ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

TELECOM ParisTech

2014-ENST-0025

EDITE - ED 130

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

TELECOM ParisTech

Spécialité « Informatique et Réseaux »

présentée et soutenue publiquement par

Rui Pedro FERREIRA DA COSTA

le 27 Mai 2014

Architecture Mobile pour Réseaux de Nouvelle Génération

Directeur de thèse : **Christian BONNET**
Co-encadrement de la thèse : **Yacine EL MGHAZLI**

Jury

M. Jérôme HÄRRI, Professeur, Communications Mobiles, Institut EURECOM	Président du Jury
M. Rui Luis ANDRADE AGUIAR, Professeur, DETI, Universidade de Aveiro	Rapporteur
M. André-Luc BEYLOT, Professeur, IRIT/ENSHEEIHT, Université de Toulouse	Rapporteur
M. Thierry TURLETTI, Ingénieur, PLANETE/DIANA, INRIA	Examinateur
M. Christian BONNET, Professeur, Communications Mobiles, Institut EURECOM	Examinateur
M. Yacine EL MGHAZLI, Ingénieur, Bell Labs France, Alcatel-Lucent	Examinateur

TELECOM ParisTech
École de l'Institut Télécom - membre de ParisTech

THÈSE

THESIS DISSERTATION

Mobility Architecture for Next Generation Wireless Networks

Rui Pedro Ferreira da Costa

THESIS DISSERTATION

In Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
from TELECOM ParisTech

Specialization: Computer Science and Networks

Mobility Architecture for Next Generation Wireless Networks

Rui Pedro Ferreira da Costa

Public defence on the 27th May 2014 before a committee composed of:

President of the Jury		
Prof.	Jérôme Härri	Institut EURECOM
Reviewers		
Prof.	Rui Aguiar	Universidade de Aveiro
Prof.	André-Luc Beylot	Université de Toulouse
Examiners		
Eng.	Thierry Turletti	INRIA
Thesis Supervisor		
Prof.	Christian Bonnet	Institut EURECOM
Industrial Supervisor		
Eng.	Yacine El Mghazli	Alcatel-Lucent Bell Labs

“ Don’t gain the world and lose your soul,
wisdom is better than silver or gold... ”

— Bob Marley

Acknowledgements

Spanning over a little more than three years, this thesis was a great experience, allowing me to grow both personally and professionally.

I was given the opportunity to work abroad in a multicultural environment, and profit from time spent both in industry (Alcatel-Lucent Bell Labs - ALBLF) and academia (Institut EURECOM). Having the chance, I interacted, exchanged ideas and collaborated with people from different backgrounds and ages, with different areas of knowledge and views.

Greatly appreciating this chance I was given, first and foremost, I must thank Dr. Telemaco Melia and Prof. Christian Bonnet, the people who made this possible and supported me with their guidance in all aspects concerning this thesis. In the matter of supervision and guidance, I want also to express my gratitude to Eng. Yacine El Mghazli, that had a special role in them, throughout the second half of my thesis.

I would like to extend my thanks to people of the CR & MRS team, at ALBLF who made me feel welcome among them, and helped to transform the boring office life into something new and interesting every day. In equal terms, I want to thank the good folks of the Communications Mobiles group at EURECOM, who were great companions and always had a perspective on the bright side of life.

I also leave here a note of thanks to all people that in one way or another collaborated with me and influenced my work, especially the MEDIEVAL people.

To my all my friends either in Nice, Paris, Portugal or now anywhere else in the world, for their support in these three years, through good and bad times, a most sincere *Obrigado*.

And lastly but definitely not least, I thank my closest family, especially

my beloved parents, for their unflinching rock solid support, and their always handy advice.

Abstract

Permanent access to mobile internet has become a feature of our daily lives. Be that through smartphones, tablets, or dongles, mobile users benefit from vast amounts of multimedia content at their fingertips.

Meanwhile, new access technologies like LTE have been deployed and keep their promises in terms of bandwidth and latency. Not only do they open the doors to a great variety of new usages and possibilities, but they also face new problems that were reserved to the fixed world until now. To maintain the level of mobile user experience provided by the previous networks generation, new ways to deliver this huge amount of content must be found.

To meet the capacity challenge this has created for mobile operators, we believe an evolution is necessary in the architecture of mobile networks. Therefore, in this work, we present a mobility management architecture (based on Distributed Mobility Management and IP flow mobility), that makes use of cross-layer information to enhance the mobile user's Quality of Experience. The work has been validated under a real proof-of-concept test bed. In addition, this work evaluates the mobility scheme in a new performance evaluation platform that embodies a hybrid approach, combining emulation with simulation.

Contents

List of Figures

List of Acronyms

1G	1st Generation Wireless Telephone Technology
2G	2nd Generation Wireless Telephone Technology
3G	3rd Generation Wireless Telephone Technology
3GPP	3rd Generation Partnership Project
4G	4th Generation Wireless Telephone Technology
5G	5th Generation Wireless Telephone Technology
ALBLF	Alcatel-Lucent Bell Labs France
ALTO	Application-Layer Traffic Optimization
ANDSF	Access Network Discovery and Selection Function
API	Application programming interface
AR	Access Router
BA	Binding Acknowledgement
BC	Binding Cache
BCE	Binding Cache Entry
BU	Binding Update
CDMA	Code Division Multiple Access
CDN	Content Delivery Network
CN	Correspondent Node
CM	Connection Manager
DARPA	Defense Advanced Research Projects Agency

DHCP	Dynamic Host Configuration Protocol
DHCPv6	Dynamic Host Configuration Protocol for IP version 6
DMM	Distributed Mobility Management
DPI	Deep Packet Inspection
eNodeB	Evolved Node B
ETSI	European Telecommunications Standards Institute
EURECOM	Institut Eurecom
EU FP7	European Union Framework Programme 7
FA	Foreign Agent
FM	Flow Manager
FTP	File Transfer Protocol
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
GUI	Graphical User Interface
HA	Home Agent
HNP	Home Network Prefix
IETF	Internet Engineering Task Force
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
ISO	International Organization for Standardization
I-D	Internet-Draft
LIPA	Local Internet Protocol Access
LLI	Link Layer Information
LMA	Local Mobility Anchor
LMD	Local Mobility Domain
LTE	Long Term Evolution
LTE-A	Long Term Evolution Advanced

MAC Media Access Control

MAG Mobility Access Gateway

MAR Mobility Access Router

ME Mobility Engine

MEDIEVAL **M**ultim**ED**ia transport for mob**I**l**E** **V**ideo **A**pp**L**ications

MIH Media Independent Handover

MIP Mobile Internet Protocol

MIP4 Mobility for IPv4 Working Group

MIPv4 Mobile Internet Protocol version 4

MIPv6 Mobile Internet Protocol version 6

MN Mobile Node

MN-ID Mobile Node Identifier

NA Neighbour Advertisement

ND Neighbour Discovery

NIQ Node Information

NIQ Node Information Queries for IP version 6

NS Neighbour Solicitation

NETEXT **Net**work-Based Mobility **Ext**ensions Working Group

NMT *Nordisk MobilTelefoni* (Nordic Mobile Telephone)

NS-3 Network Simulator 3

OLT *Offentlig Landmobil Telefoni* (Norwegian Public Land Mobile Telephony)

OTT Over The Top

PBA Proxy Binding Acknowledgement

PBU Proxy Binding Update

PMIP Proxy Mobile Internet Protocol

PMIPv6 Proxy Mobile Internet Protocol version 6

PoA	Point of Access
PTSN	Public Switched Telephone Network
P-CoA	Proxy Care-of-Address
QoE	Quality of Experience
RA	Router Advertisement
RAM	Random Access Memory
RS	Router Solicitation
RAN	Radio Access Network
SIPTO	Selected Internet Protocol Traffic Offload
SWEEN	Simulation of Wireless Environment Extensions for Netkit
TCP	Transmission Control Protocol
UME	Unicast Mobility Engine
UML	User Mode Linux
UMTS	Universal Mobile Telecommunications System
VM	Virtual Machine
VNUML	Virtual Network User Mode Linux
VNX	Virtual Networks over Linux
VoIP	Voice over IP
W3C	World Wide Web Consortium
WiMAX	Worldwide Interoperability for Microwave Access
WG	Working Group
XML	Extensible Markup Language
X-GW	X Gateway

Chapter 1

Overview

1.1 Introduction

Recent decades have seen a constant evolution on mobile data networks. Steady improvements have increased the amount and type of services provided by the mobile operators. Consequently, mobile users began enjoying an always-on mobile data connection.

In parallel, during recent years, mobile devices have proliferated. They have become more powerful, more diverse in nature and functions, and constantly try to exploit any type of available data connection. Mobile users have never before had such variety of mobile devices and supported data services. One of the results is the ability to profit from an Internet access anywhere at any time.

In addition, the fast popularization of mobile applications has eased the mobile user's access to content, further fuelling a current steep increase in mobile data traffic. The largest portion of this new traffic belongs to multimedia content, reflecting for example, the impact of social networks and nowadays common actions like video streaming and photo uploading.

Even though mobile technology has been delivering higher bandwidth at every improvement, currently deployed networks have strained to meet the generated traffic demand.

On one hand, this consumption of media is done resorting to generally free Over The Top (OTT) services. Such services like Gmail, Youtube and Facebook have become more friendly to mobile users by adapting their content, taking into account mobile conditions. A large number of online services however, web browsing for instance, still treat users the same way regardless of their type of access, thus providing a poorer experience to users with limited access bandwidth.

On the other hand, mobile operators have for years been relying on tiered services and number of clients to dimension their infrastructure, managing a low cost per bit. These new demands for content have triggered the operators to deploy higher throughput access technologies and more core network capacity.

However, internet studies project that the momentum in data consumption will render such measures ineffective in a few years, effectively returning to the scenario where the network struggles to deliver high quality media content, while the mobile user suffers from a low Quality of Experience.

It is our belief that the architecture of current mobile networks needs to evolve and take these matters into account, therefore we propose a novel

mobility scheme based a flatter hierarchy and distributed management that stands on cross-layer information to improve the mobile user's Quality of Experience.

Furthermore, the mobility approach will be designed to interact with with a content delivery scheme, providing a compromise solution to address the issue of quality of mobile service provided to the user.

To prove its feasibility the resulting design will be validated in a real proof-of-concept test bed. Moreover, to extract notions on its behaviour in a meaningful manner, the design's evaluation will take place in a specific test bed that is built upon a hybrid evaluation method under a set up with similar conditions to the validated scenario.

1.2 Thesis Goals

Towards bridging the gap between, the mobile user's growing demand for quality mobile media and the infrastructure ability to deliver such content, this thesis endeavours to achieve several goals.

The main concern the definition, development, validation and and evaluation of a novel mobility architecture, that starting from current mobile network deployments and through the use of Internet Protocol (IP) mobility standards, aims at optimizing video delivery and video content distribution for mobile users including Content Delivery Network (CDN) aspects.

In the scope of the thesis, we present the following contributions:

- Definition of a novel architecture to accommodate mobility of users and IP flow mobility.
- Definition of a new functional element (Flow Manager) as a coordinator of mobility protocols.
- Definition and implementation of a performance evaluation strategy of the mobility architecture based on a hybrid method combining emulation and simulation.

Validation of the defined architecture is done under the scope of a European collaborative project involving academia and industry partners (such as Institut Eurecom (EURECOM) and Alcatel-Lucent Bell Labs France (ALBLF)), and whose larger goal is to address issues faced by mobile operators nowadays. Under the project the core of the defined mobility architecture will be integrated and its functionalities demonstrated in its several proof-of-concept test beds.

Evaluation of the architecture in a meaningful way, requires the construction of a specific test bed that will endeavour to deploy the defined architecture in a way similar to the validated work. This test bed will be produced in a novel way combining emulation, through the virtualization of network elements, and simulation, for the mobile and wireless environment.

1.3 Document Structure

Here follows a summarized description on the topics talked about in each chapter of this document:

Chapter 1 introduces the general subject, shortly describing the main reasons that drive this thesis. It includes a summarized view on its objectives as well as how they are to be achieved. Finalizing it is an overview on the document's structure.

Chapter 2 takes a quick peek at what the current trends have been, and how they stand today in the world of mobile networks. It lays the basic background and motivation for the thesis. It culminates in how the thesis fits along with the views of the state-of-the-art and shows how it intends to push and explore those limits.

Chapter 3 is a core part of the document, showing a levelled depiction on how the architecture was developed, detailing every component, its features and work flow. It explains how its blocks work together to form the core framework, as well as how it interacts with other technologies. It finishes by describing the circumstances under which the developed mobility architecture was validated.

Chapter 4 explores the development of the purpose-built test bed on which the architecture was tested. Is details how its hybrid approach is achieved and the advantages that it brings. Moreover, the author goes through the evaluation platform construction process, detailing tools, components and their interactions.

Chapter 5 shows the most relevant achieved results, the criteria that led to the metrics collected and overview on the tools chosen to gather them. It continues with a simple analysis on the gathered data and a conclusion on both the separate and entwined performances of the test bed and mobility architecture.

Chapter 6 starts by consolidating the final thoughts on the work developed and what it achieved. It also expands beyond the reach of this thesis, giving insight on how the work here developed can be used for other purposes, or build up on it to create interesting collaborations with other technologies. Wrapping up the chapter is a list of the author's efforts in publishing this work.

Chapter 2

Background and Motivation

The world of network communications is fairly huge and comprises more fields of expertise than one can imagine. From physical radio to the cloud and all in between, the possibilities for research are enormous. Even when we restrict that view to mere mobile networks, we can still be baffled on the amount of research that is currently under way.

To put the contributions of this thesis in perspective of the global picture, the essential background information is described along this chapter.

Starting with a global view on mobile networks, this chapter gives them a quick historical look and continues by providing some insight into the current trends in mobile networks and their background, as well as giving a summary on recent architectural advances.

Background description on mobile architectures will make way for recent research done in the area of this thesis, especially on how it forms the framework in which this thesis was developed. Exposing the framework boundaries will further define the work and describe which particular area of the field of mobile networks we intend to address.

The train of thought of this chapter will culminate in the motivation and goals of this thesis. It concludes with the fashion in which the contributions in this document intend to address the questions and issues highlighted in the background research given. The contributions and the answers they pose will drive and set the tone for the work in the following chapters.

Chapter Contents

Section 2.1 introduces the subject of mobile networks with a brief notion of the first efforts on mobile networks, progressing to a quick summary on the historical advances in this area.

Section 2.2 goes into the evolutionary path of mobile network architectures, the trends settled by standardization bodies, proposals taken as cornerstones, culminating in a simple analysis of what is the direction towards which the trends are going.

Section 2.3 describes a large picture on the recent events and trends, showing what are the current issues facing mobile networks. It describes as well how these issues are currently being addressed by different entities.

Section 2.4 explores the concept of Content Delivery Networks, looking into content delivery from a different perspective, that of applications. It gives a glimpse on how they are organized in terms of structure, and how they function in high-level.

Section 2.5 starts by disclosing the main motivation for the thesis, it continues by exposing the contributions that the author proposed in this thesis and finishes by giving a comprehensive list of goals that the thesis aims to achieve.

Section 2.6 concludes the chapter by giving a compact account on the contents of the chapter. It is meant as way to share the train of thought that the author exposed in the chapter by tying together the different sections, emphasizing the most relevant items, ideas, and conclusions.

2.1 Dawn of Mobile Networks

From the moment that the human species started to speak, that we strived to carry words, and their meaning, farther than that of that individual's voice can reach. Furthermore, we also searched for means that allowed to carry one's words for longer than our own individual's lifespan. This willing force was so strong that drove mankind to produce some of the most marking inventions in our history. Such historical milestones have shown our own advancement as a species throughout the ages: the appearance of spoken dialects and languages; the knowledge of written word; the mass diffusion by the printing press, the electricity enabled telegraph; the advent of wireless radio; the popularity of the telephone; the revolutionary computer; the daily life changing mobile phone; and nowadays, wireless data networks.

In this information age, technology evolves at a lightning-fast pace, with new devices and services created at a frenetic and continuous rhythm. Almost half a century has passed since the first type of wireless data network was developed, thanks in great part to the advances in electronics, computer technology and radio wave manipulation. These first experimental wireless data networks, such as the Defense Advanced Research Projects Agency (DARPA) network [1] and the ALOHA network [2], marked the beginning of what was to follow.

The next set of important events came to pass in the early 1980s, when inspired in existing analogue mobile telephone communications, such as the *Offentlig Landmobil Telefoni* (Norwegian Public Land Mobile Telephony) (OLT) [3], the *Nordisk MobilTelefoni* (Nordic Mobile Telephone) (NMT) was one of the 1st Generation Wireless Telephone Technology (1G) networks to include data exchange, such as text messages [3].

The digital data revolution however, was only to come in the later 1980s when in Europe, the task to address the mobile communications trend was trusted upon the European Telecommunications Standards Institute (ETSI). The outcome was a standard specification known as Global System for Mobile Communications (GSM) [4], a milestone in revolutionizing mobile communications. The commonly known GSM was popular and fuelled by a widespread adoption in European countries, it quickly became the *de facto* standard in 2nd Generation Wireless Telephone Technology (2G).

Marking the advent of mobile digital data networks, GSM was the beginning of a series of technologies and standards, that in their own way aim at deliver better and improved versions of mobile networks. Other standardization entities followed suit, developing other well known wireless digital technologies that today sound so familiar, such as, Universal Mobile Telecommunications System (UMTS) (in Europe), Code Division Multiple Access

(CDMA) (in the United States of America), Long Term Evolution (LTE), WiFi and Worldwide Interoperability for Microwave Access (WiMAX).

2.2 Architectural Evolution of Mobile Networks

Since the early days of GSM that mobile networks have changed to adapt to new demands, issues and constraints, evolving the structure of its architecture along the way. The process through which such improvements are approved can vary, but an unchanging factor is that it is done under the supervision and scrutiny of a standardizing body.

Standardization bodies are recognized entities that contribute to the development and harmonization of telecommunications so that a uniform behaviour can be expected out of any implementation or equipment supporting a certain standard. Interest generated in participating on these harmonization processes, has rallied institutions across the globe to become members of these standardization entities. Members originate not only from academia, like universities and research institutes, but also from industry, like equipment manufacturers, telecommunications operators and internet companies.

Although several standardizing entities have taken upon them the task of innovating and advancing the state of the art of mobile networks, there are a few that immediately stand out from the rest as major contributors of widely accepted technologies in this field: the 3GPP [5], the IETF [6], ; and the IEEE [7].

In a quick overview over the 3GPP and IETF standardization bodies, one can easily classify these entities based on the focus area of action of each one. Firstly, the 3rd Generation Partnership Project (3GPP) group aims at developing produce a full design, top to bottom with complete specifications. This comes from the fact that its constituent members are mostly mobile operators and telecommunications industry companies, meaning, they are usually used to deal with full network deployments, thus their interest in all-round network specifications. They create the concepts, design the full architecture, develop or adapt existing protocols for all layers and specify physical radio parameters, along with accounting, billing and security services. The Internet Engineering Task Force (IETF), has very different focus choosing as its primary area of intervention the higher layers (network, transport and application). Its membership is free and open, as are their informal type of discussions organized into Working Groups (WGs). With this type of openness and mostly focusing on the TCP/IP layer, this entity attracts a wide diversity of contributors, such as academia, telecommunications op-

erators, private enterprises, equipment manufacturers and even information technology companies.

On the one hand, the 3GPP focus more on stand-alone architectures, with a tight knit between all layers, and cross of information between its entities very well defined. On the other hand, IETF strives for more loosely defined standards, a consequence of the diversity of its contributors. The developed standards are more flexible to operate in either stand-alone or in coexistence with other technologies.

The fact that both IETF and 3GPP usually cooperate with other standardization bodies, such as, World Wide Web Consortium (W3C), International Organization for Standardization (ISO), etc., and leverage on each others standards, enables for the development of mixed standard solutions.

Towards focusing on the scope of the thesis, a light description will follow, detailing the approaches and modifications done over time to the most relevant standards, concerning mobile network architectures by both the 3GPP and the IETF.

3rd Generation Partnership Project

Picking up where section 2.1 left off, the appearance of GSM and what it meant for mobile communications, its time to note how the technologies that followed in its footsteps fared and changed towards what is available nowadays.

The GSM technology was based on the circuit-switched paradigm, originally from the Public Switched Telephone Network (PTSN), where hop-by-hop paths were firstly set up so that a telephone call could be established. It was also considered a first 2G technology due to the introduction of digital control over the radio interface (whereas 1G had analogue radio [3]).

In an effort to increase the debit of data and formalize the addition of data packet management entities, the General Packet Radio Service (GPRS) [8] standard was introduced. Its innovation was based on the success of GSM with an improved and dedicated structured data plane that incorporates the packet-switched paradigm onto the already existing circuit-switched network. The better data transmission came to the cost of a large and complex system (as seen on Fig.2.1).

The next generation of architectures, the 3rd Generation Wireless Telephone Technology (3G), saw drastic improvements in data bandwidth, through better use of radio spectrum, and simplification on the access net-

work elements. Although maintaining much of the previous core network to assure the coexistence with 2G, 3G modified the radio frequencies, needing new Radio Access Network (RAN) equipment, increasing the available data rate.

The most noteworthy and nowadays still widely deployed standards of 3G are UMTS in Europe and CDMA in the United States.

Observing Fig.2.1 a radical change happened from 3G to 4th Generation Wireless Telephone Technology (4G). Reflecting on the knowledge of high use of data from 3G users, and building upon the advantages of a more streamlined and packet-oriented architecture, the amount of control elements shrunk to a handful. Thinking mainly on Internet data access, it boasts a full packet switched core network, enabling services like Voice over IP (VoIP). A single entity now lies on the RAN, the Evolved Node B (eNodeB). Increased enhancements to the use of radio spectrum enable a higher data debit, and interoperation with other technologies is ensured (such as UMTS and CDMA.

LTE [9] is the current popular 4G technology being deployed, and in the coming years Long Term Evolution Advanced (LTE-A) is sought to be deployed as well.

To finalize, there is a specification of the 5th Generation Wireless Telephone Technology (5G) under way but is still largely a matter of discussion and research on a conceptual level.

The complex and large numbered elements of the generations presented in Fig.2.1, concern not only control and data plane, but also packet and circuit switched networks. In order to narrow down the focus towards mobile data networks, the figure was filtered so that the evolution of the architecture is clearly stated. The result of this filtering lies in Fig.2.2. There we can see how the path of data has changed over the 2G - 3G - 4G evolution.

It is quite noticeable in Fig.2.2 that a series of hierarchical levels have been reduced in the RAN (light blue boxes), its functionalities either being removed or merged with other elements, thus removing control, overhead and improving the overall performance of the system. Some specifications even allow, under certain conditions, to bypass elements in the chain [10] further reducing the number of hierarchy levels.

Furthermore, although top level entities seem rather unchanged, they have suffered large internal modifications to embody the shift from packet over circuit, to an IP oriented structure.

In conclusion, the 3GPP group has over the years standardized several iterations of its views on mobile and cellular networks. These extensive and highly detailed standardizations consider an enormous amount of aspects and cover all components, protocol layers and services required to run a mobile cellular network. Standards by 3GPP have acquired large recognition

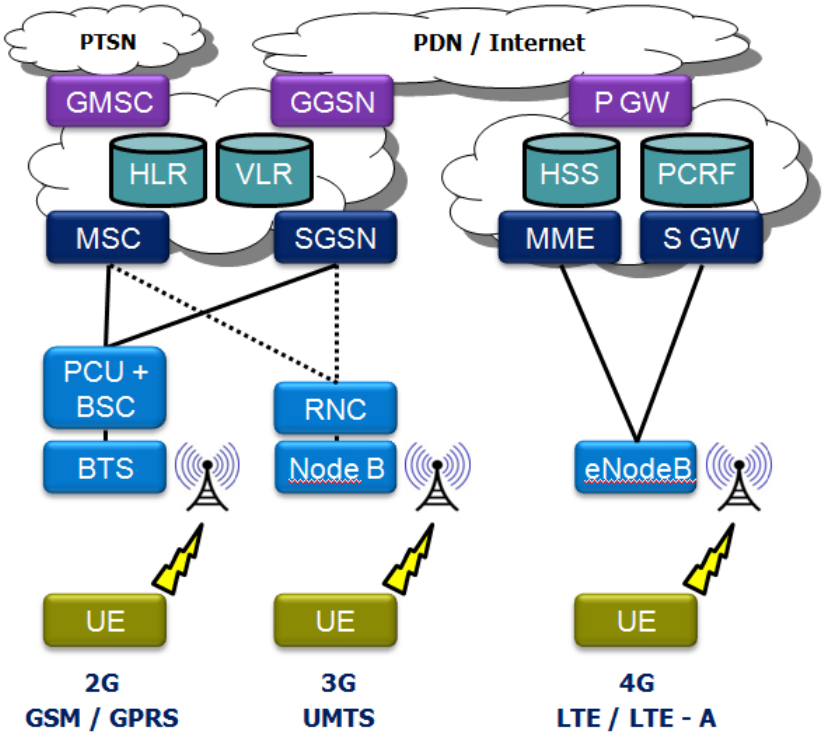

Figure 2.1: 3GPP Architecture Evolution

and have been deployed worldwide.

