

Using IETF Service Discovery Methods in IPv6 and Middleware Platforms And Implementing SLPv2 for IPv6

Bilhanan Silverajan, Jaakko Kalliosalo, Ilkka Karvinen

Abstract-- In this paper, we discuss key issues pertaining to service discovery methods, and requirements for their successful operation in future IP networks with fixed and mobile topologies. We provide an introduction of 2 important service location mechanisms supporting dynamic discovery, proposed by the Internet Engineering Task Force (IETF), Zeroconf-based service discovery and Service Location Protocol (SLP) based service discovery. We then discuss their migration from IPv4 to fixed and mobile IPv6 networks. The main benefits and use with middleware in such future IP networks are mentioned. Finally, we quickly present our implementation of SLP Version 2 in IPv6, and then briefly introduce the Distributed Event Monitor (DEMON), a runtime tool in the DOORS framework that uses SLP and CORBA to discover and monitor DOORS events in a network.

Index terms-- Service Discovery, IPv6

I. INTRODUCTION

Today, the cost of public network connectivity and computing devices have become so affordable, the ordinary user is able to consider electronic services and transactions for daily needs. At the same time computing devices such as laptops, phones and handheld devices, are becoming very powerful and increasingly more portable and mobile. They are also being developed to take advantage of widespread network connectivity with low-cost GPRS and public or company-wide wireless LAN hot-spots and intranets. Consequently, not only is the number of users increasing daily, they are becoming increasingly mobile as well. It may hence become impossible to manually configure every device being networked.

These expectations will place important demands in future IP networks. It is therefore commonly hoped that the migration to IPv6 will fulfil the need in future for large address spaces as well as address and routing auto-configuration. In addition to being used as backbone and

enterprise-level fixed, wireless and mobile networks, IPv6 networks are widely anticipated to become ubiquitously utilised as intelligent home networks, automobile networks, airplane networks and ad-hoc mobile networks, while offering highly improved mobility and security features.

This expected increase in the number of networked devices will also bring with it a natural diversity of applications and services. From a service development aspect, the impetus for deploying middleware solutions that adapt well to fixed and mobile IPv6 environments is high. With demands being placed from both the networking as well as the telecommunications worlds, all indications are several different middleware solutions may concurrently reside in devices and fixed and mobile IPv6 networks, and would need to have some level of interoperation with each other.

All these imply, there will also be a significant impact on the way autonomous and interactive services and middleware are discovered, implemented, used and managed. Support for location awareness in the network and device become very important. For example, when devices roam across networks using mobile IPv6 wirelessly or otherwise, we cannot assume that they are always able to rely on static configuration for rapidly locating application and network services in their immediate surroundings. Dynamic discovery is needed, especially in foreign networks the device has no previous knowledge of.

In this paper, we discuss key issues pertaining to service discovery methods, and requirements for their successful operation in future IP networks with fixed and mobile topologies. Of the many different service discovery mechanisms available, we introduce and focus on the 2 important service location methods worked on by the IETF to support dynamic discovery, and discuss their migration from IPv4 to fixed and mobile IPv6 networks. The main benefits and use with middleware in such future networks are mentioned, as we anticipate the dynamic service discovery paradigm to encompass the need to interwork and support middleware services and architecture, in addition to locating traditional network services and devices. Finally, we briefly present our implementation of the Service Location Protocol (SLP) Version 2 in IPv6, and give a quick overview of how SLP

Dept. of Information Technology,
Tampere University of Technology,
P.O. Box 553, FIN-33101 Tampere, Finland
Tel: +358-3-3115 3906 Fax: +358-3-3115 4988
{bilhanan, kalliosa, ik}@cs.tut.fi

and CORBA are used by the Distributed Event Monitor (DEMon), a runtime tool that is part of the DOORS framework, for monitoring DOORS events in a network.

II. LOCATING SERVICES

In general, the mechanism of locating a specific resource, object or service can be performed in two ways [1], "Lookup" and "Discovery".

"Lookup" refers to a passive process of locating a specific object, resource or service based on some matching criteria. It is initiated by a seeker, and requires the existence of some agent to answer the request. "Discovery" on the other hand, is used to refer to a more spontaneous process, in which many entities become aware of other entities on the network, and present themselves to other entities. Discovery protocols have the overall goal of making digital networks easier to create and use. A discovery service may be used for lookup, but many lookup services do not support discovery. Lookup services are also generally implemented as point-to-point, whereas discovery services are dependent on the underlying network's ability to use broadcast or multicast mechanisms.

