Toward Customer-centric SOA: Services Resource Active Provisioning Approach

Bin Wen, Ziqiang Luo
School of Information Science and Technology, Hainan Normal University, Haikou, China
Email: binwenwebb@gmail.com

Peng Liang
State Key Lab of Software Engineering, Wuhan University, Wuhan, China
Email: liangp@sklse.org

Abstract—**With the rapid development of services computing, services expression, registration and assembly have been partially solved. But for the ultimate goal of computing, namely on-demand service, it has not been able to achieve by far, especially active provisioning approach for services resource. Faced with key issue for requirements-dominated services virtualization, active services resource provisioning will be focused. Customer-centric SOA (Service-oriented Architecture) will be systematically investigated with requirements model and distributed requirements elicitation. In this paper, we have established software architecture with personalized active customization for services resource that will select unsatisfactory requirements fragment of customers to dynamically generate custom needs during services composition. By means of custom preferred provider selection designated to push the customization requirements, service providers complete the services resource production and customized products evaluation just in time, and the process will effectively prevent failure of services aggregation and services resource provisioning exceptions occurring. These efforts of the work will improve on-demand services resource provisioning capability. The experimental results and prototype show the efficiency and effectiveness of our approach.**

Index Terms—**services resource provisioning, customer-centric SOA, personalized customization, Web service**

I. Introduction

With the deep fusion of software and network, methods and techniques for software development inspire new innovations. Software as a service (SaaS),i.e. making full use of available online software resources to meet the changes and diversified requirements for personalized and distributed stakeholders, we are moving to the age of service-oriented software engineering(SOSE) [1, 2].

Internet of Things will organically link computing space and physical world. Social networks will further connect the computing world and human society to provide application innovations. The next higher development stage of the computing discipline should move towards the road to servicelization and user experience style [3]. So, on-demand service theory and related methods and technology featured with services resource aggregation, the mainstream networked software production, will have broad application prospects and academic innovation opportunities.

A. Services Resource Provisioning

Services resource represents a set of well-defined and interoperable Web software component. The basic physical unit will be understood as Web service, namely autonomous, open, self-describing and implementation-independent networked component [4]. Services resources provisioning means production style, the provisioning methods and sources structure of services resource.

The importance of services resources provisioning is mainly reflected as follows: (1) Services resource that is adequate to meet the personalized needs, is material basis of services aggregation and service-based software production; (2) Effective services resource provisioning approaches should be widely used to ensure smooth implementation and efficient completion for service-based software production.

The concept of services resource is from traditional software component repository that is directly related to the productivity and quality of final software product according to the degree of resource richness and meeting customer needs. Especially discipline basis of service-oriented software development paradigm, i.e. Services Computing, services components disperse in the Web widely. If services resources production, retrieval and dynamic binding can not effectively be implemented, services resource will be difficult to supply. Also, software production based on services and SaaS can not be achieved [5].

Manuscript received June 18, 2013; revised September 29,2013; accepted .

This research is supported by the Natural Foundation of Hainan Province (No.613162) and Higher School Scientific Research Project of Hainan Province (No.Hjkj2013-17)

TABLE I.
MASS CUSTOMIZATION AND PERSONALIZED SERVICES RESOURCE CUSTOMIZATION

Comparison Items	Feature/destination	Applicable stakeholders	Applicable artifacts
Mass customization(MC)	Large-scale productions cost and time to meet the individual needs of the users; the basic idea is that the customization production will transform or partially convert to mass production through restructuring and process reengineering; mainly divided into two phases, i.e. common component production and personalized assembly.	service requester, provider, software architect	final delivery of products
personalized services resource (component) customization	Produce a wide range of services resource (components) collection to adapt for individual users through personalized customization.	service requester, provider	atomic or simple composite service

To this end, a large number of studies have focused on effective provisioning of services resource. For example, SOA(service-oriented Architecture) [6] changes the availability of resources from end-to-end style to indirect addressing mode and decouples the direct relations among software representation and actual services through registration mechanism to more effectively address distributed positioning of services resource. Thus SOA becomes the classic distributed architecture for Internet computing.

From the views of traditional software component repository, these studies including description, management, storage, retrieval and resource sorting method of services resource mainly focus on service registration [7].A large number of academic and industry researchers are attracted to improve services resource provisioning efficiency through the efforts of the syntax, semantics, and other methods. There have been implemented some proposals such as UDDI[1], ebXML[2], Websphere Service Registry and Repository[3] etc. Meanwhile, ISO meta-registration standard will be studied and developed to improve interoperability among current registered repositories.

The importance of services resource provisioning has become consensus for industrialized service-oriented software development and applied to software production. And it has been partially supported by mainstream tools [8, 9].

B. Main Issues for Services Resource Provisioning

Services computing and virtualization are two supporting technologies for cloud computing. Services computing as cloud computing base and software production technologies of network era has been partially solved services representation, registration and assembly. At present its body of knowledge mainly includes service software lifecycle planning, resource production, service publishing, billing and governance. But, the industry and academic community that is lack of considering on-demand services and related issues, i.e. the ultimate goal of computing, only focus on service discovery and composition in general. The starting point of current study assumed that services resource is sufficient, but actual development stage is not the case. Services resource that meets users' needs often does not exist or can not be run satisfactorily. As described in the literature [1]: "If no services are available for some parts, the application developer can register them in the service brokers' directory and wait until the needed services are available." Thus it leads to a serious case with difficulty, complexity and low availability of services composition.

At the same time, due to the dynamic nature of Web services and error-prone services resource provisioning environment, a variety of exceptions in services resources provisioning can occur during the execution of composing services [10]. Exceptions refer to service failure (fault), network errors and abnormal events that are subject to resources or requirements changes. Problems caused by lacking of exception handling mechanism , such as poor performance , waste of resources, non-optimized service provider or even the failure of process execution. Services resource provisioning mechanism must be able to take actively, i.e. they should be able to adapt to the runtime exceptions.

Current SOA is producer-centric, basic idea is that service providers publish their produced services in registry and allow consumers to search for available service resources, and then compose it for business process. But user preferences and needs are constantly changing; only depending on current available resources composition is very difficult to build applications. Industrial software development platforms, such as IBM RSA[4], ActiveBPEL[5], Websphere Integration Developer[6], etc., are also lack of design considerations for services resources active provisioning, which also affect convenience and practicality of these tools on actual use.

As a result of the above problems: large-scale and distributed software development have been invested with a lot of manpower, material and tools to construct services resource, but outcome of high investment is not obvious. Regarding diverse and individualized user needs, services resource shows the relative absence.

The root causes of such an awkward situation are mainly lies in:

1) The assumption that services resource is rich enough in current development stage of services computing does not hold [11]; Meanwhile, even when the rich resources to serve, runtime services resource aimed at on-demand personalized

[1]http://uddi.org/pubs/uddi v3.htm
[2]http://www.ebxml.org/
[3]http://www-01.ibm.com/software/integration/wsrr/
[4]http://www.ibm.com/developerworks/rational/products/rsa/
[5]http://www.activebpel.org/samples/samples-4/samples.php
[6]http://www-01.ibm.com/software/integration/wid/

provisioning is still important to solve for application building;

2) Resources provisioning style and organization structure are still single [12]. Current services resource is from production in advance by providers, consumers (Requesters) can only be passive to choose.

