Dynamic Autoconfiguration in 4G Networks: Problem Statement and Preliminary Solution

Rui Campos
FEUP and INESC Porto
Rua Dr. Roberto Frias, 378
4200-465 Porto, Portugal
rcampos@inescporto.pt

Manuel Ricardo
FEUP and INESC Porto
Rua Dr. Roberto Frias, 378
4200-465 Porto, Portugal
mricardo@inescporto.pt

ABSTRACT
The Internet is characterized by the coexistence of two Internet Protocol (IP) versions and multiple autoconfiguration mechanisms which are deployed targeting specific communication scenarios. This heterogeneity requires user pre-configurations, namely with respect to the proper autoconfiguration mechanism to be used at each time. On the other hand, future networks may imply that users own personal networks demanding self-configuration and self-management, and being part of very dynamic scenarios. In this paper we make a survey of the autoconfiguration mechanisms available for IP networks, and argue that a new solution is needed, so that the proper autoconfiguration mechanism can be selected automatically, dynamically and efficiently, and future communication paradigms can be properly addressed.

Categories and Subject Descriptors
C.2.1 [**Computer – Communication Networks**]: Network Architecture and Design – *Network communications, Packet-switched networks, wireless communications*.

General Terms
Management, Design, Theory.

Keywords
Autoconfiguration, Personal Area Networks, Self-management.

1. INTRODUCTION
In the early days of the Internet, when it was just a small research network, users were networking experts capable of performing the needed network configurations. With the years, this network evolved into a huge, worldwide network, where users were no more networking experts. This implied that either users needed to acquire "know how" in networking, or that administrators needed to configure every computer on the network; this revealed to be a hard task as the networks grew. Hence, new solutions allowing automatic configuration of each computer were developed. In the early nineties the Dynamic Host Configuration Protocol (DHCP) [10] started to be deployed, providing the means for transferring configuration information to hosts on IP networks automatically. On the other hand, the worldwide spread of the Internet, made home networking a reality, and Internet Service Providers (ISPs) started to offer Internet access. First, over dial-up accesses and, more recently, over technologies such as Digital Subscriber Line (DSL). For this type of access, namely for dial-up accesses, the Point-to-Point Protocol (PPP) [20] became the de-facto standard, providing a method for transporting multi-protocol datagrams over point-to-point (home-to-ISP) links. PPP defines a family of Network Control Protocols (NCPs) for establishing and configuring different network-layer protocols; the Internet Protocol Control Protocol (IPCP) [5] is defined for IP.

Foreseeing the exhaustion of IPv4 addresses, the Internet Engineering Task Force (IETF) began to work on a new version of the Internet Protocol (IPv6). In contrast to its precedent, IPv6 has built-in autoconfiguration mechanisms. The link-local address autoconfiguration provides network connectivity within a link; the global address autoconfiguration uses Router Advertisement messages [19] and provides global connectivity. The stateful autoconfiguration frameworks defined for IPv4 are also made available, namely DHCPv6 [11] and IPv6CP [3]. Furthermore, the stateless autoconfiguration framework specifies a mechanism, based on the Router Advertisement messages, through which a host learns about the proper autoconfiguration mechanism to use [17].

More recently a new communication paradigm based on ad-hoc networks came up, posing new challenges to automatic connectivity. In this new paradigm, the aforementioned stateful frameworks become difficult to implement, and alternative mechanisms have to be used. For IPv6, the built-in stateless mechanisms can be applied. Regarding IPv4, new amendments are needed, though. Thus, the solution presented in [13] came up, providing a mechanism for each terminal autoconfigure a link-local IPv4 address. Other solutions are provided in [8].

In spite of these multiple IP autoconfiguration mechanisms, there is a lack of a solution which integrates them and enables the dynamic, automatic, and efficient selection of the proper mechanism according to the network context. Furthermore, in future communication environments the users will own small networks, whose terminals will have heterogeneous interfaces and be mobile and multihomed, thus increasing the dynamics and the alternatives to get connectivity. This paper surveys existing autoconfiguration frameworks, demonstrates their shortcomings, and provides the directions to solve the problem using a new framework that is able to adapt to emerging network contexts.

