

MEDIEVAL

Deliverable D4.3

Final Specification for mobility components & interfaces

Editor:	Sérgio Figueiredo, IT
Deliverable nature:	Public
Suggested readers:	MEDIEVAL Board
Due date:	June 30 th , 2012
Delivery date:	June 30 th , 2012
Version:	2.0
Total number of pages:	91
Reviewed by:	Davide Chiarotto (CFR)
Keywords:	MEDIEVAL Specification, Distributed Mobility Management, IP multicast, Connection and Flow management, Mobile Video

Abstract

In an Era where Internet video is dominant, operators demand novel solutions for embracing mobile video in an efficient way. Being IP mobility a key field for achieving this goal seamlessly, and pushed by the limitations identified in centralized mobility models, the present deliverable presents MEDIEVAL's proposal for a flat IP mobility architecture, which comprises mechanisms for dealing with IP flow, network and multicast mobility following the novel distributed mobility management model.

As such, this document has the main goal of providing the updated specification of the referred architecture, including the detailed description of related modules, interfaces and primitives.

List of authors

Company	Authors
IT	Sérgio Figueiredo, Daniel Corujo, Seil Jeon
TI	Loris Marchetti, Elena Demaria, Marco Marchisio
ALBLF	Telemaco Melia, Rui Costa
UC3M	Fabio Giust, Carlos Jesús Bernardos Cano, Antonio de la Oliva
EURECOM	Nguyen Tien-Thinh, Christian Bonnet
PTIN	Nuno Carapeto, Carlos Parada

History

Modified by	Date	Version	Comments
IT	15/01/2012	1.0	Interim version
IT	30/06/2012	2.0	Final version

Executive Summary

The purpose of this deliverable is to provide a description of the final specification of the Mobility Sub-system within the MEDIEVAL Project.

As stated in the previous architectural description [9], the MEDIEVAL mobility architecture leverages on two main concepts: the capability to activate mobility only when required (i.e. when the video session cannot support a change in IP address), and the distribution of the mobility anchors towards the edge, against the traditional centralized placement at the network's core. For such, mobility management capabilities are replicated at the Access Router (AR) level. These two guidelines impose several changes in the way the architecture is designed. One of those changes is the inclusion of IP flow awareness in the mobility architecture, where MEDIEVAL has given a significant contribution, for example by emphasizing the possibility for better user plane congestion management. Additionally, the integration of Content Delivery Networks (CDNs) in the mobility architecture copes with the Distributed Mobility Management (DMM) concept, by enforcing the advantages in placing the anchors near the network edge, besides allowing the caching of ever-increasing user generated content (UGC). Moreover, the presented Mobility architecture follows a cross-layer design, with part of its modules having awareness to external layers, enabling such functionalities as the aforementioned selective flow mobility or content-aware handover decision (i.e. sensitive to content location, availability, etc).

On the other hand, IP multicast is core for operators to achieve the desired network efficiency in video transport, namely for real-time video services. This deliverable contributes to this field with considerable innovative input, and updates the work initiated in [10] both in legacy PMIPv6 mobility protocol and on the synergy of this mechanism with DMM -based protocols.

Before proceeding with the implementation of the architecture, an evaluation for selecting the mobility framework was required. The exhaustive metrics' comparison has lead to the choice of EURECOM's Open Air Interface, against the other candidate framework – IT's OPMIP. Besides, due to the necessity to be able to evaluate different components separately before the final demonstration in EURECOM's testbed, an extensive search and deep evaluation of realistic simulation tools was done. The result was the selection of NETKIT, which has already allowed the referred PMIPv6 frameworks' comparison, as well as deploying ODTONE and OpenCDN.

The deliverable presents both the updated specification and implementation status of each of the modules belonging to the Mobility sub-system, going beyond the theoretical plane. As expected, some modules are in a more advanced state than others, which enabled the start of the integration work between different modules and sub-systems (out of the scope, but described in [12]).

Table of Contents

List of authors.....	2
History	2
Executive Summary.....	3
Table of Contents	4
List of Figures.....	7
Abbreviations	8
1 Introduction	12
2 Key Contributions	13
3 Scientific work	15
3.1 Considerations on Mobility Architecture.....	15
3.2 Key Mobility Architecture Developments	16
3.2.1 Introduction to current practices for deploying mobility architectures.....	16
3.2.2 Network-based DMM.....	20
3.2.3 Multicast Mobility	24
3.2.4 Flow Mobility	27
3.2.5 SVC and Mobile Networks.....	30
3.2.6 DMM and Content Delivery Networks	35
3.3 Dissemination and standardization plan and results	36
3.3.1 IETF WG DMM	36
3.3.2 IETF WG NETEXT.....	37
3.3.3 IETF WG MULTIMOB.....	38
3.3.4 IETF WG MIF	38
3.3.5 IEEE 802.21.....	38
3.3.6 3GPP.....	39
4 Mobility Architecture.....	41
4.1 Modules.....	42
4.1.1 Flow Manager.....	42
4.1.2 Connection Manager.....	44
4.1.3 Unicast Mobility Engine.....	46
4.1.4 Multicast Mobility Engine	49
4.1.5 NEMO mobility engine	50
4.1.6 Media Independent Information Server.....	51

4.2	Interfaces.....	52
4.2.1	CM_MIIS_If.....	52
4.2.2	FM_MIIS_If	52
4.2.3	FM_FM_If	53
4.2.4	CM_FM_If.....	53
4.2.5	CM_UME_If.....	53
4.2.6	FM_UME_If.....	54
4.2.7	FM_MUME_If	54
4.2.8	UME_UME_If.....	55
4.2.9	MUME_MR_If.....	55
4.2.10	MUME_MUME_If.....	55
5	Status of the implementation work.....	57
5.1	Selection of the PMIPv6-compliant open-source software.....	57
5.2	Modules Status.....	58
5.2.1	CM.....	58
5.2.2	FM.....	58
5.2.3	UME	59
5.2.4	MUME.....	60
5.2.5	MIIS.....	61
6	Status of the simulation work.....	62
6.1	ALBLF Network Simulation/Emulation framework	62
6.1.1	Motivation.....	62
6.1.2	Requirements	62
6.1.3	Test bed Design Solution.....	62
6.1.4	NETKIT	62
6.1.5	SWEEN (Simulation of Wireless Environment Extensions for Netkit)	63
6.1.6	Current Status	64
6.1.7	Future Work.....	65
7	Conclusion and next steps.....	66
	Acknowledgements and Disclaimer	67
	References	68
Annex A	Complete Updated Specification	72
A.1	Internal Interfaces	72
A.1.1	CM_MIIS_If.....	72
A.1.2	FM_MIIS_If	72

A.1.3	FM_FM_If	72
A.1.4	CM_FM_If.....	72
A.1.5	CM_UME_If.....	73
A.1.6	FM_UME_If.....	74
A.1.7	FM_MUME_If	76
A.1.8	UME_UME_If.....	84
A.1.9	MUME_MR_If.....	84
A.1.10	MUME_MUME_If	85
Annex B	Demonstration Posters	90
B.1	Alcatel-Lucent Bell Labs Open Days Poster	90
B.2	MEDIEVAL demo poster used during the 83rd IETF meeting	91

List of Figures

Figure 1: DSMIP tunnelling approaches	18
Figure 2: Initial registration in partially distributed mobility management.....	22
Figure 3: Protocol operations when CMD is in relay mode	22
Figure 4: Data forwarding after handover	23
Figure 5: Protocol operations when CMD is in locator mode	23
Figure 6: Protocol operations when CMD is in proxy mode.....	24
Figure 7: IP multicast scheme use-cases for DMM.....	25
Figure 8: Problems description when existing IP multicast approach is applied on DMM	26
Figure 9: Multicast operation and function in presented CM-DMM.....	27
Figure 10: Experimental setup for SVC video enhanced delivery	31
Figure 11: Marks distribution for the video samples.....	32
Figure 12: SVC multicast mobility scenario	34
Figure 13: CDN and DMM architectural integration	35
Figure 14: Functional architecture of the Mobility subsystem.....	41
Figure 15: Mobility modules in the MT	42
Figure 16: Mobility modules in the MAR	42
Figure 17: Mobility modules in the mMAR.....	42
Figure 18: FM internal architecture and interfaces.....	44
Figure 19: CM Architecture	45
Figure 20: MAR's UME state machine	48
Figure 21: MUME internal structure and interaction with relevant modules.....	50
Figure 22: Multicast Context Transfer messages	61
Figure 23: Architecture of the SWEEN/NETKIT test bed to evaluate scalability	64

Abbreviations

A-MAR	Anchoring Mobility Access Router
ANDSF	Access Network Discovery and Selection Function
API	Application Programming Interface
AR	Access Router
BC	Binding Cache
BCE	Binding Cache Entry
BA	Binding Acknowledgment
BID	Binding Identification
BU	Binding Update
BUL	Binding Update List
BULE	Binding Update List Entry
CCS	Channel Control Server
CDN	Content Delivery Networks
CEF	Channel Enforcement Function
CM	Connection Manager
CMD	Central Mobility Database
CN	Core Network
CoA	Care-of Address
CDN	Content Delivery Network
CM-DMM	Channel-Manageable Distributed Mobility Management
DHCP	Dynamic Host Configuration Protocol
DM	Decision Manager
DMM	Distributed Mobility Management
DNAv4	Detecting Network Attachment (version 4)
DNS	Domain Name System
DSIS	Double Stimulus Impairment Scale
DSMIPv6	Dual Stack Mobile IPv6
DPI	Deep Packet Inspection
ESP	Encapsulating Security Payload

FM	Flow Manager
GTP	GPRS Tunnelling Protocol
GUI	Graphical User Interface
HD	High Definition
HI	Handover Indication
HNP	Home Network Prefix
HAck	Handover Acknowledgement
HO	Handover
HoA	Home Address
IE	Information Element
IKEv2	Internet Key Exchange Protocol (version 2)
IPSec	Internet Protocol Security
IPCP	Internet Protocol Control Protocol
LIF	Logical Interface
LIMONET	Local IP Mobility and SIPTO at the Local Network
LIPA	Local IP Access and Selected IP Traffic Offload
LMA	Local Mobility Anchor
LMD	Localized Mobility Domain
LTE	Long Term Evolution
MAG	Mobile Access Gateway
MAR	Mobility Access Router
ME	Mobility Engine
MEDIEVAL	MultimEDIA transport for mobiIE Video AppLications
MIH	Media Independent Handover
MIHF	Media Independent Handover Function
MIIS	Media Independent Information Service
MLD	Multicast Listener Discovery
MIPv6	Mobile IPv6
mMAR	Mobile Mobility Access Router
MMD	Multicast Mobility Database
MMIS	Multicast Mobility Information Server
MN	Mobile Node

MOS	Mean Opinion Score
MR	Multicast Router
MRIB	Multicast Routing Information Base
MST	Multi-Session Transmission
MT	Mobile Terminal
MUME	MUlticast Mobility Engine
NALU	Network Abstraction Layer Units
NAT	Network Address Translation
NEMO	Network Mobility
PBA	Proxy Binding Acknowledge
PBU	Proxy Binding Update
PCRF	Policy and Charging Rules Function
PIM-SM	Protocol Independent Multicast - Sparse-Mode
PMIPv6	Proxy Mobile IPv6
PPP	Point-to-Point Protocol
PoA	Point of Attachment
PoS	Point of Service
QoE	Quality of Experience
PBS	Personal Broadcasting Service
RADIUS	Remote Authentication Dial-In User Service
RDF	Resource Description Framework
RP	Rendezvous Point
RRDM	Request Router and Distribution Manager
RTP	Real Time Protocol
RTSP	Real Time Streaming Protocol
S-MAR	Serving Mobility Access Router
SIPTO	Selected IP Traffic Offload
SLA	Service-Level Agreement
SMIL	Synchronized Multimedia Integration Language
SPARQL	SPARQL Protocol and RDF Query Language
SPI	Security Parameter Index

SVC	Scalable Video Coding
TLV	Type Length Value
UE	User Equipment
UGC	User Generated Content
UME	Unicast Mobility Engine
UPCON	User Plane Congestion
X-GW	Serving- or Packet Data Network-Gateway
XLO	Cross-Layer Optimizer
WLAN	Wireless Local Area Network

1 Introduction

This deliverable focuses on the specification of the modules and internal interfaces of the MEDIEVAL mobility subsystem. The final specification of the external interfaces between the mobility subsystem and the other MEDIEVAL subsystems (Video Services Control, Wireless Access, Transport Optimization) will be reported in D1.3 [6].

In the Deliverables D4.1 [9] and D4.2 [10] the MEDIEVAL mobility architecture has been presented, essentially based on the “Distributed Mobility Management” (DMM) concept. The anchors are distributed at the edge of mobile networks and the mobility resources are activated only for applications/flows which need them.

The proposed DMM model adopts an hybrid approach, where network-based mobility management solutions (i.e., PMIPv6-alike) are used whenever possible, and client-based solutions are used otherwise (e.g., between different domains), and due to the video-centric nature of the project, multicast traffic delivery and content distribution aspects are fully supported and integrated in the mobility management solution.

In order to integrate both network-based and host-based approaches, so that a mobile node can simultaneously have sessions managed by a network-based (“PMIPv6 alike”) approach and a host-based (“DSMIPv6 alike”) approach, a novel architectural element called Mobile Access Router (MAR) has been introduced. The MAR is a network entity implementing all the functionalities of its counterparts in the standard mobility protocols (MIPv6 and PMIPv6), so it is able to play the role of plain access router, home agent, local mobility anchor and mobile access gateway on a per address basis.

Beyond mobility functionalities, realized within the Unicast/Multicast Mobility Engine module ((M)UME), further key functional modules of the mobility subsystem are the Connection Manager (CM – running in the MN) and the Flow Manager (FM – running in the MAR). These two modules are essential in exploiting the flow mobility extensions and all the mobility related cross-layer features to integrate the mobility subsystem with the other MEDIEVAL functional subsystems.

The main goal of this deliverable, besides the illustration of the advances in the scientific work, is the specification of the introduced modules. First, a formal description and state machines are presented to illustrate the operation provided by each module; additionally, the specification of the interfaces between these modules, in terms of primitives and parameters, is shown. This is the basis for the on-going implementation and simulation work which is expected to allow the validation of the proposed architecture. The ultimate goal is perform the experimental evaluation of the advantages of a DMM based approach when compared to traditional core based anchoring models (e.g., in terms of scalability and reliability, lower signalling overhead and shorter handover latencies, better control on the mobility granularity allowed). This will be the main subject of the final Deliverable (D4.4 [11]) which is expected to present the consolidated mobility architecture assessment.

Finally, the structure of this document is as follows. Section 2 summarizes the key contributions of the work documented in this Deliverable for both the scientific and specification work. Section 3 emphasizes the performed scientific work. The main areas include: mobility management with special focus on network-based DMM, flow mobility, multicast mobility, SVC support in mobile networks. Finally, the resulting publications and standardization work are also reported. Section 4 focuses on specification of both modules and interfaces. Further details about specification of the interfaces with a complete description of the format/structure of the parameters are reported in Appendix A. This specification work is the basis for the already on-going implementation work. Section 5 summarizes the current status of the implementation work including main technological choices. All the different modules building up the mobility subsystem are considered. Section 6 describes the status of the emulation/simulation work. This includes the ALBLF network simulation/emulation framework based on an open-source tool named Netkit, integrated with proper extensions for the emulation of the wireless environment and node movement. Finally, section 7 concludes the deliverable and points to upcoming steps.

2 Key Contributions

The list below represents the contributions offered by the MEDIEVAL project related to the topic dealt by this deliverable, highlighting the scientific and standardization work deployed during the second year of the project in the mobility arena, and the impact that these activities represent for the growth of the project.

- Since D4.1 and D4.2, the MEDIEVAL mobility subsystem has been further analyzed in order to improve the design, provide enhancements and integrations. The mobility components have been fully specified taking into considerations minor issues and optimizations that arose after the release of previous MEDIEVAL documents. We argue this is a vital part in the growth of the project, as many external inputs have been welcomed for the sake of improving the architecture in a dynamic way, addressing the criticisms and comments received in international events, workshops and other meetings. This illustrates the significant impact that MEDIEVAL's ideas and innovations have achieved in the research community.
- The aforementioned specification work was driven also by the experience gained during the implementation tasks: many MEDIEVAL modules have been developed and are now available either in intermediate or final status, ready for the integration stage (which is out of scope of this deliverable). The goal is to produce high quality prototypes to be showed in public demonstrations at international events, useful to tune the design and optimize the overall performance.
- A key performed activity has been motivated by the will to evaluate the distributed mobility management design against the on-going work in 3GPP and consequently the industry view. This work (still on-going) has been developed by involving relevant 3GPP experts between the different MEDIEVAL partners. Within this discussion group, different topics involving activities covered by different WPs have been considered, including the DMM concept, the impact of user plane congestion (UPCON) management and the integration of the mobility architecture with the CDN technology. This work/cooperation seems very fruitful and the plan is to prosecute this strict cooperation with 3GPP experts who may also guide MEDIEVAL to provide proper contributions to the TSG-SA – the Technical Specification Group for Service and Architecture – both in SA1 (more involved in Services and use cases) and SA2 (more involved in the Architecture and therefore technical solutions). A first result has been the UPCON submission [16]. Further advances have brought to the editing of a specific Technical Report, 3GPP TR 22.805, whose current version (2012-05) is V0.3.1.
- Concerning IETF activities, a new working group has been recently created with its focus on DMM, which will be a perfect placeholder for MEDIEVAL work. MEDIEVAL participated in its first official meeting (within 83rd IETF Meeting, which was held in Paris) presenting a comprehensive draft encompassing two network-based DMM approaches and several solutions for each approach. The presentation was strengthened by a public demonstration that showed a network prototype with the DMM features: real devices were deployed to run a live experiment using the implementation of one of the DMM solutions mentioned in the draft. This aspect increases even more the possibilities of continuing and improving our real impact on the IETF. Moreover, a first draft regarding the different use cases for IP multicast support in DMM networks [25] was submitted and discussed in the same meeting. Current plans go around the following lines: continue contributing to the DMM problem statement and scenario definition documents, extend and improve the network-based and client-based DMM solutions, as well as looking at the integration with mobile networks.
- MEDIEVAL is strongly involved in IETF work, being currently co-authoring 5 WG documents. These include the two principal documents for a flow mobility solution in PMIPv6 [19][20], within NETEXT WG; the optimization proposals for multicast support in PMIPv6, one for solving the tunnel convergence problem [25] and one for enabling fast handover [26].
- A survey on dual-stack deployment options considered by IETF has been accepted for publication in IEEE Wireless Communications Magazine [28], both considering client-based and network-based mobility solutions. Regarding network-based mobility support, an open source implementation of PMIPv6 (IT's OPMIP) was successfully used while evaluating ODTONE's (IT's IEEE 802.21 open

source implementation) features [41]. This framework will be important for achieving results prior to the final demonstrator.

- By identifying the problems associated with the combination of different mobility schemes [31], it was possible to emphasize the need for DMM solutions. In order to determine the different possibilities in designing a DMM scheme, a study on different schemes for handling the mobility control plane was accomplished [27]. Similarly important, another study tackled IP flow mobility offloading techniques currently being considered by the IETF [32] and has been published in IEEE Communications Magazine. The first method is based on extending existing client-based IP mobility solutions to allow flow mobility where the user terminal is fully involved in the mobility process, and the second one is based on extending current network-based IP mobility solutions where the user terminal is not aware of the mobility.
- Regarding multicast mobility, the work in [40], based on the base multicast mobility solution defined by IETF for PMIPv6 (i.e. deploying MLDv2 Proxies in MAGs), proposes and evaluates a solution for decreasing the handover time for multicast listeners, by applying multicast context transfer, in a predictive or reactive way. Additionally, some progress was achieved on the study of multicast mobility support in DMM. Firstly, use cases for multicast listener support in DMM were identified [46], which resulted in the proposal for different schemes: proactive or reactive, partially distributed or fully distributed. Moreover, the study of the problems involved with remote multicast subscription in DMM were assessed, which derived into a proposal for a IP multicast channel manageable solution [47].
- In an attempt to study more pragmatic scenarios, two publications were submitted. In [33], the experiments demonstrated that implementing SVC awareness at the mobility anchors can greatly enhance the video delivery process, increasing the QoE perceived at the users, while reducing the cost per bit carried over the wireless network. It was also possible to show that the combination of cellular and WiFi coverage enhance the video delivery close to locally played video streams. In [39], an analysis on multicast layered mobility support is provided, showcasing other intelligent features that could be placed in the mobility anchors and decision nodes.
- An important result concerning next implementation work has been *i)* the selection of the PMIPv6-compliant open-source software, and *ii)* the development of the network-based DMM implementation. The selection was between two existing implementations developed internally by two MEDIEVAL partners (EURECOM and IT). On the development side, the network prototype mentioned before for the IETF public demonstration represents the first stage of the Unicast Mobility Engine, i.e., the main component of the network-based DMM architecture.
- As a basic enabler for next implementation/integration phases, an important achievement has been the set-up, at ALBLF premises, of a “Network Simulation/Emulation framework”. Its main purpose was to test and use the software components developed by ALBLF, as well as its integration with software from all other MEDIEVAL partners.

3 Scientific work

The purpose of this section is to present the scientific work related to the topics covered in D4.1 [9] and D4.2 [10]. This includes considerations and discussions concerning the mobility architecture and a summary of the main contributions in terms of dissemination, including both publications and standardization effort.

3.1 Considerations on Mobility Architecture

D1.1 [4] defines the overall MEDIEVAL architecture while D4.1 [5] and D4.2 [10] detail the mobility support for both unicast and multicast communications. The novel mobility architecture has been designed taking into account latest developments in the research community, standardization fora and industry requirements. After the first year of the project, the architecture was reviewed taking into account feedback received by the experts in the above mentioned domains.

First, we evaluated the distributed mobility management design against the ongoing work in 3GPP. The feedback received by 3GPP SA2 (System Architecture Stage 2) experts is twofold. On the one hand they are convinced of the technical solution to address scalability issues and they are supportive of the IETF work on DMM (see section 3.3.1 for details). On the other hand they expressed some concerns on the impact that such studies will have on future 3GPP releases. While this is difficult to predict, the distributed mobility management is a solution that will depend on the deployment model selected by the mobile operator. The availability of small size cells, combined with the WiFi access and the ever increasing number of mobile devices, will impose new requirements on the way IP traffic is managed. In this sense two traffic offloading methods currently being studied in 3GPP are the following: “Selected IP Traffic Offload” (SIPTO) for macro networks [11] [13] and “Local IP Mobility and SIPTO at the Local Network” (LIMONET) [14] [15]¹.

The selection of the most appropriate method will depend on the geographical coverage the mobile operators will address: the larger the considered area the higher the needs for scalability and mobility anchor distributions. Summarizing, the DMM work is seen as a key study to address scalability and the addressed scenarios will heavily depend on the decisions that mobile operators will be doing. To this end, MEDIEVAL will engage through the different contacts discussions on this topic and will communicate the obtained results in the research community.