Therefore, active services resource supply and rich source structure have become key problems to be urgently addressed for SOSE.

C. Motivation

Producing software as manufacture has been the dream that software engineers pursued. Recalling the history of software development approaches, regardless of machine-oriented, process-oriented, object-oriented or component-oriented, they are forward along the road to industrial large-scale production. Ford's assembly line is a typical representative of major industrialized production mode, through the standardization system, to achieve large-scale production capability. In accordance with the "building blocks" to assembly and substitute the products, a large-scale, rapid and low-cost production style can be achieved. With the amazing production capacity released from large-scale production, today we are facing a world that is abundant material supply. In order to cope with such rich world competition in a buyer's market, production must be user-centric and further leads to market segmentation; customized production has become a key competitive factor.

Previously designed conceptual model with large-scale customization for networked software, this model also assumes that services resource is rich enough, in which the shaded part refers to individual needs of stakeholders. If personalized services resource does not exist and also can not be achieved through value-added services composition [13], then it will relate to explore the issue of customized production which captured individual needs of service. How to associate the back-end services resource with individual needs fragment of production model? How to embody on-demand customization requirements of service component resources and on-the-fly production? These issues are directions and concerns of this paper why focusing on active customized provisioning of services resource.

Customized services refer to supply suitable and satisfactory value of experience according to their own needs of consumers. Service-based software development is distinguished into mass customization and personalized services resource customization (Table I). Mass Customization combines the advantages of two production modes to meet individual needs of customers. At the same time, it remains lower production costs and shorter delivery time. But personalized customization of services resource aims at capturing absent individual needs to customized production for abounding service component library resources and transforming services resource provisioning from service provider-centric mode to consumer requirements-driven production. The two styles complement each other to improve service quality of experience for stakeholders.

Regarding the traditional SOA (service-oriented architecture) [6], services resource provisioning strategy is based on service providers' priority. Service providers produce services resource and publish these services' description information in registries. Service requesters (consumers) can establish dynamic binding association with real services by querying service registries. Traditional SOA does not furnish the consumers with publishing related information functions for services production. Consumers can only retrieve, find and match provided services resource from service providers in the registry according to the needs of application. So, the process is essentially a passive selection.

The role and status of service customer-centric SOA structure (Figure 1), in which service consumer is active position, differ from the tripartite traditional SOA. Service consumers not only initiated services' demand from the provided services resource to choose, match and composition service, but also can conduct personalized on-demand active customized provisioning for services.

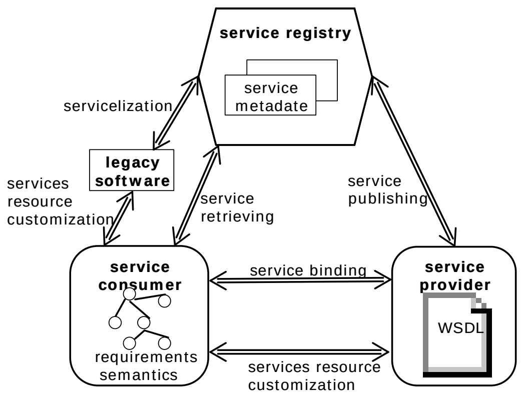


Figure 1. Service customer-centric SOA.

Therefore, regarding services resource corresponding to unsatisfied requirements fragment (individual needs) in the process of dynamic services aggregation, personalized requirements-driven active and real-time customization will be taken, namely customized provisioning of services resource. And a large number of diversified legacy software should provide huge amount of personalized services source with service customization approach. Services resource active production will meet personalized customization needs and expand resource capacity dynamically. Meanwhile, with legacy software servicelization to further enrich material basis of personalized services resource, this is also one of the feasible ways and practical claims for effective services resource provisioning. These approaches will give a suitable solution for services resource provisioning.

This paper is a research about service customer-centric SOA and focuses on services resource personalized active customized production approach. The main contributions of this paper are summarized as follows:

1) After putting forward service customer-centric SOA, our work mainly focus on personalized active customization in runtime and build

customization model for services resource based on mathematical proof.

2) In section 2, we also formalize runtime services resource selection algorithm based on historical use data.
3) In section 3, we design consumer requirements-driven customization architecture. ATOM-based customization information publishing mechanism has been adopted. We apply experiment and empirical analysis to the above approach. The resulting analysis reveals that the proposed approach has strong efforts on the feasibility and effectiveness.

The rest of the paper is organized as follows. Section 2 introduces the customization model of services resource. Section 3 presents active customized production of services resource provisioning for service consumers based on runtime information publishing/feedback mechanism, while designing platform and obtaining experimental analysis. Final remarks and comparisons with related work are reported in Section 4, while directions for future work are also touched upon in Section 5.

II. Customization Model for Services Resource

Definition 1 (Software Requirements, SR): Software Requirements (SR) is expressed as four-tuple *SR =< S;E; F;NF >*, where S represents stakeholders;E represents the context or domain of software; F represents the functional requirements specification of software, namely software systems must be able to perform activities; NF represents the nonfunctional requirements, namely software quality attributes such as reliability, security, etc.

Definition 2 (Software Requirements Semantics, Req): For *Req =< SR; Relations; Axioms >*, SR is Software Requirements; Relations denote interrelationship among the elements of each component in SR, such as equivalent, class Property, classification etc.; Axioms represent the sets of Axiom in SR.

Definition 3 (Customization Requirements, CReq): For CReq = < *Req; Parameters* >, Req represents requirements semantics of customized services resource, and Parameters denote parameter set of customized services resource.

Definition 4 (Customization of Services Resource): Given service requesters set R = $\{R_1, R_2, \cdots, R_n\}$ and service providers set $P = \{P_1, P_2, \cdots, P_m\}$,for customization requirements CReq$_i$ from $R_i \in R(1 \le i \le n)$, if there exists $P_j \in P(1 \le j \le m)$ which can complete services resource production and publishing within the specified time parameters T, we call service provider P$_j$ in response to services resource customization of service requester R$_i$.

Definition 5 (Customized Matching Degree): For customized matching degree of services resource $C_{match} = SemanticMatch_{P \to R} * e^{\frac{|t-T|}{T}}$,where $SemanticMatch_{P \to R}$ denotes semantic matching degree between the service semantics of provider's customization production and requirements semantic of requester, t represents customization completion time of provider, T is the required customization time of requester.

According to definition 5, the mean value of prior services resource customized matching degree can be calculated between n service requestor and m service providers, namely:

$$RP_{ij} = \frac{\sum^{Total} C_{ij}}{Total} \tag{1}$$

RP_{ij} is the current mean Customized Matching Degree of services resource for the provider P_j in response to customized needs of the requester R_i (Total is the total number of customization), which constitutes the services resource customization matrix RP_{Matrix}.

$$RP_{Matrix} = \begin{pmatrix} RP_{11} & \cdots & RP_{1m} \\ \vdots & \ddots & \vdots \\ RP_{n1} & \cdots & RP_{nm} \end{pmatrix} \tag{2}$$

Definition 6 (Custom Preferred Provider): Given the current services resource customization matrix RP_{Matrix}, custom requesters and providers pair off and implement assignment operating. We call the provider the related requester's custom preferred provider if there exists a provider which corresponds to arrive at the sum of the maximum matching result.

