Network Service Registration Based on RGPS Meta-Model in a P2P Network

Yong Feng Huang[1], Lizhi Ying[1], Jian Yuan[1], and Shanyu Tang[2] *

[1] Department of Electronic Engineering, Tsinghua University, Beijing 100084, China

[2] School of Computer Science, China University of Geosciences, Wuhan, 430074, China

*Corresponding author

Abstract:

Service composition-based network software customisation is currently a research hotspot in the field of software engineering. A key problem of the hotspot is how to efficiently discover services distributed over the Internet. In the Service Oriented Architecture (SOA), service discovery suffers from the performance bottleneck of centralised Universal Description Discovery and Integration (UDDI), and inaccurate matching of service semantics. In this paper, the authors describe a novel method for service labelling, registration and discovery, which is based on the Role-Goal-Process-Service (RGPS) meta-model. This approach enables ones to achieve accurate matching of service semantics by extending Web Service Description Language (WSDL) with RGP demand-information. The authors also suggest a P2P-based architecture of service discovery to address the issues in the UDDI bottleneck and the complexity of semantic computation. By adopting the proposed approach, an experiment prototype system has been designed and implemented in Beijing municipal transportation system. The experimental results show the proposed approach is effective in addressing the aforementioned problems.

Keywords: Software Customisation, Network Service, P2P, RGPS Meta-Model

1 Introduction

With the rapid development of the Internet, the patterns of software production have been undergoing fundamental evolution. Traditional software architecture called ‘once developed and keeping on’ has dominated for 40 years. The emergence of Service Oriented Computing (SOC) and Service Oriented Architecture (SOA) is now leading to an era of ‘composite service and change on demand’ mode for network software development.

SOC is a computational mode that employs basic units and network services, to rapidly construct a distributed software system. The essential characteristics of SOC are self-descriptive, self-contained and platform-independent of these elemental network services. SOA is distributed software system architecture proposed for service sharing, service-reuse and business-integration in the Internet environment, and it also provides basic logic architecture for SOC implementation. That is, SOA uses loose coupling of different services through predefined interfaces and constraints to compose software systems instantaneously. With the latest development of service-oriented approach and software architecture, SOA gradually achieves seamless integration of heterogeneous software systems and components. Thus, it shifts software development to large-scale software customisation with a distinctive process consisting of service labelling, registration, discovery and construction. Consequently, a rapid and loose-coupling network software design becomes possible. It is obvious that the cardinal part in networked software development is the service registration and discovery which considerably affect the efficiency of software development as well as reliability.

Currently, the registration and discovery of SOA-based Web services is mainly dependent on Universal Description Discovery and Integration (UDDI) yellow pages to organise the Web services. With the fast development of network services, more and more network services have been published. Therefore, the centralised UDDI cannot seemingly guarantee the users to acquire the needed services conveniently and efficiently. The main reasons for the problem are:

(1) Service-oriented architecture does not guarantee the rational matching between user queries and service description information, nor that can the service user correctly comprehend the information from the provider. Actually, it is an equivalent difficulty that how the service should be understood. For networked software development, particularly, it leads to a problem that how to discover the corresponding services accurately based on user demand documents.

(2) During the construction of distributed networked software, the centralised UDDI has insufficient extensibility and tolerance, as well as the considerable complexity of semantic-based service matching. As a result, the centralised service registration and discovery methods used for semantic-based service discovery are more likely to be the bottleneck in the whole system. It is necessary for the networked software to have the inter-system and inter-media access to the heterogeneous information systems whereas users also need to facilitate the cooperation of the networked software. Thus, the issue in deploying distributed network services must be adequately addressed.

To solve the problems above, we suggest a novel method for network service labelling, registration and discovery, and propose P2P-based distributed network service registration architecture. This paper focuses on an extended semantic Web Service Description Language (WSDL) based on the Role Goal Process Service (RGPS) meta-model, and corresponding service discovery and matching algorithms. The proposed P2P distributed architecture for network service registration is then used to implement a prototype system for Beijing municipal transportation.

The rest of the paper is organised as follows. Section 2 introduces the works relevant to the topic. In Section 3, a new method for service registration based on the RGPS meta-model is presented. The P2P-based network service registration system is detailed in Section 4. Section 5 describes how to implement a prototype system for Beijing municipal transportation. Finally, the paper ends with conclusions and directions of future work.

2 Related work

There have been some research efforts to gain an understanding of semantics and system extensibility in the centralised UDDI Web service registration. Firstly, a lack of semantic information, which deeply affects the efficiency of service discovery, prevents not only Web service functions from being accurately described but also problems like service semantic ambiguity and misinterpretation from being solved. The semantic Web service has been subsequently proposed to address such problems. It unifies semantic techniques and Web service techniques to model Web services by using semantic ontology, and then describes the interface, message, structure and interaction of services in semantic layers. Latest representative works in the area of semantic Web service include OWL-54 and WSDL-5. To enhance the intelligibility of network services, researchers proposed service description based on semantics; ontology-based service description languages, such as DAML-S [1] and OWLS [2], have been developed. These languages do not suffer from the same flaws that Web service functions can only be described in a syntax layer as the standard WSDL does. There are other semantic-based service lookup methods like tree-based classifier [3] and description logic [4]. These methods can improve the precision of matching and the efficiency of service discovery. However, these approaches only slightly improve the precision of service matching. Whereas a comprehensive solution to the problem is to accurately describe the user’s demand for services. We attempt to describe the user’s demand by utilising the RGPS meta-model, and R, G and P elements are used to describe network services through an extension of WSDL language.