A. Zero Configuration Networking

The IETF Zero Configuration Networking (zeroconf) Working Group was chartered in 1999 to understand and define the requirements for making many types of automatic host configurations in IP Networking possible, with completely no manual interference. A zeroconf system may use configured information in the network when it is available, but must also be able to operate correctly in the absence of configured information from either a user or infrastructure services such as conventional DHCP or DNS servers [2]. The benefits of a total zeroconf-aware environment of host, applications and network are obvious: It provides ease of use and management to end-users and system administrators if hosts are automatically able to obtain their own IP addresses, DNS information, resolve address conflicts, or be allocated multicast addresses. Added to these, they should dynamically discover network file and print services, LDAP servers and various other types of services. Although the zeroconf initiative is at its early stages, at least one computer vendor, Apple Computers, has begun supporting zeroconf-based services, under the trade-marked product name RendezvousTM, in its current operating systems.

Service discovery within zeroconf is covered by two ongoing Internet draft specifications: the proposal for a multicast DNS service, to enable name-to-address translation and other DNS-like operations in the absence of a conventional DNS Server, and DNS Service Discovery, which is a proposed way of extending Multicast DNS to also provide simple service discovery and network browsing.

Multicast DNS (mDNS) [3] is a joint effort by the zeroconf and DNS Extensions (dnsex) working groups to extend existing DNS infrastructure and reuse existing DNS programming interfaces, packet formats and operating semantics in a small network in the absence of a conventional DNS server. mDNS proposes the introduction of a special domain, ".local." to mean that names within this domain and all its subdomains pertain only on the local link where they originate. DNS queries for resource records, having names ending with a ".local.", must be sent to the mDNS multicast address 224.0.0.251. mDNS client queries may either be one shot (with the querier querying once and waiting for either a single reply from the network, or accumulating multiple responses before continuing), or continuous (with the querier polling the network continuously and accumulating responses in parallel to continuing operations). In the absence of a centrally managed DNS service, conventional global queries could also be sent to the mDNS multicast address.

DNS Service Discovery (DNS-SD) [4], aims to complement mDNS by allowing network browsing and service discovery using only standard DNS packets and record types. DNS-SD uses the service naming syntax and semantics of DNS SRV [5] and PTR [6] resource records for enquiring about service types and instances, and provides the possibility to browse through responses received. PTR lookups are of the form <Service>.<Domain>, whereas PTR responses and SRV lookups are of the form <Instance>.<Service>.<Domain>.

For example, an initial PTR lookup could contain a query with the record "_ipp._tcp.mycompany.com" for finding IPP printers on the mycompany.com network, where "_ipp._tcp" represents <Service> and "mycompany.com" represents <Domain>. A list of PTR record responses would then be returned, of the type "instance1._ipp._tcp.mycompany.com", "instance2._ipp._tcp.mycompany.com" and so on. The initial querier would then have the capability to browse through the list of PTR records received, for selecting the specific instance type. Subsequently, a DNS SRV record would then be requested (for example, "instance2._ipp._tcp.mycompany.com"). The result of the DNS request is a SRV record giving the port number and target host where the service may be found. Likewise, lookups for PTR records of the form "<Service>.local." are defined to use multicast, and return a list of named instances of the form "<Instance>.<Service>.local."

B. Service Location Protocol

The Service Location Protocol (SLP) was standardized by the IETF SRVLOC Working Group in 1997 [7], and subsequently revised in a second version in 1999 [8]. Being such an established protocol, SLP has formed the basis of application and device service discovery in products from several commercial vendors such as Apple Computers, Hewlett-Packard, IBM, Lexmark, Novell and

Sun Microsystems. Open-source and research versions also exist [9], [10]. SLP's functionality is contained in 3 types of agents of which the User Agent (UA) and Service Agent (SA) are mandatory elements of the protocol, and the Directory Agent (DA) is an optional element.

The UA is used by client applications to discover the locations of services. SAs tends to reside on the same hosts as server applications, but they could also reside elsewhere in the network too. Server applications register their service with SAs with a Service Type string URL. When there are only UAs and SAs present on a network, all service requests are sent to the administratively scoped multicast address 239.255.255.253. For example, a print server might register itself to an SA with "service:printer:ipp://printserver.mycompany.com/printer 1". The SA would then respond to any UA multicasting a request for a printing service, such as "service:printer" or "service:printer:ipp". SAs also multicast SA Advertisement messages on the network if solicited by a UA. In the absence of multicast, broadcast may also be used.

