Context-Aware Service Discovery in Heterogeneous Networks

Pierre-Guillaume Raverdy, Valérie Issarny

HAL Id: inria-00414949

https://inria.hal.science/inria-00414949

Submitted on 10 Sep 2009

Context-Aware Service Discovery in Heterogeneous Networks

Pierre-Guillaume Raverdy, Valerie Issarny

INRIA-Rocquencourt
Domaine de Voluceau, 78153 Le Chesnay, France
{Pierre-Guillaume.Raverdy, Valerie.Issarny}@inria.fr

Abstract

Pervasive environments bring new challenges for service management, as clients have to handle large collections of services offered by concurrent networks at their location, as well as services on networks in known remote locations. To enable service discovery in such service-rich environments, we introduce the Multi-protocol Service Discovery (MSD) platform that interacts with existing SD protocols and extend their service descriptions with context information. Service information and discovery requests are then disseminated between interconnected networks based on service and network profiles.

1. Introduction

The ubiquitous computing vision can now be realized, thanks to latest consumer-oriented mobile devices that are powerful enough to host services, and the deployment of wirelessly accessible services embedded into the environment to support mobile users in their daily life. The widespread integration of wireless technologies such as Bluetooth or WiFi has enabled complex heterogeneous networks to emerge, with a mix of infrastructure-based and/or infrastructure-less networks available to users at a location. Such environments bring new challenges to service management, and service discovery in particular.

Over the years, many academic and industry-supported service discovery (SD) protocols have been proposed for specific networks (e.g., Jini [6] for intranets, UPnP/SSDP [8] for home networks). While efficient for the targeted environment, existing SD protocols prove to be inefficient or not applicable in different network settings. It is in fact unlikely for a single efficient SD protocol to emerge, due to diverse environment constraints and the de-facto standard status of some of the existing protocols. A service discovery solution for pervasive environments must thus address the issues of: (i) interoperability among existing SD protocols, and (ii) propagation of SD information across loosely coupled heterogeneous networks. Context-awareness and semantic knowledge, which have been seldom investigated in current SD protocols, also prove fundamental for the effective execution of mobile applications in such service-rich environments [2]. To address the above issues, we propose the Multi-Protocol Service Discovery (MSD) platform, which is a comprehensive SD solution for heterogeneous networks.

In Section 2 of this paper, we describe the different components of the MSD platform and its common service description format. In Section 3 we elaborate how service discovery is performed in the MSD platform and in Section 4 we details remote service access. We finally conclude in Section 5.

2. The MSD Platform

We define an Elementary Network (EN) to be a set of devices that belong to the same administrative domain (e.g., IP subnet) and share the same IP-level set of communication functionalities (e.g., IP multicast). Examples of EN include hotspots, simple home networks, or MANETs. Several SD protocols may be in use in the same EN. We then define a Global Network (GN) to be a set of Elementary Networks, with some devices being connected to multiple ENs at the same time. Devices in the same GN, but in different ENs may or may not be able to communicate (e.g., no global IP routing, security restrictions).

2.1. Architectural Overview

The MSD platform is composed of the following components: (i) *MSD Managers* that provide context-aware SD to client applications, (ii) *SD Plugins* that interact with specific SD protocols on behalf of an

MSD Manager; and (iii) *MSD Bridges* that disseminate service information in the GN.

The MSD Manager is the central component of the MSD platform. Each EN dynamically elects an MSD Manager that activates the relevant SD plugins, based on SD protocols activity detected in the EN. The MSD Manager interacts with its SD plugins to collect service descriptions and perform discovery requests. It also interacts with MSD Bridges in the EN to disseminate service descriptions and discovery requests in the GN. Each MSD Manager maintains a repository for the local and remote service descriptions that it receives.

Each SD Plugin interacts with a specific SD protocol (e.g., Jini, SSDP) on behalf of it's MSD Manager. Depending on the SD protocol, a SD plugin either registers for service announcements or performs discovery requests. In the MSD platform, a common format is used for describing services: the MSD service description (§2.2). This description format not only contains service information derived from the original service description, but also contains context information added by the SD plugin and its MSD Manager. Different MSD Managers may support different sets of SD plugins.

Within the MSD platform, devices simultaneously connected to different ENs may be selected by the MSD Manager to act as MSD Bridges. MSD Bridges propagate service descriptions and discovery requests between the ENs on their different network interfaces according to specific propagation rules (i.e., to avoid message duplication and network flooding). Different MSD Bridges within an EN may not propagate the same service descriptions and discovery requests (e.g., if the target ENs do not have the same bandwidth or the same privacy policies). MSD Bridges also enable remote service access between clients and services in different ENs when direct connectivity is not possible by providing application-level routing. We argue that such federation of centralized managers supporting different SD protocols is the best approach to provide accurate and responsive service discovery while limiting processing and communication overhead.

2.2. MSD Service Description

MSD service descriptions generated by SD plugins contain: (i) *creation information*, (ii) *service information*, (iii) *context information*, (iv) *propagation information*, and (v) *remote access information*. The *creation information* is used by MSD Managers (local and remote) to learn about the MSD components that generated the MSD description (i.e., MSD Manager and SD plugin). The *service information* is used by client applications to learn on the basic features of the service (e.g., its methods or attributes) and on the SD protocol it was advertised with. The *context information* is primarily used to describe the environment where the service is hosted and aggregates the semantic and context information added by the MSD Manager and the SD plugin. The *propagation information* is primarily used by MSD Bridges to evaluate if they should forward the MSD service description. This information is composed of a set of propagation rules, and a list of propagation hops that represent the different MSD Bridges and networks the MSD service description already crossed during its dissemination. Finally, the *remote access* part contains information related to the remote access of the service. It indicates which methods of the service support remote access along with their requirements (e.g., bandwidth), and how to set up the communication channel to be used for remote service access.