The second point we addressed is the IP flow awareness of the mobility infrastructure. This requirement has generated a lot of interest from both a technical and business point of view. 3GPP recently agreed on studying the impact of user plane congestion (UPCON) management. The study is still at the service requirement stage (SA1) and addresses the needs to understand the type of traffic generated by each mobile user and the ability to move specific applications to specific wireless accesses. DOCOMO has been pushing the subject in SA1 and ALU is supportive of this initiative [16]. Some of the proposed ideas were discussed in MEDIEVAL and the feedback from the industry will be used to fine-tune our approach. It should be noted that MEDIEVAL already pioneered some of the logic for congestion management, being the demo presented in the Future Networks and Mobile Summit 2011 at Warsaw and submitted for publication at MONAMI 2012 (see section 3.2.5 for more details) a proof of concept tailored for handling congestion management of video applications. The third point we addressed concerns the integration of the mobility architecture with the CDN technology studied in WP5 which resulted in the submission of an architectural proposal for publication at MobiArch (co-located workshop with ACM’s MOBICOM 2012, for further information refer to Section 3.2.6). Currently mobile operators are investigating from both a technical and business point of view how to achieve efficient integration. The use cases vary from simple storage capacity offer to increase mobile users’ QoE by implementing optimizations specific to the mobile environment. These optimizations are only possible if special purpose interfaces are specified between the CDN management system and the underlying mobility architecture. MEDIEVAL is currently focusing on this interface and is planning to contribute to the service requirements in SA1 for next release. The target is to help in the definition of the service requirements such that SA2 will be then able to define the necessary interfaces and mechanisms to

¹ While SIPTO does not support mobility, but only traffic offloading, with LIMONET work 3GPP plans to provide this missing support

enable such use case. It should be noted that the CDN integration and more broadly content caching is one of the next “technology storm”. As stated in D1.2 [5] user generated content is picking up rapidly and mobile operators could leverage this new trend by offering new services. MEDIEVAL will study caching techniques for video traffic but will consider, at least from a business perspective, the caching opportunity from a more general perspective.

In regards of multicast (network-based) mobility support, MEDIEVAL has also obtained feedback at the IETF (as mentioned in section 3.3.3) about different proposals. MEDIEVAL contributions have been adopted as WG documents in two of the four main topics currently being discussed in the WG: handover optimizations and the tunnel convergence problem. Additionally, a draft on the use cases identified for the support of IP multicast in network-based DMM networks was submitted.

The following sections support the above considerations by describing MEDIEVAL’s recently developed work – serving as an update to the previous deliverables – as well as the related contributions to the IETF, IEEE and 3GPP standardization groups.

3.2 Key Mobility Architecture Developments

Within MEDIEVAL project’s previous work innovative mobility architecture was designed, combining state-of-art standards and pushing further the development of ideas that follow the latest paradigms on mobility management, as those introduced with the Distributed Mobility Management. The detailed description of the architecture, including operations, entities and messages is provided in the MEDIEVAL deliverables D1.1, D4.1 and D4.2, respectively [4], [9] and [10], related to the first 12 months of research within the project.

For the consolidation and finalization of the previously introduced architecture, additional effort was put to explore improvements and extensions to the adopted scheme. Hence, the work presented in section 3.2.2 describes different scenarios and variations for the deployment of network based distributed mobility management solutions, while the analysis of state-of-art solutions for dual stack management shown in next section. Moreover, section 3.2.3 presents the latest achievements related to multicast service support in DMM scenarios, while section 3.2.4 explores motivations and scenarios to support the deployment of flow mobility. Finally, in Section 3.2.5 we present a real case application of the key features of MEDIEVAL project implemented in a test-bed prototype, as well as the description of an envisioned scenario under MEDIEVAL multicast functionalities, both of the cases taking advantage of the state of the art codec H.264 SVC.

3.2.1 Introduction to current practices for deploying mobility architectures

The trend of increasing users’ mobility and the incipient perspectives of IPv6 transition leads to the need for developing IP mobility solutions suited for mixed IPv4/IPv6 mobile environments. A possible approach consists of deploying both the mobility management protocols defined for IPv4 and IPv6 in a dual stack node. Running IPv4 and IPv6 mobility protocols in parallel introduces a number of issues:

- two sets of signalling messages are required to be sent whenever they move;
- network administrators have to run and maintain two sets of mobility management systems;
- the connectivity across different networks may not be guaranteed due to the dependency on the IPv4/IPv6 capabilities of the networks the mobile node is visiting; i.e., a node attempting to connect via an IPv4-only network would not be able to maintain the connectivity of its IPv6 applications and vice versa.

Therefore, the approach recommended by the IETF is to have only one mobility management protocol (i.e., extending one of the existing mechanisms) that can support the mobility for a dual stack node and, consequently, enables

- IPv4 and IPv6 tunnels management;

- the mobility of IPv4 and IPv6 home addresses signalling;
- the IPv4 and IPv6 access networks visiting.

Considering that it is foreseen that IPv6 will be the only version at the end (i.e. we finally get to a situation where there is only IPv6), or at least it will be the dominant one, it seems more reasonable from a deployment viewpoint to extend IPv6 mobility protocols to handle dual stack nodes. This approach allows for a long lasting mobility solution, avoiding the need for changing the mobility solution in the future IPv6 Internet. The work in [28] surveys the standards recently defined by the IETF to extend the IPv6 mobility management mechanisms, considering both the client-based and the network-based solutions, to support the mobility of dual stack nodes roaming among IPv4-and-IPv6, IPv6-only and IPv4-only networks. What follows are the main considerations extracted from the article.

IPv4 support for MIPv6

Mobile IPv6 standards (i.e., MIPv6 and NEMO B.S.) have been extended in RFC 5555 [29] to support the operation of dual stack mobile hosts and routers, by enabling them to:

- roam over both IPv4 and IPv6 visited networks;
- register IPv4 home addresses and mobile network prefixes;
- transport IPv4 and IPv6 traffic over the tunnel between the mobile node/router and its home agent.

Basically, there are three different extensions to MIPv6 that are required in order to support dual stack nodes:

- signalling extensions to allow carrying IPv4 addresses (and prefixes) and detecting NATs;
- new types of tunnels, allowing for the transport of IPv4 and IPv6 data packets, even traversing NATs;
- support for attachment to IPv4-only networks (this involves IPv4 care-of address support and NAT detection).

It is assumed that a home agent serving a dual stack mobile host/router has also an IPv4/IPv6 dual stack. Regarding MIPv6 signalling, there are just a couple of significant changes. On the one hand, a new option (called IPv4 Home Address option) is defined, which allows a mobile node not only to use an IPv6 Home Address, but also an IPv4 one. The RFC 5555 also defines an option to carry an IPv4 care-of address, as there are scenarios in which the mobile node will not be able to configure a global IPv6 address as care-of address. A new option is also defined to allow the home agent to notify the mobile node that there is a NAT in the path, including a flag that indicates to the mobile node if UDP tunnelling should be used, and optionally including a suggested NAT binding refresh time (in case the home agent knows the NAT timeout value, e.g., when the NAT belongs to the same administrative domain that the home agent).

A dual stack mobile node can get attached to an IPv6/IPv4 dual stack, an IPv6-only or an IPv4-only access network. In case the mobile node is able to configure an IPv6 care-of address, this should be used as source address for the MIPv6 signalling (and also as the local end of the bi-directional tunnel with the home agent). In case the access network only provides IPv4 connectivity, the mobile node needs to detect if it is behind a NAT or not. In order to do so, the initial IPv6 Binding Update (source address set to the mobile's home address, destination address set to the home agent's IPv6 address) is encapsulated in UDP, which is transported using IPv4 (with source address the mobile node's IPv4 care-of address, and destination address the home agent's IPv4 address). The home agent compares the IPv4 address of the source address field in the IPv4 header with the address included in the IPv4 care-of address option. In case the two addresses do not match, that means that there is a NAT in the path, and the home agent includes a NAT detection option in the Binding Acknowledgment.

Both IPv4 and IPv6 data forwarding between the mobile node and its home agent are supported. IPv6 tunnelling is generally preferred in case the mobile node has a valid IPv6 care-of address. If the mobile is behind a NAT device, UDP tunnelling is used. If a public IPv4 care-of address is available (i.e., no NAT detected), then UDP tunnelling is generally not required, although there are a few exceptions, such as the local domain not allowing IP-in-IP traffic, where UDP might be used even when the mobile node is not

behind a NAT. Fig. 4 shows the different tunnelling approaches defined by RFC 5555 for communications between a dual stack mobile node and IPv6 and IPv4 correspondent nodes (only packets sent by the MN are shown, the other direction will follow the symmetric path, as the tunnels are bi-directional).

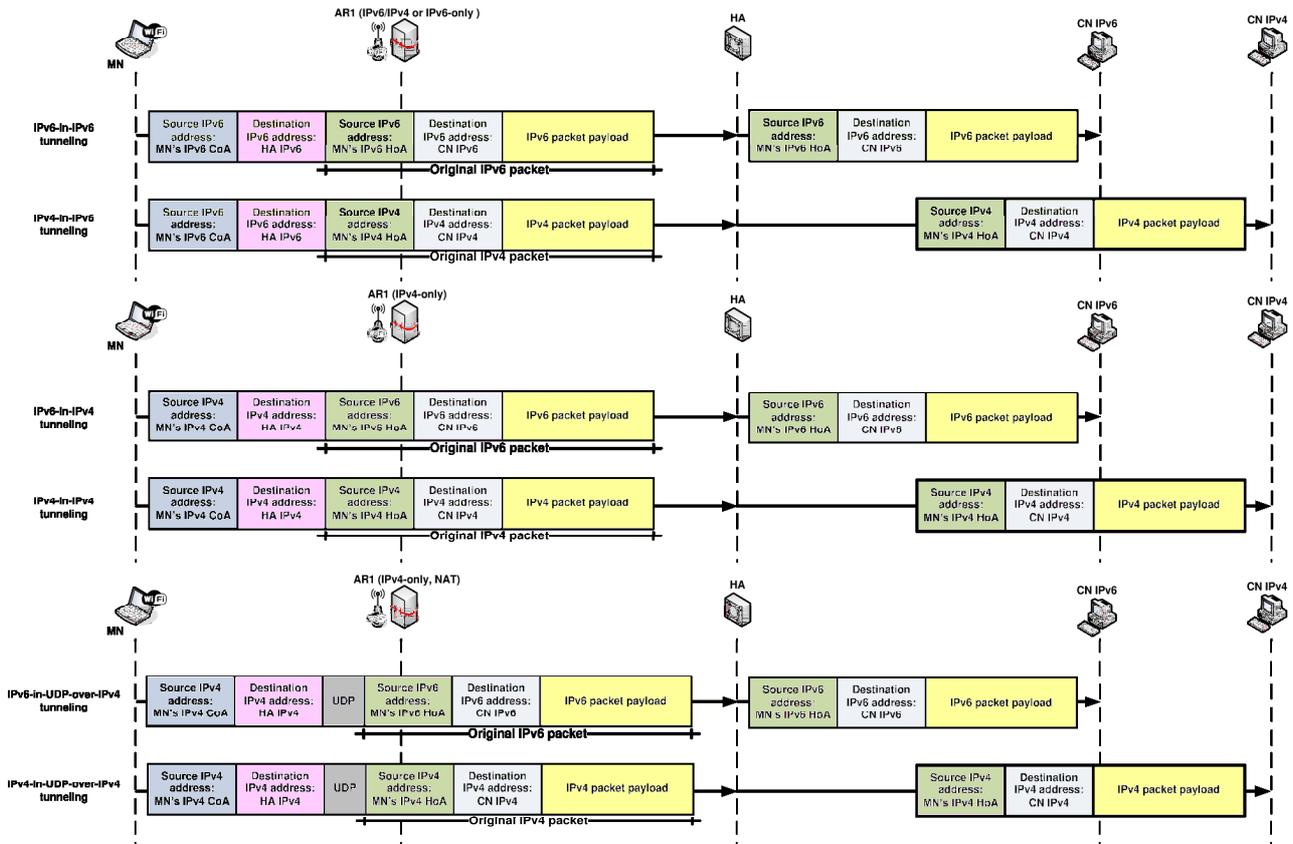


Figure 1: DSMIP tunnelling approaches

The variety of tunnelling formats defined by the RFC 5555 deserves additional attention. First, a mobile node might need to change the tunnelling format as a result of a movement, which can impact on the link and path MTU visible to the applications hosted on the mobile hosts (or on the nodes attached to the mobile network, in case of a mobile router). Because of this, it is not recommended that the mobile node changes the selected tunnelling approach unless it is aware that it can do it beforehand (note that probing the different tunnelling options takes time and therefore the mobile node should avoid doing it every time it moves). Second, the overhead introduced by the tunnelling can account for a large portion of the bandwidth consumed by the mobile node, especially for applications such as Voice over IP, which exhibit small payloads. For example, if we take the iLBC (internet LowBitrate Codec) codec (it is one of the codecs used by the well known Skype application) with an encoding length of 20ms, that results in a payload rate of 15.20 kbps. For a VoIP application using this codec and RTP/UDP over IPv4, the additional packet overhead introduced by Dual Stack Mobile IPv6 (DSMIP) is of 33.9% for IPv4-in-IPv6, 16.9% for IPv4-in-IPv4 and 23.7% for IPv4-in-UDP-over-IPv4 encapsulation.

IPv4 support for PMIPv6

In order to operate over IPv4, extensions to the basic Proxy Mobile IPv6 protocol have been defined in RFC 5844 [30]. These extensions provide two main functionalities:

- IPv4 transport network support;
- IPv4 mobility support.

A network provider managing a PMIPv6 domain can choose to deploy either one or both of the functions depending on their operational requirements.

IPv4 transport network support. Although IPv6 deployment is not yet completed, Proxy Mobile IPv6 requires an IPv6 transport network and an IPv6 home network for its operation. Network providers cannot migrate their entire PMIPv6 domain to IPv6 at once due to network backward compatibility, financial risk, service disruption avoidance, etc. Therefore, it is important for network providers that the local mobility anchor and the mobile access gateways are placed at both IPv6 and IPv4 networks during the IPv6 transition phase. Thus, local mobility anchors and mobile access gateways must be able to forward any packets meant to/from mobile nodes over IPv4 transport networks. In order to support IPv4 transport networks, both LMAs and MAGs must support dual stack and obtain an IPv4 address of the same scope. The IPv4 addresses used can be private IPv4 addresses, but it is assumed that there is no NAT between the local mobility anchors and the mobile access gateways. When LMA and MAG exchange the standard Proxy Mobile IPv6 signalling messages, these IPv6 signalling packets must be encapsulated in IPv4 packets. The use of IPv4 packets to encapsulate the signalling messages imposes that they are securely exchanged with IPsec. As opposed to DSMIP, for the operation of this extension, it is mandatory that the local mobility anchor and the mobile access gateway establish an IPsec security association between their IPv4 addresses and use IPv4 IPsec ESP (Encapsulating Security Payload). In addition, Proxy Binding Update and Acknowledgement messages are no longer carried in a mobility header but in UDP payload, due to the limitations of IPv4 options. Regarding mobile node's data traffic encapsulation (tunnelled between MAG and LMA), there are several possibilities which can be optionally protected through ESP IPsec header:

- IPv4: IPv4 or IPv6 payload packet carried in an IPv4 packet;
- IPv4-UDP: payload packet carried in an IPv4 packet with UDP header;
- IPv4-UDP-TLV: payload packet carried in an IPv4 packet with UDP and TLV (Type, Length, Value) header;
- IPv4-GRE: payload packet carried in an IPv4 packet with a Generic Routing Encapsulation header.

IPv4 mobility support. As long as operators continue providing IPv4 connectivity to applications running on mobile nodes, an IPv4 home address must be assigned to mobile nodes. In a PMIPv6 domain, IPv4 home addresses can be assigned to mobile nodes by any mechanism such as Dynamic Host Configuration Protocol (DHCP), the PPP Internet Protocol Control Protocol (IPCP) or Internet Key Exchange (IKEv2) Protocol.

Proxy Mobile IPv6 must manage IPv4 home addresses on behalf of any address assignment mechanism. For instance, DHCP is just used to deliver the IPv4 addresses assigned by the local mobility anchor to mobile nodes. This guarantees the assignment of the same IPv4 home address whenever a mobile node switches its attached MAG, hence hiding the actual movement to the IP layer. The IPv4 mobility support functionality provided by RFC5844 supports flexible DHCP settings in a PMIPv6 domain such as:

- DHCP server co-located with every MAG;
- DHCP relay co-located with every MAG and DHCP server located anywhere in PMIPv6 domain (most likely DHCP server co-located with LMA);

For detecting a link change (i.e., handover), the mobile node may run DNAV4 (Detecting Network Attachment version 4). However, it may not identify the link change because Proxy Mobile IPv6 is responsible for providing the same default router at any visiting links. Therefore, the mobile node will not perform any DHCP operation after the link change. If the mobile node does not support DNAV4, it may start DHCP rebooting procedure after the link change event. However, the mobile node will obtain the same home address anyway and continue its sessions.

Combining multiple mobility management techniques

The current trend in the evolution of mobile communication networks is towards terminals with several network interfaces (these are already a market reality) that get ubiquitous Internet access by dynamically changing access network to the most appropriate one. Handovers between different access networks are going to be usual. In this situation the different mobility solutions developed by the IETF are going to co-exist because each of them addresses different requirements. The article in [31] has identified, described and analyzed different combinations of IP mobility protocols and the functionality that they provide. An important result of that analysis is that such combination also involves a cost, both from the point of view of

additional overhead and from the point of view of handover delay. Regarding handover delay, combining different mobility solutions has an impact on the performance, even when some of the steps required by each of the mobility protocols – such as movement detection – are shared, resulting in a longer handover interruption. This performance penalty can be intolerable in certain cases. These results raise the need to develop mechanisms to alleviate the costs of combining different mobility solutions, while keeping the advantages brought by the combination.

The background on the Internet mobility field leads to conclude that the universal adoption of a single mobility solution (along with a specific and fixed set of functionalities) doesn't happen in practice. Instead we advocate for solutions that include in their design the flexibility to activate and deactivate the use of their supported mobility functionalities. For example, it would be interesting to design a mechanism so that the mobile nodes and the network can negotiate the mobility support functions required in a particular situation, choosing a particular set of functions from those available as needed. This implies placing a very active role in the mobile nodes regarding the use of mobility solutions. This is in contradiction with part of the motivation for the network-based mobility approach that has been favoured by some operators. However, the heterogeneity of access networks and the current trend towards environments with several options for network access from different operators make unavoidable to place more responsibility in the mobile nodes regarding the mobility solutions selection, at least if they want to enjoy an optimized performance adapted to their needs but compatible with network requirements. The key properties needed in such mechanisms are: flexibility to allow the activation and deactivation of mobility functionalities according to mobile and network requirements, the existence of a system so that the network and the mobile nodes can exchange the information needed to support that activation and deactivation, and the provision of a default mobility support service for legacy terminals.

The main contribution of [31] is identifying the problem of combining different mobility support mechanisms and giving an outline of a solution. Further work includes the definition of the information system in an independent way from the mobility solutions. The proposed behaviour – mobile nodes negotiating with the network which mobility support functionalities to use – has to be achieved in an automatic way. It would be completely unrealistic to have users involved in choosing mobility mechanisms, but their preferences must be taken into account. For instance, it might be explored the use of cognitive networks approaches, such as pattern-based agreement techniques, to implement this automatic negotiation.

3.2.2 Network-based DMM

MEDIEVAL's mobility support follows a brand new paradigm called Distributed Mobility Management (DMM), which is now a matter of research within the IETF to develop protocols for the client-based and network-based versions. Given the feedback from the IETF community and other topic-specific workshops, MEDIEVAL's research related to DMM was pushed further to span different scenarios and approaches, especially on the network based side.

In fact, according to [2] and [3], network-based mobility management can be divided into two categories:

- *Fully distributed*, that consists on removing any central anchor both for the data and control planes;
- *Partially distributed*, where the central anchor is still present, but for control only, as the data traffic does not need to traverse it.

Several configurations are explored for both categories, detailed in [27], and reported below.

The critical point in the fully distributed approach is that the MARs need a mechanism to learn about MN's movements to address the mobility update to the correct MAR. Indeed, differently to PMIPv6 where a database containing the users' mobility sessions is stored in the LMA, in the fully distributed approach each MAR stores only the database's part related to the MNs that are currently attached to them, and this information is also incomplete if the MN has been roaming among the access networks. Hence the MARs need to recover the whole picture about the MN (if it exists) by collecting pieces of information from the formerly visited MARs. Four options can be adopted:

- **PBU Broadcasting.** When a MAR detects the attachment, it broadcasts a PBU to all the MARs and waits for a reply that might be void in case the MN joined the network for the first time. Sending the messages only to the neighbour MARs is not enough to reconstruct the MN's past history, thus the procedure might result excessively long and introduce unnecessary signalling.
- **Terminal indication.** The MN explicitly sends the prefixes acquired previously when it joins the new access network. Although this is a fast and light procedure, it requires some capabilities on the host that are not always possible to rely on (or not even desirable), and might pose some security issues if a malicious MN misbehaves. Also, no standard messages from the mobile host to the access router are available to support such feature, and extensions might face reluctance in the community (e.g., dedicated options in the Router Solicitation message).
- **Automatic learning.** After a handover, the new MAR learns the previous MN's location by inspecting the source address of packets belonging to ongoing sessions, as they contain the old prefix configured by the MN. That is, the MAR stores in an internal database the prefix pools belonging to the other MARs, and, when an MN moves, upon receiving the first uplink packet, they check which access router the prefix belongs to and establish a tunnel with it. This requires little signalling but it may lead to an excessive delay if the MN does not send anything and the router has to explicitly request a packet.
- **Handover control infrastructure.** The handover may follow a Make-Before-Break approach and integrate layer-2 and layer-3 mobility procedures within the same framework to assist and drive the handover. IEEE 802.21 for Media Independent Handover is a suitable protocol for this purpose and the one chosen for the mobility subsystem in the MEDIEVAL project, as thoroughly detailed in [9].

The key issue of fully distributed solutions can be solved at the cost of keeping the control plane centralized and only distributing the data plane (which is the most critical one). The partially distributed approach is widely described in [17], an IETF Draft that is rapidly evolving to account the latest achievements and feedback. According to the present document, most of the control plane burden is transferred to a centralized network node. Following this principle, in the DMM architecture an additional node is added, called Central Mobility Database (CMD), storing the mobility bindings for the MNs. It maintains the control functions of an LMA but it is relieved of the data forwarding plane since it is not traversed by users' data traffic. This centralized node maintains a global view on the network's status and MARs leverage on it to access and update information related to the MNs, stored as mobility sessions. The CMD is queried whenever a MN is detected to join the mobility domain. It might be a fresh attachment or a handover, but as MARs do not store any mobility session, they contact the CMD to retrieve the data of interest and eventually take the appropriate action. The procedure adopted for the queries and the messages exchange sequence might vary to optimize the update latency and/or the signalling overhead: here is presented one method for the initial registration, and three different approaches to update the mobility sessions using PBUs and PBAs. Each approach assigns a different role to the CMD:

- the CMD is a PBU/PBA relay;
- the CMD is a MAR locator;
- the CMD is a PBU/PBA proxy.

The **initial registration** is illustrated in Figure 2 and it works as follows: upon the MN's attachment to a MAR, say MAR_1 , an IPv6 global prefix belonging to the MAR's prefix pool is reserved for it ($Pref_1$). The prefix is sent in a PBU with the MN's Identifier (MN-ID) to the CMD, which, since the session is new, stores a Binding Cache Entry (BCE) containing as main fields the MN-ID, the MN's prefix and MAR_1 's address as Proxy-CoA. The CMD replies to MAR_1 with a PBA indicating that the MN's registration is fresh and no past status is available. MAR_1 unicasts a Router Advertisement (RA) to the MN including the prefix reserved before, that can be used by the MN to configure an IPv6 address (e.g., with stateless auto-configuration). The address is routable at the MAR, in the sense that it is on the path of packets addressed to the MN. Moreover, the MAR acts as a plain router for those packets, as no encapsulation or special handling takes place.

