confusedly select granularities of architecture, it is to reduce optimal selection of sub flows and sub flows' merging; While, the value of p, q and s is too large to need more long time to waiting for empty replacement for flow topology's building and sub services' composition, that can not effectively achieve expected results of algorithm.

VI. Optimal Strategy of ac-Service Flow based on E-DPSO

Combining sub service flows' selection and ac-service flows' merging, the optimal particle velocity's formula of E-DPSO is achieved. Based on the analysis result of the fifth part in this paper, we can get the three kinds of velocity formula about d dimension of ac-service flow, as Formula (2), (3) and (4):

$$ cs_i - v_d^{q+1} = {}_s\varpi_{cs_i - v_d^q} \oplus c_1 r_{1d}(P_{id} - x_{id}) \oplus c_2 r_{2d}(P_{jd} - x_{id}) $$

Formula (2)

$$ cs_i - v_d^{q+1} = {}_s\varpi_{cs_i - v_d^q} \otimes c_1 r_{1d}(P_{id} - x_{id}) \otimes c_2 r_{2d}(P_{jd} - x_{id}) $$

Formula (3)

$$ cs_i - v_d^{q+1} = {}_s\varpi_{cs_i - v_d^q} \odot c_1 r_{1d}(P_{id} - x_{id}) \odot c_2 r_{2d}(P_{jd} - x_{id}) $$

Formula (4)

c and r stand for the circle and the free particle radius of two particles in service flow. According to Formula (1), (2) and (3), the evolution formula from qnd generation to $(q+1)$nd about sub service flow cs_i:

$$ x_{id}^{q+1} \underline{\underline{ac-Goal}} flow\_substitute(cs_i - v_d^{q+1}) $$ Formula (5)

Ac-service flow's optimization algorithm based on E-DPSO is described as follows:

STEP1 Initialize sub flows' participation of ac-service flows, and form particle swarm, ac-Goal gives the initial topology position and any velocity value to every participation sub flow, and estimate $cs-QoS$ and $sf-QoS$;

STEP2 If the building requirement of ac-service flow is interrupted in advance, or ac-Goal captures historical service flow, then turn STEP 8; If ac-service flow is the implementing requirement, turn STEP 3;

STEP3 Based on $cs-QoS$ and $sf-QoS$, calculate and get ${}_s\varpi$;

STEP4 Estimate $c_1 r_{1d}$ and $c_2 r_{2d}$, sort ${}_s\varpi$, $c_1 r_{1d}$ and $c_2 r_{2d}$ in ascending order;

STEP5 For d replacement of sub service flow cs_i, based on Formula (2), (3) and (4), calculate $cs_i - v_d^{q+1}$;

STEP6 According to x_{id}^{q+1} of cs_i, ac-Goal matches and get the fitness about ac-service flow topology rules and building requirement, the fitness is a percentage, if less than 40%, the particle position is updated to position P_{jd}; if more than 60%, the particle is not replaced; If in the range [0.4, 0.6], ac-Goal selects particles and calculates fitness, iteratively, until calculates the corresponding sub flows of ac-Goal;

STEP7 For all service sets and sub flow sets meeting for expecting optimization rules, the service composition and sub flows′ merging about ac-Goal calculate the d replacement velocity of sub flows: $cs_i - v_d = flow\_substitute(x_{id}{}^{q+1})$, its topology position is $x_{id}^{q+1} = flow\_substitute(x_{id}^{q+1})$, the optimal position is $P_{id}\,\underline{\underline{ac-Goal}}x_{id}^{q+1}$, iteratively, to achieve the optimal topology of ac-service flow, turn STEP (2);

STEP8 If meeting for service flow's requirement and the optimal merging process of ac-Goal, then output ac-service flow, or suspend this flow's requirement. According to the service selection's rules of the fifth part, sub service flows are selected about particle swarm optimally, iteratively, improve sub flows' swarm, turn STEP 2.

We can see, the merging process of service flow requires two stages: one is particles' achievement of sub flows, its time complexity is $O(m\times n)$, $O(m)$ is the selection complexity of corresponding service particles, $O(n)$ is service composition's complexity of sub services. Another is the merging and building of ac-service flow, its time complexity is $O(I\times(k\times Q^2))$, $O(I)$ is the selection process of sub service flow, $O(k\times Q^2)$ is the time complexity about particles' replacement in sub flows and different particles' evolution generations, so the whole algorithm's time complexity is $O((m\times n)\times I\times(k\times Q^2))$, its value mainly depends particles' replacement among generations of sub flow and selection's complexity.

VII. Conclusion

This paper designs the adaptive platform that ac-service flow is as the business clue. To achieve this goal, service composition and sub service flow is as the researching points, QoS is protection condition for ensuring flows and business's optimization, the architecture-centric mechanism has been built that ac-Goal is central adjustment, its key point is to build dynamic optimization process for service composition and flow's merging.

The innovation of this paper reflects flow's flexible building and matching for the whole goal, evolving discrete particle swarm algorithm, abstracting service and sub service flow as particle swarm, based on ac-Goal, E-DPSO is proposed, the corresponding material and logical basic is defined and described, rules are demonstrated, and flow is built, the previous granularities' selection and composition has been analyzed. As the theoretical supporting, this paper analyzes the dynamical properties during service composition or service flows' merging, gives the architecture-centric implementation strategy of ac-service flow, and completes the relevant demonstration.

The future work: further optimize ac-service flow, improve service library rules of platform, the autonomous analysis logic of sub service flow, merging rule library and flexible capturing and randomly adaptive building about historical service flow topology.