The mathematical models for solving custom preferred provider of services resource are described as follows:

$$\max C = \sum_{i=1}^{n} \sum_{j=1}^{m} RP_{ij} * x_{ij} \tag{3}$$

$$s.t. \begin{cases} x_{ij} = 0,1; i = 1,2,\cdots,n; j = 1,2,\cdots,m \\ \sum_{i=1}^{n} x_{ij} = 1; j = 1,2,\cdots,m \\ \sum_{j=1}^{m} x_{ij} = 1; i = 1,2,\cdots,n \end{cases} \tag{4}$$

This is similar to combinatorial optimization assignment problem, with reference to the Hungarian algorithm [14]. Its related conditions of the original algorithm are improved, such as the solved total minimum is transformed to maximum (The custom matching degree higher the better.). Meanwhile tag matrix is designed. The improved method is more efficient than original algorithm, easy to operate and implementation. Specific method you can see Custom Preferred Provider selection algorithm (Algorithm 1) which its time complexity is O(n^3) .

For the services resource custom provider selection, if the total custom matching degree is not required to reach a maximum just based on past selection chance to predict next selection, it is very suitable for Markov prediction method. A comprehensive forecast of the event, not only to be able to point out the incident with all possible outcomes, but also the probability of each outcome must be given with current state of probability to predict the best choice. This could be based on historical statistical data to build a custom provider state transition probability matrix T_{RP} , in which rows of the matrix correspond to the requester and columns correspond to custom provider. P_{ij} denotes selection probability between requester i and custom j in current state.

$$T_{RP} = \begin{pmatrix} P_{11} & \cdots & P_{1n} \\ \vdots & \ddots & \vdots \\ P_{n1} & \cdots & P_{nn} \end{pmatrix} \quad (5)$$

$$s.t. \begin{cases} 0 \le P_{ij} \le 1 \, (i, j = 1, 2, \cdots, n) \\ \sum_{j=1}^{n} P_{ij} = 1 \, (i = 1, 2, \cdots, n) \end{cases} \quad (6)$$

Definition 7 (prediction selection of custom provider): Given custom providers state transition probability matrix T_{RP} and the completed service customization selection state, next service customization selection state can be found according to Markov process without aftereffect and Bayesian conditional probability formula, in which the provider corresponding to maximum probability for each requester is the service custom provider prediction selection. Custom provider prediction model under the random Markov chain is described below.

$$\pi_j(k) = \sum_{i=1}^{n} \pi_j(k-1) P_{ij} \qquad (j = 1, 2, \cdots, n) \quad (7)$$

Where the state probability $\pi_j(k)$ indicates the probability of custom provider j in the moment (period) k under the known initial state (k= 0) to pass state transfer with k times, and $\sum_{j=1}^{n} \pi_j(k) = 1$

For example:

$$T_{Matrix} = \begin{bmatrix} 0.2000 & 0.4667 & 0.3333 \\ 0.5385 & 0.1538 & 0.3077 \\ 0.3636 & 0.4545 & 0.1818 \end{bmatrix}$$

The current state of requestor i is $\pi_i(k)$ = [0,1,0] which refers to completed customizable task of the provider with numeral 2. Then according to formula 7 we can obtain the probability of next state

$$\pi_i(k+1) = \pi_i(k) \begin{bmatrix} 0.2000 & 0.4667 & 0.3333 \\ 0.5385 & 0.1538 & 0.3077 \\ 0.3636 & 0.4545 & 0.1818 \end{bmatrix} = [0.5385, 0.1538, 0.3077]$$

So the custom provider corresponding to next requestor i can be predicted as 1.

Taking into account the formula 7 with assumed condition, namely selecting or rejecting a custom provider have the same opportunity, but the condition is not established in practice. To do this, we can build two custom state transition matrix: T_{Matrix} is the state transition matrix whose members are P_{ij} ; T^N_{Matrix} is the state transition matrix with rejecting the related providers which its elements are $P_{ij}N$. In view of these above-mentioned circumstances, the prediction model is described as follows.

$$\pi_j(k) = \sum_{i=1}^{n} (\pi_j(k-1) P_{ij} - \pi_j^N(k-1) P_{ij}^N) \ \ (j = 1, 2, \cdots, n) \quad (8)$$

The probability that the custom pair of (requestor,provider) remains at state i is $\rho_k^i = \frac{1 - \pi_i(k+m)}{1 - \pi_i(k)}$ (9)

Algorithm 1: Custom Preferred Provider selection algorithm

```
INPUT:$RP_{Matrix}$, services resource customization matrix under the current state
OUTPUT:$Map_{min(n;m)}$, Custom Preferred Provider selection set in which the element is (requester, provider) tuple.
1: begin
2: $Tag_{Matrix} \leftarrow d$ /*Initialize the tag matrix*/
3: for i=1 to n do
4: for j=1 to m do
5: $RP_{ij} = Max_i - RP_{ij}$;
/*Subtraction with the maximum value of each row*/
6: for j=1 to m do
7: for i=1 to n do
8: begin
9: $RP_{ij} = RP_{ij} - Min_j$; /*Minus minimum value per column*/
10: if $RP_{ij}$=0 then $Tag_{Matrix}(ij) \leftarrow zero$ ;
/*Modify 0 corresponding to the specific area of the tag matrix*/
11: end
12: $Z \leftarrow 0$ /*Z is the number of elements which is tagged as a*/
13: repeat
14: $k \leftarrow MinRow(Tag_{Matrix}; zero)$
/*Choose the minimum row with zero*/
15: $(k; r) \leftarrow ZeroCoodinate(k; Tag_{Matrix})$ /*Find the columns corresponding to k row which its value is zero*/
16: $Tag_{Matrix}(kr) \leftarrow a$ /*Zero in row k and column r is changed to a*/
17: cancel($Tag_{Matrix}; k; r; b$); /*The rest of zero is changed to b*/
18: $Z \leftarrow Z + 1$
19: until Num($Tag_{matrix}; zero$)=nil/*Until zero element is empty */
20: if |Z |=min(n,m) then
21: begin
22: $Map_{min(n;m)} \leftarrow Transfer(RP_{Matrix}; Tag_{Matrix}; a)$ /*Tuples corresponding to a are placed Map set*/
23: return $Map_{min(n;m)}$
24: end
25: else
26: begin
27: $S \leftarrow GetRow(Tag_{Matrix};Not(a))$ /*Rows without tagged a are marked S*/
28: $S_{col} \leftarrow GetCol(Tag_{Matrix}; S; b)$
/*For the rows of marked S, columns with tagged b are marked S*/
29: $S_{row} \leftarrow GetRow(Tag_{Matrix}; S; a)$
/*For the columns of marked S, roes with tagged a are marked S*/
30: $underline(S_{col}; S_{row}; cancel)$
/*Rows without S are marked cancel and
columns with S are marked cancel*/
31: $x_{min} \leftarrow GetMin(RP_{Matrix}; Tag_{Matrix};Not(cancel))$
/*Find the minimum element of rows and columns without marked cancel in $RP_{Matrix}$*/
32: $SubAdd(RP_{Matrix}; Tag_{Matrix}; x_{min})$ /*All elements of $RP_{Matrix}$ corresponding to the rows without marked cancel subtract the minimum, and the elements corresponding to the columns with marked cancel add the minimum*/
33: goto 12
34: end
35: end if
36: end
```

Then the probability which customized service k will be selected after M times customization is

$$\pi_{k,T+m=} \sum_{k \in S_{i,T}} \rho_k^i + \sum_{j=1, j \ne i}^{n} \sum_{k \in S_{i,T}} (1 - \rho_k^i) [\frac{p_{ji}}{1 - p_{jj}} . I\{p(k) = C\} + \frac{p_{ji}^N}{1 - p_{jj}^N} . I\{p(k) = NC\}]$$

(10)

III. Active Customized Production For Services Resource Provisioning

In the human-network interactive Web 2.0 era, users produce huge amounts of data. Therefore, data transmission and selective filter with timely, cross-system realization is particularly important. Data aggregation came into being (Figure 2), and server-side data aggregation process user-generated data content with standard data protocol instant (subscription) passed to the relevant users in order to achieve timely, on-demand data supply. Data aggregation protocol formats mainly divide in RSS (Really Simple Syndication) and ATOM. RSS format emphasize on information interaction of social software, and the ATOM format is more suitable for data fusion.

