Service Discovery Protocols in Pervasive Computing: A review

[1]Neelma Bhatti, [1]Lachhman Das Dhomeja, [2]Yasir Arfat Malkani

[1]Institute of Information and Communication Technology, University of Sindh, Pakistan

[1]neelmabhatti@gmail.com, [1]lachhman@usindh.edu.pk

[2]Institute of Mathematics and Computer Science, University of Sindh, Pakistan

[2]yasir.malkani@usindh.edu.pk

Abstract: Pervasive Computing Environments are composed of heterogeneous services. Pervasive computing applications need to discover the service(s) based on context (e.g. user preferences, user location, user activity, etc) and interact with them to perform tasks that support us in everyday life with minimal or no user distraction. This makes discovery protocols an essential requirement of Pervasive Computing. In this paper we review state-of-the-art on discovery protocols / systems and investigate their suitability in Pervasive Computing. This paper then presents some open research challenges that need to be addressed by discovery systems to meet the requirement of Pervasive Computing Environments.

1 INTRODUCTION

With the advent of Pervasive Computing [1], there has been a profusion of services to assist and support user tasks and requirements. One of the goals of Pervasive Computing is to fulfill user needs and to enrich user experience by providing the user with the service based on his/her context (e.g. user location, user preferences, etc) with minimal or no user distraction. Service discovery is an essential requirement to achieve this goal. Service discovery protocols enable devices to discover service(s), bind to them and communicate with each other. Service discovery involves three components: (1) registration of services, (2) discovery of services and (3) interaction with the discovered services. Consequently, the major goal of discovery protocols is to provide mechanisms and supporting infrastructure to support these three main components of service discovery.

There is an immense literature on service discovery. Although early service discovery protocols research focused on traditional distributed and enterprise systems, the research on service discovery now has changed its direction and is being carried out in context of Pervasive Computing. Various survey papers on service discovery [2 - 8] have been written that provide a review of various existing service discovery protocols and present open challenges that need to be addressed. However, some of the issues of discovery protocols in pervasive computing (such as context-awareness, service management, security, etc) have received considerably less attention.

Pervasive computing environments are dynamic and dynamicity arises out of many factors including mobility (both device mobility and service code mobility), frequent disconnections of a service provider or a client due to weak wireless communication link, variable network bandwidth and heterogeneity of pervasive devices in terms of device characteristics (e.g. low battery power and small screen size) and protocols, etc. All these factors badly affect service provisioning in pervasive computing.

The remaining part of this paper is organized as follows: section 2 presents the state-of-the art of service discovery, in section 3 we provide the comparison framework, which describes the requirements/challenges of the discovery protocols for pervasive computing environments, section 4 presents the comparative analysis of the reviewed protocols and section 5 concludes the paper.

2 PREVALENT SERVICE DISCOVERY PROTOCOLS

There is a large body of work on service discovery protocols. The most widely used among them include Jini [9], UPnP [10], SLP [11], Bluetooth [12] and Salutation [13]. Each of them tends to devote itself to a different administrative domain and user requirements. In this section, a detailed review of existing discovery protocols is presented.

2.1 JINI

Jini [9] is a java-based service registration and discovery technology developed by Sun Microsystems. Jini provides service/device registration, discovery and communication mechanisms for ad hoc networks. The discovery mechanism in Jini is similar to that of SLP. Unlike SLP, it uses mandatory directory service to register the services and relay them to java-enabled clients when requested. It consists of three protocols named lookup, discovery and join illustrated in Figure 2. On bootstrapping, services look for a lookup service and register themselves with it. This process is known as the Discovery and Join process. During the registration process Jini services upload their service-object along with service attributes in a Lookup Table of the lookup service. When a client needs a service, it locates a lookup service to search the required services and download the service-object. Once the client downloads the service-object from the lookup service, it directly contacts the service for further communication. Services are leased which eradicates the need to explicitly identify services which fail or become unavailable. It also provides Event Notifications subscription to clients, which notifies them of any change in the service attributes that are of interest.

