A Triplex-layer Based P2P Service Exposure Model in Convergent Environment

Cuiting HUANG*, Xiao HAN*, Xiaodi HUANG**, Noël CRESPI*
*Institut Mines-Telecom – Telecom Sudparis
9, Rue Charles Fourier, 91000, Evry, France
** Charles Sturt University, Albury, NSW 2640, Australia
*{cuiting.huang, han.xiao, noel.crespi}@it-sudparis.eu, **xhuang@csu.edu.au

ABSTRACT
Service exposure, including service publication and discovery, plays an important role in next generation service delivery. Various solutions to service exposure have been proposed in the last decade, by both academia and industry. Most of these solutions are targeting specific developer groups, and generally use centralized solutions, such as UDDI. However, the reality is that the number of services is increasing drastically with their diverse users. How to facilitate service discovery and publication processes, improve service discovery, selection efficiency and quality, and enhance system scalability and interoperability are some challenges faced by the centralized solutions. In this paper, we propose an alternative model of scalable P2P-based service publication and discovery. This model enables converged service information sharing among disparate service platforms and entities, while respecting their intrinsic heterogeneities. Moreover, the efficiency and quality of service discovery are improved by introducing a triplex-layer based architecture for the organizations of nodes and resources, as well as for the message routing. The performance of the model and architecture is evident by theoretical analysis and simulation results.

Categories and Subject Descriptors
C.2.4 [**Computer-Communication Networks**]: Distributed Systems – distributed applications, distributed databases

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – retrieval models, search process, selection process

General Terms
Management, Performance, Design

Keywords
Service Exposure, Service Publication, Service Discovery, P2P, Convergence, Service Composition

1. INTRODUCTION
Service composition, which enables the creation of innovative services by integrating existing network resources and services, has received considerable attention in these years. This is partly because of its promising advantages such as cost-effective, reducing time-to-market, and improving user experience. As a prerequisite for service composition, service exposure, including service publication and discovery, plays a critically important role in this novel service ecosystem. That is because enabling a service to be reusable implies that this service needs to be known and accessible by users and/or developers. Various solutions to service exposure have been proposed in the last decade. However, these solutions suffer from several limitations, such as convergence, user-centricity, scalability, controllability, and cross-platform service information sharing. Taking these challenges into account, in this article, we propose a distributed and collaborative service exposure model that enables users to reuse existing services and resources for creating new services regardless of their underlying heterogeneities and complexities. This new method could lead to a new paradigm of service composition by providing the agile service discovery and publication support.

The rest of this paper is organized as follows. Section 2 introduces some backgrounds on service exposure, and the relevant mechanisms of service publication and discovery. Section 3 provides an overview of the distributed service exposure model. A mechanism of two-phases based service exposure, a triplex-layer based P2P overlay for service information sharing, the generation processes of different layers, and the process of service discovery relying on this triplex-layer based architecture, are described in Sections 4 to 7, respectively. The performance analysis is provided in Section 8. Finally, Section 9 concludes this paper by discussing the advantages of the proposed solution and mentioning our future work.

2. BACKGROUND AND RELATED WORK
Service exposure plays a significant role in the evolution of service composition, since enabling a service to be reusable means that this particular service needs to be known and accessible by users and/or developers. Various service exposure solutions have been proposed. For instances, using Simple Object Access Protocol (SOAP) or Representational State Transfer (REST) technologies for facilitating the invocation of services, exposing the services through open Application Programming Interfaces (APIs), and adopting service description and publication mechanisms such as Web Service Description Language (WSDL), Universal Description, Discovery and Integration (UDDI) as well as the more recent semantic annotation mechanisms [1][2][3], Web services has gained an immense popularity in a short time. Meanwhile, Telecom operators, menaced by IT competitors, are being forced to open up to both professional and non-professional users in order to retain or expand their service market sharing. Parlay/OSA Gateway, OMA Service Environment (OSE), Next Generation Service Interfaces (NGSI), and OneAPI are all specified by standardization bodies to allow access to Telecom services/capabilities by 'outside' applications through unified interfaces. Industrial solutions, such as British Telecom's Web 21c SDK, France Telecom's Orange Partner Program, Deutsche Telecom's Developer Garden, and Telefonica's Open Movilforum, are proposed by different operators. These operators aim to expose their network functionalities to 3rd party service developers and users by using Web based technologies. In

addition to these solutions, some industry alliances, such as the Wholesale Application Community (WAC), are formed by operators, vendors, manufacturers, and integrators in order to provide unified interfaces to access device functionalities, network resources, and services.

Most of the above current solutions (both in the Telecom and Web worlds) focus on using centralized systems for storing service information and managing service access. Such centralized systems are easy to implement and to monitor their resources. However, they are also the easy targets for malicious attacks, introduce a single failure point, require a high maintenance cost, and cannot be scalable and extensible enough to deal with a dynamic service environment. Moreover, centralized systems limit the interoperation among different platforms. This leads to many isolated service islands, which inevitably incurs resource redundancy.

Overcoming above-mentioned limitations of centralized solutions, decentralized P2P appears to be an obvious choice to support distributed service exposure on a large scale. Recently, some P2P-based solutions have been proposed for enabling distributed service discovery and publication on a large scale. For example, one solution is to group peers into clusters according to their similar interests. HyperCup [4] is an early example. According to predefined concept ontology, peers with similar interests or services are grouped as concept clusters. These concept clusters are then organized into a hypercube topology in a way that enables to route messages efficiently for service discovery. METEOR-S [5] attempts to optimize registry organization by using a classification system based on registry ontology. In this approach, each registry maintains only the services that pertain to certain domain(s). Acting as peers in an unstructured P2P network, the different registries are further grouped into clusters, according to mappings between registries and the domain ontology. Similar solutions classify either registries or service providers, which act as peers in an unstructured P2P network. These peers further form a federation that represents similar interests in a decentralized fashion [6 – 9]. As we can observe from these examples, federation-based solutions are generally related to unstructured P2P architectures. Unstructured P2P networks still have some common issues such as high network traffic, long delay, and a low hit rate, even if the available solutions have addressed these issues to a certain extent. Other alternative solutions based on a structured P2P system have also been proposed. SPiDer [10] employs ontology in a Distributed Hash Table (DHT) based P2P infrastructure. Service providers with good (better) resources in SPiDer, selected as super nodes, are organized into a Chord ring to take on the role of indexing and query routing. Chord4S [11] tries to avoid the single failure point in Chord-based P2P systems by distributing functionality-equivalent services over several different successor nodes. As a hybrid service discovery approach, PDUS [12] integrates the Chord technology with the UDDI-based service publication and discovery technology. Authors of [13] and [14] introduce solutions based on alternative structured P2P topologies such as Kautz Graph-based DHT or Skip Graph. These solutions generally use peers to form a Chord ring that stores all the service information about functionally similar services. As such, these peers are required to have good resources, such as high availability and high computing capacity.