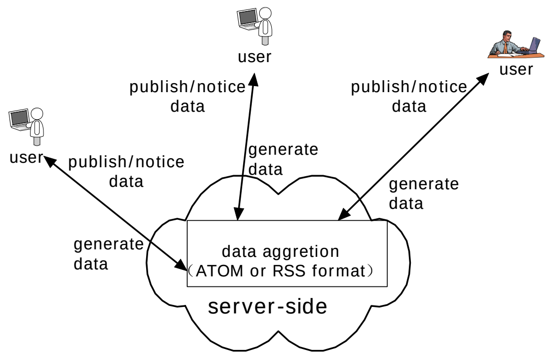


Figure 2. Data aggregation.

ATOM/RSS are a special XML file including some standard header and information items format definition. Information items contain the title, author, date, and summary etc. Once a website supports the content output with RSS/ATOM format, then the system or software (for example, Atom/RSS Reader) which can parse the ATOM/RSS file, will subscribe and syndicate the site's content in accordance with ATOM/RSS analytical specifications. Content from different systems can be integrated into a single system so as to attain the loose alliance of different system functions by Atom/RSS content syndication. For information systems, the significance of ATOM/RSS lies in the realization of the machine-readable content and integration becomes possible, thus facilitating re-classification of information aggregation.

A. Consumer Requirements-driven Customization Architecture

Requirements Sign Ontology (RSO) [15] is semantic description of overall business process including requirements-induced services resource (components) and control structure, and it also denotes knowledge about business workflow to guide services resource identifying and complete services aggregation.

For unable matched services, requirements-driven active customization will be used to achieve custom produce for service providers while services aggregating. The main difference is to be able to identify (business) process and services resource between service-oriented requirements engineering and the traditional requirements engineering. We have designed the meta-description of networked software (see [15]), in which stakeholders is composed of participants playing different roles whose required goals will be achieved by the business process. As a software agent, services resource is the constituent elements of business process.

Currently, to address the diversification and individuation of service requirements, a RGPS meta-model framework is proposed [15]. The framework includes four layers: Role layer(R), Goal layer (G), Process layer (P), and Service (S) layer. Through the associations among the four layers, the framework can be able to provide strong support for improving interoperability of requirements model. RGPS-based requirements modeling process will start from the domain analysis of problem space and complete in services-based solution.

On the one hand, the framework will guide how to organize domain ontology to create asset model; on the other hand, domain ontology depicts concepts and its semantic associations to support semantics for asset modeling that is characterized by ontology description (OWL or OWL-S format). It will facilitate the semantic reasoning in the SODR asset repository.

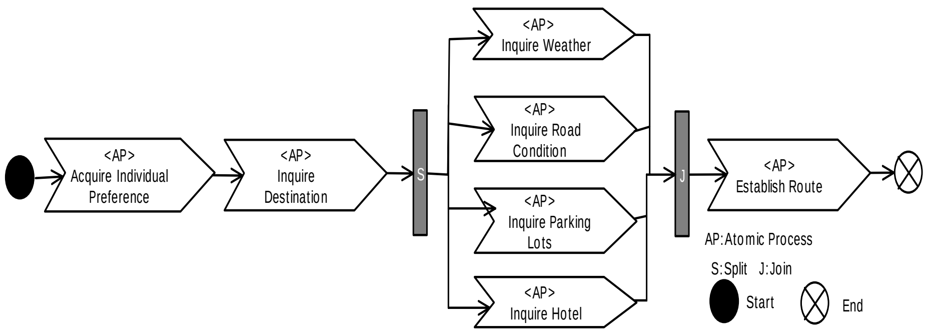


Figure 3. Visualization for RSO example.

In our previous works [16], we have presented a Requirements Rationale Model (RRM) and related reasoning rules. Combined with RGPS and RRM, experts-level requirements semantics model for services software can be created as an evolutionary base.

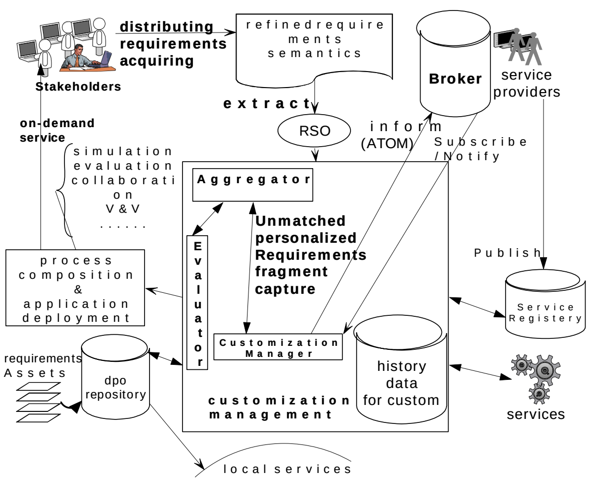


Figure 4. Consumer Requirements-driven Customization Architecture for Services Resource.

We have systematically explored requirements-driven semantics acquisition with distributed stakeholders participation and semantic-conducted services aggregation. The below will elaborate on customer requirements-enabled active customized production for services resource in details.

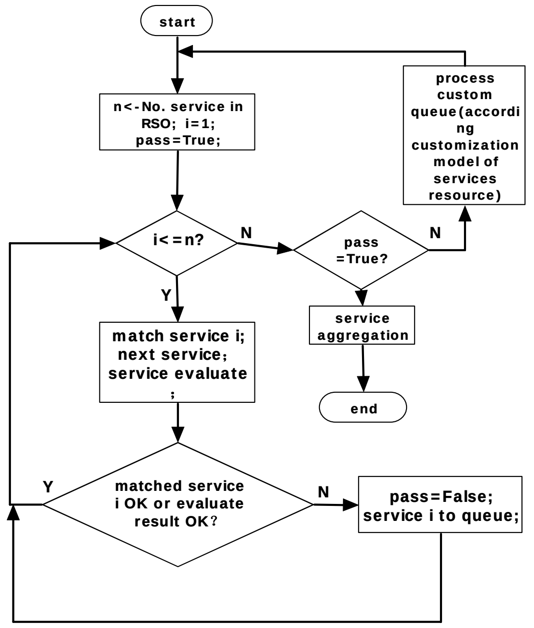

start
n<-No. service in RSO; i=1; pass=True;
i<=n?
N
Y
pass =True?
N
process custom queue(according customization model of services resource)
service aggregation
end
match service i; next service; service evaluate ;
matched service i OK or evaluate result OK?
Y
N
pass=False; service i to queue;

Figure 5. Services Aggregation Flowchart with Customization

Active service technique has become the focus for services computing. Producing the services that can meet the consumer needs actively, can change passive service delivery style, which is unable to satisfy the user's requirements effectively. To this end, functions such as publishing requirements information for consumers and evaluating service quality will be added on traditional SOA. Consumers dynamically publish requirements description of services for unmatched services resource in the light of business process to providers. Service providers will develop the service products in accord with the semantic description of requested services, and consumers will choose the best service to implement the business process for finishing interrupted services aggregation through quality evaluation.

