An Open and Extensible Service Discovery for Ubiquitous Communication Systems

Nor Shahniza Kamal Bashah, Ivar Jørstad, Do Thanh

An open and extensible Service Discovery for Ubiquitous Communication Systems

Nor Shahniza Kamal Bashah [1], Ivar Jørstad[2] and Do van Thanh[3]

[1] Norwegian University of Science and Technology, Department of Telematics, O.S. Bragstads Plass 2E, NO-7491 Trondheim, Norway - nor@item.ntnu.no
[2] Ubisafe – Gamleveien 252, 2624 Lillehammer – Norway – ivar@ubisafe.no
[3] Telenor & NTNU – Snaroyveien 30 – 1331 Fornebu – Norway – thanh-van.do@telenor.com

Abstract: In future ubiquitous communication systems, a service can be anything and introduced by anybody. Consequently, same or equivalent services may have different names and services with same name or type may be completely different. Existing service discovery systems are incapable of handling these situations. We propose a service discovery, which is able to discover all these new service types. In addition, it is capable to find services that are not exact matches of the requested ones. More semantics are introduced through attributes like EquivalenceClass, ParentType and Keywords.

Keywords: Service discovery, service lookup, service advertisement, service request, service marching, service orientated architecture

1 Introduction

Current mobile devices are connected to one network system at the time. The services offered by network system, e.g. telephony, short message service, voice mail, etc. offered by the mobile network system, are well specified and perfectly known by the mobile devices. It is hence not necessary for discovering services and a service discovery is completely unused. In a future ubiquitous communication environment people can use communication resources i.e. connections, services and devices, anywhere and anytime. The services available at one time will vary according to the network systems connected through the available connections. To let the user benefit all available services at any time, a sound and efficient service discovery is required. The future service discovery must be capable of finding relevant services offered by heterogeneous network systems. In such a ubiquitous communication environment, similar services can have different names in different languages. Furthermore, services with same name may not offer the same functions and capabilities. A future service discovery must be capable of dealing with the described challenges without confusion and returning correct answers in acceptable amount of time. In addition interoperability with existing service discovery systems must be ensured. The goal of this paper is to present a service discovery system for future ubiquitous communication system, which meets the mentioned requirements. To avoid confusion

the paper starts with a clarification of the notion of service and concepts around service. Next, the requirements on future service discovery systems are derived. The main section of the paper explains thoroughly how services are discovered. Suggestions for future works are presented in the conclusion.

2 The notion of service

What is a service?
Service is both a broadly used and confusing notion. Everyone depending on their context and standpoint will have a different definition of the service notion. For example in the Service Oriented Architecture (SOA), service is defined as a discoverable software resource with an externalized service description, which can be searched, bound, and invoked by a service consumer [1].

In this paper, the focus is on the user and a service is the work or tasks offered and performed for him/her.

More formally, the following definition is proposed:

A service is a mechanism enabling the end- user's access to one or more capabilities.
A network service is service offered to the user by a network system.

In our context, a service refers to a network service.

Furthermore, the user is making use of a device to access a service. Today, this device can be from a stationary personal computer, a laptop, a netbook, a game console, a PDA, a mobile phone, etc.

As shown in Figure 1 a network service is conceptually realized by two components a service client located on the user's device and a service server located on the network system, which are collaborating with each other.

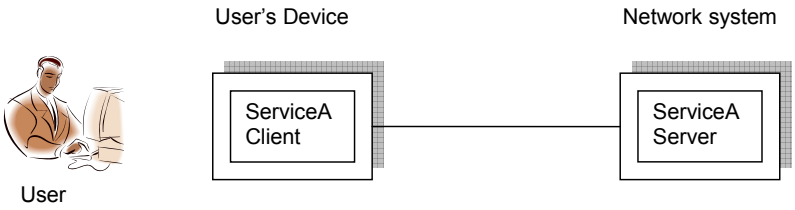

Figure 1 Network service architecture

The service client can be:

- **Generic client** such as the Web browser that is capable to collaborate with any service server.
- **Specific or dedicated client** such as that could collaborate only with some specific service servers.