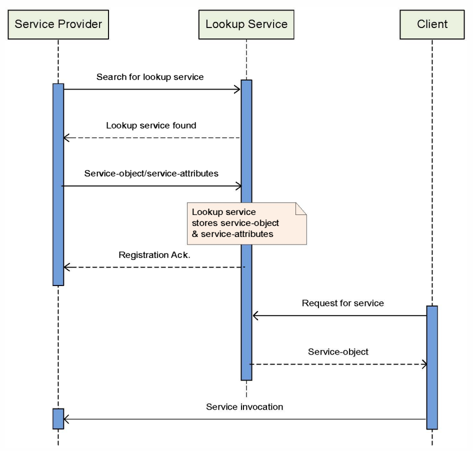

Fig. 2. *Message sequence diagram illustrating Jini's discovery□join□ loo□up process [22]*

Authorized licensed use limited to: to IEEExplore provided by University Libraries | Virginia Tech. Downloaded on July 24,2020 at 13:42:13 UTC from IEEE Xplore. Restrictions apply.

2.2 UNIVERSAL PLUG AND PLAY (UPNP)

Microsoft Corporation played an important role in UPnP's development and it is considered as an extension to the Microsoft's Plug and Play technology; however it is more than just an extension. The major objective of UPnP [10] is to enable discovery, auto-configuration through AutoIP [16], management and control of devices in unmanaged and small computing environments, such as small office or home environments. UPnP uses standard protocols like HTTP, XML, and SOAP for discovery, description, and control of devices. Since it is targeted for small home or office networks, UPnP uses a non-directory based architecture unlike Jini and SLP. Further, in contrast to SLP and Jini, UPnP uses XML for all the communication and exchange of device's information among the two entities of UPnP network (i.e. Control Point and Device). As in Jini's discovery-join-lookup process, SSDP is used for both advertising the device's (service) presence to the other devices in the proximity/scope as well as discovering other peer devices. Devices' profiles describing their capabilities and features are written in XML format.

2.3 SERVICE LOCATION PROTOCOL (SLP)

SLP [11, 14] is developed by IETF (Internet Engineering Task Force) for service registration and discovery within a particular location or scope and aims to be a vendor-independent standard. By using convergence multicast in small and directory agents in comparatively large networks [15], SLP is highly scalable. Its architecture consists of UA (User Agent), SA (Service Agent) and an optional component DA (Directory Agent) which acts like a caching node to categorize and group services in larger networks. The SA sends the service URL and other service attributes to the directory agent at the time of registration. In order to prevent the DA from storing stale service information, SA needs to refresh its registration to ensure its availability. Figure 1 illustrates the interaction mechanism between the three components of SLP in a small or Local Area Network.

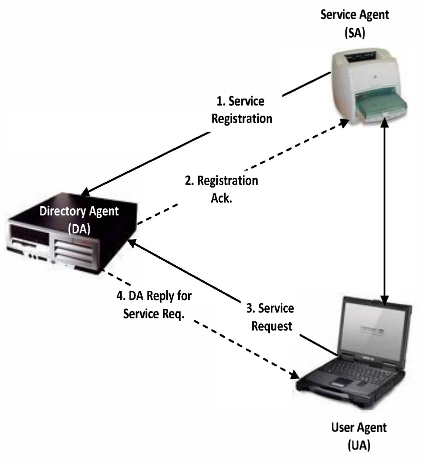

Fig. 1. *An illustration of ser□ice registration and disco□ery mechanism in S□P [22]*

2.4 SALUTATION

Salutation [13] is yet another service discovery protocol developed by salutation consortium. Salutation addresses the problem of heterogeneity in ubiquitous environments by providing an operating system, communication protocol, and physical platform independent architecture. Constituents of Salutation architecture are Salutation Manager (SLM) and Transport Manager (TM). SLM stores the locally cached services and may communicate with other SLMs to discover and access the services using Remote Procedure Call (RPC). It can use more than one TM to communicate over different network technologies. TM acts as a bridge between SLM and the communication technology, giving it network transport independence.

2.5 BONJOUR

Formerly named as Rendezvous, Bonjour [17] is an apple propriety service discovery protocol, submitted to IETF as a part of standard-creation process. It is based on IETF's Zeroconf (zero configurations) technology, which enables the seamless discovery and interaction of devices and services without prior configuration. It is an unstructured and decentralized discovery protocol which offers service discovery in local and ad-hoc networks. It supports the advertising and discovery of services using link local addressing and multicast DNS (mDNS). A service provider randomly chooses and broadcasts IP address and selects it if the response renders it unclaimed. Service publication is performed by multicasting the service advertisement to all the network devices. Bonjour carries out the service discovery process through DNS Service Discovery (DNS-SD) by browsing available services.