Algorithm 2:Customization Process of Services Resource

INPUT:RSO (i.e. process semantics description)
OUTPUT:Result of services aggregation

```
1: begin
2: n← Number of services in RSO
3: i←1; pass←True;
4: if i<=n then
5:   matching $service_i$ in RSO;
6:   $service_i$ evaluation;
7:   i++;
8:   if $service_i$ evaluation or matching is ok then
9:     goto 3
10:  else pass←False;$service_i$ into customization queue
11: else
12:  if pass=True
13:   then
14:     services aggregation;
15:     return
16:   else
17:      customization queue will be processed;
18: goto 3
19: end
```

Considering the feasibility and simplicity, we employ Atom format to encapsulate requirements semantics for service and fulfill subscription & notification with service semantics by means of subscribe/inform mechanism. Solution of active service customization framework is shown in Figure 4. Service description in Atom is from service semantics in RSO that is derived from stakeholders, so customized services are also delivered actively to satisfy stakeholders.

The core part of this framework is semantic-enabled customization management platform, including customization manager, evaluator, and aggregator. Aggregator executes searching, matching, and binding the services in line with the business process of RSO. Customization manager is responsible for the customized production and management with unmatched services. Evaluator will assess and verify the quality of matched services after production.

The process of services aggregation with custom processing is shown in Figure 5, and the corresponding flow of customization production is presented in Algorithm 2(CPSR). Unmatched and unqualified services will be appended into the queue of customization that will be processed to notify providers for on-demand production using Atom subscribe/inform style by customization manager. For customized and matched services, they will rerun to the aggregation flow to continue to complete the business process.

B. Customization Information Publishing based ATOM Protocol

Experimental platform mainly adopts Apache Abdera (http://incubator.apache.org/abdera/) to generate Atom message and publishing protocol. Informed Atom format is based on the design of service semantics that is the extension of Service segment in RGPS meta-model framework in details.

Algorithm 3: service2Atom algorithm.

```
INPUT: customized services queue(XML)
OUTPUT: Atom file for customized services
1: Intialize Atom file header;
2: form feed title,ID etc.
3: while (service queue)<>NIL do
4:   get a service segment from queue; //queue length decrease
5:   create a entry header; //<entry>
6:   create title part from ID of service segment;
7:   create ID; //GUID
8:   create content header;//Type="application/xml"
9:   begin
10:    create serviceType part from process Type;
11:    create name part from ID;
12:    create input from Input or hasInput;
13:    create output from Output or hasOutput;
14:    create precondition from Precondition or hasPreconditiont;
15:    create effect from Effect or hasResult;
16:   end
17: form updated part;
18: create summary part from comment
19: end while
20: if (entry part)<>NIL then
21:   write a Atom file;
22:   return the file;
23: end if
24: return
```

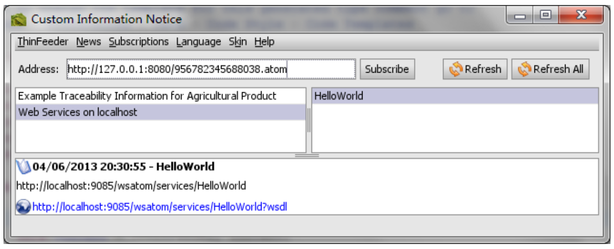
Semantics in RSO (Figure 3) are expressed as an OWL-S format. Since the OWL-S file is a bit long, only the instantiated description of Inquiry destination service segment according to the specification of service is excerpted as follows:

```
<process:AtomicProcess rdf:ID="Inquiry destination AP">
 <process:comment>This is a service about inquiring destination information.</process:comment>
 <process:hasGoal rdf:resource=" Arrange travel plan Goal.owl#Inquiry destination OG" />
 <process:hasInput>
 <process:Input rdf:ID="Address name AP 123456789123456">
   <process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#string" />
  </process:Input>
 </process:hasInput>
  <process:hasOutput>
    <process:Output rdf:ID="Inquiry result AP legal user ">
      <process:parameterType rdf:datatype="http://www.owl¡ontologies.com/UrbanTransportation.owl#legal user" />
    </process:Output>
   <process:Output rdf:ID="Inquiry result AP coordinate ">
      <process:parameterType rdf:datatype="http://www.owl¡ontologies.com/UrbanTransportation.owl#coordinate" />
      </process:Output>
    </process:hasOutput>
<process:hasResult>
   <owl:ObjectProperty rdf:ID="hasResult">
   <rdfs:label>hasResult</rdfs:label>
   <rdfs:domain rdf:resource="#Process"/>
   <rdfs:range rdf :resource="#Result">
 </process:hasResult>
</process:AtomicProcess>
```

The corresponding relationships between every item of entry in Atom format and service semantics in RSO can be listed in a table.

According to corresponding relationships table and the semantic description of service, Inquiry destination service in Atom format is as follow:

```
<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http :// www.w3.org/2005/Atom" xmlns:xml="http://www.w3.org/XML/1998/ namespace" xml:lang="en-US" xml:base="http://example.org">
<id>http :// localhost :8901/ services .atom</id>
< title >Customized Web Services</ title >
<updated>2010-06-23T23:00:35Z</updated>
<link href="http :// localhost :8901/ services .atom" rel="self" type=" application /atom+xml"/>
<author>
 <name>Customization manager</name>
 <uri>http :// localhost :8901</ uri>
 </author>
 <entry>
  < title type="text">Inquiry destination </ title >
  <id>urn:uuid:1225c695-cfb8-4ebb-aaaa-80da344efa6a</id>
  <content type=" application /xml">
   <serviceType>Atomic Service</serviceType>
   <operation>
    <input name="Address name AP 123456789123456">
     <parameterType>http :// www.w3.org/XMLSchema#string</parameterType >
    </input>
    <output name="Inquiry result AP  legal user ">
     <parameterType>http://www.owl-ontologies.com/UrbanTransportation.owl#legal user</parameterType >
    </output>
    <output name="Inquiry result AP_coordinate ">
     <parameterType>http ://www.owl-ontologies.com/UrbanTransportation.owl#coordinate</parameterType>
    </output>
    <e_ect>
     <domain>"#Process"</domain>
     <range>"#Result"</range>
    </ e_ect >
    </ operation>
   </content>
   <updated>2010-06-23T23:01:35Z </updated>
   <summary type="text">This is a service about inquiring destination information. </summary>
  </entry>
 </feed>
```



http://127.0.0.1:8080/956782345688038.atom

Example Traceability Information for Agricultural Product
Web Services on localhost

HelloWorld

04/06/2013 20:30:55 - HelloWorld
http://localhost:9085/wsatom/services/HelloWorld
http://localhost:9085/wsatom/services/HelloWorld?wsdl

Figure 6. subscribed custom information notice for services resource

The transformation algorithm from semantic description of services to Atom is shown in Algorithm 3.