The remainder of this paper is organized as follows. Section II describes the autoconfiguration mechanisms defined in IP networks. Sections III presents the problem and discusses the major shortcomings found in the state of the art with respect to autoconfiguration in IP networks. Section IV presents a new framework proposed to overcome those shortcomings. Section V presents the related work and, finally, Section VI draws the conclusions.

DIN'05, September 2, 2005, Cologne, Germany.

2. AUTOCONFIGURATION FRAMEWORKS IN IP NETWORKS

We present the autoconfiguration mechanisms defined for IP networks and divide them in two categories: stateless and stateful autoconfiguration.

2.1 Stateless Autoconfiguration

Dynamic Configuration of IPv4 Link-Local Addresses. The Dynamic Autoconfiguration of IPv4 Link-Local Addresses was deployed by the IETF Zeroconf Work Group [7], and is now available as an IETF RFC [13]. Multiple operating systems, such as Windows XP and Linux, already implement it as an alternative solution to DHCP. This solution defines how a host can automatically configure an IPv4 address within the 169.254/16 prefix being valid for communication with other devices in the same link. First, the host generates a random IP address in the 169.254/16 range. Next, it performs duplicate address detection using an Address Resolution Protocol (ARP) probe, in order to assess if the address is already in use; if a reply is received, it must consider that the address is being used by other terminal and try a new address. Finally, the host assigns the IP address to the local network interface, and link local connectivity becomes possible.

IPv6 Stateless Address Autoconfiguration. This solution, specified in [17], defines the steps carried out by a host to autoconfigure its network interfaces in IPv6, without using a centralized service. The autoconfiguration process comprises the generation of a link-local and a global address, and a Duplicate Address Detection procedure, in order to verify the uniqueness of the autoconfigured addresses on a given link. The autoconfiguration of a link-local address is performed upon network interface activation, and after combining the well-known prefix FE80::0/10 with the interface identifier (ID), based on the MAC address; configuration of a global address is accomplished by combining the prefix announced by a local router, using the Router Advertisement messages defined in [19], with the interface ID. In addition, Router Advertisements contain two flags, M and O, informing the host whether DHCPv6 server should be contacted to acquire addresses and/or to obtain optional information.

Stateless DHCP for IPv6. The stateless DHCPv6 service [12] is intended to be used by nodes that have already configured an IPv6 address, through the IPv6 stateless autoconfiguration mechanism or manually, but need to acquire optional information, such as the addresses of DNS or SIP servers. This solution is a lightweight version of the stateful DHCPv6, specifying a subset of the protocol messages and avoiding state information maintenance for each individual client. It is well suited to be deployed in networking scenarios where IPv6 autoconfiguration is based on the stateless approach. To obtain the optional information, a client sends an *Information-Request* towards a well-known multicast address, and receives a *Reply* from the server containing the required information.

2.2 Stateful Autoconfiguration

DHCP. DHCP [10] provides a framework for passing configuration information to hosts, using a client/server model. It is based on the exchange of four signalling messages. The client broadcasts a *DHCPDISCOVER* message in order to discover available servers; it may receive one or more *DHCPOFFER* messages, and after selecting one of the servers, it broadcasts a *DHCPREQUEST* containing the identification of the selected server. Finally, the server will reply with a *DHCPACK* message containing the assigned address and optional information. With the advent of IPv6 a new DHCP version (DHCPv6) [11] came up, considering a different operation model: 1) a well-known multicast address is used by clients to address all the servers in the link, instead of broadcasts; 2) unlike DHCPv4, which is used to perform the whole host configuration, DHCPv6 can be used to just complement the stateless mechanism; 3) the messages defined for DHCPv6 are different in name and format. In real implementations, DHCPv4 and DHCPv6 are used independently for dual-stack hosts. Possible solutions to integrate the two frameworks are specified in [18].