From the literature, we find that most of P2P based solutions are based on unstructured P2P architecture due to its simplicity, robustness, and wide application in the current Internet domain. This is also because unstructured P2P can easily support complex requests, which is an essential requirement for service discovery. Unstructured P2P architectures, however, face challenges such as high network traffic, long delay, and a low hit rate. Because of these limitations in unstructured P2P, structured P2P is naturally selected to address certain issues. Nevertheless, one problem of service discovery based on structured P2P architectures, especially for the most widely used DHT-based solutions, is that the search is deterministic. This implies users must provide at least the exact name information about the resource they want. Such a requirement conflicts with a real-life situation, in which users often have only a partial or rough description of the service they want.

Taking the limitations of both central solutions and the current distributed solutions into account, we propose an enhanced model of P2P based service publication and discovery to improve the efficiency of service discovery on a large scale. This solution reuses the concept of Semantic Overlay Network (SON), which was originally proposed in [15] for the content sharing among nodes. For improving the efficiency of service discovery further, it is extended as a triplex-layer based architecture. This new solution thus enables the service publication and discovery in a purely distributed and collaborative environment.

3. AN UNSTRUCTURED P2P BASED SERVICE EXPOSURE MODEL: OVERVIEW

In our proposed solution, not only traditional Telecom and Web services are exposed, but device-offered services, and user-generated services are also accommodated. When different kinds of services expose themselves to a network, they generally use different technologies, or go through the different service exposure gateways. In order to reuse these existing service exposure technologies and limit the modifications to existing service publication and discovery platforms, we propose an unstructured P2P based architecture. The architecture uses a P2P overlay to share diverse services information, while respecting and maintaining the underlying heterogeneities, as shown in Figure 1.

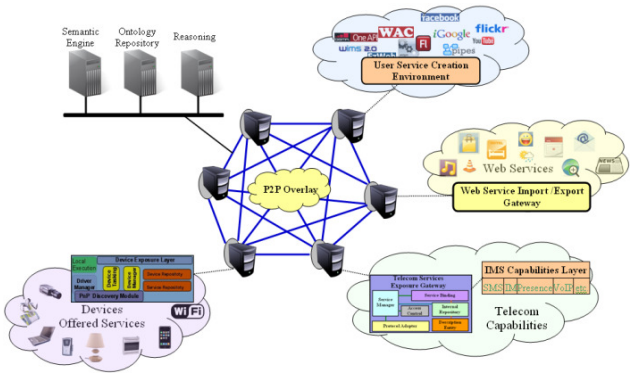


Figure 1. Overall architecture for the P2P based service information sharing system

As shown in Figure 1, this system is composed of several global servers and a number of nodes. The global servers include Semantic Engine, Ontology Repository, and Reasoning Engine, which are in charge of coordinating the representation forms of

the different descriptions of services during a service publication process. The nodes are responsible for the service information storage, service retrieval, service binding, service accessing, and service discovery. Behind these nodes, there are various service exposure gateways for catering to the different kinds of service exposure requirements. Examples are Telecom Service Exposure Gateway, Web Services Import/Export Gateway, Local Device Exposure and Management Gateway, which we introduced in [16], and even some service creation platforms which embed the functionality of enabling personal service publication by end users. These service exposure gateways can be the existing ones that are used in traditional UDDI-based centralized solutions. Within our proposed architecture, however, they are more flexible than what they are used alone in a centralized environment. The reason for this is that the gateways expose their services in a local domain to target a certain group of users or developers using their own technologies; moreover, they also share their service information with users or developers outside their domain by adding some overlay functions into their platforms.

4. LOCAL SERVICE EXPOSURE AND GLOBAL SERVICE EXPOSURE

In our solution, we divide the process of service exposure into two phases: Local Service Exposure and Global Service Exposure.

4.1 Local Service Exposure

After creating a service, service providers or users should make it to be accessible by other users through either the user-centric interface or APIs. They need to generate a service description file of this service in order to be discovered and understood by others. The service description file can be published in a global UDDI registry as a centralized solution does, or published in a local registry residing in the service platform. The latter case is related to Local Service Exposure.

Each service platform contains an internal service repository which contains the service description files for the services held on it. Once a new service is introduced, a service description file is generated. This service description file is then stored in the internal service repository. When a user wants to discover a service or create a personal service through this service platform, she/he can use the local facilities, and search in the local service repository by using the specific mechanisms of service discovery proposed by this service platform. In this context, a great number of local repositories are distributed over the network. They are managed by the respective service providers or service platform providers using the different technologies they prefer. The real situation is that these local registries are generally independent of each other and designed for a specific group of developers and/or a specific network. When a developer or a user wants to create a converged service that involves Telecom, Web, or device-based services, she/he has to search for services from several different service platforms, understand the different service description patterns, as well as their underlying heterogeneities. The need to manipulate different platforms increases the complexity of integrating the heterogeneous services into a unified composite service.