Internet Engineering Task Force

The Internet Engineering Task Force (IETF) standardization body has striven to deal with mobile networks challenges in a very different fashion than 3GPP. Using TCP/IP as its focus, has restricted and simultaneously given flexibility to its developed standards. Confinement to an area of intervention creates extra challenges towards finding a solution, but it also removes from the equation challenges or constraints that could come from intervening in other fields, thus its standards are more flexible solutions by abstracting from other fields of investigation.

From within the universe of standards produced in the IETF Working Groups, the generally considered historical milestones in the network mobility area, are Mobile Internet Protocol (MIP) and Proxy Mobile In-

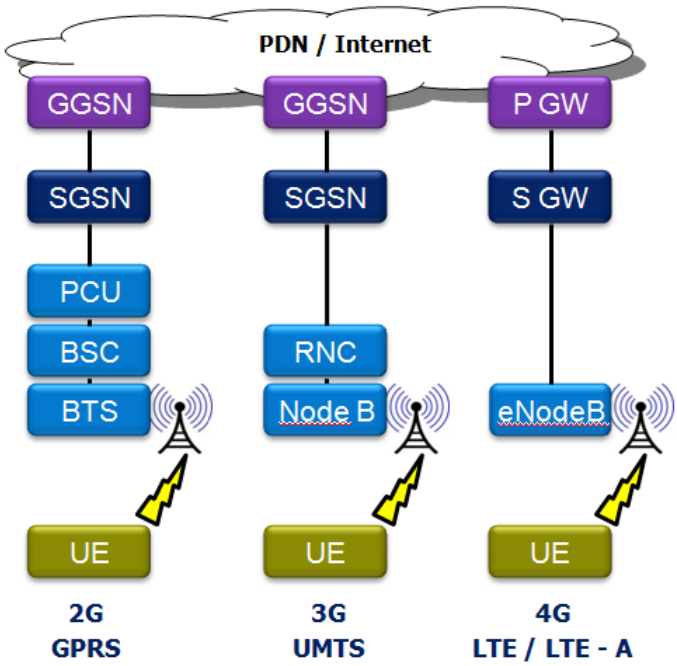

Figure 2.2: 3GPP Data Plane Evolution

ternet Protocol (PMIP). From the list of WGs, over the years, several addressed this field of network mobility, the ones still active that are worth mentioning are Mobility for IPv4 Working Group (MIP4), **Net**work-Based Mobility **Ext**ensions Working Group (NETEXT) and Distributed Mobility Management (DMM).

For the sake of brevity, and focusing on the subject of the thesis, the ones listed below will be lightly addressed to demonstrate the evolution trend of mobility architectures, within IETF. Further detailed descriptions will follow if necessary in other sections of the document.

The Mobile Internet Protocol (MIP) [11] is considered the first milestone in the attempts of standardizing an IP based solution to face the increased mobility of IP-enabled mobile terminals. The growing need for support of IP mobility for the user was made possible by the popularization of new and cheap wireless access technologies, such as WiFi [12].

MIP introduces a way of keeping the reachability of a mobile node's IP thus insuring that any undergoing traffic session is not lost during the movement to a new access. It also defines an architecture (depicted in Fig.2.3), where a hierarchy is detailed that contains three main elements:

the Home Agent (HA), the Foreign Agent (FA) and the Mobile Node (MN), which in Fig.2.3 is represented by the box `Client` (to disambiguate, since the use of the term Mobile Node is used extensively throughout this document in several contexts to refer to the terminal itself).

The three entities play parts in both the control and data plane of the protocol, forming a constant levelled structure that is present in every step of the user's mobility. The HA and FA are located at a very high level, not unlike network gateways, meanwhile the `Client` is located within the terminal, giving a very vertical perspective.

The protocol was also developed in two versions, Mobile Internet Protocol version 4 (MIPv4) and Mobile Internet Protocol version 6 (MIPv6), according to the IP version present on the network, a result of the long transition phase between Internet Protocol versions.

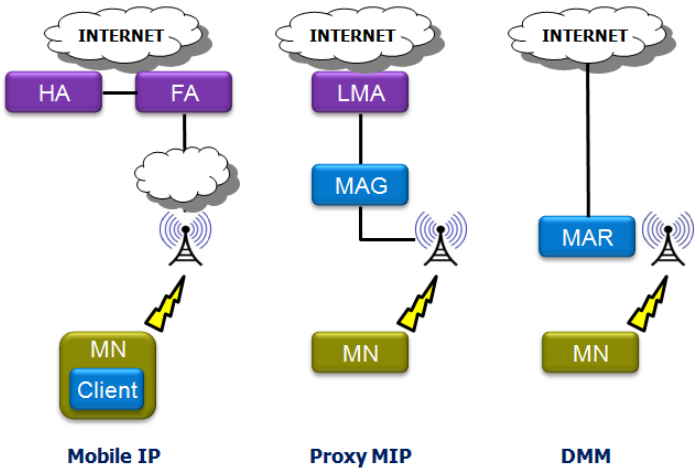

Figure 2.3: Evolution of IETF Mobility Protocols

The next interesting standard developed within IETF was PMIP, a derivation of MIP that aimed at mitigating some of its issues. One of the improvements was on tunnelling mechanisms, including the address management. The architecture became simpler by removing the functionalities from the MN and transferring the mobility protocol's entities to the infrastructure (see Fig.2.3). A consequence of this transference is the possibility to support a larger amount of mobiles, because it can include even the mobile nodes who don't support the mobility protocol, thanks to the due to the created seamlessness. It fact it signifies that the MN is not aware of its own IP mobility and thus is not required to support the protocol.

The Local Mobility Anchor (LMA) is an enhancement of the MIP component HA, while the Mobility Access Gateway (MAG) is a new entity that

acts locally on behalf of the MN. The benefits of PMIP comprise a smaller complexity, reduced tunnelling, and therefore reduced signalling and reduced data overhead, as well as optimized routing, while not requiring any intervention in the MN.

The most recent iteration on this track of mobility management by the IETF is a recently created Working Group, called Distributed Mobility Management (DMM) that is going through the phase of defining the guiding lines of a new approach. Looking at Fig.2.3, we can see one of the many views on this new mobility management approach, whose basic idea is of a flat and distributed architecture, as close to the edge of the RAN as possible. This architecture trend will be described in detail later in this document.

To summarize, IETF has been a driving force on standardization of mobility management. Standards developed by it have large recognition and application. For instance the MIP standards are nowadays integrated in Linux kernels [13], representing a very large deployment in computers all over the world. In addition, the light, flexible and open format of its standards allows for different ways of implementing them, while maintaining the aspect of easier interoperation and coexistence with other technologies developed by other standardization bodies.

In conclusion it's clear that across different standardization bodies we see a similar trend on mobile networks. A similar reflection on the evolution of different standards where it's possible to find a clear path towards smaller hierarchies, with less complicated mobility control mechanisms and improved performance on the data plane by means of a drastic reduction of the data management components which are traversed.

2.3 A new mobile world

With mobile data networks becoming commonplace and the constant evolutions in wireless technologies allowing for a steady decrease in the cost *per* bit, it was no wonder that it would cause an increase in the number of mobile data subscriptions, which in turn would cause expectations to rise on an increase in mobile data traffic [14]. Steep growth of consumed mobile bandwidth in the last few years, is due greatly to the advances made in not only in mobile devices but also in mobile applications.

A mobile user nowadays has an assortment of mobile devices from where to chose from that are compatible with his mobile data connection. From

laptops to tablets, phablets and smartphones. The average mobile user has never before had such a highly capable mobile device with such diversity at affordable prices. This financial impulse aggravates the increase in rate of bandwidth consumption in mobile environments.

In addition, mobile applications have become widespread and allow a user to access almost any service from its own mobile device [15], from online banking to social networking, media streaming and online browsing, the mobile device has even reached the edge of replacing, in some cases, the traditional home desktop computer.

Mobile networks have therefore suffered from these developments, especially from the growing demand for available bandwidth, which unlike in fixed access networks, in the wireless access medium is harder, and more costly to be augmented.

Furthermore, architectural issues arise when trying to change and improve the core network, which even in recent access technologies as LTE is still very centralized (Fig.2.1), and as such, suffer from lack of redundancy, single highly complex and expensive points and scalability difficulties.

Nevertheless, several enhancements have been put forward to tackle such issues, among them its possible to name newer improved radio technologies like LTE-A, localized traffic offload to other technologies, and small cell deployment.

Despite these well-know improvements at the current increasing rate of number of users and traffic consumption, they are proving to be only temporary stop-gap measures, until a redesign of the deployed architectures is performed and implemented.

The concern that newly deployed technologies might not have enough long-term capacity to address the user demands, has led to new research topics and raised new discussions in standardization entities, such as the IETF [6]. Within it, the discussions between contributors have resulted in new concepts that redesign the current mobile networks architecture. At the core of these new designs, there is a clear reflection of the ideas driving the evolution of mobile networks like, increased simplicity, less mobility management elements and a more flexible structure, as been mentioned in previous sections of this document.

From the pool of new ideas and approaches, DMM [16] has become an up-and-coming concept. Its popularity among the research community has granted it its own chartered Working Group in IETF. The DMM WG is (at this point in time), undergoing the definition of its principles, including what challenges it intends to address as well as the requirements a DMM solution should adopt. Despite the undergoing process, one already strongly stated

point is the intention of breaking from current centralized architectures by means of enabling the distribution of mobility functions throughout the network.

Not alone in this direction, from another perspective, in 3GPP discussions the resulting efforts towards addressing the same challenges have introduced the adoption of other techniques , such as Local Internet Protocol Access (LIPA) and Selected Internet Protocol Traffic Offload (SIPTO) [17].

From the other end of the spectrum, the Internet services industry is addressing the same issue from its own perspective. With the supported belief that video is largely the most consumed mobile data [14], OTTs have taken advantage to profit from potential revenue opportunities associated with this media. Indifferent to whatever type of technology or mobile network architecture the mobile users are connected to, OTTs are able to provide their services for free to their mobile users and still generate revenue.

In this area the remarkable dominance by OTT service providers, such as Google, Facebook, among others, has left the operators who provide the mobile users with Internet access, completely out of these opportunities on possible revenue streams associated with media.

In response, network operators began deploying their own content infrastructure locally in the core network (like mobile TV, music streaming, and online video collections), offering a good quality service to the mobile user, for which he was directly or indirectly charged. Although the operator's content had a better service, users still preferred to access the free and lower quality OTT rather than pay for the operator's content.

With such a growing client base, OTT service providers have been working hard to efficiently deliver its services to the user, despite whatever conditions of access the user has within its reach. In fact they have become experts at providing the most popular content to a mobile user by using a series of mechanisms.

One of the methods, consisted in improving the connection parameters at application level, mostly by using more efficient video codecs and better protocols for data transmission. Another very popular way of improving data delivery, is through minimizing and adapting the information exchanged, application protocols and media content towards the mobile environment. This technique is at the base of mobile-specific websites, mobile applications and mobile adapted video.

Furthermore, other techniques such as content replication and content distribution became ever more the subjects of research studies and industry development. OTTs started therefore using solutions based on these concepts to intelligently cache and distribute content throughout the Internet.

Nevertheless, content caching at high level did not prove enough to sat-

isfy the needs of the mobile user. Despite knowing the backbone, OTT content providers were still limited by the unknown that is the last-hop, particularly the ever-changing wireless last-hop, which is well known territory for mobile operators.

The existence of a disconnection is clear between the mobile operators and the OTTs, in trying to provide similar services to the mobile user. A disconnection that not only forces them to choose between services (a better but paid or a worse but free), but also impedes the mobile users of enjoying a service they like with a better experience.

With the objective of closing the distance gap with the user, OTTs and mobile network operators began compromising and arrived to new agreements where both could profit from the scenario. Agreements were established between them that specify that OTTs can have mobile operator backbone space to store content inside (see Fig.2.5), in exchange for including the network operators in the media revenue stream.

2.4 Content Delivery Networks

In the background that has been just described in the previous section, the concept of CDN thrives (Fig.2.5), with interests not only from established OTT but now also from mobile operators in search of deploying their own content.

A Content Delivery Network, also known by the derivative name of Content Distribution Network, is an infrastructure of several copies of data (caches), distributed in multiple points along a network. Their position, size and contained data is decided as to improve the delivery of cached data towards the user.

The main idea behind it consists in giving the user access to copy of the required content that is nearer to it, instead of fetching it from a centralized server, thus reducing the chance for bottlenecks at the original content server.

Content types include web objects, downloadable objects (media files, software, and documents), applications and real time media streams. From a high level viewpoint, CDN look very alike in terms of general structure (Fig.2.4).

There are generally four main groups of functions on a CDN:

- Publishing and Storage

  Relates to managing the cached content and through the content publishing processes, make it available to subscribers. Content can be

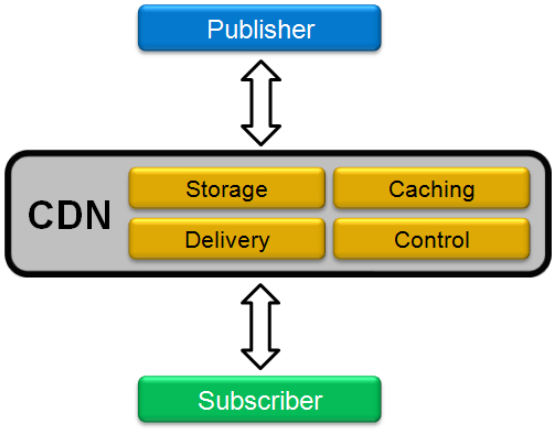

Figure 2.4: Generalized high level view of a CDN structure

cached either before or after its first request by a user. Even streams or live content should be possible to be cached. Once it is cached and authorized to be released, it should be published.

- Caching

  Concerns the replication of content within the CDN structure, comprising of the means to replicate and store content on caching storage devices, without having to re-acquire content from the source.

- Delivery

  The physical delivery of content to subscribers is the process where a consumer request is received, authenticated and routed to a relevant delivery device for fulfilment of the request. Usually it's a three stage mechanism: the subscriber request is received by a publisher, redirected to a cache and the cache allows the delivery of the local replica of the content.

- Management and Control

  These are the mechanisms that manage, monitor and control the CDN itself. They are responsible for maintaining efficiency and effectiveness, and also the configuration and provisioning of CDN services, management of physical devices and services and the day to day monitoring and support.

As a way to allow the distribution of content in a large scale at a fair cost, and with the added benefit of reducing the load on the core network of the mobile operator, the deployment of CDNs has become a focal point in the architectures of these operators.

Another advantage CDNs provide to mobile operators is the capability of enforcing content policy and content placement that can be optimized by being based on the user profile that the operator knows (e.g. location, accessed content, etc.).

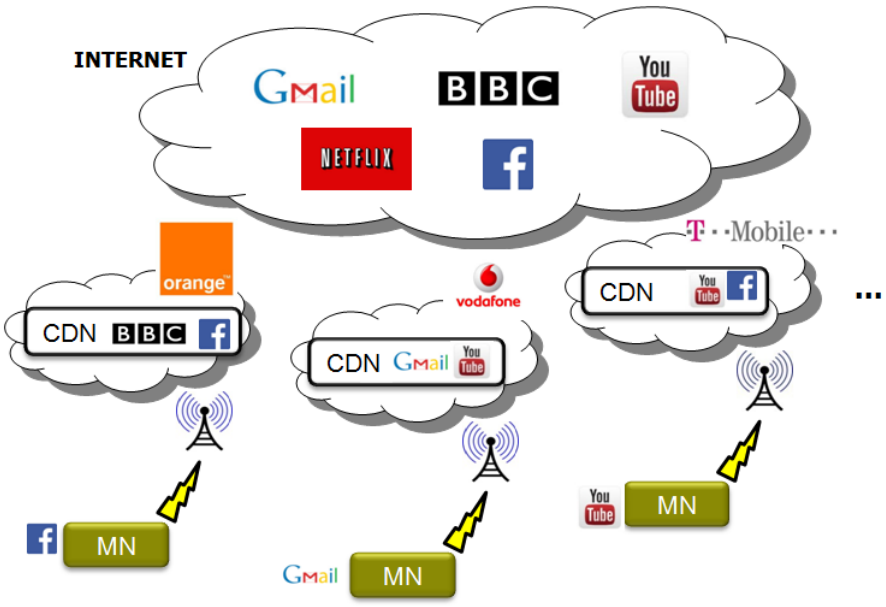

Figure 2.5: Example of a CDN deployment inside operators networks.

Furthermore, the use of CDN gives network operators the capability to place the content nearer to the user and make better use of the last hop resources (Fig.2.5). Caches located nearer to the edge of the network can help to mitigate the higher traffic demand caused by the mobile user's demand for more media content, by shifting the data exchange to the last hop.

Enabling to reduce the distance between the user and the content, this new content caching method has also brought the added benefits of ameliorating the mobile user experience, whose feedback is critical in increasing the client base of not only the mobile operator but also the OTT that has its content deployed there.

2.5 The Thesis

Taking a step back, for a moment remembering the conclusion of section 2.2, there is a common trend on the gathered state-of-the-art. Across different entities that promote and define new mobile technologies and standards, we see a road ahead looking to flatter and more decentralized management on mobile networks, the basic idea behind DMM.

This concept has been considered by mobile operators as the way to tackle the increased appearance of a new reality, a new mobile user profile, that has been fuelled by the availability of new devices and services, well characterized by its consumption of an ever more larger amount of data resources. Parallel to this affair, internet services took an entirely different approach. Excluded from the access network, they improved the mobile access to its content and services, by making an intelligent use of their placement and distribution on the network, the main drive of CDN.

2.5.1 Motivation

In the midst of this painted picture of current affairs, the mobile user is struggling to enjoy well the desired multimedia content on its mobile connection. In other words, the Quality of Experience (QoE) of a mobile user has suffered, but there is yet hope for improving it.

It is the author's opinion that the QoE of the mobile user can be improved, through the use of an efficient and evolved mobility control process that is at the centre of a new mobile network architecture. Embracing a flatter mobility hierarchy, this new architecture, would have the DMM philosophy at it's core and would welcome cross-layer interaction with other choice mechanisms towards enhancing the delivery of data (with video being in particular relevance), to the mobile user, thus increasing its QoE.

This is view is also shared by the EU FP7 project **M**ultim**ED**ia transport for mob**I**l**E** **V**ideo **A**pp**L**ications (MEDIEVAL) [18], which provides a base framework and extended motivation for this thesis.

2.5.2 MEDIEVAL

Aware of the several problems faced nowadays by mobile operators, the MEDIEVAL project motivation is within the same lines as this thesis. It addresses the mobile user QoE problems that can trace its origin to the massive traffic increase associated to the access of video services by mobile

users. The MEDIEVAL project motivation in detail can be further verified in the MEDIEVAL project's available documentation [19].

The project proposes a set of mechanisms for enhancing the efficiency of video transport and delivery, whose cross-layer functionalities can also be exploited for boosting the network performance, resulting in an increased QoE to the user. Such mechanisms include enhanced wireless access (with general abstractions for supporting heterogeneous technologies), improved mobility (for opportunistic handover across heterogeneous technologies), improved video distribution (with embedded caches in the network), and flexible video service control and provisioning (for exploiting the interactions with video applications).

In addition, the project is proposing and evaluating novel solutions that can be easily adopted and deployed into the existing mobile operators network infrastructures.

Poised along the same motivational lines, this thesis therefore finds itself at common ground with the MEDIEVAL project. It is then only natural that a strict collaboration with the MEDIEVAL project will result in contributions of the author to it, and in return the author will leverage on the MEDIEVAL project's contributions towards this thesis.

2.5.3 Contributions and Goals

Towards the goal of improving the QoE of the mobile user, the research performed within this thesis focuses on three stages: mobility architecture definition, validation and evaluation.

In the first stage there are a couple of mobility architecture designs presented as contributions. These are the result of the attempts at defining a mobility architecture based on observed trends in the background research.

In addition, the collaboration with the MEDIEVAL project stands out, where the author contributes extensively towards defining the mobility system, especially by proposing a mobility control entity, the Flow Manager (FM), that will form a central part of the global architecture, containing the intelligence of the mobility system. The thesis profits from this collaboration, where by taking advantage of the DMM core, the author will explore possible mobility designs, arriving at the definition of a novel mobility architecture design that will be a major contribution of this thesis.

Validating the work defined is a goal as well as a contribution, as the author briefly discusses the simple reasoning behind it, as well as its realiza-

tion (profiting again from the collaboration with the MEDIEVAL project), thus committing another important contribution.

Lastly in a third stage, the author poses the question on how to evaluate the defined architecture. Towards this, the author follows up with another large contribution, the defined architecture is taken as a use case for a large deployment in a purpose built test bed that uses a unique mixed evaluation method.

Based on the contributions that this thesis puts forth, the goals of this thesis can be described in a nutshell as such:

- Definition of a novel mobility architecture based on DMM.
- Definition of the mobility intelligence and control mechanisms based on IP flow mobility, mainly the FM, that makes use of cross-layer information.
- Definition of a validation and evaluation method capable of analysing a scaled scenario deployment of the defined architecture.
- Implementation of the evaluation strategy by means of a test bed where a scaled use case of the validated architecture is deployed and its performance is analysed.

2.6 Conclusion

Up to now we have established the background motivation and contributions of this thesis. We looked upon the historical point of view, where it was seen that there is a convergence towards a certain evolutionary trend, pushing towards more distributed and flat mobile network hierarchies. This is observed throughout different standardization entities, like 3GPP and IETF. The latter has created a WG called DMM that embodies this concept and has gained traction becoming a hot topic.

Moreover, an overview has been given on the recent changes on the mobile user's profile, explaining how due to the proliferation of cheaper and highly capable mobile devices, they have become more avid for content, which was reflected by a steep increase in data consumption.

We have shown what content providers and mobile operators have been doing to tackle this, namely the deployment of technologies such as CDN and

higher throughput capable radio such as LTE. The rhythm at which mobile data consumption is increasing however, can prove these to be temporary or stop-gap measures.

Towards bridging the gap between, the mobile user's growing demand for quality mobile media and the infrastructure ability to deliver such content, this thesis aims to give the following contributions: define a mobility architecture (based on DMM); validate such architecture (under the MEDIEVAL project); define an evaluation method for a scaled scenario of the defined architecture; and implement such evaluation method by means of a test bed through which a meaningful scaled use case of the architecture can be analysed.

Chapter 3

Mobility Architecture

In the last chapter the general motivation and background were described, showing the big picture in which the thesis is inserted. A quick look was given on mobile network trends, including a light state of the art in terms of architectural advances and research topics. Inserted in this large depiction were also the recent changes on the generic mobile user perspective and what new technologies have been used to answer such changes. Focusing more on the thesis at hand, the motivation was introduced, along with the summary on the main contributions and a short list on the goals that this thesis intends to achieve.

At the start of this chapter we delve into the first main contribution of this thesis: the design of the mobility architecture. On this subject, we begin by posing the question on which high level characteristics should this architecture possess, while also including some common practical points.

Afterwards we dive into the DMM concept, with the purpose of checking if the initial design is aligned with its philosophy, and taking the opportunity to classify it by which type it best reflects.

At this point, the author explores ways in which to combine elements that could work together to, in a high level architecture design, form a DMM core as base for the drafted architecture.

In the course of this exploration, the collaboration with MEDIEVAL project is considered and the author describes it's contributions to the project, together with an overview of the mobility subsystem components and operations.

We continue by demonstrating how the architecture should look like and which parts should be assimilated in what is to become the base DMM framework. With the DMM core selected, we go into the definition of the final architecture, its components, interfaces and detailed description on it's operations. In addition, the author explains how IP flow mobility can be achieved, thanks to the use of some mechanisms, amongst which the FM is paramount, thus concluding the first large contribution of this thesis.

With the architecture defined, the next step consists in performing a proper validation of the architecture, which sparkles the discussion on how the validation is to be achieved. The chapter finalizes as we lift the veil towards the evaluation of architecture, a work produced in the next chapter.

Chapter Contents

Section 3.1 describes the main pillars on which the mobility architecture is built upon. It also includes a description on the features that its high-level main blocks should perform, complemented with an initial draft on its expected structure.