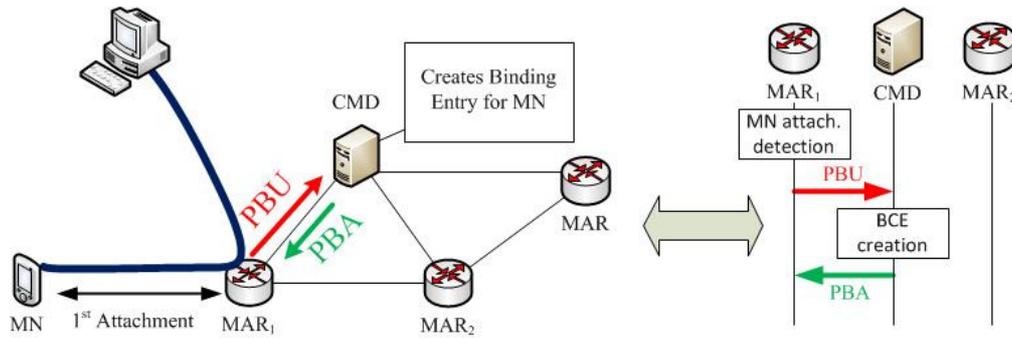


Figure 2: Initial registration in partially distributed mobility management

When the **CMD operates in relay mode**, it forwards the PBU message received from the MAR that is attempting to register the MN to the former MARs (if any) that were formerly serving the MN. In detail (see also Figure 3), when the MN moves to another MAR, say MAR₂, the latter delegates another IPv6 prefix (Pref₂) and it sends it within a PBU to the CMD for registration. The CMD has already an entry for the MN, binding the MN-ID to its former location; thus, it forwards the PBU to the MAR(s) indicated as Proxy CoA(s), (in this case MAR₁), and it updates the P-CoA field with MAR₂'s address. Upon the PBU reception, MAR₁ replies to the CMD with a PBA containing the prefix anchored to it to ensure that the new location has successfully changed. The CMD updates the BCE adding MAR₁'s address in the list of old P-CoAs and forwards the PBA to MAR₂, containing the previous Proxy-CoA and the prefix anchored to it, so that a tunnel can be established between the two MARs and new routes are installed appropriately to recover the flow(s).

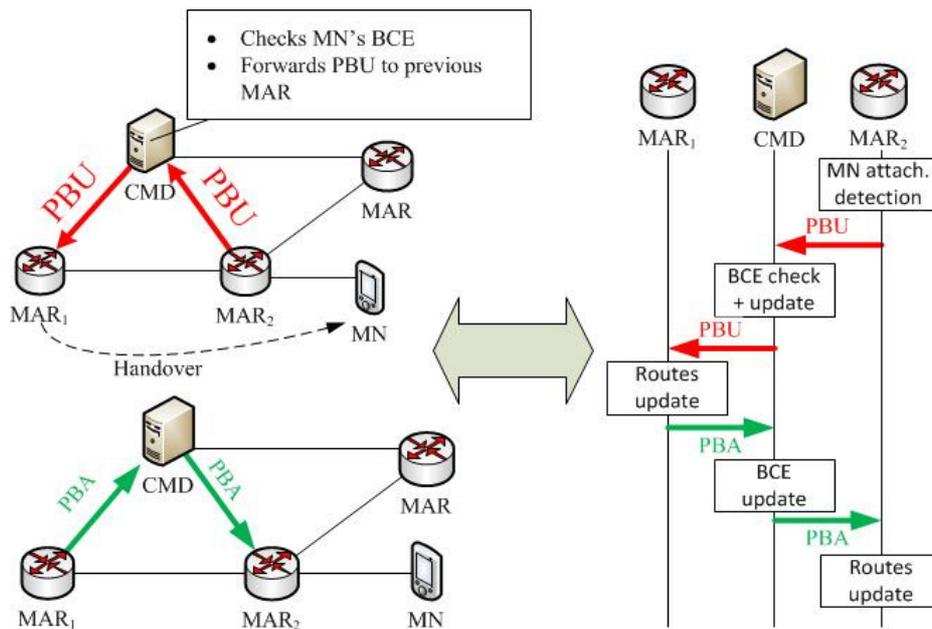


Figure 3: Protocol operations when CMD is in relay mode

As mentioned before, data traffic does not traverse the CMD as shown in Figure 4. Now, packets destined to Pref₁ are first received by MAR₁, encapsulated into the tunnel and forwarded to MAR₂, which finally delivers them to their destination. In uplink, when the MN transmits packets using Pref₁ as source address, they are sent to MAR₂, as it is MN's new default gateway, then tunnelled to MAR₁ which routes them towards the next hop to destination. Conversely, packets carrying Pref₂ are routed by MAR₂ without any special packet handling both for uplink and downlink.

When the **CMD works in locator mode**, the latency due to the mobility update can be mitigated. Indeed the time required to install tunnels and routes between MARs can be reduced if old MARs, upon receiving a PBU from the CMD, reply directly to the new MAR, by-passing the CMD. Actually, as shown in Figure 5,

two PBAs are sent by MAR₁: one to MAR₂ to build the routing state, and another to the CMD to update the parameters related to the MN's mobility session. Compared to the solution described before, this procedure permits to save some time thanks to faster message transmission, but it introduces additional control messages. The data forwarding reflects that illustrated in Figure 4.

The advantages of the two schemes proposed before can be combined when the **CMD operates in Proxy mode**. Indeed, as depicted in Figure 6, further enhancements can be achieved when the CMD sends the PBA to the new MAR before notifying the old MARs about the location change. Indeed, when the CMD receives the PBU from the new MAR for the registration phase, it is already in possession of all the information that the new MAR requires to set up the tunnel and the routes on its side. Thus the PBA is sent to MAR₂ immediately after a PBU is received. In parallel, a PBU is sent by the CMD to the old MARs to notify them about the new MN's location, so they receive the necessary information to establish the tunnel and routes on their side. When old MARs complete the update, they acknowledge the CMD with a PBA to indicate that the operation is accomplished and the information is updated in all network nodes. Again, the data forwarding is kept untouched as in previous schemes.

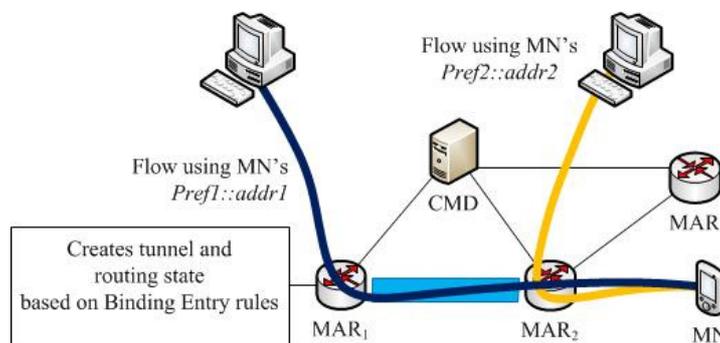


Figure 4: Data forwarding after handover

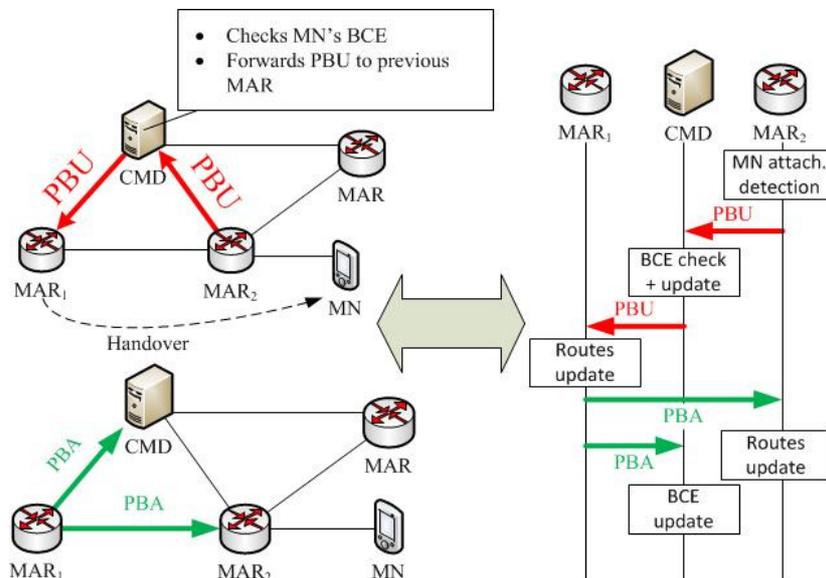


Figure 5: Protocol operations when CMD is in locator mode

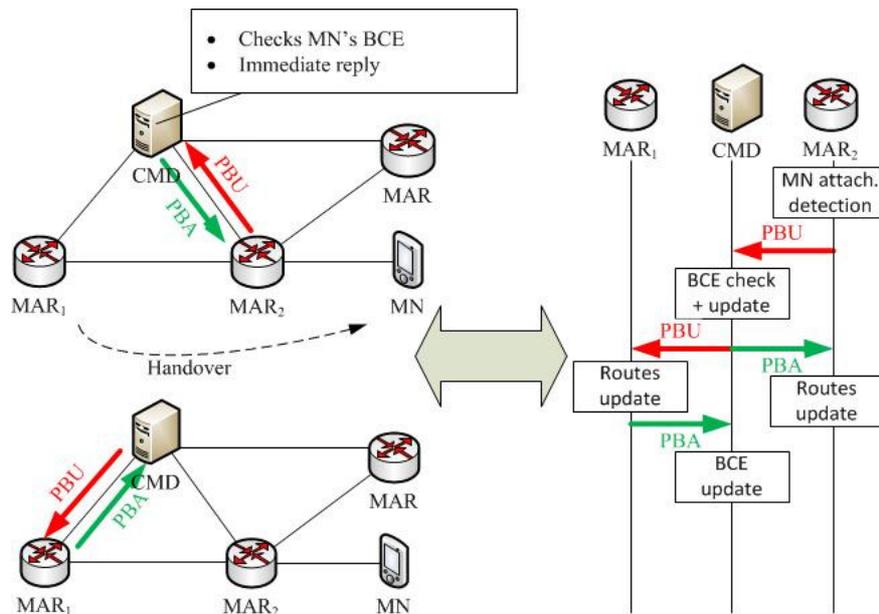


Figure 6: Protocol operations when CMD is in proxy mode

3.2.3 Multicast Mobility

IP multicasting takes a significant role for efficient multimedia data delivery in mobile environments. As DMM basically aims at distributing data packets previously concentrated towards a single anchor throughout multiple anchors located at access routers, it is required to find out appropriate IP multicasting solutions according to features and classifications of DMM architectures, i.e. fully-/partially-distributed DMM and proactive/reactive handoff modes, mentioned in Section 3.2.2.

With these objectives, [46] presents applicable IP multicast use-cases on various DMM alternatives and analyzes their performances mathematically from the user perspective. Strictly related, in [25] IP multicast use-cases are reviewed and examined in terms of efficient data delivery and routing efficiency.

Based on a series of IP multicasting studies on DMM, [47] presents a channel-manageable IP multicast framework enabling network operators to control IP multicast channels in use and determine multicast data transport mode on each access router to avoid duplicate data transmission caused by tunnel-based packet forwarding.

IP Multicast Use-Cases in DMM

IP multicast use-cases can be differently considered depending on the support of control/user-plane separation and the handoff proactivity. Regarding the former, it may be fully distributed, where the MAG and the LMA roles are collapsed in a Mobility Access Router (MAR) or partly distributed, where the control plane is kept centralized at a single entity, referred to as Multicast Mobility Information Server (MMIS), and acting as mobility signalling relay (as CMD in [17]). MMIS is responsible for storing the prefix and Proxy-CoA's of each MN, identified by respective MN-IDs, and taking an active role in the mobility decision.

Handoff proactivity is determined by whether the layer 3 handover is triggered by the previous MAR (A-MAR) in proactive case, or the new MAR (S-MAR) in reactive case. This impacts the speed of the mobility process execution and affects the multicast service disruption. Considering time-sensitive services such as live video streaming, proactive scheme would benefit real-time services such as video-conferencing.

Figure 7 presents signalling procedures of IP multicast use-cases on the various DMM alternatives mentioned above. Four combinations of IP multicast use-cases, Reactive/Proactive schemes in Partially-distributed and Reactive/Proactive schemes in Fully-distributed, are called as (RP/PP) and (RF/PF), respectively – throughout this section, RP will always refer to the scheme, and not to the Rendezvous Point function.

Reactive schemes

Reactive approaches are those triggered by the S-MAR. When using scheme RP, the mobility process is as follows: as the S-MAR detects the MN's presence, it sends a regular PBU to the MMIS (Figure 7 (a)), which forwards it to the A-MAR after checking its database. The A-MAR will then reply with a PBA to the MMIS, which forwards the message back to S-MAR, to complete the tunnel establishment. An alternative would be to send a PBA to both the MMIS and to S-MAR, as seen in [27]. Afterwards, S-MAR will query MN for its multicast interests, and subscribe the channel using an aggregated MLD Report.

In the RF scheme, it is assumed that the S-MAR knows the A-MAR(s) address(es). The exact process for obtaining this crucial information is left open, being one possibility to use 802.21 MIH protocol for dealing with the handover preparation, optimization and completion stages [41]. The mobility process can be summarized as a PBU/PBA exchange between S-MAR and A-MAR, followed by the MN's MLD Query, respective Report, and finally the aggregated MLD Report.

Proactive schemes

Proactive approaches are triggered by the A-MAR, requiring prompt identification of the MN's detachment. In PP, the A-MAR signals the MMIS using a deregistration PBU. This message contains two new options: destination MAR option, which should contain the S-MAR's address, and multicast subscription option, embedding the multicast context relative to the MN. When the A-MAR doesn't know destination MAR, it will send the option empty, leaving the decision to the MMIS. The MMIS will then reply with a PBA and send a Handover Indication (HI) containing the multicast context to the S-MAR, which replies with a Handover Acknowledgment (HACK) and then subscribes the missing multicast channels.

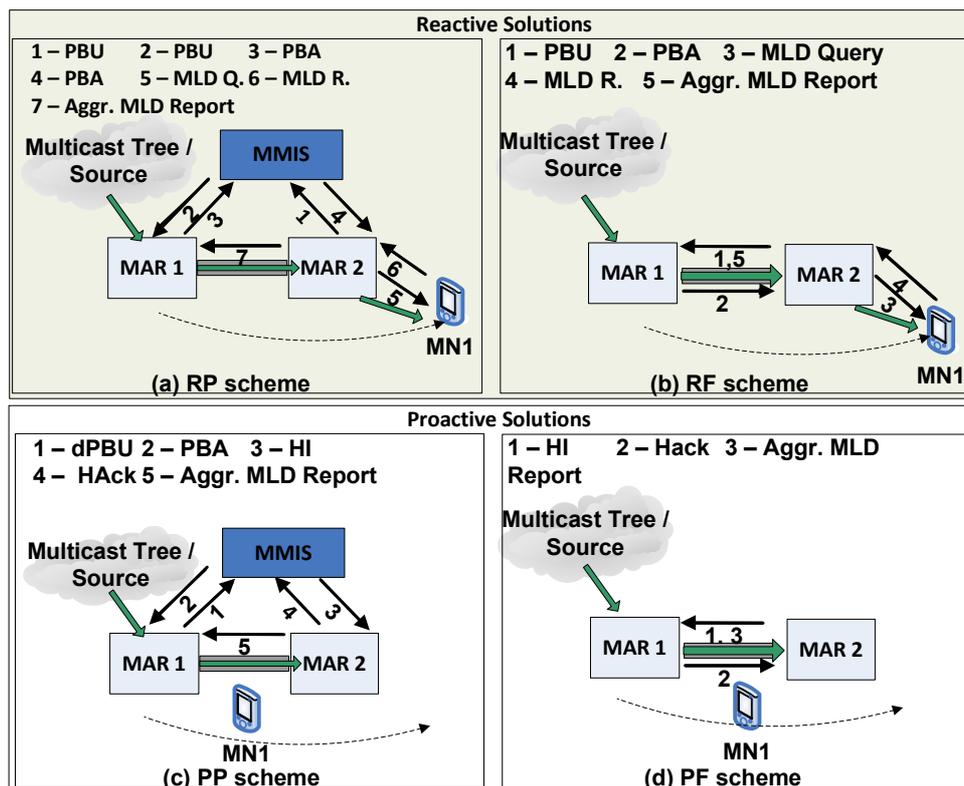


Figure 7: IP multicast scheme use-cases for DMM

Problems Description on IP Multicast Use-Cases for DMM

The work [25] focuses on describing problems that occur when IP multicasting is deployed on a network-based DMM similar to that of MEDIEVAL. It is assumed a MAR is equipped with MLD Proxy function [44] and that a tunnel-based data transport is used whenever the MN moves to another MAR.

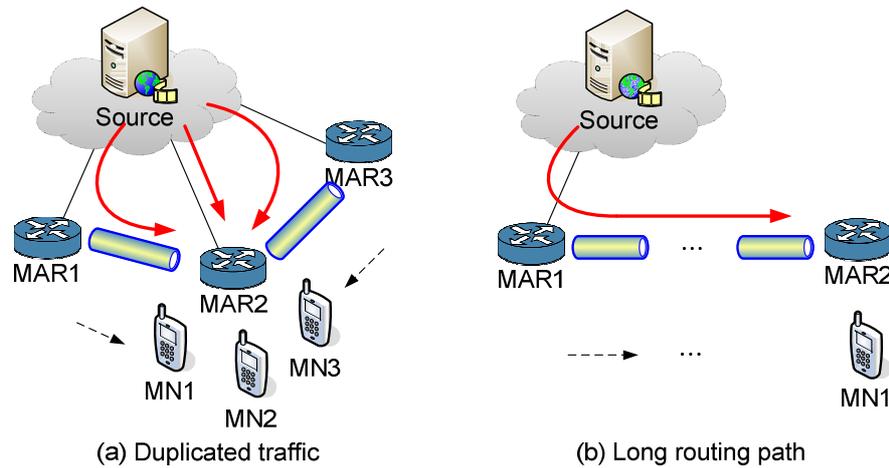


Figure 8: Problems description when existing IP multicast approach is applied on DMM

One of problems found is duplicated traffic, which is quite similar to the tunnel convergence problem occurring in [43]. As shown in Figure 8 (a), MN1 and MN3, which moved from MAR1 and MAR3, respectively, are currently located at the MAR2. Through respective established tunnels, they receive multicast packets of the same channel through different anchoring MARs. This causes duplicated traffic, converging to the MAR2, with the magnitude of replicated traffic.

As referred, when a MN first subscribes multicast content, its current MAR's MLD Proxy will forward its subscription to the multicast infrastructure. As such, an extra duplication factor may occur, if the subscription is already being received from one or multiple tunnels due to other listeners.

Another issue is non-optimal routing as shown in Figure 8 (b). If we consider a significantly large domain, there is the possibility that the multicast packets need to traverse a long distance, depending on the setup of the upstream interface of MLD-P instance, even if the current MAR is connected to the multicast infrastructure. This issue is not present if the operator deploys a MLD Proxy instance with the upstream interface towards multicast source or multicast routing network for each MAR. Such an approach is extremely simple and mobility-agnostic, but it may occur media synchronization issues and service disruption during handoff.

IP Multicast Framework Considered for DMM

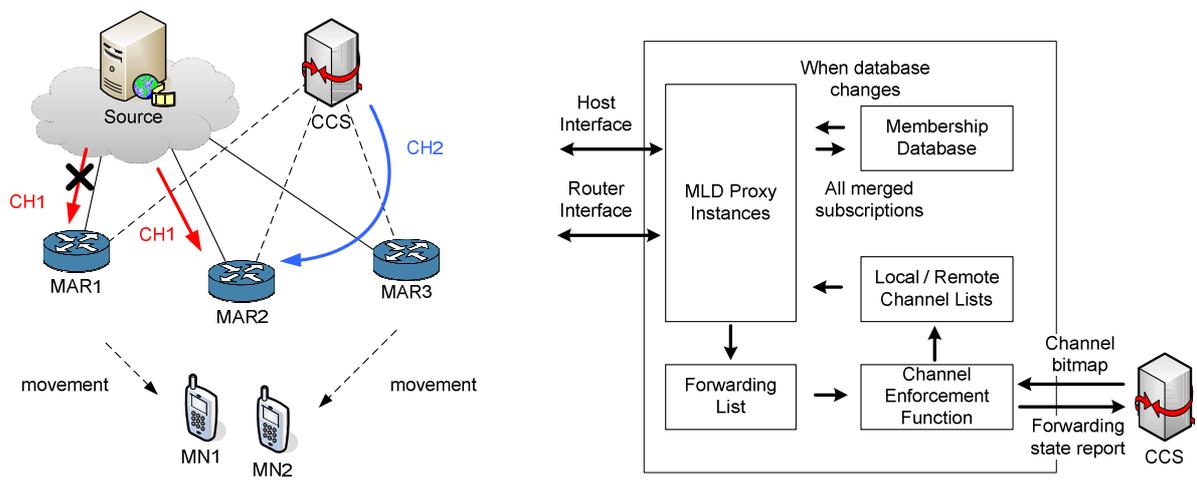
From IP multicasting studies done on DMM before, it is considered that operator-driven IP multicast framework needs to have the following concepts:

- **Minimized duplicate multicast traffic:** it should minimize the amount of duplicate multicast traffic occurrences, made by multiple tunnels established between multiple MARs in DMM architecture.
- **Centralized channel management:** it should support centralized channel control framework to facilitate effective multicast traffic distribution. To apply a variety of requirements into operators' networks, a policy-based channel control management should be possible. This policy would then depend on the operators' network environments.
- **Network-based channel management:** the channel management, regarding on where should be upper multicast delivery router for requested multicast channel, should be determined without MN's involvement. This decision should be made based on channel lists received from a channel control server.

The proposed multicast mobility scheme is a channel-manageable distributed mobility management (CM-DMM) solution. Compared to generic DMM architecture, a channel control server (CCS) is added. The utilization of a MLD Proxy on a MAR is also assumed. The CCS communicates with a channel enforcement function (CEF) integrated on MAR and provides channel lists classified by 'L' or 'R', representing which channel should be 'locally' or 'remotely' supported.

Once a MN attaches to a MAR, the direction of upstream interface is set based on the received policy from the CCS. This policy could be set based on the channel’s popularity, or any other motive the operators finds valid for the goal of good balance between network efficiency and fast multicast session initiation. An example scenario is shown in Figure 9 (a): MN1 and MN2, which were listening to CH1 and CH2 at MAR1 and MAR3, respectively, both move to MAR2. The channel policy on MAR2 defines CH1 as local only and CH2 as remote, allowing tunnel establishment to other MARs. Following the channel policy, MAR2 will not transmit an MLD Report message to MAR1 although MAR1 is MN1’s anchor, but the message is directly routed to an upstream IP multicast router. As a result, an MLD Report message for CH2 is transmitted towards MAR3 so that MAR2 receives the multicast packets through the tunnel established between MAR2 and MAR3.

The meaning of “local channel” is valid in the perspective of receiving multicast packets. If MAR2 is asked to forward CH1 packets to other MARs, it adds a new downstream interface to the corresponding MLD Proxy instance towards requesting MAR. These channel policy decisions can be made by the operators. For example, in the case of a worldwide popular sports game, the operator can make the channel supported through direct routing at every MAR while unpopular channels are allowed through both routing mechanisms, therefore a huge amount of reduced traffic are expected. In addition, improved transmission delay between source and listener may be also expected depending on the channel configuration.



(a) Data forwarding control in proposed CM-DMM (b) Multicast function structure of MAR in proposed CM-DMM

Figure 9: Multicast operation and function in presented CM-DMM

Figure 9 (b) shows multicast function inside the MAR, indicating how local/remote channel settings can be handled. Basically, a MLD Proxy on a MAR follows the operations of MLD Proxy defined in RFC 4605 [44] so that it detects newly attached downstream link from a MN and configures appropriate upstream interface to receive multicast packets. In CM-DMM, we employ the channel enforcement function (CEF) that communicates with the CCS. It receives a channel bitmap information represented with the form of channel number and bit for processing which channel is supported as local or remote. The CEF writes the information into local/remote channel list. Based on the channel bitmap list, the upstream interface of the MLD Proxy to the requested channel is determined. The CEF periodically sends status reports based on forwarding lists to the CCS. The reports can be used to check channel popularity and traffic intensity and other management functions.

