



Pós-Graduação em Ciência da Computação

“Inter Domain Negotiation”

Por

Reinaldo Cézar de Moraes Gomes

Tese de Doutorado



Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE, SETEMBRO/2010



UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE INFORMÁTICA

PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

REINALDO CÉZAR DE MORAIS GOMES

“INTER-DOMAIN NEGOTIATION”

*ESTE TRABALHO FOI APRESENTADO À PÓS-GRADUAÇÃO EM
CIÊNCIA DA COMPUTAÇÃO DO CENTRO DE INFORMÁTICA DA
UNIVERSIDADE FEDERAL DE PERNAMBUCO COMO REQUISITO
PARCIAL PARA OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIA
DA COMPUTAÇÃO.*

ORIENTADOR: Djamel Sadok
CO-ORIENTADORA: Judith Kelner

RECIFE, SETEMBRO/2010

Gomes, Reinaldo César de Moraes
Inter domain negotiation / Reinaldo César de Moraes
Gomes. - Recife: O Autor, 2010.
xiii, 190 folhas : il., fig., tab.

Tese (doutorado) Universidade Federal de Pernambuco.
Cln. Ciência da Computação, 2010.

Inclui bibliografia.

1. Redes de computadores. 2. Auto configuração. 3.
Internet. 4. Políticas. I. Título.

004.6

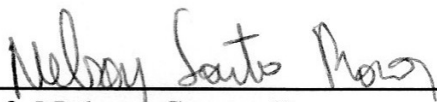
CDD (22. ed.)

MEI2010 – 0137

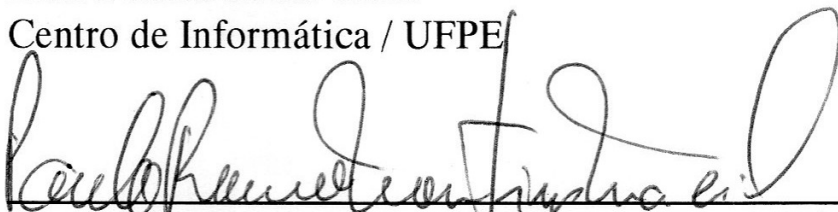
Tese de Doutorado apresentada por **Reinaldo César de Moraes Gomes** à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título "**Inter-Domain Negotiation**", orientada pelo **Prof. Djamel Fawzi Hadj Sadok** e aprovada pela Banca Examinadora formada pelos professores:



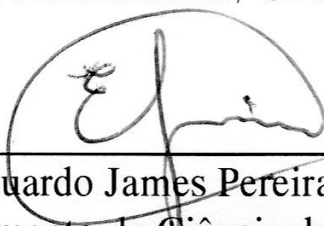
Prof. Paulo Roberto Freire Cunha
Centro de Informática / UFPE



Prof. Nelson Souto Rosa
Centro de Informática / UFPE



Prof. Paulo Romero Martins Maciel
Centro de Informática / UFPE

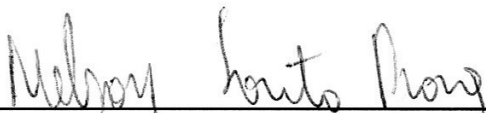


Prof. Eduardo James Pereira Souto
Departamento de Ciência da Computação / UFAM



Prof. Stênio Flávio de Lacerda Fernandes
Departamento de Ensino / IFAL

Visto e permitida a impressão.
Recife, 27 de maio de 2010.



Prof. NELSON SOUTO ROSA

Coordenador da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

Summary

Acronyms and Abbreviations	x
Abstract	xii
Resumo	xiii
1 Introduction	1
1.1 Context and Motivation.....	1
1.2 Problem Definition and Proposed Solution.....	3
1.3 Thesis Organization	6
2 Networking Autoconfiguration Technologies	7
2.1 Autoconfiguration Basis	8
2.2 Addressing Configuration	9
2.2.1 Requirements for Addressing Autoconfiguration	10
2.2.2 Stateless Solutions.....	11
2.2.2.1 DAD-based Mechanism	13
2.2.3 Stateful Solutions	17
2.2.3.1 Prophet Allocation	18
2.2.3.2 IP Autoconfiguration Suite	19
2.2.4 Hybrid Solutions	21
2.2.4.1 MANETconf.....	21
2.2.4.2 PACMAN	23
2.3 Networks Autoconfiguration.....	26
2.3.1 Requirements for Network Autoconfiguration	26
2.3.2 Autonomic Computing.....	28
2.3.3 Ambient Networks	30
2.3.4 4WARD.....	32
2.4 Dynamic Service Negotiation	35
2.4.1 Requirements for Service Negotiation Protocols.....	36
2.4.2 Service Negotiation Protocols	38
2.4.2.1 RNAP	38
2.4.2.2 COPS-SLS.....	40
2.4.2.3 Dynamic Service Negotiation Protocol.....	41
2.5 Summary	42
3 Basic Negotiation Mechanisms	44
3.1 Addressing Negotiation	45
3.1.1 Local Self-Addressing	46
3.1.2 Local Address Negotiation.....	50
3.1.2.1 Protocol Elements and Operations	54
3.1.2.2 New Node's Temporary Configuration.....	55
3.1.2.3 Negotiation Procedure.....	56
3.1.2.4 Server Element.....	58

3.1.2.5	Client Element.....	61
3.1.2.6	Backup Nodes	62
3.1.3	Address Pools Distribution.....	64
3.1.3.1	Mechanism Overview	66
3.1.3.2	Pools Allocation Mechanisms.....	68
3.2	Intra-Domain Configuration Negotiation.....	72
3.2.1	General Description	73
3.2.2	Phases	74
3.2.3	Elements.....	75
3.2.3.1	Elements Interaction.....	77
3.2.4	Operation	79
3.2.4.1	Dissemination of operating environment information.....	80
3.2.4.2	Domain Border Detection	82
3.2.4.3	Decision Process.....	82
3.2.5	Operation Example	86
3.3	Summary	88
4	Inter-Domain Policies Negotiation	89
4.1	Why dynamic negotiation might be allowed in Inter-Domains Communication?	90
4.2	General Description of the Negotiation Mechanism	91
4.2.1	Discovery	94
4.2.2	Announcement.....	95
4.2.3	Negotiation	96
4.2.4	Enforcement.....	98
4.2.5	Monitoring	99
4.3	Policies Negotiation Protocol.....	100
4.3.1	Negotiation Elements and Terminology.....	100
4.3.2	Negotiation Process	103
4.3.3	Protocol Messages	105
4.3.3.1	Data Representation Structure.....	106
4.3.3.2	Messages Description.....	107
4.3.4	Negotiation Process	114
4.3.4.1	Discovery	116
4.3.4.2	Announcement.....	117
4.3.4.3	Negotiation	120
4.3.4.4	Enforcement.....	125
4.3.4.5	Monitoring	127
4.3.5	Elements Communication and Cooperation.....	130
4.4	Summary	131
5	Experiments and Results Analysis	132
5.1	Validation of the proposed solutions	132
5.2	Experiments Environment.....	133
5.2.1	Simulation Environment	134
5.2.2	Simulation Topologies	134
5.3	Local Address Negotiation.....	135
5.3.1	Implemented Protocols	135

5.3.2	Simulation Scenarios	136
5.3.3	Evaluation Metrics.....	137
5.3.4	Network Scalability.....	138
5.3.5	Protocols' Traffic Pattern.....	142
5.3.6	Addressing Space	145
5.4	Address Pools Allocation	149
5.4.1	Implemented Protocols	149
5.4.2	Simulation Scenarios	149
5.4.3	Evaluation Metrics.....	150
5.4.4	Pools Distribution	150
5.4.5	Routing Table Aggregation.....	153
5.4.6	Reservation Degree	155
5.5	Intra-Domain Configuration Negotiation.....	157
5.5.1	Mechanism Implementation	157
5.5.2	Simulation Scenarios	157
5.5.3	Evaluation Metrics.....	158
5.5.4	Traffic Overhead.....	159
5.5.5	Routing and Decision Traffic Overhead.....	162
5.5.6	Configuration Delay	164
5.5.7	Application throughput	165
5.6	Inter-Domain Negotiation	166
5.6.1	Mechanism Implementation	167
5.6.2	Simulation Scenarios	167
5.6.3	Negotiation Delay.....	168
5.7	Summary	170
6	Conclusions	172
6.1	Summary of Contributions.....	172
6.1.1	Addressing Negotiation	172
6.1.2	Intra-Domain Configuration Negotiation	174
6.1.3	Inter-Domain Policies Negotiation.....	174
6.1.4	Evaluation of proposed mechanisms	175
6.2	Future Work	176
6.2.1	Addressing Negotiation	176
6.2.2	Intra-Domain Configuration Negotiation	177
6.2.3	Inter-Domain Policies Negotiation.....	178
	References	179

List of Figures

Figure 2.1 – Reduced Strong DAD algorithms.	14
Figure 2.2 – Weak DAD scenario.....	16
Figure 2.3 – Prophet Allocation algorithm scheme.	18
Figure 2.4 – IPAS architecture and components.....	20
Figure 2.5 – MANETconf basic algorithms.....	22
Figure 2.6 – PACMAN modular architecture.....	24
Figure 2.7 – Autonomic Computing Reference Architecture [AUT10].....	29
Figure 2.8 – 4WARD research areas[4WA09]	32
Figure 2.9 – 4WARD networks organization [4WA09]	33
Figure 2.10 – Traditional and In-Network Management in 4WARD perspective [4WA09]	34
Figure 2.11 – 4WARD network of information [4WA09]	35
Figure 2.12 – RNAP Centralized Architecture [WAN99].....	39
Figure 2.13 – RNAP Distributed Architecture [WAN99].....	39
Figure 2.14 – PBM Architecture	40
Figure 2.15 – COPS-SLS communication model [DUR00]	41
Figure 3.1 – Local Self-Addressing Negotiation.....	46
Figure 3.2 – Phases of Local Self-Addressing Negotiation.....	47
Figure 3.3 – Example of conflict not detected by controlled DAD.....	49
Figure 3.4 – Example of Local Address Negotiation when a pool is obtained	50
Figure 3.5 – Tree-format servers’ hierarchy	52
Figure 3.6 – Simple scenario with server and client nodes	53
Figure 3.7 – SooA node’s status.....	54
Figure 3.8 – New Node’s algorithm.	56
Figure 3.9 – Server’s algorithm.....	59
Figure 3.10 – Client’s algorithm.	61
Figure 3.11 – Situations for triggering and/or creating a backup.	64
Figure 3.12 – Example of disjoint Pools Allocation	65
Figure 3.13 – Example of adjacent Pools Allocation.....	66
Figure 3.14 – Set of Pools with priority and preserving allocation of adjacent pools for the same device	69
Figure 3.15 – Example of allocation sequence using Spread Allocation with binary division discipline	70
Figure 3.16 – Example of x_i extra pools allocation	72
Figure 3.17 – Intra-Domain Phases.....	74
Figure 3.18 – Coordinator Selection.....	78
Figure 3.19 – Local Decision aggregation and Final Decision execution	79
Figure 3.20 – An example of Intra-Domain Configuration operation.....	80
Figure 3.21 – Possible Networks Statuses	81
Figure 3.22 –Local Decision Overview.....	84
Figure 3.23 – Example of Local Decision Dissemination	86
Figure 3.24 – Home area network using Intra-Domain Configuration to select a routing protocol87	
Figure 4.1 – Phases of Policies Negotiation Mechanism	92
Figure 4.2 – Steps of Discovery Phase.....	94
Figure 4.3 – Steps of Announcement Phase in the Origin Network.....	95
Figure 4.4 – Steps of Announcement Phase in the Destination Network	96
Figure 4.5 – Steps of Negotiation Phase in the Origin Network	97
Figure 4.6 – Steps of Negotiation Phase in the Destination Network.....	98
Figure 4.7 – Steps of Enforcement Phase	98

Figure 4.8 – Steps of Monitoring Phase.....	99
Figure 4.9 – Overview of the Negotiation Process.....	104
Figure 4.10 – Example of negotiators and their possible policies	104
Figure 4.11 – Messages Header Information.....	107
Figure 4.12 – NEGOTIATION_START Message Information	108
Figure 4.13 – NEGOTIATION_ACCEPT Message Information.....	108
Figure 4.14 – NEGOTIATION_PARTIAL_ACCEPT Message Information.....	109
Figure 4.15 – NEGOTIATION_REJECT Message Information.....	109
Figure 4.16 – NEGOTIATION_REQUEST Message Information	110
Figure 4.17 – NEGOTIATION_RESPONSE Message Information	111
Figure 4.18 – NEGOTIATION_RELEASE Message Information	112
Figure 4.19 – RESOURCE_STATE_REQUEST Message Information	112
Figure 4.20 – RESOURCE_STATE_RESPONSE Message Information	113
Figure 4.21 – RESOURCE_QUOTATION Message Information	113
Figure 4.22 – RESOURCE_PRICE Message Information.....	113
Figure 4.23 – TN_LIST_REQUEST Message Information	114
Figure 4.24 – TN_LIST_RESPONSE Message Information.....	114
Figure 4.25 – Overview of the Negotiation Process.....	116
Figure 4.26 – Requesting Domain Announcement Algorithm	118
Figure 4.27 – Destination Domain Announcement Algorithm	119
Figure 4.28 – Requesting Domain Negotiation Algorithm.....	121
Figure 4.29 – Destination Domain Negotiation Algorithm.....	123
Figure 4.30 – Policies Framework Architecture	126
Figure 4.31 – Requesting Domain Monitoring Algorithm.....	128
Figure 4.32 – Destination Domain Monitoring Algorithm.....	129
Figure 4.33 – Negotiation information exchange modes: (a) direct contact negotiation, (b) recursive hop-by-hop negotiation and (c) direct hop-by-hop negotiation	130
Figure 5.1 – Static scenario: nodes positioning in a grid-format and controlled topology.....	134
Figure 5.2. Scalability in static scenario: (a) sent packets; (b) received packets; (c) sent bytes; (d) received bytes.....	139
Figure 5.3. Configuration delay in Static scenario.	140
Figure 5.4. Scalability in dynamic scenario:(a)sent packets;(b) received packets;(c) sent bytes;(d) received bytes.....	141
Figure 5.5. Configuration delay in Dynamic scenario.	142
Figure 5.6. Traffic pattern of received bytes in Static scenario.	144
Figure 5.7. Addressing space in Static scenario: (a) sent bytes; (b) received bytes; (c) configuration delay; and (d) configuration delay per node with addressing space of 100%.....	146
Figure 5.8. Addressing space in Dynamic scenario: (a) sent bytes; (b) received bytes; (c) configuration delay; and (d) configuration delay with addressing space of 100%.....	147
Figure 5.9 – Routing tables' size: (a) 25 Pools; (b) 50 Pools; (c) 100 Pools; and (d) 200 Pools with requesting networks varying from 5 to 25.....	151
Figure 5.10 – Routing tables' size for 5 requesting networks with available pools varying from 25 to 200.....	152
Figure 5.11 – Routing tables' aggregation: (a) 5 networks; (b) 10 networks; (c) 15 networks; (d) 20 networks; and (e) 25 networks with available pools varying from 25 to 200.....	154
Figure 5.12 – Routing tables' size: (a) 25 Pools; (b) 50 Pools; (c) 100 Pools; and (d) 200 Pools with requesting networks varying from 5 to 25.....	156
Figure 5.13 – Traffic Overhead for decision delay in 30 seconds and varying the number of DA Nodes: (a) packets and (b) bytes.....	159
Figure 5.14 – Detailed traffic overhead: (a) packets for 1 DA; (b) bytes for 1 DA Node; (c) packets for 10 DA Nodes; and (d) bytes for 10 DA Nodes.	160

Figure 5.15 – Traffic overhead from Routing and Decision protocols with a decision delay of 30 seconds and varying the number of DA Nodes: (a) Sent bytes from AODV and XDNP; (b) Received bytes from AODV and XDNP; (c) Sent bytes from DSDV and XDNP; and (d) Received bytes from DSDV and XDNP.....	162
Figure 5.16 – Traffic overhead from Routing and Decision protocols with one DA Node and varying the decision delay: (a) Sent bytes from AODV and XDNP; (b) Received bytes from AODV and XDNP; (c) Sent bytes from DSDV and XDNP; and (d) Received bytes from DSDV and XDNP.....	163
Figure 5.17 – Average Configuration Delay varying the Decision Delay: (a) AODV; and (b) DSDV.	165
Figure 5.18. Application throughput for AODV and DSDV networks varying the number of DA Nodes.....	166
Figure 5.19. Negotiation delay using Direct hop-by-hop (only Binding and Binding and one requirement) and Direct Contact Negotiation modes and varying the number of link between the domains.....	168
Figure 5.20. Negotiation delay using Direct Contact negotiation with only one link between the domains and varying the number of requirements requested.	169

List of Figures

Table 4.1. Policies Negotiation Protocol Messages from Originator Domain.	105
Table 4.2. Policies Negotiation Protocol Messages from Destination Domain.	106
Table 5.1. Summary of simulation parameter for Static and Dynamic Scenarios.....	137
Table 5.2. Summary of parameters for Network Scalability simulations.	138
Table 5.3. Summary of parameters for Addressing Space simulations.....	146
Table 5.4. Summary of simulation parameters for pools allocation evaluations.....	149
Table 5.5. Summary of simulation parameters for Intra-Domain Negotiation evaluations.....	157
Table 5.6. Summary of simulation parameters for Inter-Domain Policies Negotiation Protocol.....	168

Acronyms and Abbreviations

ACA – Adaptive Configuration Agent

AN – Ambient Networks

ARP – Address Resolution Protocol

AS – Autonomous System

CA – Composition Agreement

COPS – Common Open Policy Service

DAD – Duplicate Address Detection

DCDP – Dynamic Configuration Distribution Protocol

DHCP – Dynamic Host Configuration Protocol

DNS – Domain Name System

DoS – Denial of Service

DRCP – Dynamic and Rapid Configuration Protocol

HIP – Host Identity Protocol

HRN – Host Resource Negotiator

IANA – Internet Assigned Numbers Authority

IPAS – IP Autoconfiguration Suite

IPv4 – IPversion 4

IPv6 – IPversion 6

MAC – Media Access Control

MANET – Mobile Ad-hoc Network

MIB – Management Information Base

ND – Neighboring Discovery

NDP – Neighbor Discovery Protocol

NetID – Network ID

NGI – Next Generation of Internet

NRN – Network Resource Negotiator

NS2 – Network Simulator version 2

PACMAN – Passive Autoconfiguration for Mobile Ad-hoc Networks

PAN – Personal Area Network

PDAD – Passive Duplicate Address Detection

PDP – Policy Decision Point

PEP – Policy Enforcement Point

PMT – Policy Management Tool

QoS – Quality of Service

RNAP – Resource Negotiation and Pricing Protocol

RSVP – Resource Reservation Protocol

SLA – Service Level Agreement

SLS – Service Level Specification

SLAAC – Stateless Address Autoconfiguration

SNMP – Network Management Protocol

SooA – Self-organization of Addresses

YAP – Yelp Announcement Protocol

Abstract

During the last years we had faced the development of several technologies that intend to help and facilitate users' interaction and improve their communication. The interoperation of all these technologies has created the need of a new network infrastructure to offer a better adaptation to new networks requirements.

In this context, networks are supposed to be dynamically created in order to facilitate user's communication allowing several automatic functionalities, such as integration of control planes with other networks and services discovery. These networks would be present in many networking contexts and a foreseen challenge of them is how different technologies including wired and wireless structures, optical networks and many more will be able to cooperate and attend potential highly dynamic and heterogeneous use scenarios.

These networks aim to interconnect different technologies and domains, offering a resultant inter-network that should appear fairly homogeneous to potential users, so, a key point to building these future dynamic networks is their interconnection and cooperation. Hence, new solutions need to be defined, tailored and validated in order to ensure that new requirements, such as policy management, network mobility and self-* management objectives, are contemplated.

To support these new requirements a set of mechanism to control the automatic discovery of networks capabilities and make their configurations are proposed, allowing networks to be created and adapted in a completely dynamic structure. This initial configuration is complemented by a Inter-Domain Policies Negotiation mechanism that is responsible by discovering and negotiating new resources to be used by the networks, allowing an entirely new communication model based on opportunistic networks establishment and at the same time creating and configuration of new Inter-Domains agreements dynamically.

Keywords: Negotiation, self-configuration, dynamic networks, policies, Inter-Domain Communication.

Resumo

Nos últimos anos diversas tecnologias foram desenvolvidas com o objetivo de facilitar a interação entre os usuários e seus dispositivos e melhorar a comunicação entre eles, necessitando da interoperabilidade entre essas tecnologias e, conseqüentemente, a necessidade de uma nova infraestrutura de rede que permita uma melhor adaptação aos novos requisitos criados por esta diversidade de tecnologias.

O modelo de comunicação entre redes também está sendo modificado, uma vez que é esperado que elas sejam criadas dinamicamente para facilitar a utilização da rede pelos usuários e permitir que diversas operações sejam realizadas automaticamente (endereçamento, descoberta de serviços, etc.). Essas redes devem estar presentes em diversos cenários de comunicação e um dos seus principais desafios é permitir que diversos tipos de tecnologias cooperem em ambientes com alto dinamismo e heterogeneidade.

Estas redes têm como objetivo interconectar diferentes tecnologias e domínios oferecendo uma comunicação que aparente ser homogêneo para os seus usuários. Para a criação dessas futuras redes dinâmicas pontos chave são a interconexão e a cooperação entre as tecnologias envolvidas, o que exige o desenvolvimento de soluções para garantir que novos requisitos sejam suportados.

Para permitir que novos requisitos sejam corretamente suportados, um conjunto de mecanismos para controlar a descoberta automática de recursos e realizar a sua configuração é proposto, permitindo que redes sejam criadas e adaptadas de maneira completamente automática. Também é proposto um mecanismo de negociação de políticas *inter-domínio* responsável por descobrir e negociar novos recursos que dever ser usados pelas redes, o que traz um novo modelo de comunicação baseado na criação oportunista de redes e ao mesmo tempo permite a criação de novos acordos de comunicação entre domínios administrativos de maneira dinâmica e sem a intervenção dos usuários ou dos administradores das redes.

Palavras-chave: Negociação, autoconfiguração, redes dinâmicas, políticas, comunicação inter-domínio.

1 Introduction

This thesis is on the subject of providing negotiation mechanisms to allow automatic and dynamic networks configuration and adaptation over multiple domains in the Internet. The deployment of automatic solutions to support self-configuration and adaptation of network and their elements is increasingly capturing the attention of the Internet community. Negotiation process remains a theoretical exercise and no prior experience has been gained. Because of it, there are no technical or commercial solutions strong enough for convincing administrators to use these mechanisms and losing control over the network.

In this context, a big challenge is to develop technologies, generic mechanisms and business models capable of stimulating administrators and domains to offer automatic negotiation and configuration of the networks' elements to facilitate their interaction.

In the following sections, the need of automatic negotiation in future networks is examined. Then, the problem this thesis aims to resolve and an overview of the proposed mechanisms are presented. Finally, the organization of the next chapters is shown.

1.1 Context and Motivation

In existing networks, information mainly retrieved off-line is used to configure different network elements and this configuration remains fairly static during the period of time when the network is in operation. This prior knowledge necessary to configure networks could consist of, and just to mention a few examples, the number of network elements in a network, the topology, capacities of links and network elements, and QoS characteristics.

In future networking, a whole new level of dynamicity and heterogeneity is expected. Following this new networking paradigm, future networks' architectures and protocols must

enable the cooperation of heterogeneous networks that might belong to different operator or technology domains. This cooperation should be transparent, established on demand, and thus support “plug-and-play” operation, i.e., no previous configuration or negotiation should be required when forming the networks.

New communication scenarios are expected to be deployed in order to use the new characteristics that might be supported by new solutions and protocols. In this context, situations where new networks would be spontaneously created according to users/administrators needs will happen, creating a new networks’ connection establishment model. Networks’ connectivity is supposed to be unpredictable and, consequently, agreements defining how the domains might exchange information among them should be dynamically created to represent topology and connectivity modification of the environment.

An example of these possible scenarios is a Personal Area Network (PAN) which is created by the devices carried by a user (i.e., laptop, cell phone, PDA, etc.). In a certain moment the user needs to access some resource which is not possible to be offered by the devices (access a web site or watch a TV program). When this happens the network might contact the network which offers the requested resource to negotiate the access and create the necessary agreement to guarantee the correct communication according to networks’ requirements.

Administrative domains should consider all these characteristics when establishing the connectivity with other domains to adapt their configuration and the resources access according to the rules that should be defined on-demand, taking into consideration all discussed characteristics of networks connectivity.

In this sense, many concepts must be created or redefined in terms of dynamic networks due to the foreseeable heterogeneous nature of these networks; the same network element might be installed and used, at a certain point in time in a so called ad-hoc network environment and then later on moved and re-installed in a more fixed infrastructure type of network environment. This will demand a completely different communication model and control structure, needing an entire adaptation of the device’s configuration. For this type of operating environment a multi-protocol approach, where a different routing protocol could be used in different operating environments, seems plausible instead of a “one-size-fits-all” single protocol approach.

But, by allowing multiple protocols to simultaneously operate in the same network environment, an issue naturally arises: how to choose between the supported protocols so that the information can be routed as efficiently as possible inside and between network domains? [BAM94] This multi-protocol approach needs a more elaborated mechanism to correctly evaluate the possible protocols and select the best one depending of the characteristics of the environment [CLA96] [EDE96].

This new communication model also needs changes in the Inter-Domain connection establishment allowing it to be dynamically negotiated and configured [AMB09] [D3-G1]. In the current Internet this negotiation is performed in a static manner providing simple connectivity to the best effort service [GAO01]. Renegotiations are not usual in this scenario [DEW00] [FAN99] and because of it the domains just have to maintain bilateral agreements in order to exchange traffic and routing information with each other, with no special control to adapting to networks changes.

However, with the introduction of new users' demands (i.e. dynamic devices configuration [AUT09], content location/distribution and devices/services configuration) [AMB09], this situation tends to become more critical due to more frequent changes on communication conditions and the urge to finding new routes for using new resources that might be necessary [4WA09] [CLA09] [NIE04]. Therefore, dynamic negotiation becomes extremely necessary to guarantee that resources are correctly discovered and allocated between the involved networks.

1.2 Problem Definition and Proposed Solution

As explained in the previous section, current technologies for dynamic network's configuration are very limited. This happens because networks and their elements have no ability to become "self-aware" about their own configuration as well as their environment. This self knowledge generally sits in centralized management systems from which network elements get populated with configuration information [OLH06] [USZ03] [VER02].

Some few exceptions where dynamic configuration is supported are routing tables definition and IP addresses configuration in hosts using a host configuration protocol. But these few examples operate in well-controlled network environments, where network topology and the

distribution of certain functionality onto network elements are known prior to the operation of the network and de communication among its elements.

Today's network architectures also limit innovation in communication technologies. Absence of adequate facilities to design, optimize, and interoperate new networks currently forces a convergence to an architecture that isn't ideal for many applications and cannot support most necessary innovations, such as mobility, heterogeneity, services convergence, and dynamic environments configuration. On the other hand, in future networks, one cannot assume a network topology or any specific characteristic of technologies. And even more, network topology and functionality might rapidly change and thus, there could be needs for on-demand and real-time re-configuration of network elements due to changes in the operational environment.

This work considers that networks demand of having dynamic negotiation solutions which allow the automatic environments configuration can be solved by developing new mechanisms and protocols. These new solutions would define new configurations to be used in Intra and Inter domain communications.

To deal with environments' dynamic configuration, a set of mechanisms should be defined to allow the negotiation between network elements to verify possible configurations and select the best one(s) for the present network environment. These mechanisms are expected to consider network elements' available technologies and protocols to make available a communication infrastructure in a simple and automatic manner.

Necessary negotiations can be divided in Intra and Inter-Domain negotiation and based on these two possible classes were defined five points to be solved during this thesis:

- 1.Nodes Self-Addressing: Nodes must be able of configuring themselves when they start to operate in a network and no addressing server is available. This addressing is important to allow the basic connectivity during spontaneous networks establishment;
- 2.Local Addresses Distribution: Networks that belong to an operator are supposed to have a minimum communication structure created and one of the elements that

probably will be available is an addressing server. However, this network can become too big and complex (i.e. operator network) to be controlled by only one server. Other aspect that might need the adaptation of addressing server is the dynamic creation of new networks, what will demand new servers to control them;

3. Pools Allocation: Network might start to operate in a very dynamic and unpredictable way and when it happens they would opt to use a structured address assignment technique. In these situations, the network must be provided by an addresses pool to allow the configuration of the local elements. Today this allocation is manually configured, but with on-demand establishment of new networks it's necessary more dynamic and automatic solutions. To deal with this new distribution model a mechanism to control and distribute the addresses according to the connection of new networks is necessary;
4. Intra-Domain Automatic Configuration: Many specialized communication environments can be found according to the particular technologies and to users' requirements. Considering this scenario, many protocols can be used to change the necessary configuration to these needs. The verification of these possible configurations to decide the best one according to the environment characteristics is necessary to allow network to configure themselves and their components to their specific requirements without human intervention. The continuous adaptation of these configurations to optimize them to the networks changes is also necessary;
5. Inter-Domain policies establishment: In the same way that automatic local nodes' configuration might be performed dynamically to support spontaneous networks to be created, the establishment of communications between networks must be dynamic, allowing networks to find, negotiate and configure the communication with the desired networks. This process might result in a set of policies which will represent negotiated rules and must be used to regulate networks' communication.

These five elements will be discussed and solutions will be proposed to each one of them, supporting the on-demand and spontaneous networks configuration and adaptation, allowing the dynamic negotiation and control of networks communication.

1.3 Thesis Organization

This thesis is structured for introducing the concepts and proposals in a logical fashion. In this chapter, the problem of automatic configuration and negotiation was introduced and the proposed solution was outlined. The other chapters of this document are organized as follows. Chapter 2 describes some of the most important works on networks autoconfiguration. These mechanisms are presented according to their specific operation area: addressing configuration, networks self-configuration and dynamic services negotiation. The requirements of each one of these three areas and most important techniques which can be found in literature are discussed each of these groups. Based on this discussion is possible to have an overview of the characteristics and drawbacks of each area this thesis is proposing new solutions.

Chapter 3 describes the basic negotiation strategies which must be adopted by the domains to allow the automatic configuration of the networks considering addressing, responsible by defining basic communication configuration, and domains configuration to obtain networks' capabilities information and adapting them according to environment capabilities.

Chapter 4 describes the Inter-Domain dynamic policies negotiation mechanism. The concepts developed for policies negotiation are important contributions of this thesis. This chapter first describes the basic negotiation model and its phases. Next, a methodology and the details of the protocol messages and communication to each one of the defined phases are presented including an overview of the policies mechanism necessary to support and representing the necessary policies.

Chapter 5 is aimed at evaluating the proposed mechanisms based on a simulation study. The criteria for comparison depends of the specific mechanism evaluated but in a general way were considered the influence in the network operation and performance of the mechanisms (bandwidth, delay, etc.) and the ability of adapting networks to environment/requirements modifications without human intervention. Chapter 6 concludes the thesis and discusses directions for future work.

2 Networking Autoconfiguration Technologies

In last years we had faced an enormous grow in computer networks and their complexity. Networks which only had a little amount nodes are now composed by hundreds of elements operating in a very dynamic and heterogeneous environment, making current management technologies not capable of dealing with all new characteristics of these environments. SNMP (Simple Network Management Protocol) [HAR99], for example is only capable of working with TCP/IP networks and is not able of consider new environments' characteristics dynamically since it's based on statically defined information bases (MIB – Management Information Base).

Due to its complex nature, the management of this kind of network is a cumbersome task for any administrator. To solve this, solutions which create a degree of ambient intelligence and define mechanisms to automatically configure networks' nodes avoiding administrators to manually intervene in all elements were designed. However, these solutions are very restricted and only solve specific problems. Another problem is that these solutions also need a previous configuration administrator's intervention still necessary.

As current solutions don't solve properly all needs of automatic configuration in networks, several proposals of frameworks [4WA09] [AMB09] [AUT10] to make self-configuration in computer networks can be found in last years. The main objective of these frameworks is to make capable to nodes and networks automatically define their configuration (addresses, routing protocols, network interface speed, radio power, etc.) and keep this configuration “correct” by adapting it to modifications occurred during network operation.

This chapter presents an overview of autoconfiguration mechanisms which can be used in dynamic networks. These mechanisms are classified in three groups according to their objectives: addressing configuration, networks self-configuration and dynamic services negotiation. The requirements and most important techniques which can be found in literature are described to each of these groups.

2.1 Autoconfiguration Basis

Autonomous networks represent networking environments capable of operate with no human intervention. They are composed by nodes that must be able to automatically configure themselves, in a cooperative way or not, without any kind of manual intervention. Autoconfiguration mechanisms for dynamic networks may vary from self-addressing procedures to network layer routing self-stabilization [FOR05]. Scenarios of autonomous networking demand a set of characteristics, such as [BER09] [WIL03]:

- Self-configuring and self-healing: the capability to bootstrap with no/minor user intervention and also to recover from critical situations (failures);
- Decentralized and unmanaged nature: the capability of being independent of any centralized entity for configuration, which can become a single failure point;
- Radio coverage extension ability: the capability of providing service/resource attainability throughout multi-hop wireless networking. Wireless communication is a special concern of autonomous networks since it usually presents a more dynamic/complex communication environment than conventional wired networks;
- Independent operation: mechanisms should be designed to work independently from other technologies, such as routing protocols. Such dependency would limit the solution application to scenarios where the required technologies are operating;
- IP interface configuration: networks need a correct nodes' configuration to guarantee an appropriate communication among them. In this context, devices' interfaces need to be configured as fast as possible and in accordance with the addressing rules

established in the network. Another desirable characteristic is the minimum intervention from users avoiding misconfigurations to happen;

- Translation between the host name and IP address;
- Service discovery: Service based communication must be one of the focus of future networking technologies and services should be associated to routes (networks that provides the service), not nodes. This new communication paradigm makes possible the distribution of information more naturally decoupling resources from its location and enabling a better support to services specific communication requirements.

Some of the current networking protocols and mechanisms provide a level of self-organization in networks, allowing hosts to auto-configure themselves. However, these solutions still require intensive manual configuration to a proper network operation, such as routing protocols, OSPF routing areas, and addressing issues. Focusing on more independent networking systems, some solutions have been proposed in order to automate configuration of routers, servers, network resources and topology. These solutions range from simple local self-addressing mechanisms to research projects intending to define how future networks might be. Some of these works are discussed following.

2.2 Addressing Configuration

Many solutions can be used to allow networks addresses configuration. Static allocation was the initial technique used by administrator to configure networks' elements in order to allow the communication among them. However, this technique is only feasible in restricted networks with a reduced and well known number of elements.

Trying to solve this restriction some mechanisms were developed to allow addresses distribution in structured computer networks. However, existing mechanisms, such as DHCP (Dynamic Host Configuration Protocol) [DRO97] [DRO03], SLAAC (Stateless Address Autoconfiguration) [THO07], NDP (Neighbor Discovery Protocol) [NAR07] and DHCP-PD [TRO03], provide only partial solutions with regard to the goals previously mentioned for current and future networking environments. They are unable to dealing with

specific dynamic, multi-hop and distributed nature of MANETs (Mobile Ad-hoc Networks). Thus, additional work is still needed to fully contemplate such goals.

Autoconfiguration solutions, despite of the parameters and functionalities, must provide all active interfaces with a valid and unique address (or network prefix) within a determined domain. Moreover, these mechanisms should also implement the management of addressing resources. Addressing operation is critical to any network since it is necessary to nodes identification, packets routing and also to security related issues [BAC09].

2.2.1 Requirements for Addressing Autoconfiguration

The IETF Autoconf Working Group [AUT09] establishes a number of requirements that should be seen to in any autoconfiguration mechanism. These requirements can be complemented by the studies of Baccelli [BAC08] and Williams [WIL02]. Based on these works the autoconfiguration, solutions should consider the following main requirements:

- Design Requirements
 - Be independent from the routing protocol used: The mechanism should not require a specific routing protocol to work properly and should not depend on routing functionality. However, [BAC08] states that the solution may leverage the presence of routing protocols for optimization purposes;
 - Provide support mechanisms and/or functionality to prevent and deal with address conflicts, which can be originated for instance from networks merging, local pre-configuration or node misconfiguration;
 - Support characteristics of dynamic and heterogeneous networks such as their multi-hop nature, the potential asymmetry of links, and the variety of devices;
 - Provide backward compatibility with other standards defined by the IETF;
 - Not require changes on network interfaces and/or routers existing protocols;
 - Control networks addresses merging and splitting;
 - Support security mechanisms to guarantee network privacy and integrity;

- Be designed in a modular way. Each module should address a specific subset of scenarios' requirements.
- Operation Requirements
 - Provide the configuration a node's interface(s) with valid and unique IP addresses within a network;
 - Provide the configuration of disjoint prefixes for routers within the same network;
 - Work in independent dynamic networks as well as in those connected to one or more external networks;
- Performance Requirements
 - Generate low overhead of control messages;
 - Achieve their goal(s) with low delay or convergence time;

According to these requirements a sort of proposal have been developed trying to solve new technologies' demands. Some of the main works in addressing configuration are presented and their main characteristics are discussed.

2.2.2 Stateless Solutions

Stateless approaches are also known as conflict-detection solutions. According to [THO07], the stateless approach is implemented when a node is not aware with the addresses that other nodes are using, since they are unique and routable. Such solutions have their focus on mobile ad hoc networks. A node operating a stateless self-addressing approach must be able to self-generate a valid and unique address within a network, and configure its own interface with the generated information.

To ensure uniqueness, nodes might generate addresses by combining local information (e.g. MAC address) and information provided by available routers in the network (e.g. the network prefix information). The information provided by routers is usually gathered from periodic routing advertisement messages. Using this information, the node then creates a

unique identification to its interface(s). If no routers are available within a given network, the node may only create link-local address. Link-local addresses are sufficient for enabling communication among the nodes attached to the same link.

A simpler alternative for stateless addressing approaches is random address selection. Some mechanisms do not implement a deterministic formula for generating an address. Instead, they define a range from where a node picks up one. Consequently, some duplicate address detection (DAD) procedure is also needed. For example, such range may be determined by the link-local prefix as defined by IANA (Internet Assigned Numbers Authority): 169.254/16 [IAN02] [CHE05] [IAN09]. Ones use supposedly unique local information, e.g. MAC (Media Access Control) address, to calculate an exclusively address but depending of the addresses range it might not be sufficient to avoid addresses conflicts.

Different methodologies to calculate a valid address have already been proposed and the main drawback of stateless approaches is that they require support for duplicate address detection mechanism. Even the stateless solutions that adopt some mathematical calculations for address generation, such as by combining information or network size estimation, etc., are also required to perform DAD in order to guarantee address uniqueness.

In this sense, stateless solutions can be more suitable for dynamic networks with high mobility degree, exempting the need for centralized addressing management. Another advantage of stateless solutions is its constant configuration delay. As the basic DAD idea is testing the selected/calculated address within the other network nodes, and this procedure is based on transmission of requests and waiting for replies during a predetermined period of time, the delay tend to be constant, sometimes increasing accordingly to the network diameter. On the other hand, even with the implementation of DAD mechanisms, stateless approaches cannot ensure 100% of uniqueness when configuring nodes in a dynamic scenario, mainly when nodes have high degree of mobility and the nodes are deployed in a short interval. The more complex is the DAD procedure, the higher is the configuration delay and overload generated by the self-addressing mechanism. In addition, solutions that implement strong mathematical efforts demand more processing power and time, which may result in a higher configuration delay.

2.2.2.1 DAD-based Mechanism

DAD mechanisms have the objective of complementing stateless solutions to ensure address uniqueness. As stateless approaches cannot ensure that an exclusive address is being attributed to the node's interface, they should perform DAD before or after the interface is configured. Such mechanisms can act alone in the network as well as take advantage of control packets transmitted in the network (e.g., routing protocols). Some stateless approaches are even named after the DAD mechanism they implement. Some of these DAD mechanisms are presented.

Strong DAD [PER01] is the most classical stateless approach. It is basically the implementation of a conflict-detection mechanism, where a starting node selects an address from a predetermined interval and tests it within the existing network. It is a broadcast-based protocol where all nodes participate of the addressing making sure that any starting node will configure itself with an address already in use. Several variations and improvements have already been proposed over the principles of Strong DAD, such as Weak DAD [VAI02], Wise-DAD [MUT08], and Passive DAD [WEN03-2].

While most of stateless approaches only operate during the startup of the network, i.e. having any type of resources post-management, AIPAC [FAZ06] is a more complete stateless proposal which implements a mechanism to deal with initial configuration, networks partitioning, and merging and gradual merging. Also, AIPAC is an example of solution that another node participates of the configuration of a starting node by obtaining a valid and unique address to it. Moreover, commercial solutions, such as the ones implemented in the Operating Systems of Microsoft and Apple, use a pseudo-random generator auto-configure the interface to local networks. The generated address is within the link-local scope defined by IANA, i.e. 169.254/16 for IPv4 (IP version 4) and FE80::/64 for IPv6 (IP version 6). More details of the main proposal are presented following.

2.2.2.1.1 Strong DAD

Strong DAD [PER01], also known as Pure DAD, is the simplest solution for duplicate address detection implementing a conflict-detection mechanism to ensure a correct address allocation. Its algorithms are presented in a general view in Figure 2.1.

The operational steps of new node which is trying to get configured running Strong DAD is presented in Figure 2.1(A). Initially it randomly picks two addresses: a temporary address and a tentative address, which will be really tested in the network. The tentative address is selected from the range of FIRST_PERM_ADDR to 65534, from 169.254/16. The temporary address is selected from the range of 0 to LAST_TMP_ADDR, and will be the source address of the node while performing the uniqueness check.

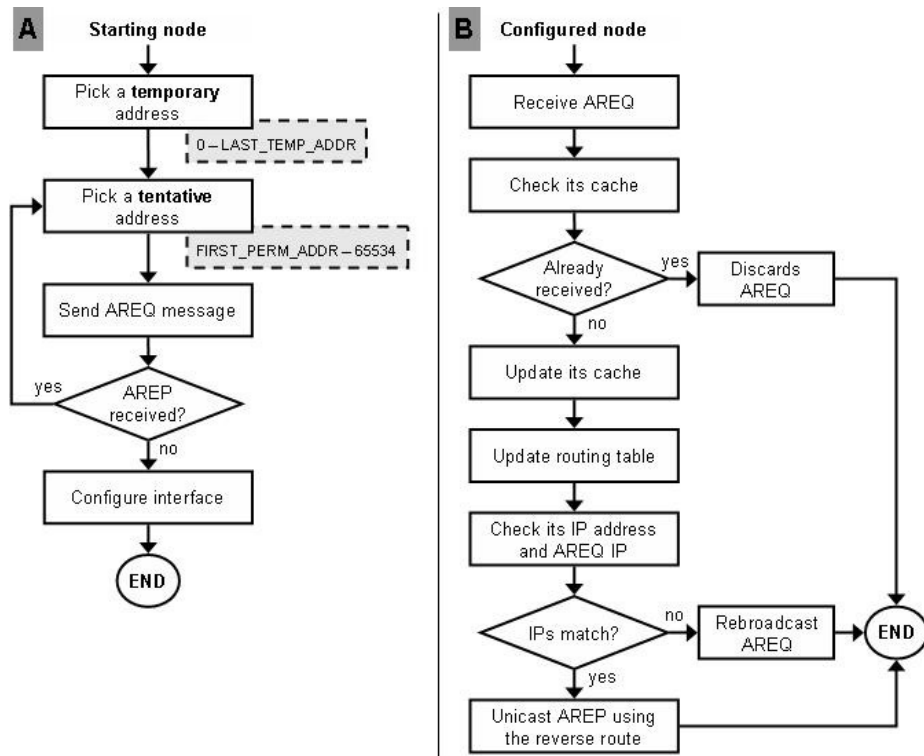


Figure 2.1 – Reduced Strong DAD algorithms.

Address checking consists of two messages: Address Request (AREQ), from the new nodes, and Address Reply (AREP), from configured nodes in response to addresses conflicts detection. After selecting the addresses, new nodes send an AREQ to check if the tentative address is in use. It waits for an AREP during a pre-defined period of time and if no AREP is returned, the starting node resends the AREQ, with the same tentative address, up to a pre-defined number of retries. If no AREP is received after all retries the node assumes that the tentative address is not in use and configures its interface with the tentative address.

When receiving an AREP in response to its AREQ, the initiating node randomly picks up a new tentative address and sends another request claiming it. Then, the algorithm is repeated

until the starting node either gets configured with a valid and unique address or reaches the maximum permitted number of addresses tests.

On the other hand, a configured node, Figure 2.1(B), is expected to receive AREQ messages and check if the requested address is already in use. When it occurs, it seeks the message in its cache to verify if it was already received and discard it if so. Otherwise, the node stores the information in the cache and updates its routing table to the new node.

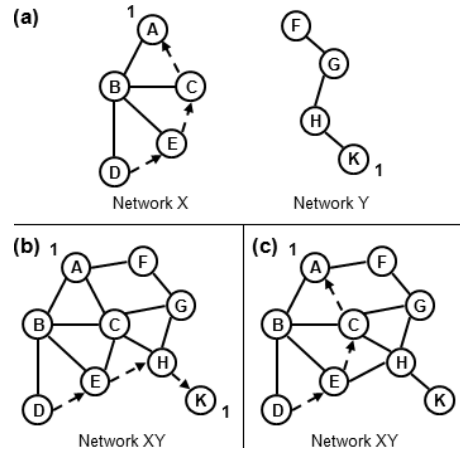
Then, the configured node checks if its own IP address matches the tentative address on the AREQ message. If not, the node rebroadcasts the packet to its neighbors. On the other hand, if the node has the same IP of the received AREQ, the configured node must send an AREP message to the AREQ's source node.

2.2.2.1.2 Weak DAD

Weak DAD [VAI02] was proposed as an alternative to Strong DAD. Strong DAD is not applicable in unbounded delays networks, since it requires the applicability of timeout-based duplicate address detection solutions. Since Strong DAD implements timeout-based DAD, it is not applicable in unbounded delays networks because it will not always detect duplicate addresses due to delay constraints. Weak DAD can be used independently or in combination with other schemes like in [JEO06].

The main difference of Weak DAD to other duplicate address detection mechanisms, is that it relaxes the requirements for detecting duplicate addresses. That is, it does not require the detection of all conflicts in the network. Weak DAD imposes that a packet sent to a specific destination must be routed to this destination and not to another node within the network, even if the packet's destination and the other node have the same address.

In Figure 2.2(a) the nodes are divided in two different and distinct networks. The route to be considered for this example is from node D to node A (dashed route). In this first moment, this packet from D to A travels via nodes E and C, and it is routed using the destination IP address 1 included in the IP packet header. In this scenario, nodes A and K have selected the same address. Figures 2.2(b) and 2.2(c) illustrate the network resulting from the merging between the two networks of Figure 2.2(a). Now, as nodes A and K have the same address, the network has an address conflict.



Source: adapted from [VAI02].

Figure 2.2 – Weak DAD scenario.

In Figure 2.2(b) the route to the IP address 1 has changed. Before the merging, the packets were routed to node A, and now they are routed to node K. According to Weak DAD, this situation is not acceptable. In Figure 2.2(c), on the other hand, the packets addressed to 1 are still being routed to node A, what is a desirable situation. Therefore, Weak DAD suggests that duplicate addresses within a network can be tolerated as long as packets reach their intended final destination correctly, even if the destination's IP address is duplicated.

These modifications start in nodes basic configurations. The Weak DAD assumes that each node in the network might pre-assigned with a unique key. MAC address [IEE02] can be used as the node's key, although the nodes can use another alternative identifier or procedure to generate a key, since this key has a small probability of being generated more than once. Therefore, the Weak DAD uses the key for the purpose of detecting duplicate IP addresses within the network, without actually embedding this key in the IP address.

This initial configuration is used by Weak DAD to make the modifications necessary in the routing protocol to ensure its correct operation. Weak DAD was designed to work with link-state routing protocols where the entire routing table is located in each node and contains an entry for each known node in the network.

2.2.3 Stateful Solutions

Differently from stateless paradigm, stateful solutions keep track of all resources allocated within the network, or at least most of them. To do so, part of the nodes that compose a network, or even all of them, are aware of the addresses allocated to other nodes. This resource tracking enables an easily post-allocation management, which is more complicated within stateless mechanisms, and also exempt the support of DAD. Therefore, stateful mechanisms can also be called conflict-free solutions.

Some solutions build a local allocation table that is updated passively, with information from routing and/or addressing mechanism packets. Others divide a starting pool of addresses among the nodes in the network. These may differ in that some of them implement a technique where the pool of addresses is divided among all nodes in the network, whereas others assume that just part of the nodes might take part on addressing tasks.

Independent of the specific mechanism adopted stateful solutions need to deploy addressing authorities in the network, which are responsible for storing information about nodes active in the network and the addresses in using. Each node which wants to get connected to the network must contact directly or indirectly these authorities in order to confirm that the chosen address is not in use or even to obtain configuration information from this entity.