There are also several possible collaboration schemes between the service client and the service server that produce different types of service.

Definition of service availability
Similarly to service the concept of service availability is not easy to define [1]. Availability in general covers a range of quality attributes ranging from reliability to maintainability and resistance. In a specific sense, availability refers to the up time of a service.

In this paper service availability is defined as the time when the service can be accessed.

Definition of service continuity
In this paper *service continuity is the ability for a user to maintain an ongoing service during mobility across domains, networks, and devices*.

Definition of service discovery
In this paper *service discovery is defined as the process of finding services that match the requirements of the service requestor*.

3 Requirements of service discovery for ubiquitous communication systems

1. A service in future ubiquitous communication systems can be anything:
    - This leads to a large variety of names, which makes the limitations in number of digits or in types of characters quite difficult or almost impossible.
    - The consequence is the situation where the same name or word can have several meanings and denote different services. Ambiguity and confusion are hence introduced.
    - The service discovery must be capable of handling different services with same names without confusion.

2. A service can be introduced by anybody at any time:
    - The result is that not all services can be standardized as in current service discovery systems where a service is well specified and has a uniquely defined name.
    - Another consequence is that the same service can be given different names by different service providers.
    - Since there is no regulation about a service definition a service may be close to another one but not 100% similar. There are two cases as follows:
        - A service b contains similar feature elements with service b but have also different elements as shown in Figure 2 1). The intersection of a and b is not empty and different from both a and b.

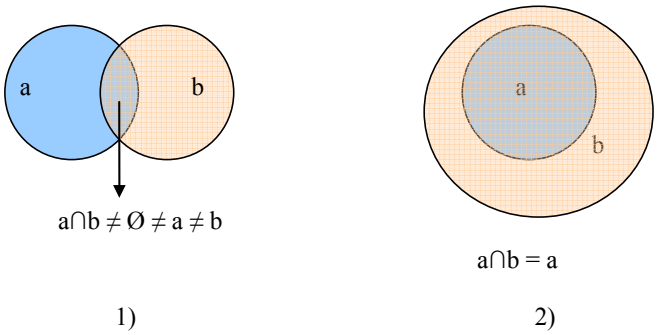

Figure 2 Relations between services

- A service A is a subset of service b since all the elements of a are also elements of b.

- A service name can be in other language than English e.g. Norwegian, Chinese, Urdu, etc.
- A service name may have several names in different languages.
- The service discovery must be capable of services with multiple names in multiple languages.
- The service discovery must allow the introduction of any service anytime by anybody.
- The service discovery must be capable of services with multiple names in multiple languages.

3. In a mobile environment, services must be discovered very fast:
   - This means that the service discovery must be very efficient
     - This calls for shorter service name

4. It is crucial not to misunderstand about a service or to confuse one service with another one:
   - The service discovery must be error-free
     - This calls for longer service name

5. All the services available in an area must be discovered:
   - The service discovery must be sound

6. It is also essential to be able to verify that a service is offering what it announces:
   - It must be possible to extend the service discovery with verification functions

7. It is also essential to be able to conclude that a service is trustful:

- It must be possible to extend the service discovery with security functions to validate a service

8. The user must be able to move everywhere in the world:
   - The service discovery must be capable of functioning ubiquitously

9. Since there are currently many services and service discovery system it is important to ensure interoperability:
   - The service discovery must be capable of discovering current existing services.
   - The service discovery must be capable of operating with existing service discovery systems.

More details about future service discovery requirements can be found in [9] Let us now continue with the design of the service discovery system.

4 Discovering a service

Service registration

In order to be discovered a service must advertise itself to potential clients. However, this requires that the service must be equipped with advertisement capabilities and be present everywhere there are clients. To relieve this burden of the service we propose the introduction of a service registry or repository or also called service broker, which is mediating the services to the consumers. In fact, the usage of a service registry is quite common to most current service discovery systems.