To improve user experience and enhance the reuse of existing services, a mechanism on efficient collaboration among the independent service registries is considered as necessary. Addressing this requirement, we add a complement process to service exposure, which enables seamless collaborations among different service registries. That is the Global Service Exposure.

4.2 Global Service Exposure

Global Service Exposure relies mainly on the Service Exposure Gateway for reporting the internal service information to the large scale P2P based network. That is to say, we assume that a Service Exposure Gateway is added to each service platform. It is used to connect the internal services with outer world application, expose them to external users or applications, and enable sharing its stored service information with other service platforms through the P2P method. An example of such a Service Exposure Gateway is shown in Figure 2.

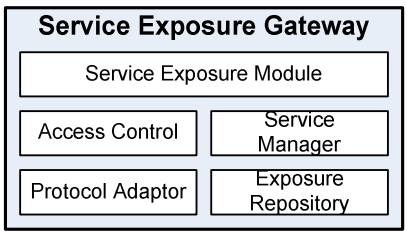


Figure 2. An example of Service Exposure Gateway

When a new service is introduced to a service platform, its service description file is added into the internal repository as we mentioned in Section 4.1. If the provider of this service platform wants to share it with other service platforms, the service description file is sent to the local Service Exposure Gateway. Service Exposure Gateway then contacts the Global Semantic Engine for translating the service description file to a globally-comprehensible format, and for adding the necessary semantic annotations into the description file by consulting with the global Ontology Repository. After that, this unified and semantic enriched service description file is added into the Exposure Repository. Meanwhile, some access control rules and protocol adaptation rules associated with this particular service are also added into the relevant components by the service provider.

For a service publication process, we further introduce two kinds of publications: abstract service publication and concrete service publication. An abstract service is a virtual service that only contains the generalization information about one kind of services (e.g., SMS service). One abstract service can be mapped into several concrete services (e.g., Orange SMS service, Telefonica SMS service and BT SMS service). The abstract service publication is handled by the members of the system administration group. They send a service description file that contains only the generalization information of one type of services to Semantic Engine. Semantic Engine then contacts Ontology Repository for supplementing the necessary semantic annotations. The transformed service description file is then sent to a global Semantic Abstract Service Repository. The concrete service information is published in the local Service Exposure Gateway, as we introduced in the generation process of service description file in the local Service Exposure Gateway.

In order to make concrete services discoverable by other service platforms, each gateway needs to map its concrete services into the corresponding abstract services, create a Local Abstract Service Table, and store this table in the Service Exposure Module. After this, the number of gateways are then interconnected each other through an underlying P2P network in

which they act as peers. This process is considered as Global Service Exposure.

5. A TRIPLEX-LAYER BASED SOLUTION FOR SERVICE INFORMATION SHARING IN P2P ENVIRONMENT

As introduced before, diverse service description files are published in the respective registries residing in the Service Exposure Gateways. The registries share their service information through the P2P pattern for avoiding the single failure point, improving the service discovery efficiency, and reducing the maintenance cost.

In order to improve the query performance and maintain a high degree of node autonomy, a set of Semantic Overlay Networks (SONs) are created over distributed gateways. Essentially, each gateway is connected to a global network as a peer in a P2P system. These gateways share the service information stored in their Exposure Repository through Service Exposure Module. As the description files of services are semantically enriched, the peers that hold these semantic description files can be regarded as semantic-enriched as well. Nodes with semantically similar service description files are then "clustered" together. Consequently, the system can select the SON(s) that is (are) in the better position to respond when a user wants to discover a service. The query is then sent to one of the nodes in the selected SON(s), being further forwarded to the other members of the SON(s). In this way, the query performance is greatly improved, since queries are only routed in the appropriate, selected SONs. This would increase the chance of matching the files quickly with a limited cost.

To implement the above-mentioned SON based semantic service discovery, we propose a triplex-overlay based P2P system as shown in Figure 3.

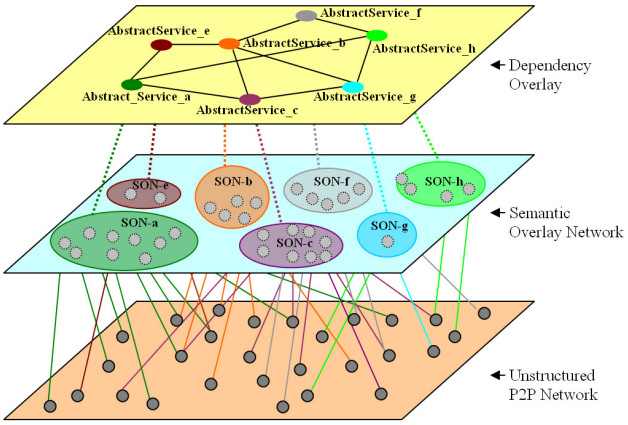


Figure 3. Triplex overlay for P2P based service discovery

In Figure 3, the diverse network repositories, service discovery and publication platforms, service creation environments, and device gateways join the P2P based distributed network. Acting as nodes in an Unstructured P2P Network layer, they interact with each other using blind search solutions (e.g., Flooding based solutions or RandomWalk based solution).

In the above unstructured P2P network, the nodes providing similar resources are clustered together. We call this kind of clusters as SON (Semantic Overlay Network). The benefits of this strategy are as follows. On the one hand, a request is sent to the nodes with a high probability of offering target services, so that the request can be answered quickly. On the other hand, the nodes with a low probability will not receive the request. A waste of resources can therefore be avoided in transferring the request, so that other relevant requests are allowed to be processed in a quick way.

For further improving the efficiency of service discovery, an additional Dependency Overlay is introduced upon the SON layer. This is because different kinds of services may be able to interoperate with each other. For example, the output of one service can be the input of another. Such cooperation among services allows service providers/developers to provide some more meaningful services to end-users. Thus we can infer that the services stored in the same gateway have a very high probability of having certain interdependency amongst each other. The service dependency relationship can also be specified according to some social network information, such as service usage frequencies, users' service usage habits, or some network statistic data. Consequently, defining the dependency among the abstract services, then providing recommendations for the message routing during a service discovery process, will improve the success rate. This definition of service dependency is not limited to one kind of services, but rather includes the dependency among the different kinds of service, such as the devices offered services, Telecom services, and Web services, e.g., the dependency of "camera -> MMS". As the publication of abstract services is performed by the members of the system administration group, the dependency relationship is created simultaneously, once a new abstract device or abstract service is published.