Stateful approaches have some known weaknesses. When sharing the role of addressing among all nodes, some level of synchronization may be required. If each node keeps an address allocation table, the information held by them must be shared with other nodes within the network. This way, all nodes will have coherent updated addressing information knowing at any time those in use and the ones available for possible future assignments. On the other hand, stateful approaches have the advantage of being more reliable when presenting a lower probability of address conflicts than stateless ones. Also, depending on their communication pattern, stateful protocols can generate low control overload, which also reflects in an improvement on the configuration delay.

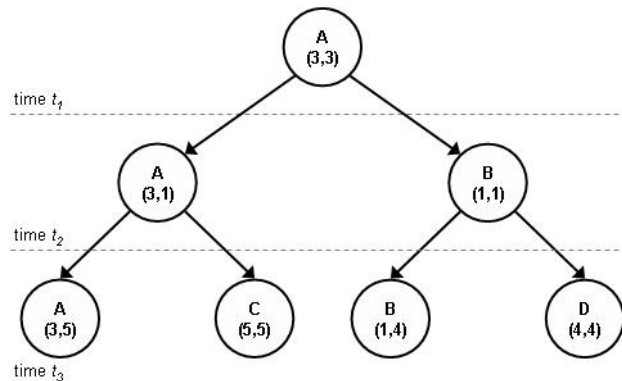
This efficiency with regard to address conflicts comes at the price of having an exhaustive control by the addressing mechanism over its resources. When dividing a pool of addresses among network nodes, these must keep connected in order to determine, for example, if the

network still having available addresses for new nodes, recovery of unused resources and, probably, information backup. Such important control overhead, when compared with a stateless approach, is different according to the stateful approaches. Depending on the considered scenario, one may however be willing to pay the price due to advantages such as address conflict free guarantee, and possibly having applicable scenarios. More importantly, stateful approaches fit better into the future heterogeneous scenarios of the next generation of computer networks. Some stateful approaches are detailed next.

2.2.3.1 Prophet Allocation

Prophet Allocation [ZHO03] is a scheme to deal with addresses allocation in large scale MANETs. According to the authors, it's a low complexity, latency, and overhead mechanism, capable of dealing with the problem of network partition and merging. This solution was named Prophet Allocation because the very first node of the network is assumed to know in advance which addresses are going to be allocated. The base of the algorithm is stateful, where a sequence consisting of numbers is obtained in a range R through a function $f(n)$ to represent the addresses to be allocated. The initial state of $f(n)$ is called a seed, where each seed leads to a different sequence of addresses.

Figure 2.3 illustrates an example of Prophet Allocation operation, where every node is identified by: (address, state of $f(n)$). In this example the range R is $[1,8]$, the effective address range is $[1,6]$, and $f(n) = (a \cdot s \cdot 11) \bmod 7$, where a and s are the node address and the node state respectively.



Source: adapted from [ZHO03].

Figure 2.3 – Prophet Allocation algorithm scheme.

Initially, at time t_1 , node A is the very first node in the network and uses the value 3 as both its IP address and seed. When node B joins the network, node A obtains the value 1 for B, through $f(n)$. Then, at t_2 , A changes its state of $f(n)$ to 1, and assigns 1 to B. At time t_3 , nodes C and D join the network. Using $f(n)$, node C is assigned the value 5 from A, and node D gets configured by B with value 4. Both nodes A and B change their state of $f(n)$ to the values 5 and 4, respectively. At time t_3 four of the six available addresses for the range R have already been assigned without conflict. However, due to the small allocation range of R, the next round of allocation will result in a conflict. According to the authors, address claiming is not implemented by Prophet, needing an auxiliary mechanism, like DAD solutions, to solve that. Conflicts will indeed occur, but the minimal interval between two occurrences of the same number is extremely long when considering big amounts of available addresses. Another point is that, when a new node is assigned an “old” address, the previous node that was using this address has likely already left the network.

Regarding network disconnection and merger, the authors state that Prophet Allocation can easily handle both situations. Considering a scenario with a network B, which used to be part of a network A, when a merging process to network A is started the sequences of each network are different, thus, no address conflict will occur if the networks merge again.

Nonetheless, for the merger between distinct networks, Prophet Allocation implements the concept a Network ID (NID). The NID is randomly generated by the very first node in the network and is supposed to uniquely identify this network. If the NID is large enough, two networks will rarely have the same identification. This ID is known to the network during the address allocation process, and merging can be detected by routing information.

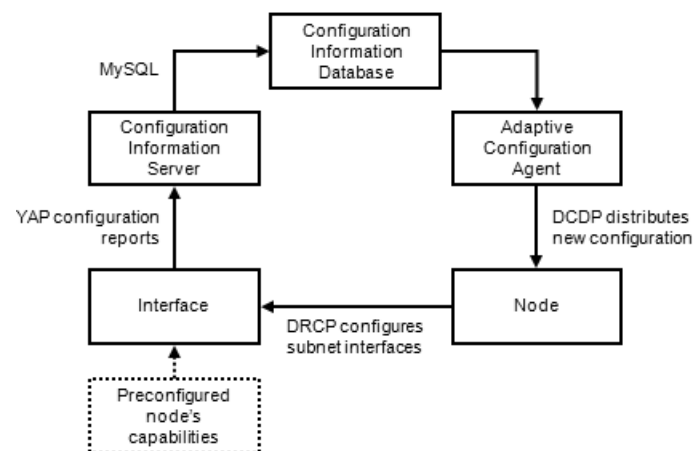
Although Prophet Allocation is categorized as a stateful approach, it may require the DAD operation depending on the size of the range of addresses, to guarantee that no conflicts would exist in the network, as discussed. But, as the DAD procedure is not mandatory during address allocation process, this solution can be considered as a stateful one.

2.2.3.2 IP Autoconfiguration Suite

The work presented in [MAN05] proposes a framework for autoconfiguration of host, router and server information. It includes the automatic generation and maintenance of a

hierarchy, under the same architectural, algorithmic and protocol framework. This framework is composed by two parts. The *Decision Making* part is responsible for obtaining network and hierarchy configuration, according to the network performance objectives. The *Communication* part is responsible for distributing the configuration decisions and collecting the required information used by the *Decision Making* part.

The Communication part of the framework is constituted by the DCDP (Dynamic Configuration Distribution Protocol), DRCP (Dynamic and Rapid Configuration Protocol), and YAP (Yelp Announcement Protocol). These modules are part of the IPAS (IP Autoconfiguration Suite) that is responsible for the network configuration. The modules DRCP and DCDP constitute the core of the autoconfiguration suite. Figure 2.5 illustrates the architecture of the IPAS and the interaction among its components.



Source: adapted from [MAN05].

Figure 2.4 – IPAS architecture and components.

The autoconfiguration suite functionality can be seen as a loop. It initiates with the ACA (Adaptive Configuration Agent) distributing new configuration, from the Configuration Information Database, through the DCDP. In each node, the DRCP configures the interface within a specified subnet. When configured, the interface reports its configuration information, through YAP, to the Configuration Information Server. Finally, the server stores this configuration information in the Configuration Information Database, which will be accessed by ACA to start the operations again.

According to [MAN05], the DCDP is robust, scalable, has low-overhead, and is lightweight (minimal state) protocol. It was designed to distribute configuration information on address-pools and other IP configuration information such as, DNS Server's IP address, security keys, or routing protocol). To work in dynamic environments, one of the DCDP features is that it operates without central coordination or periodic messages. It also does not depend on routing protocols for distributing its messages.

The DCDP relies on the DRCP to configure the node's interface(s). The DRCP protocol is strongly based on the well known DHCP. However, it adds features for supporting roaming users. This protocol is responsible for detecting the need for reconfiguration due, for example, to node mobility. It is reached through periodic advertisement messages. The author also states that DRCP allows for: (a) efficient use of scarce wireless bandwidth; (b) dynamic addition or deletion of address pools for supporting server fail over; (c) message exchange without broadcast; and (d) clients to be routers. In each sub-network there is at least one DRCP server. The other nodes are set to DRCP clients.

2.2.4 Hybrid Solutions

Finally, solutions that follow the hybrid paradigm implement mechanisms that combine characteristics from both stateless and stateful approaches. Usually, these mechanisms allow the self-generation of addresses, the registration of the generated information within a responsible entity in the network, and the execution of DAD procedure to ensure maximum reliability concerning the addresses uniqueness.

2.2.4.1 MANETconf

MANETconf [NES02] is a distributed dynamic host configuration protocol designed to configure nodes in a MANET. It considers a scenario with a mobile ad hoc network while having a connection to an external network. This mechanism establishes that all nodes in the network must accept the proposed address for a new node. That is, a starting node will be configured with a proposed address checked within the network.

The network starts with a single node that configures itself, as illustrated in Figure 2.5(A). After executing the neighbor search, and obtaining no responses from configured nodes, the very first node in the network configures itself with an IP address. Thus, the network is

initialized. When a new node i wishes to join the network, it broadcasts a message *Neighbor_Query* to discover a possible neighbor node. If it receives a reply from an already configured node j , it selects this node as its initiator. Then, node i sends a request message to node j that maintains some information: (a) the set of all allocated IP addresses in the network; (b) the pending IP addresses (addresses that are in the process of being allocated to other nodes). These pieces of information are kept as node j 's knowledge.

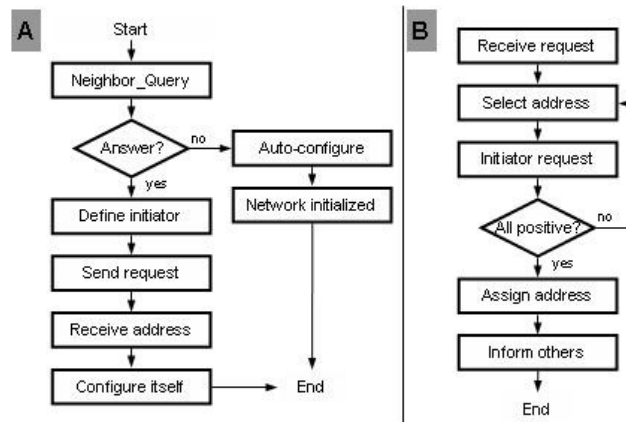


Figure 2.5 – MANETconf basic algorithms.

When node j receives the request from node i , Figure 2.5 (B), it selects an address that is neither in the allocated address set, nor in the pending address set. It is then added to j 's table of pending allocation, and then this address is claimed within the network through a broadcast *Initiator_Request* message. When receiving the initiator request message, a configured node checks if the claimed address matches with the information in its allocated and pending tables. If not, it adds the information {initiator, address} in its pending table and replies to the initiator node with an affirmative message. Otherwise, the receiver sends a negative message to the initiator.

When receiving a positive answer from all nodes, the initiator j assigns an address to node i , adds this address to the allocated addresses table, and informs others about the assignment in order for them to also update their allocation tables. Receiving a negative reply, the initiator node selects a new address and starts the procedure again.

If a node is able to inform about its departure (graceful departure), it floods the network with an *Address_Cleanup* message. When an active node in the network receives this message

it simply removes the leaving node's address from the allocated address table allowing this address to be used for future assignments.

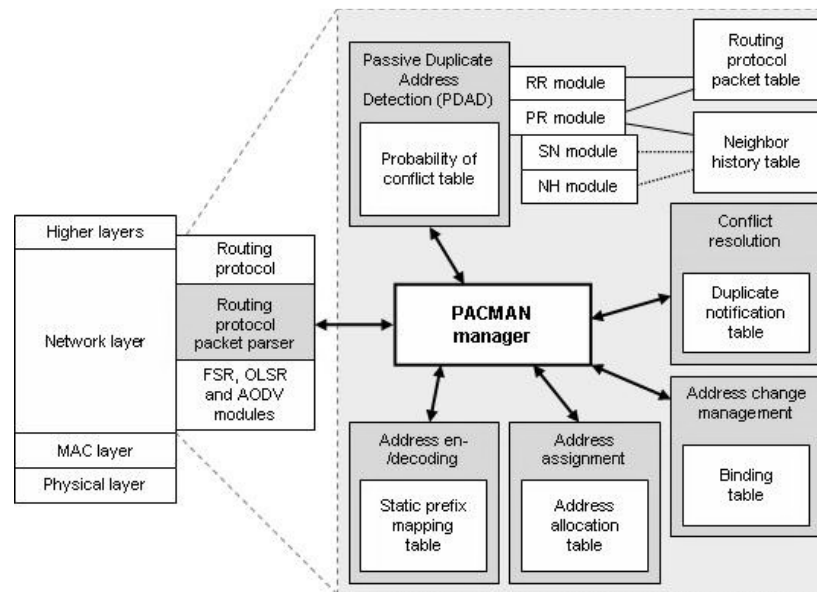
Concurrent initiations, i.e. situations where two initiators pick up the same address for different assignments, are solved considering the initiators' IP address. The one with the lower IP address value has higher priority in the allocation process. When receiving an initiator request message for an already requested address, the node checks the initiators' IP addresses. If a node receives a lower priority initiator request, but it has already received a request from a higher priority initiator, this node sends a negative reply to the lower priority one. Otherwise, if a node receives the request from the lower priority initiator before receiving the request from the higher priority one, both initiators' will be replied affirmatively. However, according to the authors, among multiple conflicting initiations, only the highest priority initiators will receive all affirmative responses, while all other initiators will receive at least one negative reply forcing them to select a new address and start the approval process again.

MANETconf also implements a procedure to handle situations where initiator and requester nodes lose communication with each other during the addressing process. If the requester node i moves away from the range of its initiator node j , they are no longer able to exchange messages. Consequently, node j is not able to inform node i about the valid negotiated address for its configuration. Node i realizes that it lost the communication with its initiator and selects an adjacent configured node k as its new initiator. Then, node i informs k about its former initiator, and k sends a message informing the migration to j . After finishing the address negotiation, node j sends the outcome of the task to the new initiator, and k forwards the result to i . Finally, node i configures accordingly.

2.2.4.2 PACMAN

PACMAN (Passive Autoconfiguration for Mobile Ad Hoc Networks) [WEN05] was defined as an approach for efficient address autoconfiguration of MANETs, using cross-layer information from ongoing routing protocol traffic and utilizes elements of both stateless and stateful paradigms. The PACMAN system architecture can be visualized in Figure 2.6.

PACMAN has a modular architecture. The called Routing protocol packet parser has the objective of extracting information from incoming routing packets. This information is sent to the PACMAN manager, an entity responsible for delegating the information to the respective components. A drawback is that a protocol parser must be implemented to support the routing protocols differences. The Address assignment component is responsible for the self-generation of addresses, by selecting them through a probabilistic algorithm, and the local allocation table maintenance.



Source: adapted from [WEN05].

Figure 2.6 – PACMAN modular architecture.

The module of Passive Duplicate Address Detection (PDAD) is responsible for address conflict detection. The advantage of using PDAD is that this mechanism does not send control messages. Instead, address monitoring is done by analyzing incoming routing protocol packets. When detecting an address conflict, PACMAN triggers the Conflict resolution component, which notifies the conflicted node. In addition, the Address change management component can inform communication partners about the address change in the network and, consequently, it may prevent transport layer connections failure.

The maximum number of uniquely identified nodes, within a network, strongly depends on the size of the available addressing space. Considering this, the solution also proposes a component for IP address encoding. The basic idea of this approach is to encode addresses on ongoing routing packets in order to decrease the routing control overhead. The encoded

addresses are used below the network layer, and decoded back to the original IP address to the higher layers. This also allows compatibility with the IP addressing architecture.

The address self-assignment in PACMAN is done through a probabilistic algorithm. Using the information of a pre-defined conflict probability, an estimation of the number of nodes and an allocation table, the algorithm calculates the virtual address space. Then, it randomly selects an address from the calculated space. Lastly, using the local allocation table, it ensures the address has not already been assigned to other node. If no local conflict is detected within the local allocation table, the address is assigned immediately by the node. According to the author, the probability of address conflicts is almost zero, and if a conflict happens, it is resolved by the PDAD in a timely manner.

The address conflict probability depends on the size of the address space and on the number of nodes in the network. The larger the address space and lower the number of nodes, the lower the conflict possibility. To evaluate the probability of address conflict, the value of this probability is calculated. An analogy with the well-known birthday paradox [SAY94] is done.

PACMAN defines that each node is free to choose its virtual address space size. It is because each node is responsible for assigning an address to itself and it does not depend on a global state. The author states that, regarding the desired conflict probability as a predefined quality-of-service parameter, and given that the number of nodes within the network is known, the optimal virtual address space size can be calculated by each node.

According to [WEN05], equation (2.1) gives an estimate of the conflict probability when using an allocation table with j being the number of hidden allocated addresses, r the virtual address space size, and n the number of nodes in the network. The number of hidden allocated addresses j can be estimated from the allocation table. If the number of hidden allocated addresses is equal to the number of nodes, i.e. $j = n$, then the allocation table is empty. And, finally, if the number of hidden allocated addresses is zero, this indicates that all the allocated addresses are known and, consequently, the conflict probability is zero.

$$P(Ec) \approx 1 - (r \cdot e)^{-j} \frac{(r - n + j)^{r-n+j+\frac{1}{2}}}{(r - n)^{r-n+\frac{1}{2}}} \quad (2.1)$$

2.3 Networks Autoconfiguration

Next generation of networks need to facilitate interaction between users and its devices and more than that, facilitate the whole process of communication establishment. In this context self-* mechanisms represent a unique role: work as the element that will make possible elements is far away networks start to communicate without any interaction from the users and no previously defined agreement. Many proposals can be found in this way and some of them will be discussed during the rest of this chapter.

2.3.1 Requirements for Network Autoconfiguration

With the development of new networking technologies with support to mobility and heterogeneity, such as Ad hoc, Mesh or Sensor Networks, autoconfiguration has become a definite challenge for the networking community. A new network model might support spontaneous and multi-hop networking without relying on any precise and a priori infrastructure, allowing networks to have an automatic startup process.

Autoconfiguration of nodes in this context is a crucial issue since they are expected to form spontaneous multi-hop networking environments without infrastructure in which physical topology are dynamic due to node mobility and nodes/services availability. The network topology cannot be predicted and it has a significant impact on network protocols and nodes communication. Due to such properties future networking technologies face unique technical challenges: must perform network operations with limited resources (bandwidth, battery) in a very complex environment because of nodes' mobility and heterogeneity and networks' dynamism.

Another challenge concerns the scalability of the proposed approaches. As the network grows, the routing tables will grow as well, causing more processing, storage and data exchange to maintain the tables.

Other examples of important requirements that must be considered to networks' autoconfiguration are [AUT09] [BAC08]:

- **Compatibility with Internet Protocols:** as many specific technologies are expected to interoperate in Future Internet, and networks might support Internet's protocols in

order to communicate with current networks and to have a standardized communication which allows networks to exchange information;

- Robustness to mobility: mobility within a network is very common, but it is also necessary to consider the mobility of entire networks, keeping information consistency and not degrading network's performance;
- Management of networks fusions and partitions: These procedures are important to allow networks to be created automatically with no predefined infrastructure and obliges the networks to modify their control planes to represent the new state;
 - Fusion: integration of separate networks in a single one that will share the same address and name spaces. This integration might create configuration conflicts which need to be solved in order to guarantee networks operation;
 - Partition: Splitting of a single network in separated ones that will have different address and name spaces to execute their functions;
- Support to heterogeneity: Many technologies have been developed during last years and much more is expected to next years. These new technologies represent different solutions based on specific requirements and consequently might have specific protocols, applications and communication technologies which might interoperate;
- Low complexity of addresses spaces management: addressing is one of the basic mechanism that all networks have to implement in order to guarantee correctness in the basic communication of the nodes;
- Low complexity of routes discovery: users expect that networks deliver as fast as possible and one of the key points in this processes is routing. Obtaining information regarding available paths rapidly and with the lower cost to the network is extremely important to improve networks' operation;
- Low overhead to keep control plane up to date: networks' control plane might be updated to ensure the correct representation of their capabilities and resources state.

However this operation consumes some resources (bandwidth, memory, CPU time, etc.) and might be diminished to not compromise nodes communication;

2.3.2 Autonomic Computing

Autonomic Computing [AUT10] is a term used to describe a broad range of standards, architecture, products, services and tools that enable technology systems to be self-managing. This self-managing can be separated in four main classes of necessary activities: self-healing, self-tuning, self-configuring and self-protecting.

This name is derived from the body's autonomic nervous system. It controls activities, such as heartbeat, blood pressure and breathing that enable the body to self-regulate and adapt to changing conditions. In much the same way, self-managing autonomic capabilities anticipate system requirements and resolve problems with minimal human intervention, and as a result, support professionals can focus on tasks with higher value to the business.

However, there is an important distinction between autonomic activity in the human body and autonomic activities in computer systems. Many of the decisions made by autonomic capabilities in the body are involuntary. In contrast, self-managing autonomic capabilities in computer systems will perform tasks defined by the administrator following a set of policies.

In short, an autonomic system senses its operating environment, models environment's behavior, and takes action to adapt the environment in order to solve detected problems. It has the following four activities:

- Self-Configuring
 - Characteristics that enable systems to adapt to changing conditions by changing their own configurations
 - Functionality that allows the addition and removal of components or resources without service disruption
- Self-Healing
 - Capacity to recognize and diagnose deviations from normal conditions and take action to normalize them

- Capability to proactively avoid situations that could cause service disruptions
- Self-Optimizing
 - Ability of the system to monitor its state and performance and proactively tune itself to respond to environmental needs
- Self-Protecting
 - Incorporation of intelligence to recognize and avoid security threats
 - Facility of a system to protect itself from physical damage

The promise of Autonomic Computing includes capabilities unknown in traditional products and toolsets. It includes the capacity not just to take automated actions, but to do so based on a new ability to sense and respond to change. Not just to execute rules but to optimize environment performance in real time. Not just to store and execute policies, but to incorporate self-learning and self-managing capabilities, allowing the system to take an appropriate action based on one or more situations sensed in the environment.

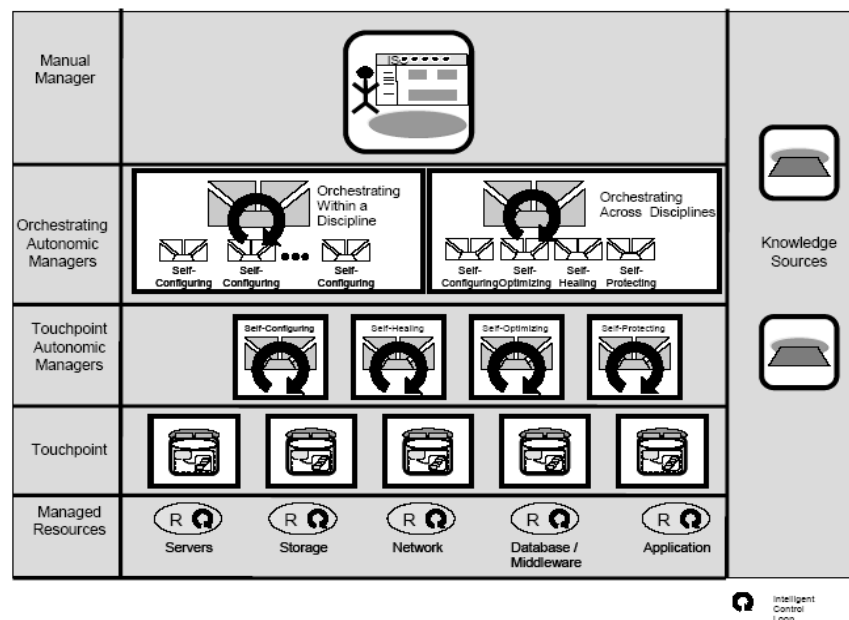


Figure 2.7 – Autonomic Computing Reference Architecture [AUT10]

To facilitate this evolution, IBM has detailed an architectural blueprint for Autonomic Computing that describes the building blocks required for disparate products to work in concert. These building blocks (Figure 2.7) are as follows:

- **Autonomic manager (Orchestrating and Touchpoint management):** The autonomic manager manages a system resource by collecting information from a monitored resource, analyzing the information, using policies to plan responses, and executing the appropriate responses for a particular state.
- **Touchpoint:** A touchpoint, sometimes called a manageability endpoint, is a consistent, standard manageability interface for accessing and controlling a managed resource.
- **Managed Resource:** A managed resource can be any type of hardware or software resource, for example a server.
- **Knowledge source:** A knowledge source is a registry, dictionary or other repository that contains data relevant to the autonomic system. This knowledge source shares its information with the autonomic manager, giving it supporting data required for its activities.
- **Manual manager:** This is a user interface or console that enables a human to perform management activity. It enables the IT professional to perform tasks related to the Autonomic Computing infrastructure, such as policy definition and delegation of actions and tasks to autonomic managers.

2.3.3 Ambient Networks

Ambient Networks (AN) [AMB09] represent a networking paradigm which intends to establish an interoperation among heterogeneous networks, allowing users to access the services they request, independently of their location or the access networks they are using and where such services are provided. Based on this new level of interoperation ANs have as their main goal creating scalable and affordable network solutions for mobile and wireless systems beyond 3G [NIE04].

Research in the Internet community on future networks architectures is mainly influenced by the discovered deficiencies of the current Internet [NIE04], where mobile networks and mobility aspects are treated with comparatively low priority. Ambient networks are based on all-IP mobile networks and can be regarded as the outcome of a continued adoption of Internet design principles [NIE04]. The problem of heterogeneous control for the services is solved through the establishment of an Ambient Control Space, which embraces a well defined set of control functions required to guarantee the cooperation between networks.

Main concepts involved with Ambient Networks are network heterogeneity, mobility and composition. Network Composition is seen as the dynamic realization of a Composition Agreement (CA) that resulted from negotiations among networks and users, and which establishes rules and policies for these to follow while cooperating with each other [D1.5]. In other words, these agreements establishes, for instance, a safe path over which two networks will communicate, the functional cooperation or composition involving the physical integration among the network nodes, called network integration composition, where there is a complete merging of networks.

Mobility includes dynamic networks, dynamic services (service dynamicity involves mobility and availability of services), moving networks and roaming users as well as constant updates of information stored by networks. Moving networks may be seen as a user network, composed of different devices connected to him/her, and that they can be in movement. PANs (Personal Area Networks) are examples of these networks. The Mobility requirement involves many solutions to mobile and wireless networks, to provide communication services better quality, scalability, easy of use and reachability by everyone. Furthermore, AN mobility requirements also include the support of constant updates of information.

Heterogeneity includes a variety of aspects such as different types of devices, different network operators, and a multitude of technologies, e.g. link technologies, IP versions, media formats, and user contexts.

There are other AN features and requirements such as security and privacy, backward compatibility and migration, network robustness and fault tolerance, quality of service, multi-domain support, accountability and context communications.

2.3.4 4WARD

Current Internet technologies have brought new requirements and much more attention to application level and how it might improve users' experience when using the network. On the other hand structural changes are also necessary to allow current and future solutions to obtain as much as possible from networks infrastructure. 4WARD [4WA09] overcomes this through a set of architectural approaches built on a mobile and wireless scenario. The interoperability of many architectures co-existing through a carrier-grade virtualization of networking resources is defined to better adapt to new networks requirements.

4WARD reaches this by creating a new generation of dependable and interoperable networks providing direct and ubiquitous access to information, evolving and replacing today's Internet paradigms. This will pave the ground for more advanced and more affordable communication services using a “clean slate approach” to address these issues.

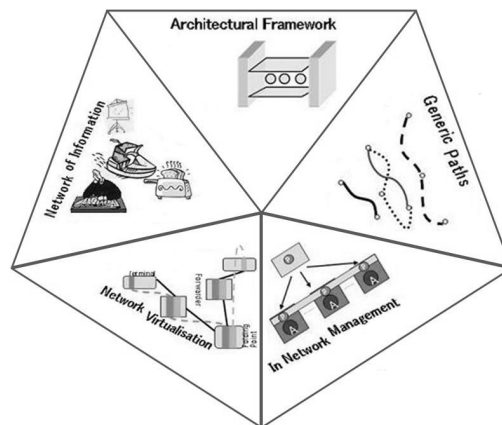


Figure 2.8 – 4WARD research areas[4WA09]

The term ‘clean slate approach’ stands for a coherent solution that breaks the current network's stagnation imposed by the need to support current technologies. 4WARD creates a framework of innovative networking models that together define the direction towards a ‘Network of the Future’, designing an entirely new global communications infrastructure. The key technology research areas identified for the 4WARD approach are (Figure 2.8):

- Architectural framework: 4WARD will improve the ability to design inter-operable and complementary network architectures and develop an integrated framework to represent, design, implement and operate all specific “network instances”.

This approach allows a plurality of network architectures designed according to specific characteristics of each environment, enabling the selection of “the best” network for each task, device, customer, and technology.

- **Virtualization:** 4WARD will enable the co-existence of multiple networks on a single infrastructure through the virtualization of networking resources, providing means to managing different networks inter-operating on a single infrastructure in a commercial setting. Virtualization can also be used to provide a smooth path for the migration towards more evolutionary approaches.

By decoupling the infrastructure from the services, virtualization can provide the opportunity to roll out new architectures, protocols, and services without going through the slow and difficult process of creating such consensus. Virtualization further provides a way for networks to share a common physical infrastructure, what is particularly interesting in networks which might to change rapidly its resources allocation to support users' applications requirements. Virtualization problem space is divided into three main areas, shown in Figure 2.9. Network Resources represents the virtual network which will represent the resources used according to the accesses service, Provisioning of Virtual Networks indicates de elements necessary to create each specific virtualized network, and Management of the virtual networks manages resources allocation and reallocation when necessary.

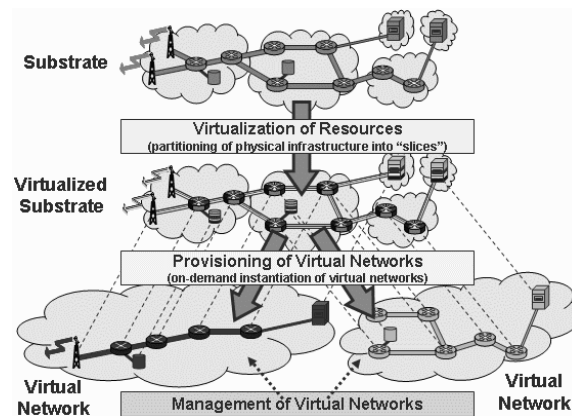


Figure 2.9 – 4WARD networks organization [4WA09]

- **In-Network Management:** 4WARD plans to change networks management structure creating an embedded ‘default-on’ management capability which is an inseparable part of the network itself, different of traditional networks management which remains an external process (Figure 2.10). Based on this internal management all elements are capable of controlling their own operations facilitating devices adaptation to environments modifications.

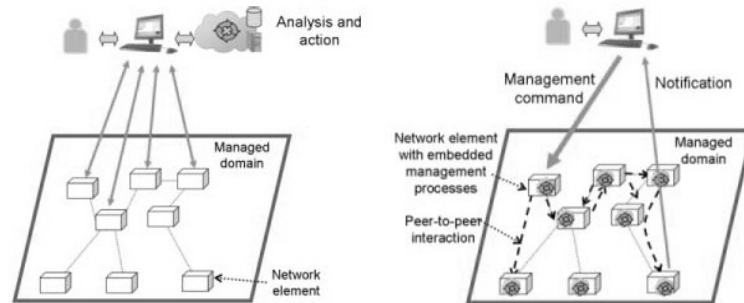


Figure 2.10 – Traditional and In-Network Management in 4WARD perspective [4WA09]

- **Generic Connectivity:** 4WARD will replace the current point-to-point forwarding and routing scheme by a new paradigm of functionally rich communication paths, increasing the capacity and dependability of networks composed of mixed technologies and allowing them to deal with new requirements in an integrated way.

This generic path abstraction can takes advantage of recent communication techniques (network coding, multi-path routing, opportunistic contacts, etc.). Generic path concept also provides an easier and more efficient adaptation of the network to modification of the underlying network, improving communications’ performance and resilience.

- **Content-Centric Network of Information:** 4WARD intends to reorganize applications implementing an information-centric paradigm instead of the current host centric communication. Based on this concept a “network of information” is created and objects have their own identity and are no longer bound to specific hosts. Figure 2.11 illustrate this new approach: two users interacting with the network and according to the specific information requested each one will have a different view of the network.

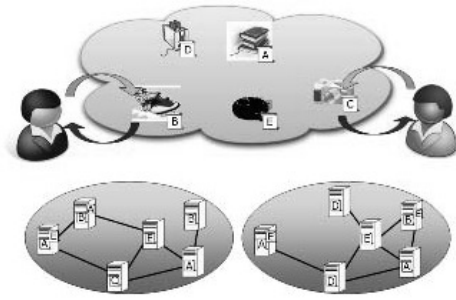


Figure 2.11 – 4WARD network of information [4WA09]

These 4WARD technology solutions shall embrace the full range of current and future network technologies. 4WARD technical results will be disseminated in the context of nontechnical drivers to bridge the gap between innovative research results and utilization for the benefit of the economy and the society at large.

2.4 Dynamic Service Negotiation

The Internet has experienced a rapidly evolution in its technologies and became a very attractive commercial communication infrastructure to many companies that were looking for a new solution to distribute information and offer their services. This commercialization of the Internet has brought many new requirements from users that are demanding new services which could guarantee a better experience during their connections, ensuring specific characteristics for the requested contents.

It is envisioned that in future networks, users will enjoy different levels of service providers. An important characteristic that might be supported by these future networks is to allow users to dynamically adjust their desired service levels. This feature is needed not only to provide flexibility for users, but also because of the heterogeneity in the wireless subsystems and end users' devices.

To control how the resources of their networks are used, organizations need to express how these networks might be accessed and which resources are available. These organizations apply enterprise logic on business rules by specifying and enforcing policies about access control, Service Level Agreement (SLA) and dynamic resource provisioning on applications. Policy-based management has the advantages of being able to dynamically change the behavior of a managed system according to the rules that are identified during system's

operation, however, in a general way, business rules remain static in backbone networks, not allowing users to interact with them and customize the necessary configuration.

Trying to minimize this problem and allow users to define dynamically the characteristics of the services they want to access some solutions has proposed mechanism to negotiate services between users and providers, establishing communication characteristics in a dynamic way and taking into consideration the specific requirements from the users and the available resources from the networks. In the rest of this section the main requirements necessary by protocols to support service negotiation and discussed and some of the proposed solutions are presented.

2.4.1 Requirements for Service Negotiation Protocols

Some studies have analyzed the most important Service Negotiation Protocols proposals regarding to their characteristics and the requirements considered during their design [GOD01] [SAR06]. To facilitate the understanding of these characteristics they were divided in two groups: requirements for basic and advanced capabilities. Most important of these requirements are discussed next:

- Basic negotiation capabilities: Any service negotiation protocol should support the following set of operations to be able to negotiating and establishing SLs (Service Level Specification) dynamically [GOD01]:
 - A client should be able to specify and request a new service to its ISP;
 - A service provider should be able to communicate its acceptance or rejection of the requested service to the client;
 - The protocol should enable a service provider to modify a requested service and renegotiate it with the corresponding client if necessary;
 - A client should be able to accept or reject a service proposed by the service provider according to its needs;
 - The service provider should be able to modify a client's accepted service if it's not possible to maintaining it.

- Advanced negotiation capabilities [GOD01] [SAR06]:
 - Compatibility with QoS architectures: traffic between the end hosts may have to pass through networks owned by several service providers and the protocol employed for end-to-end service negotiation should be compatible with standard QoS architectures;
 - Independency of communication technology: traffic between two hosts may have to pass through many communication technologies. Hence, it's necessary having a service negotiation independent of the specific technology used by the user to get connected;
 - Reduced signaling overhead: the service negotiation protocol should be scalable in terms of the signaling required between the subscriber and the service provider;
 - Transparency to SLS parameters: SLS might change according to the specific characteristics that might be negotiated and the information necessary to be defined. In this way, service negotiation protocols should not predefine the format of an SLS;
 - Lightweight: mobile subscribers with ubiquitous connectivity are one of the main subjects of NGI (Next Generation of Internet) and dynamic service negotiation proposal. Consequently, these protocols are expected to be used across devices with varying capabilities in terms of battery, computing power, and memory;
 - Policy specification, analysis and enforcement: dynamic service negotiation is one solution planned to be used to facilitate dynamicity in networking environments and the result of this process should feedback the environment. This process might be controlled by policies in order to make it as automatic as possible, allowing networks to reconfigure their agreements. In this sense, a complete policy framework should include a policy specification language to allow for the expression of policies, a policy deployment model for the

distribution of policies, a policy analysis mechanism for making the correct policy enforcement decisions, and a policy monitoring and enforcement mechanism to allow for the identification and execution of policy actions;

Based on these main requirements it possible to define the most important characteristics that a service negotiation protocol should have and how it have to operate in order to allow the dynamic definition of the necessary characteristics to access services. Some of the most important works on service negotiation protocols are discussed following.

2.4.2 Service Negotiation Protocols

Since negotiation is a generic concept it can be applied to different areas in order to define a set of characteristics of any kind of device. In computer networks many situations could be improved by applying negotiation techniques. Service negotiation in particular can give to the users the possibility of defining the characteristics of the services that they want to use and to the networks of establishing how these services will be offered. Following some examples of service negotiation protocols are discussed.

2.4.2.1 RNAP

RNAP (Resource Negotiation and Pricing Protocol) [WAN99] was one of the earliest protocols developed to facilitate dynamic service negotiation in NGI and can be considered an extension of Resource Reservation Protocol (RSVP) [BRA97].

RNAP defines a negotiation process which allows user applications to request and use service in accordance to their requirements through an agent called Host Resource Negotiator (HRN). The customer and network negotiate an agreement upon specifications such as the type of service the user will access, the constraints of the traffic, and the price to be charged for the service. The protocol supports multiple delivery services and environments (IntServ, DiffServ, and best effort), service negotiation at different levels of granularity (flow- and aggregate-based), negotiation by both sender and receiver, and “in-band” and “out-band” resource reservation mechanisms. Another feature of RNAP is that it employs a soft state approach to negotiation; hence, periodic signaling from the subscriber is required to refresh the negotiated services.

During the negotiation process, RNAP allows the service provider to communicate service availability, estimated prices, and allow the user to request a specific service. It also supports dynamic service re-negotiation between the user and the network, allowing the network to adjust pricing in response to changes in network load, and allowing the user to respond to changes in application requirements.

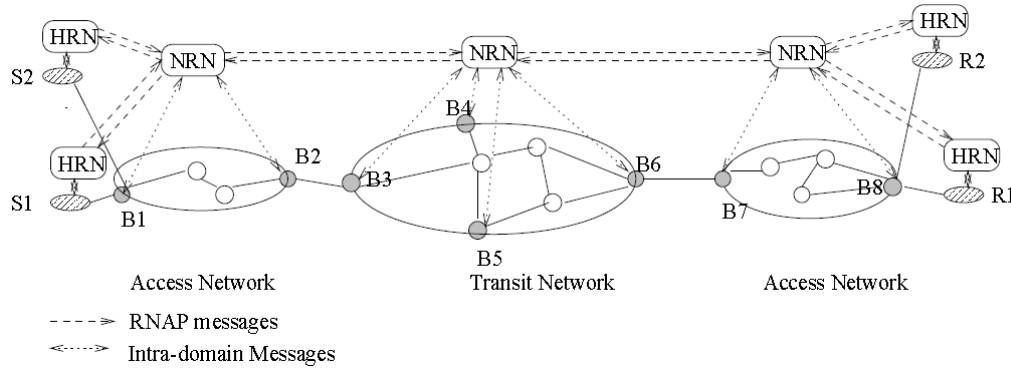


Figure 2.12 – RNAP Centralized Architecture [WAN99]

RNAP was designed to work with both centralized and distributed implementations. In a centralized architecture (Figure 2.12), the network negotiates through a Network Resource Negotiator (NRN). In Figure 2.12 it is possible to see that each administrative domain has at least one NRN which is responsible for delivering price quotations for the available service levels to HRNs, answering service requests from HRNs, and maintaining and communicating user charges for a particular session.

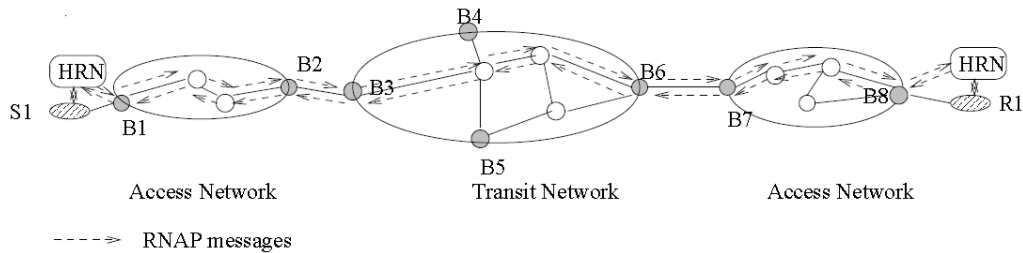


Figure 2.13 – RNAP Distributed Architecture [WAN99]

On the other hand, when working in a distributed architecture (Figure 2.13) no centralized negotiating entity is present. Instead, the protocol is implemented at routers in the network, and RNAP messages propagate hop-by-hop, from the first-hop router to the egress router,

and vice-versa. Figure 2.13 presents a network where only HRN are implemented and consequently RNAP messages are directly transmitted through the routers. In this case the routers are obligated to implement RNAP protocol since no NRN is present in the network.

A second process defined by RNAP is a pricing strategy to provide services' information. This pricing system includes monitoring of user traffic, price formulation at one or more points of the network, computation of a global, or end-to-end, price for a particular service, and a mechanism to communicate pricing information from the network to the customer.

2.4.2.2 COPS-SLS

COPS-SLS [NGU02] is an extension of the COPS protocol [DUR00] used to negotiate SLSs between a customer and a network or between two networks. COPS protocol is based on Policy-based Management Architecture [VER02], which defined a set of elements and their interaction as presented in Figure 2.14.

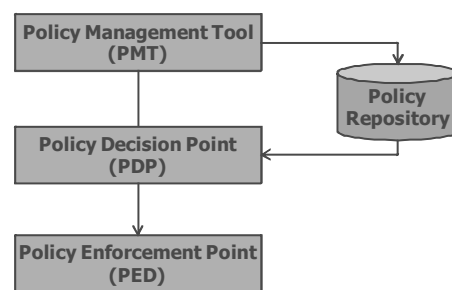


Figure 2.14 – PBM Architecture

The PBM Architecture is comprised of Policy Decision Points (PDPs, also known as policy servers), Policy Enforcement Points (PEPs), Policy Management Tool (PMT) and the Policy repository. The PDP is responsible for handling requests, querying the policy repository, making decisions and distributing them to the PEPs, which are the entities (e.g. routers) where the actions are actually implemented and/or enforced. The PMT supports the specification, editing and administration of policies, possibly through a graphical interface.

However, COPS just define device-independent abstractions for network management, services for SLA deployment within the abstractions, and architectural methods supporting highly automatic provisioning of these. Simple and automatic network configuration methods are necessary to decrease the time needed for SLA negotiation and deployment.

These automated configuration methods improve the network providers' flexibility in support of the customers' dynamic needs.

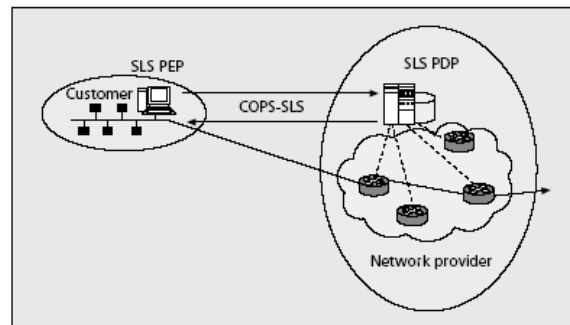


Figure 2.15 – COPS-SLS communication model [DUR00]

According to the COPS-SLS model shown in Figure 2.15, the SLS-PDP represents the network provider and the SLS-PEP the customer. These elements are responsible by detecting problems with network congestion and QoS adaptation and remediate them by enforcing new policies, what requires end systems to be aware of the traffic they generate.

A characteristic feature of COPS is that it distinguishes the interactions between a subscriber and the network manager into two phases: configuration and negotiation. In the configuration phase, the service provider informs the subscriber how to request a specific level of a desired service through the PDP. Having the information about the negotiation mode, the subscriber sends the negotiation configuration to the PEP, which install this configuration and starts the negotiation phase. During negotiation the PEP sends a request for its desired level of service and the PDP can accept, reject, or propose another level to the client. The PDP can also send an unsolicited decision if the network is not capable of keeping the service level negotiated and needs reconfigure the negotiation characteristics.

2.4.2.3 Dynamic Service Negotiation Protocol

DSNP [CHE02] is a protocol developed exclusively for dynamic service negotiation from host to network, network to host, and network to network. DSNP can be used in both wired and wireless networks, since it is independent of network architecture, and how resource reservation and provisioning is done.

DSNP was designed to be independent of the link layer. To do so DSNP negotiates at the network layer and considers that IP protocol is the protocol responsible by this level in wired and wireless networks, representing a unique protocol supported by all network elements. As the negotiation takes place at the IP layer, DSNP's QoS architecture should work well in an IP environment and, consequently, support all IP networks QoS frameworks proposed or any new one since IP protocol has no changes.

One advantage of DSNP only negotiate at the network layer is that, devices only need to have one common protocol (IP) to negotiate when roaming in a heterogeneous environment. It isn't necessary to them have any specific information related with the particular communication technology. This doesn't mean that specific negotiations are not necessary in the network; however, a higher level solution might exist for dynamically negotiating end-to-end services across networks with incompatible access technologies.

Whenever a mobile subscriber negotiates for some service, the network manager disseminates the QoS profile of the subscriber not only to the edge router that serves the wireless network in which the subscriber is currently located, but also to those that serve the wireless networks adjacent to the subscriber's current location. Consequently, when the subscriber moves into any of its adjacent networks, it continues to enjoy the negotiated service without any additional signaling.

Service negotiation may also involve human interaction, since users or the service providers may pre-define and store their policy. These "static" policies, which represent information related with the business model of the operator or with the requirements defined by users, are also supported by DSNP and allow the networks to perform negotiations without human interactions at the same time it considers their preferences.

2.5 Summary

Network Autoconfiguration is a vast research area in computer networks and have received more attention in last years since new solutions to facilitate networks operation and users' interaction has been developed.

Many functions can be influenced by autoconfiguration in order to control specific characteristics of the networking environment and allow the operations of the networks to be performed with no human interaction or with the minimum interaction possible. Three of these functions were discussed here: addressing autoconfiguration, networks autoconfiguration and service negotiation to allow networks to configure their communication with other networks.

Addressing autoconfiguration has as its main objective is configuring nodes within a network allowing them to have basic connectivity and making possible disseminating information in the network. Existing protocols can be classified in three approaches: stateless, stateful and hybrid. Stateless approach is a mechanism which allows nodes to generate its own address, i.e. the node is not aware about the addresses in use by other nodes within the same network. Approaches following the stateful solutions differ from stateless ones because they keep the addresses state within a network. Finally, hybrid approaches combine characteristics of stateless and stateful paradigms. Usually, hybrid solutions maintain an allocation table and also perform DAD attempting to ensure a higher level of reliability when configuring a new node with a valid address.

Network autoconfiguration solutions are basically frameworks proposals in which the configuration of the environments can be performed automatically according to users' communication requirements and to the available resources of the network. The major problem of these solutions is that they are too complex since they need the implementation of many elements and protocols to ensure the correct configuration of the environment.

Service dynamic negotiation protocols aimed to allow networks to define services that can be accessed by the users according to their requirements, making possible the network to have an optimized configuration of the services that might be offered. The solutions, in a general way, are extensions of existing protocols and only focus on QoS requirements reservation, not considering other characteristics that might be controlled, such as mobility, security, and pricing.

3 Basic Negotiation Mechanisms

Current networks structure was not designed to dealing with an environment with a high dynamicity and needing automatic mechanisms to discover and negotiate characteristics and enforce configurations in a network [MANO05] [WIL03].

In current structures, a lot of information must be previously obtained in order to create the necessary knowledge to configure networking environments [MAN05]. This configuration remains static since it's based on off-line information used by the system, requesting administrator's intervention by changing inputs to keep the network up-to-date.

Other shortcoming of current structure is that depending which service or device the administrator is trying to configure, a different set of specific information must be obtained by him and passed to the device, obligating the administrator to know lots of details and characteristics of any particular element present in the network.

On the other hand, future networking expects a completely different perspective regarding networks' dynamicity [CLA90]. The cooperation of heterogeneous networks that might belong to different operators or technology domains will become a reality and mechanisms to enable that must be designed [MANO05] [MOR03]. New mechanisms might consider that cooperation should be transparent, established on demand, and support "plug-and-play" operation, what give us a situation where no previous configuration/negotiation should be required when forming new networks neither while interconnecting different networks.

All possible characteristics and their configurations should be considered and negotiated by networks and their nodes to allow the dynamic adaptation of them [CLA06]. Considering these negotiations, some mechanisms should be designed to deal with specific environment's characteristics that must be controlled according to networks' technologies.

This chapter describes the basic negotiation mechanisms necessary to allow the automatic configuration of the nodes in an Intra-Domain environment. These mechanisms are necessary to the Inter-Domain negotiation to guarantee that each Intra-Domain involved in the process are correctly configured and based on that are able of negotiating the communication rules with other networks.

In the following sections the addressing negotiation and the Intra-Domain configuration negotiation will be described. The first is responsible by making the minimum configuration necessary to allow the connectivity of the nodes (interfaces configuration), and the second is responsible by verifying the capabilities of the network elements and define higher levels configurations such as routing, services distribution over the nodes or intra-domain policies.