3.2.4 Flow Mobility

The typical real world scenario for mobile networks is a user equipped with a dual mode mobile phone (e.g., integrating 3G/4G and WiFi radio devices) attaching to the available networks either sequentially or simultaneously. The latter case is commonly referred as multi-homing case, that is, the user can receive data over different networks (WiFi or 3G/4G) simultaneously. In this situation, IP flow mobility allows a Telecom operator to seamlessly and selectively switch over a single IP flow (e.g., user application) to a

different radio access, while keeping all other ongoing connections for this and the rest of the users on both radio accesses untouched. The use of greedy access with several technologies implies the downside of higher energy consumption, representing a trade-off between better QoE and device battery lifetime. The technology is currently being standardized in the IETF and it has been adopted by 3GPP as technique for seamless WiFi offload.

IP flow mobility technology has the following key advantages:

- it allows the user to enjoy high bandwidth connections in proximity of WLAN hotspots while being always reachable from the Internet;
- it allows the operator to manage the bandwidth in the presence of greedy user connections;
- it allows the operator to provide different levels of service by applying different policies for different users, tariffs and specific traffic types evolving from a simple pipe provider to a high leverage network provider.

The access and core networks are therefore capable of classifying data traffic traversing their nodes and, in agreement with the mobile devices, can apply policies to deliver the best available Quality of Experience (QoE). Note that traffic redirection always introduces some kind of delay, due to processing, forwarding and the different properties from distinct radio access networks. Nevertheless, the solutions being investigated at the 3GPP do not consider the redirection of real-time or delay sensitive traffic, such as real-time video or voice. Offloading traffic to the non-3GPP access is currently considered only for non-critical traffic such as bulk downloads, non real-time video or P2P traffic.

The article in [32] analyzes and compares the two possible approaches to IP flow mobility, namely client-based and network-based IP flow mobility; it summarizes the key functional boxes and associated protocol operations, discusses the pros and cons of each solution. The paper also generalizes the adoption of network-based solutions in the context of 3GPP and the use of alternatives network-based mobility protocols like the GPRS Tunneling Protocol (GTP).

Flow Mobility extensions for MIPv6

The basic Mobile IPv6 specification and the extensions defined to enable IPv4 operation provide a very limited multi homing support, as each permanent address (home address) can only be associated to a single temporal address (care-of address). Therefore, the only possible scenario in which a mobile node can use more than one care-of address simultaneously is that in which the node is using different home addresses (one per care-of address). This limits the scope and usability of this basic solution as it prevents different flows to be routed to different care-of addresses, and consequently, does not support a scenario in which a mobile node is reachable – via a single home address – through different physical interfaces. In order to enable flow mobility in a client mobile IP context, the IETF has standardized the basic components that are required. These components are:

- i) multiple care-of address registration support;
- ii) flow bindings support;
- iii) traffic selectors definition.

Basic Mobile IPv6 protocols provide the tools to bind a home address to a single care-of address. Since flow mobility requires the ability of receiving traffic destined to the same home address via different care-of addresses, Mobile IPv6 needed to be extended to support the registration of several care-of addresses with the same home address. This is the purpose of the Multiple Care-of Addresses Registration extensions, standardized in the RFC 5648. These extensions allow a mobile node to register multiple care-of addresses for a home address and create multiple binding cache entries. In order to do so, the Binding Update (BU) message defined by Mobile IPv6 is extended with a new mobility option used to carry a care-of address and a number to uniquely identify the binding entry, called Binding Identification (BID) number. A mobile node can include a number of these new mobility options in the BU message, triggering the creation of multiple

binding cache entries in the home agent, each of them identified by the respective BID. Note that the binding cache and binding update list structures are also extended to support the multiple care-of address registration.

In addition to the capability of associating a single home address with multiple care-of addresses, the ability to use and control them simultaneously is required. This is the goal of the second set of extensions, the Flow Bindings in Mobile IPv6 and NEMO Basic Support – standardized in RFC 6088 – which allows mobile nodes to bind one or more IP flows to a specific care-of address. With this extension, a mobile node can instruct the home agent (or the correspondent node) how to route inbound packets (i.e., to which care-of address packets of a specific flow should be sent). Note that the mobile node also needs to have support to be able to route outbound packets via different care-of addresses, being that packet forwarding coherent with the inbound policy signalled by the mobile node. The flow bindings specification basically defines a set of Mobile IPv6 options and sub-options allowing the mobile node to associate a particular IP flow (which is also assigned a Flow Identifier, called FID) with a particular care-of address (identified by its BID). These bindings between IP flows and entries in the binding cache are stored in a different conceptual list, that is looked up in order to determine which entry of the binding cache has to be used to forward a data packet. This list basically includes the FID, a traffic selector that is used to assign packets to flows (i.e., a flow is defined as a group of packets matching a traffic selector), and a FID priority – used to break the tie between overlapping flow bindings.

The last above-mentioned extension required to enable IP flow mobility is the definition of traffic selectors for flow bindings, standardized in RFC 6089. This extension basically defines binary formats for IP traffic selectors to be used in conjunction with the flow binding extensions, so IP flows can be identified according to different criteria, such as: Source Address, Destination Address, IPsec SPI (Security Parameter Index) value, Flow Label, Source Port, Destination Port, Traffic Class or type of Next Header. A flow can be identified by any subset of these parameters or by specifying a range of values for each one, hence identifying several flows at the same time.

Flow Mobility extensions for PMIPv6

Although the basic specification of PMIPv6 provides limited multihoming support to multimode devices, it does not include the ability to move selected flows from one access technology to another. This functionality is currently being developed by the IETF NETEXT WG as described in [20].

Flow mobility assumes simultaneous connection to the same PMIPv6 domain through different interfaces. The simultaneous use of different attachments to the network increases the complexity of the solution due to two main reasons:

- i) in order to support flow mobility, the MN must be able to send and receive traffic to/from any prefix associated to it through any of its interfaces. This functionality can be provided by different mechanisms. Two of the mechanisms that have been studied at the IETF are the Weak Host Model and the Logical Interface. On one hand, the Weak Host Model corresponds to the implementation decision taken while designing the IP stack. In a mobile node implementing the Weak Host Model, the IP stack accepts any locally destined packet regardless of the network interface on which the packet was received. On the other hand, the Logical Interface is a software entity which presents one single interface to the IP stack, and hides the real physical interface implementations (e.g., modems). Hence, the IP stack binds its sessions to this Logical Interface and it is oblivious of the actual physical interfaces receiving or sending packets. One of the principles of PMIPv6 is to achieve a mobility solution in which the IP stack of the mobile node is completely unaware of the mobility. In order to maintain the MN's IP stack unaware of mobility while providing flow mobility support, the IETF has chosen to rely on the concept of Logical Interface (LIF) [19].
- ii) In the general case, through the use of flow mobility, the MN will be able to receive any traffic destined to any of its IPv6 addresses through any of its interfaces. This represents a problem at the MAG level, since in order to support flow mobility, the MAGs must be able to forward any prefix associated to the MN even if this prefix was delegated by a different MAG. This situation is being solved by the IETF through the addition of extra signaling to the standard PMIPv6 so that the MAGs can be configured appropriately.

Logical Interface. The Logical Interface is a software entity that hides the real physical interface implementation to the host IP layer. Its use allows the MN to provide a single and permanent interface view to IP and the layers above, that can bind to this interface in order to establish any remote communication. Internally the logical interface is able to leverage several functionalities such as inter-technology handover, multihoming or flow mobility, while presenting always the same IP address (or set of IP addresses) to higher layers. The logical interface is commonly implemented as part of the connection manager software of the mobile terminal, which is in charge of handling and automatically configuring the different network interfaces. Therefore, although the implementation of the logical interface concept require some changes on the client side, those are part of an already required terminal component (the connection manager), and do not have any impact on the IP stack, which remains standard. This interface is implemented as a logical entity that bonds several physical interfaces (e.g., WiFi and 3G) into a unique interface, which is used by IP and higher layers. The LIF hides to the IP layer the physical interface used to actually send each data, hence a movement of a flow from one interface to another is transparent to the IP and higher layers. Even more, it supports sequential attachment of interfaces as they come up, so the flow mobility features can be started in order to offload some interface or network (e.g., 3G offload) as soon as a new interface becomes active (e.g., a WiFi interface associates with an Access Point), without the higher layers being aware of it. The LIF is sometimes referred to as Virtual Interface.

Signalling extensions to PMIPv6: As explained above, signaling extensions to PMIPv6 are required in order to provide the MAGs with the information regarding the different prefixes used by the MN. This information exchange is needed since, in general, a MAG will not forward traffic from/to a prefix that has not been delegated by it to the MN. In [20] several cases showing the possible configurations for the combinations of prefixes and interfaces are detailed. The IETF currently focuses on two scenarios:

- i) the so-called “handover with full flow granularity”, which consists in the movement of a specific flow from one interface to another (e.g., a video-conference where the voice is going through a reliable interface such as 3G and the video through a high bandwidth link such as WiFi, but both flows are addressed to the same prefix);
- ii) the movement of a complete prefix and all the communications using it, to another interface, scenario often referred to as “partial handover”.

Both cases face the problem of requiring the target MAG to get knowledge regarding the prefixes through which the MN is receiving traffic. Flow mobility signaling takes place whenever the LMA decides to move a flow from one access to another. At the time of movement, either the prefix is already known at the target MAG or the LMA must advertise it to the MAG which is going to receive traffic addressed to this prefix. In the case the MAG already knows the target prefix, the LMA simply switches the flow to the target MAG, and no extra signaling is required. In the case signaling is required, the IETF is defining new messages to manage the notification to the MAG of the new flow/prefix to be forwarded. It should be further noted that one of the most common technologies to perform traffic classification is Deep Packet Inspection (DPI). It can be performed offline – for instance for billing and charging purposes, or for security checks – or online. In the context of IP flow mobility we envision the use of online DPI to classify traffic according to operator policies. By means of DPI the network becomes intelligent and enables innovative use cases on traffic handling (e.g. multi link diversity utilization, bandwidth aggregation, etc...).

3.2.5 SVC and Mobile Networks

The article in [33] showcases a typical scenario for SVC video enhanced delivery that represents a starting point for the development of the MEDIEVAL projects implementation. In this paper, we study innovative and efficient video delivery methods for dual mode handsets implementing cellular and WiFi technology in a network-based mobility management architecture using Proxy Mobile IPv6 (PMIPv6). By exploiting the characteristics of the SVC mechanisms and the possibility to be simultaneously connected to both cellular and WiFi networks, we show how policies can steer video traffic across both wireless access technologies. Media encoded with SVC has the peculiar property of generating different IP packets each containing a different quality layer. The reception of low quality layers does not prevent the player to still play the video, at the mobile side, even if at a lower quality than planned. The experiments conducted confirm that it is

possible to split the video flow across different wireless access technologies and that the WiFi connection can be used to boost the received video quality without impacting the overall playback experience.

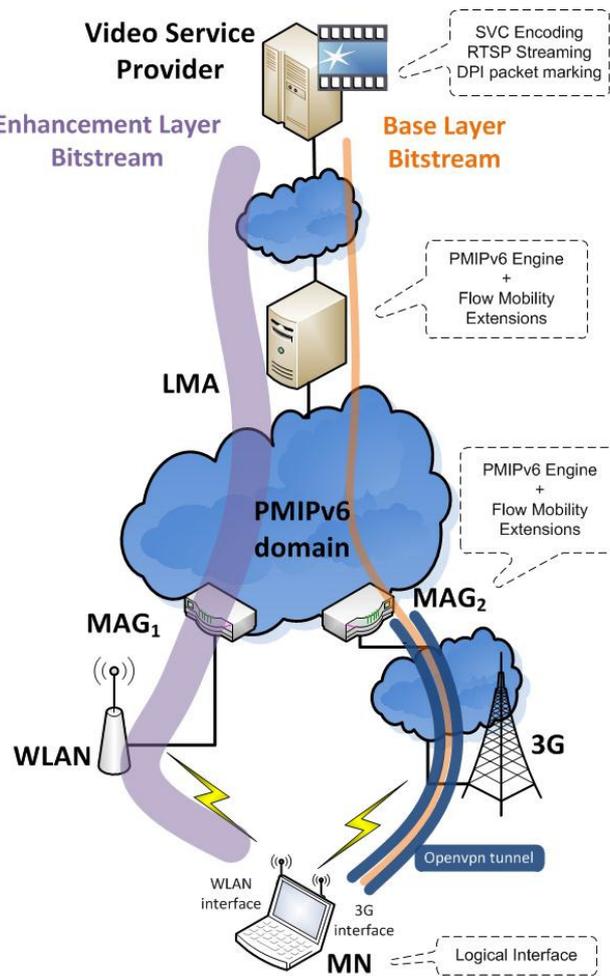


Figure 10: Experimental setup for SVC video enhanced delivery

Figure 10 depicts the experimental setup where our tests have been conducted. The testbed features an MN implementing the LIF over 3G (USB dongle) and WiFi (built-in) interfaces. The cellular part relies on an Alcatel-Lucent in-house UMTS network while the WiFi is a standard IEEE 802.11bg Access Point. Each access technology is connected to a different MAG: WLAN is configured for direct IPv6 connectivity between MAG and MN, while over the 3G access a VPN software (OpenVPN[51]) is used to encapsulate IPv6 over IPv4, due to the limited availability of IPv6 in 3G access. The MAGs implement the IP flow Mobility extensions developed for [36], while the LMA box includes flow mobility management software. The video server, based on Live555 library [52], streams an SVC based video through an RTSP encapsulation. It also takes advantage of DPI techniques in order to set the flow label field in the IPv6 header with a mark according to the SVC layer. As video client, the MPlayer [53] application was selected, since it supports the SVC codec through Open SVC Decoder [54] library. The video used for our tests is a two-minutes long scene taken from the “Big Buck bunny” animation movie [55] with resolution 640x360 encoded with JSVM [56] software, using SNR MGS mode with two layers: *basic*, with QP = 46 resulting in an average bit rate of 150 Kbps, and *enhancement*, with QP = 26 and rough average bit rate of 900 Kbps. The overall video stream is hence transmitted at more than 1 Mbps on average. We argue these characteristics are typical for streaming good quality videos on hand-held devices.

Our experiment consists of streaming the SVC codec video, which in the remainder will be referred as *SVC local*, from the server outside the PMIPv6 domain to the MN, under three different scenarios, which depend on the availability of connectivity options from the MN:

- *SVC 3G* scenario: the MN is attached to the 3G MAG only, and both SVC video sub-streams are delivered to the client, as no policy is defined at the flow manager;

- SVC 3G base scenario: the MN has only the 3G link active, but, according to an operator's decision (e.g., due to congestion), the flow manager at the LMA drops the flow related to the enhancement layer, based on the assumption that the 3G network cannot cope with a satisfactory delivery of both layers, or that it consumes too many resources on the access;
- SVC 3G/WiFi scenario: the MN is connected to the network through the 3G and WiFi links. The attachment to multiple MAGs is detected by the LMA and the FM, which installs routing rules to forward the low quality sub-stream via 3G MAG, and the high quality layer through the WiFi MAG.

The system can automatically switch among any of the described scenarios, as the flow manager, Mplayer and the LIF run custom tools to react promptly to a change in the network conditions (which are simulated in our platform), producing the proper adjustments smoothly. From the user's perspective, no manual intervention is required, as MPlayer can seamlessly switch between SVC layers, by using an ad-hoc LIF-to-MPlayer API based on flow and interface information (it can be easily extended to allow sending commands to the NICs, e.g., to power up and/or to associate to an ESSID/APN).

In order to assess the validity of our proposal, we conducted a Double Stimulus Impairment Scale (DSIS) test. Following ITU-R Rec. BT 500-11 for the subjective evaluation of video and audio quality, we showed to 25 users the 3 videos related to the corresponding test scenarios against SVC local, which is taken as reference. After watching the videos, the people involved in the test were asked to rate the degree of impairment with respect to the reference on the standard discrete five-level scale: *Very annoying* (mark 1), *Annoying* (2), *Slightly annoying* (3), *Perceptible, but not annoying* (4) and *Imperceptible* (5).

The results obtained are summarized in Table 1, where the Mean Opinion Score (MOS) and the 95% confidence interval are shown. The complete distribution of the ratings collected for each video sample is depicted in Figure 11. We can observe from the stacked histogram that the video started on the 3G interface does not produce a good experience at the user. Also, it causes considerable resource consumption, hence after a network's decision, the packets belonging to the enhancement layer are dropped. Unfortunately, the video with the base SVC-layer only (SVC 3G base) was rated the poorest, meaning that there was a sensible deterioration in the user's QoE. However, the bandwidth availability can be augmented by establishing an additional link using WiFi. The PMIP+Flow Mobility intelligence is now able to re-direct the video layers through both paths to the terminal, therefore restoring a video quality that is equal or better to what experienced with the 3G only. More, a key-aspect in this latter scenario, is that the resource consumption in the 3G network (the most critical for an operator) is kept identical as that in scenario 2, where the enhancement layer is dropped.

Video	MOS	95% Confidence Interval
SVC 3G	2.44	±0.30
SVC 3G base	1.36	±0.22
SVC 3G + WiFi	3.36	±0.27

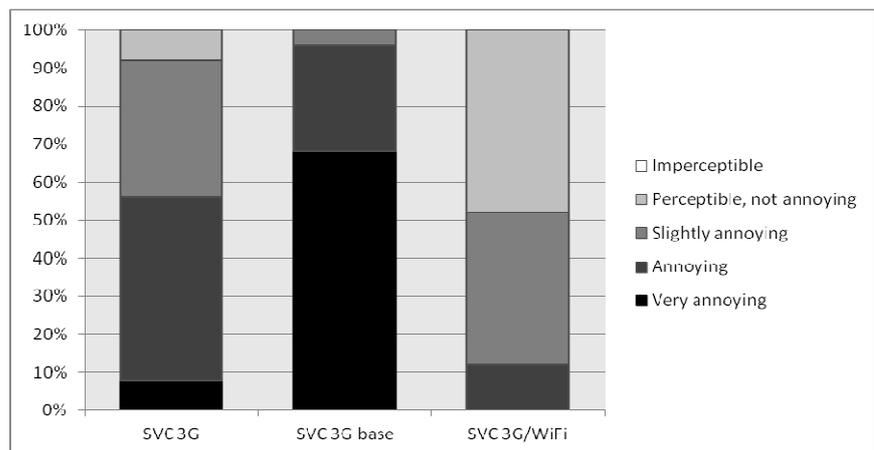


Table 1: Video ratings summary

Figure 11: Marks distribution for the video samples

Live Demonstration

The aforementioned work was demonstrated during the Alcatel-Lucent Bell Labs Open Days, where people were able to have a live sample of how MEDIEVAL-based work can improve the delivery of video as well as the perceived Quality of Experience of the user. To show the broad spectrum of attendees, we hosted people from different fields, backgrounds and areas of expertise, such as: French academia, Alcatel-Lucent Bell Labs co-workers, telecommunications operators, software and hardware manufacturers, businessman from start-ups and large corporations and finally international press.

As can be seen in Annex B, the demonstration architecture is the same as the one seen in Figure 10. The only difference is the software used to encode, decode and stream the live video content. For the purposes of this demo, we decided to use the software provided by MEDIEVAL partner LiveU for encoding, decoding and streaming the video content. As for input, we used a High Definition (HD) webcam to capture a local live video feed.

To maximize visual impact in the streamed video, the two video coded layers were encoded with very different resolutions and bitrates, being the base-layer encoded at a resolution of 320x240 pixels at 120kbps, while the high quality layer was encoded in a resolution of 640x480 at 900kbps.

The objective of the demonstration was twofold. Firstly, we would show the WiFi offload use case, in which the user starts in the 3G access with low quality video streaming and an increase in video quality is triggered by switching to a higher throughput access technology (implying a network managed handover between 3G and WiFi) thus showing the difference in quality by receiving the higher quality layer through WiFi. Secondly we would showcase the advantages of using SVC coding by triggering dynamic adaptation of the video stream due to a congested network. To showcase this second feature we put the user attached to WiFi while streaming a high quality video and then he would be automatically and seamlessly switched to a lower quality streaming due to (manually induced) congestion in the network.

The demonstration concluded that independently of changing video quality and access technology, it is the seamlessness of both quality layer switches and handovers as well as the continuous fluidity of the video stream that are paramount to maintain the user happy and thus maintaining a good QoE.

3.2.5.1 Multicast video layering in mobile scenarios

Previous works have explored SVC as a method to deliver layered multicast video [37], with its advantages over simulcast delivery coming from the protocol inherent scalabilities options (in terms of spatial, temporal and quality-driven levels), which is turned into efficiency. A SVC stream is constructed from Network Abstraction Layer Units (NALUs) to represent a part of the picture's encoded bit stream, which belong to a single layer. The stream is constructed from a basic layer which is not dependent on any other layers, and enhancements layers, dependent on lower layers. Due to this scalable property, the SVC encoding is an ideal technique for providing multimedia multicasting to heterogeneous networks and devices.

In order to transport multicast SVC in RTP packets over heterogeneous environments, Multi-Session Transmission (MST) was specified in [38]. Using this mode, multiple RTP sessions are used to carry the SVC data. Albeit, depending on the application requirements, this may translate into transporting one layer per RTP session or encapsulating multiple layers in one RTP session, by using a Media-Aware Network Element for aggregating RTP sessions into a single RTP stream for each client (unicast) or group (multicast). Besides, different layer combinations (base layer only, enhancement layer(s) only or base and enhancement layer(s)) are possible. Additionally, distinct packetization modes exist, which differ in whether the SVC data can be interleaved (transmitted in a order different than the intended decoding order), and in the mechanisms used to recover the correct decoding order of the NAL units.

In [39] the analysis of MEDIEVAL's functionalities towards the support of mobile multicast sessions was provided, contributing with a scenario depicting the architecture's cross-layer nature and the showcase of a Personal Broadcasting Service (PBS) session. By PBS, we refer to a video service provided by the mobile operator in which the video content source is the network consumer i.e. any mobile terminal attached to the network that support's the service functions.

Scenario summarization

The scenario (depicted in Figure 12) is as follows, Mike has just started his own e-club channel for broadcasting video. Using this service, he shares with his subscribers the latest news and events in his town, thus sometimes needing to record on the move (represented in section (a) of the figure).

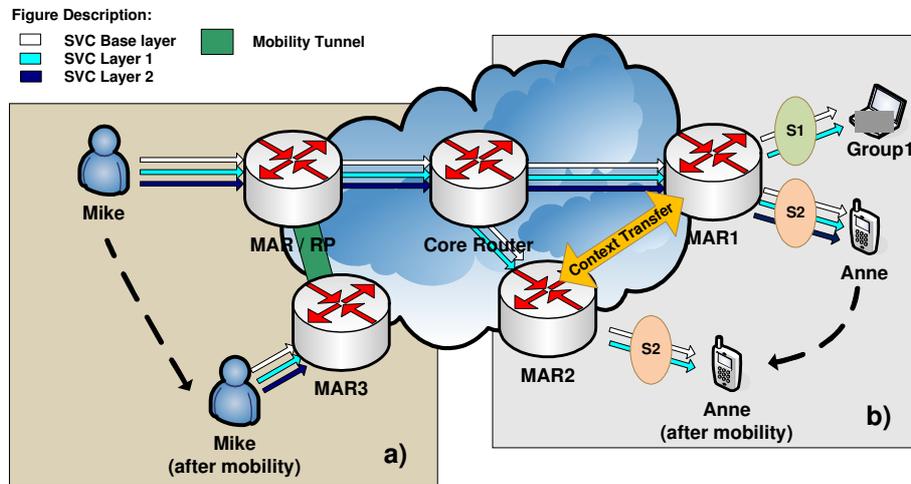


Figure 12: SVC multicast mobility scenario

Anne is one of his subscribers, and has the latest and MEDIEVAL-enabled mobile phone. As such, she wants to take the most of it, watching videos with high level of quality even when moving. The MAR where she started viewing the live video has already some subscribers requesting the same video (stream S1 in the figure), but with inferior level of quality associated to their profiles, either because of their terminal characteristics or due their SLA – service level agreement with the mobile operator. This fact is important in the network decision, as will be seen further in the section. At a later time, due to mobility Anne is now associated to a new MAR, nevertheless the service is not interrupted.