Many researchers have been working on the extensibility of network service registration. UDDI 2.0 has some flaws in the extensibility and fault-toleration as well as complexity of semantic-based service matching. As a result, the centralised service registration methods are more likely to be a bottleneck in the whole semantic-based service discovery system. To solve this problem, UDDI 3.0 adopts a rationally centralised and physically distributed mode, which allows the coexistence of multiple registration systems [5]. However, this registration method synchronizes the data across all the systems, bringing about great complexity. Some proposals are based on logically and physically distributed mode, in which registration data are distributed in several independent systems [6] [7]. In these cases, the key issue is how to efficiently manage data among the registration systems. Sheng *et al.* [8] suggested a P2P-based semantic service discovery approach, in which the service discovery consists of two phases, identifying corresponding servers and matching corresponding services. During the service discovery, the service description information is encoded by ontology and the server is located according to the coded information. There are, nevertheless, two main disadvantages with this approach. First, it is difficult to locate the server through the coded information of service, and so the recall ratio of services tends to decrease. Second, as the method is restricted to one ontology database only, it is unsuitable for an open environment where multiple ontology databases coexist. Elenius *et al.* [9] of Linköping University, Sweden, proposed an ontology-based the service discovery method in P2P networks based on JXTA and OWL-S. With their method, the service discovery is divided into two phases, seeking peers that satisfy certain conditions and matching services on these peers based on OWL. However, this method failed to analyse the network structure of the P2P network, nor provide any suggestions on how to construct the P2P network for the service discovery. To take these challenges, we study how to construct a service registration system in non-structural P2P mode, with emphasis on the P2P network service discovery method and performance analysis.

3 Service registration and discovery based on RGPS meta-model

He *et al.* [10] proposed the demand RGPS meta-modelling framework to address the demands of networked software. The framework focuses on the individuation, diversification and variability features of user demands, and applies abstraction to establishing meta-models for the user demand description when networked software is designed. Fig. 1 illustrates the demand RGPS meta-model, which can be divided into four layers, the Role layer, the Goal layer, the Process layer and the Service layer, in a top to bottom manner. Role is the abstraction of organization behaviour, and it is played by different Actors such as Human Actor or Software Agent. The properties of Actors are described by Static Profile and Dynamic Context. Goal is the responsive organisation targets of Role; that is, Role participates in a series of Processes, and takes the corresponding responsibility. According to achievability, Goal can be categorised as Functional Goal and Nonfunctional Goal. The former indicates a function needed by the system, whereas the latter is used to describe how the function is implemented. Goals set by users need to be refined or subdivided, which usually ends at Operational Goal. Process directly specifies an Operational Goal, consisting of four parts: Input, Output, Precondition and Effect. Input and Output represent the data flows of Process conversion; Precondition and Effect stand for the constraint conditions that should be satisfied rigidly in the initiation and termination stages of Process, respectively. Process can be classified as Atomic Process and Composite Process. The latter includes at least one Control Structure, whereas the former does not. Typical Control Structure includes

Sequence, Loop, Choice, Join and Any Order, particularly.

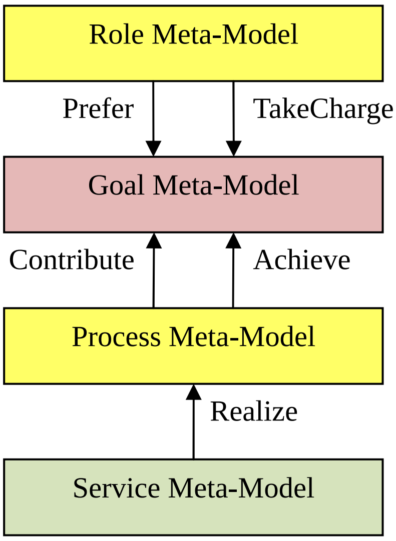


Fig. 1 Demand RGPS meta-model

The relationship between the Process layer and the Goal layer is important in two aspects. Firstly, the Operational Goal has a corresponding Process. Secondly, the Process can achieve a corresponding Functional Goal and contribute a Nonfunctional Goal.

Service has functional and nonfunctional descriptions. The functional description specifies a service Process, whereas the nonfunctional description includes the static QoS property and the dynamic contextual property. A typical QoS property includes response time, cost, reliability, maintainability, availability, etc. A Service can be substituted by other Services. Each Service has corresponding Actors. It can be categorized as Atomic Service and Composite Service.

On the basis of the user demand meta-model above, this paper describes a new method of labelling the network service with Role (R), Goal (G), and Process (P) elements. A novel method for service discovery in P2P service networks is also established by using the R, G, and P information of user demand.

3.1 Network service labelling

In current SOA architecture, Web service labelling is achieved by means of WSDL, which is used to describe the service interface, access address and protocol. The architecture restricts the performance of service discovery process to looking-up through key words, which yields poor efficiency and precision. To address this problem, Deng *et al.* [2] proposed a semantic-based WSDL to extend WSDL information by adding semantic information about operation interface. This extension is used to match service at the semantic level as well as improve the recall ratio. But in the actual process of network service discovery, especially in the process of networked software composition, network service matching still faces problems at the demand information level. For example, guaranteeing the consistency of description format between network service functions and user demands is the key issue at demand level. For this reason, we make further expansion on top of the semantic WSDL. Based on RGPS meta-model of networked software, the R, G, and P elements are added to the extended WSDL model to describe the user demand. This model will guarantee the consistency between the user demand description and the function description of services, thereby improving the efficiency in service discovery.

