

# Profile-based Service Browsing – A Pattern for Intelligent Service Discovery in Large Networks

**Martin Gitsels, Jochen Sauter**  
Siemens AG, Corporate Technology  
Otto-Hahn-Ring 6, 81739 Munich, Germany  
{martin.gitsels,jochen.sauter}@mchp.siemens.de

August 30, 2000

## ***Keywords***

Ad-hoc networks, Spontaneous networking, Service Browsing, Service Discovery, Quality of Service (QoS)

## ***Introduction***

This position paper describes an architectural pattern to discover network services depending on user preferences and service and terminal capabilities. For the description of the pattern we use the format proposed in [1].

## ***Intent***

Provide a system architecture which allows capability- and preference-based discovery of network services in ad-hoc networked service spaces that host large numbers of services.

## ***Motivation***

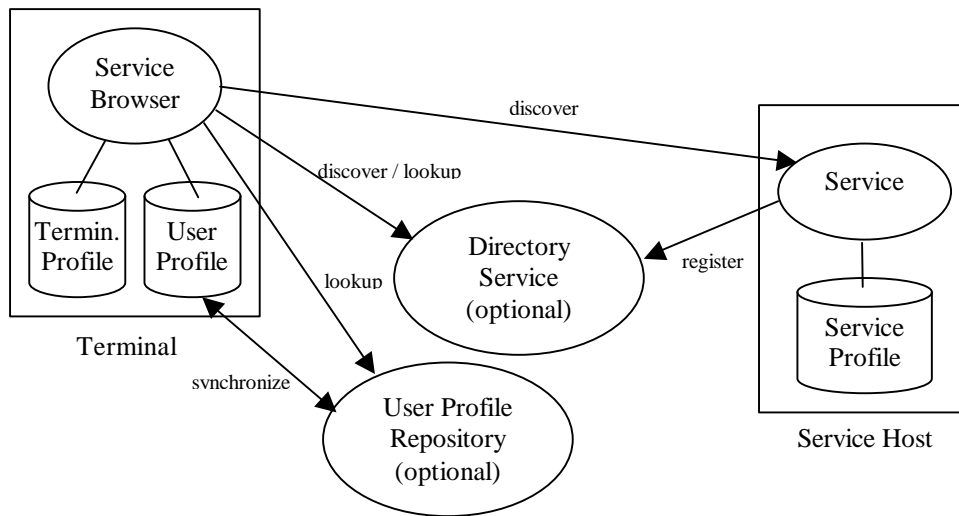
Consider an ad-hoc network of devices and services where services announce themselves spontaneously on the network. Technologies like Jini [1] or Universal Plug and Play [2] allow such dynamic service infrastructures to be managed. In order to discover services and utilize them, users can typically use a service browser. However in networks with a potentially large number of services, displaying all of them in the browser might confuse its user. Lots of services would be listed which the user is not interested in or cannot even use due to the restricted capabilities of the terminal or the network. For instance, a video service that has Quality of Service (QoS) requirements such as a certain network bandwidth cannot be used from today's mobile wireless devices because current wireless networks still have very limited bandwidth. Also, a user might want to express personal preferences such as using services only offered from certain service providers. A perfect solution would be an automatic filtering of the list of services according to QoS requirements of services, capabilities of the terminal and network, and the personal preferences of the user.

## ***Application Areas***

Use *Profile-based Service Browsing*

- when the considered ad-hoc networked service environments have potentially large numbers of services
- when a service browser is employed to find and utilize services

## Structure



## Participants

- Service: provides a device- or software-based service for clients on the network
- Service Browser: discovers available services, matches service profiles against user and terminal profiles, and provides a (graphical) interface for the user
- Directory Service (optional): provides a directory of services that are available on the network; this is especially useful in large service infrastructures
- User Profile Repository (optional): stores user profiles to be retrieved by terminals; this is especially useful if users tend to change their terminals frequently

## Collaborations

- The service browser sends out discovery messages on the network (or optionally asks a directory service) to find available services
- All available services reply and send their profiles (or alternatively the directory service returns a list of service profiles) to the browser
- The service browser performs a matching of service profiles against terminal and user profiles:
  1. The required minimal capabilities of the terminal as specified in the service profile are matched against the actual capabilities defined in the terminal profile.
  2. The user's preferences as defined in the user profile are matched against the service profiles.
- All services that do not match the specified rules are removed from the list of services which the browser presents to the user.

### ***Consequences***

1. The key benefit for the user is a dramatically restricted set of services displayed by the browser. This allows an easier and quicker interaction with the service browser.
2. Profiles for users, terminals and services have to be defined and managed. There have to be standardized formats for profiles to ensure seamless interoperability of services and browsers from different vendors.

### ***Implementation***

The following issues should be considered when implementing the pattern:

1. *Use a standard way of defining profile formats.* A meta language such as XML is useful when it comes to defining formats for user, terminal and service profiles.
2. *Restrict profile size to avoid performance drawbacks.* Large and complex profiles for each service, terminal and user can cause a lot of processing on the client side. This can result in performance problems especially on small consumer appliances such as mobile phones. In order to avoid this, keep profiles minimal.
3. *Keep matching restricted to special attributes.* The matching rules have to be carefully designed and should only include certain attributes. There are potentially many attributes in profiles that are not suitable for matching.

### ***Known Uses***

The Jini Service Browser developed by Siemens ZT [4] provides management of user and terminal profiles and matching against service profiles.

The Jini browser provided with the Jini Starter Kit [2] by Sun Microsystems provides a basic way of restricting the list of displayed services. User can specify groups that services have to belong to and service interfaces that services have to implement to get displayed by the browser.

### ***Related Patterns***

There are no known related patterns.

### ***References***

- [1] E. Gamma et al.: Design Patterns, Addison-Wesley, Reading, MA, 1995
- [2] Jini Connection technology, <http://www.sun.com/jini>, <http://www.jini.org/>
- [3] The Universal Plug and Play Forum, <http://www.upnp.org>
- [4] Jordan Grouevsky: Intelligent Service Browsing with Jini, Internal Concept Paper, Siemens AG, ZT SE 2, August 2000