Section 3.2 shares the concept of Distributed Mobility Management, the main pillar of the proposed architecture, listing its requirements and its different types. It contributes as well with an analysis on the initial drafted architecture from the DMM point of view and a classification of the proposed draft based on DMM types.

Section 3.3 explores different possibilities, that the author developed, towards forming a core framework for the main architecture. This experimentation process contains descriptions on architecture, protocols, mechanisms and operations performed on use case scenarios for each explored path.

Section 3.4 introduces the contributions made by the author to the MEDIEVAL project, followed by an overview of the project's architecture. It puts particular emphasis to the manner in which the contributions to the project came to influence the work presented in this thesis.

Section 3.5 defines the final set up of the constructed mobility architecture, detailing the components, their functionalities, how they operate together in concrete scenarios and what interfaces are available towards other technologies. It includes as well some notes on flow mobility and how it can be achieved.

Section 3.6 informs the reader, in broad terms on how validation of the contributions towards the core of the mobility architecture were carried out.

Section 3.7 concludes the chapter by giving a compact account on it's contents. It is meant as way to share the train of thought that the author exposed in the chapter by tying together the different sections, emphasizing the most relevant items, ideas, and conclusions.

3.1 Initial Design

The previous chapter has given us a general overview of, not only the general direction that mobile networks architectures are following, but also how different standardizing entities are looking into the next evolutionary step in this subject.

As concluded previously (section 2.2), current trends point towards an evolution towards architectures with a specific set of characteristics. The author here intends to extract the best principles from these trends, forming a path that leads to the logical next evolutionary step in mobile network architectures. Employing these characteristics on the base of the architecture soon to be defined, will not only to help it mitigate and avoid some of the structural restrictions present in current mobile architectures but also establish itself among the nowadays state-of-the-art.

Therefore the selected aspects will constitute the pillars on which the core of the architecture will be based. Here the most important ones will be enumerated, in order to establish this base as well as for further reference:

- **Flatter hierarchy:** Simpler with less levels and less entities, enabling a clearer and less cluttered communication, reducing management overhead.
- **Closer to the edge:** Putting mobility management functions as close to the access as possible, minimizing the distance between, not only the mobility decision making but also the enforcement of policy, and the mobile user.
- **Distributed functionality:** This would avoid centralized points of control and allows more flexibility and adaptation. It also induces the principle of better scalability.

Despite setting the tone for the envisioned architecture, the above described points are a mere starting point from which the desired structure will spawn. In fact, while investigating the trends on different standardization entities, it was noted that an approach such as the one taken by IETF would be easier to follow for several reasons.

Firstly, because it would be easier to focus on the mobility procedures themselves, at a higher level, and in a platform independent way, rather having to define a large and possibly complex set of mechanisms that would

involve several perspectives, from radio access management to application layer.

Secondly, because it would allow the freedom of a sort of plug-and-play style, that comes with the benefits of protocols developed under this style. It would enable the desired structure to interact and lean on the cooperation of other standards to achieve its goals. By doing so it would have also the added benefit of if needed, to reach a more extensive set of subjects (that are out of the scope of this thesis), thus making the architecture more broad, and more acceptable for the research community. As an example, some of these (of out of scope) subjects are, among others, security and authentication.

Finally in a more technical and practical perspective, an architecture following this approach would be by far easier to implement and test, which as mentioned before (section 1.2), is one of the goals of this thesis.

Upon this exercise of reflection on the general trends, and stipulating the basic pillars on which the architecture will stand, another exercise is required. This next step consists in gathering a list of desirable attributes or requirements for the envisioned architecture. These should be drawn from a practical point of view, since they should concern the specific objectives and environment of this thesis.

After drawing inspiration from the objectives of this thesis, and thinking on how it could be implemented in practice, the items below aim at better defining a set of practical starting points for the work developed.

1. **In line with current trends:** meaning in practice, to adopt as basis one of the already existing tendencies of mobility management in the research community.
2. **Build on existing work:** if possible, it should reuse, modify or extend an already existing model or protocol.
3. **Access technology agnostic:** it allows the architecture to be decoupled of technology specific details.
4. **Interface with other architectures:** namely CDNs, but also other protocols if needs be.
5. **Transparent to the MNs:** meaning that the MNs are unaware of mobility.

The above list, combined with the core principles listed before, provide a base from which to start forming a rough initial design of the envisioned architecture.

On a point-by-point approach, point (1) and (2) are not only related to all points given on the previous list that states the core pillars of the architecture, but go further in saying that in an implementation perspective, it would make sense to re-utilize an already existing approach and make it fit the intended purposes. In this way, by basing on a current mobility philosophy it would be possible to obtain a three-fold benefit: show that this work is aware of current research; give it a better acceptability within the research community; have a smaller and easier development and implementation process.

For the purpose of cross-platform use, as for instance, in heterogeneous networks, point (3) was added so that more underlying access technologies can be addressed.

Item (4) exists to conform with the thesis motivation and allow a cross-layer exchange of information, which is crucial to improve the user's mobile experience.

Lastly (5) informally proposes the removal of the MN from the mobility procedures, so that it can be able to roam seamlessly and unaware that mobility is being supported solely by the network. A by-product is that the MN is not required to contain any mobility entity or support any additional protocols.

Based upon these requirements, we can build up an initial draft of the architecture (see Fig.3.1).

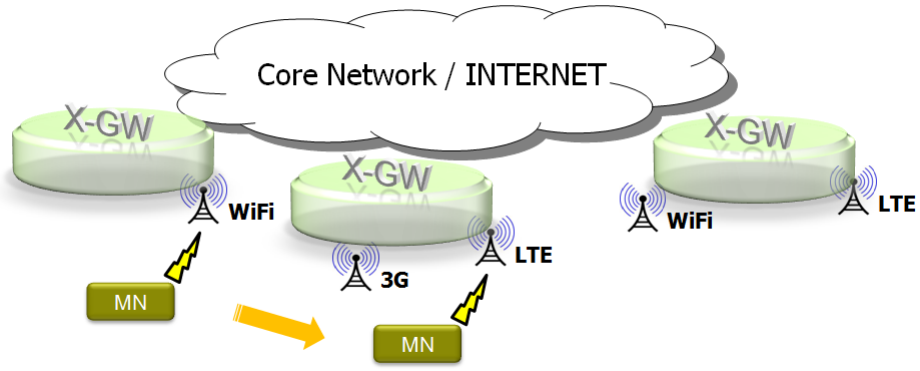

Figure 3.1: Initial Architecture Design

In Fig.3.1, we can see how these concept lines of the envisioned architecture take shape. The figure starts with a single entity, the X Gateway (X-GW), which is used as the sole point for connection access and control of all mobile users within its range. Applying the concept of distribution became easy, all that is needed is to increase the amount of X-GW, while enabling

them not only to be stand-alone and self-sufficient, but also, to work together in a cooperative manner.

One of the benefits that come from this is the need for a less complicated and possibly smaller backbone so as the X-GW need only be interconnected by whichever type of IP network is available, or as more direct approach, the Internet.

Each X-GW represents the basic building block of the architecture, being distributed, should also be independent and self-sufficient, containing all necessary mobile management functions to handle any users within its range and added functionality to support mobile users roaming between domains, meaning, between X-GW. This includes, of course the, ability to interface between X-GWs, enabling them to exchange information on roaming mobile nodes thus effectively working with each other to support user roaming.

With the mobility base abilities and functionalities defined, it was necessary to investigate how the mobile user experience could be enhanced beyond the classic roaming scenario. It became clear, that this goal could not be achieved without further information being provided to the mobility mechanisms.

Thus, towards increasing the mobile user experience, the X-GW must also include an interface that could receive external information that could influence the mobility of the user. Receiving such information and having mobility support act upon it, creates a trade-off: a degree of the user's freedom of to roam is exchanged for an informed mobility decision that can improve, or minimize the loss, of the quality of its current connection.

Once this structure's elements and functionalities contained within became clear, so did the path to follow. In fact, further research was needed in order to uncover what existing protocols or solutions existed already in the research community, that could fit either partially or completely in this architectural vision.

3.2 Distributed Mobility Management

It is no news, that the research community has been working to simplify complex and hierarchically dense networks for some years now, and standardizing entities, such as IETF and 3GPP, have been, at a slower or faster pace, working towards accepting research works along these lines [20].

Among these efforts, within IETF a WG was formed, that aims at investigating techniques focused on changing the current paradigm of mobility management. This Working Group's name is Distributed Mobility

Management (DMM).

Advocating a concept that breaks and simultaneously evolves from previous structural views of networks, their control, functionality disposition, information management, amongst other characteristics, this WG is fertile ground for new ideas and innovative solutions.

The DMM Working Group has recently published the major guidelines to develop DMM solutions (extensively described in [21]). They act as main goals of any solution proposed within the Working Group and are summarized below:

- Diverge from other network architectures, especially centralized architectures.
- Promote distributed processing, reducing non-optimal routing and single points of failure and attack.
- Improve on the scalability of session information and associated tunnelling management.
- Improve efficiency by removing unnecessary mobility support to clients that do not need it.
- Reduce mobility signalling overhead which concerns peer-to-peer communications.
- Enable coexistence with other deployed mobility solutions.
- Remove possible duplications of multicast traffic.
- Target first and foremost IPv6, and prioritize the reuse of existing IPv6 mobility protocols in DMM architecture implementations.

A quick look at the above items, is enough to recognize that several points are related to known limitations of centralized structures, as well as, tunnel-based architectures. Therefore, besides representing guidelines, they also represent issues to be addressed with currently deployed infrastructures, which in turn is a statement towards the motivation for DMM.

The dual purpose of the above given guidelines, along with the aim of coexistence with other solutions, and reuse of existing protocols, builds a notion that the nature of DMM is much more about evolution at its core, rather than disruption. It positions itself along the lines of presenting itself as the next step in architecture evolution, rather than a concept that stands apart to revolutionize the mobile communications world.

3.2.1 Types of DMM Architectures

Although the DMM Working Group has exposed conceptual guidelines defining which criteria is used to classify a solution or protocol as DMM compliant, it does not specify the architecture itself nor its implementation parameters. Showing the diversity of proposals at the DMM Working Group , several Internet-Drafts (I-Ds) have already been put forward, each with its own take on the principles on which the concept is established. Varying in their approach these solutions have originated a classification system based on a simple criteria: where the management logic of the solution is based from.

With different bases from where to act on and from, solutions can be generally classified [20] in three categories:

Routing-based approaches, such as [22], make use of the elements on the core network and it's routing properties, propagating the user's reachable address by means of frequently updating the routing tables of all elements in the path of mobile data at the core network. This type of approach does not imply tunnelling.

Client-based solutions, as [23], focus on the mobile node to manage tunnel sessions and addresses, adding to it a considerate degree of complexity, and implying the need to deeply modify the way a mobile node works.

Network-based solutions ([16] has a few examples), concentrate on the infra-structured network edge, particularly at the level of access routers. These edge elements constitute, in a general sense, session anchors, thus being used to manage tunnelling for session continuity, and manage addressing of its own attached mobile nodes.

Within the Network-based category, its possible to further classify a proposal considering it's degree of distribution, in which approaches can be roughly divided in either fully distributed or partially distributed. The distinction between the two classifications, is made at the level of coupling between control and data plane.

Following the aim of reusing existing protocols to create a DMM solution, is not always simple, as many existing protocols prove difficult to extend or modify. More often than not, existing control plane mechanism are unsuitable or undesirable to be transformed into a distributed functionality Consequently, sometimes a compromise needs to be considered, thus network based solutions can sometimes display a trade-off where the data plane management is distributed, along the network edge elements, and rely,

at a point or another, on a centralized mechanism for the management of the control plane [16] (Fig.3.2).

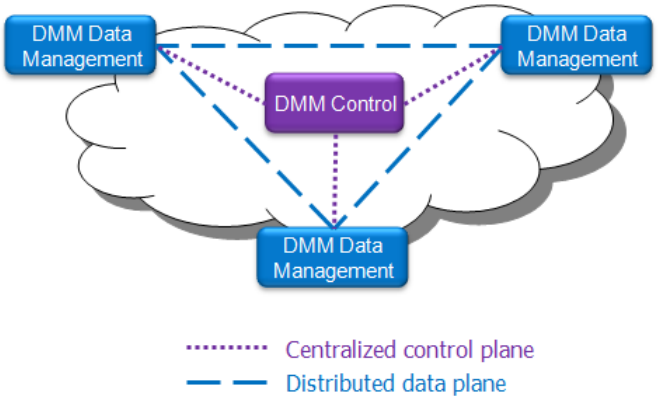

Figure 3.2: A partially distributed DMM design.

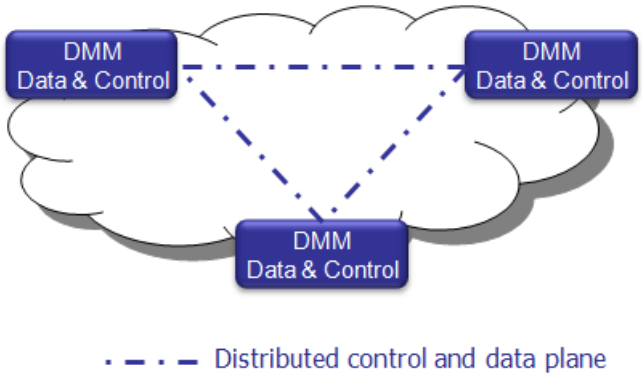

Figure 3.3: A fully distributed DMM design

When a proposal abides to such a trade-off, it has created at its core, broken the coupling between control and data plane, thus being called a partially distributed approach.

On the other hand when the coupling of data and control planes is strong, meaning that they are both distributed functionalities along the edge of the network, the proposal is considered to be fully distributed [16] (Fig.3.3).

Furthermore, the intersection of the initial design features with the classi-

fication of DMM approaches, defines our type of design as a fully distributed network-based approach. In view of this, throughout the document any item related to this approach will be emphasised, while any item related to other approaches will only be slightly addressed.

3.3 DMM solutions based on PMIPv6

Having conformed the initial draft architecture with the DMM philosophy in the previous section, the author moves forward in search of ways to combine protocols or technologies to obtain a base framework for the architecture.

Below are detailed the two first contributions that resulted in two different approaches: one fully distributed and one partially distributed, both attempting to provide a solution that could fit the requirements of the envisioned mobility architecture.

The first approach shows how a design can be achieved by combining a DMM core with the Dynamic Host Configuration Protocol for IP version 6 (DHCPv6) [24] protocol that results in a architecture similar to the Proxy Mobile Internet Protocol version 6 (PMIPv6) [25], following a DMM partially distributed approach. On a different note, the second approach combines DMM with the Node Information Queries for IP version 6 (NIQ) protocol [26], forming a structure that is fully distributed, thus not having centralized entities like the previous solution.

Both solutions are based on the initial draft architecture (Fig.3.1), comprising therefore the X-GW entities, their disposition in the network, and their base functionalities.

Furthermore, forming the DMM block on both solutions, we propose a variation of the PMIPv6 protocol, located on each of the X-GWs. The proposed variation consists of merging both the functionalities from the LMA as well as the ones performed by the MAG, permitting each and every X-GW to act in both the role of router for the data plane, and the role mobility anchor for the control plane, effectively distributing the functionalities throughout the network.

Moreover, for the data plane to effectively work in a distributed fashion with these added modifications, two conditions must be verified: that the X-GWs have global IPv6 addresses on the access link, one per assigned MN Home Network Prefix (HNP) and derived from the link-local address; and the X-GWs maintain the same link-local address in the access interface.

Having established how DMM could be deployed, we now face a problem,

introduced by the removal of the central LMA in our variation of PMIPv6. Like in regular operations of PMIPv6, in order to maintain session continuity, after the handover of a MN, it is required that the current X-GW performs a mobility state update to the former X-GW, which plays the role of the extinguished LMA. According to the specifics of our distributed X-GW deployment, we must now find another way to enable each X-GW not only to find each other, but also to provide them the correct information of which previous anchored mobility session concerns the moving MN.

It is concerning a solution to these two issues that we engage the help of the DHCPv6 and NIQ, each in its own fashion of approach, so that together with our distributed version of PMIPv6 they can form two full mobility architectures, one fully distributed and another partially distributed.

3.3.1 DMM and DHCPv6 (Partially Distributed)

Starting from the initial draft of the architecture, and following a partially distributed approach, where the control and data plane of the architecture are decoupled, the roles of data plane management are the sole responsibility of our variant of PMIPv6, while signalling is shared between it and DHCPv6.

Using a protocol for host configuration is commonplace on the vast majority of today's networks. In this architecture design we aim to profit from this fact by using DHCPv6 as a natural repository of centralized information. In addition, the fact that DHCPv6 has a Relay entity that could be deployed at each X-GW gives the chance to explore a possible interaction between the Relay and the modified PMIPv6.

This solution starts then by deploying DHCPv6 Relays on the X-GWs, then we move the responsibility of attributing the MNs addresses solely to DHCPv6, while making sure that PMIPv6 own addressing scheme is synchronized with it. In this way, any MN would be referred by the same address on both protocols.

Still related to the addressing scheme, two modifications to the DHCPv6 protocol are produced, as to solve the two issues that come from the modifications of the PMIPv6: i) that a DHCPv6 Request message is replied not only with the same HNPs already used in the previous lease, but also allocates the new HNPs corresponding to the new link; ii) that HNPs belonging to any previous X-GW have their lifetime set to zero, establishing them as deprecated and to be used only for ongoing data flows, while new data flows should use newly allocated HNPs.

When intercepted by the X-GW the reply messages will show to the PMIPv6 which previous X-GWs to update based on the HNPs assigned with lifetime zero.

Work Flow

A typical handover use case scenario is depicted in Fig.3.4, where a MN moves from one X-GW to another. The list below explains what happens in its particular sequence (the numbers corresponding to the steps listed in the figure):

1. The MN arrives at the new X-GW, starting a DHCPv6 configuration process by sending a Dynamic Host Configuration Protocol (DHCP) Request message. The Relay present at the X-GW will receive and continue the address configuration process. This message is listened by the modified PMIPv6 which will create a Binding Cache Entry (BCE) for the MN.

2. IP configuration continues its normal path, with the Relay forwarding the DHCP Request message to the Server.

3. Replying to the request, the DHCPv6 Server sends a DHCP Reply message to the Relay with the appropriate IP prefixes, both the ones concerning the new X-GW access with a valid lifetime and the ones from the old X-GW link with lifetime zero. This message is listened by the modified PMIPv6 which will update the Binding Cache Entry (BCE) for the MN with the appropriate synchronized HNPs, and a check is done for prefixes belonging to the previous X-GW. If there is no presence of prefixes from different X-GWs, we skip to step 6.

4. If prefixes from previous X-GWs are present, the modified PMIPv6 will start it's mobility update process of previous X-GWs by sending a Proxy Binding Update (PBU) *per* X-GW, thus updating the mobility status and data routing. To reach the previous X-GW the destination address used is the globally accessible one on the access link that can be built based on the known HNP and the common link address to all X-GWs.

5. Each of the contacted X-GWs will set up new routing to the new anchor accordingly and reply with a Proxy Binding Acknowledgement (PBA) message acknowledging the mobility update. To which, the current X-GWs modified PMIPv6 will profit to update its mobility routing rules and the corresponding BCE.

6. In the final step, the Relay will forward the DHCP Reply message to the MN, allowing it to finalize it's IP configuration.

In what concerns de-registration, standard DHCPv6 mechanisms apply, with the slight addition of the modified PMIPv6 sending a Router

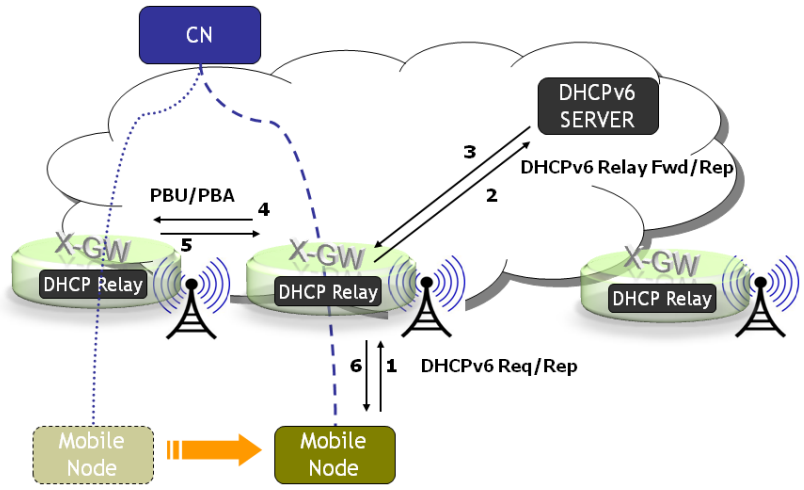

Figure 3.4: Partially Distributed Solution Architecture

Advertisement (RA) message to the MN, with the concerned HNP lifetime set to zero, as to complete PMIPv6 procedure.

For mobility unsupported MNs, meaning MNs for which the modified PMIPv6 is not authorized to provide mobility, they can still use the domain as long as the DHCPv6 standard protocol procedures authorizes the MN in question to configure itself with valid IP address(es).

3.3.2 DMM and NIQ (Fully Distributed)

Rather a bit more complex than the previous solution, thanks to the lack of centralized signalling. Both data and control planes in this approach are completely distributed. They focus only on the interactions between X-GWs themselves and between X-GWs and MNs.

For this specific solution we are still based on the draft architecture, and rely on the same modified version of PMIPv6, with the addition that we enlisted the help of the NIQ protocol.

Carrying the same issues mentioned in section 3.3, even more aggravated since in a fully distributed approach the X-GWs might not be aware of each other. Therefore any X-GW needs to know a couple of pieces of information

to properly support MN mobility. Firstly if the MN is coming from another X-GW (handover scenario) or not (initial registration scenario), secondly where does the MN have anchored traffic sessions (previous X-GWs), and lastly how to address the previous X-GWs.

Considering the last piece of needed information, there are some mechanisms that could help by providing a communication framework between X-GWs, such as, layer 2 protocols, peer-to-peer communication services, distributed discovery schemes, extensions to IP configuration mechanisms (like the Neighbour Discovery (ND) protocol), or likewise protocols (like the IEEE 802.21 MIH).

In this approach we take advantage of the defined assumption that the X-GWs have global IPv6 addresses on the access link, derived from the link-local address, which in turn is maintained as the same on all X-GWs. This enables any X-GW to know another X-GWs reachable address by building it based on a MNs HNP that was attributed by such other X-GW. As for the other two pieces of information, those can be known once that the HNPs and their number are discovered.

In fact, the distributed mobility support of this solution clings to knowing the HNPs currently assigned to the MN. Moreover, these could be easily extracted, if the configured IP addresses of the MN are disclosed.

Some mechanisms can offer the MNs configured IP addresses in the attached interface, such as the NIQ protocol. This protocol, allows two devices to communicate, at link layer level, and exchange information on a query-reply basis. It fits nicely here, because one of its described applications [26] is that is that it can be used to learn configured IP addresses and names on a point-to-point or medium-shared link, enabling us to make use of it for the MN to disclose this information when asked by the X-GW.

Work Flow

Just like in the same fashion of the partially distributed case, the work flow will of this fully distributed approach be described through a handover use case scenario. In Fig.3.5, we see the unravelling sequence of events (the numbers corresponding to the steps listed in the figure):

1. When the MN arrives at the new X-GW, it will initiate a common stateless configuration, namely, the Neighbour Discovery procedure, by issuing a Router Solicitation (RS) message.

2. Receiving the RS, the modified version of PMIPv6 will create a BCE for the MN. It will follow up by querying the MN for any configured IP addresses, through the issue of a Node Information Query message.

IP configuration continues its normal path, with the Relay forwarding the DHCP Request message to the Server.

3. The MN will then put the correct IP addresses in a Node Information Response message and send it to the X-GW.

4. Upon reception of the Node Information Response message the modified PMIPv6 will extract the HNPs, build the IP addresses for each of the HNPs. If no HNPs exist in the message the process skips to step 6, otherwise PBU messages are sent to each one of the proper X-GWs.

5. Any previous X-GWs receiving a PBU will process the mobility update by reconfiguring routing to the new anchor of this MN and issue a reply, the PBA message. Having received a PBA, the modified PMIPv6 will adapt its routing and mobility state, as well as the proper BCE.

6. In the end the modified PMIPv6 will release a RA message to the MN, indicating the current X-GW valid X-GWs HNPs, while the prefixes belonging to any other X-GW will have its validity set to lifetime zero. Once the MN receives this message he is able to reconfigure it's IP addresses.