3.1 Addressing Negotiation

The first negotiation that will be considered by this work is related with the addressing mechanism necessary to allow the connectivity of the nodes in the network.

In a dynamic situation the current solutions designed to solve addressing problems are not efficient and cannot deal with a variety of situations necessary [PER01] [JEO06] [SUN03]. Existing mechanisms, such as DHCP [DRO97] [DRO03], SLAAC [THO07], NDP (Neighbor Discovery Protocol) [NAR07] and DHCP-PD [TRO03], are unable to dealing with dynamic, multi-hop and distributed environment as expected in MANETs, for example.

Three different phases should be considered to deal with different situations in which the network might be and need a specific solution:

- Local Self-Addressing: used to allow the configuration of the network nodes when no connection with external networks is available and, consequently, no Local Address Distribution mechanism can be used;
- Local Addresses Distribution: allocation of the addresses to the nodes that are composing the network based on the obtained pools, and;
- Addresses Pools Distribution: distribution of address pools among networks to provide them with unique addresses to be used by local elements.

The details of the proposed mechanisms to each one of these areas are presented in more details in the following sections.

3.1.1 Local Self-Addressing

In many situations networks that are not connected to any other networks can be created. Networks may stay separated from others because they are not able to (no connectivity or it was not allowed to connect with other networks) or just because they don't need connecting with external elements to obtain the desired resource [THO07] [MOH02].

However, don't have any connection with external networks not necessarily means that no routing will be necessary to allow the communication among the elements that are composing the network [JEO04] [MAS06]. Figure 3.1 presents a situation where a set of networks are already connected and capable of exchanging addressing information and a new network starts to operate with no connectivity with others and needs to allow nodes addresses assignment independent of the existent infrastructure.

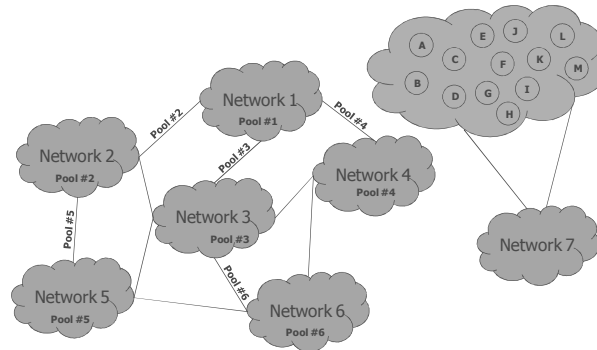


Figure 3.1 – Local Self-Addressing Negotiation

When this situation happens, the network should initially configure its nodes' interfaces and just after that these nodes are able to connect with other nodes. At this point the problem is: How to get an address without any available pool and no servers to control it? To solve this problem is necessary design a solution which allows network's nodes to automatically select the necessary addresses to all their interfaces and verify if any other node in the network is already using this address [TAY04].

As presented in Chapter 2, many solutions were designed to deal with these environments; however, individually they present some drawbacks such as, high configuration delay,

dependency of routing protocols, complex computational processing, etc. Trying to avoid these restrictions was designed a process which integrates some solutions using their benefits in a way to suppress their disadvantages. This solution is composed by three steps and in the end ensures that no conflict of addresses will be present in the network: node identification definition, interfaces' addresses calculation, and conflict detection. Conflicts detection is composed by two operations: active duplicate address detection and network monitoring to search nodes using the same address.

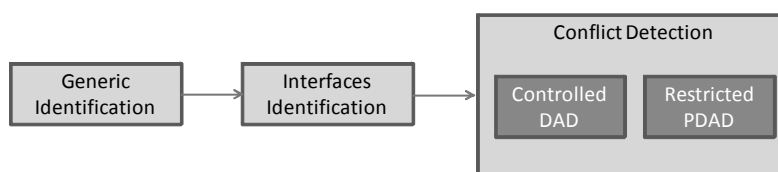


Figure 3.2 – Phases of Local Self-Addressing Negotiation

Self-Addressing phases are demonstrated in Figure 3.2. Initially the node should calculate a generic identification and assign it as the node global identification that might be used during all communications with other nodes within the same network [AHL06] [MOS06]. This generic identification should be used also as a global identifier to inform nodes location and allow the abstraction of the specific localization of the node and its interfaces addressing.

After the definition of the node's generic identification the configuration of the interfaces' addresses should be performed [BAC08]. Local link self-configuration has been very discussed by many authors and a recommendation from IETF, describing how it should be performed can be found in [THO07] [REK96]. In general, this process just considers the MAC address as the information to be used as the IP Address in IPv6 networks.

In the designed solution not only the MAC address is considered to configure nodes' interfaces. The unique identification calculated by each one is also considered during interface identification generation what can improve the address representation and reduce the probability of addresses conflicts. Another question considered by the mechanism is the possibility of having networks running over IPv4. Because of these two possible addressing schemes, the interfaces identification might be able of being represented as an IPv4 or as an IPv6 according to the configuration of the network, also influencing in the conflict probability of the created addresses.

Considering this dual representation, performing the configuration of interfaces based on the generic identification has one clear problem: the possibility of creating conflicts of addresses between two nodes, even being reduced since it is calculated base on unique values. It could happen because the generic identification in a general way might need more information to guarantee its uniqueness that network address could support. One example is the HIP (Host Identity Protocol) [MOS06] that generates a key of 128 bits to ensure the uniqueness and keep the compatibility with IPv6. However, when at most 32 bits can be used as in IPv4, the statistical guarantee of uniqueness offered by HIP is not possible of being kept. In this situation an auxiliary mechanism is necessary to test the network and ensure that, in situations that a conflict might exist, no other node in the network is using the address calculated by the new element.

The third step then takes place in the mechanism: the conflict detection. Conflict detection is composed by two phases that are responsible to check if the address is available in the network. The first phase is responsible by testing the network to verify if any other element already configured is using the address that the node is intended to configure its interface. The most referenced technique considering this restriction is the DAD (Duplicate Address Detection) [VAI02] and its optimizations [WEN05] [WEN03-2] [MUT08]. The basic operation of DAD consists of calculating, in a random way, the interface address and transmits a broadcast packet over the network asking if any other node that is already connected is using a specific address. If so, the new node must recalculate the address and restart the test process until finding a free address. The biggest problem of this mechanism is that it just uses a random address to be used by the node, not considering information already received from the network to define possible addresses to be assigned.

The objective of this phase is adopting the interface identification calculated using unique information to avoid the generation of conflicting addresses together with the Duplicate Address Detection mechanism to verify the uniqueness of the address, since in many situations a reduction from unique identification information to the interface address configuration will be necessary and inevitably conflicts will be created. The probability of conflicts occur is reduced because of the key used to create the address but if it happens they would be detected by a controlled DAD.

A controlled DAD would be a modification in the original technique to reduce the “area” which might be tested. As the probability of having a conflict is diminished in most cases it’s possible not test the whole network as defined by DAD. Applying this reduced test, networks’ resources can be saved and their operation is not affected even if a conflict exists.

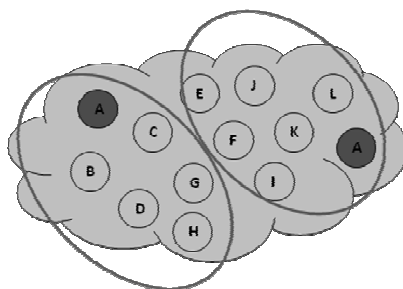


Figure 3.3 – Example of conflict not detected by controlled DAD

This protection to addresses conflicts is possible because the network might appear like partitioned, as presented in Figure 3.3. In this situation the nodes that are in the left part of the network are not communicating with the nodes from the right part and, because of, it doesn’t have information regarding the existence of a node already configured with A address. When they are inquired, in their perspective, this address is available and consequently the new node will be assigned with the same address of A. As the nodes from the left area of the network are just communication among them, the existence of a conflict will not compromise the communication, as explained in Section 2.2.3.1 when Prophet Allocation was described. However, in a future moment, communication between these two areas might be necessary and the network will be affected by this conflict.

The second phase of conflict detection is detects the conflicts that would not be detected by the controlled DAD. During this process PDAD [WEN05] [WEN03-2] algorithm is used to obtain information about conflicts detected in the network. This mechanism evaluates the network control information that is exchanged by the nodes in order to check if some address is duplicated. DAD guarantees that no duplicate address will be attributed to nodes that are running the proposed mechanism, but if a statically configured node comes to the network or the node is very distant and the message expires before reach this, it can just break all the control of the proposed mechanism, since it will use the configured address without any verification. To deal with this kind of situation, PDAD check information from

the network and if a conflict is detected after DAD verification over the network it reinitiates the configuration process, making the network conflict free again. In this context not all available functionalities are necessary of being used by the networks' elements, since most part of the conflicts will be avoided by the unique identification or detected by DAD, so only the basic information such as IP Origin and IP Destination fields are necessary.

3.1.2 Local Address Negotiation

When a network is able to obtain a set of addresses to configure the nodes, it doesn't necessary use a self-addressing mechanism to define the addresses of the interfaces of each node. In this case a structured solution, similar to the one presented in Figure 3.4, can be used to distribute the addresses to new nodes that are arriving in the network and managing their allocation. Figure 3.4 illustrates a network which has received a valid and unique pool of addresses and uses this pool to assign the addresses to all interfaces of the devices located in the network, distributing valid Internet addresses and allowing these elements to be connected to the rest of the network structure. Using this kind of solution it would be easier to keep the correct state of the active nodes and of the network available addresses [ZHO03] and, consequently, a better control of networks' resources.

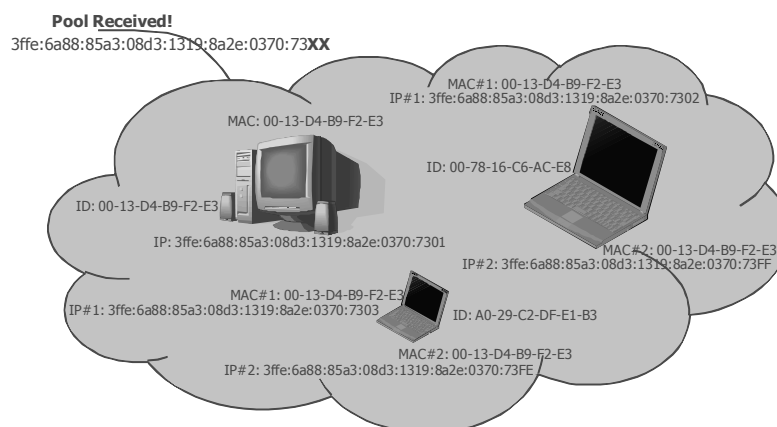


Figure 3.4 – Example of Local Address Negotiation when a pool is obtained

Current Internet solutions don't consider automatic mechanism to be responsible by addresses assignment in mobile and heterogeneous environments. Current proposed solutions are only applicable to relatively small, temporary, and most cases non-heterogeneous dynamic networks and most of the proposals for addressing in dynamic

networks consider that a connection with an external infrastructure is possible in a certain moment. This external support could be provided by mechanisms such as DHCP in a called connected MANET [BAC08]. Another drawback of some of the proposed approaches is that they depend on specific technologies. For instance, there are mechanisms which use information from a specific routing protocol. Such strategy restricts solution's applicability only in scenarios where the network is operating the required routing protocol.

Taking into consideration these restrictions of current technologies and keeping in mind that a structured solution to distribute addresses might be necessary to many networking environments, mainly when more stable technologies are in place, a more structured solution to managing addresses should be considered. This is necessary to dynamically configure the elements of a network considering an available pool of addresses which should be correctly distributed in the network instead of using a self-addressing mechanism as described.

This kind of structured solution has a set of natural benefits, since it's a stateful solution: no risk of allocating duplicated addresses in the network, information about networks' size, etc. On the other hand, some problems also exist when this kind of structure is used. The main part of these problems is caused because of the existence of a single point of failure that is represented by the centralized server responsible by the addresses allocation. To solve the problems involved with structured solutions, a set of mechanism should be created, allowing the network not only to support servers faults but also to take into consideration all the dynamicity that might exist in the network.

To deal with this situation a specific protocol, called SooA [SCH09], was designed. SooA (Self-Organization of Addresses) is a protocol for self-distribution and self-management of network addresses in dynamic networks, which was designed to operate in core networks. The protocol organizes the nodes within a network in a hierarchical structure composed by addressing servers and clients to allow them to be configured.

The servers are nodes responsible for controlling the network addresses management, needing at least one pool of addresses which contains valid and unique addresses within the network. Therefore, servers are responsible for both distribution (allocation) and management (reallocation, returning and recovering) of addresses. Due to the interaction

among the addressing servers, the available addresses are spread in the network and allocated to the nodes which are trying to establish a communication with the network.

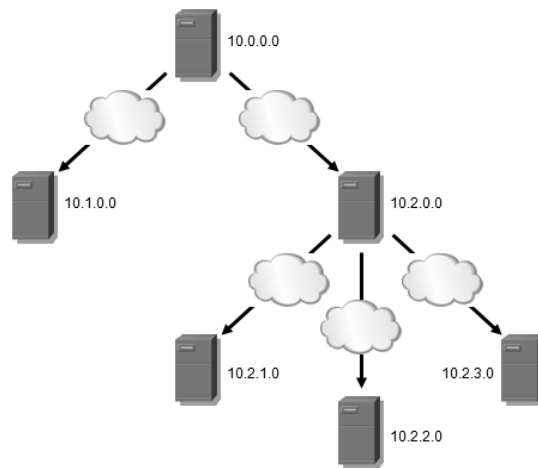


Figure 3.5 – Tree-format servers' hierarchy

When a node assumes a server position in the network, it is provided with a pool of available and unique addresses from an already configured server. This distribution creates a tree-format hierarchy relation among the servers (Figure 3.5). This structure behaves similarly to the DNS (Domain Name System) structure [MOC87] to keep compatibility with current Internet solutions. However, instead of controlling names to addresses translations, the server stores information about the allocation of its address pools and addresses.

The nodes which will be configured only as clients in the network do not have any control over the addressing issues, i.e., distribution and management of addresses. However, they play a fundamental role when enabling the communication between servers and new nodes, which are trying to get connected. When a new node tries to connect to an existing network, it requires configuration information through a broadcasted request (i.e., a valid and unique address). If the new node cannot reach a server directly, due to geographical distance, an already configured client can intermediate the communication between this new node and the addressing server. The client is able to bridge the initial communication for resource negotiation by forwarding the messages from the new node to the server and vice versa. This way, a server will be always reachable through either direct or bridged communication.

As the existing addressing servers are responsible for electing new servers in the network, the bridging functionality gives them additional information that can be used when taking decisions related to it. When a server detects that one of its clients is handling too many connections, this server may decide to offer an opportunity for this client to become a new addressing server in the hierarchy. It is done by providing the client with a pool of available and unique addresses. Its own workload is another situation that can lead a server to decide for deploying a new server. If a server realizes that it has too many clients, it may decide for selecting one of its clients to become a new server within the network. Hence, this new server starts to assume new clients directly.

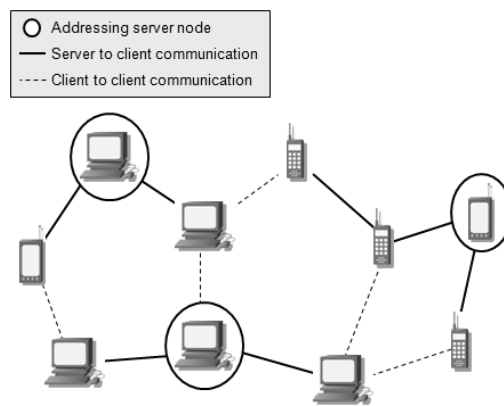


Figure 3.6 – Simple scenario with server and client nodes

Each server keeps the information of the available addresses, allocated addresses, available pools and allocated pools. Such information is stored in allocation tables. Tracing the addresses and pools, the mechanism attempts to avoid the resources loss. If each server is aware of what happens with their resources, the protocol does not need to be supported by Duplicate Address Detection mechanisms or garbage collection techniques to recover unused resources. Figure 3.6 illustrates how the servers and clients may be spread in the network. As mentioned before, the clients do not need to be directly connected with their servers. The network can be composed by clients which are 1-hop distant from their servers and the ones which are n -hops distant from their respective servers (being n greater than 1).

SooA protocol classifies every node into one of the following categories: new node, client or addressing server. A node may also be requested to be a server's backup. Each of them has different functionalities and responsibilities. When a node starts its interaction with an

existing network, it assumes the new node status. If it is successfully connected, it can assume the status of a client or a new server, depending on the offer done by its server. When assuming the role of client, the node is ready to communicate with other nodes within the network. After established as client, a node may also be requested to become a new addressing server in the network. This request comes from its current server that decided to elect a new addressing server within the network due to network/server's overhead.

The server has also the responsibility of selecting its backup nodes, and they must be chosen among its clients. The backups will take proper actions to keep the network integrity when facing situations of server failure. All elements that are part of the protocol's architecture, as well as their interaction, are better described in the following sections.

3.1.2.1 Protocol Elements and Operations

As abovementioned, regarding the addressing issues, SooA classifies the nodes by the roles they play in the network. As illustrated in Figure 3.7, when a node is initiated in the network it assumes the status of new node. From this position, the node can get configured as a client or a new addressing server within the network. Once configured, a client can also be offered to become a new server. This offer comes from its respective server. A client can also be selected to assume the position of server's backup. As a backup, the client will be responsible for monitoring the server activity in the network and, in case of server failure, assumes the addressing management ensuring the resources integrity. Each SooA's element has its respective operation, which is performed by a set of subroutines. Following, the subroutines of all elements are described in details.

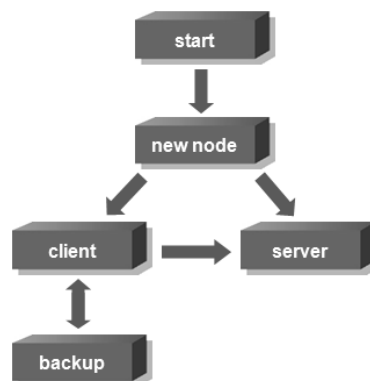


Figure 3.7 – SooA node's status.

3.1.2.2 New Node's Temporary Configuration

When a new node initiates in the network, it needs to configure its interface with temporary information in order to contact an already configured node and negotiate for a valid configuration within the network. This initial configuration can be done by several different approaches, and is valid only during the address negotiation procedure (i.e., negotiation for an address between a new node and an addressing server).

The first possibility for temporary configuration is that the new node creates an address using the self-addressing mechanism already discussed. A second approach for the initial configuration of a new node is the use of a pool of temporary addresses. Before contacting an already configured node, the new node randomly selects an address from a predefined pool and configures its interface. The addresses from this predefined pool must not be used for valid allocation performed by addressing servers, i.e., a server must not be configured with this pool or part of it. After selecting an address, the new node configures its interface and then is able to exchange messages with other configured nodes in the network and, consequently, negotiate for a valid configuration.

Upon concluding successfully the negotiation procedure and obtaining a valid address from a server, the new node stops using the temporary address and configures its interface with the allocated one. Regardless the implemented approach for new nodes temporary configuration, it is not necessary the implementation of DAD procedures. As long SooA's negotiation procedure is performed by unicast message and, when not directly connected to a server, each node is connected and represented by only one configured client, the probability of conflicts between temporary addresses is very low and not significant.

Considering very populated networks, conflicts of temporary addresses may occur if a client intermediates many negotiations simultaneously on behalf of different new nodes, and if this addresses are selected from a very limited pool for temporary configurations. It is easily solved by implementing the client's forwarding table with multiple attributes as the primary key. This way, clients would be able to check if another new node is already connected to it using the same temporary address than a new requester node, and then enforce the latter to select another temporary address.

3.1.2.3 Negotiation Procedure

New nodes in the network need to be allowed to communicate with other elements in the network and establish a connection with an addressing server to be correctly configured. When a node starts in the network, it starts with the status of new node and executes the procedures illustrated in Figure 3.8.

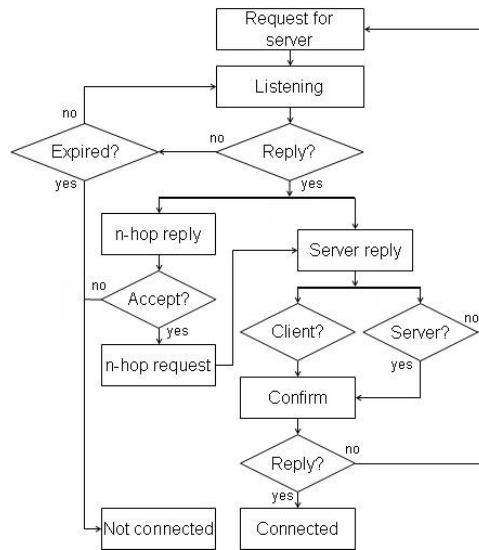


Figure 3.8 – New Node's algorithm.

The first step taken by a new node is to broadcast a message searching for active addressing servers (*Request for Server*). If, after a number of retries, the new node doesn't receive a reply, it assumes that it isn't possible to get configured because there is no active server or clients configured near to it. In this case the node will continue using the self-generated address until receive a server announcement from a server which has started in the network.

A new node may receive a reply to its broadcasted request for server and when this happens, the new node will start a negotiation procedure for getting configuration information, i.e. a valid and unique address from an addressing server. This response can be received from an addressing server or from a client, which is working as a proxy. If the response came from an addressing server it means that the new node is negotiating directly with the server. Otherwise, if the response came from a client, it means that this client is offering the new node an *n-hop* connection to its own server, when $n > 1$. Upon receiving more than one response, the new node will threat then in the following preference order:

- If the new node receives a response from a server, it ignores subsequent responses;
- If the new node receives first a response from a client, and then a response from a server, it abandons the negotiation procedure through the client without sending any cancellation message, and restarts the negotiation directly with the server;
- If the new node receives two or more responses from clients, it chooses the client which is nearer to the server than the other clients. If the first response is from the nearest client, the new node ignores the subsequent responses from clients. Otherwise, if the new node receives a response from the nearest client after the farthest one, and the new node has not completed the first phase of the negotiation, it abandons the current negotiation procedure through the farthest client without sending any cancellation message, and restarts the procedure with the nearest client.

If the reply is from a server (*Server Reply*), the new node starts the procedure to negotiate its address and, consecutively, connection to the network. After receiving the reply from the addressing server, i.e. an offer with an address, the new node sends a confirmation message to this server and waits for a final confirmation to assume the client status (bounded to the server that offered the address).

When the new node receives a reply to its request from a client, it decides if it will try to establish a n-hop connection with a server. If the new node accepts the n-hop connection (no answers from servers), the negotiation procedure is the same than the presented before. The difference is that the client will intermediate the connection between the new node and its server. The new node receives an address offer or an address pool offer from the client's server and executes the same message exchange already explained.

A second situation is that the new node receives an offer to become a new server from the addressing server, instead of the offer to become a client. Differently from the procedure of becoming a client, in this situation the new node receives a response to its request which contains a pool of addresses and not only an address to configure its interface or a timeout event. The second phase of negotiation is the same, where the new node sends a confirmation message to the server and waits until the server replies it with a final confirmation message, allowing it to use the offered resources. The node configures itself

with the first address from the range provided by its father server and uses the other addresses to assign to new nodes or even for creating new addressing servers.

The new node can also opt for not accepting the offer of becoming a new server due to, for example, predefined policies within this node. Upon deciding for refusing the offer for becoming a new server in the network, the new node sends a deny message to the server and waits for an offer to become a client. When receiving the offer to become a client, the new node proceeds as described towards the negotiation's conclusion.

3.1.2.4 Server Element

As the responsible nodes for the addressing management, the servers have the most complex algorithm. Depending on the applicability of the protocol, the selection of the nodes to work as addressing servers in the network could consider their hardware characteristics, e.g. processing and storage capabilities. The decision of creating a new server may come as a result of: a) the necessity of an active server of sharing its workload; b) a node acting as a proxy on behalf of many other nodes; and/or c) before a server shuts down or leaves the network. A server selects one of its clients and requires it to become a new server.

When starting the server's algorithm (Figure 3.9), the node enters in the *Listening* state where it waits for an interruption. This interruption can be a received message or a protocol's operation. A received message by a server can come from a new node, a client or another server. A message received from a new node can be a request for server or a message regarding a running negotiation for address. The client can send to its server messages mediated on behalf a new node that cannot send them directly through a 1-hop connection. The server also receives from clients the replies regarding a previous announcement. The messages exchanged between servers are related to the maintenance of the servers' hierarchy, i.e. these messages are related to the father-child relationship between the addressing servers.

When receiving a request message from a new node (*Request for Server*), the server checks if it has an available address in its pool to allocate to the new node. If so, the server replies the new node with an offer message which contains the offered address (*Server Offer*). At this time, the server sets the address's status to "reserved" and doesn't use it in other offers. The server waits the response from this new node confirming its intentions on using the offered

address. This waiting time is not a dedicate procedure and the server can execute other procedures while waiting for responses. Upon receiving the confirmation message from the new node, the server sets the address's status to "allocated", which forbids further allocation of this address to another node, and sends a final confirmation message (*Final Confirmation*) to the new node, allowing it to use the offered address. This address is periodically checked and if the node is not operational the address can be redistributed to other nodes.

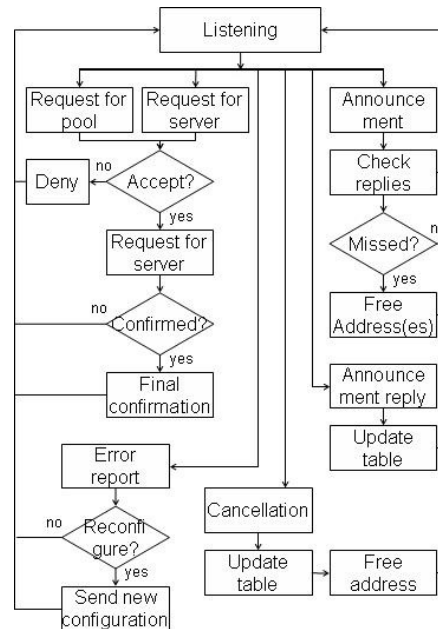


Figure 3.9 – Server’s algorithm.

If the server does not receive a confirmation reply from the new node within a predetermined and configurable period of time, it sets the status of the address back to “free”, which allows its usage in further offers, and the new node is forced to restart the procedure by sending a new request message. The negotiation procedure executed through an n-hop connection is the same than the above explained. The only difference is that the server keeps record of which clients are directly connected or n-hop distant from it. This information can be used in further decision making procedures such as when selecting a client to become a backup or a new server. After sending the final confirmation message, the server assumes the new node is now its client.

In a different situation, the server can send to the new node an offer to become a new server in the network. The only difference in the negotiation procedure is that instead sending an

offer containing only an address, the server sends a portion of its own pool of addresses. The new node can accept or not the offer to become a new server. Upon accepting, the negotiation procedure continues as normal. Otherwise, the new node sends a message to the server denying the offer (*Error Report*), and the server sends a new offer containing only a valid and unique address so that the new node can become its client (*Send new configuration*).

The server can also opt to not accept a new node to connect in the network (*Deny*). When denying connection to a new node, the server sends a deny message to it. The server then updates its cache with the information about the new node identifier and the reason for denying it. Decisions to deny nodes access can be based on, for example, black lists could be created, and shared among the servers, to avoid problems with malicious attacks like (D)DoS (Distributed-Denial of Service).

A second option is receiving messages from already configured clients. In this case three different messages can be received, despite the mediated negotiation messages. These messages are: (a) a reply to a previous announcement done by the server (*Announcement*); (b) a confirmation message in response to an offer to the client become a new server (*Final Confirmation*); and (c) a cancellation message from a client announcing its departure and cancelling the addressing lease (*Cancellation*); Upon receiving a reply to a previous announcement (*Announcement Reply*), the server just updates its allocation table confirming that the client is still active in the network and using the allocated address (*Update Table*). If the server receives a confirmation message, in response to a previously sent offer to become a new server, it continues the negotiation procedure as already explained. When receiving a cancellation message from a client, the server does not reply to the originator, but only assumes that this one is not active anymore and releases the allocated address (*Free Address*).

Regarding the relationship between servers, respecting the hierarchy of father and child servers, the father server can receive a message from one of its child servers requesting for a new pool of addresses (*Request for Pool*). It can occur when the server runs out of addresses and is not able to assume more connections. To increase its pool, the server sends a message to its father server which is the one that can provide a new pool of addresses. Upon receiving such request, the father server checks if it has available resources and sends the new pool of addresses that will be used in parallel with the previously allocated one. The

negotiation procedure continues as already explained, followed by a confirmation message from the requester and the final confirmation message from the father server.

3.1.2.5 Client Element

A client is a configured node in the network. That is, a node using a valid and unique address in the range of a domain where it is connected to. Being configured, the client can exchange data with other clients and servers in the network. Also, a client may be useful when helping new nodes that want to get in touch with an active addressing server. In this section the client's algorithm, illustrated in the Figure 3.10, is explained.

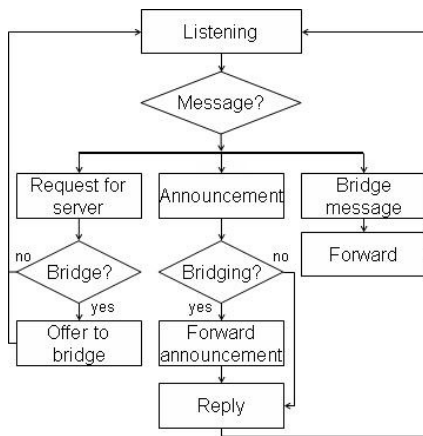


Figure 3.10 – Client's algorithm.

The client, as well as the server, has an initial state of *Listening*. In this state, the client just listens to the mean waiting for broadcast messages. This message can come from its server, from a client or from a new node. From the server, the client can receive the server's periodic announcement (*Announcement*) in order to maintain the leasing for allocated addresses. From other clients, the client can receive replies to the server's announcement (*Bridge message*), which must be forwarded towards the server (*Forward* or *Forward Announcement*). A message received from a new node can be a request for server or subsequent messages in the negotiation procedure that the client is involved.

When receiving a request for server originated by a neighboring new node, the client can reply to the new node with an offer for an *n-hop* connection to a valid server (*Offer to bridge*). In this offer, the client already informs the distance from the new node to the server in number of hops. The new node, if deciding for accepting this mediated connection, will

reply to the client with a confirmation message (*Reply*), and then the client forwards the request to its server and starts to intermediate the negotiation procedure. When receiving a message from the server, the client forwards it to the new node, and vice versa.

A client can also receive announcement messages from its addressing server (*Announcement*). The client then replies this announcement in order to keep the leasing for the allocated address to it (*Reply*). But, before replying to the server, the client verifies if it is currently mediating the communication between its server and other clients. If so, the client firstly forwards the received announcement (*Forward announcement*) and then replies to the server. Consequently, if the client is mediating the communication on behalf of other clients, it will receive the announcement reply from those clients. Upon receiving the replies, the client just forwards them to the server.

Another message the client may receive from its server is an offer to become a new addressing server in the network. When receiving this offer, the client may accept or not such offer. This decision can be made based on pre-defined policies. Denying this offer, the client just sends a deny message to the server. On the other hand, accepting the offer, the client communicates with the server to obtain the information about the address pool that is being offered. If the client accepted the server offer, and the negotiation was successfully concluded, the client turn to the server status and begins to behave as one.

Finally, the client can decide finishing the agreement with its addressing server. If the client is aware about its departure, it sends a message requesting to cancel the addressing lease to the server. After sending this message, the client immediately assumes that it is not connected to the network anymore.

3.1.2.6 Backup Nodes

Failures in highly dynamic networks are more likely than in fixed and structured networks. Therefore, the backup solution added to the proposed approach comes to contribute to the self-management and self-healing of the mechanism as a whole. The backup's objective is to guarantee the integrity of the addressing. It attempts to recover network's address distribution after a critical situation such as a server or a communication link failure.

Each server element in the network has, by default, two backups. The redundancy is to ensure the network's recovery even when facing situations where an addressing server and its backup fail at the same time. This way, a server has a first and a second level backups. Only the first level backup communicates directly with the server. The second level backup only exchanges messages with the first level one in order to be aware of the network status.

Each addressing server in the network is responsible for selecting its backup nodes. The selection procedure could be based on clients' information such as the total time a node is connected to the network, i.e. the node's stability. Having information about the state and stability about its clients, the server can select its backups. The information about the clients can be collected by a higher level protocol that deals with network's configuration parameters, but it is apart of this approach scope.

After being chosen, the backups have to manage their functions. That is, they are responsible for controlling themselves. As a monitoring task, the first level backup sends periodically "hello" messages to its addressing server and waits for a reply. Receiving a reply from the server, the first level backup reports the situation to the second level one. Thus, the second backup is also aware of the network status. When receiving the report from the first backup, the second one also sends a message to the first backup informing that it stills active, hearing and aware of the current network's situation.

As only the first level backup keeps in contact with the second level one, it is responsible for identifying a possible failure in that node. If the second backup does not answer the reports for a determined and critical period of time, the first backup assumes that the second one has failed or that it is no longer reachable. Then, the first backup reports this situation to its server. The server selects another client to assume the second level backup position, and reports its choice to the first level backup. Finally, the first backup starts to communicate with the new second one. The second level back only contacts directly its server, if it identifies that the first backup no longer sends reports, possibly, it failed. Then, the second backup assumes the first level position and the addressing server selects another node to assume the role of the second level backup.

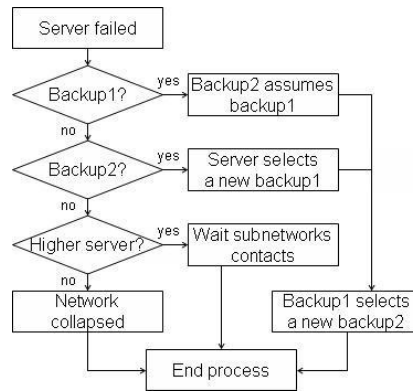


Figure 3.11 – Situations for triggering and/or creating a backup.

The Figure 3.11 illustrates the situations when a backup is triggered. The first level backup assumes that *Server Failed* state when it does not receive replies for its “hello” messages. To be sure about this situation, the first backup may decide for contacting the immediately higher server in the hierarchy, in order to confirm the server’s failure. If the higher level server confirms the situation, then the first backup assumes the server’s position, the second backup assumes the first backup’s position, and a client is chosen to assume the role of the second backup. Similarly, when the second level backup realizes that the server and the first backup can be inactive, it contacts a higher level server in the hierarchy to confirm the situation. Receiving a positive reply, the second backup assumes the server’s position and selects two clients for being its backups. However, if the higher level server, in any of the mentioned situations, does not confirm the server failure to the backups, the backup just assumes that it lost the communication with the server and stops acting as a backup.

When the failed addressing server is the top of the server hierarchy, the backups attempt to confirm the failure with all of the immediately lower level servers which were directly connected to the failed server. Finally, in situations that the server and its two backups failed, the immediately higher level server in the hierarchy identifies such situation and retrieves the control over the allocated address pool to the failed server.

3.1.3 Address Pools Distribution

Networks might configure their elements considering a control policy to assign the addresses over the nodes. This control policy need a set of addresses to distribute and, based on these addresses, is possible to know select and indicate the correct addresses to each element. This

set of addresses is known as a pool of addresses. A pool of addresses is a range between an address x_1 and x_2 which the server may use for configuring its respective clients. The pool only contains addresses which are valid and unique within the network.

This control policy can also enable the pool allocation to be done in an automatic way, despite that these policies has to be configured with default values. However, automatic pools allocation not necessarily will distribute the addresses considering the better options to improve networks communication, since pools distribution will directly impact in routing tables' size what can compromise routing performance.

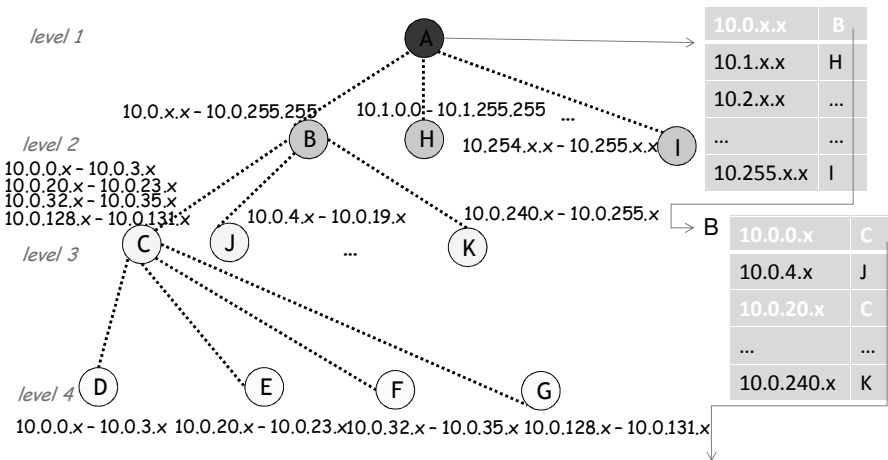


Figure 3.12 – Example of disjoint Pools Allocation

The main objective of an allocation mechanism is then, allocating adjacent subnets to the same request as much as possible, trying to avoid multiple discontinuous segments assigned to segments connected to the same network. For example, many networks request pools of address when start to operate and according to their needs of addresses the pools are allocated (Figure 3.12 and Figure 3.13). Particularly, network C requests four subnets in different moments and if no distribution control mechanism is applied probably four completely disjoint addresses might be obtained as presented in Figure 3.12. This allocation will demand from the intermediate devices four separate entries in their routing tables for each one of the four subnets that are connected with network C in order to reach all possible networks. In general, this is less efficient than having a single entry in the routing tables for all the hosts connected through the same router, as presented in Figure 3.13, where Network C had four subnets allocated with adjacent address pools. Then, it would be

preferable having all subnets allocated in an adjacent way, since they can be represented in routing tables by a single entry.

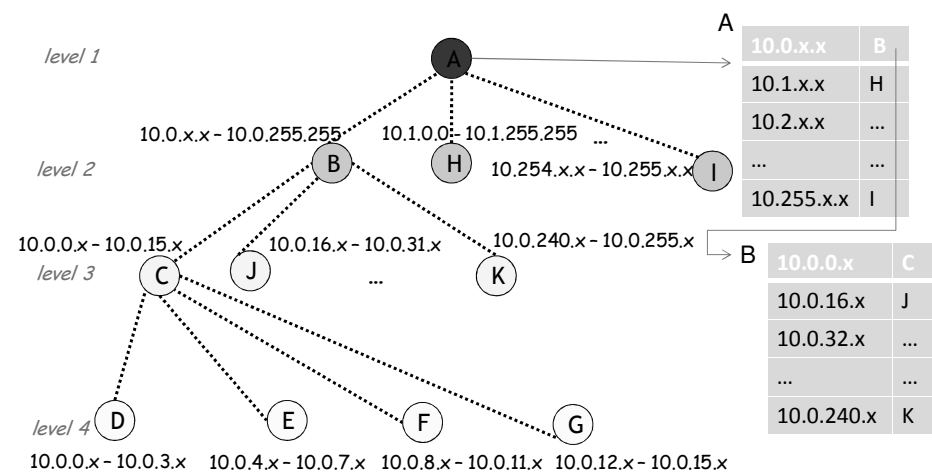


Figure 3.13 – Example of adjacent Pools Allocation

The following descriptions explain the basic concept of the proposed mechanism for distribution of address pools to networks that are able of dynamically requesting/returning addresses according to the modification in its local need of addresses.

3.1.3.1 Mechanism Overview

The proposed mechanism defines two allocation techniques that might be used in order to distribute addresses pools in a highly dynamic environment, allowing the networks to obtain the pools necessary to address all their clients and at the same time aggregate as much as possible the allocation information and, consequently, the routing information.

These mechanisms are planned to improve the pools assignment in a global scale organizing the allocated addresses in a way that they could be distributed following some disciplines and also having the ability of considering policies to control the addresses distribution and adapting to the particularities of each network.

A typical situation to apply the proposed mechanism is the automatic configuration of a set of networks that are connecting with each other and need to obtain a pool of valid addresses to configure their local network elements.

Initially a network or a set of networks is supposed to exist and will be configured with the initial addresses pools. This configuration might not be performed by a negotiation, needing the definition of policies to allow the configuration of these servers without any human intervention. These initial servers can be compared to DNS root servers which present less dynamism and more computational capabilities, and because of it are expected to be responsible by this initial distribution of the addresses using some pre-defined policies.

Having these initial pools distributed among the root servers, new networks that are starting to operate should receive a pool of addresses from an already connected network to be managed by the addressing server of this new network. This distribution is performed in a hierarchical way, but there are no guarantees that no more addresses will be necessary to the networks, needing an allocation mechanism that improve and facilitate this allocation.

This distribution of the addresses can be performed following many different strategies and according to the selected algorithm the results might be different to the allocation itself and to the influences that this addressing mechanism will have in different levels. The most important of these influences is in the routing level, since the allocation will influence in the construction and, consequently, in the structure of the routing tables, so a bad structured allocation can badly influence the routing mechanism by the increase in the amount of information stored in the routing tables.

To deal with this problem and allow a completely dynamic distribution of pools two approaches are proposed in this work. Following these approaches the networks are able of allocating their pools to new networks in a more structured way and then avoid problems in the routing perspective described in the previous section.

An example of the operation of this mechanism is when a new network starts to operate and needs a valid pool of addresses to configure the network elements that are composing it. Using the current technologies it would be necessary configuring all the servers with their specific pools of addresses and when the network need more addresses the administrator must obtain more addresses and configure them. On the other hand new nodes just cannot be added using valid IP addresses. Using the proposed mechanism the networks are able of requesting pools dynamically according to their demands, adapting to the particular needs that each network can find during their operation.

When a new network (a spontaneously network created according to users' demands) starts to operate it must request a pool of addresses to an already connected network that have available pools. This network will then allocate one or more pools to the new one and make a pre-reservation of other pools that might have the preference of being allocated by this network instead of other networks, grouping the addresses allocated to all the network and improving the way that addresses will influence in the other levels of the communication.

3.1.3.2 Pools Allocation Mechanisms

Two approaches were defined in order to allow a better adaptation of allocation mechanism to the diversity of characteristics of connecting networks. These approaches are based on two basic structures that are intended to support a diversity of situations, from stable networks, that can rigidly control their needs of addresses, to high dynamic networks, where this control is not possible. In the first situation a consecutive allocation might suffice but in the second moment probably an allocation that is possible of spreading the addresses might adapt in a better way to the dynamic aspects of the networks.

Independent of the used approach two basic allocation techniques should be considered in pools allocation: fixed and variable pools size. These two techniques can also be used together, defining, for example, part of the allocation using a fixed size and other parts reserved to use a variable size. But only differentiate the way that the pools are supposed to be allocated is not enough to solve the problems of allocation and their effects in address distribution and in other network mechanisms like routing.

Considering the questions already exposed, it's easy to note that only allocate the pools is not enough to allow the network to operate in an optimized way. Allocate pools without any special control can solve the dynamic allocation problem but it brings us another problem: routing table would increase, since to each new pool a new entry would be necessary to reach the nodes allocated with the new range of addresses. To control this allocation and avoid the routing problem two approaches were planned and are described following.

3.1.3.2.1 Consecutive allocation with Neighbor Reservation

The first approach is based on a reservation of neighbors' addresses pools and can be applied with both, a fixed or a variable pools size. In this initial approach, at the moment of

the allocation of a pool, a reservation of a set of pools that are consecutive to the allocated one might, indicating that the “owner” of the allocated pool have the preference to use the reserved ones. The amount of addresses that will be reserved to each pool allocated might be different and could be defined using a set of policies that will evaluate the stability of the network and its possible increase.

This reservation could be done, for example, modifying the priority of the sequential pools (that were not leased) of the allocated one, to a lower priority to be allocated. Figure 3.14 presents an example of this mechanism: suppose that the first pool was allocated for a new network. The defined policies verify that, considering its characteristics two pools of reservation might be enough to future clients that could connect in the network. Based on that, the two subsequent pools, not leased yet, are marked with a lower priority to be allocated for a different device, indicating the preference to be used to “extend” the pool used by the new network. The number of sequential pools allocated when a request is received can be determined by the monitoring of the environment, predicting in a dynamic way the demand of each subnet. After the allocation of this initial pool a second network might also request a pool. To the second network the same verification to define the size of the reservation will be performed and only one pool will be necessary. After allocate the pool and reserve the next one the network will way for new requests and will continue the same mechanism until the end of all available pools. Only when no more available pools with the standard priority exist the reserved pools will be used by new requests.

leased	not leased with low priority	not leased with low priority	leased	not leased with low priority	not leased with standard priority	not leased with standard priority	not leased with standard priority	...
--------	------------------------------------	------------------------------------	--------	------------------------------------	--	--	--	-----

Figure 3.14 – Set of Pools with priority and preserving allocation of adjacent pools for the same device

This mechanism facilitates pools there are adjacent to the already allocated one, been allocated when the same network requests more pools later. This control intends to avoid the increase in the routing table, as described as one of the biggest problems of no controlled allocation. The advantage of this mechanism is that pools not leased yet and with the standard priority are organized in a sequential order, which facilitates that the routers routing table do not increase.

3.1.3.2.2 Spread Allocation with binary division based reservation

Another approach that can be used as a focus is the situation where the pool is divided in two different parts every time an allocation should be done. Dividing an existing pool in two new ones, each network would initially have half of the available addresses of this pool. Performing this allocation, the network tries to allocate the pools as far as possible and avoid that for example, a bad calculation of the amount of reserved addresses obligate the network to allocate a non consecutive pool. The following description explains the general mechanism and how it avoids the increase of routing tables.

Initially it is necessary to divide the original pool that will be used to distribute addresses to other networks in a set of sub-pools. Considering that a server has a pool X with size z , this pool will be divided in n sub-pools with the same size z/n (or similar sub-pools sizes). This mechanism could be also applied with a variable pools size allocation, but with a fixed allocation the management of the pools is easier, so the description will be based on this allocation structure to facilitate the understanding.

When the first request for pool is received by the server, it allocates the first sub-pool and considers that all addresses that follow the allocated one are pre-reserved to the same network for future allocations (Figure 3.15(a)).

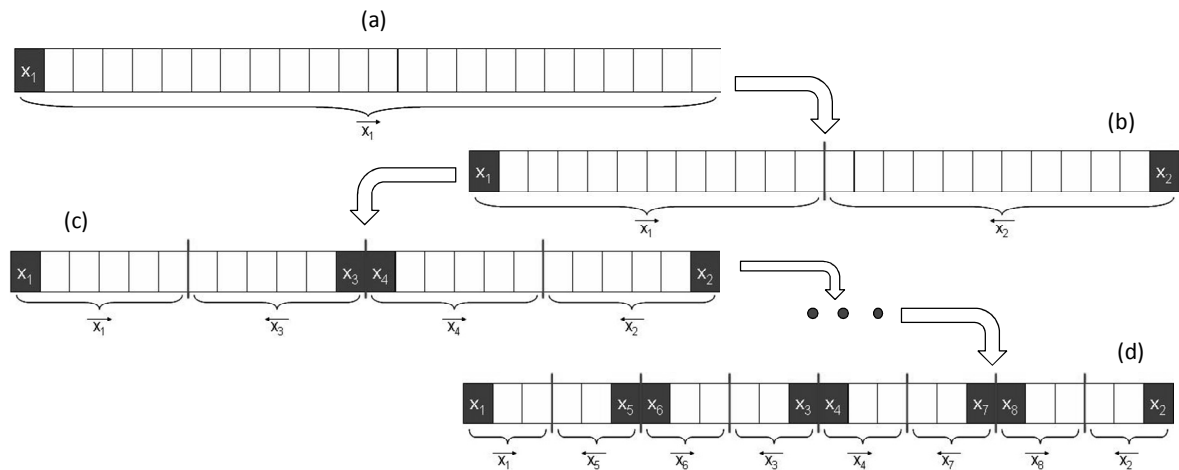


Figure 3.15 – Example of allocation sequence using Spread Allocation with binary division discipline

If new pools are requested by the first network the new pools should use the pools pre-reserved to the allocated one (x_1). As all pools are pre-reserved to x_1 the new pools will be allocated sequentially to right until use all available addresses. When a second request for

pool is received, the space reserved to x_1 will be divided in two equal parts and allocates the farthest sub-pool from x_1 to the new network x_2 (Figure 3.15(b)). This way, the allocation to the network x_2 will grow sequentially towards x_1 reserved space (i.e. to left).

The main idea of this approach is to make the sub-pools allocation, for different networks, grows towards each one. Therefore, it makes easy the sub-pools aggregation when allocating them to the same server, and also will allow a “free” growing to the allocated space to a determined server, avoiding the need of the verification of the characteristics of the network to calculate the amount of addresses to be reserved. Every time that a request for pool is received, a part of the pool that is already reserved to a specific network is divided in two equal parts and each one will have half of the available addresses pre-reserved to them.

Continuing the same example, when a third request is received from a new network, the area reserved to x_1 is divided in two equal parts and the farthest sub-pool from x_1 , considering only the divided area, is allocate to x_3 (Figure 3.15(c)). This way, the allocation to the third child will grow towards x_1 . Now, the allocation to x_2 and x_3 are “growing” to the same direction, left. Therefore, when receiving the fourth pool request from network x_4 , the area reserved to x_2 will be divided in two equal parts and the farthest sub-pool from x_2 allocated to x_4 . This way, the allocation to x_4 will grow towards x_2 (Figure 3.15(c)).