PPP/IPCP. PPP is used as the standard transport framework for multiple network layer protocols over serial links. Typically it is used by a host connected to an ISP. In conjunction with other two components – a method for encapsulating multi-protocol datagrams, and a Link Control Protocol (LCP) for establishing the data-link connection – PPP defines a family of NCPs enabling autoconfiguration of different network layer protocols. The IPCP [5] is defined to configure IP over PPP, namely to autoconfigure IPv4 addresses; extensions allowing configuration of optional information are specified in [15]. The protocol is based on the exchange of two messages, *Configuration-Request* and *Configuration-Ack*, and it may involve more than one negotiation round if a peer does not accept the configurations requested by the other. After IPv6 came up, the IPv6CP was defined [3]. IPv6CP specifies the same two configuration messages, and provides the means for the negotiation of an interface ID used to configure the link-local address at the local end of the link; autoconfiguration of a global address and optional information can be performed by using the stateless or stateful mechanisms above-mentioned.

Packet Data Protocol (PDP) Context. Cellular networks can interwork with IP networks through a node called Gateway GPRS Support Node (GGSN). GGSN is responsible for delivering the required configuration parameters to a Mobile Station (MS) upon PDP context Activation [1]; before this, the MS shall perform a GPRS Attach [1] to a so-called Serving GPRS Support Node (SGSN) placed between the MS and GGSN; the messages *Attach Request*, *Attach Accept*, and *Attach Complete* are exchanged. The PDP context Activation involves the three entities; the MS interacts with the SGSN which, in turn, interacts with the GGSN. The MS sends the *Activate PDP Context Request*, indicating the PDP type (e.g., IPv4, IPv6) and requesting an address and optional information; the SGSN contacts the GGSN in order to activate the PDP context, and the MS receives the *Activate PDP Context Accept*. The information carried in this message depends on the PDP type. In IPv4, it contains the assigned address and optional information, whereas in IPv6 it provides an interface ID used by the MS to configure a link-local address; the stateless or stateful mechanisms are used to configure the global address and optional information.

3. PROBLEM DESCRIPTION

In the previous section, the state of the art on network autoconfiguration was presented. We described the stateless and stateful solutions associated to both IP versions, and showed that each mechanism has its own applicability domain. However, the mechanisms available are not enough to deal with the very dynamic and heterogeneous scenarios envisioned for future

networks, as the selection of the proper autoconfiguration framework usually requires user intervention. A new solution addressing the new communication paradigms needs to be created.

3.1 Autoconfiguration of Terminals

Considering that IPv4 and IPv6 may coexist for a long time in the Internet and that future communication scenarios will be very dynamic, the heterogeneity associated with two IP versions, the availability of multiple autoconfiguration mechanisms and its simultaneous use, implies inefficiencies and demands user intervention. A prominent future communication scenario is, for instance, a user's Personal Digital Assistant (PDA) (dis)connecting to multiple devices (e.g., Mobile Phone, Laptop, other PDAs) and networks (e.g., home, car, train) during a day. In this scenario, let us consider that the PDA is dual-stack. Since specific autoconfiguration solutions are deployed for each IP version, separate mechanisms for each stack and per-interface are executed, even when a single IP version is available. Focusing on a specific interface, and assuming that the PDA is connecting to a device (e.g., user's Mobile Phone) which only supports IPv6, the following process may occur. For IPv4, a *DHCPDISCOVER* is sent; after a timeout, upon no reception of a *DHCPOFFER*, the dynamic IPv4 link-local framework is run, and an ARP probe is broadcast; no ARP reply is received and the configuration is made. For IPv6, autoconfiguration of a link-local address is first accomplished; a *Neighbour Solicitation* [19] is sent to detect whether the current address is already used, and no *Neighbour Advertisement* is received. Next, a *Router Solicitation* is sent to trigger a *Router Advertisement*, which is never received. After a timeout, a *DHCP Solicit* [11] is sent. Again, no reply appears and autoconfiguration is terminated. From this analysis, we conclude that 4 messages were sent unnecessarily, as just the IPv6 link-local configuration is used. Hence, if multiple interfaces and the dynamics associated with the scenario are considered, it can be concluded that the overhead may be significant. For instance, if the terminal has four interfaces, each (dis)connecting to five different links along a period of time, an overhead of 4*5*4 = 80 messages for a single terminal will be achieved.

