Process Semantic-enabled Customisation for Active Service Provisioning

Bin Wen, Keqing He, Peng Liang
State Key Lab of Software Engineering
Wuhan University, Wuhan, China

Lai Xu
Software Systems Research Centre
Bournemouth University, UK

Abstract—Service oriented architecture (SOA) promotes highly standardized, loosely coupled and Web-enabled services to foster rapid, low-cost and easy composition of distribute applications. Critical to SOA software development is to acquire users’ requirements and to provide service analysis and design techniques for identifying, conceptualizing, profiling, and rationalising service-enable applications. Semantic-enabled requirements engineering (SRE) for developing service-oriented applications helps to transform disordered users’ needs into ordered requirement specifications. It facilitates requirements acquisition with multiple stakeholders’ participation. It further enhances semantic-conducted services aggregation and eventually provides customized service productions according to process semantics of requirements. This paper concentrates on user-centric service customization with a paradigm shift from classical provider-centric service provision to user-centric active service production. Subscription and pushing technology of ATOM/RSS is adopted to actively push users’ needs to service providers. Semantic description of service requirements is proposed, and characterizes in identifying process (workflow) and service. A customized service production platform is designed to demonstrate its practicability and effectiveness.

Keywords—service requirements; process semantics; customised service production

I. INTRODUCTION

Methodologies for developing software with high quality and in accordance with user requirements are mainly divided in two categories: formalization and engineering. Accordingly, Software engineering (SE) is an engineering subject that is designed to realize the successful software delivery.

Requirements engineering (RE) is crucial to the success of software development projects, especially for networked software based on services [1]. RE has its special considering issues and mainly includes dynamic elicitation and analysis, evolution modeling, requirements management and model verification for user’s needs and so on. Presently, typical RE approaches include goal-oriented, ontology-oriented, scenario-based, problem framework, pre-requirements analysis based on domain modeling, and aspect-oriented method [2].

The applicability of those above-mentioned approaches must be reconsidered while service-oriented software gradually becomes the new paradigm. Significant differences exist between service-oriented and object-oriented software development, service-oriented software development, namely networked software [1], focuses on finding and reusing the existing services resource to dynamically satisfy users requirements, whereas current requirements modeling methods are deficient to support that. RE methods for service should be able to effectively identify and model processes (business processes) and services.

Apart from small and instant services-assembled software, internet users with same interest and similar benefit, namely stakeholders including end customers, sponsors, project managers, architects, designers etc., also have urgent demand for domain-specific large scale service-oriented software based on services. The service-oriented software systems have following characteristic: enormous functions and complex requirements; stakeholders expose more concerns than end users; software development is a long-term, continuous, evolutionary growth process; aggregate all of resource from local or internet as well as services [1].

Faced with complex stakeholders and varied needs, large-scale service-oriented software should combine with the research of information technology and information system to complex structure by introducing complex science theory. Analysis about requirements of large-scale complex system indicates that requirements expose the characteristics of self-adaptive, collective evolution, and emergence. Now plenty of Web 2.0 applications have fully showed the above-mentioned characteristics. How to explore the novel RE techniques to answer complex situation for the development of service-oriented software? How to facilitate the subsequent automated and on-demand services software delivery just in time through attaching semantic information? Especially, how to actively customize service production according to stakeholders’ needs? These issues are the fundamental motivation of this paper. In our previous work, we have partly investigate these questions and argued that requirements acquisition should fully make use of stakeholders participation and collective intelligence [3]. Through collaborative platforms of social software such as semantic wikis and exploiting simple annotation function plus evolutionary mechanism, requirements specification and requirements semantics will efficiently and gradually emerge. At the same time, requirements acquisition process is continuous in accordance with networked software features, i.e., on-demand change and continuous growth.

The rest of this paper are organized as follows: We explore semantic-enabled RE for service software in Section 2.
Section 3 investigates the active customization conducted by process semantics for services through Web 2.0 plus semantic web with a paradigm shift from provider-centric to user-centric service production. In Section 4, related work and the comparison with our work is presented. At last, we draw up our conclusions with future work directions in Section 5.

II. SEMANTIC-ENABLED RE FOR SERVICE

RE plays a key role in software development projects, and the main purpose of RE is to transform stakeholders’ unordered requirements statements into ordered requirements specifications. Requirements specifications generally adopts two methods: natural language or a formalized way. Natural language is not accurate and always has some ambiguities. While for normal users, it is difficult to master the formalized methods. Through adding the semantics of the key elements to the requirements, i.e., describing concepts and concept relationships explicitly, users needs can be strictly defined in the formalized specification. Adopting ontology language to express requirements semantics, and asking stakeholders to annotate requirements can reduce their learning curve to a certain extent. Namely, requirements semantics represent the knowledge about all the users’ needs through knowledge capture supported by ontology. Requirements semantics are integrated in the whole production of service software (see Figure 1), including requirements acquisition, requirements refinement, and semantic-enabled customized service production.