While receiving pool request messages from new networks, the server keeps the same methodology for the allocation. That is, the server chooses an area that is reserved to a child, considering pre-defined policies such as the largest will be split, and divides it in two equal parts. Next, the server allocates the farthest sub-pool from the child the area was reserved for in order to the new child allocation grows towards the allocation to the already existing child. The probably future of the pool’s division is illustrated in the Figure 3.15(d).

In order to add flexibility to this mechanism, it is possible that the area limiter (red line in the figures) is moveable. It permits a network allocates more sub-pools than its initially reserved area. However, considering the structure illustrated in the Figure 3.16(a), network x_1 only can allocate more sub-pools if network x_3 is not using all the sub-pools reserved to it, i.e. those that are part of network x_3 ’s pre-reserved area.

Following the same methodology already presented, sub-pools' allocated to network x_1 grows towards the sub-pools' allocated to network x_3 . In the situation that network x_1 is already using all pre-reserved sub-pools (Figure 3.16(b)), i.e., those that belong to its reserved area, the flexible area limiter allows sub-pools from x_3 's reserved area be allocated to x_1 .

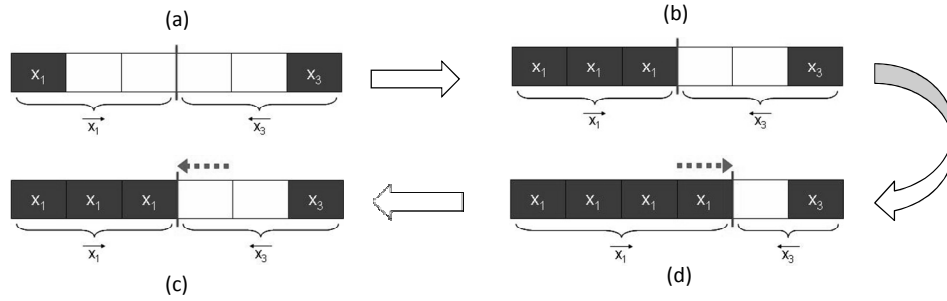


Figure 3.16 – Example of x_1 extra pools allocation

When receiving a request for pool from network x_1 , the server checks the reserved sub-pools from child x_3 's area. If x_3 is not using all its reserved sub-pools, it is possible to reallocate one of them to x_1 , as presented in Figure 3.16(c). This way, network x_3 lends the farthest sub-pool from itself to network x_1 , which would be the last pre-reserved to be used by x_3 if necessary. This methodology guarantees the integrity in the sub-pools aggregation when keeping the sub-pools sequentially allocation. If the network x_1 gives the sub-pool back to its higher server, the area between x_1 and x_3 is recalculated and the limiter comes back to the previous position Figure 3.16(d).

During this process in many situations the pool that should be returned is not necessarily the farthest one. The available addresses can be located in an intermediate pool, for example in the second pool, but if network x_1 gives this pool back to the server it will create a “hole” in the allocation. To avoid this situation, only the farthest pools can be returned to the servers and all networks/nodes that are currently configured with addresses from these pools should be reconfigured and then a reassignment using intermediate pool's addresses should be done, restoring the initial structure and keeping the available pools always together.

3.2 Intra-Domain Configuration Negotiation

In a second moment, after address negotiation, nodes are already connected and have their interfaces configured and their connections correctly established. New negotiations to

optimize their individual operation and how they are supposed to cooperate with each other to improve the network functions will take place then. At this moment a different kind of negotiation should take place: a configuration negotiation is necessary in order to obtain the information of capabilities from the nodes within a domain and based on that information establish the protocols and configurations to improve the communication among nodes.

3.2.1 General Description

In the following descriptions, the basic concepts of the Intra-Domain Configuration are expressed in to present how the mechanism is able of selecting an appropriate configuration to be used by a network such as network interface addresses, signaling protocols for connection establishment, and protocols for mobility management.

Typical situations where Intra-Domain Configuration can be applied are the configuration of functionalities and protocols to be used by network elements in a network without any previous configuration of any element. The configuration of such functionalities and protocols are made possible by the features of Intra-Domain Configuration, which makes the network elements to become “self-aware” of the network operating environment. Some examples where Intra-Domain Configuration can be applied are the negotiation of a codec that a video application must use to improve network resources utilization, the routing protocol to be used and the radio transmission power in order to improve battery lifetime.

To be able to decide the configurations for a specific network environment and adapt these configurations to networks’ dynamicity, it’s necessary to obtain information from network elements. Based on this information regarding the current operating environment, Intra-Domain Configuration can have a complete view of the capabilities of the network and then make a proper decision. This information could consist of, but is clearly not restricted to:

- The protocols supported by the network elements;
- Hardware specification;
- Information about interfaces of network elements;
- Identities of network elements;

- The “priority” of each network element;
- State of each network element, including supported capabilities and enforced policies;
- Optional parameters that can be negotiated.

Following, the elements, general phases and the operational steps are presented and described, indicating how the elements might exchange information and make decisions capable of configuring and adapting Intra-Domain environments automatically.

3.2.2 Phases

Intra-Domain Configuration operates in three basic phases, as illustrated in Figure 3.17: Discovery, Negotiation and Decision. During Discovery each network element disseminates to other network elements in a network its own local operating environment information.

In Negotiation, a network element capable of decision making establishes a local decision for one or more configuration parameters using its own local decision algorithm (Section 3.2.4.3). This decision is disseminated to other elements in the network.

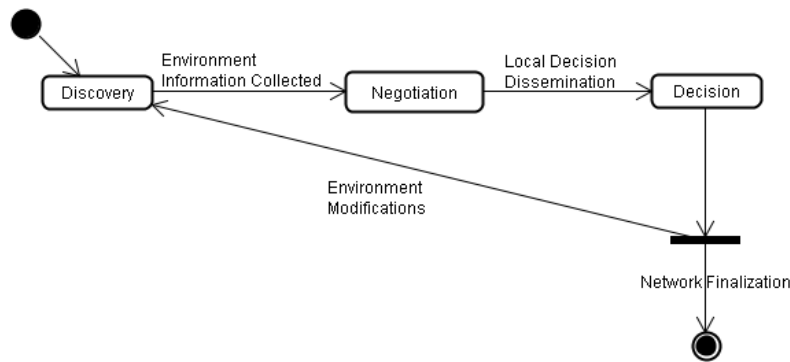


Figure 3.17 – Intra-Domain Phases

Decision is the last phase and is performed after receiving local decisions made by other network elements. This information will be used to execute a final decision algorithm which will select and make one final decision for the network, which is of uttermost importance as each network element may have made different local decisions. This decision process is responsible by adapting to networks dynamicity and generating new decision to represent networks' environment.

To speed up negotiation, especially in less capable devices, Intra-Domain Configuration may rely on the existence of default configurations that could be used in many different types of operating environments. Using this approach networks can initially use this default configuration while the decision process is executed, allowing networks elements to communicate. After obtaining an optimized configuration, it may take place to improve the network performance.

Policies are also other element that should be considered to allow the proposed method to be as automatic as possible. Using policies the mechanism is capable of adapting the environment to new rules that can be defined during the operation of the network elements and represent new desired characteristics to be considered in the network. Decision is one of the possible elements that might be influenced by the policies and should obtain different results depending of networks' objectives.

3.2.3 Elements

Some elements are necessary in the network to allow the correct execution of the negotiation. These elements must execute specific roles in the network allowing information to be disseminated and decisions to be made. These roles are:

- **Network Node:** any element in the network responsible by collecting and disseminating their local configuration to the elements with the ability of executing Decisions. Network Nodes will also wait for decisions to be enforced in order to optimize their configurations;
- **Decision Algorithm Node (DA Node):** is a Network Node with the special capability of collecting information from other elements in the network and calculate Decisions regarding some configuration to be used by the Network Nodes;
- **Coordinator DA Node:** is a DA Node responsible by the control of all DA Nodes present in the network and also by summarizing the Decisions over the network and establish a Final Decision to be used by all the Network Nodes of the network;
- **Backup DA Node:** DA Node selected to store a copy from all information received by the Coordinator DA Node from other DA Nodes in the network. The Backup DA

Node is also responsible by storing all decisions created by the Coordinator DA Node and by assuming its function in case of fails.

Only the definition of the elements which must be used to allow the distribution of the information over the network is not enough to guarantee the correct execution of an automatic process to negotiate Intra-Domain characteristics. The way these elements will cooperate and exchange information is extremely important to the success of the process. Most important interactions among the nodes are executed by the following functions:

- **DA Nodes announcement:** Every time a DA Node connects in a network it might transmit an announcement message to advertise itself in the network. Using this information all DA Nodes in the network know the other DA Nodes with whom they are supposed to communicate in order to make decisions;
- **DA Nodes Selection:** Not necessarily all the nodes capable of execute the DA Node role are necessary to perform this function. A selection of some of these DA Nodes to really work as a DA Node and with this facilitate the negotiation control and reduce the amount of information transmitted in the network is necessary;
- **Local Information Dissemination:** After receiving an announcement from a DA Node the Network Nodes might select one of them and send their Local Information to the selected DA Node. This information will be used by each DA Node during the Local Decision process in order to check the capabilities of the network;
- **Local Decision:** Local Decision process is responsible by perform a decision based on the information obtained by the DA Nodes from the Network Nodes connected to them. Each DA Node makes its Local Decision in an independent way and transmits it to the Coordinator DA Node to give the information to create a Final Decision;
- **Final Decision:** Since the DA Nodes within a same network will make its own Local Decision is necessary finding a compatible Decision to be used in the entire network if possible. The Final Decision is executed in order to define this compatible decision or to verify that it's not possible and them choose a way to split the network;

- **Final Decision Dissemination:** The Final Decision must be enforced in all nodes that are composing the domain. The DA Nodes are responsible by transmitting the configurations established by the Final Decision Process to all Network Nodes in the network ensuring that the decision will be applied.

In the following subsections more details about of the specific processes involved with the decision mechanism are presented.

3.2.3.1 Elements Interaction

Based on the necessary elements and information to be exchanged is possible defining elements interaction to be able of adapting themselves according to networks modifications. Only part of the devices present in a network is expected to participate of the decision process, in this context, the DA Nodes are the elements responsible by participating of the Final Decision Process.

During the announcement process, DA Nodes might collect information necessary to define which one of them should be selected to become the Coordinator DA Node, indicating which one will be responsible by performing the Final Decision during a period of time. To make this selection DA Nodes should inform a score calculated by them which represents the probability of each one of becoming the Coordinator DA Node. This probability can be calculated using any kind of information from the DA Nodes (processing power, memory, interfaces, time active, etc.). The node with the highest probability is automatically selected as the network Coordinator and if more than one node has the same probability, higher operational time is considered. All DA Nodes should create a DA Nodes List to store others DA Nodes probability. Using this information all of them are able possible to know which one of them will be selected as Coordinator, as indicated in Figure 3.18.

The selected DA Node will send a message to the other DA Nodes confirming that it will be Coordinator DA Node and indicating the period of time it will execute this function. This period can be defined based on many parameters and depending of the characteristics of the DA Nodes, a Coordinator could be selected with no expiring time to its coordination if it checks no other DA Nodes are so capable or stable to assume this role.

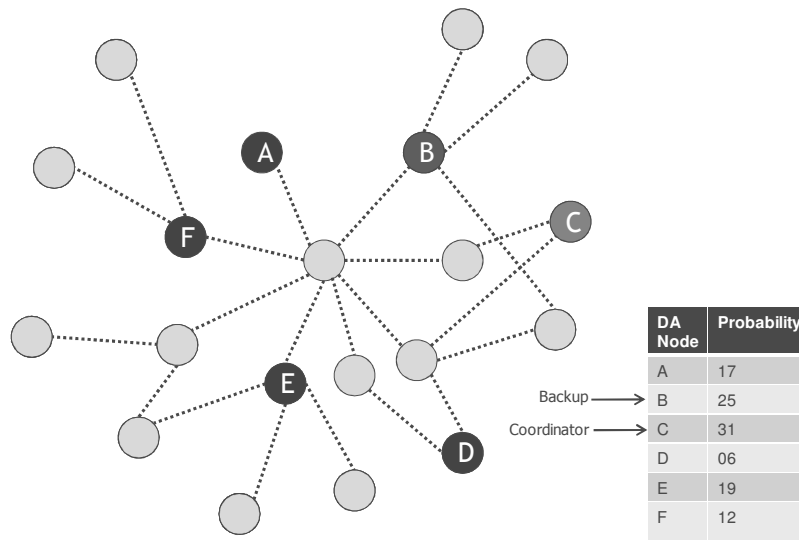


Figure 3.18 – Coordinator Selection

Other responsibility of the Coordinator is selecting a Backup DA Node. The Coordinator selects the Backup based on the probability of being the coordinator of the other DA Nodes in the network: the DA Node with best probability (second highest in the network considering the Coordinator) is chosen.

The communication between Coordinator and Backup is done during all the Coordination Period of the node. This communication is responsible by transmitting a copy of all Decision received by the Coordinator from all the DA Nodes of the Network and also the information of the decisions already made by the Coordinator.

The backup will assume the Coordinator function in two situations: if the Coordinator is down or if the Coordination period has expired. The first case is detected by a connectivity control between the nodes in order to ensure that both nodes are still operating. The second case is detected when the validity informed by the coordinator expires. The Backup will automatically assume the Coordinator function when the Coordination time expires, avoiding the Coordinator selection and all the transmission of the Decisions Database, since all this information is already stored by the Backup.

In the case of failure of the Backup, the Coordinator will select another node to be the Backup using the probabilities already received and stored in the DA Nodes list. This modification will only influences in the Coordinator operations and because of it the other DA Nodes will not participate of the selection of the Backup.

Another function performed only by the Coordinator is the Final Decision process. The Coordinator will be a concentration point to which all the DA Nodes in the Network will transmit their Local Decisions. The Coordinator collects all decisions from the other DA Nodes of the network and creates a decisions base as presented in Figure 3.19. After receiving the decision from all the DA Nodes the Coordinator will perform the Decision Algorithm considering the information located in the decision base order to obtain a Final Decision to be applied in the network. During its Coordination Period only the Coordinator is allowed to execute the Decision Algorithm to calculate the Final Decision to be used.

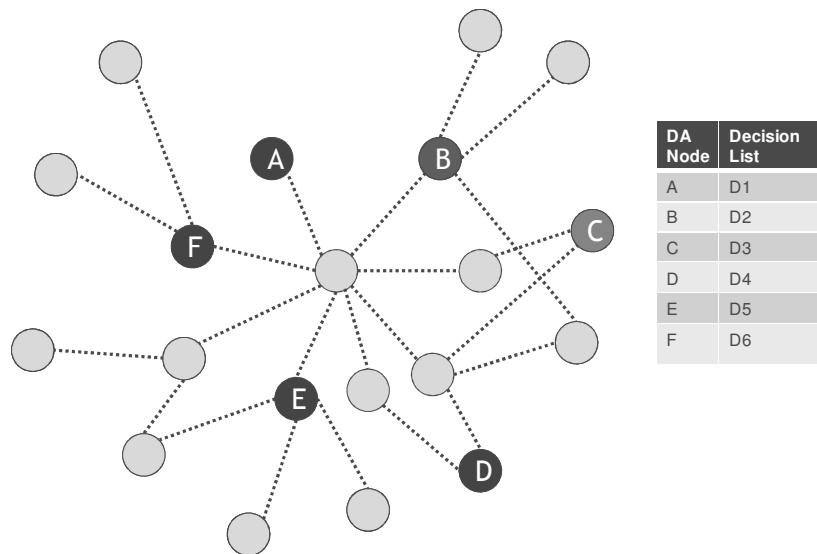


Figure 3.19 – Local Decision aggregation and Final Decision execution

After calculating a Final Decision the Coordinator will disseminate this information to all other DA Nodes located in the domain. The DA Nodes will then transmit the decision to the network nodes located in its Decision Region. When a decision is received from a Network Node the configurations necessary to allow the node to operate in accordance to this decision will be enforced.

3.2.4 Operation

A scenario that can illustrate a complete Intra-Domain Configuration operation is following (see Figure 3.20): a network element N_1 which was initially out of range arrives in a wireless network and its WLAN interface detects a signal from network element N_2 . Through Intra-Domain Configuration, N_1 and N_2 will mutually discover their local operating environment

by transmitting such type of information. N_1 and N_2 could then, also via Intra-Domain Configuration, negotiate, decide and configure what radio transmission power to use.

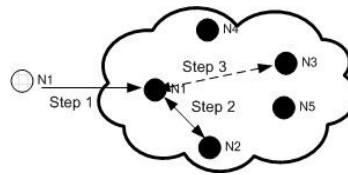


Figure 3.20 – An example of Intra-Domain Configuration operation

After this initial phase a second negotiation and decision “level” may be initiated to acquire and configure an address that allows the entrance of N_1 in the network. Then N_1 will request a service (a videoconference session with node N_3 , for example) and should verify if it is possible to connect directly with the desired service (and which is done by other means than Intra-Domain Configuration). If it’s not possible, it becomes necessary to start another negotiation to configure the routing protocol that will be used to allow the communication with N_3 , and possibly other network elements.

For the protocol negotiation, information about the supported protocols and other parameters regarding the operating environment are exchanged between the network elements and a decision is made according to some pre-configured policy. Examples of such a policy could be, but not limited to, a majority decision, a weighted decision or a default protocol (see further descriptions in Section 3.2.4.3).

The defined phases of the method need specific steps to be correctly executed and achieve the desired result: a configuration or a set of configurations to be used by the nodes after negotiating necessary characteristics. During the rest of this section these specific steps and the involved elements are presented to facilitate the understanding of the whole process.

3.2.4.1 Dissemination of operating environment information

Each network element has the responsibility to transmit to a selected DA Node located in its domain the local information about the operating environment as described in Section 3.2.1. “Local” in this context refers to information specific and delimited by one network element, and which also includes the interfaces and the links by which it is connected in the network.

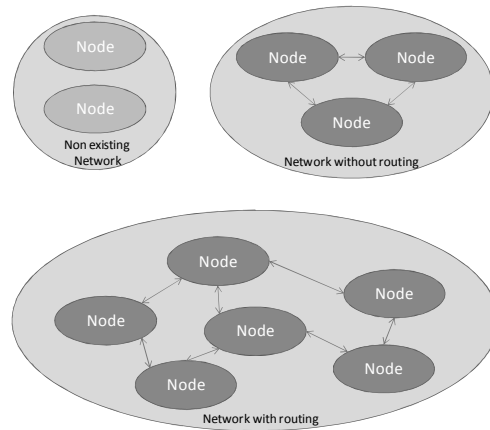


Figure 3.21 – Possible Networks Statuses

Figure 3.21 illustrates some possible Network status. In the simplest case, no network exists and each node would make its own decision if necessary. The next case is where a Network is formed but no routing protocol is being used. This could be the case, for example, where no routing is necessary since all elements can communicate directly. The last case is a fully formed Network, corresponding to a domain in this context, with an established routing protocol, which was chosen as a result of this method.

The distribution of local operating environment information shall ensure that the network element which is responsible to run the Decision Algorithm has up-to-date operating environment information available from all the network elements in a domain.

Due to expected dynamics of future networks, operating environment information distribution shall be done both periodically and possibly also instantly if a change has occurred regarding the operating environment information.

In networks making use of wireless links, connectivity might come and go frequently. This could result in that not all network elements may be reachable from every other network element. In this case, these elements are unable to disseminate their operating environment information and will not take part in the decision process.

Because of these different behaviors depending on what technologies are being applied, the distribution mechanism for distributing operating environment information between network elements can be instantiated in different ways depending of the particular

characteristics of the communication and the supported capabilities of the network technology. Distribution of the local operating environment information can be in different ways directed and limited to save network resources, e.g., only towards the network element(s) running the decision algorithm.

3.2.4.2 Domain Border Detection

The distribution of operating environment information shall generally not reach network elements outside a domain. A domain border can be characterized by a boundary such as between address areas, security constraints, and administrative and/or ownership responsibilities. Thus, the distribution mechanism should generally start by identifying whether by sending information across a certain interface/link implies the crossing of a domain border. Identifying a domain border can be achieved by for instance exchanging address prefix information and/or some kind of network (element) identities. If and when a domain border has been detected, there should generally not be any further exchange of operating environment information across that interface/link.

It should be observed that a network element might belong to several domains at the same time. In such cases, the network element should be properly configured to ensure that operating environment information is not traversing domain borders which are located within the network element itself.

3.2.4.3 Decision Process

One or more network elements within a domain are expected to have the responsibility to run a decision algorithm which ultimately results in the configurations that must be used by the network elements in a domain, based on the capabilities that were collected as part of the distribution of operating environment information. The selection of the configurations will depend of the information disseminated in the network and is not restricted to only one of them, since several negotiations can be performed, each one to a specific characteristic.

Each network element having this responsibility first makes its own (local) decision, using the network element's own decision algorithm. If more than one network element runs the decision algorithm, they must also inform each other about their local decisions. In case where the different network elements running their local decision algorithms have come to

different decisions out from the available set of configuration options, they need to collectively agree on one uniform and final decision. This converged decision can be achieved in different ways, some example are:

- **Majority Decision:** based on the decision made by the majority of the network elements running the decision algorithm. Majority could be so that at least $50\% + 1$ of those network elements have made the same decision, and this is then also agreed to be the decision accounted for the whole domain;
- **Weighted Decision:** the decision accounted for the whole domain is to decide upon a configuration selected by the network element running the decision algorithm having the highest “weight”, depending on its importance in the network. If several network elements have the same weight, and which also is the highest, but their decisions are not uniform, one could then in addition make a majority decision among those (with highest weight) as described in the previous alternative;
- **One consistent, coherent decision is not possible:** If the two decision alternatives described above still wouldn't result in a uniform decision to be accounted for the whole domain, one need to consider alternatives such as splitting the domain, or possibly fall back to a default configuration.

In any case, all network elements being capable of running the local decision algorithm must then use the same logic for calculating the final decision (i.e. this needs to be pre-configured, and/or take a default value). As result of Decision Algorithm, new configurations will be select to be used in the network based on the information received from the network elements that send their local operating environment information. The network elements responsible by the decisions must disseminate this information about the new configuration (Local Decision) to the other DA Nodes present in the network.

After receiving other local decisions made in the domain, the DA Nodes are able to calculate the Final Decision, which will indicate the summarization of all Local Decisions and, consequently, will be sent to all other network elements in the domain as the configuration that must be used.

3.2.4.3.1 Decision Process Example

The main objective of the Decision Process is allowing nodes and networks to adapt themselves to modifications in the environment and optimize their configurations to improve network's performance. These new configurations are obtained by Decisions that will represent the necessary modification in the nodes to operate in a more optimized way considering the new characteristics of the environment.

All decisions are performed by the DA Nodes and are based on Network Nodes configurations or other decisions received from other DA Nodes. In the first case the result will correspond to a Local Decision. Local Decisions will basically represent the capabilities of all Network Nodes located in a certain area of the network and connected with the same DA Node. In the case of other decisions, it will represent a Final Decision, which will try to represent a homogeneous configuration to be used by the network.

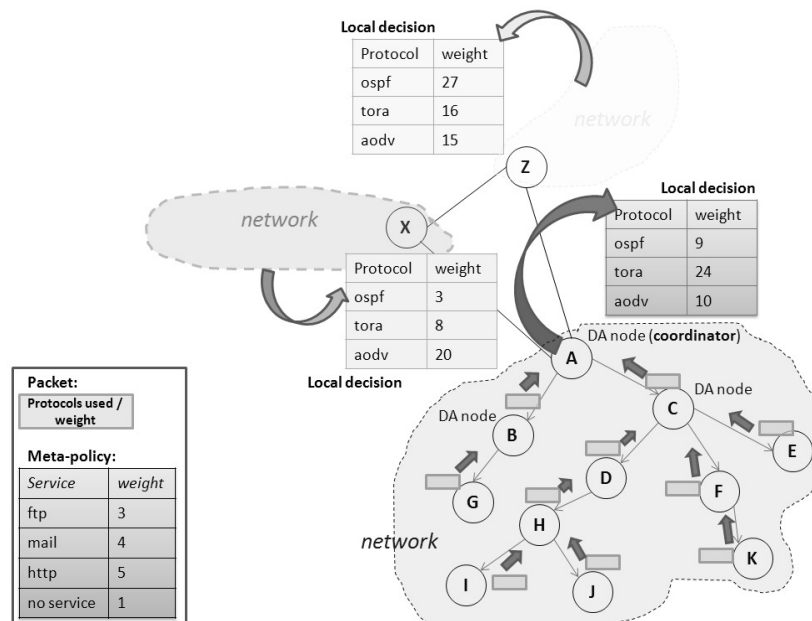


Figure 3.22 –Local Decision Overview

Figure 3.22 presents an overview of Local Information Dissemination and Local Decision Process. Initially, each domain might check its DA Nodes and if not all of them are necessary a selection mechanism can be applied in order to reduce the overhead caused by DA Nodes communication. In a second moment the Coordinator DA node is chosen, what enables the network to start making decisions and define new configuration to optimize

networks' operation. During this process, every single Network Node sends its information to the closest DA node. This creates a set of areas within the domain, having a DA Node responsible by each one, where information will be collected.

Each DA Node will receive messages containing the configuration from the Network Nodes to inform their capabilities. Hence, DA Nodes should calculate the “coverage” of each protocol and send this information to the Coordinator DA node. First possible result is an homogeneous environment where all the nodes are capable of using the same protocols/configurations. Other option is when network's elements have heterogeneous capabilities and no compatible configurations can be used by all of them.

After calculating the Local Decision the DA Nodes must perform a Final Decision. If just one DA Node is present in the network it will perform the Final Decision based on its own Local Decision so, it will be basically a confirmation of the Local Decision. If more than one DA Node is present in the network they must exchange their Local Decisions in order to obtain a complete knowledge of the network and performing the Final Decision.

Having all Local Decisions stores, it is possible to make a Final Decision. All DA Nodes are potentially capable of making a Final Decision, however, this function will be performed only by the coordinator, which is the only one responsible by making this decision. The Final Decision will consider all “summaries” (i.e., Local Decision) received and then try to find a homogeneous solution to be used in the entire network. Other parameters could be considered, but in a general way homogeneity is the main requirement to allow an easier environment to operate, in the routing perspective.

In Figure 3.23, two DA Nodes communicating with the Coordinator DA Node. The Coordinator will receive the information of each DA Node containing a summarization of Network Nodes' (Nodes A, B and C) capabilities. These information will be used by the Coordinator DA node in order to verify if exists a totally homogeneous solution. According to the protocol that is being checked, the Coordinator can answer this question only by looking if the percentage of this protocol in all areas of the network is of 100%. If it is, the protocol can be used by all the nodes without causing any trouble or interfere in the network performance, since no translations would be necessary.

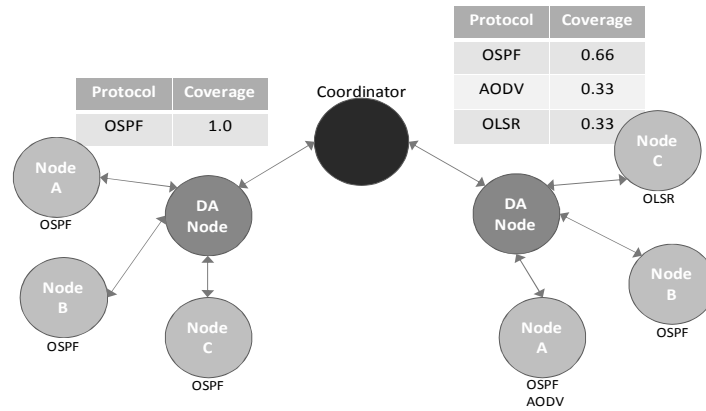


Figure 3.23 – Example of Local Decision Dissemination

However, it's expected that will not be possible to find a homogeneous protocol to be used in the entire network. The Coordinator must be capable of verifying when it's necessary to divide the network in different domains and the protocols that must be used in each of them. If it's not possible to define the protocol to a domain a “delegation” of the final decision must be sent to the DA Nodes present in the domain and the Decision process will be restarted in that specific area.

3.2.5 Operation Example

Figure 3.24 illustrates a home area network with five different devices, as well as another device residing in the access network on the other side of a domain border. All devices in this example run the proposed method and three of these devices in the home area network also run the Decision Algorithm (DA). Each device can support several communication interfaces and consequently several routing protocols. It is also assumed that the devices running the DA have equal weight, and is in this case going for a Majority Decision.

Via the negotiation mechanism, the domain border will be identified, e.g. by transmitting some information that identified network or node/device identities. In this case, this will result in that no further information will be transmitted between the access and the broadband router, and that no routing protocol will be used across the link. However, an alternative scenario would have been to decide that across that domain border link, BGP could have been chosen as the routing protocol to use between the two domains.

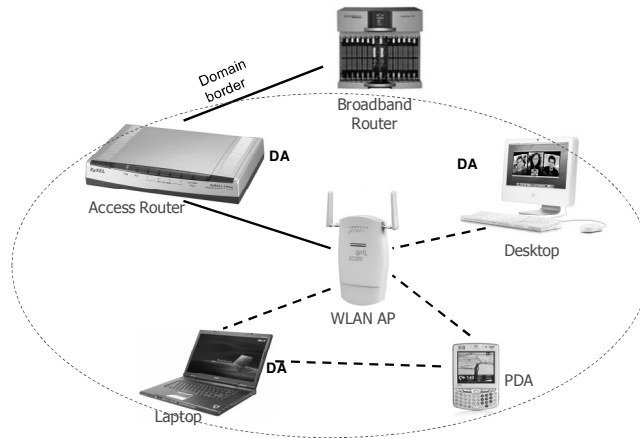


Figure 3.24 – Home area network using Intra-Domain Configuration to select a routing protocol

For the home area network the disseminated information about supported and used interfaces, as well as what routing protocols that are supported by each of the devices, are used during the execution of the Decision Algorithm. In this case Intra-Domain Configuration Negotiation is being used for negotiating a routing protocol, and many possible decisions can be made:

- No routing is needed. Basic connectivity is enough;
- Routing is needed and a suitable intra-domain protocol is chosen;
- A new protocol is chosen to replace the current one
- An inter-domain routing protocol shall be used (occurs at domain borders);
- Network element is not willing to route. Forward packets to a peer that can route.

Based on these basic possibilities, the access router and the desktop makes Local Decisions and indicates that an Ad-hoc routing protocol shall be used. On the other hand, the execution of the DA in the laptop results in that a Bluetooth protocol shall be used to save power. A Majority Decision will be necessary and is taken, selecting the Ad-hoc protocol as the network routing protocol. For reasons of saving battery power, the PDA decides to back off and not use any routing protocol, and will thus act as a host.

3.3 Summary

This chapter has described some key mechanisms, which can be used to support self-managed and self-configured networks. Initially nodes basic configuration of the elements might be executed. To deal with that a set of mechanism responsible by configuring elements interfaces (in a completely autonomous way or using an addressing assignment mechanism) and also to distribute addresses pools among networks.

In a second moment, configuration might deal with not only nodes individual characteristics but also with networks configuration in order to improve nodes communication. By disseminating network operating environment information between network elements, the network elements can negotiate and decide collectively the proper configuration of the network elements defining, for example, different protocol options, addresses, etc.

The mechanisms also supports a dynamic network environment where e.g. network elements may join and leave the network, and which leads to that a previous decision is continuously re-evaluated and which can then further lead to re-configuration of the network elements in a network.

Overall, self-configuring network elements and networks significantly improves on the costs for managing networks, and the present invention can provide an important step towards lowering total cost of ownership. But only configure the Intra-Domains is not enough. Typically, networks need connecting with other networks, and the configuration of this Inter-Domain communication might be performed in an autonomous way to guarantee the dynamic aspects already discussed.

4 Inter-Domain Policies Negotiation

Policies Negotiation is not an easy task to be done in any situation and when considering the application of an automatic mechanism to a dynamic network environment it becomes much more complex to be solved.

Networks need more and more a structure which allows them to negotiate in a dynamic way their agreements and consequently their communication rules [AND06] [PSA06]. This is necessary because with the higher level of content and services distribution together with user's higher interaction with the environment in the future it will be mandatory the establishment of agreements with networks never contacted before and absolutely no rules were defined to regulate this communication [AKH07].

Another advantage of this kind of solution is breaking completely with the current policy establishment and negotiation model that is based on a complete offline and administrative process [DEW00], not taking into consideration the dynamicity that is one of the main characteristics of computer networks operation [FAN99].

Despite the benefits and growing requirement for immediate solutions with support for policy negotiation, it remains seen as a poorly explored segment. Some reason to this are:

1. A lack of a general consensus on how policies should be represented;
2. The negotiation process itself remains a theoretical exercise and no prior experience has been gained;
3. Manager's fear of loss of control and increased unpredictability when automating policy negotiation among Autonomous Systems (AS) and domains.

Trying to solve 1 and 2 a policy negotiation mechanism was designed to be applied in Inter-Domain communications, allowing networks to contact and configure each other in a completely dynamic way instead of the static procedure used today.

4.1 Why dynamic negotiation might be allowed in Inter-Domains Communication?

Negotiation is a complicated process because it might take into consideration diverse characteristics defined by each domain and then try to find an acceptable trade-off among the required characteristics, the offered communication and the price that should be paid by this allowed communication [DEW00]. In the end of this process a SLA (Service Level Agreement) is created indicating the rules that should be obeyed by the involved domains.

Today this SLA is in a general way agreed, either verbally or in writing, by both involved domains before their need by communication to create an agreement that can be used when these domains try to transmit information through each other [DEW00]. The domains store the information of the agreements in a repository and use it to condition the traffic received by/transmitted to the other domain. To change the SLA, normally the responsible by the domains have to contact each other to negotiate the new characteristics and then manually change the defined SLA. These modifications also have restriction of time, not allowing the agreements to be adjusted every day, for example, to fits better to the specific applications that domains' users are running or the desired communication level. This operation can just be done once or twice a year, limiting networks' adaptability [FAN99] [GAO01].

NGI scenarios also impose new requirements since it's expected that a user will use different devices with different capabilities according to the specific situation that they would be. For example, a PC may be used at home or inside an office, a cell phone while they are at the bus or a laptop when traveling. As each one of these devices have different processing and communication capabilities, different applications will also be used in each one. These specific applications will generally require different network transmission characteristics (bandwidth, delay, codification, etc.) what would need modification in users' SLAs and also on networks communication quality, and consequently, domains' agreements.

As discussed in the previous chapter, mobility and the diversity capabilities and transmission technologies in users' devices create a very dynamic environment in which current technologies are not prepared to operate. In this perspective, is very complicate for a network administrator planning the resources allocation considering a precise amount of resources to be offered to all users and based on that to create the necessary agreements with other administrative domains.

Considering all these questions, a dynamic negotiation mechanism is extremely desirable to make possible domains to adapt their available resource according to communications' requirements in a specific moment and not only based on a previously established contract [AKH07]. This dynamic negotiation should be automated and should be able to be executed in a small time scale in opposition to today's manual negotiation in a large time scale and with no restriction of its periodicity.

Domains are expected to negotiate the rules among them depending on their available resources and the requirements established by users' applications. A domain may also be able of advertising unused resources if the network is underutilized or renegotiate existent agreements to adequate them to new networks' characteristics if necessary. To deal with that there should be defined a standard protocol which can be used to negotiating resources for multi-vendors and multi-service providers.

4.2 General Description of the Negotiation Mechanism

Five phases were defined and represent the basic steps necessary to allow the establishment of a new agreement, the negotiation itself and the verification of the correct operation of the defined agreement to adapting it if necessary. The phases are:

- Discovery: find where the desired resource is located;
- Announcement: identify mandatory requirements to the connection of the networks;
- Negotiation: dynamic process to "choose" the requirements necessary by the communication and the desired characteristics to users/services;
- Enforcement: application of the established policies in the networks;

- Monitoring: verification of networks' conditions and adaptation of the negotiated policies to adapting them to any detected modification in the environments.

This negotiation process (Figure 4.1) is initiated when a client tries to find a specific resource (host, service, user, etc.) and realizes that this desired element is not present in its local network. It's necessary then start a discovery process in order to find where this resource is located. Many options are possible to perform this discovery process, using a diversity of mechanism and technologies which can be defined by the users or by the networks according to their goals with the communication.

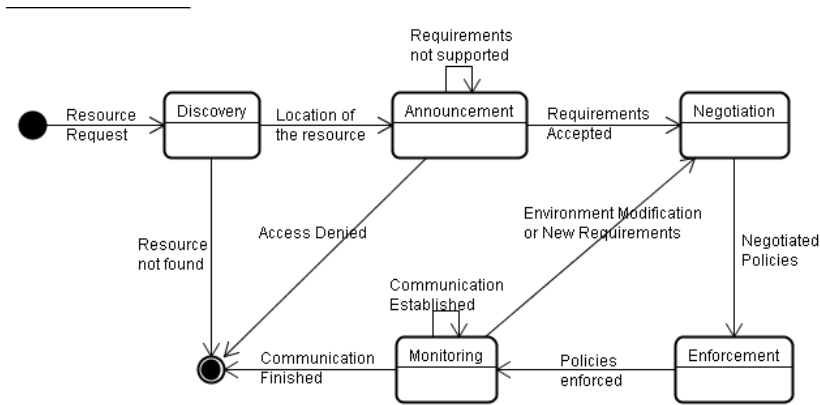


Figure 4.1 – Phases of Policies Negotiation Mechanism

After discovering the location of the desired resource the origin network is able of starting to negotiate the policies necessary to allow the connection of the networks and regulate the communication of them, in a second moment [HAW96]. This negotiation is separated in two main phases: the announcement and the negotiation itself.

The announcement phase represents a pre-negotiation of the basic characteristics necessary by the networks to allow and support the connection with each other. During this phase the networks will just inform the requirements necessary to establish the connection between the nodes keeping a communication quality acceptable by both networks.

In a second moment the negotiation starts to work and assumes the negotiation of the characteristics necessary by the resources that will be used by the networks. Many characteristics might be negotiated, such as QoS constraints, Security level, cryptography algorithm, users' profiles to be supported, etc. [CLA99]. It's important to note that the

proposed negotiation mechanism is not restricted to any specific characteristic and could be used by the characteristics that are already known in the same way of future ones that might be defined according to new services and technologies [NSI09].

After the definition of the characteristics that should be supported by the networks and the negotiation of all policies that might be used, is necessary somehow apply these policies in the hardware elements responsible by the routing in both networks. This configuration of the policies in the network elements is responsibility of Enforcement Phase.

Depending of the elements used in the network this enforcement might be done directly by the elements, only applying the policies using the same language and syntax used during the negotiation process. On the other hand, if the device uses a specific policy technology is necessary to translate the negotiated policies to the desired policy language. This translation can be created to any specific language, what makes the whole process more generic and portable to different network environments.

The last phase of the general mechanism is the Monitoring of the network. During the monitoring both networks should evaluate the already negotiated agreements in order to verify if the rules established with other networks are used in the correct way and also if they are still enough to support the traffic that is generated between the networks.

The monitoring is also responsible by verifying the network status and checking any environment modification that could change the resources available in the networks. When modifications are detected, involved networks returns to Negotiation phase to redefine the necessary policies according to new environment conditions.

These phases are responsible by the general operation of the mechanism and many sub-phases could be added to help them in specific problems, such as authentication and accounting. These auxiliary mechanisms were not described, since the objective is focus on the phases necessary to the negotiation mechanism itself. However, when creating a very flexible and modular mechanism, it would not be a hard work adding new elements.

Each one of these phases will be described in more details during the rest of the proposal, indicating the particularity in the operation of each one.

4.2.1 Discovery

The first phase that is necessary to start the negotiation process is the Discovery (Figure 4.2). Initially network elements should verify if the desired resource is present in the same domain in order to check if only local communication (direct connection or intra-domain routing) would be enough to access the resource. If the resource is located in the same domain of the network element that is requesting it, the discovery has finished and the network element will connect with the resource without needing to negotiate any Inter-Domain policy.

If the resource is not present in the same domain, the network element should verify if a connection with external networks exists to discover the location of the desired resource. Not having any external connection, the network element would not be able to search the service and will consider that it is no accessible.

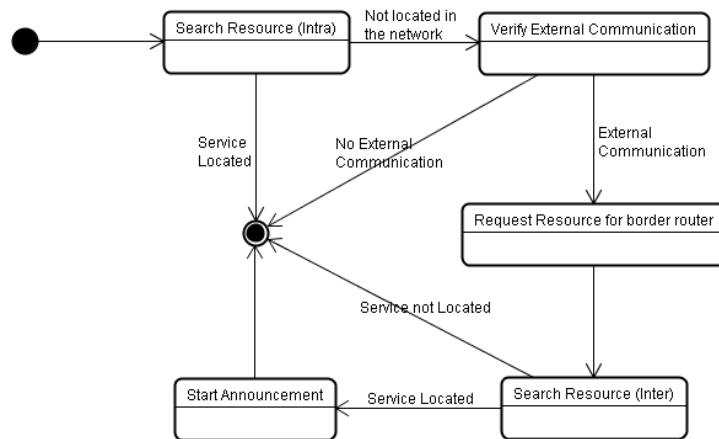


Figure 4.2 – Steps of Discovery Phase

On the other hand, having at least one connection with other networks (external communication) the network element is able of contacting other networks and tries to find the desired resource. Finding the resource the network will start the announcement phase, but if the resource is not found it means that it's impossible connecting with it.

Many solutions can be used during Discovery phase: flooding, Distributed Hash Tables, service directories, etc. However, no specific solution was selected to be used because it would restrict negotiation process. As Discovery doesn't execute activities related with the negotiation itself this flexibility will not compromise negotiation process and allow a bigger diversity of scenario to be considered.

4.2.2 Announcement

When the location of the desired resource is found the announcement phase should take place and establish the initial communication between the involved networks. During this initial contact the networks will start the negotiation of some characteristics that are necessary to the basic communication of them.

Initially the origin network (Figure 4.3) sends a Hello indicating that it wants connecting with the destination network and start to negotiate the characteristics necessary to establish the communication between them. This message might contain the requirements necessary to initiate a negotiation of the basic characteristics that should be supported by both networks.

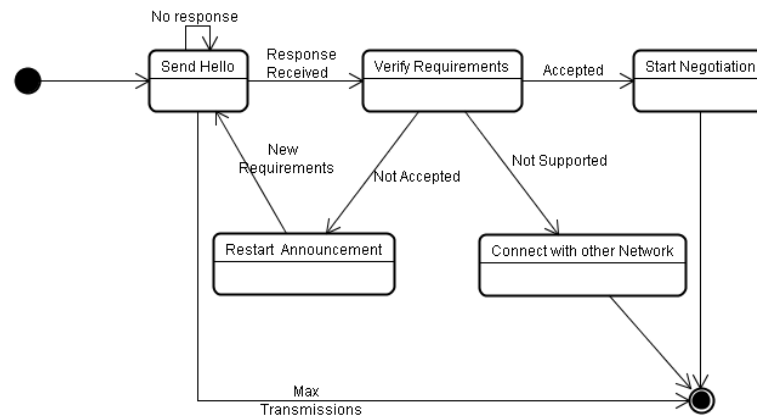


Figure 4.3 – Steps of Announcement Phase in the Origin Network

Announcement's requirements can be separated in two categories: soft requirements and hard requirements. Soft requirements represent the desired requirements of the networks to improve the quality of their communication. However, these requirements can be negotiated in order to adapt what the networks want with what is possible to be offered. Hard requirements, on the other hand, represent characteristics that are really necessary in the connection and the absence of them makes impossible the connection establishment.

After transmitting this Hello message, the origin network will wait for an answer from the destination network indicating if the requirements were accepted and containing destinations' requirements. The origin network might then check if these requirements are supported and if they are accepted start the negotiation. If the requirements are not

supported another network should be contacted and when requirements are not accepted they can be redefined and transmitted again.

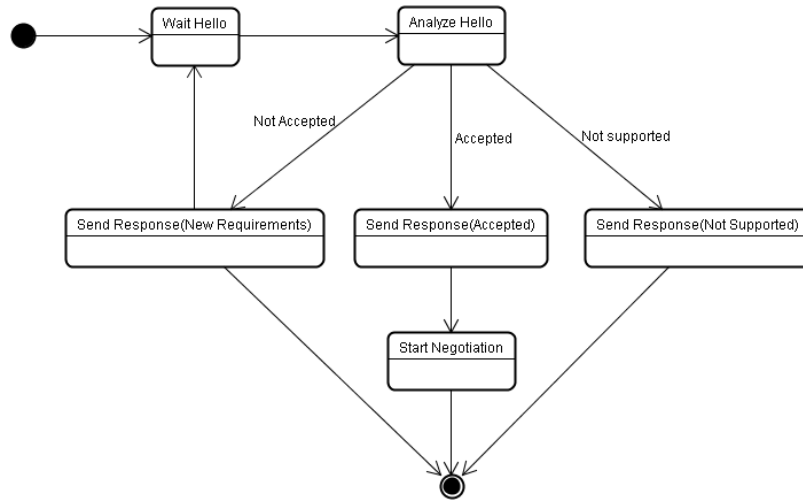


Figure 4.4 – Steps of Announcement Phase in the Destination Network

On the other side of the network, the destination network (Figure 4.4) will proceed basically in the complementary way of the origin network. Initially it's waiting for hello messages which indicate that other networks need to connect and then analyzes if the necessary requirements are supported and also the requirements to be requested to the other network.

If both processes execute without any problem and both networks support the requirements requested by the other network the connection will be established and the negotiation phase will start. On the other hand, if any problem exists during this process the origin network will try to connecting with other network capable of supporting the desired requirements.

4.2.3 Negotiation

The negotiation phase is responsible by the communication between the two involved networks allowing them to discuss the characteristics necessary to the communication and that could not be defined during the announcement phase. Many situations could be considered to characteristics not be defined, for example, negotiable requirements or application requirements that could not be established by the routing layer.

This process is responsible by finding the better configuration according to the requirements and the services that must be shared between the two networks. There's no restriction on

the characteristics that should be supported or on the future addition of new characteristics to be negotiated (service access, routing rule, QoS constraint, security mechanism, etc.).

The originator network (Figure 4.5) should identify the characteristics that are necessary and request then to the destination network trying to obtain the desired communication quality. When the network transmits these requirements, a response is expected indicating if they were accepted. If so, policies are created and enforced in both networks. If not, the origin network evaluates the proposed characteristics to verify if they are sufficient to support the requested resource and then accepts the new values and create the policies or generates a new request to the destination network with a new proposed values.

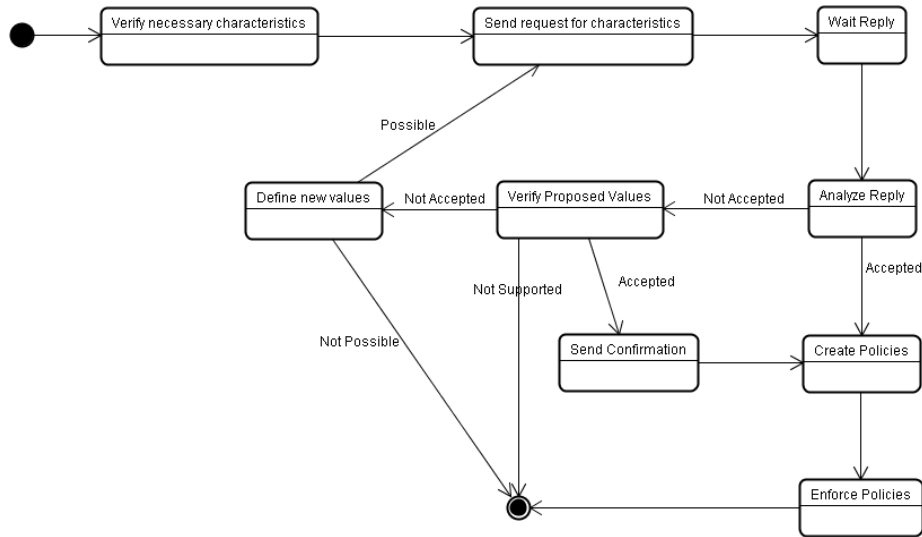


Figure 4.5 – Steps of Negotiation Phase in the Origin Network

Any kind of characteristics might be supported by the negotiation mechanism, allowing it to be adapted according to the particular needs of each network. Some examples of negotiable characteristics are: QoS levels, cryptography algorithm, access control, billing, accounting, etc. These characteristics can also be negotiated in different ways, such as a direct negotiation of each particular characteristic or the negotiation of services profiles responsible by representing the set of characteristics necessary to a particular resource.

On the other hand, the destination network (Figure 4.6) is responsible by evaluating the requested characteristics from the origin network and verifying if they are supported and if the proposed payment is enough. This payment can be representing a financial payment or any other kind of benefit such as access to other networks or to some services.