6. SON LAYER GENERATION

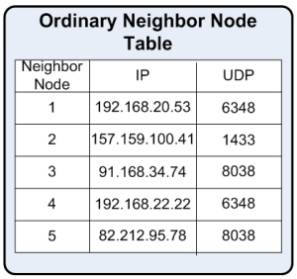
We assume that a set of bootstrap nodes can guarantee the minimal running requirements for the proposed system. These bootstrap nodes form a small scale SON overlay before running the system. That is to say, when an abstract service is introduced to a network by a system administration member, this new abstract service is assigned to a bootstrap node randomly.

Each gateway contains a table called Local Abstract Service Table. This table is created by mapping concrete service description files stored in a local exposure repository, into abstract service profiles stored in the global Abstract Service Profile Repository. In this table, each abstract service entry contains the basic information about the relevant concrete services (e.g. concrete service's name), as well as the links (e.g., a URL) to the corresponding concrete service description files. Based on Local Abstract Service Table, a local gateway can join the relevant SONs automatically. Since several abstract services are contained in one registry, one gateway can join several SONs according to the different abstract services.

To clarify the SON generation process and the update process, we consider two cases in the following: (1) a new gateway is added into the network with a list of abstract services to be exposed; and (2) an existing gateway updates its list of abstract services, which means a new type of service has been introduced to its local network.

When a gateway is introduced to the system, it first joins in the global network through some bootstrap nodes as the ordinary P2P networks do. That is, once receiving the *Join* message from a gateway, the selected bootstrap node broadcasts the *Join* message

to the global network. According to certain neighborhood selection rules (e.g., the solutions introduced in [17] and [18]), some nodes are selected as the logic neighbors for this newly introduced registry and added into the Ordinary Neighbor Node Table (ONNT) as shown in Figure 4. In this example, we assume that each node contains information about 5 neighbor nodes (e.g. IP and UDP port). This process provides another possible way to search a service: if both the SON Overlay based search and the Service Dependency Overlay based search have not found out the relevant services, the system can use the basic Random Walk or Flooding solution according to the information of neighbor nodes stored in the Ordinary Neighbor Node Table.

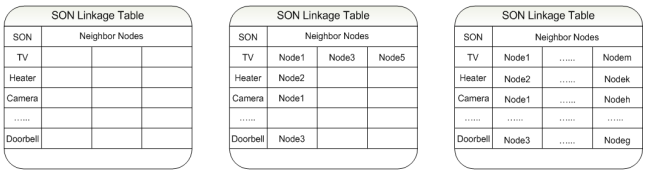


Ordinary Neighbor Node Table

Neighbor Node	IP	UDP
1	192.168.20.53	6348
2	157.159.100.41	1433
3	91.168.34.74	8038
4	192.168.22.22	6348
5	82.212.95.78	8038

Figure 4. An example of Ordinary Neighbor Node Table

We assume that each gateway contains another table called SON Linkage Table (SLT) as shown in the Figure 5, which is used to form the SONs. When a gateway joins the network for the first time, this SLT is empty as shown on the left side of Figure 5. This means the gateway has not joined any SON yet. After joining the unstructured P2P network, this new gateway extracts the names of the abstract devices from its Local Abstract Service Table, and encapsulates them as an *Update* message. This *Update* message is injected to the network by the blind search method according to the neighbor nodes' information stored in the Ordinary Neighbor Node Table. In particular, we use the Random Walk as the basic message routing approach, in which only one neighbor node is selected from the Ordinary Neighbor Node Table for routing the *Update* message. The names of the extracted abstract services in the *Update* message are put into a list, with each entry assigned a time-to-live (TTL) called Time-to-Live for Node (TTL_node). TTL_node is a pre-assigned number *n,* which indicates how many neighbors in the relevant SON have to search for. A global time-to-live called Time-to-Live for Hop (TTL-hop) is also attached to the *Update* message for setting the maximum number of the logic hops for the message.



SON Linkage Table

SON	Neighbor Nodes		
TV			
Heater			
Camera			
.....			
Doorbell			

SON Linkage Table

SON	Neighbor Nodes		
TV	Node1	Node3	Node5
Heater	Node2		
Camera	Node1		
.....			
Doorbell	Node3		

SON Linkage Table

SON	Neighbor Nodes		
TV	Node1	.....	Nodem
Heater	Node2	.....	Nodek
Camera	Node1	.....	Nodeh
.....	.....	.....	.....
Doorbell	Node3	.....	Nodeg

Figure 5. An example of SON Linkage Table in a device gateway

The *Update* message is first forwarded to one of this gateway's neighbors, denoted by *Ni*, and the node that receives this message checks its own SLT. If this table has no entries containing the same name of the abstract service contained in the *Update* message, this request is directly forwarded to another neighbor of *Ni*. As such, only the TTL-hop decreases by 1. Otherwise, if one or several entries in the table contain the same name(s) as the abstract services listed in the *Update* message, both the TTL-node(s) for the relevant abstract service(s) and the TTL-hop decrease by 1. The node information (e.g. IP and Port number) is sent back to the original node. The original node then stores the node information in relevant entries of SLT as a neighbor, as shown in the middle table of Figure 5. After this, the *Update* message is forwarded to one ordinary neighbor of *Ni* node. The above process repeats until either all the TTL-nodes or the TTL_hop have reached 0. If all the entries in the SLT have been populated with information of SON neighbors, the join process for SON is regarded as a successful one; otherwise, the original node selects one of its other neighbors to send the same *Update* message. If all the neighbors of this original node have been selected for forwarding this *Update* request, and some entries in the SLT are still not completely filled, then the original node sends the *Update* message to its bootstrap. This connected bootstrap searches in the bootstrap nodes for finding out which bootstrap node(s) is responsible for the corresponding abstract service(s), and sends information of the relevant bootstrap node(s) back to the original node. The original node adds the information into the relevant entries in the SLT and marks it as bootstrap node's information. This process guarantees that if such an abstract service has been predefined in the network, even the relevant SON has not been created yet; the original node can create a new SON, or join in an existing SON by connecting itself to the bootstrap node directly. Once another neighbor in the same SON is found, the information of the relevant bootstrap node is replaced by the newly found neighbor. This process is repeated on a regular basis to make sure that all the information stored in the SLT is up-to-date. In other words, once either the information of a neighbor node becomes obsolete, or a new abstract service is added into a gateway, a new *Update* message with the names of relevant abstract devices is sent out again to the network and a new join SON process begins.