3.2 Autoconfiguration of Networks

New problems come up for the autoconfiguration of networks. In order to demonstrate them we use the scenario shown in Figure 1.

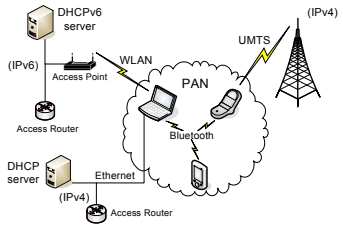

Figure 1. Example Scenario.

Scenario Description. Figure 1 shows a PAN composed of three devices: Laptop, Mobile Phone (MP) and PDA. All devices are dual-stack and support the autoconfiguration mechanisms implemented by each access network. Communication within the PAN uses Bluetooth, and the connection to the Internet is performed using UMTS, WLAN, and Ethernet. The cellular network supports IPv4 and delivers network connectivity parameters to the MP upon PDP context Activation. The WLAN provides IPv6 connectivity and supports stateless IPv6 autoconfiguration along with DHCPv6 to acquire optional parameters. The Ethernet access provides IPv4 connectivity and autoconfiguration through DHCP.

Phase I – Creation of the PAN. The PAN is assumed to be isolated from the outside world. Dynamic configuration of IPv4 link-local addresses is carried out, in order to enable services such as file sharing; multicast DNS [14], for instance, may be used to perform local name resolution. The IP version could be pre-configured by the user or negotiated automatically between the PAN's devices using a mechanism to be defined. The Bluetooth PAN profile [4] is employed to create the PAN and, in particular, the Bluetooth Group Ad-hoc Network scenario is assumed; one of the terminals implements the Group Ad-hoc Network (GN) service and the others connect to it as PAN Users (PANUs).

Phase II – Connecting over UMTS. The PAN gets global connectivity over UMTS. This requires some reconfigurations. The Bluetooth PAN profile can be used, but a different scenario has to be selected. The Bluetooth Network Access Point (NAP) scenario is selected [4]; the MP supports the NAP service and the other terminals connect to it as PANUs. Both the Laptop and the PDA have to be reconfigured in order to connect to the device offering the NAP service. At the network layer further configurations are needed. From the cellular operator perspective, only the MP is known, and the requests from other terminals to autoconfigure connectivity parameters are not accepted. On the other hand, just a single IPv4 address is assigned to the MP. Then, the MP establishes a PDP context with the cellular network and acquires an address and the optional information (e.g., DNS server address). In order to grant global connectivity to the PAN, the MP is configured as a router and run Network Address Translation (NAT) [9]; the IPv4 link-local addresses, already configured, can still be used within the PAN. Furthermore, a new lighweight mechanism allowing the distribution of optional parameters, such as DNS server address, and enabling the autoconfiguration of the local routing tables of the Laptop and PDA, should be implemented.

Phase III – Connecting over WLAN. The PAN is a moving network, and as it moves new access networks offering better services (e.g., broader bandwidth) may be found and preferred. This happens when the PAN detects the WLAN access and selects it over the UMTS connection; reconfigurations are required again, since the IP version and autoconfiguration framework supported by the new access network is different. The Laptop autoconfigures an IPv6 address for the WLAN interface based on the Router Advertisement messages sent by the local Access Router (AR), and collects optional parameters from the local DHCPv6 server. Concerning autoconfiguration of the PAN, the Laptop acts now as a Bluetooth NAP and as an IPv6 router between the PAN and the WLAN. Additionally, it relays the Router Advertisements received from the local AR to the PAN, so that the other devices can autoconfigure a global address. For autoconfiguration of optional information, a stateless DHCPv6 server [12] could be implemented by the Laptop; the relayed Router Advertisements would inform the other devices about the presence of that server.