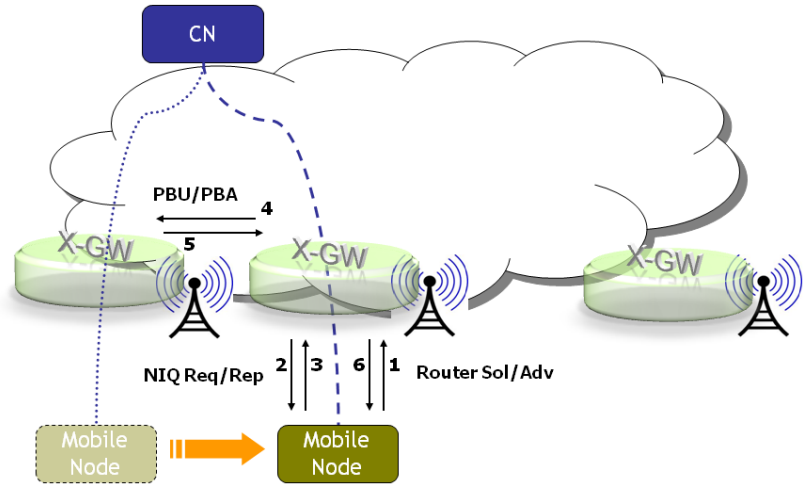

Figure 3.5: Fully Distributed Solution Architecture

Simpler than in the previous approach, the de-registration methods used here are the same as the standard PMIPv6. Moreover, although a differenti-

ation was made between mobility supported nodes and unsupported nodes, the use of this type of network is restricted to the nodes that not only have to support the NIQ protocol, as well as being included in the mobility support. Since in this approach the attribution of IP addresses is directly connected with the support of mobility.

On the two contributions made on this section, we see that unfortunately neither quite fits the needs and requirements of the desired mobility architecture design for this thesis.

The method based on DHCP requires a semi-distribution of signalling, which is not conforming with the expressed requirements of the design for the architecture, consequently it relies on a synchronization of attributed addresses present in two different protocols (PMIP and DHCP), which is undesirable. As a comfort it presents a solid signalling structure thanks to the protocols that make its base, and shows that MNs not belonging to the domain can still be supported by standard DHCP functions.

Based on the NIQ protocol the second method shows a true distributed nature and ease of function, but fails to keep off a direct intervention of the MN. In addition, the choice of NIQ could show itself to be a less than perfect option, since it's not a very popular standard, and it will require its existence in the MN.

In view of these facts, we will continue the search for viable options to form a base framework for the envisioned mobility architecture in the next sections.

3.4 The MEDIEVAL influence

Large contributions made by the author to the MEDIEVAL project, made this project a strong candidate from which to draw some inspiration and help to build a DMM core for the thesis mobility architecture.

Consequently, this section will also introduce, in a brief way, the work developed under the project. Special focus will be given to pieces that formed an influence on the thesis, aiming at making clear how the framework provided by the project has moulded the work developed in the thesis. Implied in the work described in this section, are the facts that the descriptions in this section can be consulted in further detail on MEDIEVAL project documentation ([19] [27] [28] [29] [30] [31] [32]).

The link between the project and the thesis is, in simple terms, that ALBLF was responsible for contributing to the architecture definition as well as providing some of the mobility components, namely the FM and

part of the Connection Manager (CM) [33]. In these circumstances, I was trusted with the tasks of aiding in the definition of the mobility architecture and producing the FM component.

A project as broad and as varied as MEDIEVAL implies a rather large and complex set of architectural blocks, consequently in this section this subject will only be approached lightly and filtered to a point of relevance to this thesis (see [27] for details).

To give a more compartmentalized view, the architecture is divided in several subsystems. These subsystems are: Video Services Control, Wireless Access, Mobility Management and Transport Optimisation. Among these, the most relevant concerning this thesis, is by far, the Mobility Management subsystem.

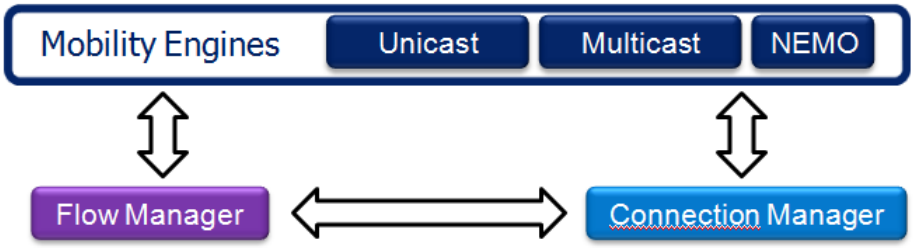

Figure 3.6: MEDIEVAL Mobility Subsystem [27]

The architecture of the mobility subsystem is shown in Fig.3.6 and it is composed of three main components: Connection Manager (CM), Flow Manager (FM) and Mobility Engine (ME). These are present in two main elements: the Mobility Access Router (MAR) which contains the FM and ME; the MN, where the CM resides.

Below follows a brief description on the relevant components [28]:

The Unicast Mobility Engine (UME) is the component that is responsible for IP mobility procedures in unicast and the core representation of DMM in the mobility subsystem. Residing in the MAR, it performs network based mobility management. It is triggered by the FM to support network-based mobility management whenever the MN needs to keep an IP address outside the subnet where the address is topologically valid.

The Connection Manager (CM) is responsible to manage any connectivity actions required in the terminal side. Residing in the MN, the CM implements access network policies, selecting the preferred access interface to use or splitting the traffic along the multiple access networks available, when the terminal is able to use them simultaneously.

The Flow Manager (FM) resides in the MAR, its the most important function is the management of data flows, since the mobility management is, in fact, applied on a per-flow basis. The FM is the main path of communication between the Mobility Subsystem and other subsystems such as Wireless Access, Transport Optimization and Video Service Control, playing therefore a key role in the whole architecture. The main focus of the FM within the mobility framework is to keep track of data flows that traverse it and, either according to events from network entities, or implemented policies, manage the data flows to provide the mobile user with the best possible service.

The mobility architecture (depicted in Fig.3.7) follows a DMM approach, positioning mobility anchors at the edge of the network. This approach is of a hybrid type, mixing both fully distributed network-based mobility along with client-based mobility, so that one can be used in the absence of the other.

The access network is organised in Local Mobility Domains (LMDs) where a network-based scheme is applied. Mobile users are expected to be most of the time roaming within LMDs, but when this is not possible, a client-based DMM approach is followed. In order to integrate both approaches, so a mobile node can simultaneously have sessions managed by a network-based (PMIPv6 alike [34]) approach and a client-based (MIPv6 alike) approach, the Mobility Access Router (MAR) is introduced.

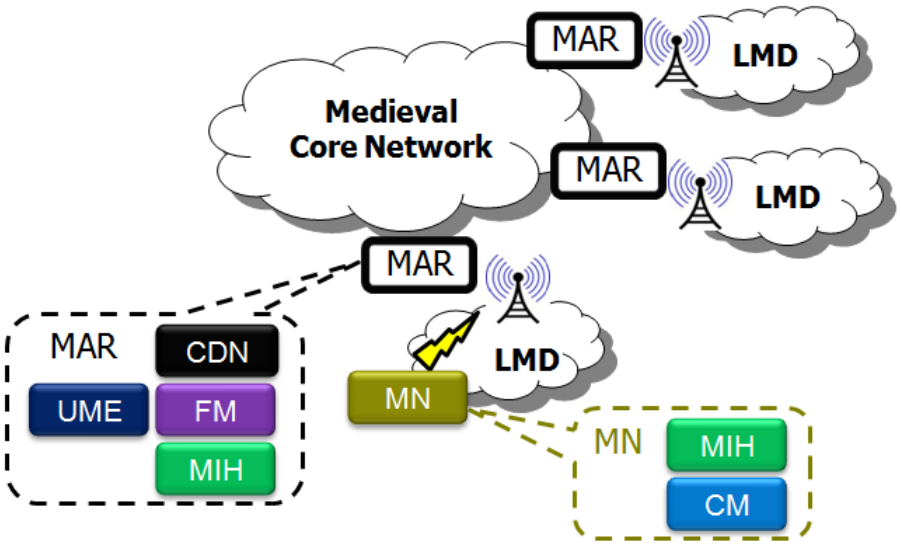

Figure 3.7: A partial view on the MEDIEVAL Architecture

The MAR is an entity at the network's edge that implements all the functionalities of the network elements in standard mobility protocols, namely MIPv6 and PMIPv6. Therefore, it encompasses several roles, among such as, Access Router (AR), Home Agent (HA), Local Mobility Anchor (LMA) and Mobility Access Gateway (MAG).

Mobility Operations starting with, roaming within a LMD is simple since the mobility session keeps anchored at the same MAR, however when a MN roams between LMDs in a fully distributed network-based DMM approach poses a problem. Unlike a centralized approach, where the MNs session information is concentrated on a single point, in a distributed environment, the complete information on the MNs session is most likely spread through multiple MARs.

Moreover, the fully distributed capacity increases the complexity of this issue, creating a three part problem that can be described in a single question, *"who came from where, carrying what?"*. The *"who"* refers to the roaming MN, meaning there is a need to identify the arriving MN. While the *"from where"* concerns the previous MARs visited by the MN that contain anchored IP flows concerning this MN (the *"what"* being carried).

Thinking in a mobility protocol perspective, this relates not only the trigger caused by a MN attachment (to a new LMD), but also to what is consequently triggered, meaning, the process of session context transfer and update between mobility anchors.

Thus whenever a MN has arrived in a new LMD there must be a mechanism that provides the mobility system with information regarding who he is (there lying the first problem). Furthermore, in order to properly trigger the mobility protocol, the identifiers of the any previous MARs are required to be known (there lying the second problem). Only then it is possible to be known where to fetch from and update this MNs flow anchoring, as to insure proper session continuity.

By means of using a solution which follows a Make-Before-Break approach for the handover operation and integrate Layer-2 and Layer-3 mobility procedures within the same framework to assist and drive the handover, it is possible to solve both problems. The Make-Before-Break approach will ensure that the session is transferred before the movement occurs, meaning that the destination will be known beforehand. Integrating the Institute of Electrical and Electronics Engineers (IEEE) 802.21 for Media Independent Handover (MIH) [35] [36] as a means to exchange cross-layer information, makes possible to learn not only when a MN has attached, but also which MN has attached.

The mobility subsystem has interfaces with other components besides MIH. Another important interface is with the CDN set of components.

This interface enriches the mobility decision process in a very special way. The CDN is asked to intervene in mobility decisions by sharing its view on the best destination. This view is based on its own content caching distribution information, especially its knowledge on the location of content and the content being currently accessed by the MN.

By involving the CDN in the process, a more informed decision can be produced taking into account the content profile of the MN, pointing it towards a destination where the desired content can be accessed while enjoying the best possible QoE.

The Content Delivery Network (CDN) is a component that aims at the optimal placement and management. of CDN nodes and optimal selection of content locations based on the specific layout of the operators core network and the policies defined by the network operator. This also includes maintaining an efficient and stable overlay topology for the control and management of the CDN nodes, performing load-balancing among the cached video sources and network elements, as well as relaying connections for mobility, caching, or confidentiality reasons. To function, such decisions require a continuous monitoring of the current conditions of the entire CDN system, in particular the status and content distribution of the CDN nodes and the popularity of the video content. Using the collected data an optimal configuration of a set of servers for content distribution is dynamically maintained and an optimal candidate from the set of available sources is selected for transmitting the videos to the users.

3.4.1 Leveraging on MEDIEVAL

As seen above, MEDIEVAL has a large and fairly complex architecture from which only the most relevant components in regard to the mobility subsystem and its interfaces have been described. This has been done for three reasons: firstly for the reader to have a light understanding of the architecture and component functionality; secondly so that is clearer how some of the components relate to this thesis; and lastly to point out which were the author's contribution to the project and therefore to thesis.

In order to leverage on the work performed under MEDIEVAL, in this section it will be evaluated, which components can or cannot be used to form a DMM core for the thesis mobility architecture.

The first and foremost aspect to mention is the convergence, or better yet, the shared inspiration for both the thesis architecture and the project's mobility architecture. As their development was made simultaneously, under the same general philosophy of DMM, its natural that they include

similarities. Despite this overlap, both are still quite different as the some of the used methods (namely the use of the CM in MEDIEVAL) do not fit in with the requirements of the thesis (namely the thesis stated avoidance to deploy mobility entities on the MN).

Since there is a similarity between both designs, which resided at the DMM level, a quick evaluation was performed to understand which MEDIEVAL components could be reused to fill in the required features and how.

During analysis, some components were rejected because their functionality was either not needed or not desirable to be included. For instance, the IEEE 802.21 MIH was not integrated because the functionality was felt as unnecessary and cumbersome, as the information it provides could be procured in another way. The CM was dropped completely as in our vision of the architecture, the MN is not involved in mobility procedures.

On another take, the CDN component was decided to be partly included. In more detail, the functionalities and presence of the caching nodes in the MAR will be preserved for the sake of their interface with the FM which crucially works towards the objective of enhancing the MNs QoE.

The DMM core that results from the close coupling between FM and the UME was imported, because they are able to produce together the necessary mechanisms to provide DMM support. For these reasons, the UME was imported into the thesis work as enforcer of mobility, *as is*, unmodified and unadjusted.

While the FM on the other hand as the intelligence behind the mobility architecture and the hub through which all mobility information flows thanks to its several interfaces, needs adaptation. To be discarded are its interfaces and implied functions that involve the CM. Although the MIH is not integrated in the architecture, the interface that the FM has with it was kept but heavily modified, resulting in the renamed Link Layer Information (LLI) interface. The interface with other FMs and the interface with the CDN component will be adapted, while the UME interface is to be kept unchanged.

Note: for practical reasons, in the context of the contribution made by this thesis, the UME when addressed as MEDIEVAL component will be mentioned as UME while when addressed in the context of the thesis mobility architecture, will be renamed and mentioned as DMM engine.

3.5 Final Architecture

Settled the matter of the DMM core in the previous section by constructing it with the FM component coupled to the DMM Engine, now we push with the development of the architecture, where, up to now only have been introduced the basic principles on which it stands, the adopted guidelines and additional desirable features. In the previous section, it was described how the MEDIEVAL project framework is reflected in this design, therefore at this point, what is expected is a wrap-up of the final aspects of the architecture, a detailed explanation on how it all ties together, and how the mobile operations unfold in typical scenarios.

Starting with an overview of the final mobility architecture (depicted in Fig.3.8), we can see the familiar distribution of X-GWs along the edge of the infra-structured network. The distributed X-GWs are now updated with the DMM engine component and the FM.

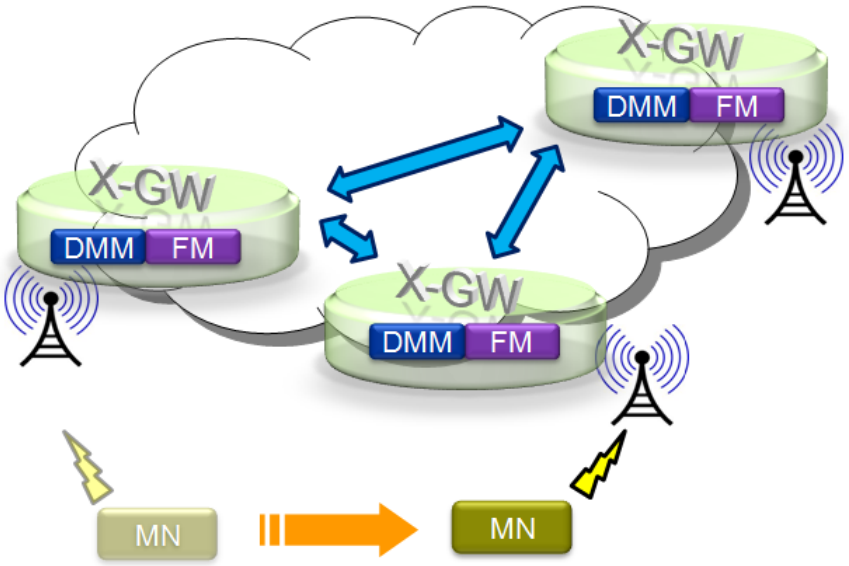

Figure 3.8: Final Mobility Architecture

In plain view is the tight connection between the DMM engine and the FM, the former enforcing all mobility decisions taken by the latter. The arrows in Fig.3.8 represent the distributed control and data plane connections, showing no necessary involvement from the MN as well as no constraints

concerning the underlying access technology.

Cooperation between the components to achieve IP mobility is very straightforward. The FM monitors IP flows that traverse the X-GW, when a MNs mobility is detected the FM determines which flows require IP mobility support. This requires the local FM, to communication with other FMs located in other X-GWs for a notion of the current ongoing flows belonging to the same MN. Once known which require mobility support, the DMM engine will be provided with the X-GWs affected and will use its mobile protocol functionalities to update the anchoring of the flows to itself.

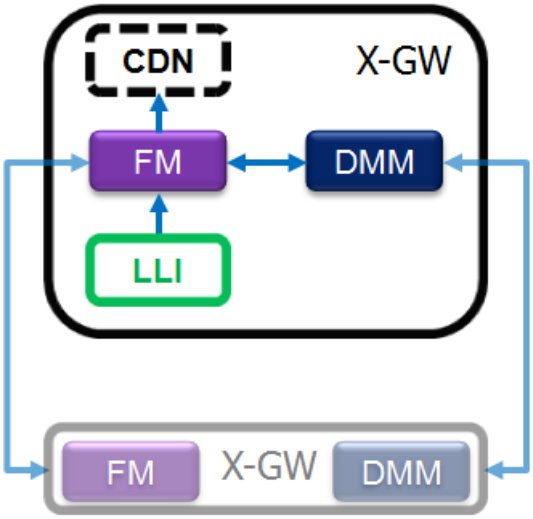

Figure 3.9: X-GW internal components and interfaces.

Inside the X-GW we can better see how these interactions flow, by checking the interfaces between the modules that compose it, clearly shown in Figure 3.9. The close connection between FM and DMM engine is shown by a direct communication interface, while both communicate with their respective peers on other X-GWs through their own dedicated interfaces.

On the upper and lower side of the FM there are also two other depicted interfaces, the input interface with the LLI, and the CDN notification interface. The LLI interface is a necessity for the FM to know certain information regarding the MN, while the CDN notification exists with the purpose of enhancing the benefits of mobility knowledge upon its distribution and delivery of content to the MNs.

3.5.1 Distributed Mobility Management engine

In this section the DMM engine is addressed in a summarized manner based on detailed information that can be found across [29], [30], [31], and [32]. Imported from MEDIEVAL *as is*, the UME (from here on renamed as DMM engine for the purposes of this architecture), represents the means chosen to perform IP mobility operations. Its design starts from the PMIPv6 protocol [25], which was modified so that its operations conform with a fully distributed network based DMM approach.

Placing it at the X-GW, is the most obvious choice, and by being located at the first IP entity on the infrastructure side, it's in the best place to perform its two objectives: setting up the IP connectivity of the MN upon arrival, and assuring the same connectivity when MNs are roaming between different X-GWs. Implicit in its latter objective, is keeping up with the updates in location of any MNs with data flows still anchored at it. This procedure is achieved thanks to collaboration with its peers in other X-GWs, through a dedicated interface (visible in Fig.3.9, in blue dotted line).

Towards fulfilling its objectives the DMM engine makes use of some information databases, mechanisms and interfaces. This implies the task of managing them and their information. Here is a short list of tasks that the engine performs to achieve its goals:

- Maintaining a database with the mobility sessions of the MNs connected under the X-GW. This is inherited and extended from the one defined in [25] and [37] as Binding Cache (BC);
- Sending and receiving the messages defined in [25], as PBU and PBA and the messages defined in [37] as Binding Update (BU) and Binding Acknowledgement (BA);
- Handling ND messages as RS, RA, Neighbour Solicitation (NS), Neighbour Advertisement (NA) to properly communicate with the MN at link-local level;
- Managing bidirectional IP tunnels and installing the appropriate packet forwarding rules to build the routing topology for the MNs;
- Local information exchanged with the FM through a defined interface, which contains a single primitive (`ActivateMobility`) that is composed of a request and a response message.

Operations

The DMM engine is by design a passive instrument, waiting for certain events to trigger its different mobility protocol procedures. In fact, the engine reacts to receiving or intercepting a specific set of four messages, from different interfaces: the RS intercepted from the X-GW interface connected to the MN, the PBU and PBA from the interface with other engines placed at other X-GWs, and the `ActivateMobility` request coming from the interface with the FM.

Upon receiving a RS from the MN, the engine will get the layer 2 identifier and from there, retrieve the domain-unique Mobile Node Identifier (MN-ID). Afterwards it will use a mechanism to retrieve the allowed corresponding HNP. With all this information the engine will make a BCE for this MN. The retrieved HNPs will then be sent to the MN inside a RA message that will enable the MN to configure one or more valid a addresses, allowing it to communicate freely with the network.

Receiving an `ActivateMobility` request from the FM located in the same X-GW, only happens when a MN has just arrived from a handover. This message contains the MN-ID and the addresses corresponding to any previous X-GWs where IP flows related to this MN are anchored. A match of the MN-ID is done against the BC returning the corresponding BCE. A PBU message is filled with the right parameters (exception made for the HNP mobility option, which must be set blank) and sent to the proper X-GW addresses, as the FM duly informed.

Being the recipient of a PBU means that at least one of the MNs that were connected to this X-GW have moved on to connect to another X-GW. The engine learns two facts from this message, the MN that moved by checking the MN-ID that is contained in it, and to which X-GW it moved by extracting the source address of the PBU.

After fetching the MN-IDs correspondent BCE, it will set the Proxy Care-of-Address (P-CoA) with the address of the new X-GW, and establishes a tunnel (if not already established) towards it. Following this, it will set up forwarding rules for the traffic concerning any locally attributed HNPs of this MN and route it through the tunnel.

The finishing touch is the sending of a PBA message towards the new X-GW (P-CoA address) filled properly with same MN-ID and the HNPs that were locally allocated.

Ultimately, the last expected message is a PBA. It signifies the mobility procedure, if successful, is about to conclude. Sent from the previous X-GW

to the current X-GW, the latter will set up the tunnel from here towards the previous X-GW and complete the BCE information with the HNPs extracted from the PBA, the tunnel identifier and previous X-GW address.

The next step is to set up the routes through the tunnel for the HNPs in the PBA, thus routing the uplink traffic. At last, the engine notifies the FM on the final status of the operation, by sending a `ActivateMobility` response message.

3.5.2 A touch of content

In previous sections it has been mentioned how media content has increasingly become the larger type of data causing traffic in mobile communications [14]. It also has been stated that the technology of choice by content providers, to address this change is the use of CDNs, adapted to the mobile user's needs.

Although a very interesting subject, the derivations and adaptations on CDN structures and protocols, that were developed to respond to this change in type of consumed mobile traffic, are out of the scope of this thesis.

Nevertheless, this thesis develops an architecture that intends to achieve its goals by going further than just its primary mobility function. Collaboration with other other well known technologies that share the same goal, like CDN, is both an opportunity and an advantage.

Concerning a collaboration with CDNs, our architecture is based on two assumptions, that are themselves based on the evolution of the latest approaches in CDN technology described in previous chapters. It has been described that a similar path is shared between mobile architectures and CDN deployments, that they are both being moved closer to the user and in a more distributed fashion.

Thus its clear that the first assumption is that content caching will become ever close to the user, and as an extension of this, a second assumption is that at the limit the caching deployment can overlap with the mobility architecture at the X-GW level. Enabling the possible existence of a content caching agent in the X-GW, means that the CDN can be fed locally with mobility information that can be used by it to optimize the distribution and delivery of content. Logically the step to be taken is to open an interface between the local caching agent and the entity that knows the relevant mobility information, the FM.

Interfacing with a CDN local caching agent, however, must not be a fact to be relied upon, since the deployment strategies of mobility and content might be different. Therefore the interface between the FM and a caching

agent is only enabled when both are present at the X-GW. Furthermore in this situation, FMs cannot rely in the possibility of exchange information that could be received from the content caching agent and put it to use to enhance mobility.

Therefore the interface assumes a different role of punctual information exchange that can be used by the CDN to explore internal optimizations of its own primary functions. Mobility information, like the MN-ID, flow information and the history of previously known mobility anchors can be used to recalculate the positioning of content, recalculate the best source of content for the MN and redirect its content fetching.

In practical terms, during operations the interface between the content caching agent and the FM acts as notification of user mobility once the MN has properly finished any connectivity and mobility procedures. In terms of information, this notification would contain the MN-ID, the concerned flow identifier and a possible list of previous mobility anchors.

3.5.3 Flow Manager

Being one of the largest contributions of the author in this thesis, the FM is the second half of the DMM core of this architecture, and in fact, the main intelligence of the mobility architecture. It is here that mobility decisions are taken, and mobility support is triggered, and enforced, through the collaboration with the DMM engine.