Architectural features

It is important to highlight the interaction between transport-aware entities (Decision Module – DM) and mobility-related ones (e.g. FM). This interface avoids mobility operations towards congested access points, due to a candidate network weighting process, and other intelligent decisions. The referred work proposed the use of a different multicast group for each expected quality (temporal, spatial, SNR) set. This can be seen as a hybrid simulcast-layered solution, splitting the pros and cons of each transport mechanism. We foresee that in real networks the number of deployed layers per video will typically be low (e.g. four), and therefore the replication of information is bearable, and inferior to current simulcast proposals. On the other hand, most terminals will be spared from the SVC decoding effort, having an IP multicast session addressed to it. As such, when Anne starts receiving the video, MAR1 subscribes the missing layers, aggregates them in a single RTP session (represented as S2 in the figure) adapted to her terminal’s needs.

In order to reduce the packet loss and delay during the mobility process a multicast context transfer takes place when Anne moves from MAR1 to MAR2. As the new MAR is informed by MEDIEVAL’s Decision Module that there is limited bandwidth available in its link to the core, it doesn’t subscribe to all the layers corresponding to the expected quality by Anne, aggregating a lower-quality version in a new RTP session to the same multicast destination address. Such mechanism is analogous to the SVC layer drop that may occur at wireless transmission, but takes place before the last hop. This means that, having two versions of the same video, V1 and V2, where V1 has more enhancement layers than V2, at some point in time V1 may actually be delivered with the same quality as V2, by network decision. Regarding the example, when the context transfer takes place between MAR1 and MAR2, the DM, which establishes a mapping between the multicast group being requested and the effective layers to be requested, is responsible for informing the FM

placed at the MAR about the subscriptions to be made for this multicast session. In practice, it leads to MAR2 not joining all the multicast groups (layers) of the video during the congestion period. This same kind of content adaptation may also be done as a consequence of a different trigger, such as QoE level decrease. As soon as MAR2 verifies its ability to support a better (and more bandwidth-demanding) version of the video, informed by DM, it subscribes to the missing channels and aggregates them transparently to Anne's mobile.

3.2.6 DMM and Content Delivery Networks

In the paper [34] we designed a novel mobile network architecture where we merged the DMM functionalities with CDN architectural elements. We further inferred which functionalities are required to address the content distribution issue among CDN caches and the redirection of mobile users in the access part to support mobility-enabled video services.

Our current mobile architecture is implemented on a virtualized testbed using a modified version of PMIPv6 to fulfil DMM requirements (see Section 6.1). The basic DMM functionalities were demonstrated during last 83rd IETF meeting (see Section 3.3.1) and the well known open-source CDN software, OpenCDN², enhanced with the necessary extensions, completes the experimental setup. Thus, the resulting framework opens the door to a range of enhancements for the support of video services/applications, with special focus on the impact of such mobile and distributed architecture on the video quality perceived by the mobile users (QoE).

The submitted work, referring to the integration of the CDN platform in our DMM architecture is as follows. Each Serving or Packet Data Network-Gateway (X-GW) implements the agent for content caching. The algorithms to select the content to be cached are out of scope of this paper, although traditional schemes enhanced with knowledge of mobile users should be investigated. The Request Router and Distribution Manager (RRDM) is a control point in the mobile architecture and the video portal collects information on available content and its current location and it is a service offered by the Mobile Operator.

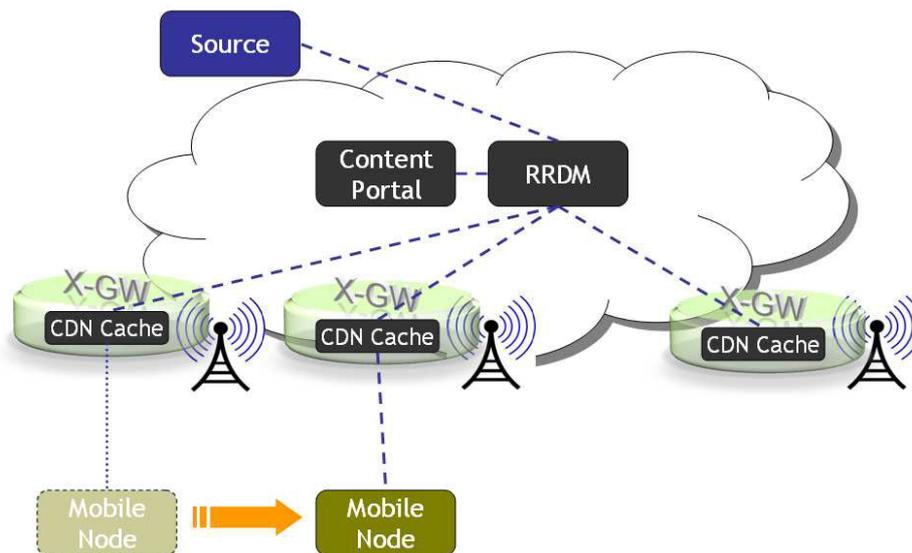


Figure 13: CDN and DMM architectural integration

When a mobile user wants to request a specific content he/she contacts the Video Portal to receive all the info regarding the specific video stream. Upon selection the Video Portal delivers to the mobile device the connection parameters including the node from where to download content. If the content is cached, the node to be contacted is most likely the X-GW, otherwise it will be a node outside the Mobile Operator network.

² OpenCDN Project: <http://labetl.ing.uniroma1.it/opencdn/>

The selection criteria for the target node can take into account several parameters. In [35] the authors proposed a method to select the end points based on wireless medium conditions combined with core network parameters. The approach was proposed for a mobile device roaming in a heterogeneous wireless environment and can be equally applied to a network node performing similar selection procedures.

Upon node selection the mobile device is instructed to download from such a node. During connection time this information is delivered to the mobile device by the Video Portal. During handover and change of the point of attachment to the mobile network the procedure requires further steps. First of all the CDN control node (RRDM) evaluates where the node is connected in terms of wireless medium. Secondly, it needs to evaluate if a better caching node (e.g., X-GW) can be selected. If this is the case the RRDM informs the mobile device that a better cache (in terms of latency, load, response time is available). This procedure is usually called redirection mechanism. In traditional CDN systems DNS based schemes are used. However, in a highly volatile mobile environment the IP bearer conditions can change over time quite rapidly and so do the conditions of the agents caching the content.

Mechanisms to dynamically update a video player with a fresh playlist are currently being standardized. An example is one proposed where the use of the SMIL message updates the mobile device in a simple and transparent manner. That is upon completion of the handover procedures, the mobile device will always download content from the most suitable cache.

We argue that to optimize the video delivery value chain Mobile Operators need to better optimize the way content is accessed. The combination of flat networks (optimal routing, reduced overlay and tunnelling overhead) with intelligent video distribution systems (distributed caching with central control) is a promising path. In addition it has been demonstrated that local caches reduce the transport costs to a great extent and that mobile CDN helps the service providers to address their scalability issues .

3.3 Dissemination and standardization plan and results

In this deliverable we highlight the work done between June 2011 and June 2012 and the plan for next contributions. For each contribution we refer to its relationship with MEDIEVAL work also referring to considerations raised in section 2.1.

3.3.1 IETF WG DMM

The Dynamic and Distributed Mobility Management (referred to as DMM or DDMM) topic started to be unofficially presented and discussed in the IETF around two years ago. Since then, MEDIEVAL partners, especially in the last year, had a very relevant role and presence in all the discussions, co-authoring some problem statement and scenario drafts, as well as solution proposals (completely designed within the framework of the MEDIEVAL project, being actually the main pillars of the mobility subsystem of the project).

The DMM work became official when a new charter item was added to the MEXT WG, aimed at looking at operational considerations for distributed use of Mobile IPv6. Just recently (in March 2012), and based on ongoing discussions where MEDIEVAL has been involved, the IETF has decided to charter down MEXT WG and create a new working group just focused on DMM. Starting from existing work on Requirements and Gap Analysis the WG decided to first get consensus on a requirement list and then to proceed updating the existing documents and adopting them as WG documents. MEDIEVAL partners actively contributed to the initial definition of the requirement list and the plan is to continue contributing/co-authoring these documents.

Different Drafts have been presented during the March 2012 IETF meeting³. Here in the following a summary for each contribution:

- the draft “A PMIPv6-based solution for Distributed Mobility Management”, (draft-bernardos-dmm-pmip-01), explores different PMIP-based DMM solutions both for the fully and the partially distributed approach, focusing in more detail on the latter. The key aspects are the specification of the mobility options included in the signalling and the sequence of operations performed by the entities;
- the draft “Dimensioning considerations for distributed mobility architecture” (draft-demaria-dmm-dimensioning-considerations-00) describes a general method to calculate the costs of deployment for an operator comparing a centralized and a distributed scenario. Main purpose is to provide a model, even if simplified, at least in this first version, that each operator may use to discover the optimal architecture based on traffic observed in the network.

During the same IETF meeting some MEDIEVAL partners (UC3M and ALBLF) also presented a demonstration: “Network based Distributed Mobility Management demo”, in collaboration with InterDigital company. The demo shown a prototype implementation of one partial-DMM solution from draft-bernardos-dmm-pmip-01 using real devices. A poster (shown in B.2) was generated to help during the explanation of the demo, as well as to disseminate MEDIEVAL work.

Current plans go around the following lines: continue contributing to the DMM problem statement [2] , and scenario definition [3] documents, extend and improve the network-based [17] and client-based [18] DMM solutions, as well as looking at the integration with mobile networks. We aim at contributing/co-authoring both the problem statement and scenario documents (which might end up being merged), to submit a gap analysis document, and to have a relevant role (possibly co-authoring) on one of the final standardized solutions.

The work under the umbrella of the IETF will be well coordinated with the project strategy at the 3GPP, to maximize the potential impact of MEDIEVAL work.

3.3.2 IETF WG NETEXT

Within the context of the NETEXT WG, MEDIEVAL members are currently editors of two of the main documents of the working group: the one that defines the logical interface concept [19], which is critical in order to achieve mobility among different physical interfaces in a transparent way to the IP layer; and the solution document to enable flow mobility in PMIPv6 [20]. Both documents are heavily impacted by the concepts and work developed in the project.

Plans for the future include the finalization and publication as RFCs of these two important documents. Another aspect where MEDIEVAL has been successful is in the area of the problem of integrating mobile networks in a PMIPv6 domain. MEDIEVAL is co-authoring the WG document [21] that specifies how to delegate prefixes to a router in a PMIPv6 domain using DHCPv6, so then the router (with attached devices) can roam within the domain. A step beyond that is the full integration of mobile platforms in a PMIPv6 domain [22], in such a way that a mobile user cannot only benefit from transparent mobility support while roaming between fixed points of attachment, but also while roaming between mobile ones. An example scenario of this is Internet access in public transportation systems, where points of attachment are not only deployed in the stations but also within the mobile platforms themselves (e.g., trains, buses, subways, etc.) [24].

³ 83rd IETF Meeting, Paris, March 2012

3.3.3 IETF WG MULTIMOB

MEDIEVAL has significantly contributed so far to two key topics within the MULTIMOB WG: handover optimization [1] and solution to the tunnel convergence problem [25]. Our contributions have been regularly presented and discussed at IETF meetings, and updated considering the received feedback. Through the discussion with various MULTIMOB experts on mailing list, routing optimization solution was designated as WG document, and a revised version reflecting received feedback was presented in IETF 83rd meeting. Since handoff optimization for IP multicast was also designated as a WG document in the meeting, this version was uploaded [25]. We expect both WG documents to be presented in the next IETF plenary meeting.

In IETF 83rd meeting, there was discussion about re-chartering items. Among them, our IP multicast use-cases analysis contribution on DMM, which is a key mobility framework of MEDIEVAL, was presented in IETF 83rd meeting [25]. Although there was no decision to take a WG document of the presented item, mainly because DMM concept is not fully defined, it was regarded as meaningful discussion to identify the application of the base solution made by MULTIMOB WG on generic DMM architecture. We plan to continuously update this document for next IETF meetings while we closely keep monitoring the progress and milestones of IETF DMM WG with several MEDIEVAL partners.

3.3.4 IETF WG MIF

The planned work on the Connection Manager inherits from the discussion in the IETF MIF working group. MEDIEVAL will investigate issues related to source address selection, use of multiple DHCP and DNS entries not anymore based on a single physical interface. The implementation work executed in the project will serve as experimental platform and the results will be fed back to the IETF community hopefully as open source code.

3.3.5 IEEE 802.21

The extensive use of the IEEE 802.21 standard within the MEDIEVAL architecture is leading to constant improvements of the standards itself and the definition of new supported features. MEDIEVAL is providing contributions to the IEEE 802.21 WG and its different amendments regarding several aspects. The latest contribution related to the MEDIEVAL mobility architecture is related to security keys exchange mechanisms to speed up the handover phase. A first draft document was produced and submitted to the IEEE 802.21a WG in September [57]. The proposal was enforced at last IEEE 802.21c meeting in November [58]. A further contribution, providing the follow-up version according to the comment received [59], has been produced in March [60]. Finally the proposal has been accepted to form part of the IEEE 802.21c in the May 2012 meeting.

This latest contribution addresses the required extensions to support proactive pull key distribution; the goal is to allow MN to establish a key with target network elements (tPoA and tPoS) without communicating with home network. This work is relevant for MEDIEVAL since it can be applied to DMM scenarios. The solution provides mechanisms to authenticate the MN on the target PoS previous to handover with the constraint of using only static parameters such as keys and IDs. These parameters are transferred between sPoS and tPoS. No dynamic parameters such as cipher suite and SAID are transferred.

It is worth to mention that this contribution is very important for the IEEE 802.21c draft since it defines the basic primitives to be used by this standard. This importance has been acknowledged by accepting the contribution into the main IEEE 802.21c draft document.

3.3.6 3GPP

3.3.6.1 User plane congestion (UPCON) management topic

An important topic addressed in MEDIEVAL concerns the user plane congestion management and notification. MEDIEVAL initially contributed to build up a Study Item in 3GPP on this by providing to 3GPP SA1 [16] an exemplary use-case: upon congestion in a cell/NB/RA, the operator does not want to discard all traffic of all users but to preserve premium users (example: to throttle p2p traffic of non-premium users).

Further advances have now brought to the editing of a specific Technical Report, 3GPP TR 22.805, whose current version (2012-05) is V0.3.1.

This TR considers scenarios and use cases where high usage levels lead to user plane traffic congestion in the RAN, and proposes requirements for handling user plane traffic when RAN congestion occurs. The aim is to make efficient use of available resources to increase the potential number of active users while maintaining the user experience.

Scenarios that are considered include handling of user plane traffic when RAN congestion occurs based on:

- the subscription of the user;
- the type of application;
- the type of content.

Besides use-cases definition, under discussion are also possible available solutions.

The solution proposed within MEDIEVAL is based on the PCRF (Policy and Charging Rules Function) that receives notification from the RAN and then performs proper restrictions.

This solution has not yet been accepted by 3GPP with the main reason that it doesn't preserve the original design of congestion handling made in 3GPP. Congestion is now handled in the radio access in a fast way. The notification sent to the core network is in the opposite direction and the risk could be that this solution is not fast enough; moreover, to prevent the network to become unstable, the operator must set up hysteresis and this can become a difficult task.

A further technical problem is that with the current model if you want to throttle only p2p traffic there is no way from the network to distinguish traffic behind the 5-tuple. A possible work to be done in this area is then to extend the scope of the existing 3GPP Traffic Flow Template (TFTs) behind the 5-tuple and to update them to current requirements of Internet traffic.

3.3.6.2 DMM in 3GPP

Mobility management in the 3GPP system has been traditionally handled relying on a centralized mobility anchor, that is the GGSN for the GPRS system, and the PDN Gateway (PGW) for the Evolved Packet Core (EPC). Nonetheless, the original assumption that the mobility anchor was not supposed to change during a working session was somehow weakened starting from 3GPP Rel-10, with the specification of the Selected IP Traffic Offload (SIPTO) function, that allows the mobile operator to re-allocate the GGSN, or PGW, on the fly in order for the mobile terminal to be served by a gateway closer to its current point of attachment. This provides the means for the operator to optimize routing of traffic in the mobile packet core, and may potentially improve the user experience experimented by the customer.

A similar concept was introduced also for femtocell deployments, with the specification of support for Local IP Access (LIPA) for the H(e)NB subsystem. The LIPA service allows the mobile customer to be served by a Local Gateway (L-GW) collocated with the H(e)NB (also known as femto access point), e.g. to access services available in the local network, such as printers or media servers. In 3GPP Rel-12 this functionality is being further enhanced adding the capability to provide the LIPA service relying on a standalone L-GW, as a

way to enable mobility support inside the local network for femto deployments involving multiple H(e)NBs (e.g. typical situation in enterprise scenarios).

Finally, 3GPP is also defining the so called “SIPTO at the local network” service, that is an extension of Rel-10 SIPTO where the serving gateway is not allocated in the macro network (typically above the RNC) but inside the H(e)NB subsystem. This allows to further optimize traffic routing in the operator’s network, since a customer that was accessing a certain APN through a GGSN, or PGW, in the mobile packet core while walking in a street can continue to access that very same APN through a L-GW in the H(e)NB subsystem when inside a corporate network with femto coverage. Again, as with Rel-10 SIPTO, this requires a re-allocation of the serving gateway, which, based on the current specifications, implies a change of IP address, and hence may cause disruption of on-going sessions. These impacts can be mitigated triggering the gateway re-allocation when the UE moves to idle-state, but this approach surely limits the effectiveness of the solution, considering that nowadays many smartphones exchange traffic very frequently, and hence may remain in active state for a very long time.

The usage of Distributed Mobility Management (DMM) mechanisms, though still not foreseen in any of the existing 3GPP work items, could be a way to overcome the limitations of the existing SIPTO (in macro and/or local network) solutions, enabling the re-allocation of the serving gateway with no impacts on on-going sessions even when the mobile terminal is in active state, and is sending and/or receiving traffic. There is therefore very close relation between DMM and ongoing 3GPP activities on SIPTO and/or LIPA. It is expected that the evolution of these features in future 3GPP releases might incorporate DMM as a way to support seamless gateway reallocation for active state and/or idle state terminals.

4 Mobility Architecture

We briefly present the MEDIEVAL mobility architecture, as already deeply described in D4.1 [9] and D4.2 [10], before looking at the detailed specification for each module of the Mobility Subsystem.

The MEDIEVAL mobility architecture follows a DMM approach, where mobility is anchored at the very edge of the network, it adopts an hybrid approach, where network-based mobility management solutions (i.e., PMIPv6-alike) are used whenever possible, and client-based solutions are used otherwise (e.g., between different domains), and due to the video-centric nature of the project, multicast traffic delivery and content distribution aspects are fully supported and integrated in the mobility management solution.

The access network is organized in Localized Mobility Domains (LMDs) in which a network-based scheme is applied. Users are expected to be most of the time roaming within a single LMD, but, for those cases where this is not possible (e.g., roaming to a network owned by a different operator or running a different mobility support scheme), a host-based DMM approach is followed. In order allow mobile nodes to simultaneously have sessions managed by a network-based (“PMIPv6 alike”) approach and a host-based (“MIPv6 alike”) approach, we introduce a novel architectural element called Mobile Access Router (MAR) integrating both approaches .

A MAR is a network entity implementing all the functionalities of its counterparts in the standard mobility protocols (MIPv6 and PMIPv6), so it is able to play the role of plain access router, home agent, local mobility anchor and mobile access gateway on a per address basis.

MEDIEVAL envisions the possibility to support a MAR moving within an LMD: this special entity is called Mobile MAR (mMAR) and it is supposed to gain connectivity from a fixed MAR, and to be able to move from MAR to MAR, being deployed, for instance, in vehicles as cars, buses and trains. Current specifications for the mMAR are at a earlier stage than those for the other mobility modules, being the “NEMO for DMM” topic under evaluation to be presented at the standards community. The architecture of the Mobility subsystem is shown in the following figure and it is composed of three main components: Connection Manager (CM), Flow Manager (FM) and Mobility Engine (ME).

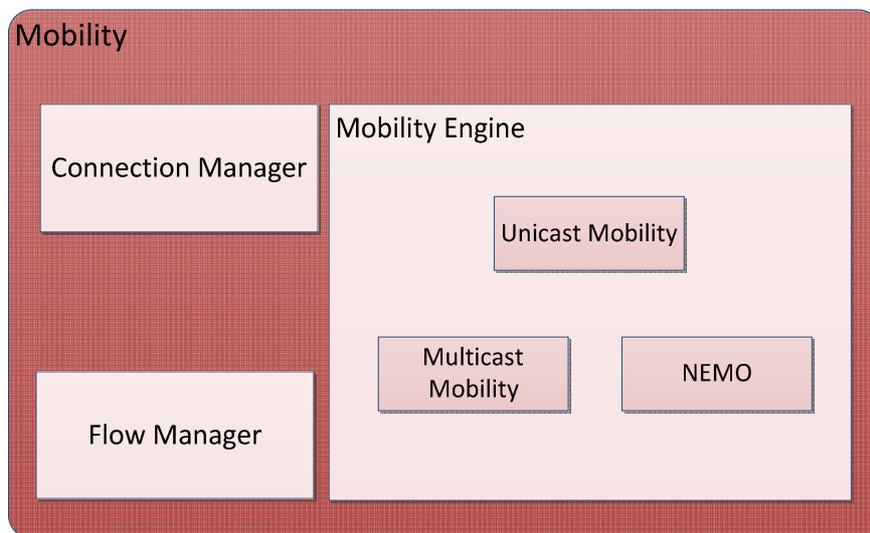


Figure 14: Functional architecture of the Mobility subsystem

Figure 15, Figure 16 and Figure 17 depict the subsystem’s modules mapping into devices – MT, MAR and mMAR, respectively.

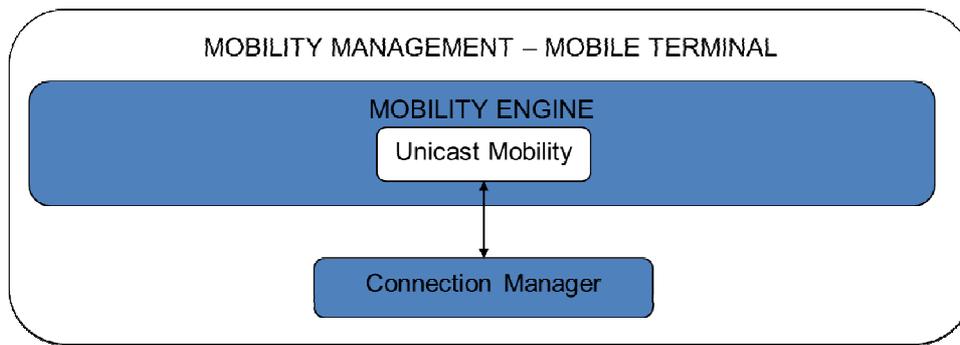


Figure 15: Mobility modules in the MT

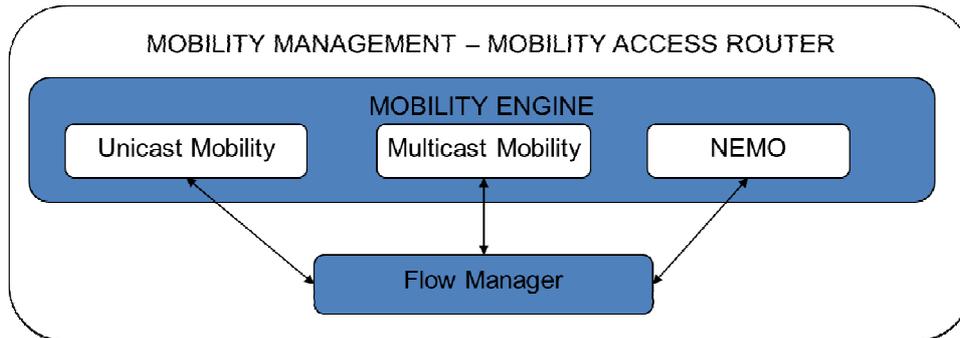


Figure 16: Mobility modules in the MAR

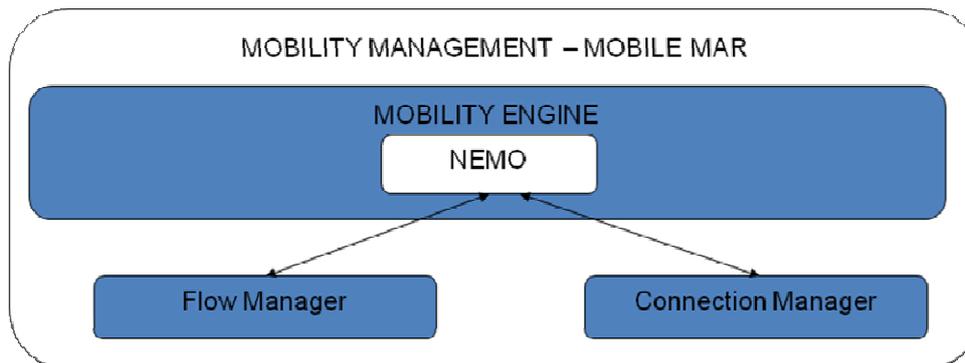


Figure 17: Mobility modules in the mMAR

In the next sections specification and operational details for each module are presented.