2.6 PROOF-OF-PROXIMITY (PoP) FRAMEWORK FOR DEVICE PAIRING

Malkani [18] advocated that security has never been a major concern or major design goal of previously designed discovery protocols and thus proposed a generic framework for secure pairing of devices for home or small building based pervasive computing environments by exploiting the common capabilities of communicating partners. This framework integrates the discovery mechanism and number of different pairing schemes, which comprehensively provides the support for a wide range of device pairing scenarios in pervasive computing environments. This framework consists of two parts: (1) device registration and discovery, and (2) Authentication. It provides confidentiality and integrity protected device discovery and registration mechanism by combining several features of the existing well-known discovery mechanisms, such as SLP and UPnP. However, further work is required towards standardization of the proposed discovery mechanism.

2.7 PCRA

PCRA [19] is a policy-based context-aware reconfiguration and adaptation system that allows the development of adaptive context-aware applications from policy specifications. Two of the core features of this system are (1) context-aware discovery and (2) context-aware adaptation. Context-aware discovery allows discovering service(s) based on context and some other search criteria and then creating bindings between the user of the environment and the discovered services, while context-aware adaptation allows modifying (adapting) the behavior of the bound service in response to context. PCRA does not have context monitoring and processing capability, instead simulated context widgets have been used to provide context to the discovery system. PCRA's binding model addresses the issue of invalidity of bindings and its detailed description can be found in [20]. Another feature supported by the binding model is caching of bindings for improved performance and the detailed

Authorized licensed use limited to: to IEEExplore provided by University Libraries | Virginia Tech. Downloaded on July 24,2020 at 13:42:13 UTC from IEEE Xplore. Restrictions apply.

description of this is available at [21]. PCRA lacks the support for dynamic service characteristics.

.KONARK

Konark provides a framework for connecting isolated services offered by proximal pervasive devices, over a wireless medium [22]. It has two major aspects: service discovery and service delivery. Konark uses a decentralized, peer-to-peer mechanism by enabling devices to act as a server and a client simultaneously.

Konark uses XML-based service description and a micro-HTTP server present on each device handles service delivery, which is based on SOAP. Konark assumes an IP level connectivity in the ad-hoc networks, considering most of the modern devices run operating systems providing zero configuration techniques. [16][23]. It presents a service registry based on a tree-structure, with service description being generic on top and becoming more specific while moving down the tree levels. [22].

2.8 SCOOBY

Scooby [24] is a system that provides Scooby language and supporting middleware to support service composition in pervasive computing environments. The core part of the underlying Scooby middleware is a service discovery/binding model, which enables devices to discover the services based on some search criteria and interact with them. Scooby language includes high-level binding constructs which are used to develop composed services. The composed service is compiled by the Scooby compiler, which produces the corresponding Java source code, and then this Java source code is compiled to produce Java byte code. When a call is made on the bound service, Scooby discovery model first ensures its availability by initiating contact with it. If available, the call is made and if not, it starts discovering another service. Moreover, the discovery system periodically checks to make sure that the bound service is still reachable. If, for some reason, it is no longer available, it reinitiates the service discovery process. To summarize, the bound service availability is periodically checked and also prior to a remote method call.

Another important feature of Scooby is the support for dynamic service characteristics, which allow changing service characteristics dynamically. This feature is important because static service characteristics cannot be changed after the service has been deployed.

3 COMPARISON FRAMEWORK FOR SERVICE DISCOVERY PROTOCOLS IN PERVASIVE COMPUTING

Highly dynamic nature of pervasive computing environments give rise to various research issues that need be addressed in context of service discovery systems. Some of the researches issues have already been addressed and while others haven't received significant amount of attention.

In this section we present a comparison framework for service discovery protocols used in pervasive computing environments. The framework is shown in figure 3. This framework highlights the issues that we think have not received much attention from the research community and must be addressed by discovery protocols for their use in pervasive computing environments. The issues listed in the framework are described in subsequent sub-sections.