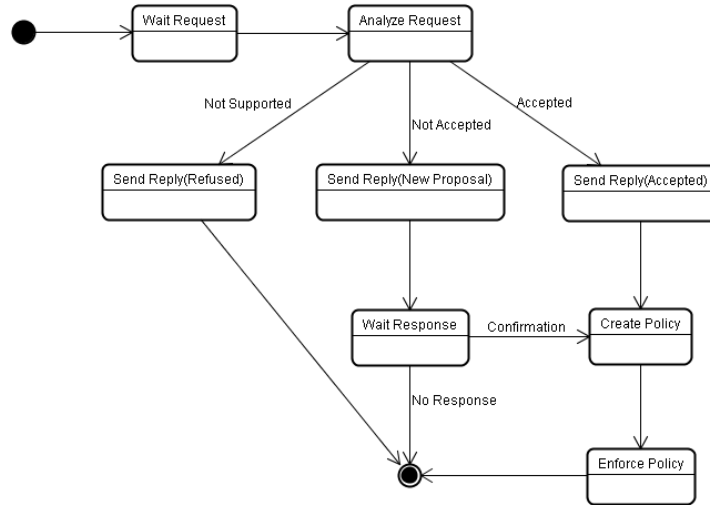


Figure 4.6 – Steps of Negotiation Phase in the Destination Network

Initially the network waits for a request from other networks and when receiving it, analyses the requested resources and their levels. If they are accepted the necessary policies are created and enforced in the environment. If the request is not accepted a new proposal containing a suggestion of resources/levels to be used is transmitted. The network can also not support the request and in this case a message refusing the negotiation is transmitted.

4.2.4 Enforcement

The enforcement (Figure 4.7) is responsible by implementing in the network the set of policies created during the negotiation [SLO01]. This enforcement of policies should initially verify the policies in order to check if any kind of syntax or semantics error exists on the policies. Conflicts verification is also if any conflict is present among all policies [USZ03].

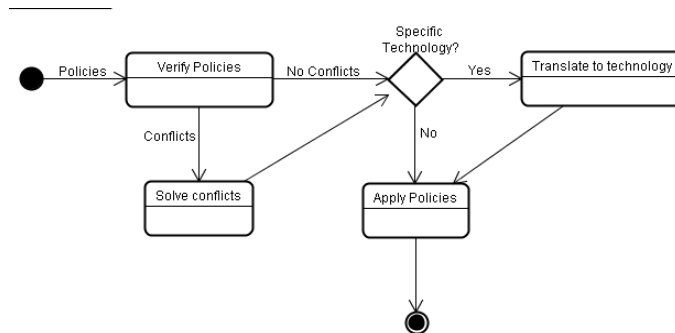


Figure 4.7 – Steps of Enforcement Phase

Existing conflicts they should be solved avoiding the existence of policies with problems in the networks. Having all the policies correct the enforcement phase must verify if a specific technology is in use in the network. If so, the negotiated policies must be translated from the language they were negotiated to the language that is supported in the network [ALA99] [BLU05] [ALA99-2] [YAN03]. As last step, independent of translation, the policies are applied in the network and represent the rules established between the networks [VER02].

4.2.5 Monitoring

The last phase necessary to this automatic negotiation is the monitoring phase. This phase is responsible by verifying the status of the networks and detecting all modifications in the environment. These modifications might represent the need of modifying the policies already negotiated with other networks and can be grouped in two groups:

- **Environment Modifications:** represent modification in the resources available in the networks which changes the support to the communications and obligate the network to adapt the policies that were configured.
- **Access Renegotiation:** represent modification in the resources available in the networks which could be used to improve the connection established with other network and offer a better communication with them.

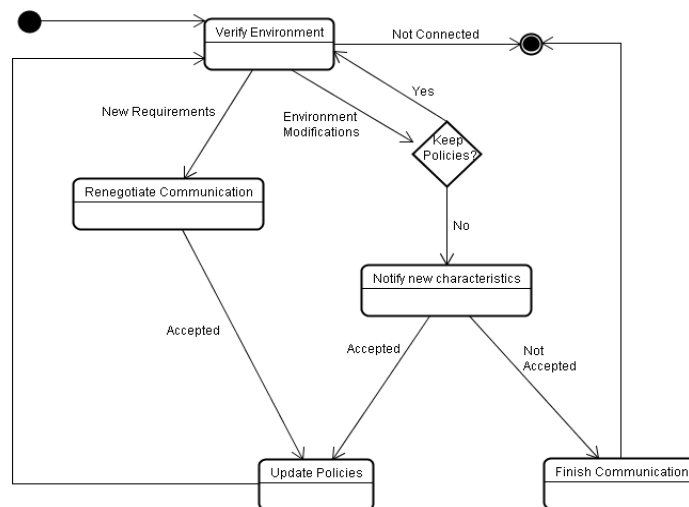


Figure 4.8 – Steps of Monitoring Phase

As presented by Figure 4.8, monitoring phases starts verifying any modification the network. When these modifications are detected the network must check the group of the modification and then start the renegotiation of policies. This renegotiation is based on the same mechanism of the negotiation phase, but the initial notification of the group of the modification and to the new values that were identified to the network resources will restrict the negotiation process.

If new requirements are identified during networks' operation the communication might be renegotiated to consider the new characteristics and create new policies to control them. On the other hand, if modifications on environment characteristics are identified and the existent policies are not capable of controlling networks' resources. These new characteristics are informed and the policies updated if the new levels are still sufficient to support the resources, or the communication is finished if new levels is not capable of keeping communication levels already established.

4.3 Policies Negotiation Protocol

Each one of the defined phases need a deeper specification and understanding to allow the proposed functional steps to become a communication protocol capable of exchanging information and negotiate new connections among the domains. In this section a set of specifications are presented and discussed in order to make possible all the phases to be executed and, consequently, creating a set of specific sub-mechanisms responsible by generating a dynamic agreement between new networks. Some new elements are also necessary to be defined and their interoperability must allow the networks to contact each other and create the necessary policies to establish their connection.

4.3.1 Negotiation Elements and Terminology

As already discussed some new elements are necessary to be deployed in the networks to make possible an automatic negotiation mechanism to be applied. Here necessary elements are presented explaining their specific activities to the policies negotiation scenario is explained, indicating how they are expected to influence and be influenced by new elements. These necessary elements are described next:

- Negotiation: process of agreeing the features necessary to enable the establishment of communication between two domains that have no predefined rule. Negotiation is a two parties end-to-end process that can be classified in two groups:
 - Binding Negotiation: first negotiation executed, responsible by negotiating some core setup negotiation parameters necessary to allow the communication between the involved networks. The result of this negotiation remains active as long as the agreement is necessary to maintain the established communication and any other negotiation necessary to define particular characteristics in the communication might be attached to this negotiation;
 - Requirement negotiation: negotiation performed by specialized technology negotiator to negotiate a particular characteristic of the communication which should be considered during agreement's definition (i.e. QoS, encryption, roaming, etc.);
- Negotiation Profile: description of the characteristics intended to be negotiated by Requesting Domain and the elements which might be instantiated during the negotiation (Technology Negotiators necessary). This description is used by the domains to check their availability of the characteristics/elements to check if all the necessary elements are present in the Domain.
- Negotiator: Element responsible by interacting with other elements in order to create new agreements to allow the communication between networks that are not connected yet. According to their operations two classes of negotiator can be created
 - Contact Negotiator: a negotiator responsible by receiving or sending information to other negotiator located in different networks. This element is the “front-end” of the network and might control the communication with other networks and also distribute the requests done during a negotiation to the specific Technology Negotiator capable of dealing with the requirement/characteristic requested during the negotiation;

- Technology Negotiator: a negotiator specialized in negotiating a specific characteristic. Each characteristic is represented by a set of rules and a structure necessary to define how these policies might be created. These elements are defined by an Ontology, which is used as the basis element to the negotiator knowing what and how it is able of negotiating.

Hosts are able of implementing more than one Technology Negotiator, to that they just have to understand different Ontologies, and consequently they would be able of knowing and negotiating different characteristics.

- Requesting Domain: Domain which has detected the need of communicating with other domains, with whom not previous agreement was established, to allow users to obtain a requested service/information. The requesting domain is responsible by initiating the negotiation process through its Contact Negotiator;
- Destination Domain: Domain in which the resource that was requested by the Requesting Domain is located. The Destination Domain will be the destination of negotiations requests and might evaluate if the requested characteristics should be allowed or not to the Requesting Domain;
- Transit Domain: Intermediate domain located in the path that might be followed by the packets from the Requesting Domain to the Destination Domain. This Transit Domain might also be subject of negotiation depending of the characteristics which have to be negotiated to guarantee the correct communication of the two final Domains (i.e. QoS Constraints);
- Policies: Set of rules used by each domain in order to define which operations are allowed or not to the networks' elements;
- Automatic Policies Control: mechanism responsible by using policies to automatically regulate the operations that are allowed to be performed in the network. This mechanism need to have a specification of the policies, an information repository and a deployment solution to disseminate the rules over the network;

- **Ontology:** description necessary to define a standard to structure policies in a domain, and to describe generic characteristics of an environment and the devices connected to it;
- **Policies Engine (Conflicts resolution and Policies Storage):** responsible by manipulating the policies in a domain and offer a structure to check if the policies are correctly created, and if there is no conflict with other policies. In this process a mechanism solve conflicts and another to store the policies are necessary to guarantee the correct storage and manipulation of the information;

Having this description of the elements involved in the negotiation process and an overview of their interaction, more details regarding elements communication and the negotiation methodology and operational process are discussed in the following sections.

4.3.2 Negotiation Process

A methodology defining the stages of the negotiation procedure is necessary to specify which macro steps are necessary to arrange information which would be used, select the essential elements to each negotiation session and obtain the negotiation results of the requested resources and policies related with each one of them. Figure 4.9 presents an overview of these negotiation generic operations.

First stage is the policies classification. Policies definition is the objective of negotiations; hence the proposed methodology starts by dividing them according to their particular purposes in order to facilitate their management and future identification of candidate policies to be used during the negotiations. With this classification, specialized Technology Negotiators are possible of being create and makes the negotiation methodology more manageable and extensible, since new elements and policies can be implanted, extending domains' capabilities. For example, a domain may have a Technology Negotiator specialized in security requirements, another to work with accounting policies, etc. This way, each specific negotiation may manipulate a different set of policies from the other and possibly a specific ontology might be necessary to dealing with each one.

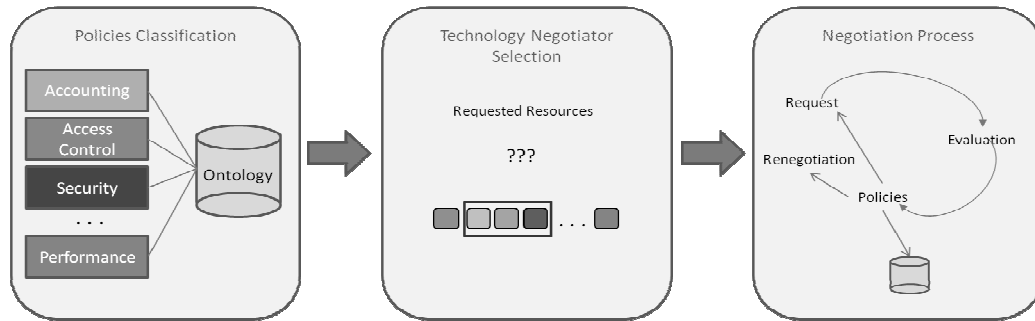


Figure 4.9 – Overview of the Negotiation Process

It is expected that the supported ontologies may contain some intersections which represents common characteristics that will be controlled by the negotiators. However, the ontology used by a negotiator and the policies that are attached to this should not limit the utilization of the same policies structure by other negotiators, allowing negotiators to be as much independent as possible. Figure 4.10 presents an example of this structure: four negotiators are available in the network, and according to the specific negotiator to be used a particular set of policies is considered and in some cases, policies related with a same characteristic (access control and security negotiators have authentication related policies) but each one of them only being part of one negotiator policies base.

Second stage selects the Technology Negotiators necessary to dealing with a negotiation. As policies were classified according their specific objectives, the same must be done with the Technology Negotiators, defining their functionalities and the “functional area” that they are related to, and create a description to each one responsible by identifying them.

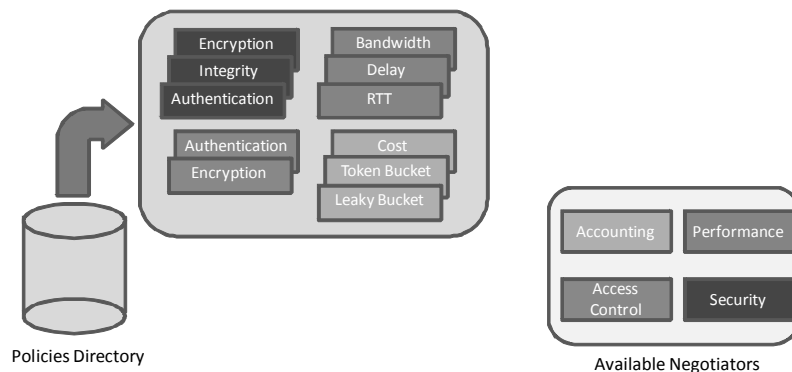


Figure 4.10 – Example of negotiators and their possible policies

As a domain is expected to have several Technology Negotiators, every time a domain wants to start a negotiation with another domain the specific negotiators capable of controlling the necessary characteristics to guarantee the negotiation should be identified. Binding Negotiations are the first ones to be performed and if they're successful, subsequent requirements negotiations should also identify and designate the Technology Negotiators that would be necessary.

Last stage is the negotiation of the policies process will result in the policies that might be used during the communication and would represent the agreement create between the domains and the subsequent specific Requirement Negotiations. The details of the Negotiation are presented in Section 4.3.4.

4.3.3 Protocol Messages

To distribute and control all the information necessary to perform the policies negotiation some messages are necessary to inform the necessary/possible characteristics and create an agreement. Two types of messages can be distinguished: originator's message and destination's messages. The messages are indicated in Table 4.1 and Table 4.2 and explained in more details following sections.

Table 4.1. Policies Negotiation Protocol Messages from Originator Domain.

ID	NAME	DESCRIPTION
0000	NEGOTIATION_START	Initiate a negotiation session
0001	NEGOTIATION_ACCEPT	Indicates the acceptance of a initial negotiation
0002	NEGOTIATION_REJECT	Rejects the proposed agreement and finish the negotiation session
0003	NEGOTIATION_REQUEST	Requests specific characteristics which indicate connections' preferences or modification in a previous agreement to adequate to modifications in the environment/requirements
0004	NEGOTIATION_RESPONSE	Reply for a negotiation request indicating a counter offer.
0005	NEGOTIATION_CLOSE	Tear down the negotiation
0006	RESOURCE_STATE_REQUEST	Requests the information of a resource utilization
0007	RESOURCE_STATE_RESPONSE	Informs a resource utilization
0008	RESOURCE_QUOTATION	Requests the price of a resource to be used
0009	TN_LIST_REQUEST	Requests the list of the offered Technology Negotiators
0010	TN_LIST_RESPONSE	Indicates the offered Technology Negotiators

Table 4.2. Policies Negotiation Protocol Messages from Destination Domain.

ID	NAME	DESCRIPTION
0100	NEGOTIATION_ACCEPT	Indicates the acceptance of a initial negotiation
0101	NEGOTIATION_PARTIAL_ACCEPT	Indicates the acceptance of a initial negotiation
0102	NEGOTIATION_REJECT	Rejects the proposed agreement and finish the negotiation session
0103	NEGOTIATION_REQUEST	Requests specific characteristics which indicate connections' preferences or modification in a previous agreement to adequate to modifications in the environment/requirements
0104	NEGOTIATION_RESPONSE	Reply for a negotiation request indicating a counter offer.
0105	NEGOTIATION_RELEASE	Acknowledges to close and indicate the charging of a session
0106	RESOURCE_STATE_REQUEST	Requests the information of a resource utilization
0107	RESOURCE_STATE_RESPONSE	Informs a resource utilization
0108	RESOURCE_PRICE	Informs the price of a specific resource requested
0109	TN_LIST_REQUEST	Requests the list of the offered Technology Negotiators
0110	TN_LIST_RESPONSE	Indicates the offered Technology Negotiators

4.3.3.1 Data Representation Structure

The information that is used by the messages defined to the policies negotiation protocol is based on a conventional structure of individual fields or (information, values) pairs. However, some of this data uses a hierarchical representation to indicate the elements which might be requested/used. The structure of this information follows the Object Identification organization like the one used by SNMP, where the objects have a unique identification that can be represented as a sequence of names or numbers separated by “.”. Each element in the sequence represents a level in a tree structure and the last element indicates the specific element to be accessed on that level.

Having this structure in mind identification fields of the messages as requirements' IDs, Technology Negotiators' IDs and resources' IDs will be organized through this hierarchical structure to facilitate the representation of the information and also the search for the specific elements that might be considered when receiving a message. This structure also facilitate data organization since more than one information can be represented using the same structure. An example of this flexibility is that NEGOTIATION_REQUEST message is able of indicate the specific requirement it wants to negotiate or a predefined negotiation profile with no modifications in message structure. When the identification is received the Domain will check if it corresponds to a requirement or to a profile and take different action in each particular situation.

4.3.3.2 Messages Description

The messages for the policy negotiation protocol require some common information that is necessary to indicate the basic information used to identify the information and elements involved in the process. The common information to the messages is (Figure 4.11):

- Agreement ID: Indicates a unique identification to the agreement created by the domains when a Binding Negotiation is well succeed;
- Negotiation ID: Indicates a unique identification to a specific characteristics negotiation performed between the domains when a Requirement Negotiation is well succeed. This information should be used in addition to the Agreement ID;
- Message ID: Unique identification of each message sent by a party. This identification allow the involved parties to take track of the operations performed during a negotiation;
- Response ID: It includes the Message ID of the message to which this message is a response. By examining this parameter a party is able to determine whether it received a correct response from the other party;
- Expiration time: The time period that one party is willing to wait for the other party to respond. If this period elapses then the protocol terminates by issuing a rejection of the communication.

<i>Agreement ID</i>	<i>Negotiation ID</i>	<i>Message ID</i>	<i>Response ID</i>	<i>Expiration Time</i>
---------------------	-----------------------	-------------------	--------------------	------------------------

Figure 4.11 – Messages Header Information

Following, the defined messages are presented and the necessary information to each one of them is discussed.

4.3.3.2.1 NEGOTIATION_START Message

The NEGOTIATION_START message is the first message that must be transmitted during announcement phase by a Requesting Domain to initiate the negotiation process. As presented in Figure 4.12, this message contains a list of mandatory requirements, represented

by unique identifiers used to indicate the requirements that are necessary to be supported by the Binding Negotiation, and the Binding Negotiation expiration time. If the value of this expiration time is zero it indicates that the negotiation doesn't expire and is only finished when requested.

<i>Header</i>	<i>Mandatory requirements</i>	<i>Binding Negotiation Expiration</i>	<i>Specific Requirements</i>
---------------	-------------------------------	---------------------------------------	------------------------------

Figure 4.12 – NEGOTIATION_START Message Information

As optional information NEGOTIATION_START message can have an initial list of specific requirements, which represent the specific requirements recommended improving communication quality. Instead of being expected to be negotiated during negotiation phase, these requirements can be transmitted to speed up the negotiation process.

This initial contact is executed by the involved domains and, consequently, might perform a Binding Negotiation and as the result will indicate the Agreement ID of the negotiation if it's succeed or will indicate that the negotiation has failed.

4.3.3.2.2 NEGOTIATION_ACCEPT Message

The NEGOTIATION_ACCEPT message indicates that the requested characteristics by a domain through a requested negotiation were accepted by a Domain and the policies necessary to indicate the agreement will be created to regulate domains communication. As presented in Figure 4.13, this message contains the negotiation expiration period and can also contain (optional information) a list of the accepted requirements to indicate the specific requirements which were already accepted during the binding negotiation.

<i>Header</i>	<i>Negotiation Expiration</i>	<i>Accepted Requirements</i>
---------------	-------------------------------	------------------------------

Figure 4.13 – NEGOTIATION_ACCEPT Message Information

NEGOTIATION_ACCEPT message can be used by the Requesting or by the Destination Domain during announcement or negotiation phases to indicate acceptance of either binding or requirement negotiations. NEGOTIATION_ACCEPT indicates to which agreement it's related and the negotiation when running a requirement negotiation.

4.3.3.2.3 NEGOTIATION_PARTIAL_ACCEPT Message

The NEGOTIATION_PARTIAL_ACCEPT message indicates that only part of the requested characteristics by a domain through a requested negotiation was accepted by the Destination Domain. This message has the same structure of the NEGOTIATION_ACCEPT message, as presented in Figure 4.14.

<i>Header</i>	<i>Negotiation Expiration</i>	<i>Accepted Requirements</i>
---------------	-----------------------------------	----------------------------------

Figure 4.14 – NEGOTIATION_PARTIAL_ACCEPT Message Information

The difference of these messages is that NEGOTIATION_ACCEPT indicates that the binding negotiation was succeeded and some of the requested specific requirements (or all of them) were also accepted. On the other hand NEGOTIATION_PARTIAL_ACCEPT represents that only part of the Binding Requirements was accepted; indicating the supported characteristics.

NEGOTIATION_PARTIAL_ACCEPT message can be used only by the Destination Domain during announcement phase to indicate partial acceptance of the requirements defined in a binding negotiation. As NEGOTIATION_ACCEPT message it also indicates to agreement which is related to the negotiation.

4.3.3.2.4 NEGOTIATION_REJECT Message

NEGOTIATION_REJECT message is used to refuse a request that was received but that cannot be offered to the other domain. To realize that a requested negotiation will be not accepted many negotiation rounds can be done and when one of the domains decides withdraw the negotiation it might reject the requested agreement. NEGOTIATION_REJECT message informs the requirements that weren't supported (Figure 4.15) allowing the Origin Domain to restart the negotiation process avoiding the unavailable requirements if necessary

<i>Header</i>	<i>Unavailable Requirements</i>
---------------	-------------------------------------

Figure 4.15 – NEGOTIATION_REJECT Message Information

Both Requesting and Destination Domain can transmit this message since it's not used only to Binding Negotiations.

4.3.3.2.5 NEGOTIATION_REQUEST Message

This message is transmitted by the Requesting Domain to the Destination Domain to request a particular characteristic or a set of characteristics that might be supported in order to guarantee the correct communication between the domains. This message is able of requesting specific aspect of the communication that are necessary or consider predefined negotiation profiles to summarize a set of standardized characteristics to be considered. NEGOTIATION_REQUEST message (Figure 4.16) is composed by the list of the requested requirements, the respective level that is requested and the payment which is offered to have access to this requirement. The expiration of the negotiation is also indicated.

<i>Header</i>	<i>(Requested requirements, Requirements levels, payment)</i>	<i>Negotiation Expiration</i>
---------------	---	-----------------------------------

Figure 4.16 – NEGOTIATION_REQUEST Message Information

This message can also be transmitted when modifications in an already established agreement are necessary and, in this case, it will be transmitted by the Requesting Domain or by Destination Domain and update the requirements supported by the communication.

As NEGOTIATION_REQUEST message works with specific characteristics negotiation it is only used when a requirement negotiation is taking place during the negotiation phase.

4.3.3.2.6 NEGOTIATION_RESPONSE Message

The NEGOTIATION_RESPONSE message is used to propose revisions on a previously received proposal. This proposal can be received by a NEGOTIATION_REQUEST or by another NEGOTIATION_RESPONSE.

In the first case NEGOTIATION_RESPONSE message is sent by the Destination Domain as a response to a request from the Requesting Domain which is not possible to be accepted. This message includes a new proposal indicating the modifications suggested by the Destination Domain in order to make the negotiation possible of being executed.

The NEGOTIATION_RESPONSE message can also be transmitted in response to a previously received NEGOTIATION_RESPONSE. In this case this message indicates that a proposal was already received trying to change a previous proposal but modifications are still necessary to create a negotiation which could be in accordance to the domains' interests.

<i>Header</i>	<i>(Requested requirements, Proposed requirements levels, price)</i>	<i>Negotiation Expiration</i>
---------------	--	-----------------------------------

Figure 4.17 – NEGOTIATION_RESPONSE Message Information

This message, as shown in Figure 4.17, has the same basic structure of the NEGOTIATION_REQUEST message. However the NEGOTIATION_RESPONSE message only contains the information related with the requirements with were not accepted by the Destination Domains and informs the proposed level and the price to access these requirements.

NEGOTIATION_RESPONSE message is only used during the negotiation phase and the Requesting or the Destination domains are allowed to send it.

4.3.3.2.7 NEGOTIATION_CLOSE Message

When the communication which was negotiated has been executed and the agreement and the following negotiations are no longer necessary the domains might finish the communication and remove the policies which were created to make it possible.

NEGOTIATION_CLOSE message is the responsible by indicating that a communication is not necessary anymore and can be closed, and, consequently all the policies necessary to control the agreement and the requirements negotiations should be removed from the domains to put them in the original state. This message is composed only by the header, since it already contains the Negotiation ID and the Agreement ID, which are the necessary information to finish a negotiation.

The NEGOTIATION_CLOSE message is transmitted only during the monitoring phase, when one or both of the involved domains realizes that the communication is no longer necessary. This message can be sent only by the Requesting Domain.

4.3.3.2.8 NEGOTIATION_RELEASE Message

The NEGOTIATION_RELEASE message acknowledges to a NEGOTIATION_CLOSE message to finish the communication and remove the created policies from the information base. This message (see Figure 4.18) can also contain the cumulative charging information of the whole communication period to inform the Requesting Domain, but the mechanism to calculate this charging is out of the scope of this proposal.

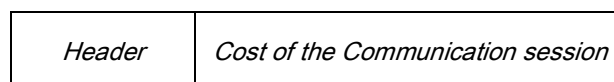


Figure 4.18 – NEGOTIATION_RELEASE Message Information

As NEGOTIATION_RELEASE message is transmitted as a reply for a NEGOTIATION_CLOSE, only the Destination Domain is allowed of sending it during the monitoring phase.

4.3.3.2.9 RESOURCE_STATE_REQUEST Message

This message is sent by the domains to ask the current usage of a specific characteristic or a set of characteristics of the environment. Both involved domains are allowed to send this message in order to check networks' resources state. Information used by this message refers to the list of resources which the utilization must be demanded, as shown in Figure 4.19.

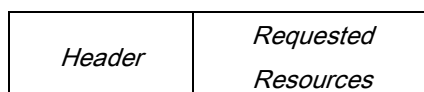


Figure 4.19 – RESOURCE_STATE_REQUEST Message Information

RESOURCE_STATE_REQUEST is a message used to check the state and the availability of networks' resource and because of it can be transmitted during the negotiation or the monitoring phases. In negotiation it's used to check if a resource can be used even before request it. Monitoring uses this message to check modifications in the environment which would lead to renegotiations to adapt the policies to the new network conditions.

4.3.3.2.10 RESOURCE_STATE_RESPONSE Message

RESOURCE_STATE_RESPONSE message is transmitted in a response to a RESOURCE_STATE_REQUEST. The requested domain will collect the necessary

information and sends it to indicate resources usage, as presented by Figure 4.20. This message needs the following information:

<i>Header</i>	<i>(Requested Resource, utilization)</i>
---------------	--

Figure 4.20 – RESOURCE_STATE_RESPONSE Message Information

This message is transmitted during the negotiation or during the monitoring phase, in the same way of the RESOURCE_STATE_REQUEST message.

4.3.3.2.11 RESOURCE_QUOTATION Message

In the same way that a domain can request information about the resources utilization it would be necessary obtaining information about the price to access a specific service/resource from the other domain, since, as in telephony system, for example, users and networks might pay to have access to the resources from other operator. To obtain a quotation of the resource a domain wants to access the Requesting domain might indicate the required resources as indicated in Figure 4.21.

<i>Header</i>	<i>Required Resources to Quote</i>
---------------	--

Figure 4.21 – RESOURCE_QUOTATION Message Information

RESOURCE_QUOTATION message can be transmitted during the negotiation phase and only the Requesting Domain is able of sending this message.

4.3.3.2.12 RESOURCE_PRICE Message

Upon receiving a RESOURCE_QUOTATION message during the negotiation process, a domain determines the price for each resource which the information was requested and returns a list of resource and price pairs inside a RESOURCE_PRICE message, as indicated by Figure 4.22.

<i>Header</i>	<i>(Requested resource, Price)</i>
---------------	------------------------------------

Figure 4.22 – RESOURCE_PRICE Message Information

If the return of price is zero it indicates that the service is free of charge and if a negative value is informed it indicates that the resource is not available to be negotiated in the network. This message is transmitted by a Destination Domain to inform the cost of requested resources from a Requesting Domain.

4.3.3.2.13 TN_LIST_REQUEST Message

TN_LIST_REQUEST messages are transmitted to request a list of the available Technology Negotiators of a domain. This message can request information of all the Technology Negotiators present in the domain or only part of them according to a hierarchical identification of the negotiations, as presented by the necessary information on Figure 4.23.

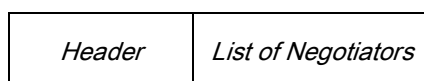


Figure 4.23 – TN_LIST_REQUEST Message Information

As the domains just exchange information after a Binding Negotiation exists, TN_LIST_REQUEST messages are only sent when the domains are executing the operations of the negotiation phase.

4.3.3.2.14 TN_LIST_RESPONSE Message

This message is sent by the domains in response to a TN_LIST_REQUEST message. This message lists all Technology Negotiators supported by a domain, as indicate in Figure 4.24.

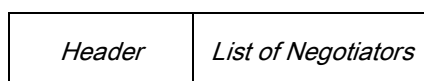


Figure 4.24 – TN_LIST_RESPONSE Message Information

TN_LIST_RESPONSE message also considers a hierarchical identification to indicate the networks' Technology Negotiators which are in accordance with the received request.

4.3.4 Negotiation Process

The negotiation process might take into consideration all the operations necessary to create the policies in networks. As discussed this process is composed by five phases and each one of them should be defined and have the necessary messages and interaction designed.

Considering that the negotiation process might be executed following these phases some assumptions related with this process are necessary to be clarified:

- Negotiation might be executed in rounds: not necessarily the negotiation would be executed directly in just one round. In most cases agreements will not be defined in the first contact and rounds are necessary to haggling between the negotiators while making suggestions until reaching an agreement or aborting the process;
- Negotiation might be composed by sub-negotiations: when requesting a characteristic during the negotiation, a contact negotiator can actually being requesting a set of specific characteristics which might be negotiated individually. These sub-negotiation should be supported and identified by the involved domains;
- Negotiation can have a partial success: not necessarily all the requested characteristics in a negotiation are possible to be offered by the destination domain or by all the transit domains (when a reservation among the entire path is necessary). However, even in this situation the policies necessary to allow the communication between the domains can be created since they guarantee a minimum and acceptable communication level instead of the initially desired one; and
- Negotiation might start other processes: policies negotiation is a high level mechanism which is not directly involved to any specific technology responsible by dealing with a particular requirement and control it, as QoS allocation and management, for example. In this sense, negotiation will be responsible by defining high level policies and start the specific mechanism necessary to configure them in the environment.

Based on these assumptions is possible to have an overview of the negotiation process and the interaction of the elements. Initially a Binding Negotiation might be performed. This initial negotiation guarantees the first contact between the involved networks and creates the policies necessary to allow the basic communication of the networks. After this initial negotiation, specific negotiations might be executed. These sub-negotiations are responsible by defining specific characteristics of the communication which are controlled by the Technology Negotiators and will refine the initial agreement according to networks' requirements during the communication. This overview is presented in Figure 4.25.

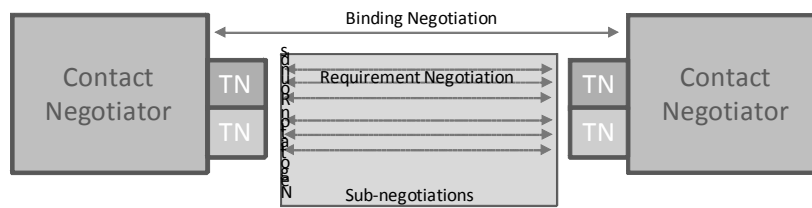


Figure 4.25 – Overview of the Negotiation Process

Since each phase has its specific characteristics which might be considered by the protocol, elements interaction and messages exchange will be presented grouped by the phase they are related and in the end the entire process will be presented. The diagrams indicating the operations of each phase will be presented, even though they have been discussed in Section 4.2, to facilitate the understanding.

4.3.4.1 Discovery

The discovery phase is a pre-configuration phase that is necessary during the Inter-Domain Policies Negotiation. This phase is necessary to verify if a requested resource (host address, service, etc.) is located in the same network of the requesting node and if not check if it is located in any other network.

It's easy to note that this phase is related with the specific mechanism used by the network to locate resources and according to the information used the networks will be compatible or not. As routing protocols in Inter-Domain communications are not able of being negotiated, since the networks need to configure them according to their internal rules, the discovery phase is a “black box” to the rest of the negotiation mechanism.

On the other hand, the sequences of events that are generated by the discovery phase are important to the overall understanding since it will be responsible by initiating the communication among domains and by the “detection” of the need of starting a new negotiation.

Initially the users' device might find out if the resource is located in the Intra-Domain in which it's connected. A diversity of routing protocols can be used according with the preference of the user or the operator, but the specific mechanism would not influence the way the negotiation process will be performed.

After checking the Intra-Domain with no answer to their requests, users' devices might request an Inter-Domain discovery, trying to discover if the resource is accessible in another domains or if it just doesn't exist. In the same way of Intra-Domain routing, the particular protocol used in the Inter-Domain communication will not directly influence in the negotiation since it's just responsible by discovering if the requested resource is available and, if so, the domain that might be contacted to access it.

Even not being responsible by how the execution of the Inter-Domain Policies Negotiation will be performed, the discovery phase has important function over this process. According with the protocol or set of protocols used by each Domain in their local communication and the protocol used to communicate with other Domains, a different set of information might be obtained from the users' devices/networks and depending of this information the negotiation process would have a better chance of finding a network which could offer the resource and, consequently, being succeeded.

4.3.4.2 Announcement

Announcement phase is responsible by starting the interaction between the involved domains in order to create the schema necessary to their communication. Announcement phase is started, as presented in Figure 4.26, by the Requesting Domains when it creates a NEGOTIATION_START message and sends it to the Destination Domain. After sending this message the Requesting Domain will keep waiting for a response until the value of "Expiration time" field be reached. If the maximum number of retransmissions was reached the Domain will finish the announcement process and decide if a negotiation with another domain will be started.

During the "Expiration time" period of the NEGOTIATION_START message, if the Requesting Domain receives a message it will process it. Three messages can be received at this point of the operations: NEGOTIATION_PARTIAL_ACCEPT, NEGOTIATION_ACCEPT and NEGOTIATION_REJECT.

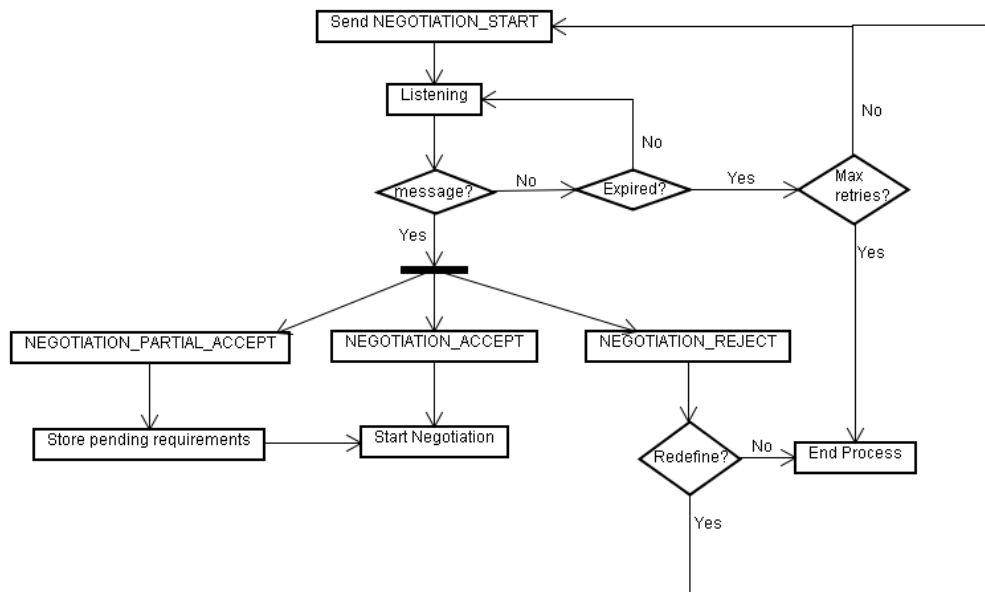


Figure 4.26 – Requesting Domain Announcement Algorithm

When receiving a `NEGOTIATION_PARTIAL_ACCEPT`, the Requesting Domain identifies that mandatory requirements were accepted but only part of the specific requirements or none of them were accepted by the Destination Domain. At this moment the Requesting Domain stores the information related with the requirements which were not accepted, allowing them to be negotiated during the negotiation phase, and having an initial agreement already defined, starts the negotiation phase.

If a `NEGOTIATION_ACCEPT` message is received, the Requesting Domain considers that all requested requirements (mandatory and specific requirements) were accepted and the negotiation phase might be started. In the last case, after receiving a `NEGOTIATION_REJECT`, a Requesting Domain note that some mandatory requirements are not possible of being supported. The Requesting Domain checks the unavailable requirements, if this information is present in the message, and decides if it wants to change the requirements to them retransmit a new `NEGOTIATION_START` message or just finish the announcement and, consequently, the negotiation process with the selected Destination Domain. If the announcement fails a new Destination Domain can be selected and the process can be started again.

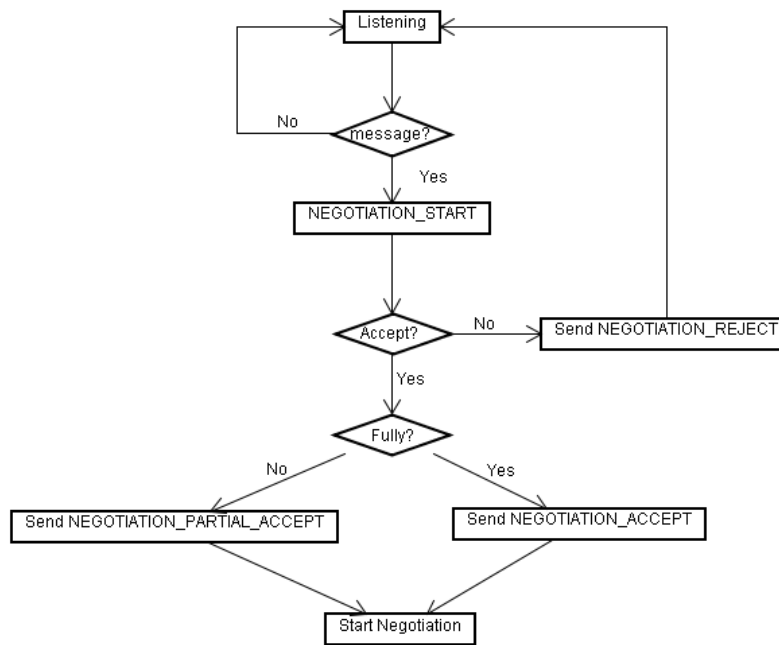


Figure 4.27 – Destination Domain Announcement Algorithm

In the Destination Domain (Figure 4.27) a `NEGOTIATION_START` message is expected to be received. When this message is received the domain will check the requested characteristics and decide if it should be accepted or not according to domain's available resources and to some predefined policies that can be defined by the domain's administrators. If the request is not accepted the Destination Domain will transmit a `NEGOTIATION_REJECT` message to the Requesting Domain indicating that the Binding Negotiation was not accepted. On the other hand, if the request is accepted, the domain will decide which requirements will be offered and then transmit a `NEGOTIATION_ACCEPT`, if all specific requirements are supported at the indicated levels, or a `NEGOTIATION_PARTIAL_ACCEPT`, if only part of the specific requirements will be offered. When a `NEGOTIATION_PARTIAL_ACCEPT` is transmitted it can also contains the list of the requirements which will not be offered in an initial moment.

After finishing the announcement phase the basic policies to represent the initial agreement created by the involved domains must be created in order to establish the communication between them. This agreement in a general way will just indicate the access control rules necessary to allow the basic connectivity to the communication. The negotiation phase is

then expected to be started with the objective of creating the policies necessary to support the specific requirements which will be necessary during domains communication.

4.3.4.3 Negotiation

Negotiation phase is the core of the proposed mechanism. This process is the responsible by trying to defining consensual requirements to be used in the communication according to the particular needs identified by each involved domain when trying to establish a new agreement to allow their automatic connection. During the definition of the necessary requirements, their levels and the cost to use this service during the connection can be also negotiated and used by the domains while the requirements are valid. In the end of all this process the necessary policies to allow the communication between the domains are created and can be applied in the specific operational environment of each domain.

The negotiation process starts when the Requesting Domain sends a message to contact the Destination Domain, as shown in Figure 4.28. Four messages can be transmitted at this moment: RESOURCE_STATE_REQUEST, NEGOTIATION_REQUEST, RESOURCE_QUOTATION and TN_LIST_REQUEST.

Resource control messages will be responsible by obtaining information from the Destination Domain and how its resources utilization is in a particular moment. RESOURCE_STATE_REQUEST message asks the current use of a specific resource in the Destination Domain. This message will contain the list of resources to be requested to the Destination Domain. RESOURCE_QUOTATION message will request the cost to access a service or a set of services from the Destination Domain and can help in the negotiation process to be used as one of the parameters to services selection. These messages will basically have a “direct” answer and with this information the Domain is capable of store the necessary data of other domain’s state.

NEGOTIATION_REQUEST message will initiate a requirement negotiation process, to which is a set of policies might be created in order to make possible the Origin and the Destination Domains identify the actions that might be performed with the traffic that is coming from the other domain. This message contains the all the specific requirements that a Requesting Domain want to negotiate in a given moment, indicating the requirement, the

level that is necessary and a payment the Domain would be able of offering for this access. Negotiation ID field is also created, following a sequence number used to control that, allowing the Requesting Domain to identify the result of this specific request when a message replying it is received. 'TN_LIST_REQUEST' will request information regarding the available Technology Negotiators in the Destination Domain.

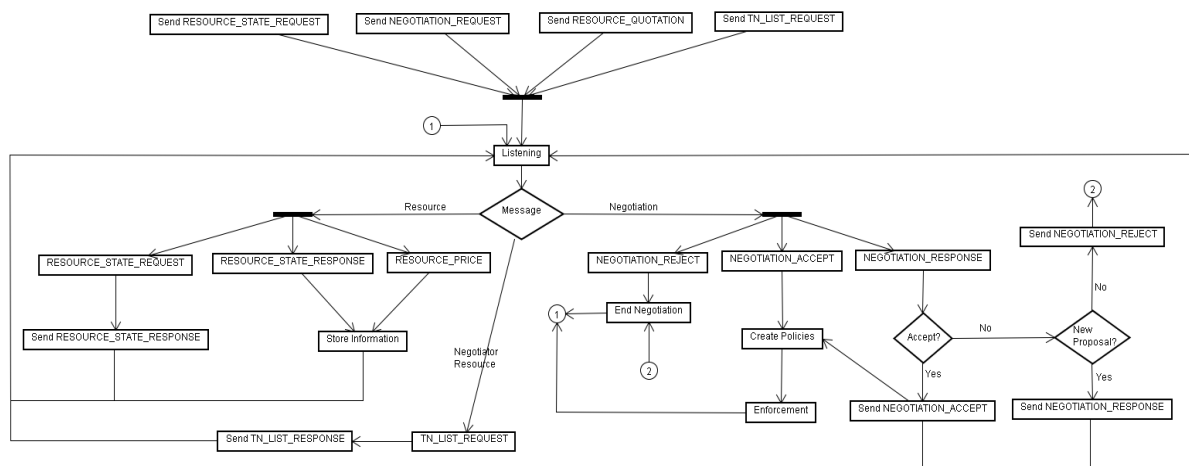


Figure 4.28 – Requesting Domain Negotiation Algorithm

The transmission of one of these four messages will start the negotiation process and then the domain will wait for a message which will contain a response for them. As three groups of messages can be transmitted, Requesting Domain needs to check the type of the received information: a resource, a negotiation or a negotiator resource message.

If a 'TN_LIST_REQUEST' is received from a Destination Domain, it indicates that this domain needs information of Technology Negotiators that are present in the Requesting Domain. It will happen when modifications in the agreements are desired by the Destination Domain and to start the negotiation it needs this information. After receiving this message a Requesting Domain sends a 'TN_LIST_REQUEST' with available negotiators information.

When resource-related information is received three types of message are possible: 'RESOURCE_STATE_REQUEST', 'RESOURCE_STATE_RESPONSE' and 'RESOURCE_PRICE'. A 'RESOURCE_STATE_REQUEST' message contains a request from the Destination Domain to obtain the information of current utilization of a resource located in the Requesting Domain. This message will generate a 'RESOURCE_STATE_RESPONSE', which is sent back to the Destination Domain with the

requested information. The second possible type is a RESOURCE_STATE_RESPONSE message. In this case, as the message represents an answer to a RESOURCE_STATE_REQUEST that was transmitted by the Requesting Domain, the information of the resources state is stored to be used in the future. If the Requesting Domain receives a RESOURCE_PRICE, information regarding resources which were asked to be quoted is received and then the Requesting Domain stores the information to be used in the decisions necessary to define if the resource is feasible of being accessed.

In the case of receiving a negotiation message the Requesting Domain checks if it's a NEGOTIATION_REJECT, a NEGOTIATION_ACCEPT or a NEGOTIATION_RESPONSE. In the case of a NEGOTIATION_REJECT message, the Requesting Domain will be notified that requested resources by a negotiation, indicated by the Negotiation ID, are not possible of being allocated and, consequently, the negotiation cannot continue. The negotiation is then finished and the Requesting Domains return to wait for messages.

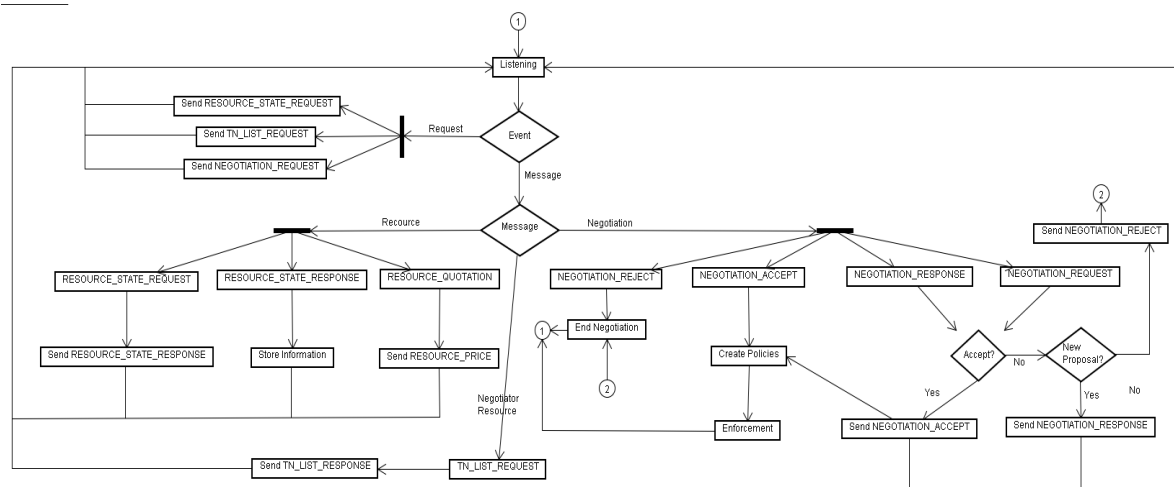
A NEGOTIATION_ACCEPT message indicates that all requested requirements are supported by the Destination Domain and the specified levels and payments were accepted. When this happens the policies necessary to represent the negotiated access are created and enforced in the environment.

When receiving a NEGOTIATION_RESPONSE message the Requesting Domains is informed that a request was not accepted and the Destination Domain proposed some modification in the requested levels of the requirements or in the proposed payment for it. The NEGOTIATION_RESPONSE message contains a list of the requirements, the new proposed levels and the payment expected to that. This list not necessarily is composed by all the requested requirements, since some of them can be accepted by the destination domain, so, only changed requirements are transmitted.

A Requesting Domain should check if the new proposed levels and payments are in accordance with the necessary communication level and the possible maximum value that can be paid. If so, a NEGOTIATION_ACCEPT message will be send to inform the Destination Domain that the proposal was accepted and it can create the communication policies related to that. If the proposal is not accepted but new modifications are done in the

proposal, a `NEGOTIATION_RESPONSE` message is create to inform the Destination Domain of the new possible level and payment proposed by the Requesting Domain. If the proposal is not accepted and no new proposals are created a `NEGOTIATION_REJECT` is transmitted to terminate the negotiation of the requirements. In this message the domain will indicate if none of the requirements will be used (no policies are created) or if only the ones that were not agreed. This information is obtained by the List of Unavailable requirements which would have all the requested requirements or only the rejected ones.

Figure 4.29 – Destination Domain Negotiation Algorithm



The Destination Domain when starting the negotiation process after executing the announcement will wait for messages from the Requesting Domain, as shown in Figure 4.29. Events in the network might also be captured by the Destination Domain and in this case an appropriate message should be transmitted.

Events can create three different types of messages: `RESOURCE_STATE_REQUEST`, `NEGOTIATION_REQUEST` and `TN_LIST_REQUEST`. `TN_LIST_REQUEST` asks information of the available negotiators and is used when the Destination Domain needs to renegotiate some requirements and want to check possible negotiator to be used. `NEGOTIATION_REQUEST` starts a renegotiation of one or more requirements. When transmitted by the Destination Domain, the ID of the negotiation that might be changed is identified and new requirements' values are informed. `RESOURCE_STATE_REQUEST` message is transmitted to require the Requesting Domain to inform the status of some

resources. This message can be used to verify if some characteristics of the communication can be improved based on resources availability.