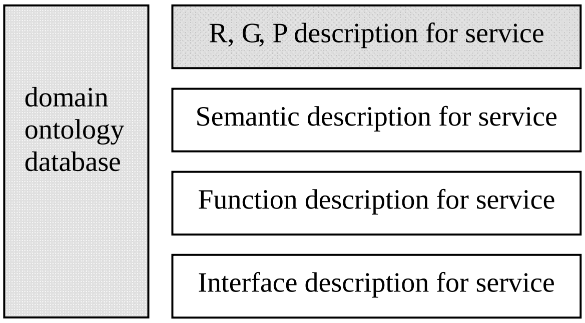

domain ontology database
R, G, P description for service
Semantic description for service
Function description for service
Interface description for service

Fig. 2 Network service labelling based on RGPS

The extended WSDL model with R, G, and P elements is illustrated in Fig. 2. The expansion rules are described as follows.

Firstly, each network service must be labelled with its *Role-layer* information when registered. Role-layer information can be represented by a set R, i.e. $R = \{r, A\}$, where r describes the Role name and A represents the description property set of r. Particularly, r is mandatory, $a_i \in A$, $i = 1, N$, and A is optional.

Secondly, the service is labelled with *G-layer* information. G-layer information is denoted by a set, $G = \{g, NG\}$, where g is a mandatory component that represents the name of service functional goals described by domain ontology. NG is an optional set describing corresponding nonfunctional goals of the service, *i.e.* $NG = \{ng_1, ng_2, ..., ng_m\}$, where ng_i represents the *i*th nonfunctional goal of the service.

Finally, the service is labelled with *Process-layer* information. According to RGPS meta-model, a *Process* can be divided into atomic type and composite type. For atomic process, four elements were used for labelling, i.e. $P = \{In, Out, Pre, Effe\}$, representing *Process* input and output interfaces, and constraint conditions at a *Process* initiation and end, respectively. For composite process, five elements were used, *i.e.* $P = \{In, Out, Pre, Effe, Control\}$, in which $Control \in \{Sequence, Loop, Choice, Join, Any\ Order\}$ represents the structural information of a *Process*.

It is worth noting that the demand information of a network service is represented by $S(R, G, P)$, which conducts labelling based on the domain ontology database intended for providing semantic interpretation in service discovery and matching phases.

3.2 Network service discovery and matching method

As mentioned above, the extending RGPS model on top of WSDL semantic description requires registering services in a P2P distributed system. When a piece of networked software is to be customized, user demand information needs to be submitted in accordance with the RGPS model, and a demand specification document is attained. The networked software customisation platform then conducts service discovery according to the demand document and retrieves services that meet user demands. A piece of networked software can then be combined with these services.

Researchers have proposed a two-phase discovery method based on semantic web services [2]. Furthermore, we propose a three-phase service discovery algorithm based on the extended WSDL model. The three phases are service-demand matching based on RGP, service-interface matching based on semantics, and service-functionality matching. The latter two are not the primary concerns of this study, so the service-demand matching algorithm is our focus.

The essence of RGP-based service-demand matching is to match the demand specification document with the RGP information of service. If $S(R, G, P)$ represents the demand information of a network service, and *ServiceProfile* denotes the demand specification document, then the matching process can be described as follows: if $S(R, G, P)$ belongs to a subset of user demand *ServiceProfile*, the service can partially or wholly satisfy the user’s demand. Further matching of the latter two phases can then be carried out by using semantic WSDL registration information.

In the service-demand matching phase, it is necessary to conduct semantic matching between the demand specification document and the service registration information, and carry out different procedures in accordance with the similarity degree among concepts. This can be achieved by using the domain ontology database in compliance with the RGPS meta-model and rules for user demand description.

In the demand specification document, *R-layer* demand information contains a term C_1, and the *R-layer* of a service registration contains a term C_2. The similarity degree between the ontology terms, $SimCC(C_1, C_2)$, can be calculated by using equation (1).

```
IF $SimCC(C_1, C_2) = 1$
    $C_1 = C_2$;
ELSE IF $C_1$ is a leaf of ServiceProfile;
    $C_1 \neq C_2$;
ELSE
    Search next term in ServiceProfile;
ENDIF
```