For the service registration the service provider must supply the details about the service that are necessary to detect, identify and use the service as follows:

Service name:

The first item is the name or identifier of the service that is necessary to denote and recognise the service. Until now in most systems such as UPnP, Jini, Salutation, etc. the service name is both well-defined in terms of format and convention. For example, with UPnP [2] a service name must be descriptive of function, less than 64 characters, having capitalized first letter of each word used in the name and reflecting the ServiceType and version e.g. ServiceType 0.8. This naming convention is quite reasonable but fails to satisfy requirement 2 which requires having names in different languages. We propose hence to extend the service name format of our service discovery to allow any language although this will make the name less descriptive.

Service type:

The Service type is necessary to carry on a search and identification of the functions offered by the service. In most current systems such as UPnP, Jini, Salutation, etc. the service type is usually standardised by the respective organisation in charge of the systems. For example in UPnP the service types are defined and published by the UPnP Forum.

The standardisation of the service type is conflicting with requirement 1 and requirement 2, which together state that a service can be anything introduced by anybody at anytime. To meet the requirement 1 and 2 our service discovery solution must allow that a new service type can be introduced by any player including the user without the approval of any responsible organisation.

In addition, a service type may have several names in different languages.

To introduce a new service type the service provider has the following options:

- **Brand-new service type:** without relation with any existing service type: A service type description in XML (eXtensible Markup Language) containing the service type Name, Keywords, ParentType, StateVariables, etc. as shown in Figure 3 has to be elaborated. In addition an URI (Uniform Resource Identifier) has to be assigned to this service type description. This service type description will be used later in the service matching.
  Since the service type is a brand-new type that was not derived by any existing service type ParentType and ParentType has to be set to nil. The service type does not have any alias since there exists no equivalent service type. The EquivalentClass is then left empty.
- **Equivalent service type**: The service type has a different name and may be another implementation but is equivalent to an existing service type. The service provider has to add the name of the existing service type and also all the known service type in different languages.
- **Subtype of an existing service type**: The service type has all the functions and features of an existing service type but has also additional ones. The parent type is hence indicated.

Service type description template

A service type description has the following items:

- *Name:* The name can be in any language and less than 64 characters.
- *Keywords:* Some words that can be used in the first round discovery
- *ParentType:* The name of service type that the current type is derived from.
- *ParentTypeURI:* The URI of the parent type
- *EquivalenceClass:* All alias in any language are given here
- S*tateVariables.* The *state variables* determine the states of the services. They are left empty in the service type description
- *Actions*: Actions are the methods that can be called by clients or other services
  - Each action has a *name* and a set of *parameters*
    - Each parameter has a *type*, *allowable values* (for enumerated types), and *direction* (in or out)
- *Events:* Enable clients to subscribe to the occurrence of a particular event

Figure 3 A Service type description template

5 Advertising and requesting services

Service advertisement:

Now that a registry has registered all the available services are registered for a location it has somehow to inform all the potential service consumers or clients about them.

The service advertisement is the process of indicating to all potential users/consumers represented by a software client that a set of services is active and ready for use.

The service advertisement enables the users to discover not only the services that are familiar and used by the users but also new services that are so far unknown to the users.

Basically, there are two alternatives of implementing the service advertisement.

Alternative 1: The service registry or broker can take the initiative and broadcast the list of available services periodically. All the users should listen to a certain channel or address to get the list. This alternative has both advantages and disadvantages [7] as follows:

Advantages:

- Reaching all the potential users simultaneously and hence avoiding repetitions for each user. This can save bandwidth, resources and time.
- This alternative is suitable when the number of users is high.

Disadvantages:

- Users starting to listen in the middle of the broadcast will have to wait for the next broadcast.
- This alternative is not suitable if the list of available services is long because a client coming in the middle of a broadcast may have to wait long to get the list of available services and the service discovery may take unacceptable long time.

