Chapter 5
Service Discovery Architecture and Protocol Design for Pervasive Computing

Feng Zhu
University of Alabama in Huntsville, USA

Wei Zhu
Intergraph Co, USA

Matt W. Mutka
Michigan State University, USA

Lionel M. Ni
Hong Kong University of Science and Technology, China

ABSTRACT

Service discovery is an essential task in pervasive computing environments. Simple and efficient service discovery enables heterogeneous and pervasive computing devices and services to be easier to use. In this chapter, we discuss the key issues and solutions for service discovery architecture and protocol design for pervasive computing environments. Service design addresses the static and dynamic properties of services. Directory design focuses on scalability, topology, and infrastructure issues. Service integration uses services as building blocks to achieve complex services. Cross-layer design optimizes the performance of the protocols for ad hoc and sensor networks by integrating service discovery processes into lower layers of the network protocols. Security and privacy design protects the information, communication, devices, and services. We also point out the future research issues.

DOI: 10.4018/978-1-60960-735-7.ch005

INTRODUCTION

In pervasive computing environments, people are surrounded by a variety of computing devices. Those devices communicate with each other and provide network services and information without people's active attention (Weiser, 1991). Presently, PCs, smartphones, MP3 players, and laptops surround us. In the near future, additional networked computers, ranging from sensors, RFID tags to extremely dynamic and heterogeneous devices will provide a variety of services. It becomes overwhelming to manage these devices, configure different kinds of applications, and dynamically find the available computing services in such pervasive computing environments.

Service discovery protocols enable computers to be easier to use. They facilitate interaction between computers, with an aim to approach zero administration overhead and therefore free users from tedious and redundant administrative and configuration work. Therefore, service discovery research is critical to the success of pervasive computing (Kindberg & Fox, 2002).

The objective of this chapter is to discuss the key design issues and solutions for service discovery protocols in pervasive computing environments. In the last 15 years, many service discovery protocols were designed by industry, academia, and international standards development organizations. The protocols emphasize on aspects of the service discovery. We analyze the design of the major components, their interactions, service selection, performance optimization, and security and privacy issues.

BACKGROUND

Before we present representative service discovery protocols, we describe three general models. The models focus on the functionalities of computing devices or software in the service discovery processes and the basic steps that each device or software component is taken.

Service Discovery Models

There are three service discovery models. A trivial service discovery model is one in which a *client* (computing device) knows a *service* (network service) in advance, or the client has already cached the service's information, so that the client does a local lookup before contacting the service. The second model is the client-service model as shown in Figure 1 (a). The model performs best in simple environments such as home environments. Clients inquire about all services. If a service matches the client's enquiry, it replies back. Then, the client communicates with and accesses the service. To support thousands of computing services, such as the services in public environments, we may optionally use *directories* to store all the service information. This third model is call client-service-directory model. A client queries a directory for service information and then contacts services. We discuss the different perspectives of the clients, services, and directories.

Client View

In most cases, a client is a program that runs on behalf of a user and interacts with the user. It usually takes the following steps.

- A client queries directories for services. A client either browses services or looks for a specific service.
- Alternatively, without going through directories, a client directly queries all the services. All the services that meet the query requirement reply back to the client.
- Then the client program or the user selects a service to use.
- Finally, the client uses the service.