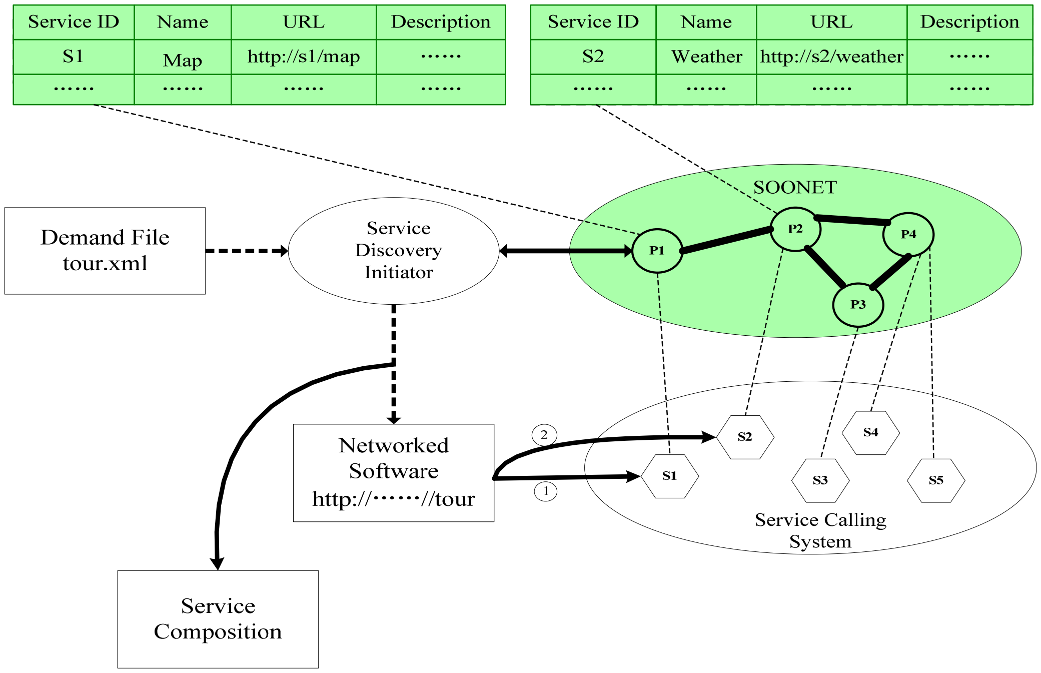
A graph-based method is used to calculate the similarity between the terms, given by

$$SimCC(C_1, C_2) = \begin{cases} e^{-ai} \tanh \beta h & C_1 \neq C_2 \\ 1 & C_1 = C_2 \end{cases} \tag{1}$$

where, l is the length of the shortest path between two terms, h is the height of the upper-level term, and β and α are used to adjust the influence degrees of l and h, respectively. In general, $\alpha = 0.2$ and $\beta = 0.6$ yield better results [6].

4 P2P-based service discovery system

The centralised service discovery architecture in UDDI 2.0 suffers insufficient extensibility and reliability. In order to build a robust, scalable environment for networked software customisation, we propose a registration system for non-structural P2P networks in this section.



Service ID	Name	URL	Description
S1	Map	http://s1/map	······
······	······	······	······

Service ID	Name	URL	Description
S2	Weather	http://s2/weather	······
······	······	······	······

Fig. 3 Architecture of a P2P-based network service discovery system

4.1 Architecture of a P2P-based service discovery system

An unstructured P2P network discovery system is composed of many servers that store service registration information. So the system can be logically divided into two layers, *i.e.* service discovery and service calling, as shown in Fig. 3. P1 and S1 in the figure represent nodes belonging to the discovery system and the calling system, respectively. In the process of implementation, they could be a server providing two interfaces and functionalities for different systems. If a server wishes to share services in a service discovery system but does not intend to join the system, its service description information can be registered with other nodes like *S*5 in Fig. 3.

4.2 Parallel random-walk search algorithm

The P2P service discovery is conducted by using the *k-Walker* search algorithm. All parallel random walks will stop if one of them finds the proper service. So a synchronization mechanism among the walks is necessary. It uses the following mechanism to achieve synchronization: the system consults whether the source node has received responses after walking every four steps [11]. A positive result means the search is successful, and the random walk should stop then; otherwise, the walk continues. Analogously, the service discovery system contains a node-status maintenance mechanism. That is, each node in the network caches the ID of search information it has forwarded along with the next neighbour node. If the node receives the same information again, it only forwards the information to neighbours that have not received.

It is possible for the degree of the source node to be smaller than *k*, especially when *k* is a large number. In this case, the source node will send duplicate search information to some neighbours, thus wasting great network bandwidth. To avoid such a situation, the proposed P2P service discovery system introduces a token mechanism. Each service-discovery information contains a token number set by the source node. The sum of the numbers for all the search information is equal to *k*. Nodes receiving the search information can forward it to multiple neighbours according to the token number, but must comply with the rule that the sum of all the forwarded token numbers is equal to the number of the received tokens.

The proposed *k-Walker* search algorithm is an optimal random-walk search algorithm independent of the network structure. It initialises *k* searches simultaneously in an attempt to overcome larger search delay due to random-walk. In order to further reduce the searching cost of network service discovery, each node temporarily caches the ID of the search information it has forwarded and the next neighbour node; if the same information is sent to the node, it will be forwarded to other neighbour nodes.

4.3 Strategy for service registration information duplication

In order to improve service discovery efficiency and fault-toleration, P2P networks usually duplicate service information to improve performance. It is shown below that the $r(t)$~$q^{1/(a+1)}$ power distribution of service duplication number in a service discovery system yields the smallest search delay. So in this section we formulate a strategy for duplicating registration information along the discovery path to optimise the performance of P2P service discovery.

Theorem 1: In a network G(V, E), WS_{i} is an arbitrary network service whose search rate is q and the number of duplication at time t is $r(t)$. Assuming the search delay of WS_i is T_q~$r(t)^{-a}$, $a \in [0.66, 0.68]$, and each search replicates $c*T_q$ duplication of WS_i (c is a constant), $r(t)$ satisfies the following equation:

$$r(t) \sim q^{1/(a+1)} \tag{2}$$

Proof: Suppose $T_q = mr(t)^{-a}$, where m is a constant. In each search process, the possibility of the target service, WS_i, is q. Consequently, the increment in duplication of i is given by

$$\frac{\partial r(t)}{\partial t} = qcm \cdot r(t)^{-a} \tag{3}$$

Equation (3) can be rewritten as:

$$r(t)^a \partial r(t) = qcm\partial t \tag{4}$$

Calculating integrals for both sides, and assuming the initial duplication number is 1, i.e. $r(0) = 1$, it has

$$\begin{aligned} r(t) &= [(a+1)qcmt+1]^{1/(a+1)} \\ &\approx [(a+1)qcmt]^{1/(a+1)} \sim q^{1/(a+1)} \end{aligned} \tag{5}$$

Since there is an asymptote for large t, the above approximation is valid when t is sufficiently large.