Requirements semantics obtained from requirements evolve constantly. Stakeholders-driven requirements semantics acquisition should design conceptual model (i.e., domain requirements ontology) by domain/requirements experts, then extract folksonomic semantics from collaborative tagging according to various baselines in order to revise the conceptual model, and form the SORSO (Service-oriented Requirements Semantics Ontology) [4] which combines the experts and folks intelligence. Next, under the guidance of SORSO, stakeholders and requirements engineers annotate and reason on the model to generate requirements semantic artifacts.

With the refinement of requirements semantics, we can get RSO (RSO, Requirements Sign Ontology) [5] which provides a semantic description for business process. Virtually, RSO is the knowledge related to the process (workflow) from requirements semantics. According to the services and related control flow (e.g., sequence, split, join etc.) in RSO, services will be aggregated by adopting cluster match strategy to achieve networked software production towards local requirements assets and distributed service resource.

Definition 1 (Requirements Sign Ontology, RSO). RSO consists of 3 elements $\langle$ ReqSemantics, Concept, Control $\rangle$, where ReqSemantics represents the acquired requirements semantics; Concept represents the set of extracting concepts from ReqSemantics; Concept $\supseteq$ DPO.C: Control represents control structure among matched service ontologies, mainly include sequence, choice, split-union, any order, cycle. In a sense, RSO represents the knowledge about the process (workflow) from requirements semantics.

Consequently, RSO is the requirements-conducted semantics of entire business process (workflow) with service components and control structure.

For unmatched services, customized production can apply inform mechanism (e.g., RSS/Atom) to publish on-demand requirements semantics for services providers. Service providers produce service in accordance with informed semantics in the fly. Requirements semantics come from stakeholders (e.g., users), and service customization will also be user-centric production.

In this way, SRE will greatly upgrade RE quality and level for services. Main differences between service-oriented RE (SORE) and traditional RE lie in that SORE should be intelligent (business) process and service (networked component) effectively. We introduce the meta-description of networked software as shown in Figure 3. Stakeholders are composed of actors who act as difference roles, and their goals are realized by the business processes. Service, as networked component, is the key composition element of business process.

Based on the analysis mentioned above, we believe that requirements semantics will span the entire lifecycle of services software development, and requirements semantics acquisition is also a process of continuous evolution.

To address the diversification and individuation of service requirements, a RGPS meta-model framework [6] is proposed. The framework is composed of four layers: Role layer(R), Goal layer(G), Process layer(P), and Service (S) layer. RGPS-based requirements modeling process will start from the domain analysis of problem space and end in

![Figure 1. Semantic-enabled Requirements Engineering for service][1]
resolving proposal of services.

On the basis of RGPS meta-model framework, user needs can be depicted and portrayed from different perspectives, layers and granularities. Then, the framework will better improve human-network coordination and leverage the interaction and synergy among services to satisfy user individual demand. The framework can provide the guidance for service-oriented requirements modeling, including service-oriented domain requirements (SODR) assets, domain ontology, etc. Domain ontology is structured domain knowledge, which is constructed by domain concepts and its associations, and can be interpreted and utilized by machine.

On the one hand, the framework will conduct 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.

In our previous works [3], we have presented a Requirements Rationale Model (RRM) and related reasoning rules support. Combined with RGPS and RRM, experts-level requirements semantics model for services software (see Figure 4) can be created as an evolutionary base. Goal part of RGPS model directly inherits from Chosen Requirements concept of RRM model. RRM model mainly focuses on pre-requirements reasoning, and RGPS model emphasizes particularly on the relationships from requirements to services. Experts-level requirements semantics model with reasoning rule furnishes elementary requirements description and reasoning support for further requirements refinement. The model spans from goal to service realization. The goals will be translated into processes which can be further matched by services.

We have systematically investigated requirements-driven semantics acquisition through stakeholders participation [4], and semantic-driven service aggregation [5]. The next section will elaborate on semantic-enabled customized service production in details.
III. REQUIREMENTS SEMANTIC-ENABLED CUSTOMIZED PRODUCTION FOR SERVICE

A. stakeholders-centric customized production architecture

Figure 5. process semantic-enabled customized production architecture

SOA (service-oriented architecture) based development approach is a service provider-centric method. Providers produce services and publish description information of service into service registry and repository. Consumers (requesters) search the repository to build the binding with real services. So, traditional SOA is insufficient to support publishing function for consumers. Consumers can only inquire, find, match the service resources that the service provider has already supplied in service repository according to application demand.

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

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

Algorithm 1: Customized service production process–CSPP.

Active service technique has become the hotspot for services computing [7]. 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 publish requirements description of services in the light of business process. Service providers will develop the service products in accord with the semantic description of requested service, and consumers will choose the best service to implement the business process through quality evaluation.

Regarding semantics refinement of service requirements, unmatched services in the course of RSO-centric aggregation stage and unqualified services should publish their semantics extracted from RSO just in time. Providers that have received the information will actively produce the services to achieve requirements-conducted business process and satisfy stakeholders’ demand.