Acknowledgment

This paper is supported by National Natural Science Foundation of China (No.60072014), Natural Science Foundation of Shandong Province of China (No.ZR2009GM009), the Key Science-Technology Development Project of Shandong Province of China(No. 2009GG10001014) and Promotional Foundation (2005BS01016) for Middle-aged or Young Scientists of Shandong Province, SRI of SPED(J07WH05), DRF and UF(3XJ200903, XJ0609)of QFNU.

References

[1] Zeng Liangzhao, Benatallah B, et al. QoS-aware middleware for Web services Composition[J]. IEEE Transn on Software Engineering, 2004, 30(5): 311-326

[2] Shalil M, Walker D W, Gray W A, A framework for automated service composition in service-oriented architectures[G]. LNCS 3053: Proc. of the ESWS 2004. Berlin: Springer, 2004: 269-283

[3] Fan Xiaoqin, Jiang Changjun, Fang Xianwen, et al. Dynamic Web Service Selection Based on Discrete Particle Swarm Opitimization[J]. Journal of Computer Research and Development. 2010, 47(1): 147-156 (In Chinese)

[4] Curbera F, Goland Y, Klein J, Leymann F, Roller D, Thatte S, Weerawarana S. Business process execution language for Web services version 1.1. IBM Document, 2002.http://www.ibm.com/developerworks/library/specification/ws-bpel/

[5] Casati F, Ilnicki S, Jin LJ, Krishnamoorthy V, Shan MC. Adaptive and dynamic service composition in eFlow. In: Benkt W, Lars B,eds. Proc. of the Int'l Conf. on Advanced Information Systems Engineering. LNCS 1789, Stockholm: Springer-Verlag, 2000.13−31.

[6] Hamadi R, Benatallah B. A Petri net-based model for Web service composition. Proc. of the 14th Australasian Database Conf. on Database Technologies, 2003,143(17):191−200.

[7] Tang Y, Chen L, He K, Jing N. SRN: An extended Petri-net-based workflow model for Web service composition. In: Proc. of the IEEE Int'l Conf. on Web Servies (ICWS 2004). San Diego: IEEE Computer Society, 2004. 591−599. http://doi.ieeecomputersociety.org/10.1109/ICWS.2004.1314786

[8] Liao J, Tan H, Liu JD. Describing and verifying Web service using pi-calculus. Chinese Journal of Computers, 2005,28(4):635−643 (in Chinese with English abstract).

[9] OWL-S Coalition. OWL-S 1.1 release. 2004. http://www.daml.org/services/owl-s/1.1/

[10] YE Rong-Hua, JIN Zhi, WANG Pu-Wei etc. Approach for Autonomous Web Service Aggregation Driven by Requirement. Journal of Software. Vol.21, No.6, June 2010: 1181-1195.

[11] Chang L, Lin F, Shi ZZ. A dynamic description logic for representation and reasoning about actions. In: Zhang ZL, Siekmann J, eds. Proc. of the 2nd Int'l Conf. on Knowledge Science, Engineering and Management. Berlin: Springer-Verlag, 2007. 115−127.

[12] Gu YL, Soutchanski M. Decidable reasoning in a modified situation calculus. In: Veloso M, ed. Proc. of the 20th Int'l Joint Conf. on Artificial Intelligence. Menlo Park: AAAI Press, 2007. 1891−1897.

[13] CHANG Liang, SHI Zhong-Zhi, CHEN Li-Min, NIU Wen-jia. Family of Extended Dynamic Description Logics. Journal of Software. 2010, 21(1): 1-13(In Chinese)
[14] Zhang Chengwen, Su Sen, Song Jingyu. Genetic algorithm on Web services selection supporting QoS[J].Journal of Computer, 2006, 29(7): 1029-1037(In Chinese)
[15] Zeng Jianchao, Jie Jing, Cui Zhihua. Particle Swarm Optimization[M]. Beijing: Science Press, 2004(In Chinese)
[16] Ran S. A model for Web sevices discovery with QoS. ACM SIGEcom Exchanges, 2003,4(1):1−10.
[17] Liu Shulei, Liu Yunxiang, Zhang Fan, et, al. A dynamic Web service selection algorithm with QoS global optimal in Web service composition[J]. Journal of Software, 2007, 18(3): 646-656.
[18] Canfora G, Di P M, Esposito R, et al. A framework for QoS-aware binding and re-binding of composite Web services[J]. The Journal of System and Software, 2008: 1754-1769.
[19] Eberhart R, Kennedy J. A new optimizer using particle swarm theory[C]. //Proc. of the 6th Int Symp on Micro Machine and Human Science. Piscataway: IEEE Service Center, 1995: 39-43.
[20] Banks A, Vincent J, Anyakoha C. A review of particle swarm optimization. Part I: Background and development[J]. Natural Computing, 2007, 6(4): 467-484.
[21] Ardagna D, Pernici B. Adaptive service composition in flexible processes[J]. IEEE Trans on Software Engineering, 2007, 33(6): 369-384.

Xiaona Xia, born in 1980, associate professor and master of Qufu Normal University. Her main research interests include software engineering, software architecture and Web service composition**.**

Baoxiang Cao, born in 1955, professor, master supervisor of Qufu Normal University, his main research interests include software engineering, software architecture and enterprise information technology and system integration.

Jiguo Yu, born in 1972, PhD of Shandong University, professor and master supervisor of Qufu Normal University, his main research interests include Algorithm Analysis and Design, WSN and graph theory