Theorem 1 indicates that when the duplicating ratio of service registration information is proportional to the search delay, the distribution of duplication numbers follows $r(t)\sim q^{1/(a+1)}$, which yields the smallest delay with the k-Walkers search algorithm.

To further demonstrate Theorem 1, we performed a simulation whose results are shown in Fig. 4. Here, $Unif()$ represents the uniform distribution of the service:

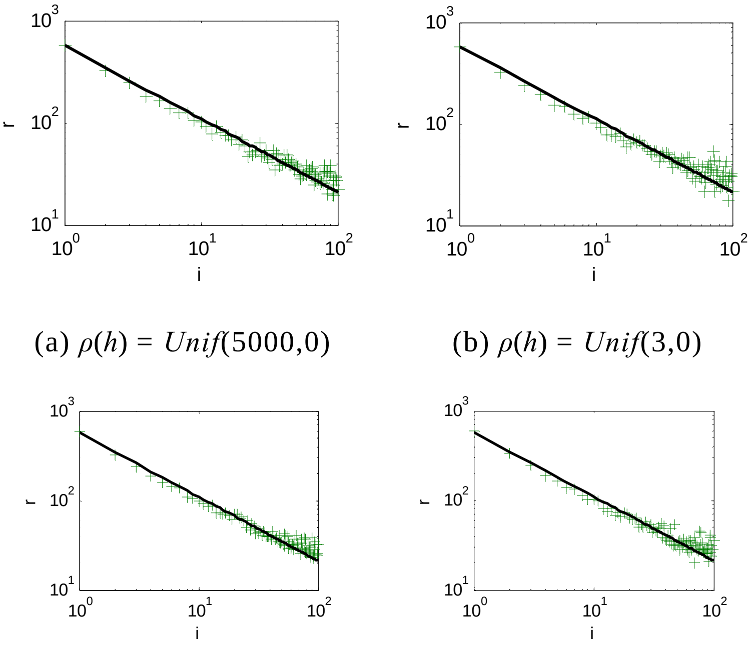
$$\rho(h) = Unif(h_0, d) = \begin{cases} \frac{1}{d}, & h_0 - \frac{d}{2} \le h \le h_0 + \frac{d}{2} \\ 0, & otherwise \end{cases} \tag{6}$$

where h is the fitness, h_0 is the initialization fitness, and d is the distance between nodes. And, $Pow()$ represents the power exponent distribution of the service:

$$\rho(h) = Pow(h_0, r) \sim \left(\frac{h}{h_0}\right)^{-r}, \quad h = h_0, 2h_0, \cdots \mathrm{L} \tag{7}$$

where L is the average path length, $\rho(h)$ denotes the distribution function of the initial fitness, m_0 denotes the initial nodes of SOONET network, m denotes the incremental speed of the network connections in SOONET, and r denotes the number of duplication generated by the delivering nodes. When $h_0 = m_0$, $\rho(h) = Unif(h_0, d)$ and $\rho(h) = Pow(h_0, r)$, indicating that the Web Service Server stores the non-native Web Service descriptive information. On the other hand, $\rho(h) = Unif(m_0, d)$ implies the similar intrinsic attribute and inferior heterogeneity of the Service Servers in the system. On the contrary, $\rho(h) = Pow(m_0, r)$ means the mutual exclusive intrinsic attribute and significant heterogeneity of the Service Servers in the system.

Figs. 4 and 5 show the results for the simulation experiments, where the network parameters are N = 5000, m_0 = 7, and m = 3, and the parallel walk number is k = 32. Parameters related to the service registration information are: M = 100, $q_i \sim i^{-1.2}$, c = 0.1, and a = 0.68. The solid lines in the figures are based on theoretical calculations, whereas the plus signs are simulation results.



(a) $\rho(h)$ = *Unif*(5000,0)

(b) $\rho(h)$ = *Unif*(3,0)

(c) $\rho(h)$ = *Pow*(5000,3)

(d) $\rho(h)$ = *Pow*(3,3)

Fig. 4 Numbers of files when using the duplication strategy

The results show the simulation data approximately coincide with the theoretical curves, indicating that the duplication mechanism of service information produced the optimal duplication number distribution. This strategy is comparable to a negative feedback process. Suppose WS_{i} is a service in the service discovery system. If a search for WS_i yields a long search path, then the description information of WS_i will be mass-duplicated along the path, which reduces the path length of the next search. With this approach, the search delay could be optimised by assigning each service a suitable duplication number.