Phase IV – Connecting over Ethernet. The PAN finds out that Ethernet access is available. Again, reconfigurations should occur, as IPv4 is again supported. The Laptop uses DHCP for autoconfiguring its Ethernet interface. Furthermore, the Laptop,

previously acting as a Bluetooth NAP, needs now to operate between the Ethernet connection and the PAN, and be configured as an IPv4 router supporting NAT to offer global access to the PAN. Inside the PAN, the IPv4 configurations carried out in Phase II need to be applied, since the global connectivity is now obtained through NAT; a similar approach to the mentioned in Phase II may be employed to deliver optional parameters.

After analysing this scenario, we may conclude that the multiple reconfigurations needed cannot be performed automatically using current solutions. Furthermore, for non-expert users the deployment of the scenario is almost impossible; huge and specific configuration efforts are required. On the other hand, since the scenario is very dynamic, manual configurations are impractical; they would become invalid from time to time, leading to frequent reconfigurations. Thereby, a new framework capable of detecting changes in IP connectivity and associated autoconfiguration mechanisms, and capable of configuring terminals automatically and dynamically is required. Concerning, for instance, the creation of the PAN in Phase I, it will be required that terminals agree upon the IP version and autoconfiguration framework to be used. Additionally, in the subsequent phases, configuration of the terminals as routers and dynamic deployment of network services (e.g., DHCP, NAT) requires knowledge of the IP version as well. Existing autoconfiguration frameworks provide configuration of IP addresses and optional information, such as DNS, SIP, or proxy servers. However, they target specific scenarios and are tied to a particular IP version. A mechanism enabling the selection of the proper IP version and autoconfiguration framework, according to the network context is missing.

4. NEW FRAMEWORK

The new framework we propose to solve the above mentioned problem is named Meta-autoconfiguration Framework (MAF). It is IP version independent and capable of selecting dynamically and automatically the proper IP version and autoconfiguration mechanism (step 1 in Figure 2). MAF exists at the control plane and runs directly over Layer 2 protocols, as it must be acting before IP connectivity is available. It does not provide any autoconfiguration mechanism by itself; rather, interaction with available frameworks should occur, as shown in Figure 2, step 2, for the selection of the proper mechanism. Furthermore, automatic configuration of network services and nodes is performed, according to the network context (e.g., IP version, available autoconfiguration framework, available network services). MAF comprises three messages: *ELECTION*, *NEGOTIATE* and *INFO*. The first is used to elect a master device when multiple terminals are forming a network; the second is applied in the negotiation of the IP version and autoconfiguration mechanism to be used between connecting peers; *INFO* is used for notification purposes, e.g., when a new way to get global connectivity exists and changes on IP version and autoconfiguration mechanism are required.

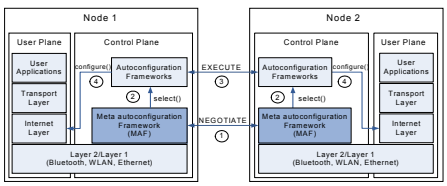

Figure 2. Autoconfiguration using MAF.