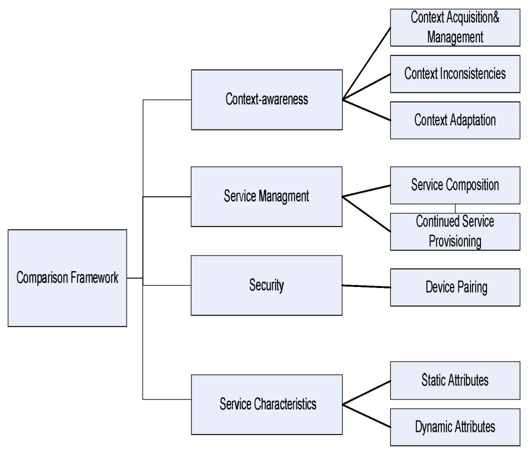

Fig. 3. *Comparison framewor for disco ery protocols in per asi e computing en ironment*

3.1 CONTEXT-AWARENESS

Context-awareness is a fundamental requirement for realizing the vision of pervasive computing, which was put forward by Mark Weiser in his seminal paper [1]. There exist a number of definitions of context and consequently context-awareness in literature [25-27], but the most widely quoted definition of context and context-awareness are given by Dey et. al [28], which defines context as:

"any information that can be used to characterize the situation of an entity. An entity is a person place or ob ect that is considered rele ant to the interaction between a user and an application including the user and applications themselves."[28]

and context-awareness as:

"a property of a system that uses context to provide relevant information and/or ser ice to the user where rele ancy depends on the user's task".[28]

Various researchers [25-27] have categorized context-awareness, and two of the categories identified and discussed by these researchers are contextual resource discovery (also called contextual reconfiguration) and contextual adaptation. These are adaptive features of context-awareness in which application adapt their behavior in response to context. These two different forms of adaptation are central to realizing Mark Weiser's vision of pervasive computing. The protocols discussed thus far have not provided the support for contextual adaptation.

3.1.1 CONTEXT ACQUISITION & MANAGEMENT AND CONTEXT INCONSISTENCIES

One of the challenges associated with Context-aware systems is that of the process in which contextual information is received from environment and acted upon. Context acquisition is the first step towards context-aware service discovery. Contextual data is obtained from either sensing devices or by inferring the user intent through prediction or user profile and service usage history. Therefore, the support for context acquisition and processing is a fundamental requirement for discovery protocols

Authorized licensed use limited to: to IEEExplore provided by University Libraries | Virginia Tech. Downloaded on July 24,2020 at 13:42:13 UTC from IEEE Xplore. Restrictions apply.

in pervasive computing. However, Context can become noisy and incomplete and hence inconsistent due to some reasons, such as malfunctioning of sensing technology or incompetent sensing technology (for instance, RFID readers just detect about 60% - 70% of the tags in their surroundings [29] and remaining ones go undetected. Context-aware discovery involves discovering a service based on context, and an inconsistent context may lead to discovery of unwanted services or system may fail to find any service even when the required one is available. The context inconsistency needs to be addressed by discovery protocols for smooth operation of applications.

3.1.2 CONTEXTUAL ADAPTATION

Context-aware adaptation is a process of modifying service behavior in response to context. In order to cope with dynamicity of pervasive computing environments (for example. variable network bandwidth and heterogeneity of pervasive devices in terms of device characteristics (e.g. low battery power and small screen size) and to satisfy user needs with minimal or no user distraction, service discovery protocols must be able to provide adaptation support. For example, in response to variation in bandwidth, the discovered service offering video content may be adapted to provide textual version of the content.
in a simple home lighting scenario the light service may be modified to adjust the light value to some user preferred value based on the user‘s activity (sleeping, reading or watching TV); or in some other context-aware scenario the TV volume may be reduced when someone is talking on the phone. Many context-aware service discovery protocol architectures have been proposed and they exploit the use of context for discovery of the best possible service but don't provide the support for adaptation to context.

3.2 SERVICE MANAGEMENT

Many factors including mobility, node failures and weak wireless communication badly affects service provisioning. Mobility can be classified as (1) device mobility and (2) service code mobility. Device mobility causes service unavailability, while the service code mobility (the service moved to some other node over the network) makes a binding between the client and the moved service invalid. Weak wireless connectivity may disconnect the service provider or client. Moreover, Individual services available over the network may not be sufficient to satisfy the user goals and they may need to be combined to form a composed service to meet the user's need (service composition).