Vital to the mobility architecture, the FM is located at the centre of the architecture, in the X-GW, near to the access, thus possessing a good perspective on the this part of domain, knowing for instance which are the deployed access technologies and which Point of Accesss (PoAs) are available.

In such a central position it takes upon itself the most important task of monitoring MN communications, by keeping track of the flows that traverse the X-GW, as well as managing them.

Not limited to that role, it additionally executes the tasks of monitoring MNs movement and sharing mobility information with other components, becoming a true cross-layer component.

Combining the cross-layer information with its IP flow management capabilities and the central placement in the X-GW near the access network, we get a FM that possesses a broader vision, meaning knowledge, on what goes on in its domain. These characteristics, make it the place of choice to

implement and enforce a mobile operator's network policy.

In order to properly perform its functionalities, either DMM related, IP flow related or cross-layer related, the FM relies on both internal mechanisms and external interfaces. The arrangement of its internal mechanics can be seen in Fig.3.10. For the moment we start by describing what each of the internal blocks is responsible for. Following that comes the explanation on how they work together, describing the typical operations of the FM.

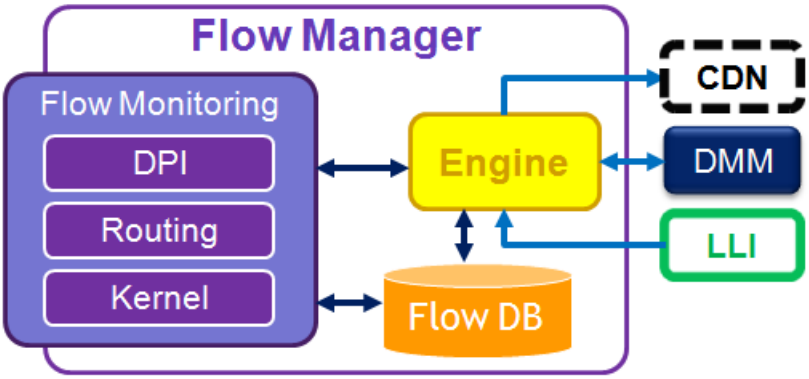

Figure 3.10: FM internal composition and interfaces.

Composing the FM are three main blocks: the Engine, the Flow Database and the Flow Monitoring block.

- Engine

  Here is implemented the logic and intelligence of the FM, in a state-machine fashion. Every decision is pondered based on internal policies, rules and flow information. Interaction with the Flow Database is mainly for consultation, altering the database information happens only in extraordinary conditions, when policy needs to be forced on specific parameters.

- Flow Database

  This is the main repository of information of the FM containing information on flows, such as, flow identifiers, source and destination IP addresses, transport protocol ports, associated Point of Access, link identifiers, MN identifiers, among others.

- Flow Monitoring

  Running in parallel with the Engine, its job is to fill the Flow Database with information, based on the policy established by the

Engine. Working closely with the operating system, it performs the duties of detecting active flows and gathering their related information. In addition, it works as the interface through which the Engine communicates with the X-GWs operating system to push its flow policies.

Flow Manager Operations

Main operations within the FM revolve mainly around the Flow Monitoring block and the Engine, while the Flow Database is more passive and only performs a couple of functions.

The Flow Database performs three implicit functions to support its main purpose as repository of flows. Firstly it feeds on the flow information input of the Flow Monitoring to create flow entries, secondly it triggers the Flow Monitoring to verify the existence of a flow when the flow's lifetime expires, and thirdly it enables the flow information to be shared with the Engine when queried for it, and updated by it when necessary.

Running separately from the Engine, the Flow Monitoring initially sets up packet filters and rules to capture the appropriate data packets. This means it is constantly doing Deep Packet Inspection (DPI) on selected packets. If an IP flow is discovered, he sends the flow information to the Flow Database which will either create a new entry if the flow does not exist in the database, or reset the lifetime of the flow, in case the flow is already registered. Besides feeding the database with information, the Flow Monitoring is also the intermediary between the Engine and the operating system in the X-GW. It executes commands on the operating system on behalf of the Engine. Such executed commands include: IP configuration, kernel communication and interface management.

The Engine itself is a state machine that acts upon triggered events. On start-up it triggers the initial configuration, DPI filter parameters and interface set up. Afterwards it waits for events to appear on its input queue, that can either be internal FM events (like a policy change notification), local X-GW events (like if an interface is put down) or messages from its external interfaces (as the reception of an `ActivateMobility` message from the DMM engine). In addition it directly consults and updates the Flow Database whenever required, as for instance, to know which flows belong to

a certain MN.

To better detail the internal interactions of the FM in a more descriptive way, below is a sequence of what exchanges happen internally in the FM when different events occur :

- Reception of a notification from the LLI.

  Consequently the Engine will try to ascertain if the reported MN is performing an initial attachment or otherwise a handover. To that effect, it extracts the MN-ID of the MN and checks the previous Point of Access (PoA) against its knowledge list of PoAs, which is provided by network policy. If the check returns:

  - A valid and mobility supporting PoA under the management of the current X-GW.

    Then we fall under an internal handover. Resulting in a simple update of the Flow Database concerning any flows that match this MN through the previous PoA, changing the values for the new PoA and technology identifier for the flows.

  - A valid and mobility supporting PoA under the management of another X-GW.

    Then this is an external handover, where a `Query Flow` request message is issued to the FM located in the previous X-GW. This message contains the MN-ID, link identifier of the MN, the technology identifier and both the current and previous PoAs.

  - An invalid or void return on the PoA value.

    Meaning that there is either no previous PoA or no record of it, or in an extreme case, the PoA is known but does not support mobility. All of these situations fall under the default umbrella that an initial registration needs be performed, and the FM terminates the sequence of events here.

- Receiving a `Query Flow` request message from a FM in another X-GW.

  A check is done on the previous PoA value, to validate that it concerns this X-GW, which if positive, triggers a Flow Database search for all active flows that match the reported MN-ID, MN link identifier, PoA. If there are any active flows, the Flow Database will return them to the Engine, which will add the flow identifiers to the contents of the `Query Flow` request message and create a `Query Flow` reply message, issued to the new FM at the new X-GW. After the message is issued, the Engine will update the concerned flows on the Flow Database with the new PoA, and technology identifier.

- Receiving a `Query Flow` reply message from a FM in another X-GW.

  Firstly a check is done to the PoA value to ensure this is the proper destination of the message. Secondly the Engine analyses the need for mobility of the flows. Flow identifiers are analysed for destination IP prefixes that are attributed by this X-GW, that if found, imply discarding the corresponding flows, since they are initially anchored at this X-GW and do not require mobility. For all other flows, their HNPs are extracted based on the same destination IP addresses present in the flow identifiers. These HNPs will be matched against the knowledge of the FM to know which X-GWs they belong to. The addresses retrieved are then put in an `ActivateMobility` request message, together with the MN-ID and the link identifier, then sent to the DMM engine.

- Receiving an `ActivateMobility` reply message from the local DMM engine.

  When such a message is received, the result code of the mobility operation performed by the DMM engine is verified. If negative the procedure ends, otherwise, the CDN component is notified with a message containing the MN-ID, both X-GW addresses, link and technology identifiers.

- Detecting an IP flow.

  When the Flow monitoring captures an amount of packets that pass through its filters, it will take note of its flow identifier, which is composed by: source and destination IP address, source and destination port number and the type of transport protocol used. This information is copied into a new database entry that is then completed with the link identifier which includes the PoA and the technology identifier. In the case that a specific flow is already registered in the database, then the Flow Database will alter its status to active and reset the lifetime associated to it to the default value.

- Lifetime expired IP flow.

  Upon the validity expiration of any of the active flows present in the database, the Flow Database will ask the Flow Monitor to check if the flow still exists. If yes, then the flow lifetime is reset to its default value, otherwise, the flow will be marked as inactive, and the lifetime value of the flow set to a specific value. Once the lifetime of an inactive flow expires, this flow is automatically removed from the database.

3.5.4 Link Layer Information

In pursuit of the objective of keeping a technology agnostic architecture, the Link Layer Information (LLI) block represents an abstract interface to an underlying technology specific module.

By means of this interface with the FM the underlying technology will provide the FM with two pieces of crucial information: the identifier of an attaching MN and the identifier of the previous PoA to which it was attached, if any. Both pieces of information are enveloped in a notification message and sent to the FM as soon as the MN attaches itself to the current X-GW. With this information in hand the FM is able to ensure the session mobility for the attached MN.

The implementation and mechanics of this abstract module are out of the scope of this thesis, but some examples are here mentioned as means to gather the information required by the FM.

- 3GPP Access Network Discovery and Selection Function (ANDSF)

  Some technologies like 3G and LTE, implement its own lower layer mobility mechanisms, or network functions from which this information can be more or less easily procured, as for instance the ANDSF [38]. Using a small piece of tailored software could be the key to that get such information from available interfaces and pass it on to the FM.

- IEEE 802.21 Media Independent Handover

  A choice already seen in section 3.4, is the use of the IEEE 802.21 MIH protocol to obtain the required information in heterogeneous environments [39], from across different types of access technologies.

- NIQ

  There are also some protocols that can interact with the MN, like the NIQ [26] that if specifically supported in the MN can be used to trigger the MN to share both its MN-ID an its configured IP addresses from which the previous X-GW could be extracted.

- Neighbour Discovery protocol

  As an *in extremis* solution, listening to the RS message sent by the MN can solve the issue if two conditions are met: firstly that the source address in the RS message is assured to be the one assigned by the last X-GW (unspecified address in case it is a first attachment); secondly if this does not interfere in any way with the exact same process that the DMM engine implements.

3.5.5 Architecture Work Flow

In this section the two most significant scenarios will be described to demonstrate how the architecture works together to perform its primary functions. Both scenarios described below contain the sequence of events and messages, the essential parameters that are contained within the messages, and the reasoning behind the executed step.

Starting with the procedure of a first arrival or attachment of a MN to a domain that supports the developed mobility architecture, the scene evolves to a the point where a handover to another X-GW is executed.

Initial Registration

When a MN makes an attachment to a domain that is under the control of the proposed approach, the expected initial registration procedure is quick and simple. Depicted in Fig.3.11, the procedure implies only a few steps: an initial attachment notification (1); a Neighbour Discovery (ND) procedure; and finally the MNs IP address self-configuration to obtain IP reachability.

Upon the attachment of the MN, the FM will receive from the local technology a notification message containing the identifier of the MN (1) (its Media Access Control (MAC) address for instance) and the previous PoA identifier. Receiving the message the FM extracts the MN-ID based on the provided identifier and checks that there the last PoA identifier is either void or invalid. The FM then concludes this is a new attachment and with no further steps to be taken will continue with its main task of monitoring flows.

As soon as the link layer attachment is complete, the usual following step of an IP terminal is to execute the Neighbour Discovery (ND) protocol (2) by sending a Router Solicitation (RS) message. The DMM engine will intercept this message, extract the link layer identifier and construct the MN-ID, with which it will fetch any Home Network Prefix (HNP) allowed to this MN. The available HNPs are then put into a Router Advertisement (RA) message and sent to the MN. Receiving this message the MN will configure a global address that enables him to start valid IP communications with Correspondent Nodes (CNs) (3).

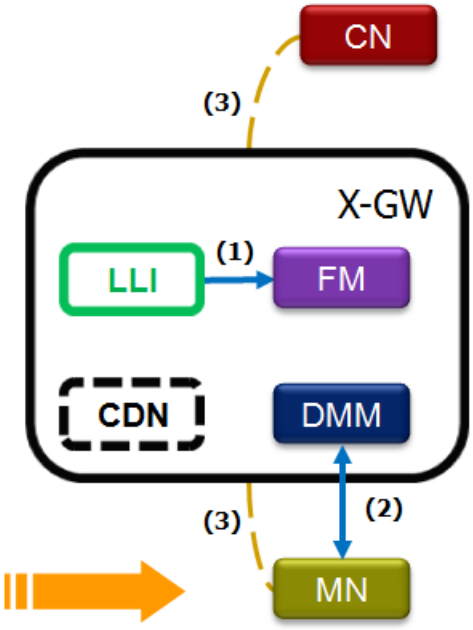

Figure 3.11: Initial Registration Operations

Handover

A handover process (Fig.3.12) presumes that an initial registration process has already been performed by the MN at some other X-GW in the domain that supports our mobility approach. In fact, in this architecture, a handover can be considered an extension of an initial registration, since the latter is always performed regardless of the nature of the attachment.

Just like in an initial attachment, this procedure starts with a notification containing the link identifier of the MN and the previous X-GWs PoA(1). It is issued from the LLI towards the FM, once the link attachment to the new X-GW has been detected.

At this point in time two actions are performed in parallel: the initial registration procedure at this X-GW is performed with the help of the ND protocol and the DMM engine (2); and the FM evaluation of the content of the LLI notification.

The FM notices that this MN has possibly performed a handover, due to the valid previous PoA present in the notification from the LLI. The

FM does a check for a known X-GW associated to this PoA. With the query returning a valid previous X-GW, the FM will immediately issue a `QueryFlow` request message to the FM located at the previous X-GW (3) to know which flows the MN had traversing the previous X-GW.

Receiving the `QueryFlow` request message, the FM in the previous X-GW will search its database for matches, retrieving the corresponding active flow identifiers. These will be sent back to the FM in the new X-GW inside a `QueryFlow` reply message (3).

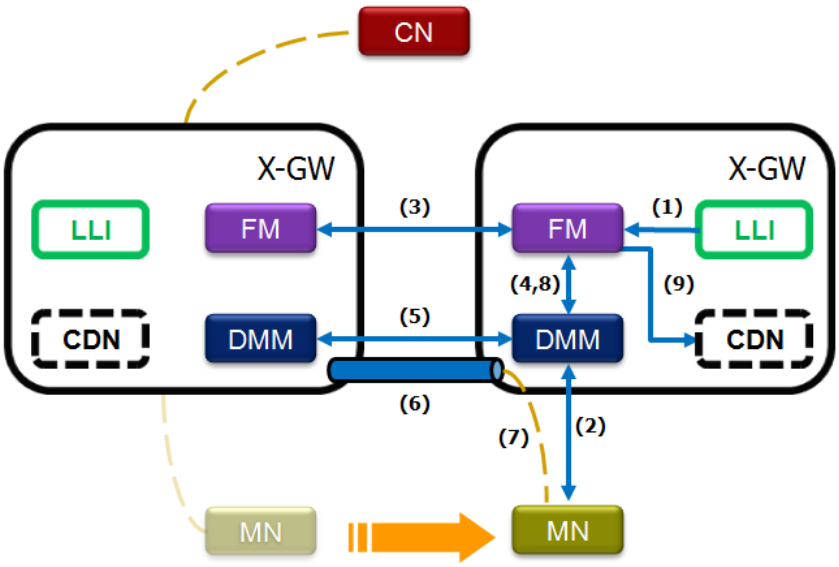

Figure 3.12: Handover Operations

Now, after receiving the `QueryFlow` reply message, the FM checks which flows need to be mobilized. Thanks to the received flow identifiers the FM also knows which X-GWs need to be involved in mobility process. Therefore the next step is to request the DMM engine to perform mobility for the assigned MN, by sending an `ActivateMobility` request (4), containing the MN-ID, the link identifier and a list of IP addresses of X-GWs where there are still flows anchored belonging to this MN.

At this point the DMM engine will do its required mobility procedures exchanging PBU/PBA sequences with each of the provided X-GWs (5), updating routing and establishing the proper tunnels (6) to forward the traffic flows, which can then be again delivered to the MN (7).

In the final stage, the DMM engine returns the result of the mobility

process FM (8). If the mobility succeeded, the FM will communicate the movement of the MN to the CDN agent, finishing the handover procedure.

3.5.6 Notes on Flow Mobility

A feature that has not been approached in detail but has been implicitly described along with the FMs capacities is the architecture's ability to support flow mobility. Flow mobility is a mechanism that targets mobility traffic at a more specific and detailed level.

Generally, mobility is done at MN level, meaning that all of the session traffic to and from a MN is integrally transferred from one PoA to another.

Some more sensitive mobility mechanisms are able perform mobility at an interface level, transferring the complete traffic session of a single interface of a MN to a new PoA. This is particularly useful for mobile devices that support multiple interfaces, either from the same technology, or (more commonly) from different technologies, enabling its interfaces to be treated independently in terms of mobility [40].

At a deeper level we find flow mobility [41]. The amount of IP connections belonging to the same session of a MN are naturally more than a few. Flow mobility is exerted at this level, where mobility is considered on an IP flow basis, meaning that instead of the whole traffic transmitted through an interface, it's possible to address the move of a single IP flow.

The best and most practical definition of a flow, in the scope of the work here presented, is by describing its characterized components. They may vary in number and type from one architecture to another. The essential set of flow attributes used in this work to classify a flow are five: the source-destination IP addresses pair along with the source-destination port numbers pair and the transport protocol type.

Thanks to the flow identifying capabilities of the FM, the mobility can be addressed at such a detailed level as an IP flow. Moreover, its interface with the DMM engine permits a coupling that results in a system that enables mobility to not only be addressed but also enforced at a flow level.

Benefits such as policy enforcement, traffic offloading and a more efficient use of the MNs access connections are then possible to be profited from. However, even if the architecture supports this, it does not mean that the MN is taking advantage of it.

In fact, for this level of flow mobility to be enforced at the MN, a component must be put in place at the MN to permit not only the proper management of the flows towards applications at the MN level, but also the

fashion in which they are distributed between the MNs interfaces. Furthermore, if the MN is required to provide information, enforce some mobility decision or be part of the mobility process in any way, an interface needs be set up with the FM and additional mobility signalling introduced.

Although the area concerning MN mobility entities is out-of-scope of the thesis, there are some solutions that could be used in this scenario, some being passive [42] and allowing a limited set of scenarios, while others are more proactive (like the CM [30]) enabling a larger set of possibilities.

3.6 Validating the Architecture

Concerning the validation of the architecture, within the scope of the thesis, the term is considered in the way that the components that constitute this mobility architecture were implemented, deployed and tested in real test beds while correctly performing their described functionalities.

Therefore, in the author's view, this architecture in order to be properly validated must have its components implemented and correctly functioning in a real test bed.

The proposed architecture core components (FM and DMM engine) were validated under the scope of the MEDIEVAL project. There, they were part of a mobility system that was included in a much larger architecture (see section 3.4). The implemented versions for this architecture were more complex and had added functionalities specifically for interfacing with other components of the MEDIEVAL system.

Moreover, the components were deployed in three different proof-of-concept demonstration scenarios: one focusing on inter-technology mobility (3G, LTE and WiFi) coupled with CDNs interaction; another on a multicast context, which executed mobility on a multicast session; and a last one on inter-domain mobility with non-MEDIEVAL domains that included network congestion mechanisms.

In these conditions the FM and DMM engine demonstrated that their concepts, design and implementations demonstrated their functionalities for mobility support, fulfilled their goals and achieved their purpose.

Although the obtained validation perfectly suits the needs of the thesis, as it showcases the abilities and functionalities of the mobility architecture in a real environment, efficiently contributing for an improvement of the Mobile Nodes Quality of Experience. In the perspective of the author, this does not seem quite enough.

The deployed proof-of-concept scenarios encompass much more than just

the mobility and draw focus away from it in favour of other concepts, like multicast, medium congestion, inter-domain and content caching. And not only that, but the demonstrated scenarios are very limited in terms of number of deployed elements (both MNs and infrastructure nodes) due to the costs involved with deployment.

Moreover, we must point out that evaluating the performance of such a small amount of elements, with a larger architecture involved, on which the mobility system represents only a part of, gives a not very accurate glimpse on how the mobility system itself actually works in a more reasonable scenario.

The following question was therefore posed. Now that the proposed architecture was defined and its validation achieved, how can we proceed to properly evaluate, in a more meaningful way, the proposed mobility contributions?

This would clearly imply a new deployment, of a more select number of components, focused on mobility, in a larger scale. This would also imply finding a good experimentation platform on which to deploy the core of the mobility architecture and its components. In addition, it would require to define a set of experimentation parameters and properties to be evaluated.

Thus leading to the subject of a test bed built to evaluate the mobility architecture that is discussed in the next chapter.

3.7 Conclusion

This chapter started by building up on the base left from the conclusions drawn from the analysis of the current trends present in the state-of-the-art. Early on, the author adopted the main trends into the heart of its work and reflected those choices in the definition of the main pillars of the envisioned mobility architecture: flatter hierarchy, closer to the edge and with distributed functionalities.

Furthermore, the author defined as well a set of desirable practical requirements to be followed by the design: it should be in line with current trends; build on existing work; agnostic in terms of access technology; if possible interface with other architectures and protocols; and be seamless to the MNs.

From there it was a logical next step to expose an initial draft of the architecture. With this starting point, it was opportune to search for similar existing protocols or solutions that could fit the requirements so that the

architecture contribution could be built upon existing work.

Searching the state-of-the-art, the author came across the Distributed Mobility Management concept, which upon analysis had several similarities with the defined base of the envisioned architecture, it was then a simple choice to decide on incorporating DMM into the work, taking a moment to classify the draft architecture contribution as a fully-distributed and network-based type of DMM architecture.

Having accepted DMM as the type of base framework for the mobility architecture, the author explored some ideas on how to combine protocols to obtain a DMM mobility scheme based on the fully distributed classification given to the initial draft.

Firstly came a contribution that combines a modified PMIPv6 with DHCPv6. The intention is to take advantage of the centralized structure of DHCPv6 for signalling and the distributed data plane of the modified PMIPv6. This approach however, did not suit our purposes, since it is only partially distributed.

On a second attempt, the author tried combining the modified PMIPv6 with the NIQ protocol for a fully distributed scheme. Still, this did not suit the requirements as well due to the MNs direct intervention on the mobility procedure.

Continuing the search for an alternative that could provide a DMM based framework, the author decided to try to take advantage of its own contributions in the European Union Framework Programme 7 (EU FP7) MEDIEVAL project.

To introduce the subject, an overview on the mobility-related aspects of MEDIEVAL was given, familiarizing the reader with the project's mobility architecture, the mechanics of it and especially the contributions that the author made to the project.

In this perspective it was easier to explain to the reader how the author leveraged the work done under MEDIEVAL towards the thesis, showing how the coupling of components such as the Flow Manager and Unicast Mobility Engine could be used to form a DMM framework. Importing these components into the thesis required some modifications that were conveniently explained.

With a DMM core framework fully built up, came the time to detail the complete mobility architecture. From high-level functionalities to component internal mechanisms, including their interfaces and interactions with other technologies such as CDN. In the end the work flow of the mobility architecture was shown for two scenarios, a MNs initial registration and a

handover between X-GWs.

Although the architecture contribution was finalized, there was still the unsolved matter on how to prove that the work would be proven to be valid under real circumstances, therefore posing the question on how to validate the architecture contribution.

To mark that point, the author demonstrated how the mobility architecture components were validated through the deployment in real proof-of-concept test beds, showcasing their functionalities and capacity as a mobility scheme under the umbrella of the MEDIEVAL project.

Left at the end of the chapter is the author's opinion on the insufficient conditions provided by the proof-of-concept test beds for a meaningful evaluation of the mobility scheme. This is based on a lack of scale of the validation platform, whose results reflect a very large architecture and not just mobility, in addition to the conditioning circumstances of the very specific deployed use cases.

The question is then posed on how a meaningful evaluation could be performed to provide results on the performance of the mobility architecture. Question to which the next chapter aims to provide an answer by proposing a test bed for a scaled deployment of the mobility architecture.

Chapter 4

A Multifaceted Testbed

The previous chapter discussed the complete mobility architecture, from base guidelines and initial draft, passing through the mobility components and going into the detail of protocol mechanisms and exchanged information. In addition it introduced some other contributions that were produced in the process of developing the architecture. Included as well in the last chapter was the way in which the mobility components were validated.

Contributions present in this part of the thesis can trace back their origin to the last question posed on the previous chapter: how a meaningful evaluation could be performed to provide results on the performance of the mobility architecture.

Trying to answer the question, the author starts by picking up on the reasons that he believes were limiting the ability of the validation platform to produce more meaningful results, such as, lack of scale and the deployment of an architecture from which mobility is only a smaller part. These reasons will be a focal point in the definition of the requirements that will drive the construction of the evaluation platform as well as the selected method on which it shall be based.

In order to go beyond the validation platform, focusing more on the mobility architecture within a scaled environment, the author starts by redefining the guidelines of the needed platform. However, not only the guidelines need be redefined, but we also need to rethink other aspects: the requirements, desired features, evaluation method, tool selection, use case scenario, and deployment set up. This array of aspects is to be considered one by one, throughout the chapter.

Thus the contributions that the author produces in this chapter are extensive. Firstly the type of evaluation method is different from the considered standard track and will require some work to properly function. Secondly the process of experimenting different tools, and trying them to comply with this unique evaluation method, will produce the final test bed solution, that forms another strong contribution.