4.1 Modules

This section presents the detailed specification of the modules pertaining to the Mobility sub-system from MEDIEVAL architecture. As such, the focus will be on describing for each module the following: provided functionality, state machine (presenting sub-modules and their operation), and interactions with external sub-systems.

4.1.1 Flow Manager

The Flow Manager (FM) resides in the MAR and in the mMAR. It has several functions, but the most important is the management of data flows. Simple tasks, such as detecting, adding, removing, enforcing, moving and adapting data flows seem little enough when apart, but when combined, they require a complex management structure. Adding to the complexity is the responsibility of main path of communication

between the Mobility Subsystem and external subsystems such as Wireless Access, Transport Optimization and Video Service Control, making the FM an important centrepiece in the whole architecture.

Although an architectural centrepiece, the FM shares the core of the Mobility subsystem with the mobility engines, and therefore, together, they share the task of providing proper DMM functionalities. Thus a close interaction is expected between the FM and the mobility engines (see 4.1.3). The mobility engines manage to bring to the fore their IP mobility execution mechanisms that ensure session and IP continuity as well as their anchoring properties. The FM will contribute with its broader information on finer IP flow granularity and cross-layer information regarding the IP flows. As a result the FM can enforce mobility (through the mobility engines) both proactively and in consequence of the changing conditions of the network, and thus provide the user with a better QoE.

The main focus of the FM within the mobility framework is to keep track of data flows that traverse him and, either according to events from entities (external or internal), or implemented policies, manage the data flows to provide the mobile user with the best possible service. To this purpose the FM leverages on two advantages: i) the tight relationship it has with the remaining MEDIEVAL mobility components; ii) the FM's central position on the MAR where it has a good perspective of both the access network as well as the infrastructure near the access, enabling it to gather information from both perspectives to provide better decisions.

In Figure 18, the FM architecture is depicted, divided in four main functional groups:

- Engine & Flow Database:

This is the core of the FM, the Engine contains the logic and intelligence to apply the correct procedures based on received events. It works closely with the Flow database where the flow information is stored as well as the current state of the FM. The Engine works as an intelligent entity, with the following capabilities: receipt of events from interfaces such as the one with the L2.5 Abstraction Layer module (component of the Wireless Access subsystem) and the one with the Decision Module (Transport Optimization); executing any necessary operations related to the MNs association with the network; and returning output through the appropriate interfaces.

- Internal Interfaces:

These interfaces include communication with the mobility engines (unicast and multicast) to trigger any necessary mobility, and also to enable communication with other FMs, CMs and the MIIS service through IEEE 802.21. This last communication is mediated through the L2.5 Abstraction Layer module.

- External Interfaces:

External interfaces are interfaces with other subsystems: with Wireless Access to receive input from the access technologies as well as to execute some commands on these technologies (through the use of MIH 802.21); and with Transport Optimization, the DM module to get a handover target decision, and the Cross-Layer Optimizer - XLO (to inform the FM and therefore the Mobility Subsystem when end-to-end congestion is affecting a flow)

This includes: the OS Interface, through which flow routing and monitoring functions are implemented and enforced; the virtual terminal and the web interface which allow to configure the FMs global parameters, change them on-the-fly and view the current status of the internal structures and data.

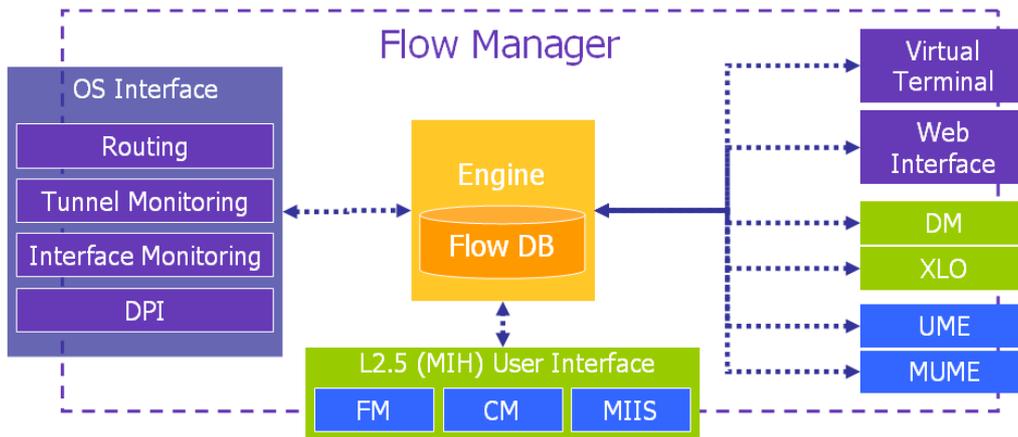


Figure 18: FM internal architecture and interfaces.

4.1.1.1 Operations

As we can see from Figure 18 the architecture of the FM is essentially built around the Engine. The Engine’s operation is quite simple: after an initial period of configuration, additional parameter discovery and routine checks, it waits for events from its input queue and reacts upon those events. Once an event is received, appropriate action is taken based on its source (internal, external and local). Upon completion of the procedure, state change or general timeout for the attempted actions, the FM resumes its waiting state.

The Engine being heavily dependent on feeding from events, is constantly polling its event queue that is filled from the messages that come from the FMs interfaces. Events generated by all interfaces can be a large assortment of things, as an example, an event can be a new configuration parameter coming from the web interface; a data flow expiration trigger from the Flow DB; a message received from the Decision Manager (DM); a trigger from the OS kernel routing table or tunnel monitor; etc.

4.1.2 Connection Manager

The Connection Manager (CM) resides in the User Equipment (UE) and is responsible to manage all connectivity actions required in the terminal side. The CM is a MIH user that interacts with the wireless access networks using 802.21 primitives. It is also the FM counterpart, using the 802.21 protocol as well.

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. These policies can be provisioned on the CM by multiple sources, namely CM GUIs, applications and operators (e.g., ANDSF).

The CM receives monitoring information, in order to improve the QoE experienced by the users. These monitoring data can be related to OS parameters (CPU, battery, etc.), network interfaces (e.g. signal strengths, up/down interfaces, etc.), end-to-end metrics (delay, loss, etc.), or any other relevant data.

Figure 19 depicts the general architecture of the CM proposed for the MEDIEVAL project. This architecture includes the whole functionality, even though some parts could not be implemented, or implemented partially, under the MEDIEVAL scope.

This Figure shows boxes with different grey levels. The inner box (dark grey) comprises the whole set of components considered part of the CM (Connection Manager) entity, while the interim box (grey) comprises the User Equipment (UE), the terminal. The outer box (light grey) comprises the entire wireless mobile environment as a whole, including operators, providers and all other players potentially involved.

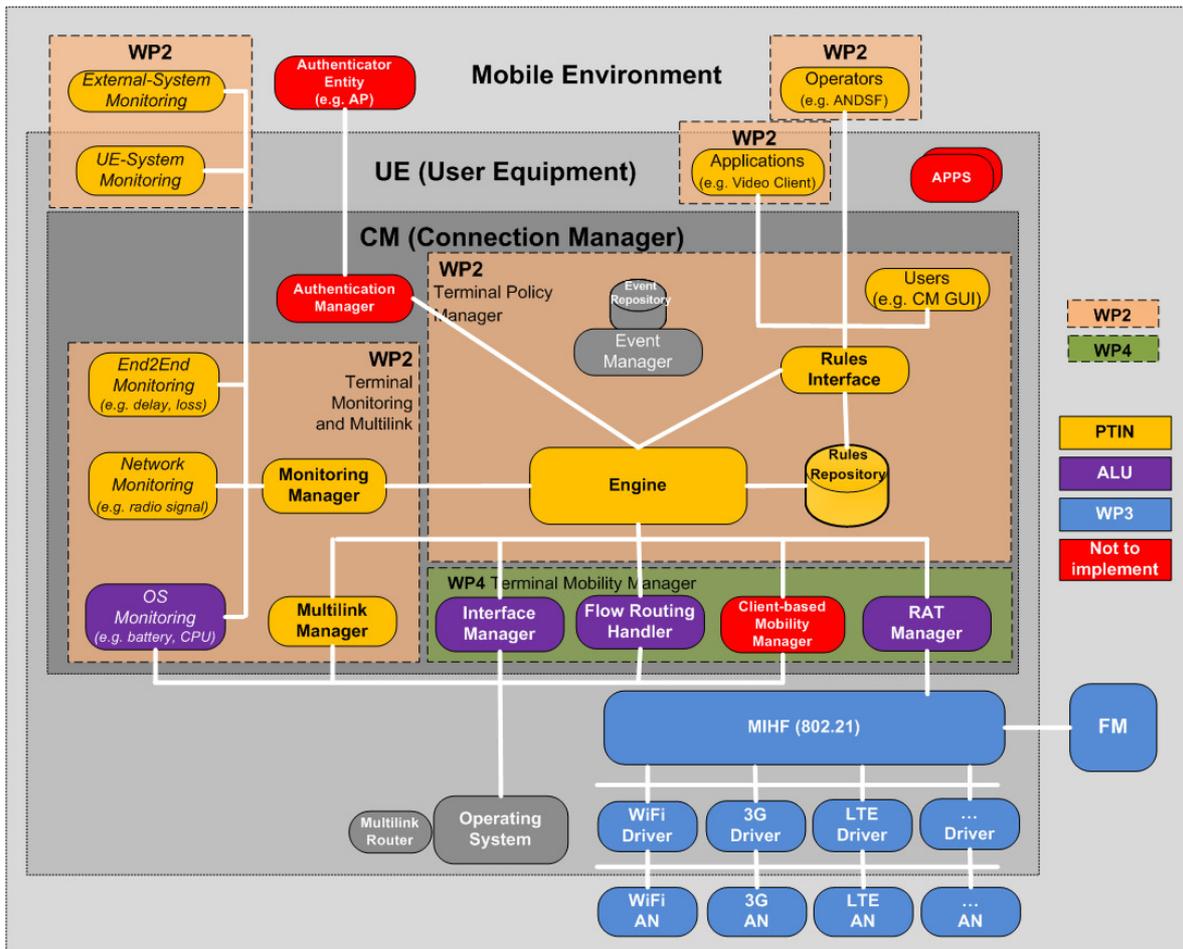


Figure 19: CM Architecture

The orange and green dashed boxes indicate whether the components and APIs inside are considered under the scope of WP2 or WP4, respectively, for the MEDIEVAL project. Also, round-squared boxes of multiple colours (see the legend) are used to identify components not considered for implementation by the project (red) and components to be implemented under different WPs (orange, blue and green).

The CM is basically compounded by three parts: the Terminal Policy Manager, the Terminal Monitoring and Multilink (both under the WP2 scope) and the Terminal Mobility Manager (under the WP4 scope).

The Terminal Policy Manager has a central component: the Engine. The Engine is responsible to handle events coming from multiple sources, e.g. OSs, RATs, interfaces, users, applications, or monitoring, among others. The Rules Repository is the component responsible to store the set of rules to be applied. Those rules will be interpreted by the Engine. The rules stored in the Rules Repository can have multiple sources: the end user (e.g. GUI), the operator (e.g. ANDSF) and applications (e.g. LU40). The Rules Interface component is responsible to adapt GUIs (e.g. function), interfaces (e.g. S14) and APIs (e.g. WS, REST) coming from these sources to a common rule set format.

The Terminal Monitoring and Multilink is responsible to manage the monitoring of the CM, allowing external monitoring sources to reach the CM. The Monitoring Manager is able to feed the Engine with events resulting from notifications, e.g. thresholds, which can influence the evaluation of a rule and, therefore, the policies to apply. The Monitoring Manager can also provide monitoring data to applications through an API. The Multilink Manager component is responsible to implement Multilink policies on the CM according to the existing rules.

Finally, the Terminal Mobility Manager is responsible to interact with low layer resources, in order to implement mobility, routing and flow handling. For that purpose, there are specific components such as the Interface Manager, the Flow Routing Handler and the RAT Manager (to deal with the 802.21 protocol). All

these components are able to feed the Engine with triggers. At the same time, the Engine is able to enforce commands on these components.

Some features such as authentication, security and client-based mobility protocols (e.g. DSMIPv6), despite important, are out of scope of the MEDIEVAL project.

4.1.3 Unicast Mobility Engine

The Unicast Mobility Engine (UME) is the module in charge of performing the unicast IP mobility operations and signalling following the DMM paradigm. It is implemented in the MAR for the network-based part of mobility management and in the terminal for the host-based part. It is triggered by the FM (in case of network-based mobility management) or by the CM (for the host-based) whenever the MN must maintain a given IP address outside the subnet where the address is topologically valid. The following specifications are intended to address IPv6 only, while for dual-stack handling we refer to the mechanisms described in Section 3.2.1.

The main operations that the MAR's UME supports are:

1. Maintaining a database with the mobility sessions of the MNs attached to the MAR. This data structure is inherited and extended from that defined in RFC 5213 [42] and RFC 6275 [45] as Binding Cache (BC);
2. Sending and receiving the messages defined in RFC 5213 as Proxy Binding Update and Proxy Binding Acknowledgment and the messages defined in RFC 6275 as Binding Update and Binding Acknowledgment;
3. Handling Neighbour Discovery messages as Router Solicitation, Router Advertisement, Neighbour Solicitation, Neighbour Advertisement to properly communicate with the MN at link-local level;
4. Managing bidirectional IP tunnels and installing the appropriate packet forwarding rules to build the routing topology for the MNs;
5. API to communicate with the FM.

The main operations that the MN's UME supports are:

1. Dealing with multiple home agents;
2. Sending the messages defined in RFC 6275 as Binding Update and receiving the Binding Acknowledgment messages;
3. Maintaining a conceptual data structure listing the binding update messages sent to the MARs. This data structure is inherited from that defined in RFC 6275 as Binding Update List (BUL);
4. Managing bidirectional IP tunnels and installing the appropriate packet forwarding rules to address the correct MAR associated with a given HoA;
5. API to communicate with the CM.

4.1.3.1 Data structures and messages

The Binding Cache is a database stored by each MAR which entries (BCEs) contain as primary parameters the association between the MN-ID and the Home Network Prefixes (HNPs) allocated for the MN. The complete list of fields belonging to a BCE is defined in RFC 6275 and RFC 5213. However, the MEDIEVAL DMM protocol extends the BCE with an additional field in order to accommodate along with a HNP (or a set) assigned by other MARs also the MAR that assigned it. This piece of information is necessary to populate MAR's routing table consistently. Hence the ancillary field, called Anchor MAR

Information and which might appear in multiple instances, should be structured to include the following parameters:

1. A single (or a set of) HNP(s), assigned by the A-MAR;
2. The Proxy CoA for that (those) HNPs. The PCoA is the address of the MAR that assigned that (those) HNP(s);
3. The network interface, i.e., the tunnel, that is used to transfer packets with that (one of those) HNP(s).

Also, the lookup of an entry in the BC should be based on the MN-ID key for network-based mobility and based on HNP key for client-based mobility: this feature is not specified in RFC 521, but it is required for the UME's state machine.

The Binding Update List is a database stored by the MN which entries (BULEs) keep track of the BU sent to the home agent(s). A BULE contains as primary parameters the association between a HoA and the address of the MAR that assigned the prefix used to generate that HoA. A BULE is created for every BU sent by the mobile node for which the lifetime has not yet expired. However, for multiple BUs sent to the same destination address, the BUL contains only the most recent BU. Further details can be found in RFC 6275.

The mobility messages exchanged for either the network-based or client-based mobility support are formed by appending a list of mobility options to a mobility header: this is the payload of an IPv6 packet. The format of a mobility header and related options for PBU and PBA messages are defined in RFC 5213, while the corresponding formats for BU and BA messages are defined in RFC 6275. No extension are defined in these specification, except for the rationale behind the appending and filling of the mobility options.

4.1.3.2 Operations

UME's operations in a MAR are triggered upon receiving specific messages, which produces a determined sequence of events. In these specifications we target the working mechanism typical of a successful procedure, while the reader can refer to RFC 6275 and RFC 5213 for treating exceptions and errors handling.

In Figure 20 are depicted the messages accepted by the UME and the main operations they yield:

1. **Router Solicitation.** This message is generated by the MN and sent in broadcast upon establishing the wireless link to discover the router and the prefix(es) to be used to configure an address on the link. The UME extracts the layer 2 identifier from the request and retrieves an MN-ID and the associated HNP(s) from an AAA policy store (the details for this operation are out of scope in MEDIEVAL, but the MN-ID must be unique in the whole domain). These parameters are stored following the RFC 5213 protocol's rule in a BCE for the MN, which is advertised the prefix(es) through a Router Advertisement message. The terminal is now able to configure a valid address and start data communications.
2. **FM_UME_ActivateMobility.request.** This message is generated by the FM co-located with the UME to notify that the MN that has just attached comes from a handover. The primitive carries the MN-ID and the previous A-MAR(s) address(es). The UME looks for the MN-ID in the BC and retrieves the BCE for that MN, and it prepares a PBU message to be sent to the appropriate A-MAR(s) including as main parameter the MN-ID. The PBU is created according to RFC 5213 specifications, except for the HNP mobility option, which must be set blank.
3. **PBU.** This message is generated by the serving MAR and received by an anchor MAR, which extracts as most important parameters the source address of the message and the MN-ID. After retrieving the BCE corresponding to the MN-ID, it sets the source address (i.e., the serving MAR's address in the DMM domain) as Proxy-CoA in the BCE, and establishes a tunnel (if not already present) towards it. The A-MAR continues the procedure by preparing a PBA message filled with the P-CoA as destination address, the same MN-ID and the prefix(es) it allocated for the MN when it was attached to it. The update procedure is over when the A-MAR installs the routes to forward the

packets containing these latter HNP(s) through the tunnel. The PBA just prepared can now be sent to the S-MAR.

4. **BU.** This message is generated by MN and received by an anchor MAR, which extracts as most important parameters the source address of the message (the CoA), and home address (HoA) that the MN wants to keep reachable. After retrieving the corresponding BCE, it sets the source address as CoA in the BCE, and establishes a tunnel (if not already present) towards it. The A-MAR continues the procedure by preparing a BA message filled with the CoA as destination address, and the prefix(es) it allocated for the MN when it was attached to it. The update procedure is over when the A-MAR installs the routes to forward the packets containing these latter HNP(s) through the tunnel. The BA just prepared can now be sent to the MN.
5. **PBA.** This message is generated by an A-MAR to acknowledge the status of the mobility update towards the S-MAR. If the outcome is positive, the message contains the HNP(s) that the A-MAR is anchoring for the MN. Hence the S-MAR establishes a tunnel towards the A-MAR (note that at this stage the tunnel is bi-directional) and fills the Anchor MAR Information field of the MN’s BCE with the just received HNP(s), the tunnel ID and A-MAR address. The update is over when the S-MAR installs routes to forward in uplink packets with the desired HNP(s) as source through the tunnel. At last, the UME notifies the FM the status of the operation, sending a *FM_UME_ActivateMobility.response* message.

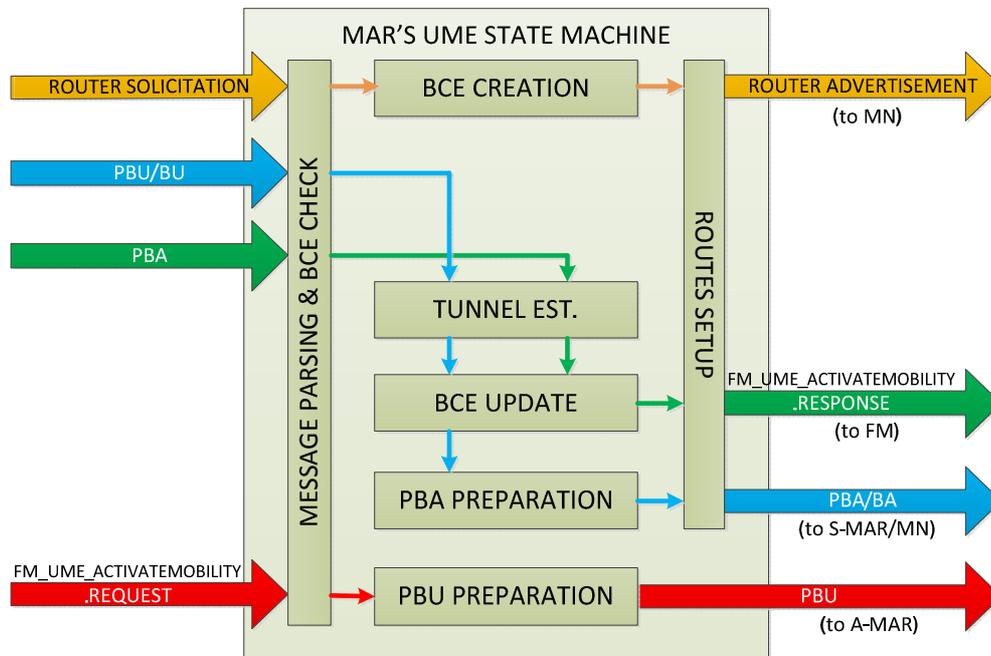


Figure 20: MAR’s UME state machine

Similarly to what described above, UME’s operations in a MN are triggered upon receiving specific messages, which are slightly different from the MAR’s UME case. Again, in these specifications we target the working mechanism typical of a successful procedure, omitting the exceptions and error handling.

The messages accepted by the MN’s UME are:

1. **CM_UME_ActivateMobility.request.** This is message is generated by the CM co-located with the MN to notify that the MN has just configured a new CoA. The UME prepares as many BU messages as the number of A-MARs. The BU for each A-MAR contains the CoA formed with the prefix assigned by that A-MAR.
2. **BA.** This message is generated by an A-MAR to acknowledge the status of the mobility update towards the MN. If the outcome is positive, the MN establishes a tunnel towards the A-MAR (note

that at this stage the tunnel is bi-directional). The update is over when the MN installs routes to send uplink packets with the desired HNP(s) as source through the tunnel. At last, the UME notifies the CM the status of the operation, sending a *CM_UME_ActivateMobility.response* message.

4.1.4 Multicast Mobility Engine

Residing in the MAR, MUME manages the IP mobility support for the multicast flows, having its operation harmonized with MEDIEVAL's DMM architecture. For doing so, it effectively depends on four main functions: i) a multicast group management function, ii) a multicast routing function, iii) a mobility management function and iv) a context transfer function. The first function refers to the multicast group management operations and information storage, realized using mostly MLDv2 router-side functions (i.e. by means of MLDv2 Queries) with the mobile hosts. The multicast routing function corresponds to the multicast routing protocol stack of the node, which in the considered scenarios will be PIM family protocols. As for the mobility management function, it resembles the mobility protocol stack, which corresponds to different functionality depending on the role of the multicast host: for multicast sources, the mobility solution will be an adaptation of the unicast mobility one, whilst for listeners; the last of the referred functions is used. That way, the context transfer function is responsible for pre-emptively modifying the multicast tree in cases of multicast listener's mobility, avoiding the problems involved in transporting multicast traffic in the mobility tunnel.