We describe below the MAF operation for the first two phases of the scenario drawn in Figure 1:

1. The Laptop is pre-configured to provide the GN service; the other devices are pre-configured as PANUs, and search for that service using the Bluetooth Service Discovery Protocol (SDP) [4]. Pre-configurations enable connectivity between the GN and each PANU, and in order to provide connectivity between the PANUs as well, the MAF running in the Laptop configures this node as a bridge between those two connections, providing connectivity inside the PAN.
2. A MAF master device is elected using an election algorithm and the *ELECTION* message; the Laptop, as the most capable device in the PAN, is elected. Then, each MAF slave (PDA and MP) sends a *NEGOTIATE* message to the master proposing an IP version and autoconfiguration framework. The MAF master, in turn, collects the slaves' proposals and, after considering its own proposal, disseminates the agreement. Let us assume that, in this case, the agreement results in the use of IPv4 and its associated link-local autoconfiguration mechanism, because, for instance, Multicast DNS is only available for IPv4.
3. The MP acquires global connectivity through UMTS upon GPRS Attach and PDP context Activation, triggered by the user, for instance. MAF in the MP becomes aware of that, and it sends an *INFO* message to let the other devices know that global connectivity is available through it. This message contains information about the IP version and autoconfiguration mechanism to be used. According to our scenario, IPv4 is selected and no new IP autoconfiguration is required.
4. MAF configures the MP as a Bluetooth NAP, overriding the pre-configurations applied in Phase I. Besides, MAF configures the MP as a router and starts the NAT. Upon receiving the *INFO* message, MAFs in the other devices configure the local routing tables accordingly, considering that the MP is the default gateway. There is no need to do IP address reconfiguration. On the other hand, MAFs running in the Laptop and PDA reconfigure their Bluetooth protocol stacks, and become PANUs searching for NAP service. Additionally, the bridge previously configured in the Laptop shall be deactivated.

5. RELATED WORK

Existing work related to our problem and proposal is presented below.

TurfNet. TurfNet [16] represents a new network paradigm enabling the integrated operation of networks with different address realms (e.g., IPv4, IPv6), thanks to the so-called TurfNet gateways. In this context, a TurfNet is defined as an autonomous network domain, encompassing its own address space and associated control plane functions (e.g., routing, address allocation, name-to-address resolution) integrated into the TurfControl. This solution uses the emerging concept of dynamic network composition being investigated in the European project *WWI Ambient Networks* [6], which enables the merging of multiple TurfNets into a single one, or the simple interoperation between TurfNets. In principle, TurfNet could be used to implement the scenario given in Section 3.2. Each network in the scenario could be considered as a TurfNet comprising a TurfNet gateway to enable interoperation with other TurfNets.

Nevertheless, the node implementing the TurfNet gateway inside the PAN would have to change according to the different TurfNets the PAN would compose with along the time; currently the TurfNet framework does not support this.

ANs and Composition. Considering a scenario similar to ours, in [2] a solution enabling automatic and dynamic creation of a PAN and selection of the proper access network is provided. The solution is based on the Ambient Network (AN) and Network Composition concepts being studied in the European project *WWI Ambient Networks* [6]. Nonetheless, this solution just concerns about IPv6 and does not address the heterogeneity coming up with coexistence of IPv4 and IPv6 in the current Internet. Therefore, it just solves part of the problem presented in Section 3.

MANETs autoconfiguration mechanisms. The IETF MANET AUTOCONF working group addresses autoconfiguration in Mobile Ad-hoc Networks (MANETs). This group is searching for a standard autoconfiguration solution for MANETs, where each terminal is always acting as a router and running a routing protocol to maintain IP connectivity with the other terminals in the same MANET. Some autoconfiguration protocols for MANETs are already defined [8], and certainly the IETF solution will be influenced by them. In contrast with our solution, MANETs require that each terminal acts as a router; in our approach terminals may assume different roles along the time according to the network context, and their interaction is based on offered and required services. Furthermore, current MANETs autoconfiguration and routing protocols suffer from a problem mentioned before: separate solutions are provided concerning each IP version, and there is no mechanism enabling the selection of a proper framework considering the network context.