The DA provides a centralized service for all service announcements in a very large network, so that a point of single contact exists for a UA trying to discover various services. DAs also advertise their presence periodically and SAs are required to register their services with DAs they discover from DA advertisements. UAs also interact directly with the DA instead of SAs, if one is present in the network. In this mode, UAs essentially use the DA as a lookup service.

The discovery of a DA in a network by the UA can be accomplished in 3 ways: The UA can be statically configured with the location of a DA, it can dynamically discover the presence of a DA through DA advertisements, or it may actively solicit a DA advertisement by requesting a directory agent service type at any point in time.

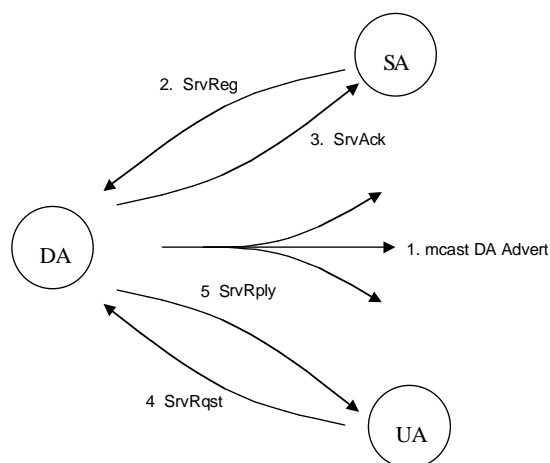


Fig. 1 SLP Service Discovery

Figure 1 depicts interactions and a possible event sequence among the three agents after the DA multicasts

its presence on a network and responds to SA Service Registrations and UA Service Requests.

C. Comparison of SLP and Zeroconf

1) Design Aims

SLP has been designed for locating services in enterprise-level networks, in which application services and devices generally do not have a very high degree of movement while possessing a large reachability likelihood. Consequently, SLP assumes a fairly constant network topology is already in place. In order to perform service discovery with SLP, existing infrastructure would need to be made aware of this new protocol as well as the presence of SLP agents on the network.

Zeroconf, on the other hand, works well for environments in which devices come and go quickly, forming ad-hoc networks or split and merge from one or more networks. Its aim is to reuse existing APIs and common DNS infrastructure already present in the network.

However, in order to fully deploy Zeroconf in a network with mDNS as well as DNS-SD, dynamic service registrations need to be rendered possible, and the registration uses the DNS Dynamic Update mechanism. Since this also implies that only trusted systems conforming to a zone update policy should be allowed to update DNS resource records, the use of DNSSEC is implicitly encouraged. At this point of time, it is a matter of contention as to whether the use of DNSSEC and DNS Dynamic Updates are prevalent at all.

2) Protocol Behaviour in Future IP networks

Because the Zeroconf draft specifications take into account IPv6 networks, its usage is relatively straightforward. All link-local multicast transmissions would use FF02::FB, the IPv6 equivalent of the IPv4 link-local multicast address. The use of purely link-local addresses for any multicast query, assures that Zeroconf is guaranteed to work across both fixed IPv6 and mobile IPv6 connections, and remains backwards compatible with its IPv4 counterpart.

In mobile IPv6 cases, the routers residing in the local link assuredly would not need to pass the multicast packet out of the link, regardless of whether they originate from a local node on that link, or a Mobile Node (MN) visiting on that link. This will allow direct interaction between the MN and other nodes on the foreign link without intervention from the home network.

On the other hand, the use of SLP in IPv6 networks is considerably different from its IPv4 counterpart. Service requests using broadcast methods are replaced with link-local multicast. Also, instead of a fixed multicast address for SLP messages, IPv6 multicast addresses are dynamically calculated using a hash algorithm for the

various “service:” types, effectively rendering the IPv6 version incompatible with the IPv4 version. The addresses range from FF0X:0:0:0:0:1:1000 to FF0X:0:0:0:0:1:13FF, and DA advertisements use FF0X:0:0:0:0:0:123. The value of X can be 1, 2 or 5 and thus defines the scope of the service to be either node-local, link-local or site-local [11].