Summarizing, this chapter defines, develops, implements and tests a new type of test bed, based on a mix of evaluation methods. Moreover, the test bed will be used to evaluate a the proposed mobility architecture on a scenario, that, in its essence, is as similar as possible to the validated one. The mobility architecture deployment though, is done in a manner more meaningful for evaluation, by scaling the size of the deployed network and more focused on the mobility system.

Chapter Contents

Section 4.1 describes the reasoning behind the evaluation platform, and the definition of requirements and features that author thinks would be desirable to include into the evaluation platform.

Section 4.2 takes us on tour of the most popular evaluation methods, towards selecting the one that would be most convenient for our evaluation platform.

Section 4.3 shows the research work on the state of the art for tools that can perform the required tasks while following the selected evaluation method.

Section 4.4 gives us an overview on the selected tool on which to base the platform, going through its components and features.

Section 4.5 features the account on the development of a software suit that was developed specifically for this platform, to achieve satisfaction of the totality of requirements for the test bed.

Section 4.6 concludes the chapter, making a reflection on the achievements and contributions of the author.

4.1 Purpose and Requirements

As mentioned previously, the designed mobility architecture was to be validated in a way that was already described in the previous chapter (see sections 3.4 and 3.6), while its evaluation is performed through the deployment of the architecture on a test bed built for the effect.

Therefore, in the overall scope of the thesis, the main purpose of this test bed is to provide a platform on which to perform a meaningful evaluation of the designed mobility architecture. A very close second purpose is to explore a new method that would enable test beds to deploy mobile networks.

In this light, based on the existing mobile architecture design, its components and features, and taking into account the desirable conditions for the evaluation of the design some requirements were put up for consideration as guidelines on to help deciding which method of approach and tools are a more appropriate choice.

A list of requirements and desirable features is given below, along with the reasons that make them worthy of significance.

The test bed itself *should*:

- T1 : Be scalable.

  One of the focus points of the evaluation of the mobility architecture will be the deployment of the X-GWs. For the evaluation to be meaningful, several nodes and network elements should be deployed, therefore it is prudent to consider the test beds scalability as a main requirement.

- T2 : Have a low resource footprint.

  As a derivative based on the scalability requirement, the deployed elements must have a small resource consumption, thus improving the overall performance of the test bed.

Network elements in the testbed *must* be able to:

- N1 : Run a real IP stack.

  Running a real IP stack permits on the one hand reliable IP protocol functions without the need for modifying the code designed, and on the other hand, more realistically accurate results by using the real stack.

- N2 : Run Linux customized kernels.

  An inherited requirement from the DMM engine since most out-of-the-box Linux distributions do not come with MIPv6 extensions integrated in the kernel by default. These extensions are crucial to produce PBU and PBA messages.

- N3 : Run already deployed code, preferably without modifications.

  If the test bed can run code that was already deployed on a Linux machine, that would save much needed development, implementation and test time, as well as avoiding cross-platform incompatibilities that might come from it.

- N4 : Run measurement tools.

  Having a test bed that cannot measure the performance of the deployed elements goes against the purpose of test bed itself. Therefore the test bed must have or be able to support measurement software.

- N5 : Simulate Mobile Environment

  Evaluating the mobility architecture design, means that test bed must include mechanisms that, at least to a light degree, reproduce the conditions of a mobile environment.

In addition, the following features would be desirable:

- Interaction with real machines.

  This ability would be desirable to promote work in cooperation with other systems or architectures enabling at them at least to interface from a network perspective.

- Simulate Wireless Technologies.

  The possibility of including models of wireless technologies in the simulated mobile environment, would add to the accuracy of produced results.

- Node Movement Input.

  Introducing mechanisms or models of node movement adds not only diversity to the collected results, but also a higher level of realism to the results.

Bearing in mind the long list of requirements, the next steps consist in going through a careful selection of the evaluation method, which the test bed will use as starting point to evaluate the architecture.

Afterwards, the same list will be pondered while choosing framework and tools that could be helpful by becoming part of the test bed. And finally, these same items will later enter into account at the development and implementation of the test bed.

4.2 Evaluation Methods

There are mainly four globally acknowledged approaches towards evaluating a system's behaviour and performance: mathematical modelling, simulation, emulation and field testing, simulation and emulation [43] [44]. These methods are quite different and each one has its own advantages and disadvantages. In this section surface of these methods will be touched, explaining their essence and noting the essential advantages and issues of each one.

The objective is to give a light background on the most relevant evaluation methods to, so that a selection of can be conducted as to which method is the best fit to our goals. Moreover, this decision is based on the features listed in the previous section, which will be matched with the conditions offered by each evaluation method.

4.2.1 Mathematical Models

Mathematical modelling is the most simple way of evaluating a system [45]. It has been the oldest used method in research, enabling the advance of science since mathematics were invented in the early days of human civilization.

The principle of this method is quite elementary. The rules and features a system to be evaluated are to be abstracted or simplified enough to the point of allowing them to be described as a set of mathematical rules or relations. This set of mathematical rules could then be mathematically manipulated as to produce a study how one or several of its variables perform under a reasonable set of assumptions.

The advantages of mathematical modelling make this method the preferred one among analytical researchers, especially thanks to the broad level of abstraction allowed. Additionally, the intrinsic mathematics mean that data output availability is only conditioned by the time it takes for the model to be mathematically solved.

The issues with this method are mainly related with the correlation of its results with the real world results, meaning the accuracy of the results. More often than not, the results obtained by this method are not in line with the ones obtained from field testing.

4.2.2 Simulation

Simulation is one of the most widely used methods for evaluating a systems performance. Its origins are closely connected with the definition of system. A system could be seen as a black box that generates output. Output being the result of certain input conditions or otherwise be self-generated output [45].

To simulate a system is regarded as to reproduce a systems behaviour on a computer platform. Or better yet, as definition we can state that the concept of simulation is the computerised representation of the reproduction of behaviours of processes or entities, which exist in the real world. A generalization of this definition could apply to reproducing any type of system and its behaviour.

Reproducing a behaviour of a system requires therefore a good degree of knowledge regarding its components, procedures and communication protocols, to achieve a good level of accuracy. After realizing the depth and limits of the system, the rules that control it are transposed to a model that can be interpreted and processed by a computer. Since computer logic is mainly based in mathematical logic at the most simple levels it is only legitimate that models can possibly be expressed in a mathematical format or similar [45].

From the several characterizing features present in the simulation method, the most striking is the nature of its discrete-event modelling. Discrete-event simulation models are not based on time as its main dimension, but the change of state of the system, or in other words, they are triggered when an event happens. This is what enables a simulation to run in a different time scale than that of a regular clock, since it advances from event to event instead of second to second for instance.

The use of discrete events in simulation enables an effect of time compression on systems that do not produce many events, while the opposed phenomenon of time expansion happens in the less frequent occasions where systems produce large amounts of events in short periods of time.

The advantages of a simulating a system, can only be achieved if the correct choices and steps are made in the modelling process. Although creating simulation models is a more complex task, there is a reflected gain in accuracy as a result.

Environment and external factors are very much controlled within certain defined boundaries. By using discrete-events it can drastically reduce the amount of time to run an experiment and therefore reduce the time to obtain results.

Another advantage is that simulations can be applied when pure mathematical analysis is not adequate and still obtain useful results with some accuracy.

The issues are mostly related with the critical modelling process. If the simulation model does not address properly the aspects of envisioned output data and systems behaviour, it can produce rather inaccurate results.

Furthermore a balance between complexity and the depiction of real world conditions must be carefully observed. A rather complex and detailed model will always only be an approximation of real world conditions and will consume heavy computational resources as well as a larger amount of time to produce results. A simpler model will be easier to construct, have lighter resource footprint and quicker to produce results, at a cost of a larger inaccuracy of results.

Other points such as the simulated scenario conditions, experiment repetition, experiment replication, and result analysis method can easily introduce distortion on obtained results [46].

4.2.3 Emulation

Despite being a less used method, emulation is a step closer to field testing than simulation. Emulation mitigates some of the critical issues encountered with simulation, in particular the lack of accuracy caused by gap between simulation models and real-world conditions.

Though based on simulation, emulation goes a few steps further. While in simulation a model is created based on the approximated rules of a system, in emulation a system's behaviour is analysed and, in effect, duplicated it by means of a software program in a computer environment.

When viewed by another software piece that used to communicate with the original system, the emulated software behaves in the exact same way as its original counterpart, thus removing any need of change in the com-

munication protocol.

Therefore a generally accepted and well-rounded definition for emulation is that emulation is a way to simulate computerised (or similar) entities by modelling the behaviour of physical components and using real implemented software to interact with them in a context scenario. This implies however that in emulation, unlike simulation, the environment runs in real time and not in a simulated time driven by discrete events.

In the networking field, the emulation method is commonly applied by having emulated software pieces replicating the behaviour of either other software or even hardware. In fact, one of the largest use cases of emulation is to replace hardware for a software emulated component, and later be used to interact with real deployed software. This technique is common today in the employment of, for instance, virtual machines, and virtual networks, two of the best emulation examples that include emulation of a system or systems in tandem.

The advantages of emulation are based on what can be accomplished from the successful duplication of behaviour and how the real parts take advantage of it. For one, the ability to integrate real implemented software in the environment saves development time by avoiding the construction of simulation models for every component. Using already developed software in an emulation environment, enables a much more accurate evaluation of its performance, and in a reverse perspective, software that is developed in such an environment profits of a better platform of evaluation, providing indications of expected behaviour.

Not less important, is its real time feature, implying real time analysis, which helps greatly in coping with dynamic behaviour and unforeseen events, namely bugs, errors, and other limitations.

In a more specific case, for a network related perspective, it enables to see the result when a link is available to real networks. Opening this door makes possible to analyse the interaction between emulated components and a real network as well as how it can handle dynamic network behaviour and topologies.

The issues in emulation mainly revolve around the usage of a real time environment and its resource footprint. Since the interaction with real world systems implies the use of a real time clock, unlike simulation, this means that an emulation environment cannot profit the advantage of time compression, meaning that the environment will take longer to produce the results than simulation could.

Introducing real components, instead of their simpler simulation mod-

els, results in a larger consumption of resources. In fact, it extreme situations, usually caused by low resource availability, it may happen that delays are introduced in the environment, causing difficulties in obeying real time constraints and complicate behaviour that can then lead to erroneous results. [43]. Scalability, for instance, is a characteristic prominently afflicted with this issue.

4.2.4 Field testing

At the other edge of the spectrum facing the theoretical mathematical modelling, rests field testing. Using nothing else but real deployed hardware and software, field testing is the most reliable method to collect measurements, obtain results, and evaluate methods.

Have been used for a myriad of scenarios, its main goals and purposes come together as to demonstrate the behaviour of the deployed components and the general system under the real circumstances that it is supposed to work under. Moreover, under these real world conditions it stands as a platform to analyse faults, preliminary results and dynamic behaviour of the deployed system.

Due to the opportunity presented of real world conditions, it is best used to prove the feasibility of a system, or a set of features that its components perform, meaning in broad terms that it is the choice method for proofs-of-concept demonstrations. In addition, it is also considered the a good method for gathering reliable preliminary results on the performance of the deployed system, making it possible to give a notion on the strengths and weaknesses of the system under real world conditions.

The advantages of this method are focused on the high proximity of its environment with a fully deployed solution. Implying the proof of the systems feasibility in real world conditions, is one of the most desired features, which also imparts with the notion that the results obtained are very accurate and reliable. Furthermore, it gives the ability to detect deployment problems under real world conditions, providing an image on behaviour and performance of the tested system in a real environment.

The issues with such a method start with the amount of effort required. The cost is considerably higher in terms hardware and software required, when compared with other evaluation methods. In the same line, a larger manpower effort is also required to set up, deploy and manage the system.

Results can be hard to reproduce and replicate, due to the ever changing real world conditions. In some cases the system's set up must be adapted

and quite specifically, making for a very narrow scope where such results may be reproduced and still keep its relevance.

4.2.5 Method Selection

Having detailed the popular methods which are largely acknowledged as common ways of evaluating the performance of a system deployment [43] [44], we now go quickly over their *pros* and *cons* analysing with particular detail their compatibility with the required and desired features for the test bed (see section 4.1).

Mathematical modelling becomes rather inadequate, when its considered that we are speaking of a selecting a method to create a test bed and used in the validation of a designed mobility architecture and its implemented components. The intents displayed in items N1, N2 and N3 alone easily dismiss the idea of using mathematical modelling alone as means to attain our goals.

Simulation could be the method used and is indeed encouraged by item N5. In addition, they perform very well in large experiments, which would help with item T1, and consume a lower amount of resources, conforming with item T2.

Unfortunately, when thinking of the requisites, the need to run real code (item N3) and a real IP stack (item N1), the usage of simulation poses an issue, since it could not support these requirements. Therefore, although required by item N5 simulation is not advisable as a standalone method for our purposes of evaluation.

Emulation by itself could be the most adequate method, since most of the points considered in section 4.1 could be easily achievable through it. Although some of them, namely items T1 and T1, could be afflicted by the use of emulation, a careful monitoring of the evaluation parameters could mitigate such danger.

The only issue then remains with the wireless part (item N5) which would be very complex to emulate and is crucial to the validation of the proposed mobility architecture design.

Field testing, despite providing a platform for a more accurate outcome, is usually only performed following extensive research using other methods. The development cycle of a platform using this method is long, very costly and uses a considerable amount of resources, thus immediately posing some

obstacles. This collides directly with items T1 and T2, which aim at making sure that a large deployment is possible.

None of the above methods alone, provides a platform that can be used as a base for the validation of the proposed mobility architecture. However, there is a method that comes close to ticking all the needs, emulation. The only part that is not desirable to be emulated (the wireless environment), could, by its own, be simulated. Connecting both parts could prove not to be as straightforward as it seems, but this path could prove easier than emulating the whole wireless environment.

It is then proposed to develop an approach that combines these two methods: emulation and simulation. In simple terms the tasks and fields of responsibility are as follows: emulation will be used to support the architecture nodes and ensure most of the requirements stated in section 4.1; and simulation will concern itself with the features related with the wireless mobile environments.

Performing the connection between the two will largely depend on the tools selected for the test bed, and in finding a way in how they can coexist while still conforming to the test beds aims and goals.

In conclusion field testing is too expensive to perform at the intended scale, mathematical models are inadequate for our goals of evaluating deployed software behaviour, and simulation can only perform part of the required features. Emulation on the other hand covers most of the procured items in the requirement list, though not all.

In view of these specificities, if simulation and emulation were to be combined, that would cover all the requirements. Therefore, the author proposes in this thesis the development of an approach combining simulation with emulation, working under the principle that the emulated nodes of the mobility architecture will be connected to and through a simulated wireless mobile environment.

Benefiting from the best of both methods, this mixed, or hybrid approach, will have deployed real software components of the mobility architecture in emulated nodes connected by a light simulated environment, aimed at producing better results and better able to cope with scaled deployments of its nodes.

4.3 Selecting the Framework Tools

Bearing in mind the conclusion of the previous section, mixing emulation and simulation methods will certainly complicate the task of finding tools that both respect the test bed requirements (see 4.1) and follow the mix of proposed evaluation methods.

Therefore we are going to take a quick look back at the requirements described in section 4.1, and ascertain how to tackle this selection by analysing what major constraints we have, and reach a general idea pointing towards a generic type of tool that we could employ.

Firstly the set formed by T1 and T2, define characteristics that are important but can be tuned with more or less hardware, therefore the author decided to use them as differentiator factor between tools rather than a primary quality to search for in the desired tool.

Secondly, the requirements posed by N1, N2 and N3, are vital, since they are very specific and concern directly the conditions imposed by implementations of the FM and DMM engine. Not respecting these requirements outright, might later prove extremely difficult to run the DMM core of the mobility architecture in the resulting test bed. Thus these are critical and will be highly prioritized in the search of the right tool.

Continuing on the requirement list next come N4 and N5, the former could be generally supported if N1 and N3 are respected. The latter point, is a hard point to address, and an isolated point in our mix of approaches as was described before. Consequently, it will be searched for as a second level priority.

As for the remaining desirable features, they will have the lowest priority while considering the tools, meaning that if the tool happens to not support them, it is the author's opinion that they could be provided in another way, as for instance, extensions developed specifically for the tool selected.

Analysing the highest priority requirements, these pull the scales towards an emulation tool that needs to be highly customizable. Pointing to a more concrete direction (including the wireless environment) a good type of tool to start with could be a network emulation tool. Network emulation frameworks usually contain the mechanisms necessary to deploy and manage several emulated network elements, and are even detailed to the point of enabling a certain degree of network element customization. Some particular ones can even, in a way, support a wireless environment.

Below follows therefore a quick overview of the most popular frameworks of network emulation, including a high-level assessment of their features. At the end, we will expose the reasons that led the author to choose the tool

presented in section 4.4 as the basis for the evaluation platform.

4.3.1 GINI

Developed by the Advanced Networking Research Laboratory (McGill University), Gini [47] is a tool kit that builds virtual networks composed of User Mode Linux (UML) machines. As many of the tools in this field, it was born in the educational field, intent on helping to teach students in subject of computer networks.

In order for students to perform their projects, the platform has a real IP stack so that real applications can be deployed on the network elements. The whole framework is open source, and in its Linux version is based on the virtualization of network elements, by means of the UML methodology. Adopting a *what you see is what you get* philosophy, it employs its own Graphical User Interface (GUI) and allows network elements, and the desired connections between them, to be created easily. Gini's GUI also enables to configure the network elements, such as their IP parameters and MAC addresses.

For efficiency purposes, it uses a customized UML file system for the virtualized machines. It makes distinctions between the entities based on their role in the network (routers, switches, wireless Point of Access, Mobile Nodes and terminals) and runs customized versions of routing protocols and UML tools.

The advantages of Gini, are essentially its fast learning curve, thanks to ease of use and deployment from the GUI, a real IP stack, and the presence of wireless extensions (WGINI) which could be helpful in our platform. Gini is lightweight and in addition, experiment deployment can be run remotely on another machine.

The issues seem to be sourced from some of the advantages, for instance, the GUI has been reported as sluggish. But more importantly, its wireless extensions only support one PoA connected to one MN, being unsuitable for reproducing mobility at the scale we require.

4.3.2 VNX

Born in the Telematics Engineering Department (DIT) of the Technical University of Madrid (UPM), Virtual Networks over Linux (VNX) [48] is a framework aimed at managing and deploying virtual networks and virtual

nodes supporting a large pool of types and operating systems. Similar to Gini, it also has incorporated the same educational purpose, as well as a friendly open-source nature.

In fact, VNX is an evolution of another tool on which its based, Virtual Network User Mode Linux (VNUML), profiting on the latter's knowledge and experience. This has enabled VNX to support several new features in comparison to VNUML, such as, the deployment of virtual machines running different operating systems, the emulation of routers from renowned manufacturers, and management on a *per* virtual machine basis. In addition, it supports interaction with real networks and a scaled approach is offered by means of specific distributed extensions.

There are two central elements composing the core of VNX, a scenario and network descriptive language based on Extensible Markup Language (XML) that feeds the configuration into the main VNX program that deploys and manages the virtual network and its virtual elements.

The advantages presented in VNX include its open-source nature, the support for multiple operating systems, the different types of network elements supported, and a scale-specific approach that favours distribution.

The issues are only a few but rather important ones, first and foremost is the non-existence of wireless elements and extensions, thus focusing on fixed networks, and secondly the fact that a the generic and multi-platform approach brings a cost of a larger resource footprint.

4.3.3 Mininet

In line with the previous tools, Mininet [49] is also a tool that performs the deployment and management of networks composed of virtual machines that embody different network functions. This tool however was built with a different purpose at heart, the deployment and test of OpenFlow [50] applications and networks.

Its close relation with the OpenFlow technology can be seen in the several specific features it has that are tailored for supporting it, namely, special versions of network elements, specific routing features, and a command line interface that enables the execution of OpenFlow related functions.

Mininet has some similarities with some other previous tools, like its open source nature, possibility of connection with real networks, and use of

a tailored Linux based environment on the virtual machines. Network deployment and management is done through a customized Application programming interface (API) written in Python, like the vast majority of the framework.

The advantages of this tool relate, naturally, to the OpenFlow technology, but are not limited to it, and boast the deployment of a very large number of network elements with a very small resource footprint. In addition, it can interact with real networks, and its customized deployment enables a fast boot procedure.

The issues concern a lack of support for wireless network elements, and the fact that the tool is very much oriented to OpenFlow presents some constraints on the deployment of a normal and regular network, since, for instance, OpenFlow interference on routing could pose a problem when deploying the DMM engine in such a network.

4.3.4 Marionnet

Initially designed as feature to complement another tool (Netkit [51]), Marionnet has since been redefined and evolved into a framework on its own right, that deploys and manages virtual networks and its virtual hosts by itself. Not unlike other tools it was intended to perform educational functions, namely, teaching students to experiment with computer networks.

Although nowadays it does not bear any connection with Netkit, it does have the same approach, by deploying UML virtual hosts running Linux, inside a host that is running Linux itself. Furthermore, it supports its virtual networks to communicate with real networks and offers a GUI, reducing the learning curve of the tool and making it more user-friendly.

Mostly programmed in OCaml, Marionnet deploys its virtual network connections with the help of Virtual Distributed Ethernet [52], profiting from its ability to manipulate the virtual Ethernet conditions, while enhancing the deployment with its custom dynamic reconfiguration mechanisms. Moreover, it comes with its own customized Linux distribution and provides mechanisms to connect a graphical interface to any deployed virtual machine.

The advantages in Marionnet have to do with its dynamical network features, specifically its ability to reconfigure the network topology and a

GUI that is automatically updated. In addition the ability to give a graphical interface to the deployed nodes is an interesting feature, as well as the ability to communicate with real networks.

The issues with Marionnet reside in the absence of wireless extensions, and the base language of development (OCaml), which limits the possibility of developing extensions on this platform.

4.3.5 Netkit

Developed at the Computer Networks Laboratory of the Roma Tre University, Netkit [51], is like many others described above, a framework initially developed for teaching computer networks to students.

It provides the necessary mechanisms for the deployment of virtual networks composed of network elements that are in fact UML based virtual hosts. In that sense, its performs not only the deployment, but also once deployed, it manages the virtual networks and corresponding elements.

In order to perform these tasks, Netkit uses a script-like description of the scenario that it parses to know the elements to deploy as well as their network connections, and then uses its own version of UML based mechanisms to enforce the described connections and virtual host parameters.

Virtual hosts are allowed to run their own Linux based distributions as well as their own file systems and Linux kernels. To save time to the user and reduce the learning curve, Netkit comes equipped with a small custom file system, a complete Linux distribution and a custom kernel, resulting in a small resource footprint.

Additionally its network mechanisms enable the interaction with real networks and even other Netkit deployed networks in other real hosts. As many of the other tools it is also open source.

The advantages of using Netkit are the scale of the allowed deployment thanks to the small resource consumption, the ability to interact with other networks, and a good level of customization allowed.

The issues on Netkit are similar to those seen in other tools, the absence of wireless network elements and use of tailored components used in its virtual hosts.

4.3.6 Conclusion and final choice

Looking back at all the described tools, we can deduce that although different in their details, they are all very similar in nature, most of them coming from university laboratories, and started as educational projects to teach classes of students on the subject of computer networks.

In a more deep inspection, all of them are belong to the category of virtual network deployment and management frameworks. They make use of emulation and virtualization as method to deploy several network entities in a single real machine, and in fact, use the same mechanism is used to attain said virtualization, UML.

According to the prioritized requirements, by using UML as a means of virtualization, the tools ensure that they satisfy N1 (see more on section 4.4.1). However, concerning N2, Gini makes no reference to supporting this feature, thus being excluded. All other tools explain how a different kernel can be patched to function properly with the tool.

Since none of the remaining tools has wireless extensions, this means that another tool or an extension needs to be combined with the selected tool for achieving the requirement N5. In addition, implementation details now need to be considered as well in case the author needs to develop the wireless extensions that could be added to the tool.

Posing this new factor in perspective, Mininet is excluded thanks both to its very narrow application scope and possible OpenFlow routing interference with the DMM engine. Still in this subject, Marionnet is excluded due to its implementation in OCaml, a language the author is not familiar with.

Moving to the next deciding factor, the ability of the tool to scale and have a low resource consumption, VNX seems to be the least desirable option, therefore being excluded as well.

The last standing candidate is Netkit, a tool that conforms with the large majority of requirements, the only exception being N5. Nevertheless, the author will use this tool as base for the evaluation platform.

Towards satisfying N5, the author will look into this matter in the following sections. Different solutions will be explored, and the reader will see how either by combining Netkit with other tools or by developing extensions for Netkit, the author will arrive at the final solution for the evaluation platform.