6. FURTHER WORK

This paper focused on the autoconfiguration of IP nodes for next generation networks. Other issues that could be considered include mobility, multihoming, and security. These problems are partially solved by existing frameworks, such as Mobile IP, which can be used in conjunction with MAF. Regarding autoconfiguration, work on the link layer still remains to be done. For instance, authentication in wireless LANs raises new autoconfiguration problems that need to be solved before the network layer autoconfiguration can be accomplished. On the other hand, MAF needs to be fully specified and validated. As defined now, it does not guarantee session continuity, and the users will notice service interruption everytime the access IP network changes. The development of a prototype enabling the proof of concept and the evaluation of the Meta-Autoconfiguration Framework (MAF) will be the next step of our work.

7. CONCLUSIONS

In current IP networks two versions of the same protocol coexist. On the other hand, multiple autoconfiguration frameworks associated to each version are defined, leading to heterogeneous environments. In this scenario, user intervention may be required to perform tasks that should be done automatically. Furthermore, existing autoconfiguration frameworks are not sufficient by themselves to enable autoconfiguration in future dynamic communication scenarios, where users will own small networks, instead of single terminals. This paper analysed the shortcomings associated to the state of art of autoconfiguration mechanisms. We concluded that there is a lack of an integrating solution capable of unifying the autoconfiguration mechanisms available and autoconfiguring nodes and start network services according to network contexts. In order to improve the current scenario, a new framework was discussed, which enables the integration of current autoconfiguration mechanisms and addresses the new communication paradigms illustrated by the example scenario provided.

8. ACKNOWLEDGEMENT

The first author would like to thank the support from FCT under the fellowship SFRH/BD/19429/2004/.

9. REFERENCES

[1] 3GPP TS 23.060, *General Packet Radio Service (GPRS) Service description Stage 2*, v6.8.0, March 2005.

[2] C. Kappler, N. Akhtar, R. Campos et al., *Network Composition using Existing and New Technologies*, in Proceedings of the14th IST Mobile & Wireless Communications Summit 2005, Dresden, Germany, June 19-23, 2005.

[3] D. Haskin, et al., *IP Version 6 over PPP*, RFC 2472, December 1998.

[4] D. Sönnerstam, et al., *Specification of the Bluetooth System* (version 1.2), November 2003.

[5] G. McGregor, *The PPP Internet Protocol Control Protocol (IPCP)*, RFC 1332, May 1992.

[6] http://www.ambient-networks.org/

[7] http://www.zeroconf.org

[8] K. Weniger, et al., *Address Autoconfiguration in Mobile Ad Hoc Networks: Current Approaches and Future Directions*, IEEE Network, vol. 18, (August 2004), 6-11.

[9] P. Srisuresh, et al., *IP Network Address Translator (NAT) Terminology and Considerations*, RFC 2663, August 1999.

[10] R. Droms, *Dynamic Host Configuration Protocol*, RFC 2131, 1997.

[11] R. Droms, et al., *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*, RFC 3315, July 2003.

[12] R. Droms, *Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6*, RFC 3736, April 2004.

[13] S. Cheshire, et al., *Dynamic Configuration of IPv4 Link-Local Addresses*, RFC 3927, May 2005.

[14] S. Cheshire, et al., *Multicast DNS*, Internet Draft, draft-cheshire-dnsext-multicastdns-05 (work in progress), July 2005.

[15] S. Cobb, *PPP Internet Protocol Control Protocol Extensions for Name Server Addresses*, RFC 1877, December 1995.

[16] S. Schmid, et al., *TurfNet: An Architecture for Dynamically Composable Networks*, in Proceedings of the First IFIP TC6 WG6.6 International Workshop on Autonomic Communication, Berlin, Germany, October 2004.

[17] S. Thomson, et al., *IPv6 Stateless Address Autoconfiguration*, RFC 2462, December 1998.

[18] T. Chown, et al., *DHCP: IPv4 and IPv6 Dual-Stack Issues*, Internet Draft, draft-ietf-dhc-dual-stack-03 (work in progress), July 2005.

[19] T. Narten, et al., *Neighbor Discovery for IP Version 6 (IPv6)*, RFC 2461, December 1998.

[20] W. Simpson, Ed., *The Point-to-Point Protocol (PPP)*, RFC 1661, July 1994