Because of the extensive need and dependence on proper multicast support, the use of SLP in mobile IPv6 is currently a problem. The current mobile IPv6 draft specification outlines two completely different, but acceptable solutions of multicast routing to and from Mobile Nodes (MNs) [12]: The first would require a tunnelling solution between the home network and the MN when it is roaming, while the second involves the foreign network allowing multicast routing with the MN as if it is a completely local node. An MN would only be able to directly discover site-local services in a foreign network if the second approach is used, but it would not possess enough information beforehand which of the two routing alternatives is enforced in the foreign network. Consequently, the MN, while situated in a foreign network, is currently only guaranteed to directly discover and offer services at the link local scope, similar to Zeroconf.

3) Application-Level Mobility Detection in Mobile IPv6

Mobility detection is a key factor for the correct functioning of service discovery by an MN. It provides a notion of location awareness to the application, supplying a realization of both its temporally and spatially localized working environment with respect to its neighbouring services, network topology, administrative management or physical geography.

This in turn would allow the application to react to its immediate environment. It might decide to perform service discovery in the foreign network to understand better the types of services it would have at its disposal. It would also allow the application to take appropriate measures in registering or unregistering its services within the networks it visits or leaves.

Unfortunately mobile IPv6 shields all mobility issues from the applications above the TCP layer, and no provision for application-level mobility detection currently exists in the protocol itself. However, the importance of application-level mobility detection, especially to support location-aware application in MNs has been recognized, resulting in the recent production of 2 very preliminary draft papers [13], [14] from the IETF MobileIP Working Group.

D. Service Discovery Issues for Middleware

Several middleware specifications were born at a time when the only requirement needed of the Internet was as a

transport medium. Consequently they have developed most of their own mechanisms for services such as naming, directory, discovery and security independently of IETF efforts. However, recent technological and business developments have encouraged the standards bodies responsible for various middleware solutions to also focus on methods of interworking amongst themselves such as CORBA, XML-based Web Services and the Java-based solutions such as Enterprise JavaBeans (EJB) and Java2 Enterprise Enterprise or Standard Editions (J2EE/J2SE).

Many open issues still exist before interworking can be achieved successfully and transparently amongst various components of a distributed application running on different middleware platforms. However specification work on interworking has already begun. Current specification efforts exist to map CORBA’s Interface Description Language (IDL) and Web Services Description Language (WSDL)/Simple Object Access Protocol (SOAP) bindings [15]. EJB specifications require CORBA’s Internet InterORB Protocol (IIOP) compliance within the environment for remote invocations from J2EE clients on Session Beans and Entity Beans, as well as requiring implementations to support web service invocations using WSDL and SOAP over HTTP to support web service interoperability [16].

At the same time, the architectures are aiming to incorporate developments in future Internet protocols into existing middleware specifications, so that the increasing underlying richness in available functionality could be fully taken advantage of [17]. A good example is the recent adoption of the Multicast Inter-ORB Protocol (MIOP) into the core CORBA specifications, allowing request brokers to communicate by multicast datagram packets in addition to the current connection-oriented mechanism.

This augers well for the eventual use of dynamic mechanisms in middleware architecture, especially for service discovery. Lightweight service discovery methods such as SLP or Zeroconf would become essential as a bootstrapping mechanism for resolving initial references to basic middleware services, especially in mobile environments. In addition, they could also be used to discover middleware components and their capabilities (whether they allow communication through CORBA, J2EE, both, etc) and possibly as an eventual addition to current lookup services such as Naming, Directory or Trading which could serve to support multiple middleware platforms as well.

III. OUR WORK

A. SLP Implementation

This section illustrates the implementations of UA, SA and DA conforming to version 2 of SLP in fixed IPv6 networks as specified in RFC 3111 [11]. These agents

were implemented with the use of the DOORS event-based C++ framework. The architecture, goals, subsystems and interworking aspects of DOORS have been presented in [18] and [19], and will thus not be discussed here. Instead only the IPv6 SLPv2 specific features that have been recently implemented will be presented. The source code of the test implementation is freely downloadable as part of the DOORS package from <https://doors.atm.tut.fi/>.

Figure 2 illustrates a simplified UML diagram depicting the classes making up the overall architecture developed. The UDP6Task object is supplied by the DOORS framework, and provides a uniform and simple message-based interface send and receive messages IPv6-based UDP datagrams to the network. It is capable of understanding unicast and multicast, and is able to join and listen to multiple multicast groups. The PTask class, also part of the DOORS framework, is a base class containing commonly functionality for protocol development, such as State Machine handling functions and specialised methods which understand communication through User and Provider Service Access Point and Peer Ports.