C. Experiment and Empirical Analysis

1) Evidence for the Prototype System: Based on the Abdera library, thinfeeder, a open source RSS reader (http://thinfeeder.sourceforge.net/), can be revised to display Atom and subscribe the services (see Figure 6). Also, adding 'notify' function to thinkfeeder, aggregator can get customized services data (WSDL or OWL-S file location, service endpoint URI etc.) to facilitate rerunning aggregation flow after completing the feedbacks of customized service queue.

Prototype of user requirements-driven services engineering is developed as a series of requirements-conducted services platform that have realized the main features, where requirements asset modeling for domain, service registration , requirements capture and analysis modules have launched a number of versions. The main part of prototype adopts language written in Java and PHP using Apache as a web server, MySQL database as a background. The whole system is based on the J2EE specification to achieve distributing operations with multi-user network environment and also solve the issue of heterogeneity. The platform uses a distributed service-oriented architecture as a background system architecture, and the whole system is divided into three parts , namely: requirements semantic acquisition module , service aggregation module and services resource customized modules. Among them, requirements-semantic export and service operation governance are packaged as RESTful interface to provide the modules of service aggregation and services resource customization for calling. The entire architecture uses B/S style that is shown in Figure 7.

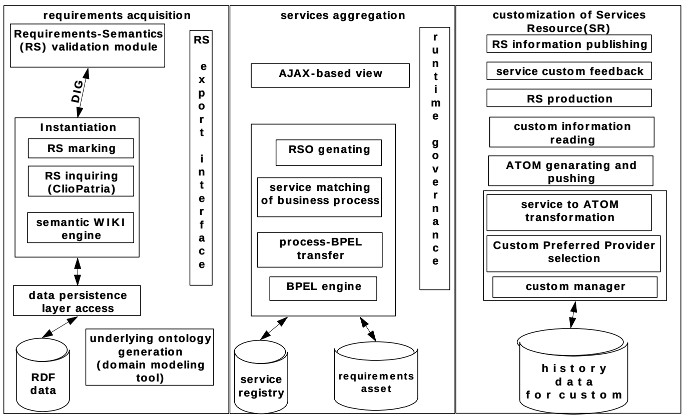

requirements acquisition
Requirements-Semantics (RS) validation module
RS export interface
DIG
Instantiation
RS marking
RS inquiring (ClioPatria)
semantic WIKI engine
data persistence layer access
RDF data
underlying ontology generation (domain modeling tool)
services aggregation
AJAX-based view
runtime governance
RSO genating
service matching of business process
process-BPEL transfer
BPEL engine
service registry
requirements asset
customization of Services Resource(SR)
RS information publishing
service custom feedback
RS production
custom information reading
ATOM genarating and pushing
service to ATOM transformation
Custom Preferred Provider selection
custom manager
history data for custom

Figure 7. Personalized active customization platform for services resource

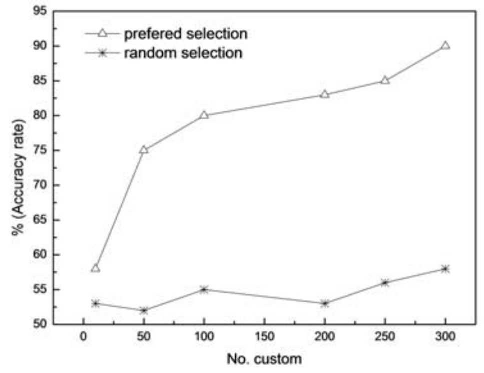
2) Experimental Analysis of Services Resource Customization Model: Regarding services aggregation process with customization, unmatched and unsatisfactory services resource will trigger custom platform to work according to the algorithm 2. Resource custom program adopts JAVA through execute algorithm 3 to transfer requirements specification of services to ATOM and push to the collection of ideal service providers with custom preferred provider selection algorithm.

Custom preferred provider selection method based on custom historical data matrix compared to common random selection method. Before the experiment starting, we will firstly generate customized historical data. Due to the absence of such data in industry, the data matrix will be automatically generated by the program.

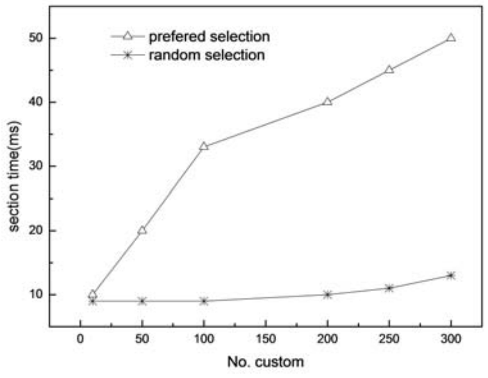
Random selection uses JAVA random function and respectively compare the accuracy rate and response time (response time including historical data to generate spending). Accuracy rate=Return/Total (Return denotes No. service providers for customized production. Total is No. service providers for pushing custom information.)

Compared measurement is the total of service providers as 10, 50, 100, 200, 250, 300. Each pushed customs are 20% ordering in front of the total number of customs. The experimental operating environment consists of Dell Intel (R) Core (TM) 2 Duo CPU and 2G memory.

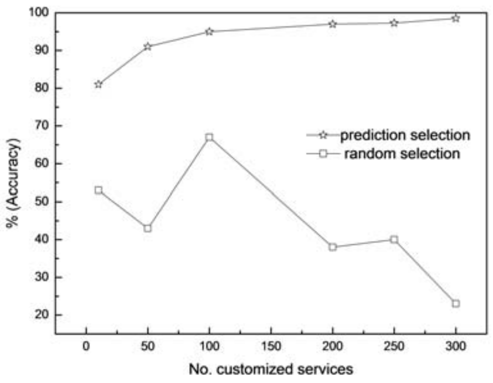
The experimental results show that custom preferred selection algorithm according to the custom historical data significantly better than random selection for selecting accuracy rate (Figure 8(a)).



(a) Custom Selection Accuracy Comparison



(b) Selection Response Time Comparison



(c) Prediction Model Comparison

Figure 8. Customization Model Experimental Analysis

With the growth of the number of providers, response time significantly increases due to the historical data generating (Figure 8(b)). However, the overall running time is acceptable within 50ms. With the increase of No. service customs, prediction accuracy rate of predict selection method continues to raise (Figure 8(c)). But, accuracy rate of random prediction is low value and instability.