7. SERVICE DISCOVERY BASED ON TRIPLEX-LAYER BASED P2P SYSTEM

Before introducing service discovery based on triplex overlay, we classify the neighbor nodes of a node into two kinds: the ordinary neighbor nodes and the SON neighbor nodes.

When a user wants to discover a service for the purpose of executing it directly or integrating it into her/his own created composite service, she/he can issue a service discovery request through a user-friendly service discovery front-end or a user-centric service creation environment. We assume that these service discovery and creation frameworks have connected themselves to the global network in advance.

Once a user issues a service discovery request either by entering keywords or natural language based text, or even by selecting the abstract service building blocks, the relevant frameworks formalize a request that contains the target service information (e.g., a *get(target_service, TTL, auxiliary_info)* message). By contacting with the global Semantic Engine, the global Ontology Repository, as well as the global Abstract Device Repository, this newly created request is interpreted into a system recognizable format. The request is injected into the unstructured P2P network through the node that initiates this request. To facilitate the request interpretation process, the relevant service discovery or

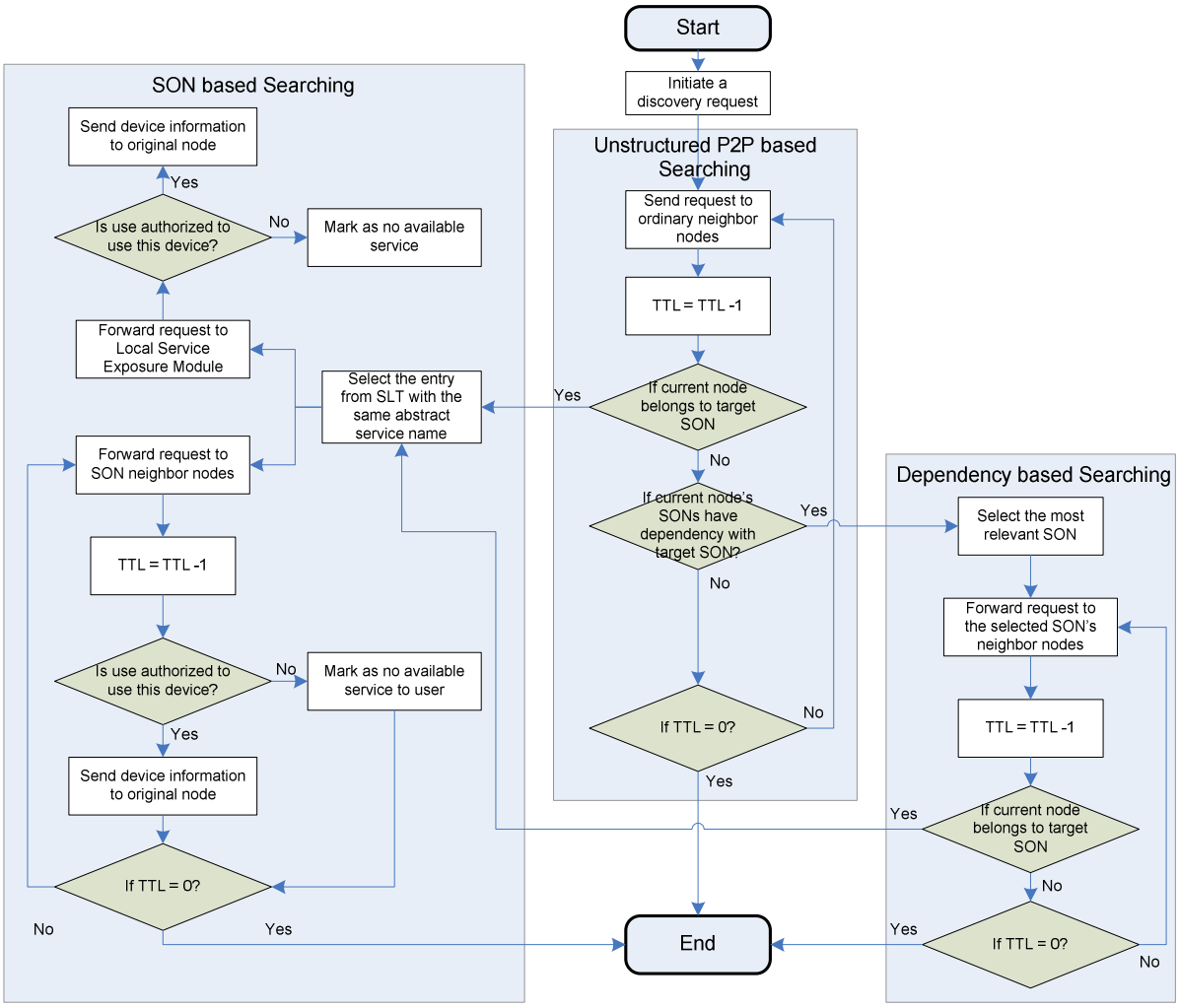


Figure 6. Flowchart for service discovery process in triplex-layer based service exposure model

service creation environment can install a local Semantic Engine and Ontology Repository, and store the abstract service description files in its local repository. In this case the request can be formalized automatically inside its local framework before being injected into the P2P network.