5 Prototype implementation and performance analysis

5.1 System architecture

Using the proposed algorithms and methods above, we designed and implemented Service-Oriented Overlay NETworks (SOONET), a non-structural P2P service-discovery prototype system for Beijing municipal transportation. As a service registration and discovery subsystem, SOONET is an integral part of the networked software customisation platform, as shown in Fig. 5. The nodes in SOONET represent the servers that host network services and their registration information. SOONET constructs a non-structural P2P network among these servers to realize service discovery. Each server node stores several services and corresponding registration information, as well as maintains a message queue and a daemon proceeding. Query service accepts a query from the discovery-initiator, and then inserts a new search message into the queue. RWQuery service implements the discovery algorithm, accepts search information from other SOONET nodes and inserts it into the queue. The daemon proceeding manages and forwards messages in the queue FIFO. SOONET nodes also contain a Ping service to implement the Ping-Pong mechanism which manages dynamic neighbours for their addings and leavings.

When SOONET was implemented on network services, each node in SOONET acted as a Universal Resource Locator (URL). Multiple SOONET nodes on the same server were deployed to construct an experimental system with a large number of nodes. The experimental environment consisted of ten hosts with each containing ten SOONET nodes. Each of the 100 nodes had a local network service, and could cache 4 non-local services information. One of the ten hosts also played the role of the service-discovery initiator that sent search requests to other nodes. The target node of a search request was chosen randomly by the initiator.

In the prototype system, we packaged 30 atomic services, labelled them with the extended WSDL description model and registered them with different SOONET nodes. Table 1 lists some of the services. Other service-related information, such as WDSL, is omitted. It should be noted that all the terms used in R, G, P information were obtained from an ontology database of Beijing municipal transportation.

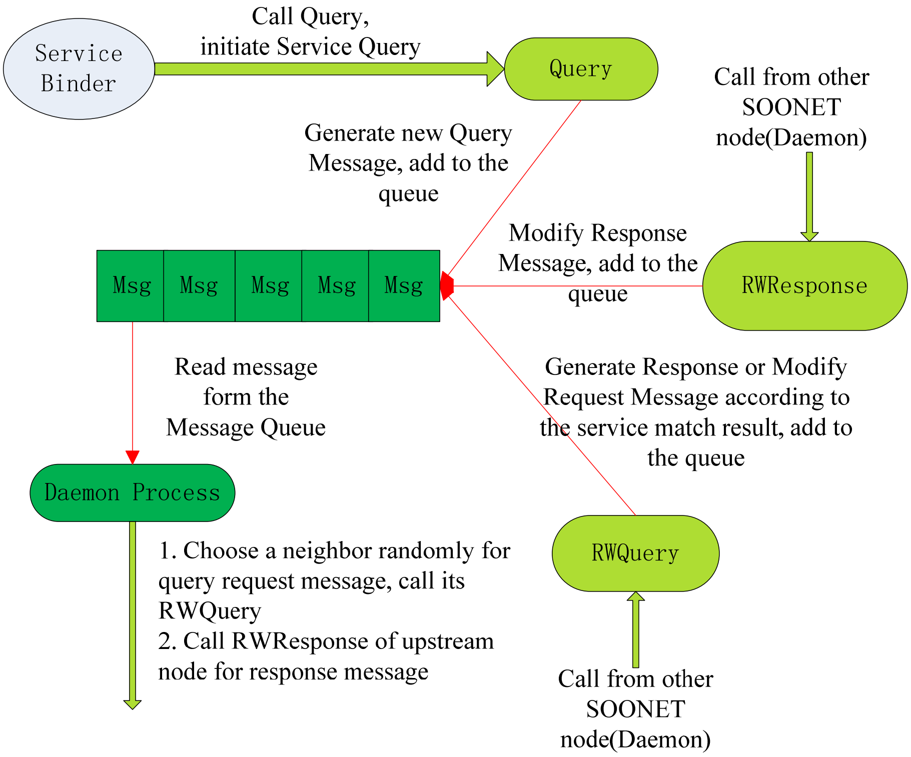

Service Binder
Call Query, initiate Service Query
Query
Call from other SOONET node(Daemon)
Generate new Query Message, add to the queue
Modify Response Message, add to the queue
Msg
Msg
Msg
Msg
Msg
RWResponse
Read message form the Message Queue
Generate Response or Modify Request Message according to the service match result, add to the queue
Daemon Process
1. Choose a neighbor randomly for query request message, call its RWQuery
2. Call RWResponse of upstream node for response message
RWQuery
Call from other SOONET node(Daemon)

Fig. 5 Experiment topology of SOONET

Table 1 Typical network services implemented in SOONET

Name	Role	Goal	Process layer information				Data source
			In	Pre	Out	Effe	
Stadium information	Audience	Consult stadium	Stadium name	Text	Introduce	Text	www.beijing.cn
Olympic game	Audience	Schedule	Date	Text	Schedule	Text	www.beijing.cn
Bus path	Traveller	Bus line	Start, end	Text	Path	Picture	www.mapabc.com
Self-drive	Driver	Travel path	Start, end	Text	Path	Picture	www.mapabc.com
Travel by Taxi	Traveller	Travel path	Start, end	Text	Path	Picture	www.mapabc.com
Weather	Traveller	Consult	Place, date	Text	Weather	Voice	qq.ip138.com/weather

Name	Role	Goal	Process layer information				Data source
			In	Pre	Out	Effe	
Stadium information	Audience	Consult stadium	Stadium name	Text	Introduce	Text	www.beijing.cn
Olympic game	Audience	Schedule	Date	Text	Schedule	Text	www.beijing.cn
Bus path	Traveller	Bus line	Start, end	Text	Path	Picture	www.mapabc.com
Self-drive	Driver	Travel path	Start, end	Text	Path	Picture	www.mapabc.com
Travel by Taxi	Traveller	Travel path weather	Start, end	Text	Path	Picture	www.mapabc.com

5.2 Performance analysis

We evaluated the performance of the service discovery system in terms of service discovery efficiency and service search performance.