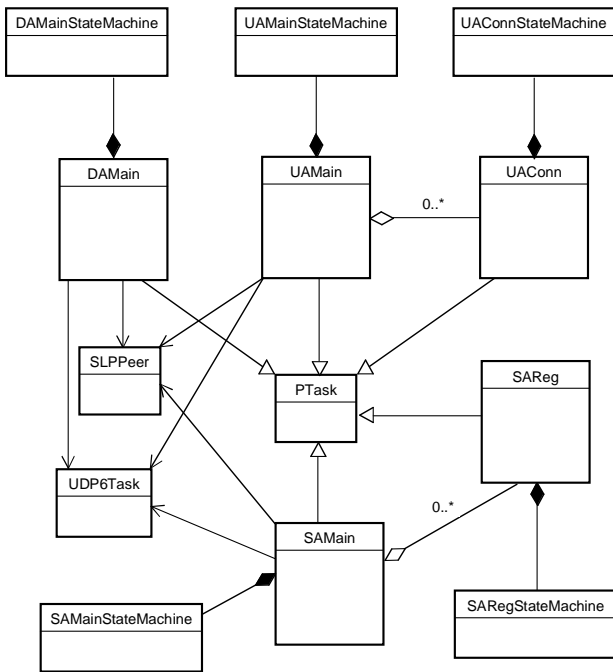


Fig 2. DOORS SLP Design for IPv6

The architecture of the SA is similar to that of the UA. Applications send service registration messages to the SAMain object, which creates new SAReg objects. Each SAReg object is responsible for 1 service registration, and it joins to the correct multicast group to respond to any SLP requests it receives.

DAMain and DAMainStateMachine implement the DA functionality. All service registrations are currently stored in memory within a C++ STL vector container, but with a little effort, its functionality can be extended to support

the integration of an LDAP-based back end to store service registrations using the schema template defined in [20].

All the five main object classes implementing the three agents of the protocol, are derived from the DOORS PTask class.

B. Distributed Event Monitoring

The Distributed Event Monitor (DEMon) is a runtime tool, developed for observing events and messages taking place within DOORS applications residing in a network. DEMon has two parts: Agents attach to running DOORS applications in one or more hosts and capture event traces. The Manager is responsible for gathering traces from the various Agents in a network, and displaying the events in chronological order to a user.

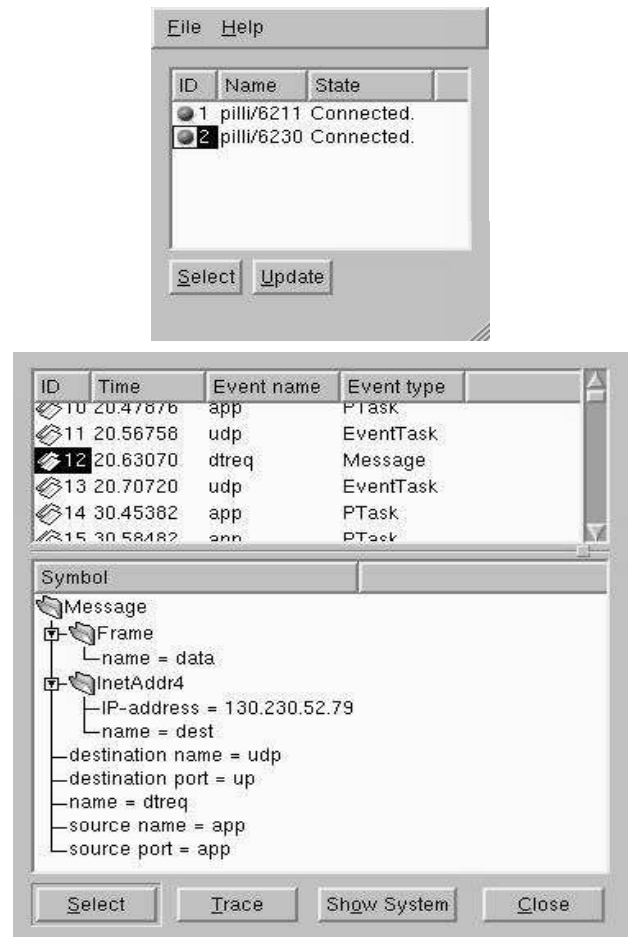


Fig 3. DEMon Manager Display

Managers and Agents are CORBA-aware, and possess CORBA Interoperable Object References (IORs). Managers and Agents in the network use SLP to dynamically discover one another's presence at startup, and to obtain one another's CORBA IORs. From that point on, all communication proceeds by way of CORBA communication: Traces sent by Agents to the Manager for

display to a user, as well as the Manager setting filters to its Agents for selectively obtaining only specific types of event traces.