The *get()* request is injected into the network through the blind search method. That is to say, the node, which initiates this request, forward the request to either all its ordinary neighbor nodes or one of its ordinary neighbor nodes, which mainly depend on which blind search strategy (e.g., Flooding or Random Walk) the system adopts. To improve the service discovery efficiency, the system first needs to discover the corresponding SON for the message routing. As shown in Figure 6, the first nodes that receive the service search request will verify if the target abstract service is contained, by comparing the entries' name stored in their Local Abstract Service Tables with the target service's name indicated in the incoming request. If matched, the entry with the same abstract name in the SLT is selected. The received request is then forwarded to the SON neighbor nodes whose IP and Port information is stored in this selected entry. Meanwhile, this node extracts the context information (e.g. user identifier, location, preference, etc.) encapsulated in the received message, and forwards this context information, together with the name of the target abstract service, to its own local Service Exposure Module. Service Exposure Module contacts with Exposure Repository for checking if any services stored in its local gateway can be used by the user, who issues this service discovery request. It then makes a primary filtering for the discovered result according to the auxiliary context information. For example, if a service is set to be public (e.g. a map service), a user can discover this service once the request reaches this gateway. If a service is set to be private, even it matches the functional requirement for the target device, this gateway, however, still needs to verify if the user identifier in the request is included in the identifiers that have been granted by the owner of this gateway for the use of this service (e.g. the user who has subscribed to this service platform). If the original user identifier is matched, the relevant service description file is sent back to the service discovery framework. Otherwise, no service description file is sent back, and this node is marked as no resource matched to user's request.

When the SON neighbor nodes of this node receive the forwarded request, as these nodes are already in the same SON, and this SON is the target SON in which all the nodes contain the service associated with the target abstract service, they thus invoke local Service Exposure Module for local service discovery directly without the need to verify if any SON they belong to match the

target abstract service. At the same time, they forward the request to their SON neighbor nodes, which are extracted from the entry in their SLT. This process continues until the TTL for this request is reduced to 0, which means the service search is terminated.

If the first few nodes that receive the service search request do not belong to the SON for the target service (e.g. Camera), this means that no entry name in Local Abstract Service Table is the same as the abstract name of the target service. In this case, this node needs to discover which neighbor nodes are the most possible ones that in turn find out the relevant SON to which the target service belongs. It is assumed that each gateway contains a copy of such an abstract service dependency relationship file. In order to select the most possible neighbor to match the target SON, the node that does not belong to the target SON extracts all names of the abstract services from its Local Abstract Service Table and the name of the target abstract service from the incoming message. Relying on the dependency file of abstract services, the node then analyzes services dependency relationships among these extracted abstract services, and selects the abstract service whose dependency intensity with the target abstract service is the highest one.

According to the name of the selected abstract service, the node selects the SON neighbor nodes from an entry in SLT whose entry's name is the same as the name of the selected abstract service. The service search request is then forwarded to these selected SON neighbor nodes. When these SON neighbor nodes receive the service search message, they first check their Local Abstract Service Table to see whether they contain the target abstract service in their local gateways or not. If a node finds that it contains the target abstract service, it searches its SLT, and selects the SON neighbor nodes from the entry whose name is identical to the target abstract service. The node then forwards the service search request to the corresponding SON. Finally the service search is limited to the SON that is responsible for the target abstract service. If a node cannot find the target service, it forwards the request to their neighbor nodes that are in the same SON as the first contacted node selected.

If the first contacted node(s) does not belong to the target SON, and it also has no dependency relationship with the target abstract service, the request in the first contacted node(s) is forwarded to its ordinary neighbor node(s) following the blind search method adopted by the underlying unstructured P2P network. When its ordinary neighbor node(s) receives this service discovery request, it repeats the service discovery process we introduced above. This process repeats until either the target SON or a dependent SON has been found out, or the TTL is time out.

8. PERFORMANCE ANALYSIS

In order to evaluate the performance of the proposed service exposure and information sharing model, the theoretical analysis on the success rate and the network traffic impact have been performed. We also compare it with the traditional blind search strategy for the unstructured P2P solutions (which are based on Flooding or Random Walk), as well as the pure SON based solution.

In the following, we use the average messages required for discovering a service as a metric for evaluating our proposed system. This metric can be regarded as an indicator of the invoked network traffic for a service discovery request. In ideal cases, in which only one message exchanges between two neighbor nodes, then the value of this metric is equal to the average number of nodes that are involved in the search process.

To simplify the analysis, we denote the success rate as S(T), and the average message as E(S). We assume that services are distributed in the network evenly and their probabilities being discovered p are the same if the blind search solution is used. The SONs are also assumed to be evenly distributed in the network, and the difference in their sizes is ignored. When a request arrives at a node, the probability that the connected node belongs to the target SON is R. If this node does not belong to the target SON, Dependency Layer makes a recommendation that has the higher probability of finding out the target SON in the next hop. This increased probability is denoted as K. In general, we have K>R. If the request arrives at the target SON, its probability of discovering the target service (q) is much higher than that of searching in the underlying unstructured layer by using the blind search strategy (p). That is because, when a request arrives at a SON, the search space is reduced. But the total number of the user accessible services within this SON remains unchanged, thus the local discovery probability is much higher. Moreover, it can guarantee that all the target resources are clustered into this SON, thereby the nodes outside this SON can be out of consideration.

In the following, the relationships between the success rates and the numbers of the visited nodes for the three solutions of Triplex, SON, and Blind, and their comparisons are illustrated by simulation results with varied values of p, R, q, and K.

