Building Web Services in P2P Networks

Jihong Guan
Tongji University, China

Shuigeng Zhou
Fudan University, China

Jiaogen Zhou
Wuhan University, China

INTRODUCTION

Nowadays peer-to-peer (P2P) and Web services are two of the hottest research topics in computing. Roughly, they appear as two extremes of distributed computing paradigm. Conceptually, P2P refers to a class of systems and applications that employ distributed resources to perform a critical function in a decentralized way. A P2P distributed system typically consists of a large number of nodes (e.g., PCs connected to the Internet) that can potentially be pooled together to share their resources, information, and services. These nodes, taking the roles of both consumer and provider of data and/or services, may join and depart the P2P network at any time, resulting in a truly dynamic and ad-hoc environment. Apart from improving scalability by avoiding dependency on centralized servers, the distributed nature of such a design can eliminate the need for costly infrastructure by enabling direct communication among clients, along with enabling resource aggregation, thus providing promising opportunities for novel applications to be developed (Ooi, Tan, Lu, & Zhou, 2002).

On the other hand, Web services technologies provide a language-neutral and platform-independent programming model that can accelerate application integration inside and outside the enterprise (Gottschalk, Graham, Kreger, & Snell, 2002). It is convenient to construct flexible and loosely coupled business systems by application integration under a Web services framework. Considering Web services are easily applied as wrapping technology around existing applications and information technology assets, new solutions can be deployed quickly and recomposed to address new opportunities. With the acceleration of Web services adoption, the pool of services will grow, fostering development of more dynamic models of just-in-time application and business integration over the Internet.

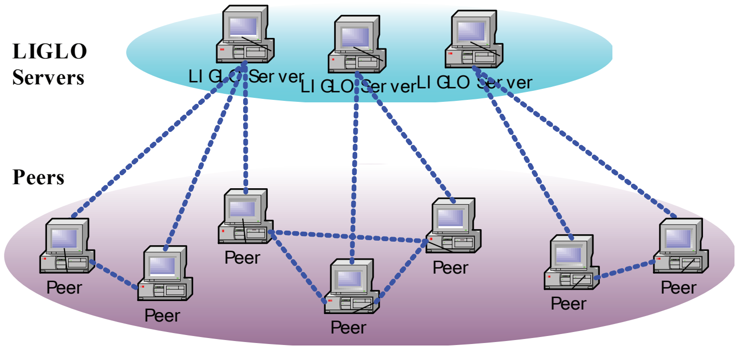
However, current proposals for Web services infrastructures are mainly based on centralized approaches such as UDDI: a central repository is used to store services descriptions, which will be queried to discover or, in a later stage, compose services. Such centralized architecture is prone to introducing single points of failure and hotspots in the network, and exposing vulnerability to malicious attacks. Furthermore, making full use of Web services capabilities using a centralized system does not scale gracefully to a large number of services and users. This difficulty is severe by the evolving trend to ubiquitous computing in which more and more devices and entities become services, and service networks become extremely dynamic due to constantly arriving and leaving service providers.

We explore the techniques of building Web services systems in a P2P environment. By fitting Web services into a P2P environment, we aim to add more flexibility and autonomy to Web services systems, and alleviate to some degree the inherent limitations of these centralized systems. As a case study, we present our project *BP-Services.* BP-Services is an experimental Web services platform built on BestPeer (http://xena1.ddns.comp.nus.edu.sg/p2p/)—a generic P2P infrastructure designed and implemented collaboratively by the National University of Singapore and Fudan University of China (Ng, Ooi, & Tan, 2002).

FITTING WEB SERVICES INTO A P2P FRAMEWORK

A *Web service* can be seen as an interface that describes a collection of operations that are network accessible through standardized XML messaging (Gottschalk et al., 2002). Web services consist of three roles and three operations: the roles are *providers, requesters,* and *registrars* of services, and the operations are *publish, find,* and *bind.* The service providers are responsible for creating Web services and corresponding service definitions, and then publishing the services with a service registry based on UDDI specification. The service requesters first find the services requested via the UDDI interface, and the UDDI registry provides the requesters with WSDL service descriptions and URLs pointing to these services themselves. With the information obtained, the requesters can then bind directly with the services and *invoke* them.

Figure 1. Network topology of BestPeer



Over the last few years, many P2P systems have been developed and deployed for different purposes and with different technologies, such as Napster (http://www.napster.com/), Gnutella (http://gnutella.wego.com/), and Freenet (http://freenet.sourceforge.com/), to name a few. The architecture of these systems can be categorized into three groups mainly based on their network topologies: centralized P2P, pure P2P, and hybrid P2P systems (Yang & Garcia-Molina, 2001). In a centralized P2P network, there is a central server responsible for maintaining indexes on the metadata for all the peers in the network. Pure P2P is simply P2P systems with fully autonomous peers—that is, all nodes are equal, no functionality is centralized, and the communication between peers is also symmetric. Hybrid P2P is a kind of tradeoff between centralized P2P and pure P2P, which is structured hierarchically with a supernode layer and a normal peers layer.

Fitting Web services into P2P framework is to adapt Web services to P2P environment, which results in the so-called *P2P Web services,* or simply *P2P services*. Here P2P service is different from the ordinary Web services at least in three aspects. First, typically a peer in P2P services takes all three roles of services provider, consumer, and registrar, whereas in ordinary Web services, a node can typically be a producer and/or a consumer, but not a registrar at the same time. Second, generally speaking, servers in ordinary Web services systems are well-known hosts, with static IP addresses and on the outside of a firewall. However, this is not usually the case in the P2P world. A services node may join or depart the P2P services network at any time. Third, the preferred method of finding Web services in the ordinary architecture is currently through a central repository known as a UDDI operator. Nevertheless, P2P services systems have no central server to hold UDDI registry; each peer node manages its own UDDI registry locally. So, new and efficient mechanisms for services discovery in P2P services environment are required.

Corresponding to the architecture of P2P systems, there may also be three schemes for building P2P services applications: centralized P2P services, pure P2P services, and hybrid P2P services. For centralized P2P services, there is a central server in P2P services systems. However, the central server is not used as a central UDDI registry server; instead it is used for storing metadata of services to facilitate services discovery, which includes business names, services types, URLs, and so forth. In pure P2P services systems, services UDDI registry is distributed on every services node, so there is no need for services publication of the ordinary sense, and UDDI registry maintenance is also simplified because all services information is published and maintained locally. And in hybrid P2P services systems, the supernodes will be used for storing services metadata. It is useful for services discovery to cluster services nodes based on metadata, and then register the nodes in the same cluster under the same supernode.

BP-SERVICES: BESTPEER-BASED WEB SERVICES

As mentioned, the BP-Services project aims to develop an experimental P2P-based Web services platform as a test-bed for further P2P and Web services research.

BestPeer (Ng et al., 2002) is a generic P2P system with an architecture more pure P2P than hybrid P2P. The BestPeer system consists of two types of nodes: a large number of normal computers (i.e., peers), and a relatively fewer number of *Location-Independent Global names Lookup* (LIGLO) servers. Every peer in the system runs the BestPeer software, and will be able to communicate and share resources with any other peers. There are two types of data in each peer: private data and public (or sharable) data. For a certain peer, only its public data can be accessed by and shared with other