D. Agricultural Product Traceability Service Design with Services Resource Active Provisioning

Regarding the experiment carrier for services resource provisioning, we select the Hainan agricultural product online traceability service system as application field. Because in a service-oriented agricultural traceability domain, Stakeholders facing the domain user is rich and diverse individual needs, which led to a variety of customization requirements. Meanwhile numerous legacy

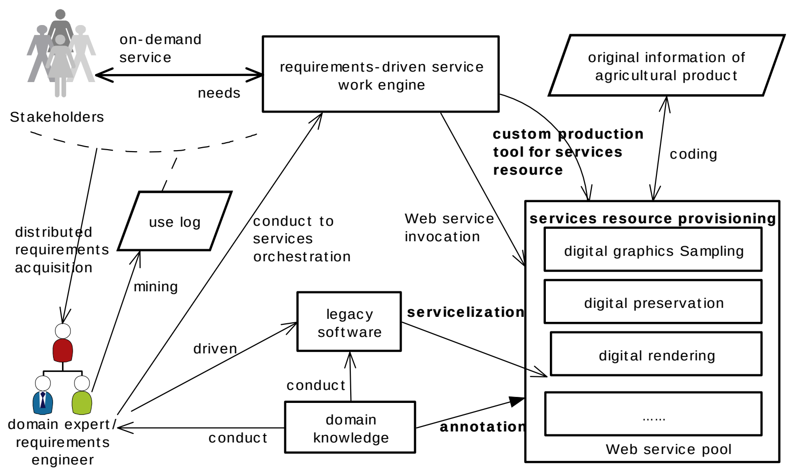

on-demand service
needs
Stakeholders
requirements-driven service work engine
original information of agricultural product
custom production tool for services resource
coding
distributed requirements acquisition
use log
mining
conduct to services orchestration
Web service invocation
services resource provisioning
digital graphics Sampling
digital preservation
digital rendering
......
Web service pool
legacy software
servicelization
driven
conduct
domain knowledge
annotation
conduct
domain expert/ requirements engineer

Figure 9. Architecture for Agricultural Traceability Service System

software of processing agricultural products exists to meet empirical demands of the project with active services resource provisioning.

How better to achieve real-time and on-demand digital service for Hainan online agricultural product traceability system under the Internet with customization based resource provisioning ecological chain extension? The issue is the focus for empirical carrier research.

The preliminary overall application architecture is shown in Figure 9. Specific functional requirements of services resource active provisioning can be acquired through full communication with stakeholders. Preliminary functional requirements of CASE tools are shown in Table II.

Agricultural traceability service system with B/S structure and active services resource provisioning is developed, and cell phone application of agricultural traceability service based on Android platform which can generate and parse QR 2-dimensional coding for agricultural product traceability is also finished and can achieve online inquiries combined with server to complete the mobile client-side access and receive subscribed custom information notice simultaneously.

IV. Related Works

The traditional SOA structure [6] is service provider-centric architecture, where service providers produce services, service providers publish service description information in service registry, and service consumers can be established binding association with true services by querying the service registry. But the traditional SOA have no functions design for service consumers publishing information. Consumers can just retrieve, discover, and match services resource which service providers able to supply according to the application requirements in the registry. It is essentially a passive selection process.

Salesforce.com (http://www.salesforce.com/) has a customer success rate with unmatched superiority in the industry as a global on-demand CRM (Customer Relationship Management) solution and the leader in cloud computing. Through the use of Salesforce.com inherent programming language, Apex, users can develop any custom operation. Process and layout can be customized to meet the needs of different clients, and also the status of a service provided by the customer is able to be tracked. Salesforce.com can only analogize services resource customization focusing on the delivery of service results (e.g. SaaS), but personalized services resource provisioning, namely service software production methods (such as SOA), have not effectively been investigated.

Consumer-Centric Service-Oriented Architecture [12] proposes that process and application template etc. are able to publish large-scale customization information and reuse as well as services resource. This is a large-grained reuse ideas embodied in the SOA structure. But the architecture can not support customization and related publishing protocol. In our work, the users' needs will further be upgraded the level of reuse. Not only can the needs be re-used in services aggregation, but also will it be used to guide services resource customized production, e.g. on-demand customization.

Cloud Computing Open Architecture [17] started to pay attention the subscription mechanism of service publication that mainly was related to the service subscription process, roles, and notice framework and conclude that service subscription notice is the core mechanism of the cloud service provider. However, the subscription does not consider the mechanism from the users' needs and lack of requirements-led implementation.

The literature [18] explores the progress and roadmap of service-oriented computing and particularly refers to the dynamically configurable run-time architecture, adaptive management and dynamic adaptive process with the importance and challenge for the future of services computing. But it does not give specific solutions. From the perspective of the service consumer, we propose a run-time solution with services resource adaptive active production. It will provide the new idea for user requirements-driven solution.

The service-oriented modeling and architecture modeling environment (SOMA-ME) [19] is the first framework for the model-driven design of service-oriented architecture (SOA) solutions using the service-oriented modeling and architecture (SOMA) method. On

TABLE II.
FUNCTIONAL REQUIREMENTS OF CASE TOOL FOR SERVICES RESOURCE ACTIVE PROVISIONING

Resource Provisioning Activities	Functional Requirements of CASE Tool
1.Requirements Acquisition	1.1 Requirements Model Import
	1.2 Distributing Stakeholders Participate to get consistent and complete requirements specification
2.Custom Manufacturing of Services Resources	2.1 Customization Requirements Slicing(Splitting)
	2.2 Custom Description Information Convert OWL-S or WSDL
	2.3 Customized Information Push ATOM and Custom Preferred Provider Selection
	2.4 Registration and Feedback of On-the-fly Custom Production Information
	2.5 Custom Management
3.Legacy Software Servicelization	3.1 Legacy Software Signature Analysis for Specific Code(such as PHP)
	3.2 Legacy Software Slicing(Module Dependency Diagram)
	3.3 Service Package based on Customization (wrapper) and Produce WSDL Description
	3.4 Services Resource Information Publish (Register)
4.Common Features	Domain Knowledge Annotation; Services Resources Query

the basis of MDA (Model Driven Architecture) and toward SOA application, SOMA-ME with UML meta-model architecture is developed to create nine layers of S3 (service solution stack) asset. Through model transformation and identification, SOME-ME platform can identify and realize service (service componentization) to accomplish business goal modeling for user. The architecture has been integrated in IBM RSA toolkit. However, SOMA-ME platform does not support service customization.

Advertising Web services with Atom 1.0 [20] establishes the corresponding relationship between WSDL and ENTRY item in Atom. This approach updates web services information through polling fashion and is the supplement to UDDI. [20] mainly applies Atom for to discover and search the services that have been published.

An Atom-based architecture for Web services discovery [21] mainly concentrates on the technique for services discovery. It aims to provide a handy and scalable discovery facility for most Web services providers and requesters, who can easily participate in service discovery via various user friendly application interfaces. With the proliferation of RSS/Atom, AtomServ (a platform for implementation) widens the adoption of service discovery by allowing simple and unified user access to frequently changing business services.

In [22], similar services are organized into service pool, i.e., homogeneous web service community. Adopting the popular atom feeds, they design a prototype to facilitate the consumers discover the subscribing Web services in an easy-of-use manner.

Compared to our work, we mainly focus on customized service production actively and use the popular Web 2.0 subscribe/publish techniques, such as Atom to realize service requests publishing and just-in-time on-demand service production.

V. CONCLUSIONS

Service-oriented approaches are gaining more and more attention since they claim to provide new and flexible ways of supporting the activities in an organization. However, current ways of implementing these approaches often lead to unmatched services and runtime exceptions without delivering the expected advantages.

Customization of the active provisioning of services resource mainly includes two aspects: Firstly, for unmatched services resource of service composition, sliced or segmentation method should be chosen to acquire the individual needs of the services resource according to the overall requirements. Service providers accomplish active on-demand production with a series of automated custom management tools to issue customization requirements on-the-fly. Secondly, a huge amount of legacy software will be comprehensively reused, namely servicelization.