4.4 Netkit

Already introduced in the previous section, Netkit [51] is an emulation environment with the ability to deploy and manage virtual network elements, and their virtual network connections. For the purposes of this thesis, in this section we will lightly approach the way that Netkit functions, how it makes use of UML and how the virtual networks are built.

Depending very much on User Mode Linux (UML) and its managing tools, Netkit instantiates each element of the required network as a Virtual Machine (VM) and manipulates them to form the desired network. Afterwards these elements can be fashioned into the different roles required by the user, through the installation of appropriate software.

Netkit is basically composed of a set of scripts and tools that work together to ease the task of deploying a UML virtual network. For instance, the topological description of the network is expressed in a file using a script-like language, that is parsed at the start up of every experiment to create the elements of the network with the adequate network configuration.

Deployed VMs are connected together by an emulated network. This is achieved through the use of virtual hubs and `uml_switch`. Both are processes in the user space and work in the same manner, but only `uml_switch` allows VMs to communicate with real networks, since its directly connected to a `TUN/TAP` interface deployed in the real machine.

In fact, the network interfaces of the VMs are unix sockets that connect to another unix socket belonging to a hub or a switch. The process running that switch or hub will replicate and forward the packets received to everyone else that is connected to that same hub or switch.

Technically, the `uml_switch` provides the physical layer simulation, while link and network layers are provided by the UML kernels of the VMs, therefore forming an emulated network.

4.4.1 Virtual Machines and UML

User Mode Linux is a linux kernel that has been modified to incorporate the ability of being run as a regular process in a linux machine. As such, it already has in itself all the necessary mechanisms to construct a virtual host, like memory and network, among others. Thus its able to control its own processes, being is seen as a normal process from the point of view of the real machine, while from the point of view of its own generated process its seen as a kernel (see fig.4.1). It is thanks to this kernel perspective, that normal user programs can be run in a standard manner, without modification.

While virtualization specific tools usually take an approach that applies

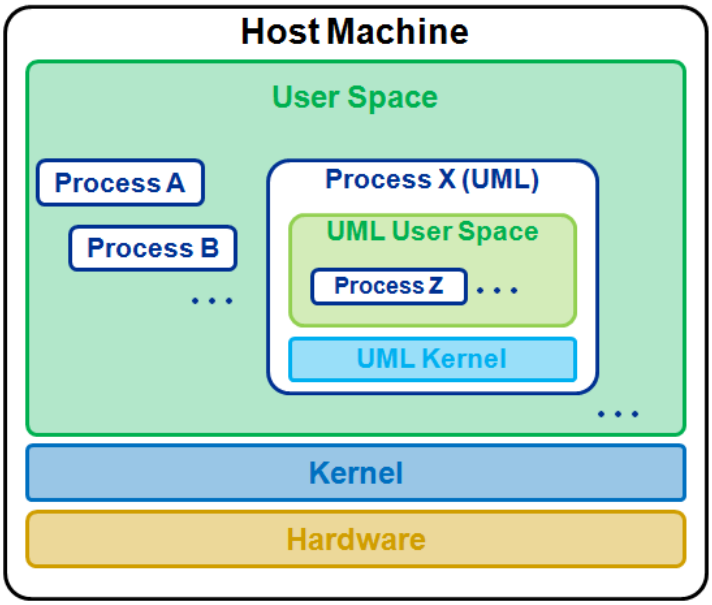

Figure 4.1: User Mode Linux (UML) running within a host.

an abstraction layer at hardware level, the type of virtualization used in UML, happens at the level of the kernel hosting it, thus depending on it and being limited by it for what type of operations are supported by its architecture.

The process management in a UML VM is very simple, any process initiated in the VM is equally instantiated in the host. Memory is managed in a way that allows the UML VM to dynamically share part of the hosts memory, and as for storage, an image of a file system is used, containing an installation of a linux distribution to be used by the UML VM. In addition, to interact with a user, a UML VM can attach a terminal interface to a file descriptor. Networking in a UML VM work in the much the same fashion as real machines.

To help with the management of VMs, UML comes with at set of tools, that enable to configure and interact with the VMs. Among them are some that are very relevant, such as, the `uml_mconsole` and the `uml_switch`. As stated above, the `uml_switch` is a network support tool, while the `uml_mconsole` is a tool that allows the user to configure devices and stop, pause or restart a VM.

4.5 SWEEN

Since that none of the tools that were researched for the evaluation platform filled all the test bed requirements exposed in section 4.1, the author decided to compromise, and selected the best fitting tool among them. The tool Netkit, is a network emulation framework that was introduced in previous sections.

Netkit however, does not possess the means to simulate a wireless environment, being designed for wired networks. The author took upon himself to develop its own solution so that Netkit could comply with the specific requirement N5 and become the desired evaluation platform.

Thus, developed specifically for this thesis, the Simulation of Wireless Environment Extensions for Netkit (SWEEN) is a set of tools or small programs, whose purpose is to complement the Netkit network emulation framework, with a simulated wireless environment.

To produce the wireless environment, SWEEN bases itself in some small new logical entities and reuses tools that are already in place, like the `uml_console` and the `TUN/TAP` interface, thus reducing its resource footprint. The drawback of this, is of course, a heavy dependence of SWEEN in UML, same as Netkit.

At an initial stage, the author investigated how Netkit and UML work to provide the networking emulation, and drew the thought of developing a similar approach to the one by `uml_switch` but bearing in mind a wireless connection. This wireless switch would just replace the regular `uml_switch` as the instantiated network device. However, this proved much to difficult to implement in a short time frame.

In the process of studying the network mechanisms, we came across the possibility of changing the VMs network device point of attachment (virtual network), on the fly, by using the `uml_console` tool. If the right parameters are used, one can change the network onto which a VM is connected, thus producing the principle of movement.

Despite this possibility, there was an issue with the PoA change on the fly, in fact, this is only possible, when the VM network device is down. This difficulty became more of an obstacle when the author noted that a mechanism was needed inside the VM to perform up and down operations on the VMs network device.

To overcome this, the SWEEN logical entities were devised. In a server-client fashion, the SWEEN Server, would effectively manage the connections to the Netkit virtual hubs, from the host side, while, the Client on the VMs acts as the enforcer of the server commands.

Upon devising this sort of mechanism, another obstacle arose, the need

for a means of communication between the VM and the host. Fortunately UML already has a mechanism in place that can help with this, the `TUN/TAP` connection of the `uml_switch`. Therefore the one had only to deploy a virtual network of this type (IPv4 only not to interfere with the rest of the test bed), and connect all MNs to it, enabling communication between Server and Clients.

As a particular point of interest the author noted that on the SWEEN Server, it was possible to track the connections of MNs at a lower level. This is exactly one of the features that could complement the mobility architecture's need to know the previous connection (PoA and X-GW) of the MN.

The author then developed an interface that would permit this information to reach the LLI component of the mobility architecture. Communication between the Server and the LLI module present in the X-GWs reuses the mechanism already in place that enables Server-Client communication, needing only to connect the X-GWs to the same virtual network. Thus creating a SWEEN virtual control network.

4.5.1 Structure

Shown in Fig.4.2 the architecture of the architecture of SWEEN is based on three main components: the Command module, the Server and the Client. Two are located on the host (Command and Server) and one on each MN Virtual Machine (the Clients). There is also an interface with the LLI which will be described later in the operations of the software.

Command

The Command component is run in the host machine and exists as a common interface and unifying method to input movement Commands to the MNs in the Netkit emulated test bed. It implements a translation from command format to an event format acceptable on the Server (communicated to the Server via UNIX socket). It both capable of treating commands issued on the fly, and of reading files and executing the commands contained within, for a more autonomous and automatic way of execution.

Server

Being the main block of this software set, the Server performs quite a few functions. It keeps track of the current connections between X-GWs

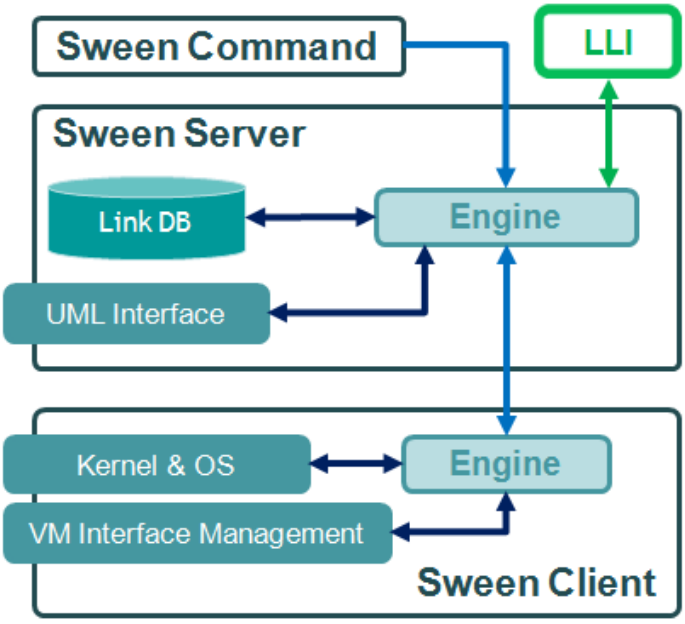

Figure 4.2: SWEEN components and interfaces.

and MNs, in its local database. Any mobile node movement input is received from the SWEEN Command module (through a UNIX socket). Moreover it communicates with the `uml_console` to rearrange the Netkit virtual hub attachments, disconnecting the mobile node from the current Point of Access (PoA) virtual hub and connecting it to the new PoA virtual hub, thus simulating the wireless behaviour through changing the endpoints of the virtual wired links. Furthermore, it also has the responsibility of notifying the Link Layer Information (LLI) present on the new X-GW of the new attachment of MN to that X-GW.

Client

Clients are executed in every virtualized MN. They listen for messages from the Server on the `TUN/TAP` interface which is connected to the real host. The Client keeps track of interfaces currently active in the MN, and manages their bonding status. It enforces any command to switch on/off any of the MNs interfaces, so that the Server is able to rearrange the connection.

4.5.2 Work Flow

Operations in SWEEN are quite simple, although a bit extensive, they can be seen in Fig.4.3.

Any mobility operation starts from the Command module where a mobility command is issued (1), its source, being either a user in real-time or a file. Receiving this command, the server will check its link database to see if the movement is allowed, if not, the server emits an error log message and continues regular operations, otherwise, it will notify the Client (2) present in the proper MN Virtual Machine of which interface to put down.

The Client will then interact with the VM kernel to put down the interface (3) and reply to the Server with the result status of the operation (4). The Server next step (5) is to call the `uml_console` to unlink that interface of that MN from the current Netkit virtual hub socket, and connect it to the desired virtual hub.

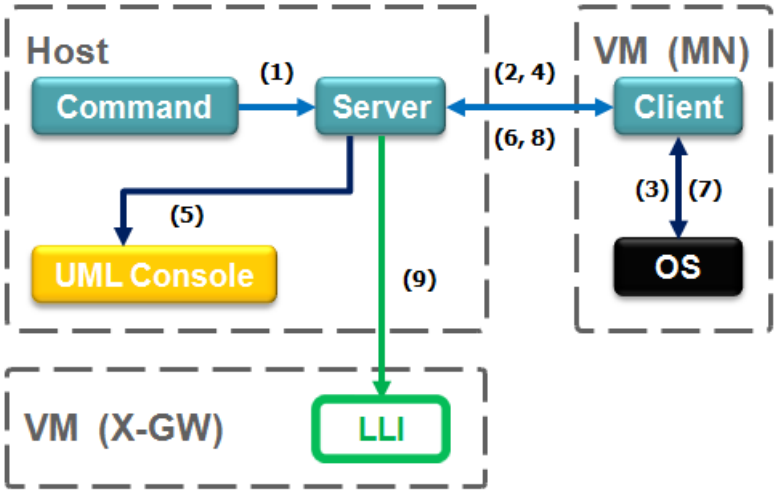

Figure 4.3: SWEEN Operations.

After successfully linking to the new X-GWs virtual hub, the Server will again contact the same Client as before (6) to reactivate the network interface. Following a communication with the VMs kernel (7), the Client will first enslave the interface to the bonding interface and then reply to the Server the result of the operation (8), and the Server will update the database accordingly.

The final step consists of a notification to the now current X-GWs LLI module (9) giving it the information on the MNs identifier and previous X-GW Point of Access.

4.6 Conclusion

In a nutshell, this chapter defines, develops, implements and tests a new type of test bed, based on a mix of evaluation methods (emulation and simulation).

The author therefore started by defining the essential requirements and features that the evaluation platform should have. Following an analysis of the popular evaluation methods, the author realized that no method alone could satisfy all of the requirements, thus opting for a unique way, in which simulation and emulation were combined. Such a hybrid method enables emulated nodes to deploy the elements of the mobility architecture to be connected through a simulated wireless mobile environment.

Continuing, the author searched for available tools that could help to implement this unique hybrid method, which led to choosing Netkit, that fulfilled all the requirements except one, the simulation of wireless environment.

Crucial as it is to have the required test bed features, the author decided to develop himself the needed wireless simulation extensions for the Netkit tool, resulting in a small set of software pieces named SWEEN. These extensions, much like Netkit are based on User Mode Linux and make extensive use of its features.

With all the requirements fulfilled, the tools chosen, developed and implemented, the evaluation platform became a reality. Thus being ready for the next step, the evaluation of the mobility architecture, discussed in the next chapter.

Chapter 5

Notions on Performance

Building upon the contributions of previous chapters, we are now able to deploy the mobility architecture in our purpose-built test bed. Thus we will begin this chapter by declaring which goals we are trying to achieve with this evaluation an how we intend to achieve them.

Later, we go into the set up of the evaluation platform, depicting general the way that the experiments were conducted and how the results were collected, mentioning the tools used. Notions on the performance of our contributions are given in two parts: firstly the test bed's own performance is checked and secondly we see how the architecture behaves in the environment.

As is customary in this document, at the end of the chapter a conclusion is presented, summarizing the results obtained and commenting on their significance.

Chapter Contents

Section 5.1 briefly states intentions of the evaluation, as well as detailing the conditions of the evaluated scenario, and test bed parameters.

Section 5.2 displays the collected data referring to the test bed itself, including system base reference data and test bed limitations.

Section 5.3 takes us to the results concerning the architecture, namely the impact on resources and protocol behaviour.

Section 5.4 concludes the chapter rounding up the most important deductions of the collected results.

5.1 Goal & Setup

Since the beginning of this document, that the author has stated his intention to evaluate the work produced in the mobility architecture (see section 1.2). In the pursuit of that objective the author went to the lengths of building a special test bed that follows a specific evaluation method, and enables a larger scale deployment.

The purpose of this evaluation is therefore to give a notion of how the mobility architecture behaves in a larger scale. But first, we must access the performance of the test bed itself before deploying the architecture, so that we know the individual impacts of both test bed and architecture. Proceeding in this fashion will also help us to source out where limitations are located.

Providing this notion on performance, is a set of relevant parameters that has to be defined together with the method used to measure them.

As far as blunt performance goes, the consumed computational resources of both the test bed and the architecture must be analysed. Regarding the test bed, a look to its scalability limits by means of number of deployable nodes is a must as well. As for architecture itself, there are several important parameters to be collected, such as, X-GW and MN resource consumption, information on handover, an overview of the signalling overhead, the existence of mobility tunnels, and the scalability of the architecture on the test bed platform.

5.1.1 Tools

To get the desired measurements, the author procured tools that have a small or no impact on the system. Therefore priority was given to tools that existed already in the operating systems of the test bed machine and the Netkit UML machines. Next in the preference line were small tools that could be easily installed. For cases where no tool would suit, the author decided to integrate some functions in the deployed programs, to obtain the required data.

Here follows a list of tools used to sample measurements:

- `Sar` [53]: is a highly flexible to obtain system statistics, like CPU and memory usage on Linux machines.

- `Tcpdump` [53]: is a very useful program that analyses network traffic, by copying the network's data packets, enabling to instantly see what is going through the network interfaces, or save it for later analysis.

- `Vlist` command [51]: is a tool integrated in the Netkit framework. It lists in real time details of the UML machines running in the system, such as, network interfaces, and connected networks.

- General code timestamps and printouts: this feature was developed by the author and is embedded in the SWEEN software suit.

5.1.2 Configuration

Having chosen the parameters to sample and the tools to obtain them, we now go onto the details of the test bed configuration. This is done for two reasons: to establish a base on which to enable others to reliably reproduce the experiments done here; and so that these parameters can be taken into account when analysing the results.

- Host Machine
  - System: HP ProLiant ML370 G5
  - Processor: Intel Xeon Quad-Core CPU E5410 (2.33 GHz)
  - Memory: 12 GB
  - Operating System: Ubuntu 10.04 LTS (kernel: 2.6.32-39-generic-pae)

- Netkit (v2.8) UML machine configuration:
  - Kernel: 2.6.32-61 UML
  - File System: F2.8 (distributed with Netkit)
  - Memory: 32MB - 512MB (min-max)
  - Operating System: Debian 5 (Lenny)
  - Simultaneous boot: 10 VMs

5.1.3 Use case scenario

Taking as base a similar version of the scenario used for validating the architecture, we expanded it to encompass a larger number of X-GWs and a far larger number of MNs (100).

In a nutshell, there is a large amount of MNs roaming through fair amount of X-GWs, while exchanging data with a deployed CN that is in a network beyond the X-GWs, as can be seen in Moreover, MNs movement is dictated by a specific mobility pattern.

Overall we deployed different versions of this deployment during one hour experiments. In order to better see the scalable behaviour produced by the architecture, the X-GWs were deployed in series of 5, 10, 15 and 20.

In addition, to see the reflection on mobility based on node movement, two different patterns were used: normal (Gaussian) distribution and uniform distribution; both with a specific cell residence time based on a Gaussian distribution.

	On average (secs)	std.dev. (secs)
Cell Residence	300	60
VoIP Session		
Length	180	60
Interval	600	180
Video Streaming		
Length	300	60
Interval	600	180
FTP Session		
Length	900	180
Interval	600	180

Table 5.1: Scenario configuration parameters

The data exchanged between the MNs and the CN is done thanks to the `mgen` [54] tool. We used it to produce three different types of traffic flows that can be used simultaneously on a per MN basis: VoIP, video streaming and File Transfer Protocol (FTP). Table 5.1 exposes the details on the data traffic flows, and MN movement details.

Please Note that all result values exposed in the next sections should be, by default, assumed as the average of all the executed replications of that experiment, unless stated otherwise.

5.2 Evaluating the Test bed

To better understand the test bed in general, and to know the limits of its tolerable usage, we measured the amount of resources that the test bed uses when deploying UML machines out-of-the-box. In fact, this measurement also gives us a notion of base reference for the coming results.

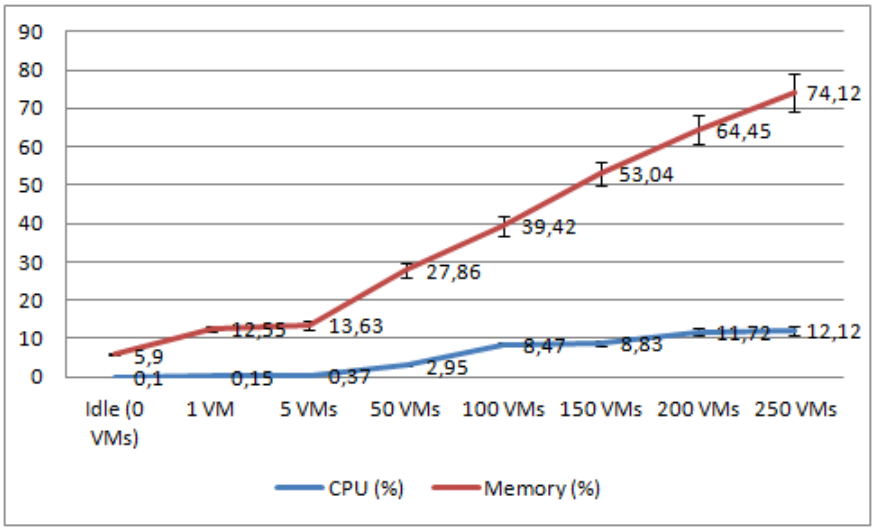

Figure 5.1: Test bed Scalability Reference

As can be seen in Fig. 5.1, we started by measuring the whole system while idle, then bit by bit we started to increment the amount of virtual machines, though not executing anything on them, reason why the memory and processing lines grow apart.

Two very interesting points can be seen on this figure: the first one is the tiny difference between one and five deployed VMs; the second is that no line on the graph reaches near 100%. Starting on the former point, this is verified because the Netkit framework is deployed as well, signifying that it is heavier than a few VMs. On the latter point, we can say that we started having some stability issues with the operating system's GUI whenever we deployed more than 200 VMs.

Therefore a limitation of the test bed concerning scalability is the number of deployable VMs, which, in our hardware conditions should not exceed 200.

Upon deploying large number of VMs, we started noting another issue, the test bed would take an enormous amount of time to deploy and shut down. To mitigate it we activated an option for parallel start up of VMs, drastically shrinking the set up time. However there is no similar option

for the shut down process, except executing the crash call (instant shutting down, but possibly damaging the VM filesystem).

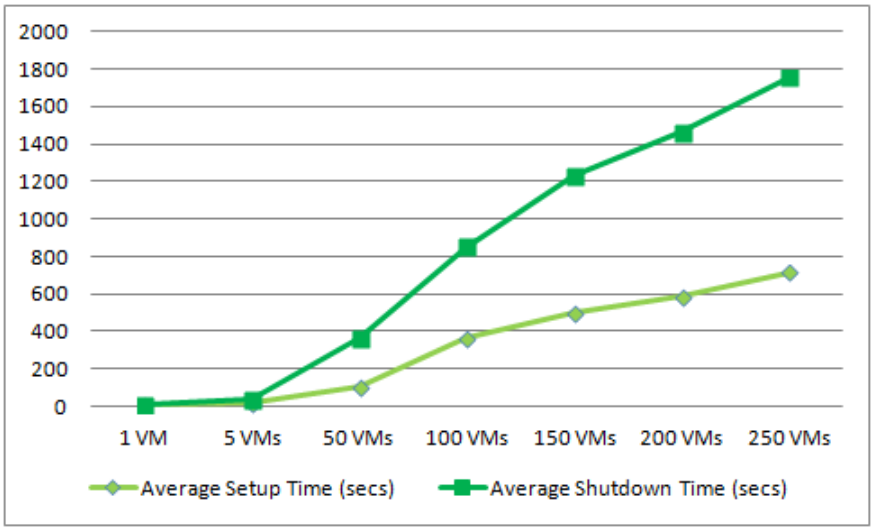

Figure 5.2: Time needed to start up and shut down the test bed.

Investigating the needed time to deploy and shut down the experiments resulted in Fig. 5.2, where we can see that after a certain number of VMs, it may become impractical to use the test bed, because the time required to deploy it, might outweigh the desired time to run the experiments.

5.3 Evaluating the Architecture

5.3.1 X-GW vs. MN

Knowing how much an X-GW and a MN cost to be deployed is useful when planning large deployments. In our type of case, it can help the author to decide on a better balance of both elements. Moreover, as can be seen in the figure below, both can prove very different under varied circumstances.

Figure 5.3 breaks down the footprint of both X-GWs and MNs in the several steps of the build up to run an experiment. As expected, X-GWs have a larger footprint than MNs, as well as a large difference between an idle and a running test bed, especially in what concerns CPU.

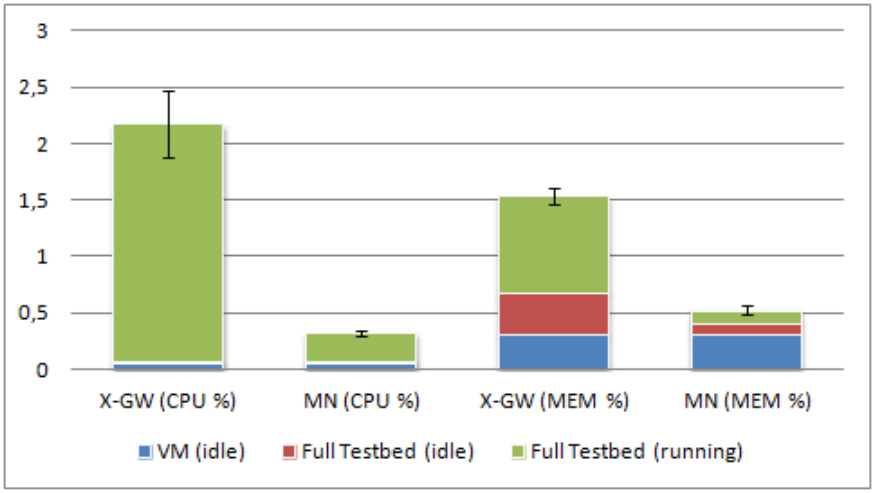

Figure 5.3: Impact of X-GWs and MNs

Interesting enough is the impact of the mobility architecture (the green and the red bars).