Alternative 2: A client wanting to find services issues a request for a list of available services to the service registry. In a variant of this alternative the list of available services is published at a predefined location and any user can fetch it when needed as in the case of UDDI [5] for Web services [6]. This alternative has both advantages and disadvantages [8] as follows:

Advantages:

- Any user can request the list of available services whenever needed.
- The variant with the list of available services published at a predefined location can be quite convenient

Disadvantages:

- This method consumes more bandwidth and may not be appropriate in wireless environments
- When the number of users is high, it may take time until the service registry manages to response to each user

The choice of alternatives is subject to the following factors:

- *Bandwidth of the connection:* Narrow bandwidth calls for the broadcast alternative
- *Number of available services:* Longer list of available services may render the waiting time unacceptable for the broadcast alternative.
- *Capabilities of the clients:* Low capability clients may be in favour of the broadcast alternative.
- *Number of clients:* Large number of clients may also favour the broadcast alternative
- *Diversity of the clients:* To support a very diversified range of clients ranging from low capability to high capability, the broadcast alternative seems to be the most appropriate.
- *Mobility of the clients:* For highly mobile clients that come in and leave a location very rapidly, the broadcast of long list of available could be inadequate because the waiting time could be too long and the mobile client has already left the location.

The selection of appropriate alternatives has to be considered properly and empirical methods have to be used.

One important issue to take into account is the size of the list of available services. If the size is too large it could be both resource and time consuming to deliver the list to the users. On the other hand, a shorter list may not contain sufficient details allowing a proper service matching and the client is forced to request additional information.

Anterior service advertisement may be a solution to be considered in future mobile environments where predictions can be made about the movement of the mobile users. Service advertisement can be made before the mobile user enters the location. Such an anterior service advertisement can contribute to speeding up the service discovery and hence ensuring service continuity.

Service request:

Instead of asking for the list of available services and carrying the service matching itself the client can request for a particular service or set of services. In the service discovery terminology, it is also called service lookup or service search. This process relieves the client from the processing burden since the *search and find* services is carried out by the network system.

The major drawback of the service request process lies on the fact that it removed the opportunity to discover new and unknown services to the mobile users. Indeed, when visiting a foreign a mobile user may be very interested to discovered local and exciting services.

We believe that an efficient service discovery for ubiquitous communication should support both the service advertisement and service request

6 Service matching

Service matching: is the process of comparing the service request against the available service advertisements and determining which services best satisfying the request [**Error! Reference source not found.**]. The service matching can be performed by the mobile client after the acquisition of the list of available services or by the network system upon receipt of a service request. Therefore, it constitutes the fundament of the service discovery.

It is worth noting that quite often the word "service" is confusingly used to designate both a service instance and a service type. In this paper, service is used to denote only service instance and not service type.

In existing service discovery such as Jini [4] the service matching is employing an exact match operation on service type and attributes. The exact match operation is successful and returns the services, which service type and attributes are the same as the ones specified in the service request.

For example in Jini, to request a particular service an instance of the `ServiceTemplate` class is to used to match and filter the set of existing services. The template specifies both the type of the required service, which is the first filter on possible services, and a set of attributes which is used to reduce the number of matching services if there are several of the right type.

The exact match operation on the service type and attributes does not fulfil the requirement 2, which imposes that a service can have different names and a name can denote different services.

Equivalent services

To solve this problem, we propose to extend the match operation on the *equivalence class* of the requested service type.

In accordance with the definition of an equivalence class [10] the equivalence class of a service type "a" contains all the services s that are equivalent to the service a. More formally, the equivalent class [a] is defined as follows:

$[a] = \{s \in S: s \sim a\}$

The binary relation ~ is said to be an equivalence relation if and only if it is reflexive, symmetric and transitive:

For all a, b and c in S:

- $a \sim a$. (Reflexivity)
- if $a \sim b$ then $b \sim a$. (Symmetry)
- if $a \sim b$ and $b \sim c$ then $a \sim c$. (Transitivity)

More details about service equivalence can be found in [11]. However, instead of prescribing a precise definition of the equivalent relation we propose to reserve to the service providers the responsibility of deciding whether their services are equivalent to other services and which ones. At introduction of a new service a service provider can specify which are the alternatives or competitors to this service.