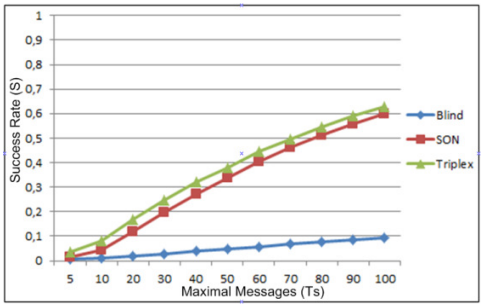


Figure 7. Comparison for the success rates of three types of P2P systems (p=0.1%, R=10%, q=10%, K=50%)

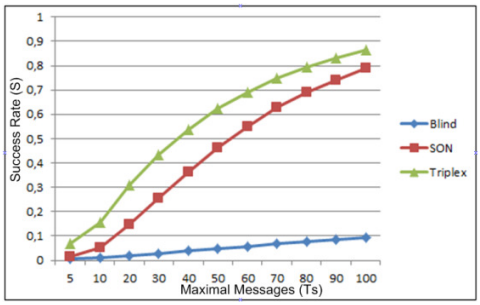


Figure 8. Comparison for the success rates of three types of P2P systems (p=0.1%, R=5%, q=20%, K=50%)

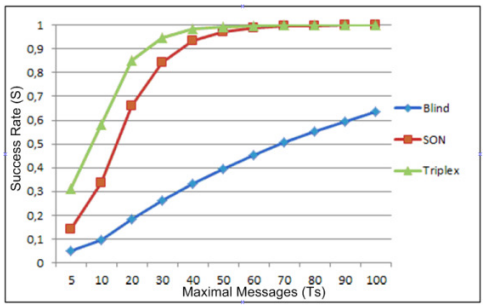


Figure 9. Comparison for the success rates of three types of P2P systems (p=1%, R=10%, q=10%, K=50%)

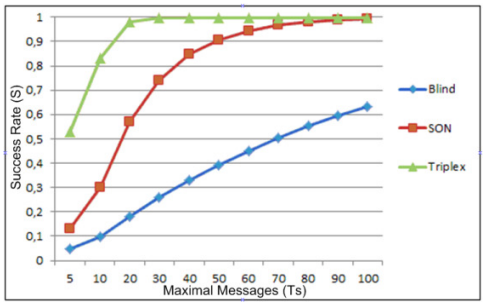


Figure 10. Comparison for the success rates of three types of P2P systems (p=1%, R=5%, q=20%, K=50%)

The simulation (Figure 7 to Figure 10) results show that our proposed solution can greatly improve the success rate with the limited number of messages transferred among the nodes. This improvement is more evident when the popularity of services is low in a network.

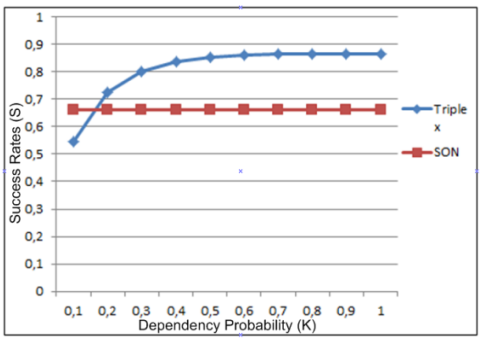


Figure 11. Comparison of the success rates (S) by modifying the dependency probability (K)

Figure 11 illustrates the degree of the influence of the dependency probability *K* on the success rates. The red line with rectangles is the success rate of ordinary SON based system without dependency recommendation, while the blue one is for the SON based system with dependency recommendation.

From the results shown in Figure 11, we can observe that as the dependency probability increases, the success rate increases accordingly. However, after K is more than 0.5, this increase rate becomes very slow. Moreover, only when K is bigger than a certain level (in our case, when R=10%, K needs to be bigger than 15.3%), our solution can improve the service discovery performance of the system; otherwise it may weaken the performance.

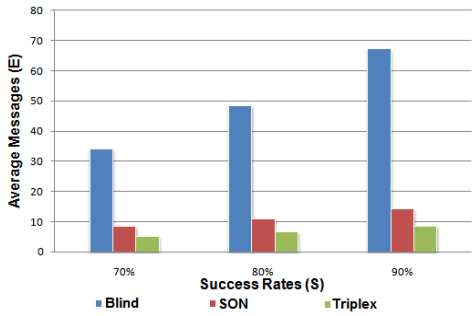


Figure 12. Average messages needed for achieving the success rates of 70%, 80%, and 90%

Figure 12 illustrates the average number of messages needed for finding out the first target service. It shows that our proposed solution can reduce the average messages, with a high success rate.

The above simulation results have demonstrated that the use of SON layer and dependency layer for routing the service discovery request are able to greatly increase the success rate and to reduce network traffic at the same time. From the simulation results, we also note that the classification of SONs should be appropriate. If too many types of SONs are in a network, on the one hand, the probability of locating the target SON will be reduced. On the other hand, once a request reaches the target SON, the probability of discovering a service within this SON will be higher than that in the network with few types of SONs. The reason for this is that more SONs in the network, the search space within each SON will be smaller. Thus we need to find a solution that balances these two parameters. The use of the dependency analysis for the SON selection process is one of the possible solutions. It can keep the high service discovery probability within a SON, meanwhile it makes efficient recommendations for the SONs selection. This aims at resolving the problems incurred by the large number of SONs. How to define such dependency rules is challenging in our proposed triplex layer based model. It is also one of our future research topics.

9. CONCLUSION

This paper has presented a model of P2P based distributed service exposure. In this model, the service exposure process is mainly divided into two main phases: local service exposure, which exposes local services to a gateway for local monitoring, and global service exposure, which enable the services to be discovered by external parties. Considering the factors of a great number of disperse gateways in a network, the single failure point for the centralized UDDI likes solution, and the possible bottleneck for the certain level number of services, we use a P2P based distributed method to enable the interoperation among the gateways. For further improving the efficiency and accuracy of service discovery, a triplex overlay based solution has been proposed. This triplex overlay architecture includes an unstructured P2P layer, a SON based overlay, and a service dependency overlay. The performance analysis and simulation results have shown that our proposed triplex overlay based solution can greatly improve the system performance by increasing the success rate of service discovery and reducing the number of forwarded messages needed for achieving certain level of success rate.