DEMon was briefly introduced in [19], as an example of interworking with the OpenSLP [21] project in IPv4. It is now being worked upon to use our native implementation of SLP in IPv6. Figure 3 shows two windows, the upper one showing an agent list connected to the Manager, while the lower one shows detailed DOORS system events as reported by each selected Agent.

IV. CONCLUSIONS AND OUTLOOK

In this paper, we have discussed the growing importance and need of standard service discovery methods in future IP networks. These techniques must be lightweight and flexible, so that apart from their traditional role of aiding dynamic discovery of just devices and network services, they can also be easily and effectively incorporated into application level middleware platforms. We also discussed how mobility in future networks will influence the prevailing methods of service usage and discovery, both in simple cases and in middleware.

Just as middleware specifications need to evolve to treat IP networks as more than just a transport medium, so must IP networks also evolve to understand the needs of upper-level applications that need location awareness and mobility detection.

The widespread adoption of such efforts however rests on several factors such as user and vendor commitment. Research and development efforts show that technologically they hold good promise for future enterprise-level networking and computing needs.

V. REFERENCES

[1] Robert E. McGrath: Discovery and Its Discontents: Discovery Protocols for Ubiquitous Computing, Presented at Center for Excellence in Space Data and Information Science, NASA Goddard Space Flight Center, April 5, 2000.

[2] Williams: Requirements for automatic Configuration of IP Hosts, draft-ietf-zeroconf-reqts-12.txt, September 19, 2002.

[3] Stuart Cheshire: Performing DNS queries via IP Multicast, draft-cheshire-dnsext-multicastdns-01.txt, December 20, 2002.

[4] Stuart Cheshire: DNS-Based Service Discovery, draft-cheshire-dnsext-dns-sd-00.txt, December 20, 2002.

[5] IETF RFC2782: A DNS RR for specifying the location of services (DNS SRV), February 2000.

[6] IETF RFC1035: Domain Names – Implementation and Specification, November 1987.

[7] IETF RFC2165: Service Location Protocol, June 1997.

[8] IETF RFC2608: Service Location Protocol, Version 2, June 1999.

[9] The OpenSLP Project <http://www.openslp.org>

[10] Christian Bettstetter and Christoph Renner, “A Comparison of Service Discovery Protocols and Implementation of The Service Location Protocol”, Proceedings of 6th Open European Summer School EUNICE 2000, Enschede, The Netherlands, September 13 - 15, 2000.

[11] IETF RFC 3111, “Service Location Protocol Modifications for IPv6”, May 2001.

[12] IETF draft-ietf-mobileip-ipv6-21: Mobility Support in IPv6, February 26, 2003.

[13] IETF draft-chakrabarti-mobileip-mipext-advapi-00: Extension to Sockets API for Mobile IPv6, February, 2003.

[14] IETF draft-yokote-mobileip-api-01.txt: Mobile IP API, June 2002.

[15] Object Management Group: WSDL-SOAP to CORBA Interworking, Doc No: mars/2003-05-07, May 12, 2003.

[16] Sun Microsystems: Enterprise JavaBeans Specification Version 2.1, Proposed Final Draft, August 2, 2002.

[17] Kimmo Raatikainen: : Middleware in the Mobile Domain, Nokia Mobile Internet Technical Architecture (Technologies and Standardization), 2002 IT Press.

[18] B. Silverajan, I. Karvinen, J. Makihonka, J. Harju: The Design of a Flexibly Interworking Distributed Message-Based Framework. Proceedings of EUNICE 2000 Summerchool, Enschede, The Netherlands September 13 - 15, 2000.

[19] Bilhanan Silverajan, Ilkka Karvinen, Joonas Hartman, Jani Laaksonen: Enterprise-level Integration and Interoperability in Future Networks with DOORS Middleware. Proceedings of IFIP WG6.7 Workshop and EUNICE 2002 Summer School on Adaptable Networks and Teleservices, Trondheim, Norway September 2 -4, 2002.

[20] IETF RFC 2609, “Service Templates and Service: Schemes”, June 1999.

[21] The OpenSLP Project, <http://www.openslp.org>