We consider service composition as a part of service management. Therefore, service management comprises two main operations: service composition and continued service provisioning. Discovery protocols must provide mechanisms and corresponding infrastructure for service management to support service composition and to ensure continued service provisioning in face of dynamism of pervasive computing environments.

3.2.1 SERVICE COMPOSITION

Service composition is the process of discovering atomic services and combining them to form a high level service to meet the user's need. At times, in order to perform the user's task smoothly, the discovery protocol needs to identify and integrate appropriate services to compose a customized set of services which best aid the requirements of a user. Service management support needs to provide the appropriate mechanism for service composition in order to constitute the service desired by the client.

3.2.2 CONTINUED SERVICE PROVISIONING

One of the most significant goals of pervasive computing is to enable a user to perform her tasks unobtrusively. It is only possible if the service provision is done in a seamless manner, ensuring the availability of service all the time. This continuous service provision can be hindered by many factors including, but not limited to: device mobility, service code mobility, lease expiry, hardware and software failures. One of the important issues that impede the continued service provision is that of invalidity of bindings. The binding is said to be invalid when the reference (stub) to the bound service becomes invalid. There are various situations that may cause bindings to become invalid, for an instance, when the bound service is moved to another location over a network for load-balancing purposes, or it has been moved closer to an entity accessing that bound service in order to improve service provisioning or to save bandwidth. In these situations a reference to the bound service becomes invalid upon its migration to a new location, causing all the bindings between the client and the moved service to become invalid. Service management support needs to include mechanisms (1) to deal with the issue of invalid bindings and (2) to proactively search for an alternative when the service becomes unavailable due to the mobility of device to immaculately keep the service provision process reliable and invisible.

3.3 SECURITY

The nomadic nature of clients and services in a pervasive computing poses various security challenges on the service discovery protocols. Clients and services in an unfamiliar environment need to maintain a level of security while exchanging data. Some of the existing approaches to ensure security include access control, encryption and cryptographic techniques. For devices to be able to communicate with each other, they need to establish a link using their wireless communication capability, such as WI-FI, Bluetooth. The process of establishing a link between two devices in close proximity is referred to as pairing or association. An example of pairing of two devices would be a pairing of PDA with a smart screen. As the form of communication is wireless, the pairing process is susceptible to many attacks including man-in-the-middle (MiTM), denial-of-service (DoS). Since pervasive devices greatly vary with respect to wireless communication capability, processing power, sensing technology, etc, it is difficult to provide a single solution for secure pairing of devices. While existing discovery protocols provide some form of security, this issue needs to be further investigated and the strong support for security be provided.

3.4 DYNAMIC SERVICE CHARACTERISTICS

Service characteristics are used to express services in terms of their attributes. For example, the printer service may have various service attributes that describe the printer service, such as printer type, printer status, location, etc. The service registers or advertises itself through the use of its service characteristics, which help in its identification and discovery. Services in the environment tend to have a change in service description due to their mobility. For example, an mp3 player service offered by a PDA whose location attribute is specified as "cafe" requires a

Authorized licensed use limited to: to IEEExplore provided by University Libraries | Virginia Tech. Downloaded on July 24,2020 at 13:42:13 UTC from IEEE Xplore. Restrictions apply.

change in location attribute of the service characteristics when the owner changes her location so that the mp3 player service can be discovered according to its new location. There may be situations where service attributes other than location attribute need to alter. For example, a service attribute of the printer service, which indicates a number of print jobs. The number of print jobs at the printer may change when more jobs are submitted to it or the pending jobs have been printed. This requires altering the service attribute of the printer service to reflect this change.

Dynamic update in the service description through its attributes is frequently required to provide up-to-date information for the selection of most appropriate service .These changes are possible only when the service attributes are allowed to be specified as dynamic attributes. Therefore, service discovery protocols need to provide a means that allow monitoring of the service state (e.g., a number of print jobs at the printer) and other service properties (e.g. location of the mp3 player service) and the support for dynamic attributes to dynamically update service attributes.

4 DISCUSSION

As can be noted from the review, much work has been done for providing efficient service discovery and dissemination using service discovery protocols in order to decrease the administration overhead from a client or service providing device. However, the suitability of current SDPs for service discovery in the pervasive computing environment is still debatable. In lieu of comparison criteria identified in section 3, we have reviewed the existent SDPs. Table 1 summarizes the design features and their support in the current discovery systems, enlightening the areas which still need consideration and revamping. We have discussed the conjecture from the table below.