5.2.1 Service discovery efficiency

Twenty user demand specification documents were submitted to the software customisation platform, and employed to test the performance of service discovery. The recall ratio, precision and response time were measured in the following service information registration modes: basic WSDL information; semantic WSDL information, and Extended WSDL information.

Table 2 Three cases of WSDL

Information type	Recall ratio	Precision	Balanced F measure	Response time
	r	p	$F_{\beta=1} = 2rp/(p+r)$	(ms)
Basic WSDL	82%	88%	84.9%	100
Semantic WSDL	96%	81%	87.9%	150

Extended WSDL	95%	91%	93.0%	170

As Table 2 shows, the Basic WSDL method had the lowest recall ratio, mainly due to its key-word matching. Semantic WSDL and Extended WSDL gained better performance because both using ontology to match a service. On the other hand, the inaccuracy of the semantic matching algorithm had a negative impact on the recall ratio of Semantic WSDL labelling. Extended WSDL improved accuracy with its extended RGPS demand information and the longer response time was caused by greater complexity under a larger amount of matching conditions. However, the distributed computation of the P2P service discovery system seemed effectively alleviating the bottleneck, which was indicated by the relatively smaller increase in response time for Extended WSDL.

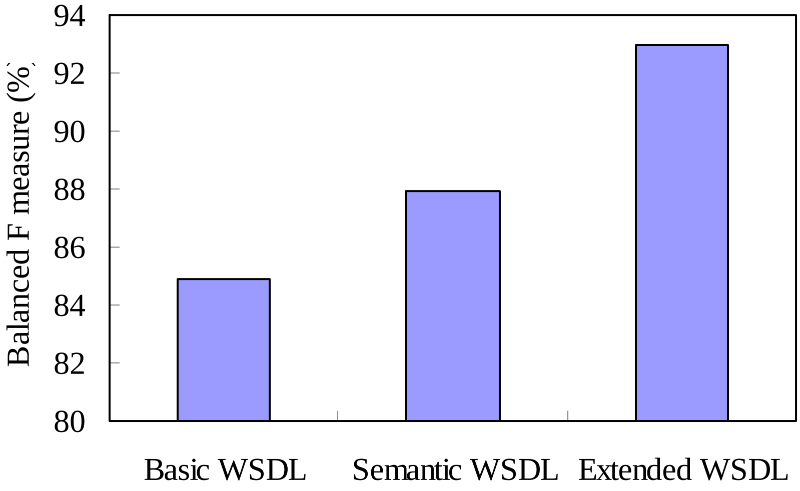


Fig. 6 Comparisons of service discovery efficiency

Fig. 6 shows comparisons of service discovery efficiency between Basic WSDL, Semantic WSDL and Extended WSDL. The balanced F measure [12] is a weighted harmonic mean of recall ratio and precision, and also a single measure that trades off precision versus recall. According to Fig. 6, Extended WSDL had the best service discovery efficiency among three WSDL methods, and Basic WSDL had the poorest service discovery efficiency.

5.2.2 Service search performance

Since all the hosts in our experiment were located in the same LAN, the search delay was not representative of the actual situation on the Internet. Therefore, we used the hop number as a performance indicator instead of search delay. First, we studied the hop number in different duplication strategies. With the recall ratio being set to q_i~$i^{-1.2}$, we considered the following strategies:

- ✧ Uniform duplication: description of each service is randomly duplicated onto 4 nodes in the network.
- ✧ Proportional duplication: description of the target service is conducted by the initiator of each search attempt.
- ✧ Duplicate service information along the search path, once every 10 hops.

The initiator sent out 2000 search queries while only recording the search hop number of the latter half, as the distribution of duplication in the network was unstable during the first 1000 times. The results are presented in Table 3, from which it is clear that the strategy of duplication and caching service information along the search path yielded a significantly less hop number than the other two approaches. Note that in Tables 3 and 4 *N* denotes the number of nodes in the network, m_0 denotes the number of initiation nodes, and *m* denotes the increasing link number when a new node has been inserted.

Table 3 Numbers of search hops under different strategies

Network topology (N = 100, m_0 = 5, m = 2)	Uniform duplicate	Proportional duplicate	Path duplicate
$\rho(h)$ = *Unif*(100,0)	8.43	7.62	5.30
$\rho(h)$ = *Unif*(2,0)	7.12	8.31	5.48
$\rho(h)$ = *Pow*(100,3)	8.40	7.90	5.32
$\rho(h)$ = *Pow*(2,3)	7.51	8.04	5.29

5.2.3 Relationship between search-hop and parallel random-walk

Table 4 shows the relationship between the search-hop number and the parallel random-walk number k. Five nodes in the network were randomly chosen to cache the description of the target service. An arbitrary node was then selected to initiate the query. The experiments were repeated for 1000 times and the numbers of search-hop were also compared. The results in Table 4 indicate that the number of search-hop decreased with increasing k, and the decreasing trend slowed down when k increased.