The proposed model improves a service exposure system in that it respects the diversity, autonomy, and interoperability of service exposure platforms. Each platform of service exposure can use its preferred techniques for exposing its services to users. By introducing a P2P based service information sharing system, the model enables the service information to be shared among different operators, service providers, and users regardless of their underlying heterogeneities. Telecom or Web service exposure platforms, Telecom or Web service discovery and publication platforms, or even some service creation environment can join this system as a peer, for example. And a user can discover a service provided by her/his service discovery platform provider through its specific technologies, or by other service discovery platform through the P2P based service information sharing system. The model also makes it possible to apply enhanced controls for service exposure in accordance with the requirements of different operators or service providers via gateways. The triplex-layer based P2P architecture can enhance the system scalability, and improve the service discovery and publication efficiency in the unstructured P2P based system. These improvements for the unstructured P2P based system also avoid the irrelevant nodes, which are responsible for other services, to be disturbed during a service discovery process. These nodes can thus reserve their resource for other relevant tasks.

Further work will focus on defining a widely-usable service description data model, and developing the more efficient strategies for service selection. The use of semantic, ontology, and social network information for providing service recommendations is also left for further study.

10. REFERENCES

[1] Semantic Annotations for WSDL and XML Schema, DOI=http://www.w3.org/TR/sawsdl/

[2] Web Service Semantics - WSDL-S, DOI=http://w3.org/2005/04/FSWS/Submissions/17/WSDL-S.htm

[3] OWL-S: Semantic Markup for Web Services, DOI=http://www.w3.org/Submission/OWL-S/

[4] Schlosser, M., Sintek, M., Decker, S., and Nejdl, W. 2002. HyperCuP—Hypercubes, Ontologies and Efficient Search on P2P Networks, In *Proceedings of 1st International Conference on Agents and Peer-to-Peer Computing (AP2PC '02)*, pp. 112-124, Bologna, Italy, July 2002

[5] Verma, K., Sivashanmugam, K., Sheth, A., Patil, A., Oundhakar, S., and Miller, J. 2005. METEOT-S WSDI: A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services, *Information Technology And Management*, Vol. 6, pp. 17-39, Jan. 2005

[6] Banaei, K.F., Chen, C.C., and Shahabi, C. 2004. Wspds: Web services peer-to-peer discovery service. In *Proceedings of 5th International Conference on Internet Computing (IC '04)*, pp. 733-743, Las Vegas, USA, Jun. 21-24, 2004.

[7] Modica, G., Tomarchio, O., and Vita, L. 2011. Resource and Service Discovery in SOAs: A P2P Oriented Semantic Approach, *International Journal of Applied Mathematics and Computer Science*, Vol. 21, pp. 285-294, Jun. 2011

[8] Li, R., Zhang, Z., Wang, Z., Song, W., and Lu, Z. 2005. WebPeer: A P2P-based System for Publishing and Discovering Web Services, In *Proceedings of IEEE International Conference on Services Computing (SCC '05)*, pp. 149-156, Orlando, USA, Jul. 11-15, 2005

[9] Papazoglou, M.P., Kramer, B.J., and Yang, J. 2003 Leveraging Web Services and Peer-to-Peer Network, in *Proceedings of 15th International Conference on Advanced Information Systems Engineering (CAiSE '03)*, pp. 485-501, Klagenfurt/Velden, Austria, Jun. 16-20, 2003

[10] Sahin, O.D., Gerede, C.E., Agrawal, D., Abbadi, A.E., Ibarra, O., and Su, J. 2005. SPiDeR: P2P-based Web Service Discovery", in *Proceedings of 3rd International Conference on Service Oriented Computing (ICSOC '05)*, pp. 157-169, Amsterdam, Holland, Dec. 12-15, 2005

[11] He, Q., Yan, J., Yang, Y., and Kowalczyk, R. 2008. Chord4S: A P2P-based Decentralised Service Discovery Approach, *In Proceedings of IEEE International Conference on Service Computing (SCC '08)*, pp. 221-228, Honolulu, USA, July 7-11, 2008

[12] Ni, Y., Si, H., Li, W., and Chen, Z. 2010. PDUS: P2P-base Distributed UDDI Service Discovery Approach, In *Proceedings of International Conference on Service Sciences (ICSS '10)*, pp. 3-8, Hangzhou China, May 13-14. 2010

[13] Zhang, Y., Liu, L., Li, D., Liu, F., and Lu, X. 2009. DHT-based Range Query Processing for Web Service Discovery", in *Proceedings of IEEE International Conference on Web Services (ICWS '09)*, pp. 477-484, Los Angeles, USA, July 2009

[14] Zhou, G., Yu, J., Chen, R., and Zhang, H. 2007. Scalable Web Service Discovery on P2P Overlay Network, in *Proceedings of IEEE International Conference on Services Computing (SCC '07)*, pp. 122-129, Salt Lake, USA, July 2007

[15] Crespo, A., and Garcia-Molina, H. 2005. Semantic Overlay Networks for P2P Systems, *Agents And Peer-to-Peer Computing*, Vol. 3601/2005, pp. 1-13, 2005

[16] Huang, C., Lee, G.M., and Crespi, N. 2012. A Semantic Enhanced Service Exposure Model for Converged Service Environment, *IEEE Communications Magazine*, pp. 32-40, vol. 50, Mar., 2012

[17] Beverly, R., and Afergan, M. 2007. Machine Learning for Efficient Neighbor Selection in Unstructured P2P Networks, in *Proceedings of Second Workshop on Tackling Computer Systems Problems with Machine Learning Techniques (SysML'07)*, Cambridge, MA, Apr. 10, 2007

[18] Liu, H., Abraham, A., and Badr, Y. 2010. Neighbor Selection in Peer-to-Peer Overlay Network: A Swarm Intelligence Approach, in *Pervasive Computing, Computer Communications and Networks*, pp 405-431, 2010