Although many of the newly developed service discovery protocols realize the importance of context-awareness to achieve the discovery of most optimal service using location context, the utilization of context beyond that has gone unnoticed The support for adaptation to contextual information such as device characteristics ((e.g. small screen size, low battery power), user activity, temperature levels, etc is an important requirement of pervasive computing so this needs to be incorporated in service discovery protocols to make them suitable for their use in pervasive computing environments. The issue of contextual inconsistencies is another important factor that leads to system incorrectness, which needs to be resolved.

Service management is an important feature that has to ensure continued service provisioning and to provide mechanisms for service composition. One of the main causes affecting the continued service provision is mobility, which is inherent characteristic of pervasive computing environments. Mobility involves device mobility and code mobility (involves moving the service from one location to another over network for various reasons, including load-balancing or better utilization of resource-limited devices). Code mobility raises the issue of invalid bindings between the application and moved services. Individual services available in the environment may not meet the user's requirements and they need to be integrated to form a composed service to fulfill a user request. The service discovery protocols supporting pervasive computing environments must provide mechanisms and supporting infrastructure for service management.

The lack of support for dynamic service attributes also hinders the most appropriate service provision. Other than that, security and privacy are some challenges which need to be addressed for the secure pairing and confidentiality of a client device in an unfamiliar, pervasive computing environment

Table 1. Comparison of prevalent service discovery systems

Comparison Criteria	**UPnP**	**Jini**	**SLP**	**Salutation**	**Bluetooth SDP**	**Bonjour**	**Scooby**	**PoP Framework**	**PCRA**	**Konark**
Context Monitoring, Processing & Support for Context inconsistencies	×	×	×	×	×	×	×	×	×	×
Context Adaptation	×	×	×	×	×	×	✓	✓	✓	×
Service Composition	×	✓	×	×	×	×	✓	×	✓	×
Resolution of Invalid bindings	✓	✓	×	×	×	×	✓	×	✓	✓
Continued Service Availability	✓	✓	×	✓	×	×	✓	✓	×	✓
Support and Monitoring of dynamic service attributes	×	×	×	×	×	×	✓	×	×	×
Auto- Configuration	✓	✓	✓	✓	×	✓	×	×	×	✓
Language Independence	✓	×	✓	✓	✓	✓	×	✓	×	✓
Secure Device Pairing	×	×	×	×	✓	×	×	✓	×	×

Authorized licensed use limited to: to IEEExplore provided by University Libraries | Virginia Tech. Downloaded on July 24,2020 at 13:42:13 UTC from IEEE Xplore. Restrictions apply.

5 CONCLUSION

Pervasive Computing Environments are composed of heterogeneous services. Service discovery is an essential requirement of pervasive computing applications for providing services to the users based on context without requiring their attention. In this paper, we have presented a discovery protocol's comparison framework and evaluated various service discovery protocols based on it. We have noted some unaddressed issues, which need to be attended by researchers while developing discovery system for pervasive computing environments. These issues include context inconsistencies and context adaptation, service management, support for dynamic service characteristics and security in terms of device pairing.

REFERENCES

[1] M. Weiser., "The computer for the 21st century," ACM SIGMOBILE mobile computing and communications review 3.3, pp. 3-11, 1999.

[2]C. N. Ververidis, George C. Polyzos, "Service discovery for mobile ad hoc networks: a survey of issues and techniques," Communications Surveys & Tutorials, IEEE, vol. 10.3, pp. 30-45, 2008.

[3] B. K. Talal and M. Rachid, "Service Discovery- A Survey and Comparison," International Journal of UbiComp (IJU), vol. 4, no. 3, Jul. 2013.

[4] Feng Zhu, Matt Mutka, and Lionel Ni, "Service discovery in pervasive computing environments," in IEEE Pervasive computing, vol. 4.4, 2005, pp. 81-90.

[5] E. Al-Masri and Q. H. Mahmoud, "Device-aware discovery and ranking of mobile services," in Consumer Communications and Networking Conference, vol. 6th IEEE, 2009, pp. 813-817.

[6] H. Tianfield, "Context-Aware Service Discovery in Pervasive Environments: A Survey," International Transactions on Systems Science and Applications, vol. 7, no. 3/4, pp. 314-338, Dec. 2011.