3. Service Discovery in the MSD platform

A discovery request or MSD request is created by an MSD Manager on behalf of a client application and is composed of three parts: (i) a *service description* part where the client specifies the attributes of the service it is looking for; (ii) a *request propagation* part, which is similar to the propagation information in the MSD service description; and (iii) a *request processing* part that defines how MSD Managers, both local and remote, should return the results of the MSD request.

A client application interacts with the MSD Manager in its EN to issue an MSD request, that specify the characteristics of the requested service as well as the characteristics of its context (i.e., execution environment). The client also selects between different methods for collecting the request results: local and remote results may be returned independently, or together after a given timeout. The initial (local) MSD Manager reviews the SD protocols it supports and, depending on whether the protocol is push-based or pull-based, it either checks for potential services in its repository, or transmits the request to the relevant SD plugins. Each SD plugin then performs the relevant SD-specific discovery request, and returns the corresponding MSD service descriptions.

The MSD Manager also forwards the request to local MSD Bridges (if any) that evaluate if they should propagate it based on network profiles and status, and the propagation rules of the request. For each request it propagates, the MSD Bridge maintains a pending-request entry in order to return the results. Once expired, the entry is deleted, and the MSD Bridge does not return results for this request anymore. An MSD Manager receiving a remote MSD request evaluates it

and returns the results. The MSD Manager also forwards the MSD request to the other MSD Bridges in its EN that evaluate if the request should be further propagated in the GN.

4. Remote Service Access

Discovering services hosted on nearby networks is only relevant if they can be accessed by client applications. The MSD platform enables such remote access. However, our objective is not to provide complete and transparent access: some methods of a service may not be accessible due, for example, to unmet requirements (e.g., bandwidth). The MSD platform relies on adaptors provided by service developers to translate calls between the generic service interface exposed in the MSD description and the native service interface, and to define the service requirements. We assume that clients explicitly use the MSD generic interface to access these remote services.

As each EN is managed independently, IP connectivity between the client and the remote service cannot be guaranteed. Prior to instantiating the service adaptor, and based on the propagation list from the MSD service descriptions, the relevant MSD Bridges are contacted to create a communication channel to provide application-level routing. Carefully selecting devices for acting as MSD Bridges is thus critical. In particular, the MSD platform targets consumer-oriented pervasive environments, and we assume that most ENs in the environment are not highly mobile, and that in most cases a few stable devices will be available.

4. Related Work and Conclusion

We presented the MSD platform that addresses context-awareness and interoperability issues for SD protocols in pervasive environments. Some projects have also investigated the use of a common representation of service information to address SD protocols interoperability [4][7]. Some middleware platforms [1][5] have also investigated the full interoperability between clients and services (i.e., both service discovery and access). These projects however only consider SD protocol interoperability within the same network. On the other hand, some projects [3] have investigated overlay networks and application-level routing to enable service discovery in heterogeneous networks, but often assume global IP routing or limited service discovery in specific networks selected by the user.

The MSD platform combines the two approaches, and further investigate how application data (i.e., service descriptions) must be propagated depending on the underlying networks. We argue that a fully transparent approach for service discovery and access, while feasible for some discovery protocols and in specific network configurations, is not feasible in UBISEC GNs due to networking and security constraints, and may not even be desirable as client applications may not be able to distinguish local and remote services (i.e., location-based services). We are confident that our solution is well suited for mobile and pervasive environments, as the mass marketing of powerful CE devices (e.g., portable gaming devices and multimedia players) with multiple network interfaces and P2P-oriented services will create a network-rich and service-rich environment. The project also aims at defining the different profile types that can be collected from pervasive environments, and how they should be processed to provide effective context information.

5. Acknowledgements

6. References

[1] J. Allard *et al*, "Jini Meets UPnP: An Architecture for Jini/UPnP Interoperability," In 2003 International Symposium on Applications and the Internet (SAINT 2003).

[2] P. Bellavista *et al*, "Context-aware Middleware for Resource Management in the Wireless Internet", IEEE Transactions on Software Engineering, Special Issue on "Wireless Internet", 2003

[3] P. Castro *et al*, "Locating Application Data Across Service Discovery Domains", 7th International Conference on Mobile Computing and Networking, 2001.

[4] A. Friday *et al*, "Supporting Service Discovery, Querying and Interaction in Ubiquitous Computing Environments". ACM Baltzer Wireless Networks (WINET) Special Issue, 10(6), 2004

[5] P. Grace *et al*, "ReMMoC: A Reflective Middleware to Support Mobile Client Interoperability". In International Symposium on Distributed Objects and Applications, 2003

[6] Jini (http://www.jini.org/)

[7] T. Koponen *et al*, "Service Discovery: A Service Broker Approach" In Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04)

[8] Simple Service Discovery Protocol - Internet Draft - (http://www.upnp.org/)

[9] UBISEC IST Project (http://www.ubisec.org)