In order to do so, we introduced these fundamental issues to analysis services resource provisioning in the view of personalized customization. We have investigated customization model of services resource and proposed custom preferred provider selection approach and prediction selection of custom provider according to history data. On this basis, personalized active customization of services resource have been designed to adaptively subscribe, push, feedback with ATOM data aggregation on runtime, and also its main part have been implemented with on-demand customization capability. These efforts will effectively promote the on-demand services resource provisioning and provide the innovative ideas for the final completion of the dynamic adaptive services resource provisioning.

The paper has established software architecture with personalized active custom for services resource that will select unsatisfactory requirements fragment of customers to dynamically generate custom needs in the running of the services composition. By means of custom preferred provider selection approach designated to push the customization requirements, service providers complete the services resource production and customized products evaluation just in time and the process will effectively prevent failure of services aggregation and services resource provisioning exceptions occurring.

In future work, we have three main goals: 1) to validate our framework in more real-life case studies, especially at enterprises that have a large amount of services, and 2) to investigate service software monitoring and management of operating mechanism. 3) to explore the feedback mechanism after the successful customization. We have plans to also evaluate the framework additionally at an e-commerce company. After these steps, we intend to use our framework as a basis for selecting the most suitable standards for services resource provisioning. It may be the case that some aspects are not covered by existing approaches and that we need new ones, but this is still to be investigated.

ACKNOWLEDGMENT

This research has been supported by the Natural Science Foundation of Hainan Province(GrantNo.613162) and Higher School Scientific Research Project of Hainan Province (GrantNo.Hjkj2013-17).

REFERENCES

[1] S. S. Yau and H. G. An, "Software engineering meets services and cloud computing," *Computer*, vol. 44, no. 10, pp. 46–52, 2011.

[2] L.J. Zhang, "Big Services Era: Global Trends of Cloud Computing and Big Data," *IEEE Transactions on Services Computing*, vol.5,no.4, pp. 467-468, 2012.

[3] D. A. Menasce, H. Gomaa, S. Malek, and J. P. Sousa, "Sassy: A framework for self-architecting service-oriented systems," *IEEE Software*, vol. 28, no. 6, pp. 78–85, 2011.

[4] H. Becha and D. Amyot, "Non-Functional Properties in Service Oriented Architecture–A Consumer's Perspective," *Journal of Software*, vol. 7, no.3, pp. 575-587, 2012.

[5] S. Marston, Z. Li, S. Bandyopadhyay, J. Zhang, and A. Ghalsasi, "Cloud computing - the business perspective," *Decision Support Systems*, vol. 51, no. 1, pp. 176–189, 2011.

[6] Gartner, "Service oriented architectures: part 1 and part 2," http://www.gartner.com/DisplayDocument?id=302868, 1996.

[7] M. Li, Y. Ma, and Y. Liang, "Study the Model of Information Resource Classified Register and Discovery based on Hierarchy in Grid," *Journal of Software*, vol. 7, no.7, pp. 1554-1561, 2012.

[8] Microsoft, "Knowledge base articles for driver development," http://www.microsoft.com/whdc/driver/kernel/kbdrv.mspx, 2013-06-16.

[9] IBM,"IBM developerworks:IBM's resource for developers and it professionals",http://www.ibm.com/developerworks/, 2013-06-16.

[10] Q. Z. Sheng, B. Benatallah, Z. Maamar, and A. H. Ngu, "Configurable composition and adaptive provisioning of web services," *IEEE Transactions on Services Computing*, vol. 2, no. 1, pp. 34–49, 2009.

[11] R. Retter, C. Fehling, D. Karastoyanova, F. Leymann, and D. Schleicher, "Combining horizontal and vertical composition of services," *Service Oriented Computing and Applications (SOCA)*, vol.6, no.2, pp.117-130, 2012.

[12] W. Tsai., B. Xiao., R. A. Paul., and Y. Chen., "Consumer-centric service-oriented architecture: A new approach," in Proceedings of the Fourth IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems and Second International Workshop on Collaborative Computing, Integration, and Assurance (SEUS-WCCIA2006). IEEE Computer Society, 2006.

[13] M. Varguez-Moo, F. Moo-Mena, and V. Uc-Cetina, "Use of Classification Algorithms for Semantic Web Services Discovery," *Journal of Computers*, vol. 8, no.7, pp. 1810-1814, 2013.

[14] Hu Yunquan, *Operations Research*, 3rd ed. Beijing: Tsinghua University Press, 2007.

[15] Wen Bin, He Keqing, Liang Peng, Wang Jian, "Requirements Semantics-driven Aggregated Production for On-demand Service," *Chinese Journal of Computers*, vol. 33, no. 11, pp. 2163–2176, 2010.

[16] P. Liang, P. Avgeriou, and V. Clerc, "Requirements reasoning for distributed requirements analysis using semantic wiki," in Proceedings of the International Workshop on KNOWledge engINeering in Global software development (KNOWING). 388-393: IEEE Computer Society Press, 2009.

[17] L.J. Zhang. and Q. Zhou., "CCOA: Cloud computing open architecture," in 2009 IEEE International Conference on Web Services(ICWS 2009). IEEE Computer Society, 2009, pp. 607–616.

[18] M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann, "Service-oriented Computing: state of the art and research challenges," *Computer*, vol. 40, no. 11, pp. 64–70, 2007.

[19] L.J. Zhang., N. Zhou., Y.-M. Chee., A. Jalaldeen., K. Ponnalagu., R. R. Sindhgatta., A. Arsanjani., and F. Bernardini., "Soma-me: A platform for the model-driven design of soa solutions," *IBM SYSTEMS JOURNAL*, vol. 47, no. 3, pp. 396–413, 2008.

[20] J. Snell, "Advertise web services with atom 1.0," http://www.ibm.com/developerworks/webservices/library/wsatomwas/, 2009.

[21] C. Wu. and E. Chang., "Aligning with the web: an atom-based architecture for web services discovery," *Service Oriented Computing and Applications (SOCA)*, vol. 1, no. 2, pp. 97–116, 2007.

[22] X. Liu., G. Huang., and H. Mei., "Discovering homogeneous web service community in the user-centric web environment," *IEEE Transactions on Services Computing*, vol. 2, no. 2, pp. 167–181, 2009.

Bin Wen joined School of Information Science and Technology, Hainan Normal University through "Elites Program" in 2011 as Associate Professor. He earned his Ph.D. in Computer Software and Theory, Wuhan University, China in 2011. Prior to this, he received his master degree in Computer Science at NUDT (National University of Defense Technology) in 2005 and Bachelor of Science degree in Mathematics from Hubei Normal University, China in 1993. He paid a research visit to State Lab of Software Engineering (Wuhan University) and National University of Singapore in 2001 and 2006 as Associate Professor, respectively. His current research interest includes services computing and requirements engineering for services. He is the member of Technical Committee on Services Computing, China Computer Federation.

Ziqiang Luo received the PhD degree in Computer Science at the PLA University of Science and Technology, Nanjing, China in 2007. Now he is an Associate Professor of School of Information Science and Technology, Hainan Normal University.

Peng Liang is currently an Associate Professor of State Lab of Software Engineering (Wuhan University), China. He received the PhD degree in Computer Science at Wuhan University, China in 2007.His current research interest includes software architecture and services computing.