ABSTRACT
The clients, in ubiquitous environment, want to be consistently offered services at any time and in any places. There are so many service discovery middleware such as Jini, UPnP and HAVi. Each one of them supports different communication protocol. Therefore, when the clients ask an inquiry of service search, they have to support communication protocol of all service discovery middleware. Namely, all the cost shall be borne by the customer. In addition, these middleware does not consider the service location and status. To solve these problems, we suggest the SOA-based framework is Web Services on Universal Networks. The framework supports a dynamic service discovery through the interoperability between service discovery middleware. Furthermore, it considers service location and status at service discovery process.

Categories and Subject Descriptors
A.0 [General]: Conference proceedings
C.2.4 [Computer-Communication Networks]: Distributed Systems – Client/Server, Distributed applications, Network operating systems
J.7 [Computer in Other Systems]: Publishing, Real time
K.6.3 [Management of Computing and Information Systems]: Software Management – Software development, Software maintenance, Software process

General Terms
Management, Design, Reliability, Human Factors and Standardization

Keywords
Ubiquitous, Dynamic Services Discovery and Web Services

1. INTRODUCTION
The common service discovery middleware, which is used in distributed network environment, is Jini[1], UPnP[2] and HAVi[3]. Each one of them supports a different communication protocol. For example, a device of Jini can communicate by using RMI, and one of UPnP supports the SOAP. Therefore, when the clients ask an inquiry of service search, they support both of them because they want to be offered all usable services. It may bring expense to the client. In ubiquitous environment, all units, included client and device, can move to anywhere and at anytime. The current systems do not support ubiquitous feature like that dynamic change of service status and movement. In Addition, they can’t support another communication protocol.

To solve the current problems, we suggest a Web Services on Universal Networks (WSUN). WSUN is SOA based framework. The framework supports a dynamic discovery protocol is Universal Service Discovery Protocol. Furthermore, It supports an interoperability between service discovery middleware.

In this paper, we shortly introduce WSUN. The detail description of WSUN is described in [4]. And we prove our approach through implementation the prototype system.

2. Design of Web Services on Universal Networks
Web Services on Universal Networks (WSUN) is based on Services Oriented Architecture (SOA) framework. The framework supports the ubiquitous web services discovery and binding. This framework has a Universal Service Broker (US Broker) (Figure 1). It is core component that supports dynamic service discovery and interoperability between service discovery middleware.

Since any Web Services client can find the services which are located in any service discovery middleware, we transfer all services in service discovery middleware to Web Services. A description of Web Services is registered in a registry. When a client searches services, Query Agent receives and parses the client message and it searches the services through US Registry.
3. Design of Web Services on Universal Networks

3.1 Design of Listener

Listener takes charge of communication between US Broker and client or WSUN device. The client has to support SOAP protocol. WSUN device supports our protocol is Universal Service Discovery Protocol.

Listener is composed of three components (Figure 2). The client and WSUN device send the multicast message to find the US Broker. Telecommunication of Listener gets this message and returns the response message. We register all service information in US Registry. We register the WSUN device information through Register. For this purpose, Register uses the Publish Agent operation. We check whether service is registered or not.

3.2 Design of US Registry

US Registry is composed of two registries: Device/Service Registry and Context Registry.

Device/Service Registry has device and service information such as their name, description, ID and category information. Service
location is stored in Context Registry. The information is used when the client searches the service. For supporting interoperability between sub-networks and US Broker, we register service ID in our framework and in sub-networks. We can map services in sub-networks to services in our framework.

### 3.3 Design of Query Agent

Query Agent takes charge of service discovery in US Registry and client message processing. For this purpose, it is consists of Parser, Matching, CheckService and GetServiceDeviceInfo(Figure 4).

Parser supports the client message parting and query planning. Before we register the service information on US Registry, we check whether the service is registered or not. For this operation, we design the CheckService component. We can get the device and service information from GetServiceDeviceInfo. The information includes service and device name, description, ID, location and category information. We find the services corresponding to client query using the result of GetServiceDeviceInfo.

Figure 5 illustrates the sequence diagram of Query Agent. Query Agent receives service discovery message from client. This message is parsed through Parser. After parsing, parsed information is sends to Matching component. We can find the service using this information. Also, the Matching searches the service location on context registry. The location of services must be near to clients’ location. The last step, the Matching joins the result from Device/Service registry and Context Registry. And then, the matching results are sent to client.
Figure 6. Insert information on registry

Figure 7. Update information on registry
3.4 Design of Publish Agent
Publish Agent registers device and service information on Device/Service Registry and Context Registry. Therefore, it is composed of four components: UUID Generator, Parser, CheckService and Register. UUID Generator generates unique UUID of service and devices which are used in the US Broker. The Parser takes charge of message from UniA or Listener. After parsing the message, we can register the service and device information on US Registry.

Publish Agent has two processes. One is insert process and another is update process.

The following is explanation of insert process (Figure 6).

① Parser receives the request message included publish service and device information.
② Parsed message is sent to Register.
③ Before publish the information service and device, Register checks whether service is registered or not.
④ If service is not registered and Register publishes the service and device information to Device/Service Registry, and the location of service to Context Registry.
⑤ If service is registered and Register update service status from invalid to valid.

The following is explanation of update process (Figure 7). This process is same process in Figure 6 except registered service and device information on US Registry.

4. Implementation of Web Services on Universal Networks
Our implementation environment is following.

- OS: Window Server 2003 R2 Enterprise Edition R2
- Execution Environment
- J2SDK 1.4.2 version
- MySQL 4.1 version
- Tomcat 5.0.30 version
- Apache 2.0.58 version
- Axis 1.3 version

We consider the scenario to prove our approach. The client enters the meeting room. He wants the three services. One is black and white printing service for materials. The other is color printing service for picture. Another is vim projector for presentation.

First, client search the black and white printing service. He can use the service name, description and category information to discovery services. And the client can receive the result included device name, service name, location and WSDL URL(Fig 8).

We also consider the black and white printing service and vim projector service are located in DPWS network, and color printing service is located in Jini network. In this environment, we can confirm that the client uses all services in any networks.

5. Conclusion
This study explains the WSUN and proves our approach. The WSUN supports interoperability between service discovery middleware, and helps clients to use ubiquitous web service without additional work by generating virtual web service for the clients’ convenience. The current WSUN is designed for the integration of Jini, UPnP, and DPWS but through further study, other sub-network environments will be included.

The further study will verify this with several scenarios by realizing the WSUN. Also, after clients’ inquiry will be maximized and their context information will be considered, more improved ubiquitous web service discovery protocol will be defined.

6. ACKNOWLEDGMENTS
This research was supported by the Ministry of Information and Communication, Korea, under the College Information Technology Research Center Support Program, grant number IITA-2006-C1090-0603-0031.

7. REFERENCES