On a MN the impact is quite small, only 0,2% in both CPU and memory (meaning a MN will occupy around 64MB of RAM). That is only natural considering that the MN only deals with a small SWEEN component and the exchange of data with the CN.

On a X-GW, the same impact is larger, around 2,1% in CPU and 1,2% in memory (around 187 MB of RAM), but smaller than expected, since it performs quite a lot more tasks.

5.3.2 Handover & Signalling Overhead

In a complex environment such as this, it is important to know the costs involved on a handover. Metrics such as latency and overhead, can give a rough idea of that cost. Moreover, such handover metrics can alert us to any protocol performance issues that might occur.

Concerning the handover, in average there are 1148 per one hour experiment, meaning roughly 11,5 per MN. As for the latency of the handover, it averages at 182 milliseconds from the order to move, until receiving an IPv6 address. Most of this time refers to time spent by the SWEEN component (129 milliseconds) and the rest to the architecture mobility procedures. A simple initial registration procedure is, of course, much faster, standing at

an average of 56 milliseconds.

In the matter of signalling overhead, an initial registration produces an average of 189 bytes, mostly of which relate to the Neighbour Discovery protocol (176 bytes). A handover however, implies a lot more signalling, reaching an average of 739 bytes, in which the largest share goes to the PBU/PBA exchange (336 bytes) and RS/RA exchange (176 bytes).

So we can infer that the handover takes a bit more time than expected to complete, considering it's a simulated procedure, and it's caused mostly due to the undeperforming of SWEEN. In addition, the overhead produced is large but comes mainly from standardized protocol exchanges (PMIP and Neighbour Discovery), thus meaning that the signalling added by our approach is rather small and scalable.

5.3.3 DMM Tunnels

A good way to have the notion of protocol load for our mobility architecture is by seeing the amount of tunnels created by the DMM engine, since these are only created when a roaming MN needs session continuity.

Moreover, to test the behaviour of the protocol we employed two different scenarios, where the MNs move based on different mobility patterns. On the one hand we have a normal distribution, where there is a notion of neighbourhood, meaning that the MN will, based on a higher probability, roam between adjacent X-GWs. On the other hand, the uniform distribution, is more equal and fair on the choice of next X-GW.

Observing the results in Figure 5.4, one can see that the normal distribution is rather kind to our architecture in terms of scalability, while the uniform distribution is always using a high percentage of possible tunnels. These results are a clear reflection of the reuse of tunnels by the DMM engine, according to PMIP operations.

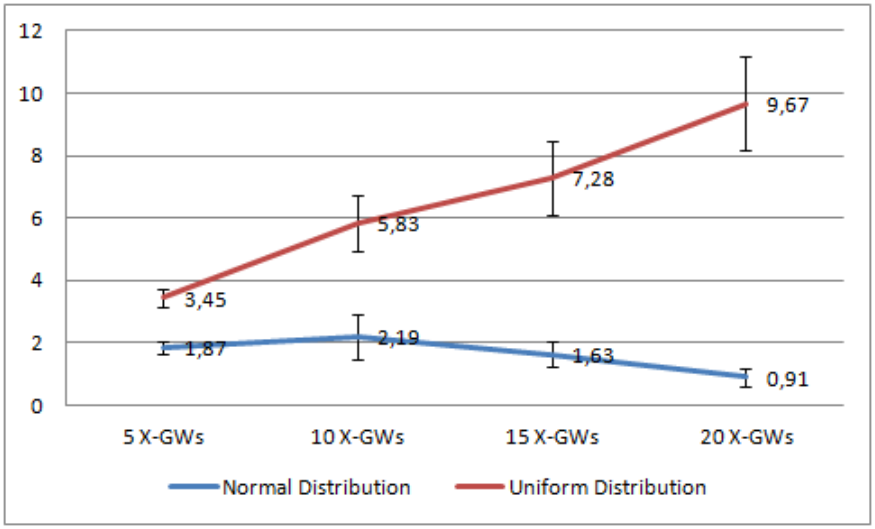

Figure 5.4: Average tunnels per X-GW

5.3.4 Overall Footprint

To have a clear image of the scalability performance of both the test bed and the mobility architecture together, we took a look at the overall resources consumed by running experiments with the full set while gradually increasing the number of X-GWs.

Seen in Figure 5.5, is the confirmation of a hint taken from Figure 5.3, where we see that the test bed, although initially requiring more memory to run will eventually require more processing capability than memory, the more X-GWs are deployed.

Focusing on the scalability perspective, the test bed is fairly scalable achieving a moderate amount of elements, while deploying the designed mobility architecture.

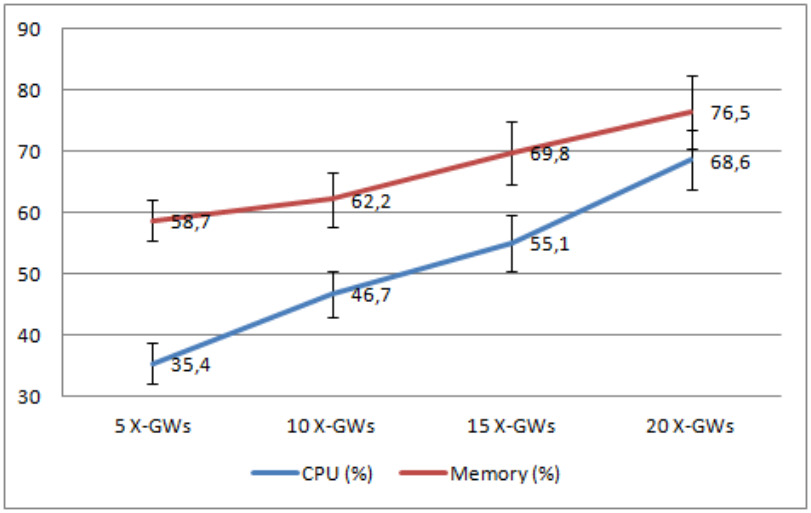

Figure 5.5: Architecture Scalability

5.4 Conclusion

Concluding the evaluation process, this chapter has shown how the test bed has been set up, which measurements were taken and which tools were used to sample results. In addition, the test bed configuration and details on the experiments were given.

The use case scenario was based on the validated deployment, but in our case it was scaled to contain a large amount of MNs and a number of X-GWs. A uniform and a Gaussian model were applied to the mobility pattern of the MNs, enabling them to roam between X-GWs while exchanging different data flows with a CN.

The initial process of ascertaining the limits of the test bed, shown us that support could only be given to about 200 VMs. Additionally, we discovered that the time required to shut down the experiments is rather large and can prove unmanageable for large deployments.

Concerning the amount of resources consumed by the MNs and X-GWs, results show that the impact is stronger in the processing and memory of the X-GW, limiting a good deal the amount that could be deployed.

During experiments, the test bed itself demonstrated a specific trend in resource consumption. It requires more memory than processing capability for a small amount of X-GWs, that when increased, will show a higher appetite for processing capability than memory.

Handover and signalling metrics were collected as well to give a notion

on the protocol overhead and handover latency of the platform. The results proved that a low signalling overhead is attained and handover latency is not ideal, noting the performance of SWEEN in this matter.

Applying different mobility patterns to the MNs has shown us that the mobility architecture is very scalable for MNs that roam between adjacent X-GWs, and rather the contrary when MNs follow different paths.

In the end we have produced a moderately scalable environment, with a deployed mobility architecture that performs better when there is a notion of movement continuity in the MNs mobility pattern. Moreover it is characterized by a low signalling overhead and a moderately low resource footprint.

Chapter 6

Conclusions and Future Work

After having presented the work and contributions of this thesis in its various branches and stages of development we now arrive at the final words. Here we shall gather the final thoughts on the presented work, and stipulate some of the possibilities that this work opens up. As an ending note, a list of the produced publications by the author is put together.

6.1 Conclusions

Influenced by several factors, the typical mobile user of today consumes large amounts of data, within which video content is by far the largest portion. To tackle the steep increase in demand, Internet services started replicating and distributing its content, while mobile operators, much to the same purpose, deployed new access technologies, more efficient and with higher throughput.

Despite these several efforts, the mobile user's Quality of Experience (QoE) continues to suffer. Upon these terms the author proposed to improve the QoE of mobile users, by defining a novel mobility architecture, its intelligence and control mechanisms. In addition, validation of the architecture was deemed necessary, followed by the design and test bed implementation of an appropriate evaluation strategy.

Concerning the design of the architecture, it adopted the Distributed Mobility Management (DMM) philosophy, that promotes flatter and closer to edge mobility management approaches. Moreover, the author considers important that it possesses an access agnostic approach, can interface with other technologies and enacts transparent mobility towards the MN (by not involving it the mobility process).

Based in this the author contributed with a few possibilities to form an acceptable architecture, selecting the best among them as a base for the mobility architecture. The selected framework, combines cross-layer information and makes use of flow tracking and flow management capabilities, while supported by a fully distributed version of the well known Proxy Mobile Internet Protocol version 6 (PMIPv6) protocol.

The needed validation of the final architecture came to be realized under the MEDIEVAL project, which saw the integration of the DMM core in a larger architecture, that was afterwards deployed in three different test beds composing different use case scenarios.

With the clear intention of evaluating the work in a more meaningful

manner, the author constructed a customized evaluation platform. This platform employs an innovative evaluation approach that mixes emulation with simulation, more specifically, the test bed platform emulates Linux hosts, in a moderate scale, by means of a virtualization tool named Netkit. Complementing this tool are the SWEEN software extensions, enabling the evaluation platform to acquire basic wireless environment simulation mechanisms.

The result is a platform that can run software modules developed for real machines in a medium scale virtualized environment that simulates MN mobility.

Entering the evaluation itself, the author took as base one of the simple use cases already validated, and expanded the scale, to analyse both the test bed as well as the architecture behaviour.

The maximum advisable number of VMs that the test bed can support is around 200, above that limit the test bed becomes unstable. We have shown as well that the resource footprint is more severe on memory for smaller deployments, while processing capability is the concern in larger deployments, but in sum the deployment of the architecture has a lower that expected impact on resources, and a moderate scale of deployment is attainable.

Furthermore, the test bed handover process noted that the architecture has a low signalling overhead, and a larger than average latency, considering we are simulating the wireless connections.

As final note on the performance, we must refer that this mobility architecture has been seen to thrive in an environment where there is a notion of MN movement continuity, rather than randomness.

6.2 Future Work

Because in the research field, the world moves in a fast pace with new developments happening every day, we look back at the work already thoroughly presented in this thesis to search for its possible future.

Searching for aspects and details that can be improved or built upon, the author will expose its views on what space is there to improve, or even how the work produced can team up with other existing efforts, seek interesting and innovative paths to follow.

Starting with the proposed mobility architecture, several new paths can be probed. For instance, since the architecture already supports flow mobility, this subject could be worth exploring further. Encouraging this idea

is the fact that the evaluation platform already supports the deployment of software in the MNs, as well as the capacity for multiple MN interfaces.

Another aspect that can be developed further is the articulation of the DMM core on a different type of platform altogether. A most desirable topic of investigation would be, in this case, the interaction with 3GPP technologies, such as LTE. This could be done on either of two styles: the more straightforward way of deploying DMM on top of a 3GPP architecture or the more rewarding process of integrating the DMM core features in an architecture more 3GPP-alike.

On the perspective concerning the evaluation platform, a possibility that was hinted at in [55] was the coupling of Netkit with Network Simulator 3 (NS-3). Some work [56] [57] shows that this approach is promising and should be given a good look, given that it is to be done on a different requirement base than that of this thesis. The work exposed in [55] shows that it would be reasonable to pursue an integration between Netkit and NS-3. The resulting test bed would profit from a larger scale of MNs, while not suffering from some of the drawbacks of the solution presented here, even though it would force us to lose some of the test bed features.

Still on the same perspective, it would be interesting to see results with detailed and improved models of the wireless environment simulation technologies. Moreover, to spice up the mix, introduce models from different access technologies.

Adopting a more vertical view on the work presented here, some more future work opportunities arise when thinking in what type of support is the architecture and the evaluation platform leaning on. For instance, the exploration of possible alternatives to the LLI module has already suggested several paths worthy of investigation. In the author's opinion, the most interesting one is the interaction with ANDSF mechanisms.

From another perspective one could see advantages in working together with the Application-Layer Traffic Optimization (ALTO) [58] architecture, by creating an awareness of cost [59] inside the mobility architecture, that could be reciprocated towards ALTO by providing it with mobility information, namely giving an insight to traffic anchoring in the topology.

6.3 List of Publications

We could not finish the document without showing a bit of the recognition bestowed on the several contributions of this thesis, by the research community. Therefore, here follows a list containing the publication efforts of the author on several formats, including, scientific articles, Internet-Drafts and demonstrations:

- F. Bo, G. Kunzmann, M. Wetterwald, D. Corujo and R. Costa, *QoE-aware Traffic Management for Mobile Video Delivery*, Workshop on Immersive & Interactive Multimedia Communications over the Future Internet, IEEE ICC, June 2013, Budapest, Hungary.

- R. Costa, T. Melia, L. Eznarriaga, F. Giust, A. de la Oliva, and C. J. Bernardos, *Wireless Multi-Access Delivery for SVC-based Video Applications*, MONAMI, September 2012, Hamburg, Germany.

- R. Costa, T. Melia, D. Munaretto and M. Zorzi, *When Mobile Networks meet Content Delivery Networks: challenges and possibilities.* MobiArch workshop at ACM MOBICOM, August 2012, Istanbul, Turkey.

- F. Giust, A. de la Oliva, C. J. Bernardos, and R. P. Ferreira Da Costa, *A Network-based Localized Mobility Solution for Distributed Mobility Management*, MMNF workshop at WPMC, October 2011, Brest, France.

- J. C. Zuniga, C. J. Bernardos, T. Melia, A. de la Oliva, R. Costa, A. Reznik, *Distributed Mobility Management: a Standards Landscape*, IEEE Communications Magazine, feature topic on Telecommunications Standards, December 2012.

- C. J. Bernardos, A. de la Oliva, F. Giust, T. Melia, R. Costa, *A PMIPv6-based solution for Distributed Mobility Management*, IETF Internet draft (draft-bernardos-dmm-pmip-01), 83rd IETF, March 2012, Paris, France.

- T.-T. Nguyen, R. Costa and C. Bonnet, *Experimental Evaluation of Wireless Mobile Networks: from Methodology to a Testbed*, Springer's Wireless Communications Journal (submitted and pending acceptance)

Technical Demonstrations

- C.J. Bernardos, T. Melia, J.C. Zuniga, F. Giust, R. Costa, *Network-based Distributed Mobility Management Demo*, presented at 83rd IETF meeting, March 2012, Paris, France
- T. Melia, R. Costa *Wireless multi-access delivery for SVC-based video applications*, presented at Bell Labs Open Days, May 2012, Villarceaux (Nozay), Paris
- B. Sayadi, R. Costa *Dynamic and Distributed Mobility Management Solution Integrating Content Delivery Networks*, presented at Bell Labs Open Days, June 2013, Villarceaux (Nozay), Paris

Bibliography

[1] C. R. Murthy and B. Manoj, "*Ad Hoc Wireless Networks: Architectures and Protocols*," Prentice-Hall PTR, 2004, .ISBN 0-13-147023-X.

[2] C. Perkins, "*Ad Hoc Networking*," Addison Wesley, 2001, .ISBN 0-201-30976-9.

[3] "Tele historical glimpse - telecommunications then and now," Norsk Telemuseum (Norwegian Telecom Museum), archives (in Norwegian). [Online]. Available: http://www.telemuseum.no/joomla/images/stories/Telehistorie/telehistoriske%20glimt1.pdf

[4] "ETSI GTS GSM 03.02-v5.1.0: digital cellular telecommunications system (phase 2+) - network architecture (gsm 03.02)," 1996.

[5] 3rd Generation Partnership Project. [Online]. Available: http://www.3gpp.org/

[6] Internet Engineering Task Force. [Online]. Available: http://www.ietf.org/

[7] Institute of Electrical and Electronics Engineers. [Online]. Available: http://www.ieee.org/

[8] "3GPP TS 23.060: General packet radio service (gprs), service description; stage 2, release 7, v7.0.0," March 2006.

[9] "3GPP TS 23.002: Network architecture, v10.1.1, release 10," January 2011.

[10] "3GPP TR 23.919: Direct tunnel deployment guideline, release 7, v1.0.0," May 2007.

[11] C. Perkins, "*IP Mobility Support*," IETF RFC 2002, October 1996.

[12] WiFi Alliance. [Online]. Available: http://www.wi-fi.org/

[13] S. Al-Buraiky, "*Mobile IPv6 with Linux*," The Linux Journal, May 2008, issue no.169.

[14] "*Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2010 - 2015*," White Paper, 2011.

[15] S. Beji and N. El-Kadhi, "*An Overview of Mobile Applications Architecture and the Associated Technologies*," International Conference on Wireless and Mobile Communications (ICWMC), July 2008, .Athens, Greece.

[16] C. J. Bernardos, A. de la Oliva, F. Giust, T. Melia, and R. Costa, "*A PMIPv6-based solution for Distributed Mobility Management*," IETF DMM WG Internet-Draft, March 2012, draft-bernardos-dmm-pmip-01.

[17] Local IP Access and Selected IP Traffic Offload (LIPA-SIPTO). [Online]. Available: http://www.3gpp.org/ftp/Specs/html-info/23829.htm

[18] EU FP7 MEDIEVAL Project. [Online]. Available: http://www.ict-medieval.eu/

[19] "*Preliminary architecture design*," EU FP7 Medieval Project - D1.1, June 2011.

[20] J. C. Zuniga, C. J. Bernardos, T. Melia, A. de la Oliva, R. Costa, and A. Reznik, "*Distributed Mobility Management: a Standards Landscape*," IEEE Communications Magazine, feature topic on Telecommunications Standards, March 2013.

[21] H. Chan, D. Liu, P. Seite, H. Yokota, and J. Korhonen, "*Requirements for Distributed Mobility Management*," IETF DMM WG Internet-Draft, February 2014, draft-ietf-dmm-requirements-14.

[22] P. McCann, "*Authentication and Mobility Management in a Flat Architecture*," IETF DMM WG Internet-Draft, March 2012, draft-mccann-dmm-flatarch-00.

[23] F. Giust, A. de la Oliva, and C. J. Bernardos, "*Flat Access and Mobility Architecture: an IPv6 Distributed Client Mobility Management Solution*," IEEE Mobiworld workshop, in conjunction with IEEE INFOCOM, April 2011, .Shanghai, China.

[24] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, and M. Carney, "*Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*," IETF RFC 3315, July 2003.

[25] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil, "*Proxy Mobile IPv6*," IETF RFC 5213, August 2008.

[26] M. Crawford and B. Haberman, "*Ipv6 Node Information Queries*," IETF RFC 4620, August 2006.

[27] "*Final architecture design*," EU FP7 Medieval Project - Deliverable D1.3, December 2012.

[28] "*Final plan for the use and dissemination of foreground*," EU FP7 Medieval Project - Deliverable D1.4, July 2013.

[29] "*Light IP Mobility architecture for Video Services: initial architecture*," EU FP7 Medieval Project - Deliverable D4.1, June 2011.

[30] "*Final Specification for mobility components and interfaces*," EU FP7 Medieval Project - Deliverable D4.3, June 2012.

[31] "*Final Operational Mobility Architecture*," EU FP7 Medieval Project - Deliverable D4.4, December 2012.

[32] "*Second Periodic Testing Report*," EU FP7 Medieval Project - Deliverable D6.4, July 2013.

[33] S. Figueiredo, C. Guimaraes, R. Aguiar, T.-T. Nguyen, L. Yadin, N. Carapeto, and P. Parada, "*Broadcasting User Content over Novel Mobile Networks*," International Conference on Communications (ICC), June 2013, .Budapest, Hungary.

[34] S. Jeon, S. Figueiredo, and R. Aguiar, "*On the Impacts of Distributed and Dynamic Mobility Management Strategy - A Simulation Study*," IFIP Wireless Days, November 2013, .Valencia, Spain.

[35] "*IEEE Standard for Local and metropolitan area networks - Part 21: Media Independent Handover*," IEEE Standard 802.21, 2008.

[36] D. Corujo, C. Guimaraes, B. Santos, and R. Aguiar, "*Using an Open-Source IEEE 802.21 Implementation for Network Based Localized Mobility Management*," IEEE Communications Magazine, Special Issue on Communications Middleware for Mobile Devices and Applications, September 2011.

[37] C. Perkins, D. Johnson, and J. Arkko, "*Mobility Support in IPv6*," IETF RFC 6275, June 2004.

[38] B. Orlandi and F. Scahill, "*Wi-Fi Roaming - Building on ANDSF and Hotspot 2.0*," White Paper by Alcatel-Lucent Bell Labs and British Telecommunications, 2012.

[39] F. Buiati, L. Villalba, D. Corujo, and R. Aguiar, "*IEEE 802.21 Information Services deployment for heterogeneous mobile environments*," IET Communications, December 2011, volume 5, issue 15, ISSN 1751-8628.

[40] R. Kuntz, J. Montavont, and T. Noel, "*Multihoming in IPv6 mobile networks: progress, challenges, and solutions*," IEEE Communications Magazine, January 2013, first issue, Vol.51.

[41] C. J. Bernardos, "*Proxy Mobile IPv6 Extensions to Support Flow Mobility*," IETF NETEXT WG Internet Draft, March 2011, draft-ietf-netext-pmipv6-flowmob.

[42] T. Melia, S. Gundavelli, C. J. Bernardos, and A. de la Oliva, "*Logical Interface Support for multi-mode IP Hosts*," IETF NETEXT WG Internet-Draft, April 2013, draft-ietf-netext-logical-interface-support-07.

[43] E. Conchon, J. Garcia, T. Perennou, and M. Diaz, "*Improved IP-Level Emulation for Mobile and Wireless Systems*," IEEE WCNC, March 2007, .Hong Kong, China.

[44] S. Wang, C. Chou, Y. Tseng, M. Hsu, Y. Cheng, W. Liu, and T. Ho, "*NCTUns 4.0: An Integrated Simulation Platform for Vehicular Traffic, Communication, and Network Researches*," IEEE WiVeC, October 2007, .Baltimore, USA.

[45] A. Law, "*Simulation Modelling and Analysis*," McGraw-Hill, 2007, .ISBN 0-07-298843-6, 4th edition.

[46] K. Pawlikowski, H.-D. J. Jeong, and J.-S. R. Lee, "*On Credibility of Simulation Studies Of Telecommunication Networks*," IEEE Communications Magazine, January 2002, first issue, Vol.40.

[47] GINI - A Toolkit for Constructing User-Level Micro Internets. [Online]. Available: http://cgi.cs.mcgill.ca/~anrl/gini/index.html

[48] VNX - Virtual Networks over Linux. [Online]. Available: http://web.dit.upm.es/vnxwiki/index.php/Main_Page

[49] Mininet. [Online]. Available: http://mininet.org/

[50] OpenFlow. [Online]. Available: https://www.opennetworking.org/sdn-resources/onf-specifications/openflow

[51] Netkit. [Online]. Available: http://wiki.netkit.org/

[52] Virtual Distributed Ethernet (VDE). [Online]. Available: http://vde.sourceforge.net/

[53] The Linux Programmer Manual Pages. [Online]. Available: http://man.he.net/

[54] MGEN, The Multi-Generator. [Online]. Available: http://pf.itd.nrl.navy.mil/mgen/mgen.html

[55] T.-T. Nguyen, R. Costa, and C. Bonnet, "*Experimental Evaluation of Wireless Mobile Networks: from Methodology to a Testbed*," Springer Wireless Communications Journal, (submitted and pending acceptance).

[56] D. Camara, H. Tazaki, E. Mancini, M. Lacage, T. Turletti, and W. Dabbous, "*DCE: Test the real code of your protocols and applications over simulated networks*," to appear in IEEE Communication Magazine, Network Testing Series, 2014.

[57] Y. Kim, A. Quereilhac, M. Larabi, J. Tribino, T. Parmentelat, T. Turletti, and W. Dabbous, "*Enabling Iterative Development and Reproducible Evaluation of Network Protocols*," to appear in Computer Networks, 2014.

[58] ALTO Status Pages. [Online]. Available: http://tools.ietf.org/wg/alto/

[59] D. Munaretto, T. Melia, S. Randriamasy, and M. Zorzi, "*Online path selection for video delivery over cellular networks*," IEEE QoEMC, in conjuntion with IEEE Globecom, December 2012, .Anaheim, USA.