For example, a Restaurant service helping the users to find suitable restaurant is equivalent to a Dining service or a Ristorante, Bistro, Restoran, etc. service depending

on the current country. The equivalence class of Restaurant will contains the elements as follows:

[Restaurant] = {Ristorante, Cafe, Bistro, Restoran, ...}

Service subtyping

Quite often there is no equivalent service available but there exist services, which have in addition to the requested functions other ones. As indicated earlier they are subtypes of the requested type and are fully eligible as substitute. To support this function we propose to extend the match operation on the parent type and an attribute ParentType is introduced in the Service type description template.

For example, both the `FastFood` service type and the `ItalianRestaurant` service type have as `ParentType` Restaurant service type. If the `Restautant` service type is specified in the service request then both the `FastFood` and `ItalianRestaurant` services should be returned as results of the matching process.

Keywords

There are cases where available equivalent services but are not discovered because the service type, the equivalence class and the parent type are simply like NULL. To avoid these cases we propose to introduce an attribute called Keywords which specifies the context, circumstances or areas that the service belongs to.

For example, Restaurant service type may have a Keywords: Meal, Dining, Supper, etc.

7 Conclusions

In this paper, the necessity of an innovative service discovery capable of handling similar services with different name and different services with same name is explained thoroughly. The feasibility of such a service discovery is also demonstrated through the design of the overall structure of a service discovery. A prototype is currently under development and will be accomplished in the near future. A natural further step will be very interesting to carry out experiments with various number of clients, number of services, different bandwidths and complicated service ontology. The collected results will then be used to optimise the service discovery prototype. Another relevant and exciting future work will be the contribution to the extension of the 802.21 MIHF (Media Independent Handover Function), which is aiming at facilitating handover between heterogeneous networks [12] for devices equipped with multiple wireless access technologies. Such an extension will ensure service continuity across multiple heterogeneous access networks.

8 References

1. IBM Software Group: Service-oriented modeling and architecture - How to identify, specify, and realize services for your SOA - Ali Arsanjani, , SOA and Web services

Center of Excellence, IBM, Software Group - http://www.ibm.com/developerworks/webservices/library/ws-soa-design1/
2. Paul Allen, Sam Higgins, Paul McRae, and Hermann Schlamann: Service Orientation – Winning Strategies and Best Practices - ISBN-13: 9780521843362 – Cambridge University Press, 2006
3. UPnP Forum: UPnP Device Architecture 1.1 - Document Revision Date 15 October 2008
4. Sun Microsystems: JiniTM Architecture Specification Version 1.2 - [Internet] December 2001 - http://www-csag.ucsd.edu/teaching/cse291s03/Readings/jini1_2.pdf
5. UDDI org. Universal Description, Discovery and Integration: UDDI Technical White Paper – September 6 2000
6. W3C Working Group: Web Services Architecture - [Internet] 11 February 2004 - http://www.w3.org/TR/ws-arch/
7. C. Dipanjan, J. Anupam, Y. Yelena: Toward Distributed Service Discovery in Pervasive Computing Environments - IEEE Journal of Transactions on Mobile Computing - IEEE Computer Society, Vol. 5, Nos. 2, February 1 2006, pp. 97-112
8. Vijay Kumar: Mobile Database Systems – ISBN-13: 9780471467922 – John Wiley & Sons Inc. Publication, 2006
9. Kamal Bashah, N.S, Jørstad, I. & van Do, T: Service Discovery in Future Open Mobile Environments - Proc. the Fourth International Conference on Digital Society (ICDS2010) - February 10-16 2010. St. Maarten, Netherlands Antilles
10. J. Daintith: A Dictionary of Computing, 6th Edition – 2008 - ISBN 978 0 10 923401 1 – Oxford University Press
11. Jørstad, I. & Van Do, T.: Personalisation and Continuity of Mobile Services. Chapter in "Mobile Multimedia: Communication Engineering Perspective", Ibrahim, I.K. (ed.) - pp. 59-84, Nova Science Publishers, Inc. ISBN: 1-60021-207-7, 2006
12. Media Independent Handover Services, "Draft IEEE Standard for Local and Metropolitan Area Networks: Media Independent Handover Services", IEEE P802.21/D7.1, August, 2007