Figure 6. Semantic description for service [8]

Considering the feasibility and simplicity issues, we employ Atom format to express requirements semantics for service and fulfill subscription notification with service semantics by means of subscribe/inform mechanism like RSS. Design solutions of active service customization framework is shown in Figure 5. Service description in Atom is from service semantics in RSO that is derived from requirements semantics of stakeholders, so customized services are also delivered actively for satisfying stakeholders.

The core part is semantic-enabled customized 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 for unmatched services. Evaluator will assess and verify the quality of
matched services.

The flow of service aggregation with customization is presented in Algorithm 1. Unmatched and unqualified services are added 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 achieve the business process.

B. customization platform design

Platform mainly adopt Apache Abdera\(^1\) to generate Atom message and publishing protocol. Informed Atom format is based on the design of service semantics that is shown in Figure 6. Figure 6 shows the extension of Service segment in Figure 3 in details.

Semantics in RSO (Figure 2) are expressed as an OWL-S format. Since the OWL-S file is a bit long, only the instance-tiated description of ‘Inquiry destination’ service segment according to the specification in Figure 6 is excerpted as follows:

```
<process:AtomicProcess rdf:ID="Inquiry destination_AP"/>
<process:comment>RDF is a service about inquiring destination information</process:comment>
<process:hasInput/>
<process:Input rdf:ID="Address name_AP_123456789123456"/>
<process:Output rdf:ID="Inquiry result_AP_legal user "/>
<process:parameterType rdf:datatype="http://www.w3.org/2001/XMLSchema#string"/>
<process:Output rdf:ID="Inquiry result_AP_coordinate "/>
<process:parameterType rdf:datatype="http://www.owlontologies.com/UrbanTransportation.owl#coordinate"/>
<owl:ObjectProperty rdf:ID="hasResult"/>
<rdfs:label>hasResult</rdfs:label>
<rdfs:domain rdf:resource="#Process"/>
<rdfs:range rdf:resource="#Result"/>
<process:hasResult/>
<process:AtomicProcess/>
```

Table I lists the corresponding relationships between every item of entry in Atom format and service semantics in RSO.

According to Table I and the semantic description of service, ‘Inquiry destination’ service in Atom format is as follow:

\(^1\)http://incubator.apache.org/abdera/

The transformation algorithm from semantic description of services to Atom is shown in Algorithm 2.

Based on the Abdera library, thinfeeder\(^2\), a open source RSS reader, can be revised to display Atom and subscribe the services (see Figure 7). Also, adding ‘notify’ function to thinfeeder, 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.

IV. RELATED WORK

Consumer-Centric Service-Oriented Architecture [7] proposes that process and application template etc. are able to publish large-scale customization information and reuse as well as service resource. But the architecture can not support customization and related publishing protocol.

The service-oriented modeling and architecture modeling environment (SOMA-ME) [9] is the first framework for the

\(^2\)http://thinfeeder.sourceforge.net/
model-driven design of service-oriented architecture (SOA) solutions using the service-oriented modeling and architecture (SOMA) method. On 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, SOME-ME platform does not support service customization.

Advertise Web services with Atom 1.0 [10] 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. [10] mainly applies Atom for to discover and search the services that have been published.

Figure 7. Requested web service information displayed in thinReader

An Atom-based architecture for Web services discovery [11] 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 [12], 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 AND FUTURE WORK

Based on semantic-enabled requirements engineering (SRE) for service, this paper mainly explores process semantic-enabled customization technology for active service provisioning. For unmatched and unqualified services at the stage of aggregation, Atom format is adopted to inform service providers in order to support customized production. We have designed an active customization framework and aggregation flow with customization. Customized services are organized in a XML-based queue that can be used to construct Atom format using service2Atom algorithm to inform service providers for on-demand service production in time. At the same time, a prototype platform has been partially implemented and verified against the feasibility and validity of this approach. Requirements semantics are coming from stakeholders, and process semantic-enabled service delivery provides active service customization from requirements semantics, namely on-demand customization for services.

We outline our next steps in two points: (1) design of a quality evaluator (e.g., matching degree to the service requirement requested, and QoS etc.) for services, so that multiple customized services produced by different service providers can be evaluated and compared with each other; (2) feedback mechanism about customized services. This active service customization platform should allow the user to send back their feedback requirements semantically (similar to the semantics of requested requirement) to the service provider after using the services. The feedback loop between service user and provider will act as an iterative process for quality refinement of services.

ACKNOWLEDGMENT

This research has been partially sponsored by the National Basic Research Program of China (973) under Grant No. 2007CB310801, RE4ULS: Requirements Engineering: Fundamentals for Ultra-Large-Scale Systems, and the Natural Science Foundation of China (NSFC) under Grant No. 60970017 and 60950110352, STAND: Semantic-enabled collaboration Towards Analysis, Negotiation and Documentation on distributed requirements engineering.

REFERENCES