The main operations supported by MUME are:

- Multicast explicit tracking function: MUME keeps per-MN information regarding their multicast subscriptions.
- Multicast context transfer: by making use of the explicit tracking function, MUME stores always up-to-date subscription information, which is the basis for correct responses provided by the queried (previous MAR's) MUME, and allows tunnel-free multicast mobility – not needing MIPv6-related stack.
- Tunnel-based listener mobility: for particular scenarios, the network operator may wish to take advantage of the DMM mobility tunnels used for unicast traffic. For such cases, MUME allows the setup of the MRIB entries based on the tunnel's endpoints addresses.
- Multicast source mobility: MUME enables transparent source mobility, achieved by pre-emptive dissemination and configuration of the RP address (which acts simultaneously as the mobility tunnel endpoint) at each MAR to which the mobile source moves. The MAR assigned as RP acts analogously to the A-MAR in unicast mobility operation. The referred pre-emptiveness is consequence of enhancements to regular MIH operation, speeding up the multicast mobility process.

4.1.4.1 Data structures and messages

MUME decisions are based on its Multicast Mobility Database (MMD), which enables explicit tracking and stores entries with the following information

- 1 – Mobile Terminal ID (relevant for FM);
- 2 – Mobile Terminal Address (required for operating the MR);
- 3 – Multicast subscriptions (aligned with the structure of MLD multicast information);

Besides, it holds a counter structure for the number of listeners per IP multicast channel, allowing it to identify when a terminal is the last subscriber of a group. This information is in particular essential for proper multicast context transfer operation.

MUME module core operation is the control of MR. The involved interactions with MR, and with other MUMEs may be split as client-based and server-based functionality. As a client, it will:

- Request multicast context transfer from other MUME (using MulticastContextTransfer.request). Request multicast tree updates to the local MR: joining of multicast trees (in case of MN multicast service initiation or arrival due to mobility – using MUME_MR_Join.request) and departure of multicast trees (in case MN moved to another MAR – using MUME_MR_Leave.request).
- Request tunnel based multicast mobility establishment through another MUME (using MUME_MUME_ExtendedProxyBinding.update).

As a server, it will:

- Transport multicast context transfer by replying to the requesting MUME (using MUME_MUME_MulticastContextTransfer.response).
- Complete mobility tunnel establishment for multicast content receival (MUME_MUME_ExtendedProxyBinding.acknowledge).

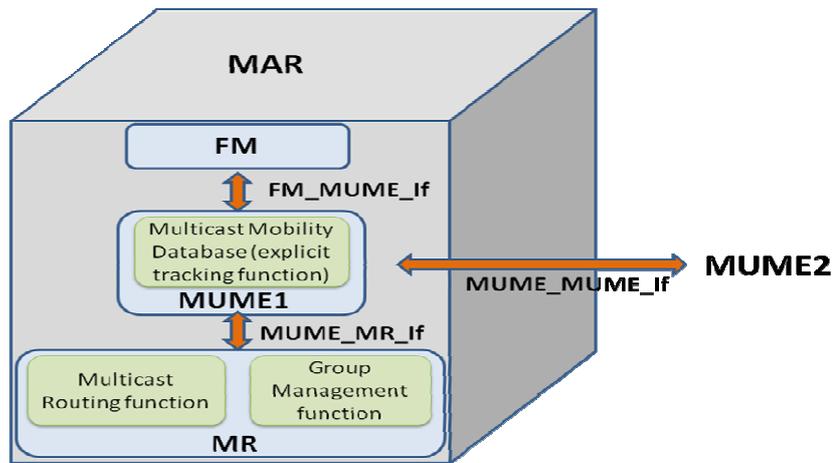


Figure 21: MUME internal structure and interaction with relevant modules

4.1.5 NEMO mobility engine

The NEMO mobility engine (also referred as DMM NEMO module) resides in the mMAR, and it resembles the behaviour of the MN’s UME, being the mMAR’s mobility operations similar to client-based distributed mobility management. However, the NEMO also provides IP access to MNs, offering mobility support to those nodes in a network-based manner. For these reasons, the NEMO mobility engine can be considered a hybrid UME, derived from those installed in the MAR and in the MN. Therefore, an mMAR should support:

1. maintaining a database with the mobility sessions of the MNs attached to the mMAR. This data structure is inherited and extended from that defined in RFC 5213 [42] and RFC 6275 [45] as Binding Cache (BC);
2. maintaining a conceptual data structure listing the binding update messages sent to the MARs. This data structure is inherited from that defined in RFC 6275 as Binding Update List (BUL);
3. dealing with multiple home agents (MARs);
4. sending and receiving the messages defined in RFC 5213 as Proxy Binding Update and Proxy Binding Acknowledgment and the messages defined in RFC 6275 as Binding Update and Binding Acknowledgment;

5. handling neighbour discovery messages as Router Solicitation, Router Advertisement, Neighbour Solicitation, Neighbour Advertisement to properly communicate with the MN and the MAR at link-local level;
6. managing bidirectional IP tunnels with the MARs and installing the appropriate packet forwarding rules to build the routing topology for itself and the MNs;
7. API to communicate with the CM and the FM.

As mentioned in Section 4, the operations of the DMM NEMO engine are not at an as mature status as those for the MAR's and MN's UME. However, there is an ongoing activity to define the DMM NEMO module operations based on the IETF drafts [22] and [23].

4.1.5.1 mMAR's mobility management

The mMAR handles its own mobility by means of the client-based part of the DMM NEMO module. A mMAR obtains a topologically correct prefix from the MAR it is currently attached to (S-MAR), and it configures an IP address from that prefix. When the mMAR changes PoA, with the support provided by the 802.21 infrastructure, it establishes a link with a new S-MAR and configures a new IP address using the prefix advertised to it. This new address is communicated as CoA to its A-MAR(s) by means of a BU message. The A-MAR(s) reply with a BA and a bidirectional tunnel is set up. In this way, the mMAR maintains active and routable the addresses configured in the previously visited access network(s).

4.1.5.2 On-board MN's mobility management

When the mMAR handles mobility for the MNs attached to it, the network-based part of the NEMO DMM module is used. On-board MNs are registered at the S-MAR by means of a PBU sent by the mMAR with a NEMO flag. The MAR assigns a prefix to the MN and acknowledges the operation with a PBA. The prefix is finally advertised by the mMAR to the MN. In this phase the mMAR acts as MAG registering the MN at the LMA and subsequently advertising the assigned prefix to the MN. When a new MAR's network is joined, the mMAR is in charge of informing the MAR about the MNs connected to the NEMO. This is required to allow the MNs to send/receive traffic with a prefix topologically correct at the new MAR, hence to benefit from traversing the closest MAR. This is obtained by the transmission of a PBU from the mMAR to the MAR per each MN attached.

4.1.6 Media Independent Information Server

The Media Independent Information Server (MIIS) is an information repository server from which a MIH-User, residing in the MN or in the network, discovers and obtains network information within a geographical area to facilitate network selection and handovers. The objective is to acquire a global view of all the heterogeneous networks relevant to the MN in the area to facilitate seamless roaming across these networks. MIIS in the network is generally a PoS-non-PoA, meaning that the entity is a specific server within the operator's core.

The information provided by the MIIS is composed by Information Elements (IEs) organized in containers reflecting a network's topology. IEs provide details about the operator, network and Points of Attachment. IEs can be represented in either binary form, allowing TLV queries, or using a Resource Description Framework (RDF) that is queryable through the SPARQL Protocol and RDF Query Language (SPARQL). The base MIIS also provides filtering mechanisms, allowing the definition of a maximum query response size and prioritizing some IEs over others.

The MIIS has the capability of being queried by the MN through the usage of a combination of MIH_Get_Information.request/response primitives, exchanged by the Connection Manager at the MN and the Information Server at the MIIS network entity. By using this query method, the Connection Manager can use the information obtained by the MIIS to:

- a. analyze handover candidate alternatives from interfaces that are inactive (i.e., the MIIS allows the provision of information about specific access link technologies, via other interfaces);

- b. obtain configuration information about link layer attachment procedures, in order to pre-configure necessary parameters for link layer handover.

Moreover, within the MEDIEVAL project we also consider the usage of the information made available by the MIIS by itself, by making usage of the MIIS-generated `MIH_Push_Information.indication` sent towards the MN. The usage of this primitive explores network-initiated informational triggers, where the network proactively sends information towards the MN, with the aim of providing configuration parameters on nearby network handover candidate points. In this way, the MN can be alerted to the existence of nearby PoAs of other access technologies, when the MN by itself wouldn't be able to detect them (e.g., due to the respective link layer interface being off for energy conservation).

Moreover, the usage of these network-initiated informational events reflects the provision of discovery and network policies defined in other mechanisms standardized in other bodies, such as the Access Network Discovery and Selection Function (ANDSF) of the 3GPP's EPC [48]. The ANDSF has the ability to provide the MN of a cellular network with information regarding about connectivity possibilities for both 3GPP and non-3GPP wireless accesses. Moreover, it can also provide policies that assist the MN in determining the best connectivity possibility, taking into consideration details such as current location (e.g., cell ID), available networks (e.g., other cells and access points), time of day and used traffic flows. These policies can also be used to determine and trigger the MIIS to determine when such informational events should be sent to which MNs.

4.2 Interfaces

The purpose of this section is to provide a high level view of the interfaces between WP4 modules. It was aimed to present a general description of the functions for which each interface is used, listing the related primitives and overall operation. Thus, the detailed description of these interfaces, enlisting the full specification of each primitive (i.e. defining the involved parameters) should be consulted in Annex A. The description and specification of the external interfaces, i.e. the interfaces between the Mobility sub-system and the other MEDIEVAL sub-systems (Video Services, Wireless and Transport Optimization), will be shown in [6].

4.2.1 CM_MIIS_If

This interface is “a conceptual interface” between the CM running in the MN and the MIIS running in the network backend. We call it “conceptual” because these two modules do not interact directly but through the L2.5 Abstraction Interface [8] based on IEEE 802.21. Concretely, the CM and the MIIS will exchange the following primitives:

- `MIH_Push_Information_request` (from MIIS to CM)

Through this message, the MIIS inform the CM about new available PoAs to trigger handover procedures (e.g., WiFi offload operations).

- `MIH_Get_Information.request` (from CM to MIIS)
- `MIH_Get_Information.response` (from MIIS to CM)

Through these two primitives, the CM asks the MIIS about new available PoAs when the MN is moving and going out of coverage of serving PoA.

4.2.2 FM_MIIS_If

This interface is “a conceptual interface” between the FM running in the MAR and the MIIS running in the network backend. It is conceptual because the involved modules do not interact directly but through the L2.5 Abstraction Interface [8] based on IEEE 802.21. Concretely, the FM and the MIIS will exchange the following primitives:

MIH_Push_Information_request (from MIIS to FM) through this primitive, the MIIS informs the FM about new available PoAs to trigger handover procedures (e.g., WiFi offload operations).

- MIH_Get_Information.request (from FM to MIIS)

MIH_Get_Information.response (from MIIS to FM) through these primitives, the FM asks the MIIS for an update on available MEDIEVAL PoAs to which the MN could connect due to its current movement towards out of coverage of the serving PoA.

4.2.3 FM_FM_If

This interface is also “a conceptual interface” between the FMs running in different MARs involved in the HO process. As previously referred, the FMs interact through the L2.5 Interface. Concretely, the FMs will exchange the following handover-related 802.21 primitives:

MIH_N2N_HO_Commit.request and MIH_N2N_HO_Commit.response The source FM tells the destination FM to start the commit phase of procedures of the handover. The destination FM will reply after it has finished the procedures of the commit phase.

MIH_N2N_HO_Query_Resources.request and MIH_N2N_HO_Query_Resources.response The source FM asks the destination FM for what resources are available on the destination FM.

MIH_N2N_HO_Complete.request and MIH_N2N_HO_Complete.response The source FM tells the destination FM to start the complete phase of procedures of the handover. The destination FM will reply after it has finished the procedures of the complete phase.

4.2.4 CM_FM_If

This interface is also conceptual and defined between the CM running in the MN and the FM running in the MAR. They interact through the L2.5 Interface. The CM and FM will exchange the following handover-related 802.21 primitives:

- MIH_MN_HO_Candidate_Query.request and MIH_MN_HO_Candidate_Query.response (from CM to FM and viceversa)

The CM asks the FM for a list of potential candidates for handover. The FM will make inquiries and then reply with any possible PoAs that match the requirements.

- MIH_NET_HO_Commit.request and MIH_NET_HO_Commit.response (from FM to CM and viceversa)

The FM tells the CM to commit the handover to the specified PoA. The CM will then reply once the handover is committed.

- MIH_MN_HO_Complete.request and MIH_MN_HO_Complete.response (from CM to FM and viceversa)

The CM tells the FM that its commit phase is finished and that the complete phase procedures should begin. The FM will then reply once the complete phase procedures are finished and thus officially terminating the handover process.

4.2.5 CM_UME_If

This is the interface between the Connection Manager (CM) and the Unicast Mobility Engine (UME) running in the Mobile Node (MN).

This interface is an API and consists in the following two primitives:

- CM_UME_ActivateMobility.request: from the CM to the UME;

- CM_UME_ActivateMobility.response: from the UME and the CM.

This interface is used when the MN is outside the MEDIEVAL DMM domain and IP address continuity can be guaranteed at the cost of handling it by the MN itself, hence activating a mobility client which is in charge of performing the required operations. Therefore this interface carries the trigger generated by the CM to the UME to activate mobility. The effect is that the UME on the terminal sends a Binding Update (BU) message to all the A-MARs the MN wishes to use as home agents for the prefixes they are anchoring, signalling the IP address it has configured on the new access (CoA).

4.2.6 FM_UME_If

This is the interface between the Flow Manager (FM) running in the MAR and the Unicast Mobility Engine (UME) running in the same MAR. This interface is an API and consists in the following two primitives:

- FM_UME_ActivateMobility.request: from the FM to the UME
- FM_UME_ActivateMobility.response: from the UME and the FM

This primitives exchange happens during mobility operation and triggers the mobility protocol between the UME in the serving MAR (S-MAR) and the UME in the anchoring MAR (A-MAR). The *.request* message carries the addresses of the A-MARs that are anchoring MN's flows. The effect is that the S-MAR signals with a Proxy Binding Update (PBU) message to the A-MARs the MN's Care-of Address, i.e. the S-MAR's IP address. The A-MARs reply with a Proxy Binding Acknowledgment (PBA) message and they update their entry in the BC for the MN with the binding between the prefix they advertised to the MN and its current location. Tunnels are established between the MARs and the flows are redirected through the tunnels, without changing the IP addresses pair in the communication.

4.2.7 FM_MUME_If

This is the interface between the Flow Manager (FM) and the Multicast Mobility Engine (MUME), and it is the basis for multicast mobility triggering and completion between the MIH and multicast planes.

This interface is an API and consists of the following primitives:

- FM_MUME_Trigger_Join: from FM to MUME;
- FM_MUME_Trigger_Leave: from FM to MUME;
- FM_MUME_MulticastContextTransfer.request: from FM to MUME;
- FM_MUME_MulticastContextTransfer.response: from MUME to FM;
- FM_MUME_ActivateMobility.request: from FM to MUME;
- FM_MUME_ActivateMobility.response: from MUME to FM;
- FM_MUME_ActivateSource.request: from FM to MUME;
- FM_MUME_ActivateSource.response: from MUME to FM.

Through this API the following major goals are achieved:

- listener mobility: multicast context transfer or tunnel-based mobility;
- faster network subscription of multicast groups during session initiation;
- faster multicast tree pruning, avoiding the leave latency problem;

- source mobility, taking advantage of the mobility tunnel activation.

4.2.8 UME_UME_If

This interface is required to update the MN's location after a change of access sub-network and it is inherited from IETF RFCs. In particular, as long as the terminal is moving within the MEDIEVAL DMM domain, the UMEs involved reside in the MARs and the messages exchanged derive from RFC 5213:

- Proxy Binding Update (PBU): sent by the serving MAR to the anchoring one, which updates the MN's BCE with its new location, it establishes a tunnel endpoint (if not already present) with the PBU's source address (i.e. the current MAR), and installs appropriate routes to setup the MN reachability through the tunnel;
- Proxy Binding Acknowledgement (PBA): sent by the anchoring MAR to the serving one, which updates the MN's BCE including its previous prefix(es) contained in the PBA, it establishes a tunnel endpoint with the PBA's source address (i.e. the anchoring MAR), and installs appropriate routes to forward packets from that MN's prefix(es) through the tunnel.

When the terminal goes out the MEDIEVAL domain and connects to a router without DMM capabilities, the mobility signalling takes place between the UME in the terminal itself and one (or more) UME(s) residing in the MAR. In this case the messages are derived from RFC 6275:

- Binding Update (BU): sent by the terminal to the MAR that is anchoring the prefix the MN wishes to maintain. The MAR updates the MN's BCE with its new location, it establishes a tunnel endpoint (if not already present) with the BU's source address (i.e. the MN's CoA), and installs appropriate routes to forward packets for the MN through the tunnel.
- Binding Acknowledgement (BA): sent by the previous MAR to the MN, which establishes a tunnel endpoint with the BA's source address (i.e. the previous MAR), and installs appropriate routes to forward packets containing the prefix(es) anchored at that MAR through the tunnel.

4.2.9 MUME_MR_If

This interface is required for operating the multicast router. It consists of the following primitives:

- MUME_MR_Join, from MUME to MR: used for joining multicast channels
- MUME_MR_Leave, from MUME to MR: used for fast leave of multicast channels.

4.2.10 MUME_MUME_If

This is the interface between the MUMEs running in different MARs (the previous MAR and the serving MAR). It is used for:

- Multicast Context transfer: Context Transfer is one approach based on the multicast-related information exchange between two MARs in order to speed up the subscription process during handover of multicast listener.
- Tunnel-based mobility: The multicast traffic is sent through the mobility tunnel between two MARs. The A-MAR, where the flow was initially created, plays the role of LMA while the S-MAR is analogous to the MAG.

This interface is an API and consists of the following primitives:

- MUME_MUME_ContextTransfer.request (sent by the serving MAR to the previous MAR): Upon receiving this message, the previous MAR gets the MN's multicast subscription information and embeds it in a message which is sent to the requesting MAR.

- MUME_MUME_ContextTransfer.response (from the previous MAR to the serving MAR): After the receipt of multicast subscription information from this message, the S-MAR joins the necessary multicast groups.
- MUME_MUME_ExtendedProxyBinding.update (sent by the serving MAR to the previous MAR): It is used to request the MN's multicast subscription and establish a tunnel endpoint with the serving MAR. MUME_MUME_ExtendedProxyBinding.acknowledge (from the previous MAR to the serving MAR): It establishes a tunnel endpoint with the previous MAR and joins the necessary multicast groups.

5 Status of the implementation work

5.1 Selection of the PMIPv6-compliant open-source software

The intra-domain mobility management scheme envisioned in the MEDIEVAL project follows a Distributed and Network-based design that conceptually takes origins from the PMIPv6 protocol. Hence, existing PMIPv6-compliant open-source software has been investigated as starting ground to write new code from, rather than carrying out a clean slate network-based DMM implementation.

Two MEDIEVAL partners made available the PMIPv6 software internally developed, with the objective to be extended and adapted according to the DMM innovations. The two implementations are:

1. Open Air Interface PMIPv6 [62] (OAI-PMIPv6), written by EURECOM from the open-source C implementation of MIPv6 (developed within the UMIP project [63]).
2. Open PMIPv6 [64] (OPMIPv6), written from scratch in C++ by IT.

The solutions have been thoroughly examined before choosing which implementation to start from, as it was not possible to allocate resources to go forward both versions and build multiple DMM codes. The following metrics have been preliminarily selected to evaluate the software (the detailed results are omitted here):

1. Dependencies on the kernel;
2. User-space library dependencies;
3. Programming language;
4. Modularity of the code;
5. Integration with existing code (e.g., ODTONE, the IEEE 802.21 implementation);
6. Hardware resources management (CPU load, thread management);
7. Dual-stack (IPv4 and IPv6) feature;
8. Attachment detection (Layer 2) mechanisms;
9. MN identification method;
10. DHCP support;
11. Flow mobility support;
12. Tunneling methods for LMA-MAG interface;
13. Dependency to particular test-bed deployment and equipment.

The analyses were conducted on two platforms settled in different laboratories. A first platform was built using 4 laptops running Ubuntu 10.04 with Linux kernel 2.6.32, used respectively for the LMA functionalities, two MAGs and an MN equipped with two WiFi cards for wireless access. The MAGs provide WiFi radio access via the built-in interfaces or with USB and/or PCMCIA plugged cards, controlled by the MADWIFI driver. Additional hosts were started as virtual machines within the LMA to emulate servers outside the PMIPv6 domain. The second platform consists on a completely virtualized testbed, of which a detailed description is given in next Section 6.

Once the code was installed and running on the platform, some MNs were used to emulate some events typical in a mobile network, as attachment, detachment and handovers. These actions are fundamental to observe whether the implementation's behaviour meets the protocol specifications and/or produces some undesired effects. A deeper insight in the code gave a better understanding of the functions and methods

performing the most important actions and allowed to adjust the protocol parameters to get a better response given the constraints of the testing platform. In fact, the two solutions were developed with some hardware requirements in mind, e.g., the use of specific layer 2 mechanisms for detecting the attachment/detachment of the mobile node while the test-bed setups were a bit limited in this sense.

After conducting the tests the two implementations were more or less equivalent. For this reason some extra attributes were considered to finally decide which software had to be chosen as the basis for the Network-based DMM extension.

EURECOM's code presented some characteristics that made it preferable over IT's. The most important fact is due to the maturity of the code, as it has been under development for more years with respect to the newer OPMIP. This comprises the possibility to trigger the mobility session registration using standard Neighbour Discovery messages, rather than relying on a particular layer 2 mechanism. Also, the code comes with a built in security system based on RADIUS, that can be manually enabled or disabled, hence augmenting the coherence with a real-world deployment. Finally, the code offers a more sophisticated control environment, providing more flexibility in the configuration of the parameters defined by the PMIPv6 protocol (e.g., registration lifetime, local routing, number of retries, etc.), and a dedicated virtual console to check the status of the mobile network (e.g., number of registered nodes with their mobility session).

For all the reasons explained above, a development team has been created within the project, addressing the extension of OAI-PMIPv6 towards the software implementation for Distributed Network-based Mobility Management.

5.2 Modules Status

In the following a description of the status of the implementation for each module is summarized.

5.2.1 CM

At this moment, the specification work on the Connection Manager (CM) is concluded with the main blocks, interfaces and APIs already been defined. As described in Figure 19 in Section 4.1.2, the Connection Manager is divided in two parts, one for mobility support, being developed on WP4 and another for supporting applications and users being developed in the scope of WP2. The implementation of the CM has started and ongoing on its first steps in both WP4 and WP2. For the later, the details can be found on the corresponding deliverable, the D2.2 [7].

Concerning the WP4, the implementation of the Connection Manager will be incremental and will introduce layers of complexity on top of each other till the end of the project, considering the defined milestones of the project. From the WP4 CM subcomponents the one being focused at this point in time is the MIH API to include the features necessary to support the basis of mobility and connectivity to the FM. The integration, at least partial, between the CM engine and the IEEE 802.21 is a priority in order to provide basic mobility functionalities for the September demos. Other components and functionalities will be developed at a later stage according to the plan that has been defined in coordination with WP6 (more detailed information can be found in [12]).