In the same way of the Requesting Domain, Destination Domain can receive messages related with resources verification, negotiation or negotiators resources. A resource message can be a RESOURCE_STATE_REQUEST, a RESOURCE_STATE_RESPONSE or a RESOURCE_QUOTATION. RESOURCE_STATE_REQUEST messages are received from the Requesting Domain and when received will create a RESOURCE_STATE_RESPONSE message with the utilization status of the requested resources. A RESOURCE_STATE_RESPONSE is received as an answer to a RESOURCE_STATE_REQUEST which was transmitted before and will contain the information related with the resources that the Destination Domain needs to know about the utilization. The information from this message will be stored to be used in negotiation decisions. Last resource message is the RESOURCE_QUOTATION, which is responsible by requesting the price of a resource or a service. After receiving this message the requested resources are checked and a RESOURCE_PRICE is transmitted containing the information of the price from the existent resources.

The Destination Domain can also receive negotiation messages from the Requesting Domain. In the case of NEGOTIATION_REJECT, NEGOTIATION_ACCEPT and NEGOTIATION_RESPONSE messages, the Destination Domain will execute the same operations of the Requesting Domain when it receives these messages.

The modification in the Destination Domain operation occurs when a NEGOTIATION_REQUEST message is received. This message contains the list of the requested resource, their levels and proposed payment from the Requesting Domain. This list is evaluated by the Destination Domain in order to check if the requested resources are available and if the levels and payment are acceptable or not. If the Destination Domain is in accordance with all the characteristics of the requests a NEGOTIATION_ACCEPT is transmitted to inform that to the Requesting Domain. If it's not accepted and no new proposals are create, the Destination Domain sends a NEGOTIATION_REJECT message indicating the requirements that were not in accordance with the characteristics from the domain or the ones that cannot be offered. In the case of receiving a

NEGOTIATION_REQUEST and make modifications in the requested characteristics or payment, a NEGOTIATION_RESPONSE message indicating the new proposed levels and the expected payment is sent to be evaluated by the Requesting Domain. As occurs with the Requesting Domain, TN_LIST_REQUEST when received will request information regarding the available Technology Negotiators to be used during negotiations.

4.3.4.4 Enforcement

The policies that were mentioned to be used by the Inter-Domain Policies Negotiation Mechanism to be used and stored must somehow follow a pattern to make all the domains compatible and, consequently, able of negotiating [OHL06] [USZ03] [KAG03] [KAM06]. Enforcement phase is the responsible by taking defined policies in a high level which is a domains' technology independent and translate them to the specific policy representation necessary to be applied in each domain. As this translation can be done to different technologies in each domain, the negotiation is capable of following a standardized policy representation. The basis of the proposed abstract policies model is presented following.

4.3.4.4.1 Policy Deployment Architecture

The Policy Deployment Architecture is basically composed by five main elements: Agent, Guard, Negotiators, Policy Engine, and Administrative Interface (see Figure 4.30), and three auxiliary elements: Ontology, Policy Directory and Communication Protocol. Main elements represent the functional elements responsible by the enforcement of the policies and the auxiliary elements indicate the elements which controls the interaction of main elements.

Agents are generic elements which should be any managed such as a cellular, a laptop, a router, an entire network or any other element, and all of them can be controlled in the same way. The Agent doesn't have the ability of accessing directly the engine and, consequently, the stored policies in order to verify if a specific action can be performed. Agent's interaction is restricted to Guards, which will allow the communication with the other elements responsible by storing and controlling the policies. Because of its ability of representing any kind of element, Agents must be created considering the specific characteristics of the element and of the environment it will be operating to guarantee the correct management.

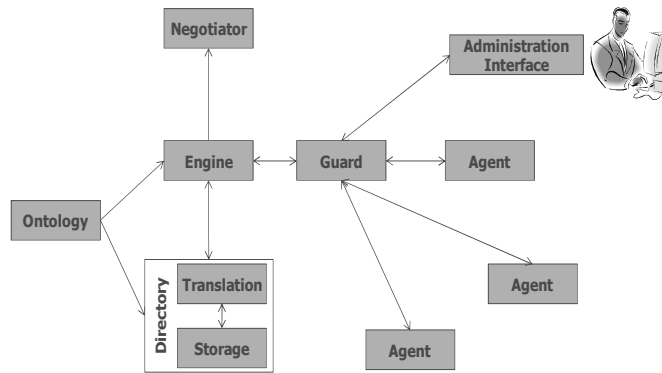


Figure 4.30 – Policies Framework Architecture

The Guard represents the intermediate element responsible by obtaining information from the policy engine and passes it to the Agents. Typically this information is related with policies that are requested by the Agents to verify if an action can be performed or not, but it should have other functionality such as notification if a new policy requested by the Agent can be stored in the Engine or if a policy can be removed. This element creates an abstraction level which guarantees to Agents independency to store and manipulate policies of specific technologies. So, Agents should connect to other manageable device and obtain the necessary policies to operation this device.

Negotiators are elements responsible by contacting other networks trying to establish dynamic agreements to define how communication among networks should be performed.

Policy Engine element is responsible by manipulate all policies in the domain and offer a structure to check if the policies are correctly created, and if there is no conflict with other policies. Two modules are necessary: a mechanism responsible by the conflict resolution and another to store the policies. The former will receive the requests from guards and negotiators, indicating the operations to be performed and their parameters, checking if the operation is consistent with the already created policies and if these operations can be executed. The latter is used to add, remove, update or obtain information of the policies. The conflict resolver and the storage mechanism are expected to use different technologies depending of the preferences of the administrator/designer and might not influence in the other elements of the architecture.

The Policy Administrator Editor is used to browse and load ontologies to define, commit new policies, and modify or delete them. Although the framework have been created to be used in a dynamic environment, where the nodes should have the capability of create policies dynamically, an administrative interface probability could be important. The interface could offer an administrative way for to manage policies, in order to verify policy conflict and also to create static policies such as users groups, resources' access control, etc.

The Auxiliary Elements are responsible by the interaction of the main elements and the description of the typical features of the environment, devices and policies. The Ontology is a formal description necessary to define a basic pattern to structure policies in any network, and to describe generic characteristics of an environment and the devices connected to network. Depending of the specific characteristics of different environments, this Ontology can be extended to represent new information and attend to new necessary requirements.

The Policy Directory is responsible by manipulating the whole information model that is managed in the environment (store basic ontology, extended ontologies). This directory can be centralized, distributed or to use any storage technology without changing the interaction between the other elements of the architecture.

The Communication Protocol is necessary to guarantee the interaction of the elements by exchanging information between Policy Directory and Engine, and the distribution of the policies through the network elements.

4.3.4.5 Monitoring

Monitoring phase is responsible by verifying if any modification in the policies defined by the involved domains is necessary to adapt them to any modification in the environment or in the requirements that are necessary by the communication.

The Requesting Domain algorithm will be waiting for an occurrence being detected in the environment, as presented in Figure 4.31. This occurrence can be a notification indicating that the communications is no longer necessary, a message that is received or a modification in the environment characteristics or in the requirements that were requested during the negotiation phase and should be modified to be in accordance the new needs from the communication. Modifications of environment or requirements characteristics will start the

negotiation procedure to the specific characteristics that might be modified and after renegotiating the necessary policies, the domains will return to monitoring phase.

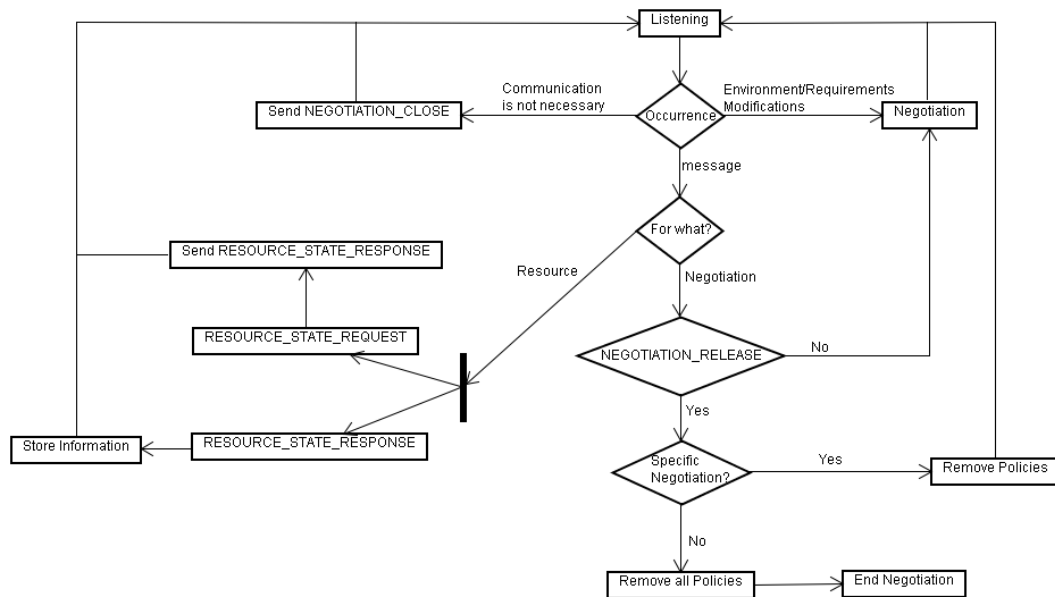


Figure 4.31 – Requesting Domain Monitoring Algorithm

Notifications which indicate that the communication is no longer necessary can be created by many reasons, such as when a binding or a requirement negotiation expires or when the domain doesn't need the communication active.

Finally, if a message is received, monitoring phase will perform actions necessary to each one of them. When a negotiation group message is received the Requesting Domain checks if it's a NEGOTIATION_RELEASE message. If so, it just checks if it's related to a specific Requirement Negotiation or to the Binding Negotiation based on the Negotiation ID field. In the case of a Requirement Negotiation just the specific policies will be removed. For Binding Negotiations all created policies are removed and negotiation is terminated. Negotiation messages that aren't to release a previous negotiation, will call the negotiation procedure in order to renegotiate the already defined agreement or the specific policies created during Requirement Negotiations.

However, messages received during monitoring phase can also be related to resource messages group. If a RESOURCE_STATE_REQUEST message is received the domain will check the current state of the requested resource or set of resources and sends a

RESOURCE_STATE_RESPONSE message. If a RESOURCE_STATE_RESPONSE is received the information received is stored and the domain goes to the initial state again.

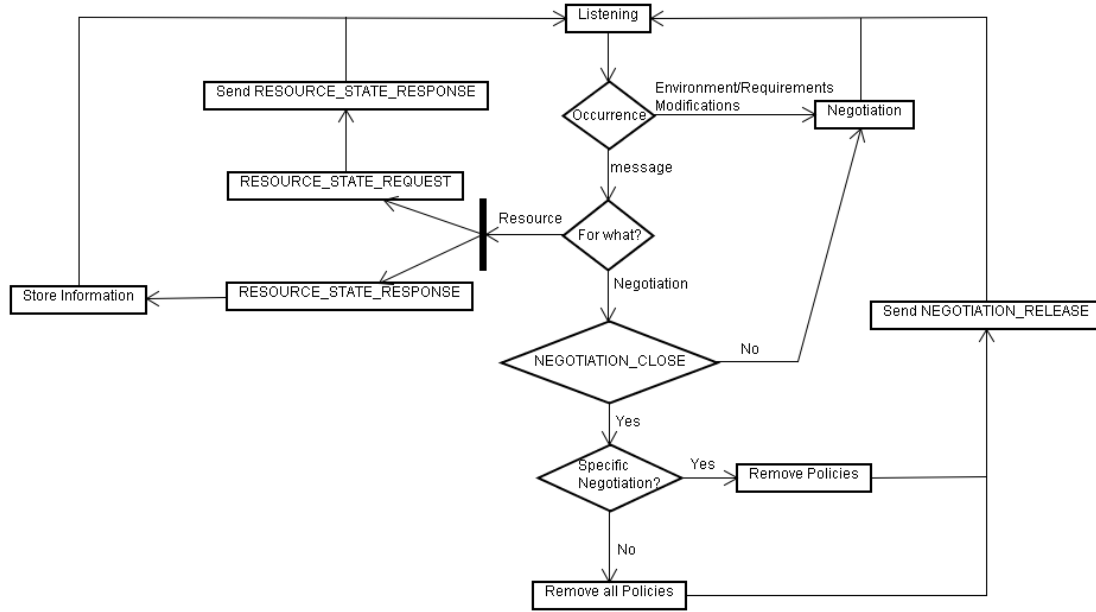


Figure 4.32 – Destination Domain Monitoring Algorithm

The Destination Domain algorithm (Figure 4.32) basically executes the same operations of the Requesting Domain. When modifications in environment or requirements are necessary the negotiation procedure is started again to renegotiate the necessary policies. Receiving resource related messages the operation will also be the same of the Requesting Domain: if a RESOURCE_STATE_REQUEST a RESOURCE_STATE_REQUEST is transmitted and if a RESOURCE_STATE_RESPONSE is received, resource utilization is stored.

Difference operation happens when a negotiation message is received. If the message is a NEGOTIATION_CLOSE the domain will check if it should deal with a specific negotiation or not. If so, only the necessary policies are removed and if a Binding Negotiation is requested to be closed all the policies related to that agreement should be removed. In both cases a NEGOTIATION_RELEASE is transmitted to indicate that the negotiation was finished. If the negotiation message received was not a NEGOTIATION_CLOSE, the negotiation will be executed to the requested requirements.

4.3.5 Elements Communication and Cooperation

During this initial phase of the negotiations, the Requesting Domain policies are negotiated with the Destination Negotiator. Similarly to other subsequent negotiations, it may take many rounds to get both sides to reach an agreement. If necessary, this negotiation must involve all intermediate routers from different Domains, since some resources might be negotiated/allocated in all domains the communication would pass by.

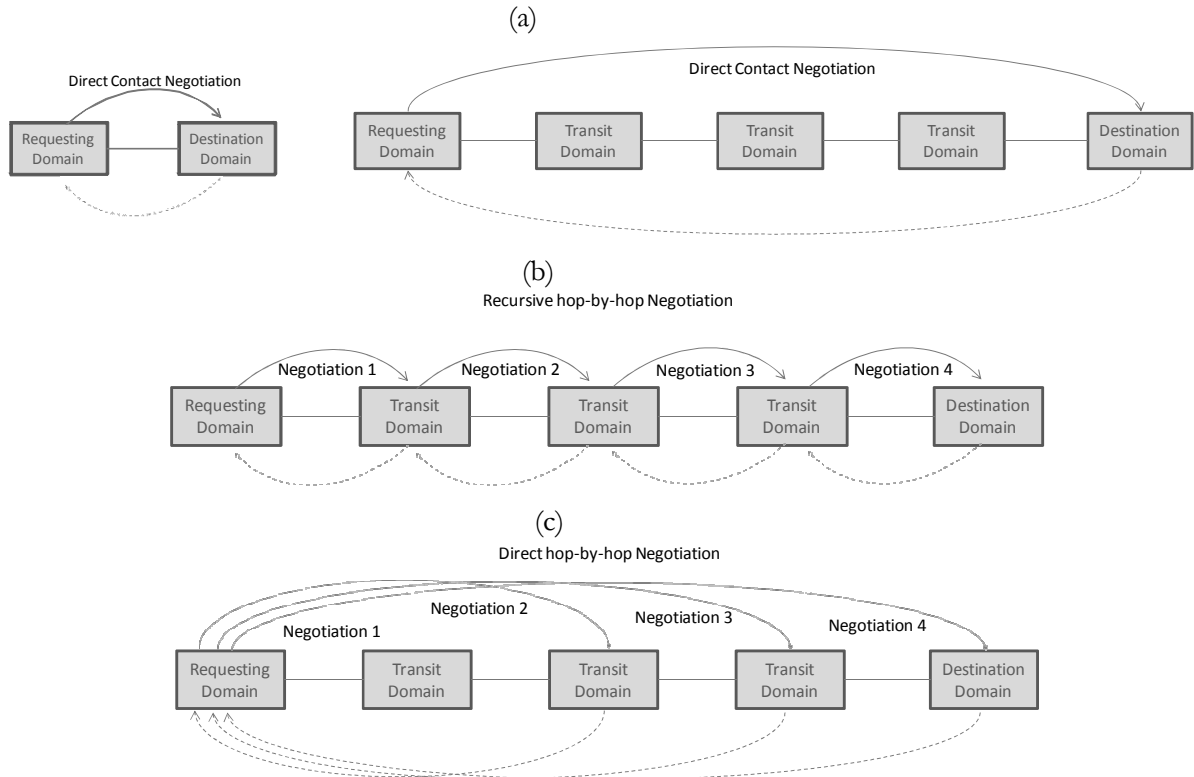


Figure 4.33 – Negotiation information exchange modes: (a) direct contact negotiation, (b) recursive hop-by-hop negotiation and (c) direct hop-by-hop negotiation

This negotiation can be done in three different ways: direct contact, recursively hop-by-hop or directly hop-by-hop. Direct contact negotiation is performed when two negotiating domains are directly connected or when the requested connection don't need the support from all the Transit Domains, i.e. when end-to-end communication must be created as a ciphered transmission where only the origin and the destination controls the negotiation.

The main difference between the two last modes is that, in a direct hop-by-hop negotiation, Origin Domain is responsible by negotiating with all intermediate routers individually, and in

recursive hop-by-hop negotiation each intermediate router negotiates the communication link with the next hop. Figure 4.33 represents how the negotiation is done in both situations.

According to the selected information exchange mode differences in performance and scalability are expected, since more negotiations would be necessary and, consequently, more information about them should be stored.

4.4 Summary

Inter-Domain communication is controlled by a set of policies that are established to represent the agreements defined between the two networks and regulate how the resources from each of them can be used by the others.

These agreements and policies are negotiated and created in a static way and is not able of dealing with users' dynamic need of communicate with new or never contacted networks. This problem is more evident when more distributed situations are considered, where many users are expected to create their own Personal Area Networks which must negotiate connectivity, services access, user authentication, etc.

To support this communication model the Inter-Domain policies might be negotiated in a dynamic way, creating an automatic mechanism that can complement the mechanisms proposed in Chapter 3 and complete the communication cycle necessary by new networking environments.

In this chapter was presented a description of a mechanism to negotiate the policies that might be created to allow this self-configuring Inter-Domain communication and the specific phases and elements to make it possible. The phases necessary to ensure the correct policies definition, monitoring and renegotiation were defined and detailed discussed, indicating the operations and interactions that might be executed by the involved domains in order to create and keep new agreements and the related policies

5 Experiments and Results Analysis

In this chapter the results obtained from the simulations prepared to validate and evaluate the proposed mechanism are discussed to show that these mechanisms are able of solving the problems described in Chapter 2. The results that will be described are related with all configuration levels proposed in this work: addressing, Intra-Domain and Inter-Domain negotiations. The results presentation is divided according to the evaluated mechanism, simulation parameters and simulated scenarios.

5.1 Validation of the proposed solutions

Before implement the proposed solutions in a simulator to perform their evaluation, they were implemented in operational prototypes in order to validate their correct operation and give support protocols modeling necessary to simulator implementation.

Addressing mechanisms were implemented and evaluated in local networks in order to check if addresses allocation would assign addresses correctly. To those assignments two network configurations were considered: a network with at least one server and a network with no addressing server. In the first case all devices which have tried to contact a server has received a valid and free address from at least one server and them has configured its network interface with the assigned address. When no addressing server is located in the network, the self-addressing takes place and node calculates its own address and tests the network to check if no other nodes are already using it.

The Intra and Inter-Domain Negotiation protocols were validated using implemented prototypes too. Intra-Domain Negotiation was fully implemented in the prototype considering dynamics which would be applied in the network and checking the actions

performed by the protocols and if the decisions were in accordance with the operations defined to all protocol's phases, as described in Chapter 3.

Inter-Domain Policies Negotiation protocol was validated through a simplified prototype. To evaluate Inter-Domain agreements definition and the enforcement of negotiated rules by creating the necessary policies to guarantee the correct communication between the domains in a roaming scenario. To do so, a user which is subscriber for an operator tries to connect in a second operator network. As no information about this user are located in the second operator it needs to connect with the origin operator of the user and create an agreement to exchange necessary information to allow user to get connected and keep information related with quality of supported communication updated.

5.2 Experiments Environment

In this section is presented how conducted experiments were prepared to evaluate the proposed solutions and other approaches to be compared with. The main objective was to simulate proposal basic functionalities to understand their behavior, evaluate their performance and validate their correct operation.

To guarantee the correctness of the evaluations it was necessary the execution of a set of simulation rounds, ensuring a higher reliability of the gathered results. To calculate the number of simulation rounds necessary to guarantee data consistency, equation (5.1) [JAI91] was used during simulations execution, checking if more repetitions would be necessary to reach the desired precision. In performed experiments were considered a confidence interval of 95% and an accuracy of 5%.

$$n = \left(\frac{100 \cdot z \cdot s}{r \cdot \chi} \right)^2 \quad (5.1)$$

In equation (5.1) n represents the necessary number of observations (i.e., simulation rounds), z is the confidence interval, s is the sample standard deviation, r is the required accuracy, and χ is the sample mean.

5.2.1 Simulation Environment

All the experiments were done in a personal computer, running the operating system *Linux*, distribution *Ubuntu 9.04*, and kernel version 2.6 [UBU09]. The simulations were done in the Network Simulator version 2 (NS2) [FAL09]. All the evaluated protocols in this work were implemented in C++ and integrated to the *ns2* simulator. The manipulation of the protocol's features was possible through scripts in *TCL/OTCL*, which were also used to program the simulation scripts (as required by *ns2*).

5.2.2 Simulation Topologies

To evaluate the proposed mechanism two different situations: a static and a dynamic scenario. These two scenarios were chosen because we intended to analyze some of the mechanisms in different networking situations, where the static scenario represented a more controlled environment and the dynamic one representing a scenario where nodes have more freedom, since they don't have a specific deployment position. In the performed experiments only addressing mechanism has used the dynamic scenario, since it would be the most affected considering the defined parameters and the selected metrics.

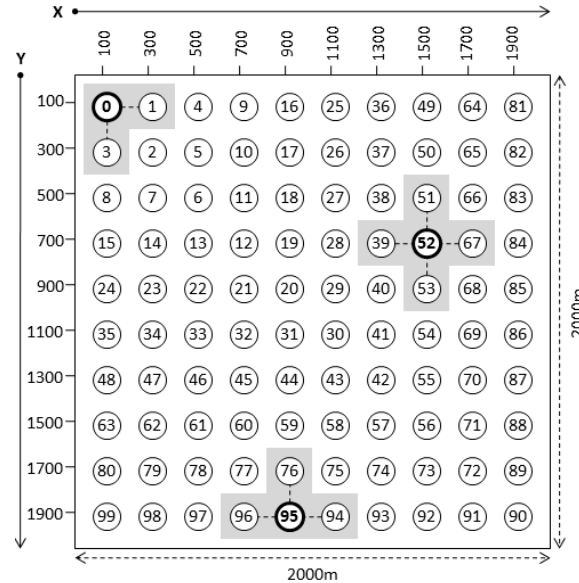


Figure 5.1 – Static scenario: nodes positioning in a grid-format and controlled topology.

In the experiments, the difference of dynamicity between the two scenarios is that in the first one the nodes were initiated following a predetermined order and position, and in the second the nodes were randomly positioned. Nodes mobility was not considered and will be worked in future evaluations.

Figure 5.1 illustrates the positions of the nodes in the grid-topology. It also indicates the initialization order of the nodes, from node 0 (the first node) to node 99 (the last node), totalizing 100 nodes. This scenario was selected since it is commonly used in other scientific works and presents a controlled and well know connectivity pattern. Moreover, depending on their respective positions in the grid-topology, nodes can have a minimum of two orthogonal connections (i.e., two 1-hop neighbors) and a maximum of four orthogonal connections (i.e., four 1-hop neighbors).

The first topology, is composed by 100 stationary nodes, placed in predetermined positions defined by (x,y), forming a grid-topology. The dynamic scenario is composed by 50 nodes. Differently from the first scenario, the topology in dynamic scenario is uncontrolled and because of that the protocols were evaluated under more dynamic situations, even not considering nodes mobility in these first experiments. Due to the random positioning, in the second scenario nodes do not have a defined minimum or maximum number of neighbors. The number of node was selected intending to represent a local network environment, since the protocols are expected to operate in these situations.

5.3 Local Address Negotiation

In this section is presented how we conducted the experiments and the parameters used to simulate SooA and other self-addressing approaches. The main objective was to simulate SooA's basic functionalities to evaluate its performance. Some of the SooA's parameters were varied and its performance was also compared with other addressing protocols.

5.3.1 Implemented Protocols

Besides SooA, we implemented and evaluated three other protocols of self-addressing: Strong DAD [PER01], Prophet Allocation [ZHO03] and MANETconf [NES02]. We opted to implement those three protocols to have at least one representative approach of each category presented in Chapter 2. Strong DAD represents the basic DAD-based stateless approach. Prophet Allocation represents stateful approaches that do not implement addressing tables. And MANETconf represents hybrid solutions with DAD procedures and distributed allocation tables.

Our main goal was to analyze and validate the procedures of addressing resources allocation executed by each protocol, i.e., to find out how reliable they are when performing such tasks and, if applicable, maintaining the addressing integrity without compromising network operation. Therefore, considering only resources allocation and maintenance procedures, some protocols were not entirely implemented.

For SooA, we implemented the addressing resources allocation and maintenance. This means that it was implemented the relation between servers and clients, and also between father and child servers. However, failure situations in the network and backups operation were not implemented. Considering this fact, the performance of SooA was analyzed regardless the traffic generated by these operations. The evaluation of these advanced functionalities of SooA is planned as future work.

Regarding the Prophet Allocation protocol, we implemented the module of address assignment and addressing maintenance. Due to the lack of information in the references where this protocol was proposed, we did not implement the conflict detection and correction module. The procedures to support network merging and partitioning were not implemented because those situations were not considered in the performed experiments.

It was also implemented the addresses allocation procedure of MANETconf. As well as with Prophet Allocation, only the basic allocation mechanism given that networks merging and partitioning were not simulated in these experiments. Finally, the protocol Strong DAD was entirely implemented, since it is composed only by a single module which is the broadcast-based DAD procedure.

5.3.2 Simulation Scenarios

Local addressing evaluation has considered the two topologies that were defined to be used in the experiments. The main objective in considering both topologies is verifying the influence that different networks organization and devices characteristics will have in the communication characteristics of the network. These modifications would influence the protocols in different ways and, based on that, it will be possible to identify the adaptability to environment's changes from each one.

Table 5.1. Summary of simulation parameter for Static and Dynamic Scenarios.

Parameter	Static Scenario	Dynamic Scenario
Simulation area	2000m ²	500m ²
Simulation time	4000 seconds	2000 seconds
Number of nodes	100 nodes	50 nodes
Addressing space	100%, 200%, 300%, 400% and 500%	
Number of servers (for SooA only)	1 to 2 servers	1 server
MAC protocol	802.11	

A summary of the general parameters for both evaluated scenarios are presented in Table 5.1. It is important to state that it was not simulated situations of nodes departure because it was intended to evaluate the address uniqueness in increasing networks. Therefore, if nodes left the network, the uniqueness analysis would be compromised.

Simulation area was selected to guarantee that, according to the radio range of the nodes, the connectivity will follow the definitions of scenarios communication and topology. The simulation time considered the necessary period to guarantee the stationary behavior of the simulations. As Dynamic Scenario is composed by a lower number of nodes and they are more concentrated the necessary simulation time was reduced.

The addressing space was varied to evaluate the protocols in different situations: initially the network is provided with an amount of addresses much bigger than the number of network elements, and in a second moment a scarce amount of addresses is available. Changing this parameter protocols adaptability according to networks' available resources can be evaluated.

5.3.3 Evaluation Metrics

To evaluate the performance of the implemented protocols under the circumstances of the previously described scenarios, we considered the following metrics:

- Transmitted Data: the traffic generated by the protocol to execute addressing and resources management procedures. We considered the data in packet and bytes. For Strong DAD and MANETconf, the transmitted data is related to the traffic generated until the last node configuration, since these protocols only operate until the last node got configured. Differently, for Prophet Allocation and SooA, the

control overhead considered the traffic generated during all the simulation time because these two protocols performed resources post-allocation management;

- Configuration delay: the average time a node needed to get configured with a supposed valid and unique address. This means the difference between the time the node got configured and the time it started in the network; and

5.3.4 Network Scalability

In the first experiments the number of nodes was varied in the network to evaluate the protocols performance when facing network growth. The simulation parameters of these experiments are presented in Table 5.2 for both Static scenario and Dynamic scenario. In this experiment, considering Static scenario, we only simulated SooA with one and two servers, to illustrate the difference in its behavior and performance when distributing addressing decisions load. The first server was positioned in the node 0 and the second in node 12, since this node has forwarded many connections when SooA operates with one server as observed by the traffic pattern of SooA (Section 6.2.5).

Table 5.2. Summary of parameters for Network Scalability simulations.

Parameter	Static scenario	Dynamic scenario
Number of nodes	100	50
Addressing space	256 addresses	
Number of SooA servers	1 to 2 servers	1 server

Figure 5.2 illustrate the performance of the evaluated protocols in Static scenario. Figure 5.2(a) and 6.2(b) shows us the number of packets generated and transmitted by each protocol. The traffic generated by MANETconf is the highest among all evaluated protocols. It is due to its DAD procedure, where for each configuration, the network is flooded with request messages and a reply from all configured nodes is required. Moreover, another drawback of MANETconf is the traffic generated by the synchronization of local allocation tables, which is also done through broadcast procedures.

Strong DAD also has a high number of received packets resulted from the broadcast based DAD implemented by this protocol. Although, differently from MANETconf, Strong DAD does not requires reply to every request sent in its DAD procedure. Only nodes with

duplicate address will reply the DAD request in Strong DAD, what reduces the amount of information necessary by the protocol.

Prophet Allocation has its messages being transmitted mostly during addressing maintenance process, i.e., the periodically hello messages (which was set to every 10 seconds in this experiments). Given that the address assignment process in Prophet Allocation is a simple 2-message exchange between nodes, it does not generate a high traffic overhead. The increasing traffic for this protocol is because the more nodes are configured the bigger is the number of announcements sent and received.

SooA generated and transmitted a low number of packets (both sent and received packets) when compared to the other simulated protocols because its operations are mostly done by unicast message exchange. In addition, it decreased the number of transmitted packets when operating with two servers.

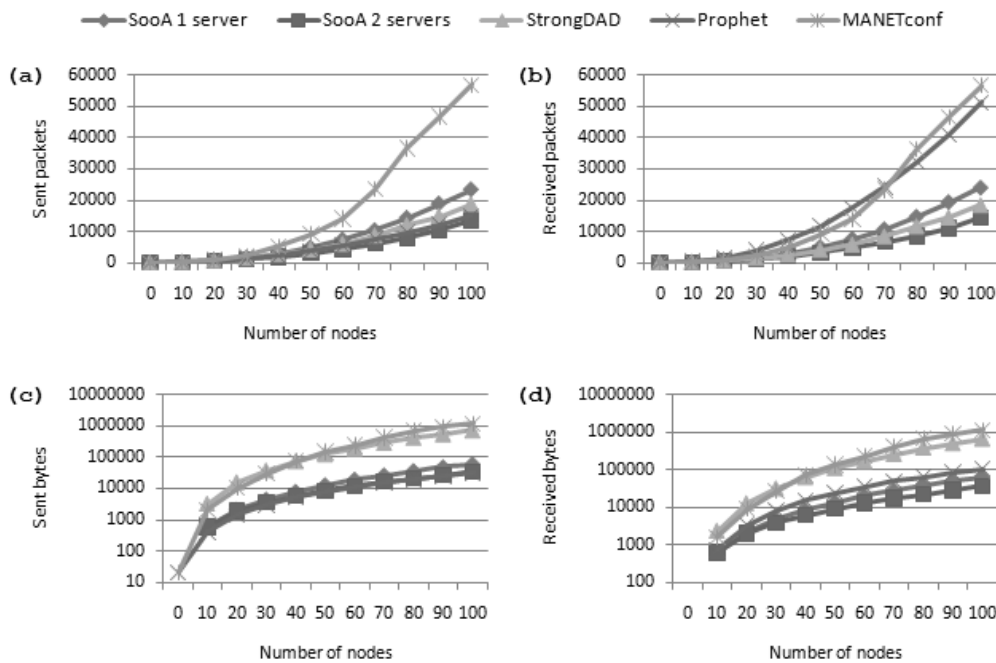


Figure 5.2. Scalability in static scenario: (a) sent packets; (b) received packets; (c) sent bytes; (d) received bytes.

Figure 5.2(c) and 6.2(d) shows the traffic generated by the protocols in bytes. Prophet Allocation and SooA, with one and two servers, transmitted a much lower number of bytes than Strong DAD and MANETconf. It is because, as abovementioned, SooA has its

functionalities based on unicast messages exchange, and the broadcast messages of Prophet Allocation are not retransmitted by the source's 1-hop neighbors.

Otherwise, MANETconf and Strong DAD generated a very high traffic in the network. Since the evaluated solutions consider no routing protocols will be running, they need to execute proper routing to distribute information when necessary. This way, in order to the nodes be able to reply broadcasted messages, a backward path must be kept within packets' information and, consequently, the overload increases in networks with a bigger diameter.

The graphic in Figure 5.3(a) illustrates the delay to configure nodes for each evaluated protocol. Strong DAD has the highest delay due to the execution of three DAD rounds. Since each DAD round lasts 5 seconds, the protocol reached an average of ~ 15 seconds of configuration delay. Prophet Allocation is the fastest protocol to configure a node. Since this protocol executes only a 2-message exchange to configure a node, its average delay was ~ 0.25 seconds. However, as abovementioned, Prophet Allocation operated only assign address and its DAD procedure is not part of this experiment. Considering that the protocol should periodically execute a DAD procedure, the configuration delay of Prophet Allocation would be increased in at least ~ 5 seconds, which is the average for a DAD execution round.

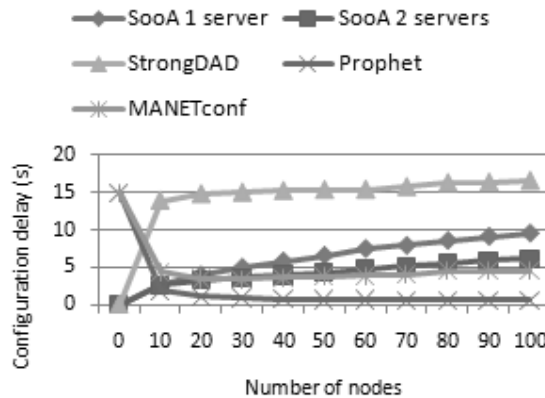


Figure 5.3. Configuration delay in Static scenario.

SooA when operating with one server configured the nodes with an average of ~ 6 seconds. The increasing line for SooA, in Figure 5.3(a), illustrates that the nodes more distant from the server need more time to be configured. However, when the second server was deployed in the network, at node 12, the delay was significantly reduced, to an average of ~ 4 seconds.

MANETconf also presented good values concerning the configuration delay. Since it implements local allocation tables, the DAD procedure executed is not exhaustive as in Strong DAD. MANETconf only executes DAD once, and then assigns the address to the new node. Therefore, the average configuration delay of this protocol was ~ 5 seconds (~ 10 seconds or two DAD rounds less than Strong DAD).

The traffic generated by protocols in Dynamic scenario is presented in the graphics of Figure 5.4. In this experiment, SooA was simulated only with one server positioned at node zero. The protocols Strong DAD and MANETconf presented a similar performance of Static scenario. However, as the second scenario presents a lower number of nodes, and a node can have more neighbors, the broadcast-based operation of these two protocols did not generate a much higher overhead, because their nodes only retransmit received messages by flooding once. Moreover, the performance of Prophet Allocation was also influenced by the bigger number of neighbors. The periodical announcement of Prophet Allocation resulted in a much higher number of received packets and bytes.

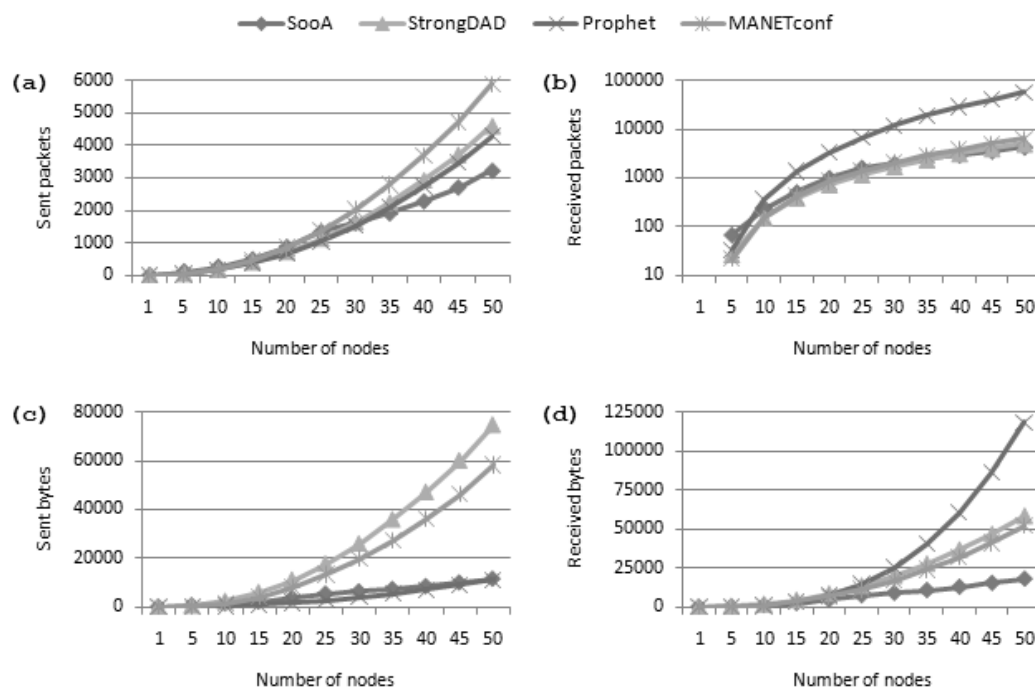


Figure 5.4. Scalability in dynamic scenario:(a)sent packets;(b) received packets;(c) sent bytes;(d) received bytes.

Even operating with only one server, SooA presented the best performance of all protocols considering the traffic in dynamic scenario. Due to the unicast message exchange

implemented by SooA, the number of sent/received packet and bytes is lower than the other protocols. The better performance of SooA is also explained by the lower number of nodes and the reduced area in dynamic scenario, which reduced the number of relayed connections since in more concentrated networks addressing servers are closer to other nodes.

Figure 5.5(a) illustrates the configuration delay of each protocol in Dynamic scenario. Strong DAD kept the same delay of Static scenario due to the 3-turn DAD procedure, as explained above. MANETconf also kept a similar average for configuration: ~ 5 seconds. Prophet Allocation presented a smooth increase in the average configuration delay caused by the distance among the first nodes which can be out of range from the other since they are randomly positioned. Consequently, the isolated nodes need to wait for other nodes being configured and then reach an initiator in the network. Despite this increase, Prophet Allocation still presented the best results of configuration delay.

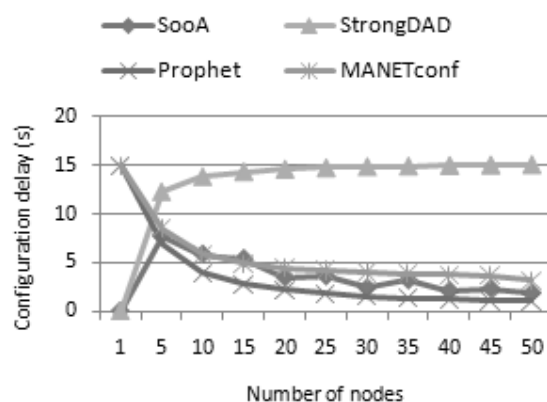


Figure 5.5. Configuration delay in Dynamic scenario.

Among all protocols, SooA presented a significant improvement in the configuration delay. The first nodes, due to the same reason of isolation explained above, had a higher configuration delay. However, in a general way, from 10 configured nodes in the network, any starting node would be able to reach the server and get configured faster than in Static scenario. This way, the average configuration delay for SooA is ~ 3.5 seconds.

5.3.5 Protocols' Traffic Pattern

In the last section the traffic generated by the simulated protocols was presented. However, only the total values might not be enough to predict if the protocols functionality is

satisfactory. Therefore, in this section we present the traffic pattern for each of the protocols. The data is the same collected in the simulations of the previous section, therefore, scenarios configuration presented in Table 5.1 is also applicable in this section.

In the graphics of Figure 5.6 we can observe how the control messages are distributed among the network nodes during the protocols operation. Each point in the grids of Figure 5.6 corresponds to a node and its position in Static scenario. The axis X and Y define the node's position (x,y), and the axis Z represents the amount of received bytes by each node.

In Figure 5.6(a) is illustrated the traffic distribution for SooA when operating with a single server at node 0 (zero). It is clear how the protocol centralizes the traffic in the region around the server. It is also easy to identify the nodes that assumed relay positions forwarding the traffic between other nodes and the server. The other nodes, which did not assume any functionality (clients), have a very low traffic. Moreover, the traffic values of SooA reflect its total operation time, which lasted 4000 seconds in the Static scenario, since SooA worked during the entire simulation period in addresses allocation and management.

The traffic pattern for SooA with two servers can be visualized in Figure 5.6(b). In this case, the second server was deployed in the node 12. We opted by the node 12 because we identified that this node relayed a significant number of connections between the server and other nodes when SooA was simulated with only one addressing server. Once again, we can observe the centralization of most traffic around the servers, and we can also identify the nodes that worked as relay. Comparing with the traffic in Figure 5.6(a), the deployment of the second server decreased the overall overhead. The first server (node 0) had its amount of received bytes reduced from ~ 10000 to ~ 2800 bytes. Consequently, the nodes that worked as relay from other clients to the first server also had their overhead reduced. Of course, since node 12 operated as a server, its traffic overhead increased, and the overhead of nodes around it. But even with this increase, the total traffic overhead for SooA with two servers was lower than when operating with one server (as presented in the previous section).

Broadcast-based mechanisms have a very different traffic pattern. In Figure 5.6(c) is illustrated the traffic distribution of Strong DAD. In this graphic we can observe the distribution of flooding traffic implemented by this protocol to probe addresses. The nodes that generated more traffic in the network are those which initiated earlier in the network,

since they had to forward more addressing discovery messages during their operation. While the first nodes had an overhead of ~ 17000 bytes, the last ones received less than 10 bytes.

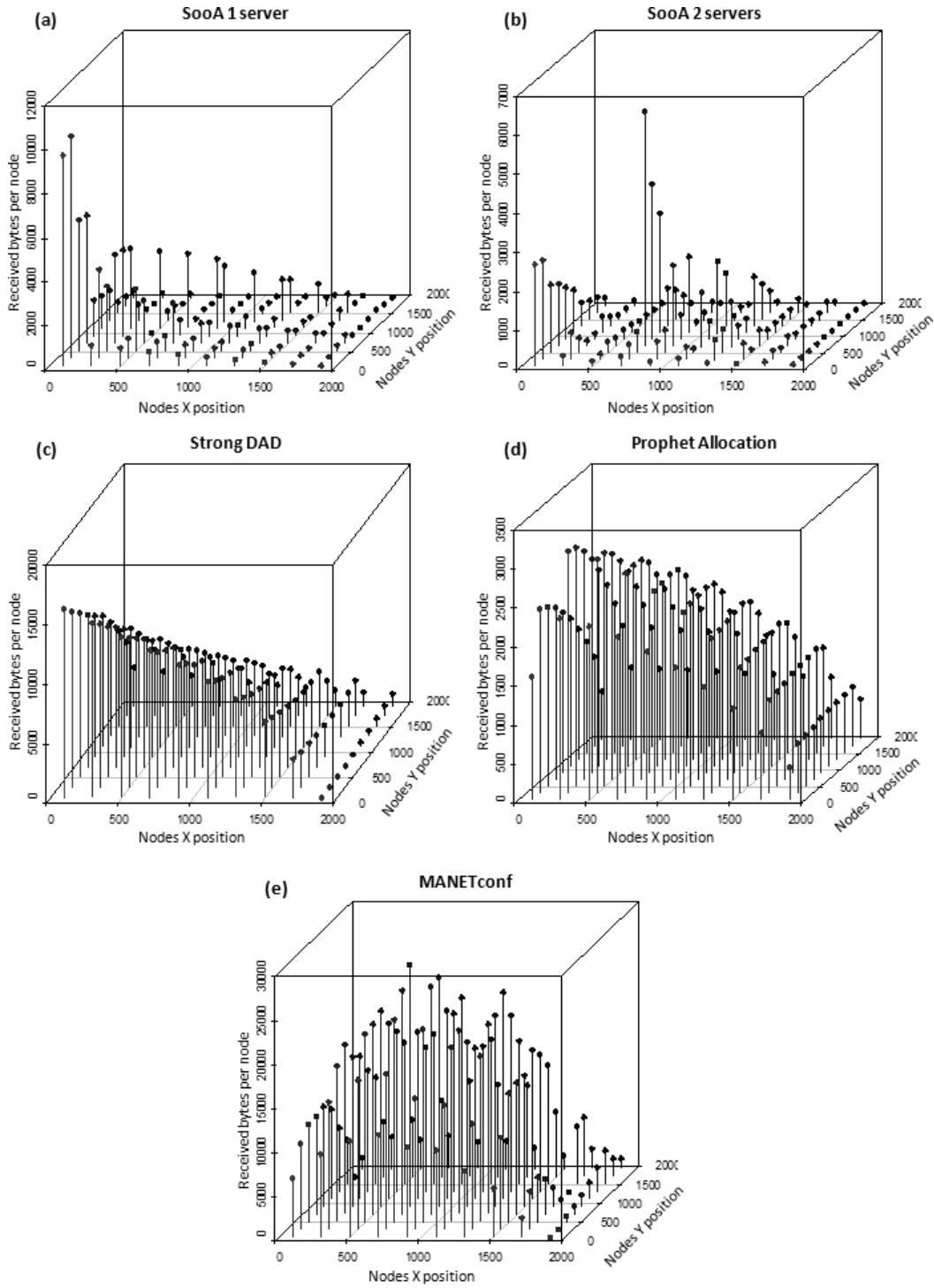


Figure 5.6. Traffic pattern of received bytes in Static scenario.

Prophet Allocation, despite the address assignment procedure, makes the periodic broadcast announcement, which is used, for example, to handle situations of networks merging and/or partitioning. It contributes to the high and invasive traffic as we can see in the graphic of Figure 5.6(d). However, since the broadcast messages are only used in the announcement procedure, the traffic generated by Prophet Allocation is much lower than, for example, Strong DAD. Moreover, the corner and border nodes have a lower overhead of received bytes because they have fewer neighbors than the middle nodes.

As a hybrid approach, the protocol MANETconf implements broadcast-based DAD procedure. Differently from Strong DAD, it executes only one DAD turn. But the main drawback of this solution is that it requires all nodes to reply the DAD requests. In addition, it also implements a flooding-based procedure for allocation tables' synchronization, which contributes to the high overhead, as we can observe in Figure 5.6(e). As well as Strong DAD, MANETconf operates only during the nodes configuration phase and, consequently, the last nodes have a very low traffic overhead. Due to the reduced number of neighbors of the corner and border nodes, the number of received bytes of these nodes is lower than the ones positioned in the middle of the grid.

By the analysis of the traffic pattern of the simulated protocols, we can state that SooA is the less invasive protocol due to the mostly unicast operation. Protocols with broadcast-based functionality both for address allocation or maintenance, tends to spread high overhead to all nodes within the network. Moreover, we can also conclude that the deployment of more servers does help SooA to improve its performance.

5.3.6 Addressing Space

In this last phase of our experiments, we varied the addressing space. This means that the protocols were simulated with different amounts of addresses available for assignments. A summary of the simulation parameters in these experiments is presented in Table 5.3.

With these simulations, our goal was to evaluate the protocols performance when operating with reduced resources. Theoretically, stateless solutions which implement random selection of addresses may have problems with limited resources. Otherwise, stateful solutions may

not face problems with it when the resources' tracking is properly implemented. In this experiment the addressing space was varied from 100% to 500% of necessary addresses to configure all nodes in the network. Results of bigger addressing spaces presented no significant modifications and because of it were not presented and discussed.

Table 5.3. Summary of parameters for Addressing Space simulations.

Parameter	Static scenario	Dynamic scenario
Number of nodes	100	50
Addressing space	100%, 200%, 300%, 400% and 500%	
Number of SooA servers	1 to 2 servers	1 server

As illustrated in the graphics of Figure 5.7(a) and 6.7(b), for Static scenario, the traffic overhead generated by the protocols was not influenced by the variation in the addressing space. The traffic for MANETconf kept the same pattern identified in the previous discussed results. Prophet Allocation also maintained the same behavior by receiving approximately twice the amount of sent bytes due to the periodic broadcasted announcement. In addition, from the values of SooA we can reaffirm that the protocol have its performance improved by the deployment of the second server.