[7] C. Bettstetter, C. Renner, "A comparison of service discovery protocols and implementation of the service location protocol," in Proceedings of the 6th EUNICE Open European Summer School: Innovative Internet Applications, 2000.

[8] W. K. Edwards, "Discovery systems in ubiquitous computing," Pervasive Computing, IEEE 5, vol. 5, no. 2, pp. 70-77, 2006.

[9] (2010) The Community Resource for Jini Technology. [Online]. http://www.jini.org/

[10] (2010) Universal Plug and Play (UPnP) Forum. [Online]. http://www.upnp.org/

[11] J. Kempf and P.S. Pierre, Service Location Protocol for Enterprise Networks: Implementing and Deploying a Dynamic Service Finder. Canada: John Wiley & Sons, Inc., 1999.

[12] B. A. Miller and C. Bisdikian, Bluetooth revealed: the insider's guide to an open specification for global wireless communication, 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

[13] Salutation Architecture Specification Version 2.0c. (1999, Jun.) [Online]. http://www.salutation.org

[14] Rakotonirainy, Andry, and G. Groves, "Resource discovery for pervasive environments," On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE, no. Springer Berlin Heidelberg, pp. 866-883, 2002.

[15] Adrian Friday et. al, "Supporting service discovery, querying and interaction in ubiquitous computing environments," Wireless Networks, vol. 10.6 , pp. 631-641, 2004.

[16] Troll, Ryan. "Automatically choosing an IP address in an ad-hoc Ipv4 network."Work in progress, IETF Internet Draft draft-ietf-dhc-ipv4-autoconfig-04. txt(1999).

[17] Bonjour. [Online]. http://www.apple.com/support/bonjour

[18] Y. A. Malkani, "A Proof-of-Proximity Framework for Device Pairing in Ubiquitous Computing Environments," Ph.D. Thesis, University of Sussex, 2011.

[19] L. D. Dhomeja, "Supporting policy-based contextual reconfiguration and adaptation in ubiquitous computing. Diss," Ph.D. Dissertation, School of Informatics, University of Sussex.

[20] L. D. Dhomeja, Y. A. Malkani. A. A Shah and K. Khoumbati ,"System Support For Managing Invalid Bindings," International Journal of UbiComp (IJU), vol. 2, no. 3, pp. 39-51, 2011.

[21] L. D. Dhomeja, Y. A. Malkani. A. A. Shaikh and A. Keerio, "Transparent Caching of Virtual Stubs for Improved Performance in Ubiquitous Environments," International Journal of UbiComp (IJU), vol. 2, no. 4, pp. 1-4, 2011.

[22] S. Helal, N. Desai, V. Verma, C. Lee, "Konark-a service discovery and delivery protocol for ad-hoc networks," Wireless Communications and Networking, vol. 3, no. 2003 IEEE, pp. 107-2113, 2003.

[23] S. Helal, N. Desai, V. Verma, C. Lee et al., "Konark: A system and protocols for device independent, peer-to-peer discovery and delivery of mobile services," in Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 33.6, 2003, pp. 682-696.

[24] J. Robinson, I. Wakeman, and D. Chalmers, "Composing software services in the pervasive computing environment: Languages or APIs?," Pervasive and Mobile Computing , vol. 4.4, pp. 481-505, 2008.

[25] N. B. Schilit, N. Adams, and R. Want, "Context-aware computing applications," Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on. IEEE, pp. 85-90, 1994.

[26] D. Chalmers, "Contextual Mediation to SupportUbiquitous Computing," PhD thesis, Imperial College, London, 2002.

[27] G. Chen and D. Kotz, "A Survey of Context-Aware Mobile Computing Research," Dartmouth College, Technical Report TR2000-381, 2000.

[28] A. K. Dey and G. D. Abowd, "Towards a better understanding of context and context-awareness," Springer Berlin Heidelberg, no. Handheld and ubiquitous computing, 1999.

[29]V. Raychoudhury, J. Cao, M. Kumar, D. Zhang, "Middleware for pervasive computing: A survey," Pervasive and Mobile Computing, no. 9.2, pp. 177-200, 2013.

Authorized licensed use limited to: to IEEExplore provided by University Libraries | Virginia Tech. Downloaded on July 24,2020 at 13:42:13 UTC from IEEE Xplore. Restrictions apply.