Table 4 Relation between the number of search hop and the parallel random-walk number k

Network topology (N = 100, m_0 = 5, m = 2)	k = 1	k = 2	k = 3	k = 4	k = 5	k = 6
$\rho(h)$ = *Unif*(100,0)	11.33	5.25	3.76	3.01	2.71	2.44
$\rho(h)$ = *Unif*(2,0)	11.02	5.46	3.82	3.37	2.76	2.59
$\rho(h)$ = *Pow*(100,3)	11.06	5.19	3.77	3.08	2.79	2.46
$\rho(h)$ = *Pow*(2,3)	10.99	5.46	3.69	3.04	2.71	2.45

5.2.4 Performance comparisons between k-Walkers random-walk and Flooding

Table 5 shows the comparisons in search performance between k-Walkers and Flooding, under the service distribution conditions of uniform (*Unif*()) and power exponent (*Pow*()), respectively. In Table 5, each column contains two records: the former indicating the performance of Flooding and the latter k-Walkers. Besides, S_q denotes the searching success rate; T_q denotes the searching delay; M_q denotes the searching expenditure. Close analysis of Table 5 reveals, under the critical demand on high searching success rate, the searching expenditure of k-Walkers was less than that of Flooding by order of magnitude. The searching delay of k-Walkers was less significant compared with Flooding; however, they both had the same order of magnitude.

Table 5 Performance comparisons between Flooding and k-Walkers in SOONET network

Model parameters	S_q	T_q	M_q
$\rho(h)=Unif(5000,0)$	0.99 / 1	4 / 11.21	3071 / 560.5
$\rho(h)=Unif(3,0)$	1 / 1	4 / 14.48	10038 / 724
$\rho(h)=Pow(5000,3)$	0.99 / 1	4 / 17.01	4431 / 850.5
$\rho(h)=Pow(3,3)$	1 / 1	4 / 13.16	9086 / 658

From the experimental results above, we could deduce that the demand of the distributed service discovery system towards Web Service has two remarkable features: first, high searching success rate is badly needed (approximate 1); second, there is no rigid limitation on searching delay. Therefore the experimental results show the performance of the k-Walkers is superior to the Flooding, when employing by our proposed distributed service discovery system with a non-structural overlay network.

6 Conclusions

This study has explored two improvements in service discovery, namely a new method of service registering and the architecture of a service discovering system. A novel service labelling method based on the RGPS meta-model has been proposed in this paper. We have also suggested the architecture of the service discovery system in an unstructured P2P network. A prototype system, SOONET, has been designed through employing the proposed methods and successfully implemented in Beijing municipal transportation. The performance evaluation on SOONET has verified the efficiency of our proposed methods.

Future work will be focused on two issues. One is the security question which is to valid the registering service. The other is to improve the tolerance performance when some nodes are invalid in SOONET.

7 Acknowledges

This work was supported in part by grants from Tsinghua University Initiative Scientific Research Program (No.20111081023), the National Natural Science Foundation of China (No. 60970148), the National Foundation Theory Research of China (973 Program, No. 2007CB310806), and British Government (No. ktp008263).

8 References

[1] H. Foster, S. Uchitel, J. Magee, and J. Kramer, “Compatibility verification for Web service choreography”, In Proc. of IEEE International Conference on Web Services, 2004, pp.738-741.

[2] S. Deng, J. Wu, Y. Li, and Z. Wu, “Consideration of Operation Composition in Semantic Service Matchmaking”, In Proc. of IEEE Int. Conf. on Services Computing, 2006, pp. 249 - 252.

[3] I. Budak Arpinar, B. Aleman-Meza, R. Zhang, A. Maduko, “Ontology-driven Web services composition platform”, *Information Systems and E-Business Management*, Vol. 3, No. 2, 2005, pp.175-199.

[4] C. Doulkeridis, and M. Vazirgiannis, “Querying and Updating a Context-Aware Service Directory in Mobile Environments”, In Proc. of IEEE/WIC/ACM International Conference on Web Intelligence, Beijing, 2004, pp. 562-565.

[5] S. Pokraev, J. Koolwaaij, and M. Wibbels, “Extending UDDI with Context-Aware Features Based on Semantic Service Descriptions”, In Proc. of Int. Conf. on Web Service, Las Vegas, USA, 2003, pp. 184-190.

[6] R. Khalaf, and F. Leymann, “On Web Service Aggregation”, In Proc. of 4th Int. Workshops on Technologies for E-Services in Conjunction with the VDLB Conf., LNCS2819, Berlin, German, 2003, pp. l-13.

[7] Z. Maamar, S.K. Mostefaoui, and H. Yahyaoui, “Toward an agent-based and context-oriented approach for Web services composition”, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, No. 5, 2005, pp. 686-697.

[8] Q. Sheng, B. Benatallah, M. Dumas, and E.O. Mak, “A Platform for Rapid Composition of Web Service in a Peer-to-peer Environment”, In Proc. of 28th Int. Conf. on Very Large Databases, Hong Kong, 2002, pp. 1051-1054.

[9] D. Elenius, and M. Ingmarsson, Ontology-based service discovery in p2p networks, The First International Workshop on Peer-to-Peer Networks, 2004, pp. 114-123.

[10] K. He, R. Peng, W. Liu et al., Networked Software. Beijing: Science Press, 2008, pp. 30-69.

[11] K. Zhang, “Modelling and Analyzing of Unstructured Overlay Networks for Service Discovery Application”, Dissertation submitted to Tsinghua University for Degree of Doctor, April, 2009.

[12] C. D. Manning, P. Raghavan, and H. Schütze, Introduction to Information Retrieval. Cambridge University Press. Cambridge, England, 2008, pp. 155-156.