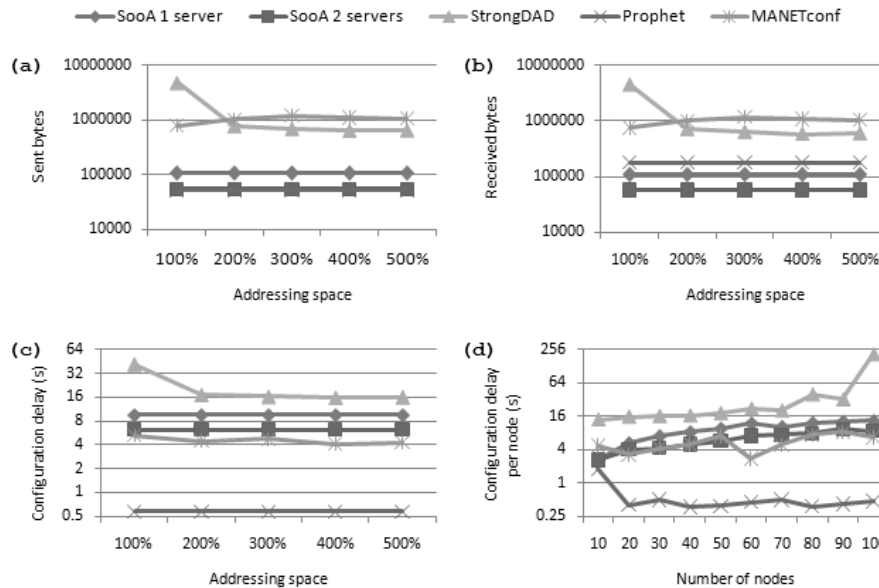


Figure 5.7. Addressing space in Static scenario: (a) sent bytes; (b) received bytes; (c) configuration delay; and (d) configuration delay per node with addressing space of 100%.

The only representative modification we can observe in Figure 5.7(a) and 6.7(b) is related to the performance of Strong DAD when the addressing space was set to 100%. The reduced performance of Strong DAD is due to the fact that the lower is the addressing space the higher is the probability of conflicts, consequently, the more the number of nodes started in the network the higher the number of conflict occurrences.

MANETconf also generated conflicts in these simulations. However, this protocol is able to detect the conflicts locally, through the allocation tables which exist in each node. Therefore, only a small percentage of conflicts are detected by DAD which reflects in a lower traffic for reconfiguration. On the other hand, Strong DAD does not have local mechanisms to detect address conflicts. It means that the conflict detection will occur after the execution of DAD procedure, which floods the network with broadcasted requests. Consequently, the exceeding traffic overhead for Strong DAD when operating with 100% of addresses is due to the reconfiguration of conflicting information.

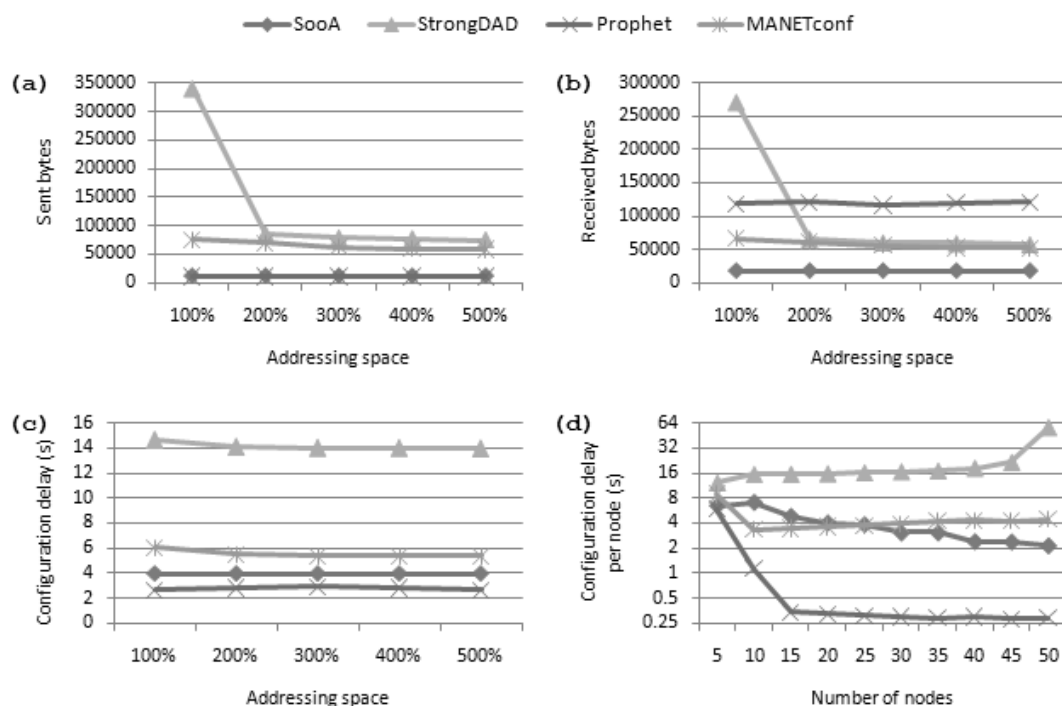


Figure 5.8. Addressing space in Dynamic scenario: (a) sent bytes; (b) received bytes; (c) configuration delay; and (d) configuration delay with addressing space of 100%.

As indicated in Figure 5.7(c), the average configuration delay was also maintained in the same level of the results obtained with no variation of the amount of addresses. Again, only Strong DAD faced more difficulties with a reduced range of addresses. The average configuration delay of Strong DAD, with 100% of addresses, was increased in more than 260%, shifting from the average of ~15 seconds to ~40 seconds.

Moreover, in Figure 5.7(d), we can visualize that the reduced addressing space have more influence over Strong DAD's when it's configuring the last nodes. It is clear that the more nodes already configured in the network the lower is the amount of available addresses for further allocation. Consequently, the probability of selecting an already allocated address for configuring the last nodes is very high and more conflicts may be generated.

As well as in Static scenario, protocols did not experiment representative changes on varying the addressing space in Dynamic scenario. The traffic with different addressing spaces can be visualized in the graphics of Figure 5.8(a) and 6.8(b). Once again, the performance of Strong DAD with an addressing space of 100% it the only different behavior we can identify in protocols' operation. The reason of this behavior is the same than above explained: the number of address conflicts. The reduced addressing space inflicted a higher number of conflicts, and the traffic was intensified in order to find an available address.

Also in the average configuration delay was not influenced by the number of available addresses, as illustrated in Figure 5.8(c). Although, the main difference between Static scenario and Dynamic scenario in this experiment is related to the average configuration delay. Strong DAD, despite generating more traffic with addressing space of 100%, the delay kept the protocol's average: ~15 seconds. It is due to the shorter network's diameter. This way, the new node receives the negative replies for duplicate address earlier and reinitiates the DAD procedure with a short delay. Consequently, the average delay is lower when compared to Static scenario.

In the graphic of Figure 5.8(d) we can observe the behavior on the average configuration delay of the protocols with addressing space of 100%. Prophet Allocation and SooA present a decreasing delay. The higher delay in the beginning of the simulation is because some nodes were isolated in the network. When more nodes start in the network, the

configuration delay decreases. Strong DAD has an increasing average delay, since with the reduced number of available addresses more conflicts are likely to happen.

5.4 Address Pools Allocation

This section presents the details of the experiments performed to evaluate the proposed mechanisms to control addresses' pools distribution in dynamic networks. These experiments were executed to verify correct protocols' operation. Conducted experiments also give details regarding proposals' performance when distributing pools of addresses.

5.4.1 Implemented Protocols

As no address's pools distributions mechanisms are used in Internet, because high level networks' addresses are statically allocated in an offline procedure, no other solutions were implemented to be compared with proposed allocation disciplines.

Both proposed allocation techniques were implemented (Consecutive allocation with Neighbor Reservation and Spread Allocation with binary division based reservation) and to have a base value for comparison with such mechanisms a consecutive allocation with no pools distributions discipline was also considered since it would be the worst case of dynamic allocation in routing table increase perspective. To all evaluated mechanisms the requesting network identification was generated randomly until have all pools allocated.

5.4.2 Simulation Scenarios

The three evaluated techniques were evaluated with a different amount of available pools to be distributed among the networks, as presented in Table 5.4. The number of requesting networks is also changed to verify how the distribution mechanisms would deal with situations where a higher number of networks are trying to obtain pools and, consequently, the possibility of having consecutive pools allocated to different networks is higher.

Table 5.4. Summary of simulation parameters for pools allocation evaluations

Parameter	Values
Number of available pools	25, 50, 100, 200
Number of requesting networks	5, 10, 15, 20, 25
Reserved pools (for reservation discipline only)	1 to 22

In Consecutive allocation with Neighbor Reservation discipline a third parameter might also be changed: the number of reserved pools when a network has its first pool allocated. However the comparison with the other proposed mechanisms only considers the results obtained with the reservation of one pool.

5.4.3 Evaluation Metrics

To evaluate the performance and behavior of the implemented pools distribution mechanism under the described scenarios, we considered the following metrics:

- Routing table size: number of entries existent in the routing table. To calculate this metric was considered that every time a number of consecutive pools are allocated to the same network it will be necessary create only one entry in the routing table to reference these addresses, consequently, with more consecutive pools allocated the mechanism can have a smaller routing table and the routing performance will be improved;
- Routing table aggregation: percentage of routing table entries that are aggregating routes to the same network. It is calculated by the number of existent routing tables over the maximum number of entries that the routing table would have. The maximum number of entries will depend of the number of available pools in each experiment, i.e. the maximum number of entries to experiments with 200 pools is 200. Entries are considered as aggregated when at least two consecutive entries for the same network are found. This indicator informs the aggregation a mechanism can apply to the routing tables and, consequently, how benefic can be to routing.

5.4.4 Pools Distribution

All implemented pools distribution mechanisms were evaluated changing the number of available pools and also the number of requesting networks. In Figure 5.9 the results regarding the average routing tables' size to performed experiments are presented.

Figure 5.9(a) presents the results obtained by the pools allocation mechanism having 25 Pools and changing the number of requesting networks. It's possible to note that when working with 5 networks requesting 25 pools the consecutive allocation with no discipline has created routing tables with more than 20 entries. When the number of requesting

networks increase the routing table's size will also increase being very close to 25 entries, that is the maximum number of possible entries, when 25 networks requests pools. As the requesting network identification is generated randomly, in some situations not all 25 requesting networks will be selected before exhaust the available pools, because of it the average routing table size is smaller than 25.

Consecutive allocation with Neighbor Reservation of 1 pool has also increased routing tables' size with more requesting networks; however, this increase is lower than with the allocation with no discipline. With 5 requesting networks routing table had about to 13 entries, instead of 20 with no allocation discipline. Only in with 25 requesting networks it reached the same level of the initial value with no allocation discipline, indicating that this allocation mechanism can reduce the routing table size even reserving only one pool for each new allocation.

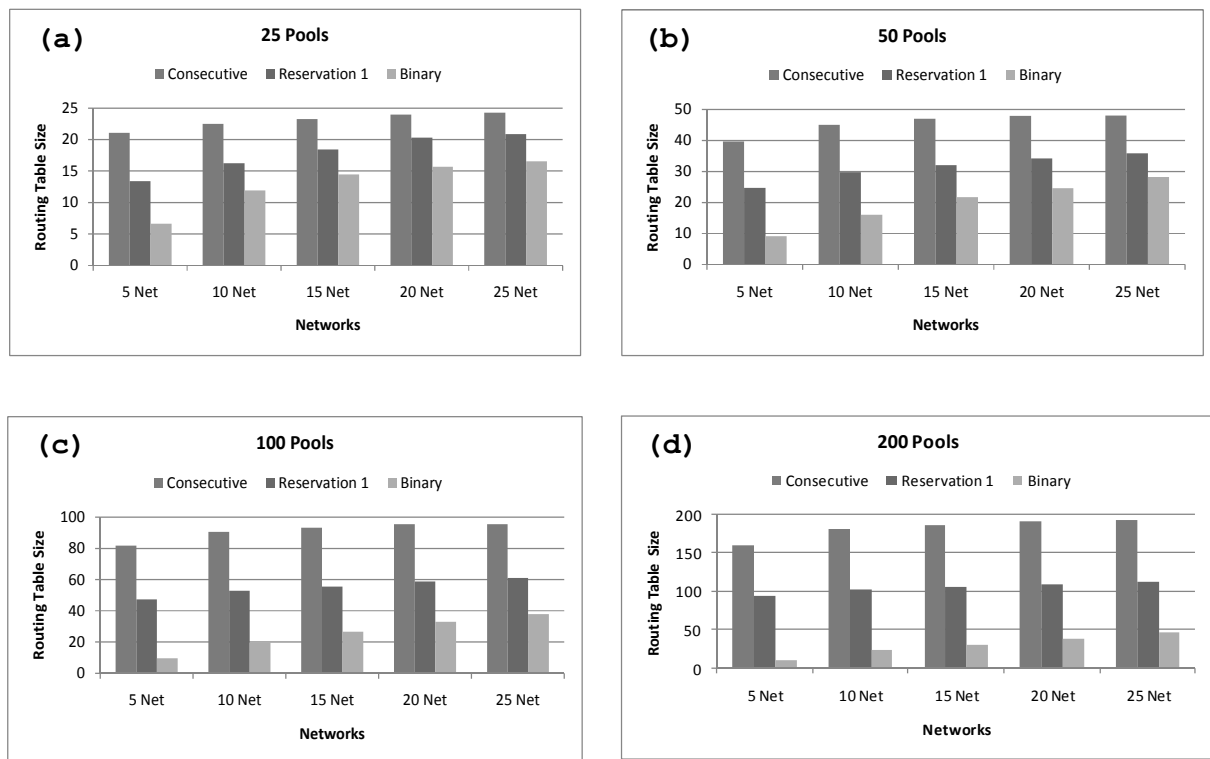


Figure 5.9 – Routing tables' size: (a) 25 Pools; (b) 50 Pools; (c) 100 Pools; and (d) 200 Pools with requesting networks varying from 5 to 25.

Spread Allocation with binary division based reservation had the best results. For 5 requesting networks it had about to 6 entries. In ideal situations it would just create 5 entries,

one to each network, but as requests are not equally distributed to all requesting networks, some of them will need to create more than five entries to store all the information. In this allocation discipline the number of entries in the routing tables also increases with more networks requesting pools, but the amount of information that might be stored is considerably lower than in the other allocation disciplines.

Figure 5.9 (b), (c) and (d) present the results obtained for the experiments with 50, 100 and 200 pools respectively. The results obtained by other pools configurations have basically the same behavior of the results from the distribution of 25 pools, having an increase in the routing table size when more networks request pools. However, when the number of available pools increases the allocation disciplines have a bigger advantage if compared to the allocation with no discipline. In other words, the proposed mechanisms can have a better organization of the routing table when more pools are available because their allocation can be more dispersed. This characteristic makes the increase rate smaller with more pools.

Beyond the understanding of the general behavior of routing tables' sizes to the different amount of pools to be allocated, it would be necessary also evaluate how it behaves when only considering the variation of the number of available address pools keeping the amount of requesting networks. This evaluation can indicate how each mechanism can adapt to networks with more resources in order to optimize address allocation and routing operation.

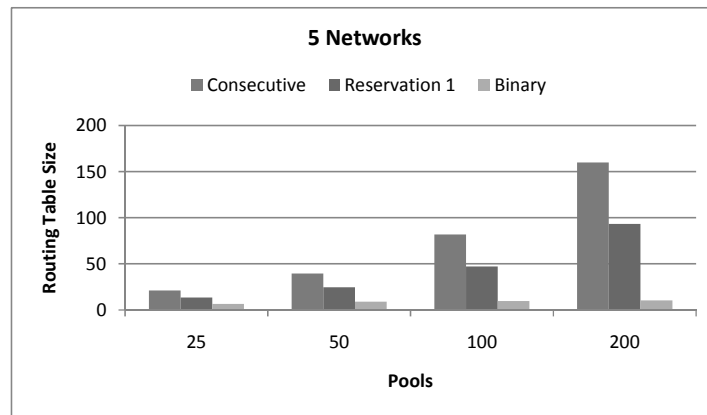


Figure 5.10 – Routing tables' size for 5 requesting networks with available pools varying from 25 to 200.

Figure 5.10, presents the results of the evaluation for 5 requesting networks when the available resources increase from 25 to 200 pools. To the consecutive allocation with no

discipline is possible to see that independent of the number of pools it will continue creating routing tables with a high fragmentation, since only a little part of the allocated pools were grouped. Allocation with one reserved pool to each new allocation also continues increasing with more pools, but in a smaller proportion than presented in allocation with no discipline.

The allocation mechanism which was presented best results about the adaptation to more available resources was the binary allocation. As presented in Figure 5.10, this allocation kept almost the same routing table size independent of the amount of available pools, having a very discrete increase in the information necessary to routing level.

5.4.5 Routing Table Aggregation

Only the number of entries existent in the routing table may make a good representation of the usage of the resources and how the mechanism can adapt or optimize their operation according to the amount of available resources.

To have a representation of the amount of grouped entries in a routing table, the routing table spreading indicator was defined. It's responsible by calculating the amount of routing tables' entries that are not aggregated.

In Figure 5.11 the results regarding the evaluation of the mechanisms are presented and makes possible identify routing aggregation behavior in the selected scenarios. The consecutive allocation basically kept the same routing entries' aggregation to each amount of requesting networks, independent of the number of available pools, i.e. in Figure 5.11 (a) it only had about to 20% of the entries aggregated. In the other experiments (Figure 5.11 (b), (c), (d) and (e)) is possible to see that the results have the same behavior of Figure 5.11 (a), however, the number of aggregated entries decreases when requesting networks increases.

The reservation discipline had a mild increase in the number of routing tables' entries that are aggregated, having nearly to 50% of the routes aggregated to the experiments executed with 5 requesting networks (Figure 5.11 (a)). To the experiments with more requesting networks the increase of the aggregation level was higher; however it doesn't mean that the mechanism had a better adaptation to these situations. Actually this increase indicates a worst performance in experiments with more requesting networks, since the routes

aggregation starts in a lower level and only with 200 pools it is closer to the results obtained with 5 requesting networks.

Binary allocation presents a considerably increase in all the evaluations. It happens because when the number of available pools increases, the number of pools which will be reserved to each network is bigger and the probability of allocating subsequent addresses to the same network is higher.

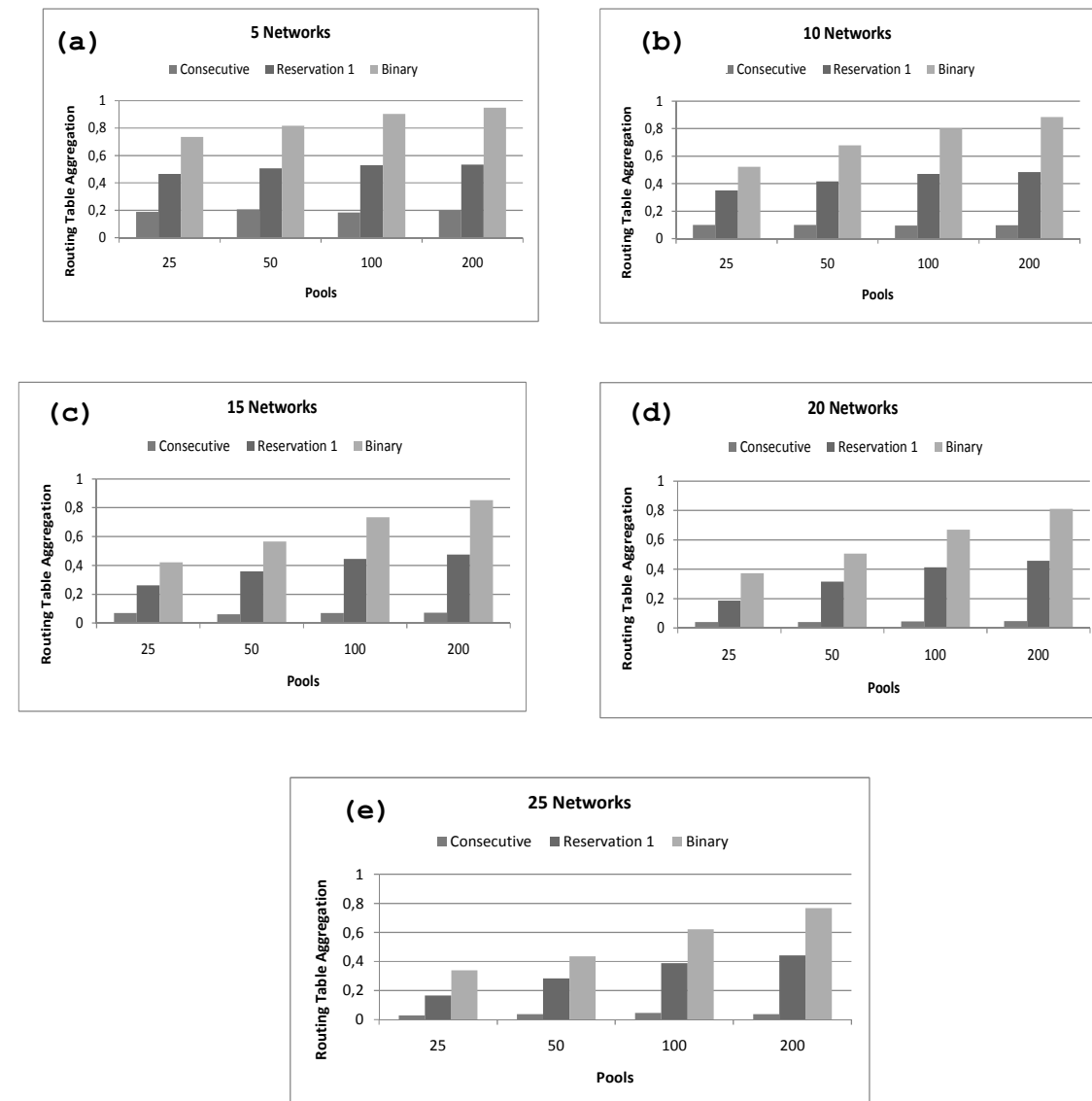


Figure 5.11 – Routing tables' aggregation: (a) 5 networks; (b) 10 networks; (c) 15 networks; (d) 20 networks; and (e) 25 networks with available pools varying from 25 to 200.

Generally speaking the results indicates that with a higher number of requesting networks the aggregation level of the routing tables is reduced. In the case of the binary mechanism this reduction occurs because with more networks requesting pools the number of pools that can be reserved to each network also reduces and, having a higher probability of exhausting the reserved pools and then need to allocate new ones in a separated part of the routing table. To consecutive allocation with no discipline and reservation mechanism with reservation of only one pool, this increase happens because when more networks request pools it's more difficult of having a free consecutive pool since there is no reservation of pools or only one pool is reserved in each allocation.

5.4.6 Reservation Degree

The conducted evaluations of the Consecutive allocation with Neighbor Reservation, only considered one reserved pool per new allocation in the previous experiments. However, it's necessary to check the influence and possible modifications in mechanism operation according to the amount of pools that would be reserved the new allocations.

During the experiments performed to evaluate these modifications, the number of reserved pools was increased until have a stabilized number of routing tables' entries, which indicates that the increase in reservation is not improving mechanism operation. The experiments considered 10 requesting networks to all available pools configurations, since some evaluations with 5 requesting networks didn't follow the same behavior of other requesting networks configuration, as presented in previous section.

The results of the performed evaluations are presented in Figure 5.12. In a general way, the number of routing entries created is reduced when the amount of reserved pools to each allocation is increased. However, when the reservation size is too big, the number of routing entries keeps stable, as it can be observed. This happens because, too much pools reserved to the already allocated networks will restrict the amount of new requesting networks which would be able of get an available pool, limiting the number of accepted networks and of routing table entries.

With 25 available pools (Figure 5.12 (a)) it starts with the routing table with 16 entries in average when only one subsequent pool is reserved to each new allocated pool. When this

reservation degree increases, the amount of information stored in the routing table is reduced and it stabilizes with the reservation of 8 subsequent pools, with an average of approximately 6 entries.

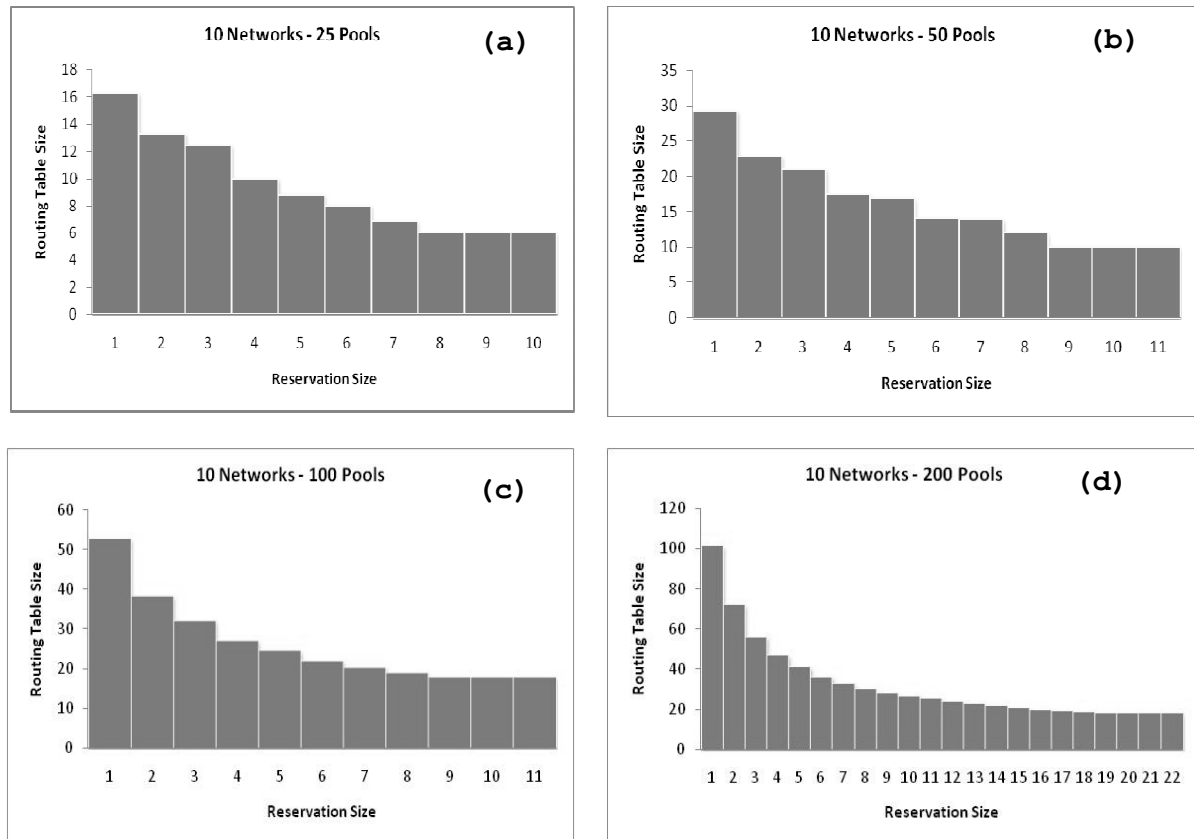


Figure 5.12 – Routing tables' size: (a) 25 Pools; (b) 50 Pools; (c) 100 Pools; and (d) 200 Pools with requesting networks varying from 5 to 25.

Figure 5.12 (b) presents the results of the experiments distributing 50 pools. It starts near to 30 entries and reduces to an average of 10 entries when 9 pools are reserved. This number of entries is kept to other reservation levels. Experiments with 100 pools (Figure 5.12 (c)) also stabilize their routing table size with 9 reserved pools. The difference is in the number of routing entries that is bigger than with 50 pools, as already expected.

Experiments with 200 pools have presented the biggest variation on pools reservation until have routing tables' size stable. As indicated in Figure 5.12 (d), only with a reservation of 19 pools to each allocation the result was maintained for the other allocations. This higher reservation occurs because having more available pools and keeping the number of

requesting networks, it would be necessary have bigger reservations to limit the number of accepted networks, as explained.

5.5 Intra-Domain Configuration Negotiation

Intra-Domain Configuration Negotiation can be performed in many levels and configure several characteristics of the nodes within the same domain. To illustrate the ability of the proposed mechanism of dealing with devices' characteristics configuration a set of experiments was performed. The results obtained from this evaluation concentrates on Intra-Domain negotiation for routing configuration, presenting and discussing the advantages of using an automatic mechanism to dynamically configure the environment.

5.5.1 Mechanism Implementation

From the proposed Intra-Domain mechanism (here named as XDNP), environment and decision dissemination procedures were fully implemented. Local and Final decision algorithms were also implemented but only considering that routing information would be received, restricting the possible results. Coordinator selection was simplified and only considers the operation time of the nodes; therefore, first DA Node which starts to operate is selected as Coordinator during the whole simulation period.

5.5.2 Simulation Scenarios

To evaluate the ability of collecting information and selecting the appropriate configuration for a network according to its capabilities a scenario only with stationary nodes was selected since nodes mobility absence would not compromise the mechanism operation allowing its functionalities validation.

Table 5.5. Summary of simulation parameters for Intra-Domain Negotiation evaluations.

Parameter	Values
Simulation area	2000m ²
Simulation time	4000 seconds
Number of nodes	100 nodes
Decision delay	30s, 60s, 90s and 120s
Number of Decision Nodes	1 to 10
Available Routing Protocols	AODV and DSDV
MAC protocol	802.11

Table 5.5 presents the simulation parameters of the conducted experiments. Three of these parameters are extremely important: decision delay, number of decision nodes and routing protocols. Decision delay is responsible by configuring the periodicity that the Coordinator will create new decision, the number of DA Nodes will distribute the decision capabilities over the network and the routing protocols are two possible configurations that the decision algorithm can receive to select the best option to the network. At least one of these two protocols will be present in all nodes of the network and the network, which is initially with no configuration of routing protocol will detect the one that can be used by all elements.

5.5.3 Evaluation Metrics

To evaluate the performance and behavior of the implemented pools distribution mechanism under the described scenarios, we considered the following metrics:

- **Traffic Overhead:** traffic generated by all supporting protocols executed to the communication in the environment. Decision and routing traffic data are considered in packet and bytes during the whole simulations period to allow a better understanding of the influence from decision mechanism when it's configuring the network or maintaining the configurations;
- **Decision Traffic Overhead:** the traffic generated by the decision protocol to collect the environment operational information and distribute calculated decisions;
- **Routing Traffic Overhead:** the traffic generated by the selected routing protocol to distribute necessary information and maintaining created routes;
- **Configuration Delay:** the average time a node needed to get configured with a new calculated decision. This means the difference between the time the node got configured and the time it disseminated its operational information; and
- **Application throughput:** average throughput obtained by a FTP application running in the network. It's expected that with more DA Nodes the control traffic will be more concentrated and the application would have a better performance.

Intra-Domain Configuration evaluations tries to capture three different information levels to understand the influence it would have in the network in routing and application layer perspective. Based on that, the benefits obtained by using this mechanism are discussed.

5.5.4 Traffic Overhead

Traffic overhead was evaluated with the objective give support information to identify how the network is influenced by both, decision and routing traffic, since both would be the mandatory traffic to any other decision which would be executed in the network.

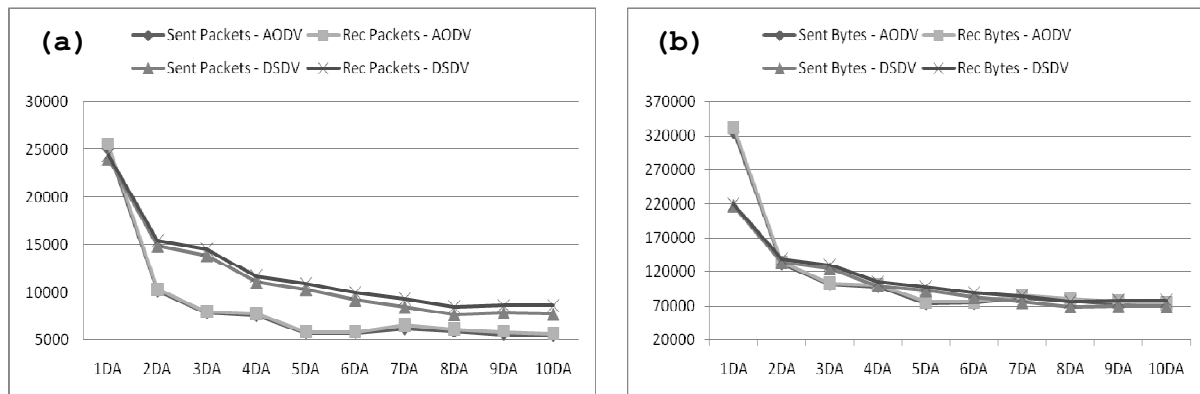


Figure 5.13 – Traffic Overhead for decision delay in 30 seconds and varying the number of DA Nodes: (a) packets and (b) bytes

Figure 5.13 the results of the traffic overhead varying the number of DA Nodes in a network configured to make decisions in each 30 seconds are shown in packets and bytes. Figure 5.13 (a) presents the results of the experiments in packets. It can be seen that the amount of packets transmitted when AODV was selected as the routing protocol was lower than with DSDV. It's an expected result since AODV operates on-demand while DSDV works based on a proactive mechanism. Network running AODV was able of stabilize the total packets faster than with DSDV. In the first case a small variation is observed after 5 DA Nodes in the network, however in DSDV it only happens when 8 DA Nodes are operational in the network. The explanation to this behavior comes again from the different operation of the routing protocols. In the case of AODV as the routing traffic is more concentrated in certain periods and regions of the networks (the ones which are trying to create routes). On the other hand, in DSDV the traffic is more constant and influences all the nodes in the network, consequently, they need to “compete” with decision traffic during

the whole network operation. Only when a high number of DA Nodes are present in the network and the decision traffic is confined in each decision area, DSDV will stabilize.

On Figure 5.13 (b) are presented the results related with sent and received bytes. It's possible to note that these results have a completely different behavior from Figure 5.13(a). It's caused by the information that is transmitted by each protocol in their packets. AODV transmits fewer packets than DSDV, but AODV packets are bigger than DSDV since it might contain all the information related with validity and loop-free routes control, what is not necessary to DSDV because it periodically updates the routing tables. Regarding the amount of transmitted and received bytes, protocols don't have a significant disparity as in transmitted packets, especially when more DA Nodes are present in the network. The only exception occurs with 1 DA Node. As the number of packets from AODV and DSDV to experiments of 1 DA node was very close and AODV have bigger packets, it's expected that initially it would transmit much more information than DSDV.

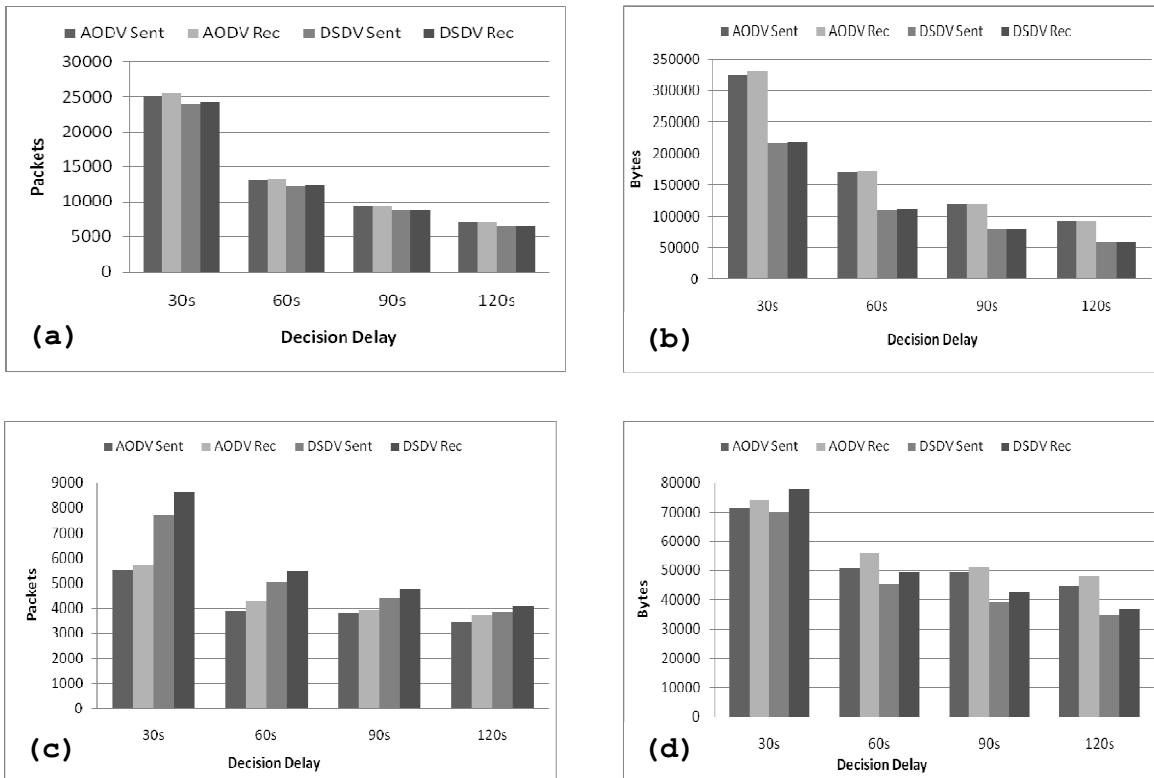


Figure 5.14 – Detailed traffic overhead: (a) packets for 1 DA; (b) bytes for 1 DA Node; (c) packets for 10 DA Nodes; and (d) bytes for 10 DA Nodes.

A detailed evaluation from the traffic overhead to 1 and 10 DA Nodes is presented in Figure 5.14. Other amounts of DA Nodes had the same behavior of 10 DA Nodes experiments, and are not presented. Figure 5.14 (a) and Figure 5.14 (b) are related with 1 DA Node traffic overhead. The first presents the results of packets transmission varying the Decision Delay which will be applied to the network. Generally speaking the decision delay has influenced in the amount of transmitted packets, as was expected. With a bigger decision delay network's elements need to transmit fewer packets than with a small delay, what also influences in the routes discovery/update rate of the routing protocols. As explained to Figure 5.13 (a), AODV need to transmit fewer packets because of its on-demand operation. This behavior is maintained independent of the decision delay applied in the network.

In Figure 5.14 (b) the traffic overhead in bytes of 1 DA Node is presented. The amount of transmitted and received bytes is reduced as occurred with transmitted/received packets. It's a completely expected result, since with less packets, traffic transmitted also should be reduced. Additionally, AODV has occupied more the network, with a higher amount of sent and received bytes, as expected by the results presented in Figure 5.13 (b).

In Figure 5.14 (c) and (d) the results related to traffic overhead to networks with 10 DA Nodes are presented. When more DA Nodes are running in the network is possible to observe that a completely different behavior is presented by traffic pattern. Regarding traffic overhead in packets (Figure 5.14 (c)), with 10 DA Nodes AODV has transmitted less packets than DSDV in all decision delays, but in higher delays the results are closer. When the decision delay increases the total number of transmitted and received packets continues to reduce, as with only one DA Node, but this reduction has a more modest rate. Bytes traffic overhead has also changed in 10 DA Nodes evaluations, as presented in Figure 5.14 (d). With more DA Nodes the amount of transmitted and received bytes is very close from both protocols but with higher decision delays DSDV becomes more efficient in network utilization than AODV. It happens because when a decision need to be transmitted, DSDV already have a valid route, on the other hand, when the period of time between two decisions is too long AODV routes has expired and a new route discovery should be started.

5.5.5 Routing and Decision Traffic Overhead

The previous section presented the results about the total traffic overhead; however it's important to keep in mind that two different protocols are operating at the same time: the intra-domain negotiation protocol, responsible by collecting information and make decisions, and the routing protocol which was selected by the negotiation protocol. Considering this it's necessary to separate the specific traffic of each protocol to see the impact of each particular mechanism.

In Figure 5.15 the division of traffic overhead from routing and decision protocols is indicated. These experiments considered a decision delay of 30 seconds and ranged the number of DA Nodes from 1 to 10, allowing the identification of the amount of traffic generated by each protocol when decision is more distributed.

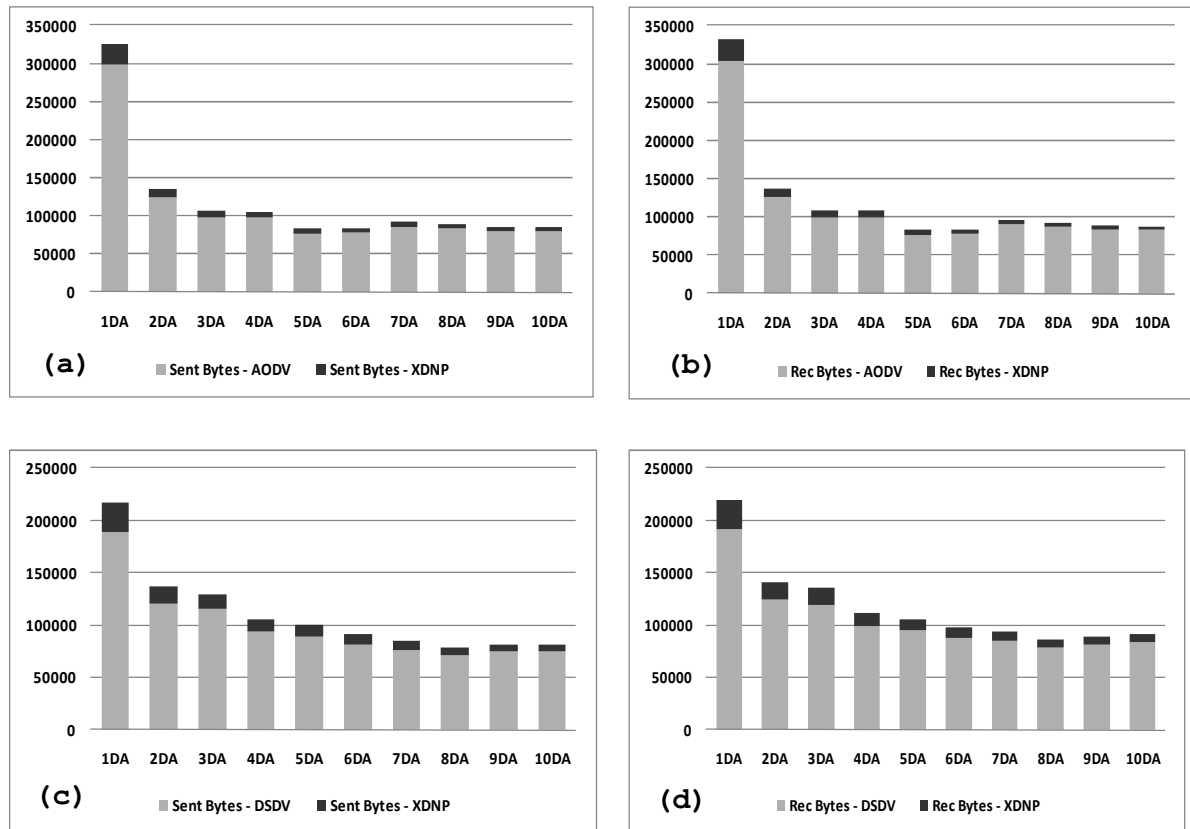


Figure 5.15 – Traffic overhead from Routing and Decision protocols with a decision delay of 30 seconds and varying the number of DA Nodes: (a) Sent bytes from AODV and XDNP; (b) Received bytes from AODV and XDNP; (c) Sent bytes from DSDV and XDNP; and (d) Received bytes from DSDV and XDNP

Independent of the specific routing protocol which is selected by the decision mechanism, the traffic overhead from both protocols is reduced adding new DA Nodes in the network. Another interesting questing that can be identified in graphics of Figure 5.15 is that when more DA Nodes are deployed in the network the proportion of decision traffic over the routing traffic decreases. This decrease is caused by the reduction of the distance from a Network Node to a DA Node. Having few DA Nodes, Network Nodes will need more intermediate elements to send and receive information from the DA Nodes. When this number increases, Network Nodes are capable of communicating with a DA Node which is closer, reducing the network utilization.

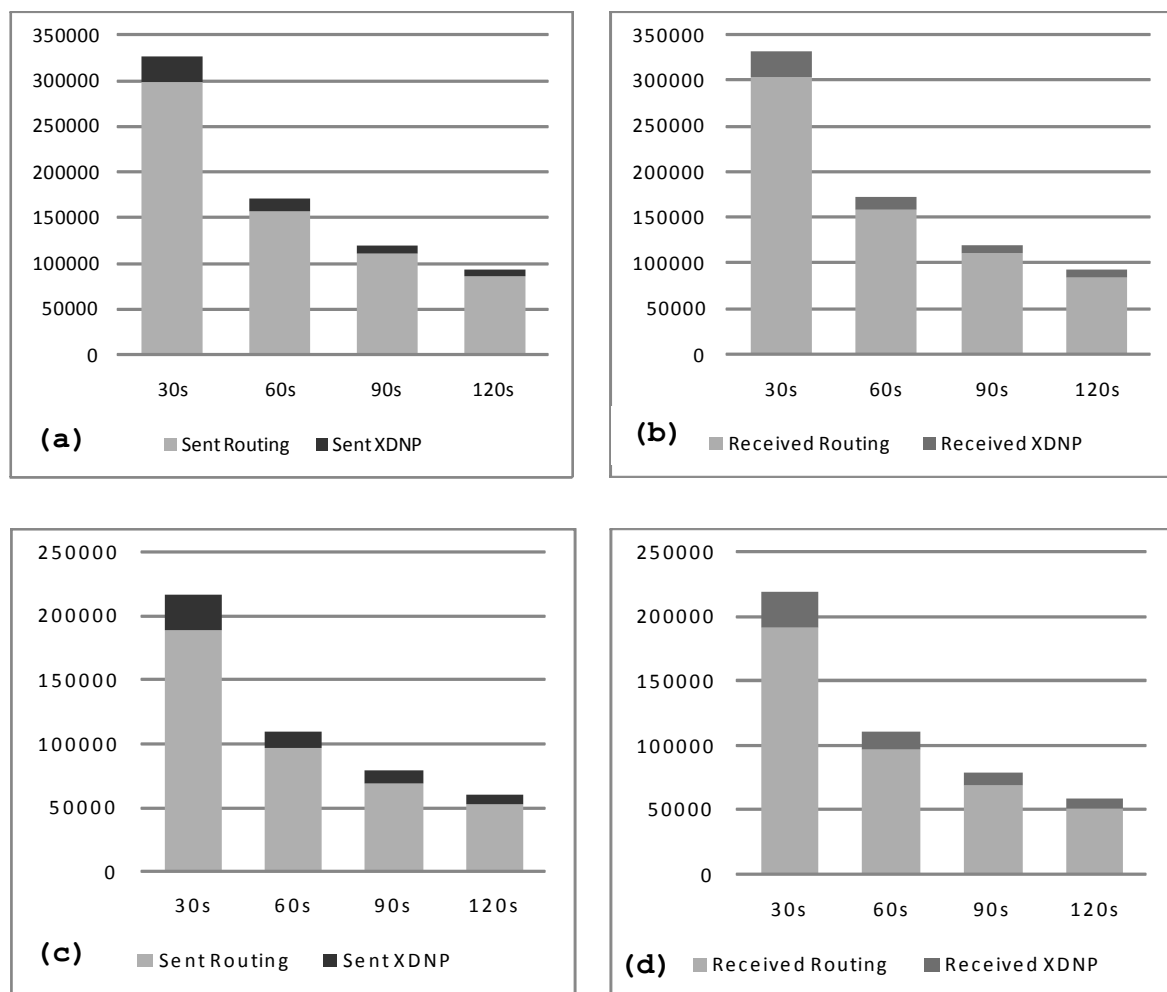


Figure 5.16 – Traffic overhead from Routing and Decision protocols with one DA Node and varying the decision delay: (a) Sent bytes from AODV and XDNP; (b) Received bytes from AODV and XDNP; (c) Sent bytes from DSDV and XDNP; and (d) Received bytes from DSDV and XDNP

The influence of the increase in the decision delay was also evaluated. Figure 5.16 presents the results of the experiments performed to the evaluations with only one DA Node is operating in the network. Figure 5.16 (a) and Figure 5.16 (b) present the results obtained when AODV is selected as the routing protocol. As expected the traffic overhead from routing and decision protocols is reduced when the decision delay is increased. The same happens with DSDV, as can be seen in Figure 5.16 (c) and Figure 5.16 (d), but in the case of DSDV, as this protocol uses fewer resources from the network, the amount of decision traffic represents a bigger part of the total traffic than in AODV, as already explained for other routing and decision traffic overhead experiments.

Despite the decrease of the traffic with higher decision delays in the network, the percentage of decision traffic is maintained the same. This happens because, as the experiments only consider one DA Node, the distance from the Network Nodes to reach the DA Node is not changed during the experiments, and then, fewer decisions are transmitted but each one maintains the same “cost” to the network.

5.5.6 Configuration Delay

Traffic overhead indicates the influence that the proposed Intra-Domain negotiation mechanism and its decisions would have in network resources utilization and network’s communication traffic pattern. However it’s also important to know the influence it would have in the common nodes operation. Processing and communication impact in the network nodes is not so representative, since they don’t participate of the decision process, since network nodes are only responsible by verifying their configurations and notify the DA Node it’s bound. Even DA Nodes and the Coordinator are not expected to have their operation and performance compromised, since they would work with summarized information. Only situations where a really large number of network nodes are connected to one DA Node would be an adverse situation to nodes operation.

In this context, configuration delay is the characteristic which will influence all the networks’ elements independent of the function it will be executing in the network. Figure 5.17 presents the results obtained for devices configuration delay in each one of the defined decision delays to from 1 to 10 DA Nodes.

Configuration delay follows the same behavior to both routing protocols. AODV (Figure 5.17 (a)) and DSDV (Figure 5.17 (b)) reached very similar results, having an average of ~ 3 seconds longer than the configuration delay to configure the nodes present in the network. This additional time is caused by the propagation time and the messages forwarding process necessary to reach all the elements in the network.

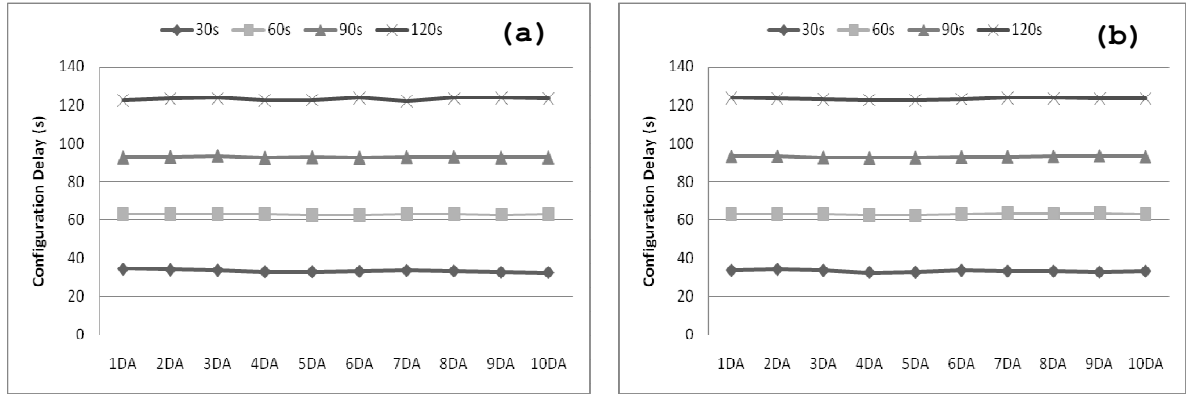


Figure 5.17 – Average Configuration Delay varying the Decision Delay: (a) AODV; and (b) DSDV.

The amount of DA Nodes operational in the network has not influenced in the results of nodes configuration delay. It was already expected because independent of the amount of DA Nodes the configurations need to be transmitted over all network's elements. The difference is that instead of receiving the information directly from the Coordinator the Network Nodes will receive information from a closer DA Node which has obtained the decision from the Coordinator.

5.5.7 Application throughput

Previous experiments has presented the influence that the intra-domain decision protocol and the selected routing protocol has in the network resources utilization through the traffic overhead analysis and also the time necessary to configure the nodes and make them capable of communicate with other in the network.

However, the benefits of an automatic configuration mechanism also influence networks' users experience when using an application. Figure 5.18 presents the throughput of an FTP application running in a network with the Intra-Domain mechanism operating.

The application throughput follows the same tendency for both routing protocols when the number of DA Nodes is increased. With more DA Nodes the application can have a better transmission rate and then improve communication link's utilization. This improvement occurs because having more DA Nodes in the network the traffic generated by the negotiation protocol is confined to DA Nodes' control area, i.e. the group of elements which are bound to the DA Node. With the negotiation traffic influencing fewer nodes, application has a lower “background traffic” to compete with and can have a better performance.

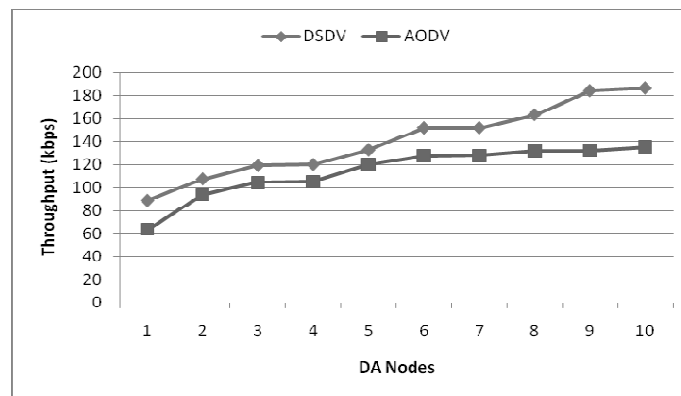


Figure 5.18. Application throughput for AODV and DSDV networks varying the number of DA Nodes

Figure 5.18 also indicates the different in application performance according to the selected routing protocol. The application has reached better transmission rates with DSDV than with AODV. Experiments were conducted in a stable topology where DSDV has the advantage of already have routes calculated to all destinations when the application needs to transmit information. In the case of AODV, as it works reactively, routes must be redefined periodically and during this process the application cannot transmit any information.

5.6 Inter-Domain Negotiation

Inter-Domain Policies negotiation is responsible by giving to networks the ability of defining with each other the characteristics of a desired communication which was not previously configured, enforce the necessary rules to allow this communication and analyze environments' characteristics to adapt to any necessary modification.

Conducted experiments try to validate the correct operation of the implementation of the proposed mechanism and evaluate the necessary time to perform negotiation and

configuration of a new communication. More complete evaluations are planned to be done to complement these results.

5.6.1 Mechanism Implementation

Inter-Domain Policies Negotiation mechanism was only partially implemented in this first moment. Announcement and Negotiation phases were implemented in order to allow the networks to contact each other and create new policies to support communications' specific requirements. No monitoring activities were implemented to adapt the agreements to environment modifications.

The domains were configured with twenty supported requirements (the same to all domains) and every time a new requirement had to be request, one of these ids was generated randomly. No experiments were executed to evaluate the situations where the requested requirements were not accepted because, for example, the time necessary by the mechanism when one requirement which was rejected and in the first renegotiation a compatible level is found would be the same of two requirements which were directly accepted. Renegotiation would influence in generated traffic, which will be studied in future evaluations.

Enforcement process was simplified since NS-2 doesn't offer support for controlling routing activities. Because of it the proposed policy framework was not fully implemented. Policies storage and the negotiator were implemented to control the policies created. Instead of enforcing policies through the Agent as defined in the architecture, an extension on IP protocol was implemented to access a secondary table which is feed by the negotiator and represents the policies. When no "conventional" routes are found the secondary table is consulted to check if a compatible entry is found and allow the communication.

5.6.2 Simulation Scenarios

As performed evaluations intended to validate the correct operation of the mechanism and confirm its ability of dynamically create policies to control the access of new networks, only two parameters were considered to give some initial insights about mechanism's performance. Used parameters are presented in Table 5.6.

To evaluate the operation of the implemented Inter-Domain Policies Negotiation Mechanism, the negotiation delay was considered. Negotiation delay represents the average

time a Requesting Domain needed to get configured with the necessary policies to guarantee the correct access to the requested resource.

Table 5.6. Summary of simulation parameters for Inter-Domain Policies Negotiation Protocol.

Parameter	Values
Exchange Mode	Direct Contact and Direct Hop-by-hop
Number of Intermediate Domains	0 to 10
Number of Requirements	1 to 5

5.6.3 Negotiation Delay

Negotiation delay is an important characteristic since it will influence the time necessary to provide access to other networks. Many communication characteristics may change the time necessary to complete the negotiation process. Depending of the exchange mode that networks will use to negotiate, the distance from the Requesting Domain to the Destination Domains and the number of requested characteristics this time will change, as presented in Figure 5.19 and Figure 5.20.

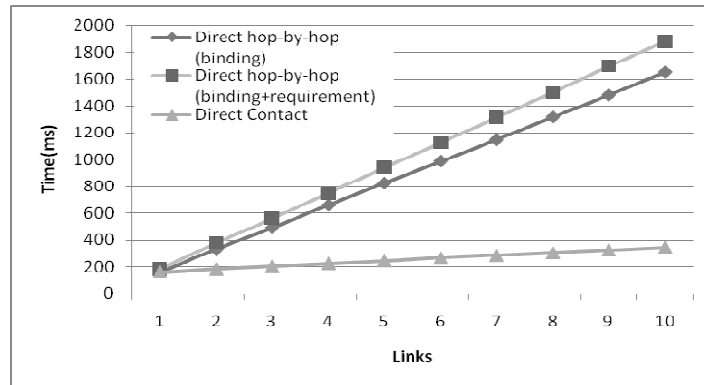


Figure 5.19. Negotiation delay using Direct hop-by-hop (only Binding and Binding and one requirement) and Direct Contact Negotiation modes and varying the number of link between the domains.

Figure 5.19 shows the results obtained in simulations where the number of intermediate routers between the negotiating domains is changed. These experiments were performed considering three possible situations of how this communication can be performed: direct hop-by-hop mode only with binding negotiation, direct hop-by-hop mode with binding negotiation and one secondary requirement and direct contact only with binding negotiation.

As expected negotiations using direct contact have the lower negotiation delay since they just need to negotiate between the Requesting and the Destination Domain. The transit domains just forward the information and the influence they will have is just because having bigger paths the transmission time will increase. However, as no allocation is done in transit domains, this increase is quite small.

Contact hop-by-hop negotiation mode has a completely different pattern. As all the transit domains should be contacted in order to also accept the necessary requirements, instead of only increase the extra transmission time, to each intermediate domain the negotiation delay will increased the time a whole negotiation, what gives a much more substantial increase.

Another evaluation performed uses a contact hop-by-hop negotiation mode with a secondary requirement being requested during the announcement. The results indicate that, as expected, negotiations having more requirements would have a higher negotiation delay than negotiations which has just created a basic agreement during the binding negotiation.

Second evaluation only considered the direct contact negotiation mode with one link between the domains in the experiments and changed the number of requirements which might be requested by the Requesting Domain.

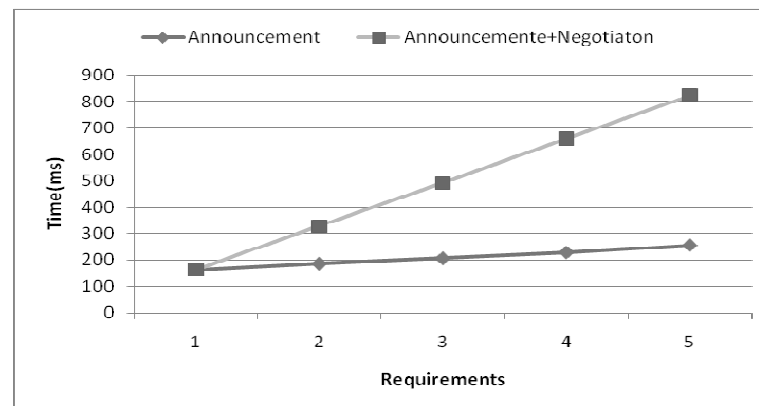


Figure 5.20. Negotiation delay using Direct Contact negotiation with only one link between the domains and varying the number of requirements requested.

When varying the number of requirements two possible situations were simulated: requirements directly requested during the announcement or individually requested during the negotiation, as presented by Figure 5.20.

In both situations the negotiation delay was increased, since with more requirements it's expected that the networks would need more time to check if the requirements are available and to create the necessary policies to each one of them. Requirements that are transmitted on announcement phase have a lower influence on negotiation delay, since in only one contact all requirements are checked and necessary policies created.

Transmitting the requirements only in negotiation phase will demand new connections to request the desired characteristics and create the policies. When the number of requirements increases the negotiation delay also increases, but in a proportion much higher than negotiations during announcement. This happens because, as explained, new connections might be created to each new requirement, what basically consumes the same time of the initial contact made during binding negotiation.

5.7 Summary

This Chapter presented the results gathered from a series of experiments done to evaluate the proposed mechanism in order to verify the correct operation of them and give an understanding of their benefits compared to other solutions. Through addressing mechanism analysis, it's possible to conclude that SooA is a good proposal for addressing in autonomous network. It performed well all the basic operations for allocation and management of addressing resources. It is clear that the protocol's performance can be improved by polishing the allocation and management mechanisms of SooA and also by implementing advanced functionalities to enable it to operate under more dynamic scenarios. However, SooA is able of adapting to network diversity in many perspectives (new clients, new servers, amount of addresses, etc.).

When compared to the other solutions, SooA demonstrated to be a promising approach. It is because the proposed protocol reached very good performance when compared to the other evaluated protocols. Considering that SooA operated also the management of resources, the overall overhead generated by the protocol is very low, and it's also applicable to a broader variety of networks' environments.

Pools allocations mechanism has compared different distribution disciplines to verify the modifications if would bring to addresses allocation when networks start to operate

unpredictably and should be supported with a set of available addresses. However, only allocation doesn't guarantee that the network operation will be really improved with this automatic allocation, even it facilitating networks' configuration. Evaluated disciplines has presented a substantial difference in their operation, considering the routing table sizes and aggregation addresses allocated to the same network, reducing as much as possible any eventual interference it would have on routing. Pools allocation proposals reached very promising results, substantially reducing routing tables' sizes and improving routes aggregation at a fairly representative rate when compared unplanned allocations.

Regarding the Intra-Domain configuration evaluations presented the results of the impact this mechanism have over the network considering its traffic overhead. Traffic overhead from selected routing protocols was also presented to indicate the difference to the network that a decision can have and the possible communication improvements it can represent.

Intra-Domain configuration results also discusses the impact of changing the routing protocol used by the nodes and consequently the performance improvement that is possible to have adapting network's configuration according to nodes capabilities that are discovered dynamically instead of only consider an initial configuration, in this case, no routing.

Impact of changing the decision rate was also evaluated and, by the results, it has just influenced in traffic overhead, keeping configuration delay the same. Last evaluations identified the modifications on application throughput according to the selected protocol and to the number of DA Nodes in the network, what has represented a substantial benefit.

Finally, Inter-Domain Policies negotiation has analyzed the modifications which occur in domains configuration when changing the characteristics of the negotiating elements interaction and in the distance between them. Interaction can be changed by negotiation mode or number of requirements and when they are negotiated. Independent of the specific characteristic that is changed, every time that a more complex situation is applied, the negotiation delay will increase. This increase is more influenced by domains distance, because of the higher transmission time, and number of requirements, because of the interaction which would be necessary to complete the negotiation process.

6 Conclusions

Networks self-configuration and adaptation have increasingly been attracting the attention of the Internet community for the last years. Currently, the scenario seems to be favorable for the deployment of self-configuration solutions: studies from Internet community, companies developing new solutions based on autonomous mechanism, research projects defining frameworks and protocols to improve networks' accessibility and users' experience. There is also a trend for evolving the Internet into a new converged network, where new requirements are expected to be supported and a heterogeneous and cooperative environment is supposed to be applied.

In this new converged network a sort of specific environments based on new manufacturers' technologies and users' requirements should be supported. However, the interoperation of these technologies should be defined in order to make them compatible with each other and allow heterogeneous environments to operate correctly.

6.1 Summary of Contributions

This thesis contributes with some step towards a scenario where the computer networks are able to configure themselves and negotiate the access and the policies necessary to regulate their communication in a completely automatic and dynamic way. The major contributions of this work can be grouped into four topics that are described in more details following.

6.1.1 Addressing Negotiation

Current mechanisms to deal with addressing in dynamic environments are not completely prepared to face the challenges in such scenarios. Stateless approaches, driven by mathematical equations, need to be supported by one or more mechanisms for duplicate address detection. But, even with the implementation of these support mechanisms,

protocols do not assure 100% of address conflict-free. One of the ways to avoid address conflicts is the implementation of centralized or distributed stateful approaches characteristics. However, most part of this kind of solutions appeals to a fixed structure, such as DHCP, in order to reach the protocol stability and addressing integrity.

To solve these problems this thesis has presented a complete approach for dealing with addressing issues in dynamic networks, considering different configuration “levels” necessary. This approach is divided in three specific mechanisms that are responsible by controlling addressing in different scopes and guarantee a correct addresses distribution.

First mechanism is based on Self-Addressing assignment to distribute addresses in networks that aren’t able obtaining valid addresses to their elements or no server is possible of running. In this case a set of operational phases was defined to make possible the network elements configure themselves even no addressing server being available in the network.

The second mechanism is the local addresses distribution in networks that are able of having a server to manage its addresses. To solve this problem an addressing protocol designed to perform the distribution and management of addresses in dynamic networks was designed and presented. This protocol implements the idea of address pools distribution among servers in the network which are responsible for the addressing issues, but it doesn’t specify how this distribution should be performed. After obtaining these addresses the servers must deal with a complete addressing procedure, implementing the distribution and posterior management of the addresses, efficient address conflicts avoidance, and the recovering of lost addresses or addressing spaces.

The last mechanism complements the functions necessary to solve the entire addressing problem: addresses pools distribution. This mechanism assigns available addresses to the servers present in the network using specific allocation disciplines to optimize addresses distribution. The main objective of an allocation mechanism is allocating adjacent subnets to the same request as much as possible, trying to avoid multiple discontinuous segments assigned to segments connected to the same network. This allocation is capable of improve routing performance, main when considering large networks, since it can considerable reduce routing tables’ size.

6.1.2 Intra-Domain Configuration Negotiation

Dynamic configuration of network elements is today limited and the main reason for this limitation is that networks and their elements have no ability to become “self-aware” about their configuration.

When networks become more complex and specialized communication environments are supposed to coexist and cooperate, it's harder to find solutions to deal with all these particular technologies.

Considering this scenario, lots of protocols can be used needing an element capable of the performing negotiations between network elements in order to verify the possible configurations and select the best one(s) for the present network environment characteristics. The verification of these possible configurations to decide the best one is necessary to allow networks to configure themselves automatically and adapt these configurations based on networks modifications.

Based on those shortcomings was proposed a method through which a network can advertise, discover, and negotiate configurations of different parameters, e.g. selection of a suitable routing protocol, interfaces to communicate and devices' configuration. But as the method provides a framework for dissemination of network operating environment information, together with how to negotiate and decide upon virtually any type of parameter, its scope reaches generally towards the ambitions of self-managed and self-configured networks. Using this mechanism we are able of controlling all networks configurations and adapt them according to environments changes.

6.1.3 Inter-Domain Policies Negotiation

Inter-Domain communication in Internet is currently controlled using a completely static model where the negotiation is done by companies in a higher "sphere" and those communication rules are defined. This negotiation model brings us static policies that are not able of adapting to requirements modifications.

This need of adaptation is extremely necessary in Internet because of its heterogeneity and dynamicity but despite the benefits and growing need for immediate solutions with support for policy negotiation no mechanisms are used in the Internet to solve this problem.

Trying to solve these problems and allow automatic Inter-Domain policies negotiation capable of configuring necessary policies to control the communication between networks which have never contacted before, a mechanism was presented. This mechanism has defined a set of phases that are necessary to allow these negotiations and how each one of them should proceed to make the negotiation possible. Note that this mechanism is both open and generic so that it could be used not only in the policies negotiation context.

6.1.4 Evaluation of proposed mechanisms

In the first experiments we evaluated the basic functionalities of the proposed local addressing protocol, and compared it with one of the most classical approaches for addressing in dynamic networks. The addressing mechanism has proved to be a promising approach since it performed well the addressing tasks, meeting most of the requirements imposed by the autoconfiguration paradigm in networks. The good performance of the addressing mechanism is due to its distributed behavior and its address conflict-free assurance, what gave to the protocol the capability of having a simple allocation procedure without overloads the network.

In a second moment the evaluation of the address pools allocation has presented the modifications that can be found in routing support information based on the organization of the addresses allocation to the networks. In this sense two disciplines to allow an automatic pools allocation over dynamic networks was proposed and presented very promising results when compared with a random distribution which doesn't consider any proximity to the addresses already allocated to a given network.

Intra-Domain configuration mechanism was also evaluated in order to present that it is capable of collecting configuration data from networks' elements and based on that create decisions and define a configuration which might be used to make possible or improve the communication over the network. Overhead and configuration delay results has presented good behaviors from the mechanism, indicating that the configuration of networks' elements using a self-configuration mechanism can give us a better performance and, most important, the possibility of creating an operational environment with no administrator' intervention.

Finishing the evaluations the initial experiments for the Inter-Domain Policies Negotiation mechanism were presented. The results collected from the performed simulations indicate the correct operation of the mechanism, showing that the policies negotiation is a possible operation to be executed in the networks and can bring benefits since it allow networks to create new agreements to adapt to new requirements/capabilities from the networks.

A general result that can be highlighted through an analysis of the results obtained by all mechanisms is that, as initially supposed, it is possible to implement mechanisms to perform automatic configuration and management of computer networks elements in various levels. Since initial work's conjectures were confirmed, further evaluations should be conducted, improving the understanding of proposed mechanisms in other situations as well as the details of the influence they have on networks' operation.

6.2 Future Work

In this section will be discuss some possible works that can be done to complement the contributions of this thesis. The activities were grouped by the mechanisms that they are involved to facilitate the understanding.

6.2.1 Addressing Negotiation

Addressing negotiation need to be evaluate in more details to verify the influences of each proposed element in the other and to create the necessary link among them, since they are supposed to work all together. The specific activities to each one are:

- Addresses Pools Distribution
 - Integrate with the Local addressing mechanism to improve addresses assignment to networks' servers;
 - Define a pools dissemination standardized protocol;
 - Evaluate routing performance with the possible routing tables configuration created by the allocation disciplines;
 - Evaluate scenarios with pools allocation and subsequent pools management.

- Local Addressing Negotiation
 - Implementation of backup mechanism;
 - Evaluate the addressing server hierarchy and server's backups;
 - Develop a methodology to allocate servers in order to maximize their benefits to the network;
 - Integration with Pools Distribution (request more pools, influence of different allocations in local distribution, etc.);
 - Evaluate the mechanism in environments mobility.
- Local Self-Addressing
 - Evaluate real implementation to guarantee the correctness of the mechanism;
 - Implement the Self-Addressing mechanism in a simulator;
 - Undertake performance evaluations of the mechanism.

6.2.2 Intra-Domain Configuration Negotiation

Intra-Domain Configuration mechanism is already implemented in a prototype for some specific environment characteristics, what allow us to verify the correct operation of the defined phases and elements and ensure that the proposal is really able of configuring and adapting networks according to their capabilities.

Simulator implementation has not considered all the necessary characteristics of the mechanism, since some restrictions were imposed in the implementation as Coordinator selection and possible characteristics considered to decisions performed by the mechanism. Next steps to be done to complete Intra-Domain configuration evaluation are:

- Finish simulator implementation to evaluate bigger scenarios;
- Implement a policies mechanism to regulate the possible operations to be performed in the intra-domains;

- Refine the decision algorithm to apply more complex decision in the network (network partition, decision delegation, etc.)
- Evaluate the Intra-Domain Configuration applied to more decisions;

6.2.3 Inter-Domain Policies Negotiation

Inter-Domain Policies Negotiation Protocol was correctly validated and an initial evaluation of the mechanism was conducted. However, more detailed analysis is necessary to evaluate the behavior and the performance of the proposed algorithms and methods to exchange information, negotiate communication rules and create policies. Some possible future activities which might be done in order to conclude these evaluations are:

- Definition of a negotiation description language;
- Definition of a ontologies learning mechanism;
- Definition of a pricing definition methodology;
- Definition of messages in XML to make them more extensible;
- Undertake more complete performance evaluations of the mechanism
 - Traffic overhead/pattern;
 - Environments modifications (monitoring phase);
 - Fully implemented policies framework;
 - Integration with routing (discovery phase);
- Implementation of the mechanism on a testbed;
- Extend the negotiation mechanism to other specific characteristics (user's roaming, networks naming/addressing plane integration, etc.).

References

- [4WA09] 4WARD Project. Available at: < <http://www.4ward-project.eu/>>. Accessed in: Nov. 2009.
- [AHL04] Ahlgren, B., Brunner, M, Eggert, L., Hancock, R. & Schmid, S., “Invariants – A New Design Methodology for Network Architectures”, ACM SIGCOMM Workshop on Future Directions in Network Architecture, August 2004.
- [AHL06] Bengt Ahlgren, Jari Arkko, Lars Eggert and Jarno Rajahalme. "A Node Identity Internetworking Architecture". 9th IEEE Global Internet Symposium. Barcelona, Spain, 2006.
- [AKH07] Nadeem Akhtar, Cornelia Kappler, Peter Schefczik, Laurensius Tionardi, Di Zhou. "Network Composition: A Framework for Dynamic Interworking between Networks". ChinaCom 2007. Shanghai, China.
- [ALA99] C. Alaettinoglu, T. Bates, E. Gerich, D. Karrenberg, D. Meyer, M. Terpstra, C. Villamizar, "Routing Policy Specification Language (RPSL)," RFC 2622, June1999.
- [ALA99-2] C. Alaettinoglu, D. Meyer, C. Orange, M. Prior, J. Schmitz, "Using RPSL in Practice," RFC 2650, August 1999.
- [AMB09] Ambient Networks Project. Available at <<http://www.ambient-networks.org>>. Accessed in: Nov. 2009.
- [AND06] Jorge Andrés-Colás, Carlos Pinho, Paulo Mendes, Yaning Wang, and José Ruela. Inter-network Quality of Service Agreements among Ambient Networks. To-QoS2006. Coimbra, Portugal, 2006.
- [AUT09] AUTOCONF, MANET Autoconfiguration. IETF Working Group (active). Available at: <<http://tools.ietf.org/wg/autoconf/>>. Accessed in: Nov. 2009.
- [AUT10] Autonomic Computing. Available at:

< <http://www.research.ibm.com/autonomic/>>. Accessed in: Oct. 2009.
- [BAA98] Baake, P. & Wichmann, T., On the economics of Internet peering, Netnomics, 1, pp. 89-105, January 1998.

- [BAC08] Baccelli, E. "Address Autoconfiguration for MANET: Terminology and Problem Statement," IETF Internet Draft, 2008.
- [BAC08] E. Baccelli, "Address Autoconfiguration for MANET: Terminology and Problem Statement," IETF Autoconf Working Group, Internet Draft, 2008.
- [BAC09] E. Baccelli, and M. Townsley, "IP Addressing Model in Ad Hoc Networks," IETF Autoconf Working Group, Internet-Draft, 2009.
- [BAM94] F. Bamberger, P. Ford, and J.M. Wing, "Interoperability," section in R&D for the NII: Technical Challenges, report edited by M.K. Vernon, E.D. Lazowska, and S.D. Personick, Interuniversity Communications Council, Inc. (EDUCOM), 1994.
- [BER09] C. J. Bernardos, M. Calderon, I. Soto, A. B. Solana and K. Weniger, "Building an IP-based Community Wireless Mesh Network: Assessment of PACMAN as an IP Address Autoconfiguration Protocol," in: Computer Networks, 2009.
- [BLA98] D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss, "An Architecture for Differentiated Services" RFC 2475, Internet Engineering Task Force, December 1998.
- [BLU05] L. Blunk, J. Damas, F. Parent, A. Robachevsky, Routing Policy Specification Language next generation (RPSLNg), Internet proposed standard RFC 4012, March 2005.
- [BLU09] Bluetooth Special Interest Group, SDP Specification. Web Site: <http://www.bluetooth.com/>, accessed on 17th February, 2009.
- [BOS05] S. V. den Bosch, G. Karagiannis, and A. McDonald, "NSLP for Quality-of-Service Signalling," IETF Internet draft, draft-ietf-nsis-qos-nslp-06.txt, Feb. 2005, work in progress.
- [BRA94] R. Braden, D. Clark and S. Shenker, "Integrated Services in the Internet Architecture: an Overview" RFC 1633, Internet Engineering Task Force, June 1994.
- [BRA97] R. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin, "Resource ReSerVation Protocol (RSVP)" RFC 2205, Internet Engineering Task Force, September 1997.
- [BUS02] Bush, R. & Meyer, D., "Some Internet Architectural Guidelines and Philosophy", RFC 3439, December 2002.

- [CAR96] Carpenter, B., “Architectural Principles of the Internet”, RFC 1958, June de 1996.
- [CHE02] J. C. Chen et al., “Dynamic Service Negotiation Protocol (DSNP) and Wireless Diffserv,” Proc. ICC, New York, NY, Apr. 2002, pp. 1033–38.
- [CHE05] S. Cheshire, B. Aboba, and E. Guttman, “Dynamic Configuration of IPv4 Link-Local Addresses,” IETF Network Working Group, RFC3927, 2005.
- [CHE07] Lawrence Cheng, Kerry Jean, Roel Ocampo, Alex Galis, Peter Kersch, Robert Szabo. "Bootstrapping Distributed Hash Tables in Dynamic Wireless Networks". ICC 2007. Glasgow, Scotland.
- [CIS09] Cisco, “Duplicate MAC addresses on Cisco 3600 series,” Available at: <<http://www.cisco.com/en/US/ts/fn/misc/7.html>>. Accessed in: Nov. 2009.
- [CLA03] Clausen, T. and Jaques, P. “Optimized Link State Routing Protocol (OLSR)”, RFC 3626, October 2003.
- [CLA06] Clark, R., Ammar, M., Calvert, K., Protocol discovery in multiprotocol networks, Mobile and Networks Applications Journal, June 2006.
- [CLA88] Clark. D., The Design Philosophy of the DARPA Internet Protocols, in Proceedings of the ACM SIGCOMM '88, Stanford, USA, pp. 106-114, August 1988.
- [CLA90] Clark, D. & Tennenhouse, D., “Architectural Considerations for a New Generation of Protocols”. ACM SIGCOMM, September 1990.
- [CLA99] Clark, E., SLAs: In Search of the Performance Payoff, Network Magazine, <http://www.networkmagazine.com/article/NMG20000509S0030>, May 1999.
- [D1.5] “D 1.5 – AN Framework Architecture”. Ambient Networks Project. www.ambient-networks.org/
- [D1.5-2] “D 1.5 – AN Framework Architecture (Annex)”. Ambient Networks Project. www.ambient-networks.org/
- [D3-G.1] “D3-G.1 Design of Composition Framework”. Ambient Networks Project. www.ambient-networks.org/
- [D3-G.1-2] “D3-G.1-Annex Design of Composition Framework (Annex)”. Ambient Networks Project. www.ambient-networks.org/

- [D5.1] "D 5.1 – SMART – Draft Architecture and Multimedia Routing Decision Logic". Ambient Networks Project. www.ambient-networks.org/
- [D5.2] "D 5.2 – SMART – Proof of Concept and Demo". Ambient Networks Project. www.ambient-networks.org/
- [DEW00] Dewan, R., Freimer, M., & Gundepudi, P., Interconnection Agreements between Competing Internet Service Providers, in Proceedings of the 33rd Hawaii International Conference on System Sciences, Honolulu, USA, January 2000.
- [DRO03] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," IETF Network Working Group, RFC3315, 2003.
- [DRO97] R. Droms, "Dynamic Host Configuration Protocol," IETF Network Working Group, RFC2131, 1997.
- [DUR00] D. Durham, Ed., "The COPS (Common Open Policy Service) Protocol," IETF RFC 2748, Jan. 2000.
- [EDE96] Brian C. Edem et al, "Method and Apparatus which allows Device with Multiple Protocol Capabilities to Configure to a Common Protocol Configuration", Patent number: 5586117. Filing date: Jun 20, 1994. Issue date: Dec 17, 1996.
- [FAN99] Fang, W., Perterson, L., Inter-AS Traffic Patterns and Their Implications, in Proceedings of the IEEE GLOBECOM'99, Rio de Janeiro, Brazil, December 1999.
- [FAZ06] M. Fazio, M. Villari and A. Puliafito, "AIPAC: Automatic IP Address Configuration in Mobile Ad Hoc Networks," in: Computer Communications, Volume 29, Number 8, Pages 1189-1200, 2006.
- [FOR05] T. K. Forde, L. E. Doyle and D. O'Mahony, "Self-Stabilizing Network-Layer Auto-Configuration for Mobile Ad Hoc Network Nodes," in: Proceedings of WiMob 2005, 2005, pp. 178-185.
- [FUL93] Fuller, V., Li, T., Yu, J., and K. Varadhan, "Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy", RFC 1519, September 1993.
- [GAO01] Gao, L., On Inferring Autonomous System Relationships in the Internet, IEEE/ACM Transactions on Networking, 9 (6), pp.733-745, December 2001.

- [GOD01] D Goderis et al., "Service Level Specification Semantics, Parameters and Negotiation Requirements." IETF Internet draft, draft-tequila-sls-01.txt, June 2001, work in progress.
- [GOD02] Goderis, D. et al., Service Level Specification Semantics and Parameters, Internet Draft, <draft-tequila-sls-02.txt>, work in progress, January 2002.
- [GOM06] Gomes, Reinaldo, Souto, Eduardo, Kelner, Judith, Sadok, Djamel. Evaluation of Energy Heuristics to On-Demand Routes Establishment in Wireless Sensor Networks. In: Third Annual Conference on Wireless On demand Network Systems and Services (IEEE WONS 2006).
- [GOM06-2] GOMES, Reinaldo, SOUTO, Eduardo, KELNER, Judith, SADOK, Djamel. Advantages of Node Energy Control for Routes Selection in Wireless Sensor Networks. In: Third Annual Conference on Wireless On demand Network Systems and Services (IEEE ICNSC 2006), Ft. Lauderdale. Proceedings ICNSC 2006, 2006.
- [GOM09] GOMES, Reinaldo, JOHNSON, Martin, SADOK, Djamel, SILVA, Tarciana, SILVA, Auristela, OLIVEIRA, Luciana, SANTOS, Felipe, SCHMIDT, Ricardo, "An Extensible Environment Discovery and Negotiation Method for Self-Configuring Networks," 4th IEEE Workshop on Advanced Experimental Activities on Wireless Networks & Systems (EXPONWIRELESS – WoWMoM), Kos, Greece, 2009
- [HAR99] D. Harrington, R. Presuhn, B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", Internet RFC2571, May 1999.
- [HAW96] Hawkinson, J. & Bates, T., Guidelines for creation, selection, and registration of an Autonomous System (AS), RFC 1930, March 1996.
- [HED88] Hedrick, C. "Routing Information Protocol". RFC 1058, Internet Engineering Task Force, June 1988.
- [HU04] Yih-Chun Hu, David B. Johnson and David A. Maltz. "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks". Internet-Draft, July 2004.
- [IAN02] IANA, "Special-Use IPv4 Addresses," IETF Network Working Group, RFC3330, 2002.

- [IAN09] IANA, "Internet Assigned Numbers Authority". Available at: <<http://www.iana.org>>. Accessed in: Nov. 2009.
- [IEE02] IEEE Computer Society, "802 – IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture," IEEE Std 802-2001, Published by The Institute of Electrical and Electronics Engineers, 2002.
- [IPN96] I-PNNI: Ross Callon, Jason Jeffords, Hal Sandick and Joel M. Halpern. Integrated PNNI. ATM Forum Technical Committee/96-0355, April 1996.
- [ISO83] "Information Processing Systems - Telecommunications and Information Exchange between Systems - Protocol for Exchange of Inter-domain Routing Information among Intermediate Systems to Support Forwarding of ISO 8473 PDUs", ISO/IEC IS10747, 1993
- [ISO84] ISO, "Information Processing Systems - Open Systems Interconnection - Basic Reference Model" ISO 7498, 1984.
- [JAI91] R. Jain, "The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling," Wiley-Interscience, New York, NY, ISBN: 0471503361, 1991.
- [JEO04] Jeong, J. "Ad Hoc IP Address Autoconfiguration for AODV," IETF Internet Draft, July 2004.
- [JEO06] J. Jeong, J. Park, H. Kim, H. Jeong, and D. Kim, "Ad Hoc IP Address Autoconfiguration," IETF Autoconf Working Group, Internet Draft, 2006.
- [JOH04] D. Johnson, C. Perkins, J. Arkko, "Mobility Support in IPv6", RFC 3775, Internet Engineering Task Force, June 2004.
- [KAG03] Kagal, L et. al. "A policy language for a pervasive computing environment," In IEEE 4th Int'l. WSh of Policies for Dist. Sys. & Nets, 2003, pp. 6-74.
- [KAM06] Kamienski, Carlos et. Al. "PBMAN: A Policy-based Management Framework for Ambient Networks". In IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'06).
- [KAN03] L. Kant, A. McAuley, R. Morera, A. S. Sethi, and M. Steiner, "Fault Localization and Self-Healing with Dynamic Domain Configuration," In proceedings of IEEE Military Communications Conference (MILCOM'03), vol. 2, pp. 977-981, 2003.
- [KUZ99] A.N. Kuznetsov, "IP command reference," Technical report, Institute for

Nuclear Research, Moscow, April 1999

- [LEI97] Leiner, B., et al., "The Past and Future History of the Internet", Communications of the ACM, 40(2), February de 1997.
- [LIM06] Jennifer Lima, "Performance Analysis of Network Composition in Ambient Networks", 2006, MSc Thesis available at Federal University of Pernambuco.
- [MAN05] Manousakis, K. "Network and Domain Autoconfiguration: A Unified Framework for Large Mobile Ad hoc Networks", Ph.D. Thesis. Department of Electrical and Computer Engineering, University of Maryland. 2005.
- [MANO05] K. Manousakis, J. S. Baras, A. McAuley, and R. Morera, "Network and Domain Autoconfiguration: A Unified Approach for Large Dynamic Networks," IEEE Comm. Magazine, vol. 43, issue 8, pp. 78-85, 2005.
- [MAS06] Mase, K; Adjih, C. "No Overhead Autoconfiguration OLSR," IETF Internet Draft, April 2006.
- [MCA01] A. McAuley, S. Das, S. Madhani, S. Baba, and Y. Shobatake, "Dynamic Registration and Configuration Protocol (DRCP)," IETF Network Working Group, Internet Draft, 2001.
- [MCG90] Paul E. McGlynn et al, "Feature Negotiation Protocol and Dynamically Adjustable Retrainig Sequence for a High Speed Half Duplex MODEM", Patent number: 4905282. Filing date: Oct 19, 1988. Issue date: Feb 27, 1990.
- [MCG90] Paul E. McGlynn et al, "Feature negotiation protocol for a synchronous modem", Patent number: 4953210. Filing date: Feb 22, 1990. Issue date: Aug 28, 1990.
- [MEY70] Meyer, E., "ARPA Network Protocol Notes", RFC 46, Abril de 1970.
- [MIC09] Microsoft, "How to troubleshoot duplicate MAC address conflicts," Available at: <<http://support.microsoft.com/support/kb/articles/Q164/9/03.asp>>. Accessed in: Nov. 2009.
- [MOC87] P. Mockapetris, "Domain Names – Implementation and Specification," IETF Network Working Group, RFC1035, 1987.
- [MOH02] Mohsin, M.; Prakash, R. "IP Address Assignment in a Mobile Ad Hoc

Network,” MILCOM 2002, 2002.

- [MOR03] R. Morera, A. McAuley, and L. Wong, “Robust Router Reconfiguration in Large Dynamic Networks,” In proceedings of IEEE Military Communications Conference (MILCOM’03), vol. 2, pp. 1343-1347, 2003.
- [MOS06] B. Moskowitz and P. Nikander, “Host Identity Protocol (HIP) Architecture” RFC 4423, Internet Engineering Task Force, May 2006.
- [MOY98] Moy, J. “OSPF Version 2”. RFC 2328, Internet Engineering Task Force, April 1998.
- [MUT08] Mutanga, M. B.; Nyandeni, T. C.; Mudali, P.; Xulu S. S.; Adigun, M. O. “Wise-DAD Auto-Configuration for Wireless Multi-hop Networks,” In the Proceedings of Southern Africa Telecommunications Networks and Applications Conference (SATNAC), Sept. 2008.
- [NAR07] T. Narten, E. Nordmark, W. Simpson, and H. Soliman, “Neighbor Discovery for IPv6,” IETF Network Working Group, RFC4861, 2007.
- [NES02] S. Nesargi, and R. Prakash, “MANETconf: Configuration of Hosts in a Mobile Ad Hoc Network,” In proceedings of 21st Annual Joint Conference of IEEE Computer and Communication Societies (INFOCOM’02), vol. 2, pp. 1059-1068, 2002.
- [NGU02] Nguyen, T. M. T., Boukhatef, N., Doudane, Y. G. & Pujolle, G., COPS-SLS: A Service Level Negotiation Protocol for the Internet, IEEE Communications Magazine, 40 (5), pp. 158-165, May 2002.
- [NIE04] Niebert N., Schieder A., Abramowicz, “Ambient Networks: An Architecture for Communication Networks Beyond 3G”, IEEE Wireless Communications, April 2004
- [NS2] NS-2 documents and notes. Available at: <<http://www.isi.edu/nsnam/ns/doc/>>. Accessed in: Sep. 2009.
- [NSI09] IETF NSIS Working Group, Next Steps in Signaling, <http://www.ietf.org/html.charters/nsis-charter.html>, last visited at 21/01/2009.
- [OBR01] K. Obraczka, K. Viswanath and G. Tsudik, Flooding for reliable multicast in multi-hop ad hoc networks, Wirel. Netw. 7 (2001) (6), pp. 627–634.
- [OHL06] Börje Ohlman, Kerry Jean, Alex Galis, Ian Herwono, Johan Nielsen. "Requirements for Policy Framework for Ambient Networks".Wireless

World Research Forum - WWRF16. Shanghai, China, 2006.

- [PAR97] V. Park and M. Corson. "A highly adaptive distributed routing algorithm for mobile wireless networks". In Proceedings of INFOCOM'97, pp. 1405-1413, April 1997.
- [PER01] C. E. Perkins, J. T. Malinen, R. Wakikawa, E. M. Belding-Royer, and Y. Sun, "IP Address Autoconfiguration for Ad Hoc Networks," IETF MANET Working Group, Internet Draft, 2001.
- [PER02] C. Perkins, "IP Mobility Support for IPv4", RFC 3344, Internet Engineering Task Force, August 2002.
- [PER02-2] Charles E. Perkins, "Mobile IP", IEEE Communications Magazine. Volume 40, Issue 5, Pages 66-82. May 2002.
- [PER03] C. Perkins, E. Belding-Royer and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing", RFC 3561, July 2003.
- [PER94] C. Perkins and P. Bhagwat. "Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers". In Proceedings of the SIGCOMM '94 Conference on Communications Architectures, Protocols and Applications, pp. 234-244, August 1994
- [PNN94] PNNI: Sullivan. E., and Gouguen, M. PNNI draft specification. ATM Forum/94- 0471, Apr. 1994.
- [PSA06] I. Psaras, L. Mamatas, Paulo Mendes. "INQA: InterNetwork QoS Agreements - A New Protocol for Dynamic SLS Control in Next Generation Networks". WNEPT 2006. Ontario, Canada.
- [REK95] Y. Rekhter, T.J. Watson and T. Li, "A Border Gateway Protocol 4 (BGP-4)" RFC 1771, Internet Engineering Task Force, March 1995.
- [REK96] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot and E. Lear, "Address Allocation for Privates Internets" RFC 1918, Internet Engineering Task Force, February 1996.
- [ROB07] J. Robinson, and E. W. Knightly, "A Performance Study of Deployment Factors in Wireless Mesh Networks," In Proceedings of 26th IEEE International Conference on Computer Communications (INFOCOM'07), pp. 2054-2062, 2007.
- [SAL84] Saltzer, J., Reed, D. & Clark, D., "End-to-End Arguments in System Design". ACM Transactions in Computer Systems 2(4), 277-288,

November 1984.

- [SAR06] V. Sarangan, J. Chen. "Comparative Study of Protocols for Dynamic Service Negotiation in the Next-Generation Internet," in: IEEE Communications Magazine, Volume 44, Issue 3, Pages 151-156, 2006.
- [SAY94] M. Sayrafiezadeh, "The Birthday Problem Revisited," Mathematics Magazine, vol. 67, pp. 220-223, 1994.
- [SCH09] R. de O. Schmidt, R. Gomes, M. Johnsson, D. Sadok and J. Kelner, "An Autonomous Addressing Mechanism as Support for Auto-Configuration in Dynamic Networks," in: Proceedings of IEEE/IFIP LANOMS 2009, 2009.
- [SCO01] D. Scott and T. Bittman. The Evolution Toward Policy-Based Computing Services. Technical report, 2001.
- [SIL07] SILVA, Auristela, SILVA, Tarciana, OLIVEIRA, Luciana, GOMES, Reinaldo, CANANÉA, Igor, SADOK, Djamel, JOHNSON, Martin. OSPF-AN: An Intra Domain Routing Protocol for Ambient Networks. In: 7th IEEE International Workshop on IP Operations and Management (IPOM 2007), 2007, San José, California, USA. Proceedings of 7th IEEE International Workshop on IP Operations and Management, 2007.
- [SIL08] Silva, A., Silva, T., Gomes, R., Oliveira, L., Cananéa, I., Sadok, D. and Johnsson, M., "Routing Solutions for future Dynamic Networks", 10th International Conference on Advanced Communication Technology (ICACT 2008). February, 2008.
- [SLO01] M. Sloman, J. Lobo. E. Lupu, Eds., Policy 2001: Policies for Distributed Systems and Networks, vol. 1995, Jan. 2001. <http://link.springer.de/link/service/series/0558/tocs/t1995.htm>
- [SUN03] Sun, Y.; Belding-Royer, M. E. "Dynamic Address Configuration in Mobile Ad Hoc Networks," Technical Report, University of California at Santa Barbara, Rep. 2003-11, Santa Barbara, CA, June 2003.
- [TAY04] Tayal, A.; Patnaik, L. "An address assignment for the automatic configuration of mobile ad hoc networks," Personal Ubiquitous Computing, 2004.
- [TEQ01] Tequila Consortium, "SrNP: Service Negotiation Protocol," <http://www.ist-tequila.org/deliverables>, Oct. 2001.
- [TEQ09] Tequila Consortium. Available at: < <http://www.ist-tequila.org/>>. Accessed in: Oct. 2009.

- [THO07] S. Thomson, T. Narten, and T. Jinmei, "IPv6 Stateless Address Autoconfiguration," IETF Network Working Group, RFC4862, 2007.
- [THO07] Thomson, S.; Narten, T.; Jinmei, T. "IPv6 Stateless Address Autoconfiguration," IETF RFC 4862, Sep. 2007.
- [TJO00] T'Joens, Y., et al., Service Level Specification and Usage Framework, Internet Draft, <draft-manyfolks-sls-framework-00.txt>, work in progress, October 2000.
- [TON] G. Tonti, J. Bradshaw, R. Jeffers, R. Montanari, N. Suri, and A. Uszok. Semantic Web Languages for Policy Representation and Reasoning: A Comparison of KAoS, Rei, and Ponder. Technical report, IHMC.
- [TRO03] O. Troan, and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6," IETF Network Working Group, RFC3633, 2003.
- [UPN09] <http://www.upnp.org>, accessed on 29th march, 2009.
- [USZ03] Uszok, A., Bradshaw, J., Jeffers, R., Suri, N., et al. (2003). KAoS Policy and Domain Services: Toward a Description- Logic Approach to Policy Representation, Deconfliction, and Enforcement. Policy 2003: Workshop on Policies for Distributed Systems and Networks. Springer-Verlag.
- [VAI02] N. Vaidya, "Weak Duplicate Address Detection in Mobile Ad Hoc Networks," In Proceedings of 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MOBIHOC'02), pp. 206-216, 2002.
- [VER02] Verma, D. C., "Simplifying Network Administration using Policy-Based Management", IEEE Network, March/April 2002.
- [WAN99] X. Wang and H. Schulzrinne, "RNAP: a Resource Negotiation and Pricing Protocol," Proc. Int'l. Wksp. Network and Op. Sys. Support for Digital Audio and Video (NOSSDAV), Basking Ridge, NJ, June 1999, pp. 77-93.
- [WEN03] K. Weniger, "Passive Duplicate Address Detection in Mobile Ad Hoc Networks," In proceedings of IEEE Wireless Communications and Networking (WCNC'03), vol. 3, pp. 1504-1509, 2003.
- [WEN03-2] K. Weniger, "Passive duplicate address detection in mobile ad hoc networks," in Proc. IEEE WCNC 2003, New Orleans, LA, Mar. 2003, pp. 1504-1509.

- [WEN04] K. Weniger and M. Zitterbart, "Address autoconfiguration in mobile ad hoc networks: current approaches and future directions," in: IEEE Network Magazine, Volume 18, Number 4, Pages 6-11, 2004.
- [WEN05] K. Weniger, "PACMAN: Passive Autoconfiguration for Mobile Ad Hoc Networks," IEEE Journal on Selected Areas in Communications, vol. 23, issue 3, pp. 507-519, 2005.
- [WES01] Westerinen, A. et al., Terminology for Policy-Based Management, RFC 3198, November 2001.
- [WIL02] Williams, A. "Requirements for Automatic Configuration of IP Hosts," IETF Zeroconf Working Group, Internet-Draft, 2002.
- [WIL03] Williams, A. "Requirements for Automatic Configuration of IP Hosts," IETF Internet Draft, 2003.
- [YAN03] G. Yang, M. Kifer, and C. Zhao. FLORA-2: A rule-based knowledge representation and inference infrastructure for the semantic web. In Proc. of the Second Int. Conf. on Ontologies, Databases and Applications of Semantics (ODBASE), 2003.
- [ZER09] ZEROCONF, Zero Configuration Networking. Available at: <<http://tools.ietf.org/wg/zeroconf/>>. Accessed in: Sep. 2009.
- [ZHO03] Zhou, H.; Ni, L. M.; Mutka, M. W. "Prophet address allocation for large scale manets," In Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003), Vol. 2, pp 1304-1311, 2003.