The core components, namely the Engine and the Rules Repository are being developed in C++ using platform independent libraries, although in MEDIEVAL we will be mostly focusing on Linux. This is an important decision taking into consideration the variety of terminal available and their market share. Some of the components developed on WP4 will have a direct interaction with the operative system and in this case we will focus on Linux only in order to provide a solution targeting the planned demos.

5.2.2 FM

The FM components presented in Figure 18, Section 4.1.1, are being coded in C and PHP/HTML for the web interface. All will run on the MAR as threads on the background originated on the Engine daemon that can

be accessed by a separate process, either the webpage interface or, if a graphical interface is lacking, a virtual terminal. Both these interfaces will allow for parameter configuration on-the-fly as well as to override and force some commands that can be useful during the implementation and mostly during the testing phase of the software. From the user point of view the output capable of being produced by these interfaces will add more value, since visualization of internal information becomes easier. The current implementation status is as follows:

- Engine & Flow Database: at the moment only the core skeleton exists, there is still no logic or intelligence implemented.
- Internal Interfaces: these interfaces are specified but not yet implemented. A part of them requires first the implementation of the MIH User interface. Also, the UME/MUME interfaces are already specified but not yet implemented.
- External Interfaces: the interface with the DM and XLO are already specified but not yet implemented.
- Implementation Specific Interfaces: a part of these interfaces are already implemented (the routing and tunnel monitoring). Also the virtual terminal and web interface are already in place.

5.2.3 UME

Following the MEDIEVAL implementation plan, the UME has been developed only for the network-based mobility management, i.e., only the module that runs in the MAR. Also, it has been first generated as a standalone module, independently to the other WP4 components, which will be integrated once all the modules have reached a mature status.

For these reasons, recalling the two DMM approaches (full and partial, as reported in Section 3.2.2), we decided to build the entities defined in the partial solution called “CMD as proxy” (see Figure 6) even if it is not the operation scheme utilized in MEDIEVAL. This choice is motivated by the fact that going for a partially distributed approach allows coders to be detached from other features of the mobility platform, as the IEEE 802.21 signalling and the interactions with the FM, and focus on the implementation of the UME-to-UME interface (refer to Sections 4.1.3 and A.1.8 for the details), the entities’ state machines, the data structures, etc. All these basic building blocks will be easily re-used in the future when the integration phase will start, with little or no adaption effort.

The software is written in C using the OAI-PMIP by EURECOM as starting platform. It is in a quite mature state, unless for some required optimizations which are detailed in D6.3 [12], offering a complete tool to be deployed for testing or demonstrations purposes. Above all, at this stage the UME provides the necessary methods to:

1. run as MAR or CMD: the software is designed to run as a routing daemon, which parameters are defined through an input configuration file (addresses of the entities, interface used, timers values, etc);
2. intercept, parse and generate Neighbour Discovery messages, as Router Solicitation, Router Advertisement, Neighbour Solicitation, Neighbour Advertisement;
3. create and handle IPv6-in-IPv6 tunnels;
4. install proper forwarding rules and routes;
5. maintain the Binding Cache data structure, that implies inserting new MNs’ mobility sessions (BCE entries), update them upon handover, refresh them with proper timers handling, delete them in case of de-registration;
6. generate, send, receive and parse mobility signalling (PBUs and PBAs);

7. basic support to L2 triggers based on IEEE 802.11, used as template to re-code them for specific drivers and/or devices (not mandatory).

It should be noted that the architecture implemented so far requires the introduction of new mobility options, which full specifications are defined in [17]. However, these new components are not part of the MEDIEVAL specifications, and they will be removed for the final implementation.

Future work envisions the integration with the other WP4 modules, which essentially consists in developing the API with the FM. The information carried in that interface is used to send the mobility binding updates to the correct recipient (currently the recipient is always the CMD, which address is statically configured). However, it is more a matter of steering current operations, rather than writing new code blocks.

5.2.4 MUME

Implementation-wise, MUME is a module which runs on the MAR, and interacts with multicast router module (mrd6 software) for requesting subscription information and for triggering join processes. Such interaction is required for enhancing the multicast session initiation and context transfer process from the MUME in Serving-MAR (S-MAR) towards Anchoring-MAR (A-MAR).

The set of features which will be implemented until the end of the project are those required for Demo2, namely the ones for fast multicast session initiation and multicast context transfer procedure. As such, we're providing multicast context transfer in a DMM environment, avoiding the problems involved with mobility tunnels.

At the moment, only the core functionality of MUME has been implemented. Namely, the primitives which are working are:

- MUME_MUME_ContextTransfer.request: sent from the MAR's MUME where the terminal will move to, to the A-MAR's MUME for requesting multicast context transfer.
- MUME_MUME_ContextTransfer.response: the response to the aforementioned request, which transports the context information.
- MUME_MR_Join.request: control sent from MUME which is interpreted by the multicast router as a subscription (e.g. MLD Report) command.

The functionality which will be provided by the end of the project is:

- MUME_MR_Leave.request and MUME_MR_Leave.response;
- MUME_MR_Join.response;
- FM_MUME interface and primitives (except those identified in the list below).

At this stage, the module is not yet prepared for receiving (and responding to) the triggers from FM, meaning that currently those triggers are being done in a static way (i.e. by calling a dedicated function or binary). The signalling order of the context transfer process is depicted in Figure 22.

Moreover, some of the functionality the was specified will most probably be implemented during the project duration. The primitives which fit in this group are:

- FM_MUME:
 - o Activate_Source primitive
- MUME_MUME
 - o ExtendedProxyBindingUpdate and ExtendedProxyBinding.acknowledge primitives

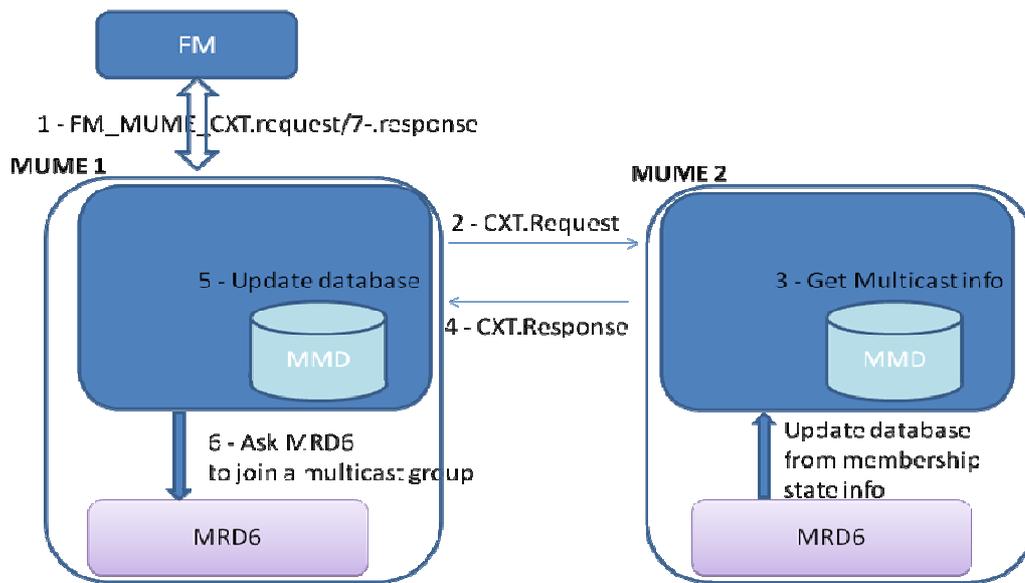


Figure 22: Multicast Context Transfer messages

5.2.5 MIIS

In the MEDIEVAL project the MIIS will be implemented with the binary version of the basic set of IEs defined in the IEEE 802.21 standard [49]. The MIIS server will be an independent network entity, composed by two modules:

- a) the Media Independent Handover Function, composing the L2.5 Abstraction Layer defined in D3.1.
- b) a MIH-User acting as Information Service Engine.

The L2.5 Abstraction Layer is based on the ODTONE [61] implementation, enabling it to receive remote 802.21 Protocol messages from other 802.21-enabled nodes. The MIH-User Information Service Engine implements a memory structure reflecting the network topology, identifying configuration details per PoA. This is the structure that is queried by the engine, when a MIH_Get_Information.request is received at the MIIS Server, sent from a MN. When this query occurs, the MIIS is able to reply to the MN with Information Elements using a MIH_Get_Information.response. This response can either provide the whole topology or, depending on configuration criteria (such as maximum response size), filtered by parameters sent from the MN such as Network ID and Network type.

Moreover, the used MIIS server can also proactively trigger an informational event towards the MN, according to pre-established conditions. The aim is to, upon certain conditions, to have the network provide information towards the Connection Manager at the mobile node, and influence handover decisions therein. An example can be the network detecting that the MN contains a secondary interface of a different access technology but, due to being inactive, is not able to detect PoAs. As such, the MIIS can push topological information via MIH_Push_Information.indication message towards the CM, indicating the alternative access technology, which can motivate it to activate the referred interface and execute a link scan.

6 Status of the simulation work

6.1 ALBLF Network Simulation/Emulation framework

6.1.1 Motivation

In order to save resources and not to deploy yet another dedicated test bed for the MEDIEVAL project, at Alcatel Lucent premises a suitable alternative has been set-up in order to test and use the software components developed by ALBLF, as well as its integration with software from MEDIEVAL partners.

6.1.2 Requirements

The main requirements for this test bed (by order of importance) are as follows:

- i) Real interaction with machines outside the test bed.
- ii) Use of a real TCP/IP stack, to enforce the results reliability.
- iii) The machines' ability to run real code, allowing them to run code developed by all MEDIEVAL partners, with little or no need for adaptation.
- iv) The machines' ability to run customizable kernels, since some software of the project has the same requirement.
- v) Flexibility (to a degree) to allow the study the scalability of the architectural design and consequently the software's performance and resource consumption.
- vi) The use of either simulation or emulation to provide a somewhat reliable substitute for the wireless environment and node movement.

6.1.3 Test bed Design Solution

In order to face some of the listed requirements, it was decided to deploy the test bed nodes in a fashion similar to virtual machines, profiting from the advantages provided by their deployment. To this end an open-source tool named Netkit, which deploys and manages networks composed of virtual machines, has been chosen.

Netkit has been customized in order to integrate it with other components to allow the inclusion of wireless technologies. Such additional components are part of the Simulation of Wireless Environment Extensions for Netkit (SWEEN) software package.

In the next two sub-sections more details are provided.

6.1.4 NETKIT

Netkit [65] is a free and open-source tool mainly used to deploy networks of virtual machines. This tool was initially designed to test and solve bugs on the Linux UML [66] kernel variant, but has since diversified its fields of use to areas like education, network emulation and software development.

The tool is heavily based on UML and connects the created virtual machines by emulating Ethernet connections between the virtual machines' interfaces. The virtual machines run a minimalistic stripped-down Debian [67] distribution (Debian 5 - Lenny - kernel v2.6.25-6), which is generic enough to allow for most commonly used Linux tools and software to be easily installed, updated and run on it. They also run a specific file system that enables efficient data storage on the host machine. These two features make possible the deployment of larger virtual machine networks without the usual huge burdening on the host machine's

resources. The tool also features the possibility of connecting the virtual networks deployed, with both the host machine and any existing network connections residing on it, enabling for instance internet access.

For the initial tests with PMIP implementations we had to introduce an upgraded Netkit kernel in order to use some of the partner developed software, such as the PMIPv6 software from EURECOM. The upgrade to kernel v2.6.28-0 included the addition of several mobility options requested by the PMIPv6 implementations.

6.1.5 SWEEN (Simulation of Wireless Environment Extensions for Netkit)

The Netkit tool, however, does not serve all our needs. One of its main constraints is that it currently allows only wired connections between its virtualized nodes, not yet entering the field of wireless technologies for communication between the nodes. Thus, Netkit needs to be customized, in order to integrate it with other components that we develop to allow the inclusion of wireless technologies (or simulation/emulation thereof). Such additional components are already partially implemented and in part constitute SWEEN.

Simulation of Wireless Environment Extensions for Netkit (SWEEN), is a simple piece of software, coded in C, based on the client-server model (see Figure 23) introduced both on Netkit's virtualized nodes (SWEEN client) and the host machine (SWEEN server). As stated before, Netkit only allows for fixed and static Ethernet-alike connections between its virtualized nodes, so in order to enable our testbed and thus Netkit to have dynamic wireless-alike connections we developed SWEEN. This software will control the node's interfaces and the point to where they are connected to allow the integration of wireless characteristics imported from the wireless simulation models.

The core (depicted in light blue in Figure 23) is constituted of the centrepiece software (Netkit) and of any software running on the virtual machines (with the exception of the SWEEN client). The access management (depicted in dark blue, Figure 23) comprises the clients and server of **SWEEN (Simulation of Wireless Environment Extensions for Netkit)**, while the wireless management is constituted of several different modules that act as input to the access management block (depicted in green and orange, in Figure 23).

The test bed architecture, depicted in Figure 23, as we can see, demonstrates a clear distinction between the Netkit environment and the management components we created for the virtualized nodes. The components we created and their functionalities follow:

- Node Position Model

The node Position Model provides the position on the grid of each mobile node at each given interval. Currently this position is manually overridden, but in the near future we intend to use traces of real node movement. These trace movements patterns will be as reliable and real as possible, replicating real world scenarios such as vehicles' paths while driving in a city [68].

- LTE/WiFi Simulation Models

The LTE/WiFi Simulation Models will take the input from the Node Position Model and analyse the possible connections to Points of Access that can be made taking into account radio-specific parameters, such as, distance, signal to noise ratio and transmission power, among others. This analysis will produce (if possible) an optimal Point of Access (of one or more access technologies) for each mobile node on that given time. If a new connection is possible for a mobile node this information is pushed to the SWEEN Server.

- SWEEN Server

The SWEEN Server is the entity that keeps track of the active connections that are being enforced through the SWEEN clients on Netkit. If input is received from the Wireless models, the server will check the database if the connection is possible to be done on Netkit. This check can add several criteria conditions, for instance if the mobile node supports multiple access technologies, if the mobile node supports multi-homing, etc. If these checks come back negative, the candidate point-of-access for handover is discarded, and the server will go back to its original state. Otherwise, the server updates the Link database and communicates to the client in the correspondent mobile node, which interfaces to connect/disconnect, in preparation for the handover. In the meanwhile the server rearranges the Netkit virtual hub attachments, disconnecting the mobile node from the current point-of-access virtual hub and connecting him to the new point-of-access virtual hub. This phase is necessary because we are simulating wireless through wired, and on wired access

we need to change the endpoints of the link to match the wireless endpoints. Once this phase is completed the server informs the client which interfaces to re-connect.

- SWEEN Client

The SWEEN client is being executed inside each virtual machine on the control virtual interface. The control virtual interface is the only interface whose connection is never severed and it's connected via a special virtual hub that allows for access to the real host, and if enabled, internet. The client keeps track of the interfaces active in the mobile node and enforces any command to switch on/off any of the interfaces.

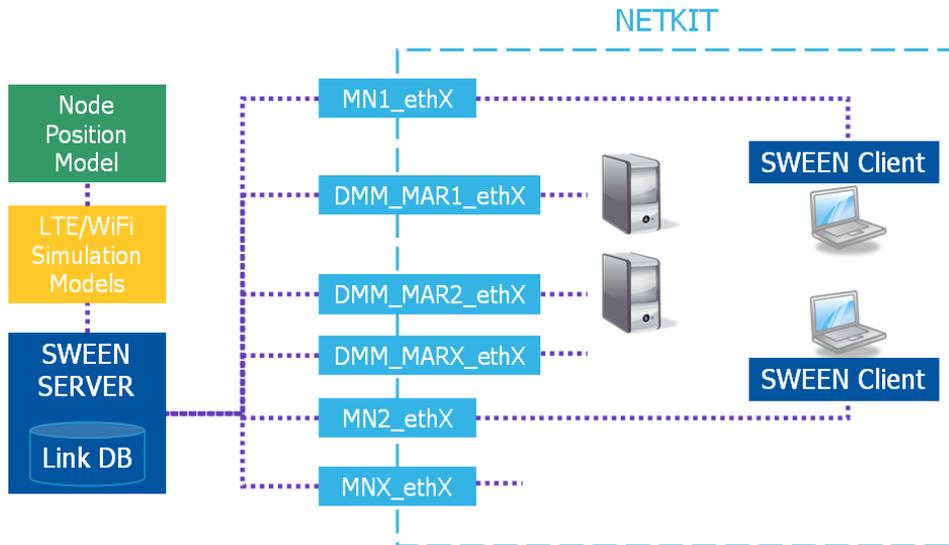


Figure 23: Architecture of the SWEEN/NETKIT test bed to evaluate scalability

The light blue interfaces depicted in Figure 23 are special interfaces, provided by Netkit, that enable communication and access between host machine and virtual nodes, in IPv4. We take advantage of these interfaces to create our SWEEN control network (consisting of one interface per virtual node) thus enabling the communication between clients and server.

6.1.6 Current Status

Currently, the test bed resides in a single powerful machine, with a quad-core CPU and several gigabytes of RAM. The use of the Netkit tool solves the challenge concerning the node implementation, already allowing some tests to be performed. Among them, are the tests related to the choice of PMIP implementation described in the previous section. We also successfully tested other partner software such as ODTONE, and more recently the test bed is being used to experiment with OPMIP extensions and OpenCDN [68]. During the project meeting in Tel-Aviv (October 2011), a small remote demonstration was performed to show some of the available functionalities.

Also the SWEEN elements have been implemented and added to the Netkit framework. Preliminary tests were carried out on this test bed by overriding the mobile nodes position with commands directly pushed into the SWEEN server, on-the-fly while an experiment on Netkit was being run with five MARs (running an early version of the DMM software) and ten mobile nodes. The mobile node handovers between MARs occur swiftly and without issue, though there is some, not at all unexpected, packet loss due to loss of connectivity. Future optimizations will mitigate this issue, but the main point to focus on is that although we haven't yet thoroughly tested the scalable performance of either the test bed potential or the mobility framework upon it, the foundations for the achievement of these results have been laid and currently are a working proof-of-concept.

6.1.7 Future Work

In the short-term we expect to conduct further software testing. In the medium and long term, the development plan includes:

- Fine tuning of the test bed parameters to increase the number of supported nodes.
- Integration of LTE and WiFi analytical modules for wireless access medium simulation/emulation.
- Integration of node movement and position simulation, by incorporating the use of real world traces of position and movement.
- Possible development of usability improvement extensions.

7 Conclusion and next steps

This deliverable described the updated description of the mobility architecture first presented in [9] and [10]. The architecture is based on the Distributed Mobility Management (DMM) model, a topic whose study formally started in IETF's DMM WG [50], and which is motivated by the need to efficiently make use of mobile operator networks entities by: i) distributing mobility anchors at the edge of mobile networks and ii) by activating mobility resources only for applications/flows which require them.

Moreover, the deliverable aimed to show the benefits of this model when applied to video services, namely by comprising a flatter mobility and an optimised and dynamic CDN distribution, and then obviate the advantages comparatively to traditional core anchoring models to cope with the increasing demand for high bandwidth-consuming video services. Aligned with the efficiency requirement, the deliverable presented work achieved on IP multicast mobility, namely on the analysis of use cases in DMM scenarios, both from the point of view of the user and network. Besides, optimization mechanisms were introduced in scenarios for SVC transport in mobile networks, both for unicast and IP multicast cases.

Regarding the architecture specification, the document provides a detailed description of the modules comprising the Mobility sub-system, enlisting the corresponding functionality, and then presenting the current implementation status – whose advances already resulted in two demonstrations. A similar work was done for each of the modules interfaces, for which the associated primitives and parameter specification is fully shown.

The upcoming specification and implementation work will allow to further validate the architecture and to perform some experimental evaluation of effective advantages (e.g., in terms of better scalability and reliability, lower signalling overhead and shorter handover latencies, better control on the mobility granularity offered by the network) of a DMM based approach when compared to traditional core based anchoring models. These assessment and evaluations efforts will be present in [6].

It is expected that the feedback and experience gained from the module and architecture completion, and in particular their integration in the full MEDIEVAL architecture, as well as the validation and evaluation work, will lead to the revision and consolidation of the specification defined in this deliverable. The resulting architecture will be part of the “Final Operational Mobility Architecture” D4.4 [11] document.

Acknowledgements and Disclaimer

This work was partially funded by the European Commission within the 7th Framework Program in the context of the ICT project MEDIEVAL (Grant Agreement No. 258053). The views and conclusions contained here are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the MEDIEVAL project or the European Commission.

References

- [1] L.M. Contreras, C.J. Bernardos, I. Soto, “PMIPv6 multicast handover optimization by the Subscription Information Acquisition through the LMA (SIAL)”, IETF WG Draft, May 2012, draft-ietf-multimob-fast-handover-00.txt
- [2] H. Chan (Ed.), “Problem Statement for Distributed and Dynamic Mobility Management”, IETF Draft, October 2011, draft-chan-distributed-mobility-ps-05-txt
- [3] H. Yokota, “Use case scenarios for Distributed Mobility Management”, IETF Draft, September 2010, draft-yokota-dmm-scenario-00.txt
- [4] MEDIEVAL Project, Deliverable D1.1 : “Preliminary architecture design”, June 2011
- [5] MEDIEVAL Project, Deliverable D1.2: “Business cases analysis”, October 2011
- [6] MEDIEVAL Project, Deliverable D1.3: “Final architecture design”, interim version: June 2012 (final version: December 2012)
- [7] MEDIEVAL Project, Deliverable D2.2: “Final specification for video service control”, June 2011
- [8] MEDIEVAL Project, Deliverable D3.2: “Final Specification for the Wireless Access Functions and Interfaces”, interim version: June 2012
- [9] MEDIEVAL Project, Deliverable D4.1: “Light IP Mobility architecture for Video Services: initial architecture”, June 2011
- [10] MEDIEVAL Project, Deliverable D4.2: “IP Multicast Mobility Solutions for Video Services”, June 2011
- [11] MEDIEVAL Project, Deliverable D4.4: “Final Operational Mobility Architecture”, due in December 2012
- [12] MEDIEVAL Project, Deliverable D6.3: “First Periodic Testing Report”, June 2012
- [13] 3GPP Work Item = 450038 (SIPTO)
- [14] 3GPP TR 23.859 “LIPA Mobility and SIPTO at the Local Network”
- [15] 3GPP Work Item = 500028 (LIMONET)
- [16] 3GPP Proposed WID for User plane congestion management (UPCON) - 3GPP TSG-SA WG1 Meeting #56 - San Francisco, CA, USA, 14th – 18th November 2011
- [17] C. J. Bernardos, A. de la Oliva, F. Giust, T. Melia, R. Costa “A PMIPv6-based solution for Distributed Mobility Management”, IETF Internet Draft, March 2012, draft-bernardos-dmm-pmip-01.txt
- [18] C. J. Bernardos, A. de la Oliva, F. Giust, “A IPv6 Distributed Client Mobility Management approach using existing mechanisms”, IETF Internet Draft, March 2011, draft-bernardos-mext-dmm-cmip-00.txt
- [19] T. Melia and S. Gundavelli (Ed.), “Logical Interface Support for multi-mode IP Hosts”, IETF Draft, April 2012, draft-ietf-netext-logical-interface-support-05.txt
- [20] C. J. Bernardos (Ed.), “Proxy Mobile IPv6 Extensions to Support Flow Mobility”, IETF Draft, March 2012, draft-ietf-netext-pmipv6-flowmob-03.txt

- [21] X. Zhou, J. Korhonen, C. Williams, S. Gundavelli and C.J. Bernardos, "Prefix Delegation for Proxy Mobile IPv6", IETF Draft, March 2012, draft-ietf-netext-pd-pmip-02.txt
- [22] C. J. Bernardos, M. Calderon, I. Soto, "PMIPv6 and Network Mobility Problem Statement", IETF Draft, March 2012, draft-bernardos-netext-pmipv6-nemo-ps-02.txt
- [23] X. Zhou, J. Korhonen, C. Williams, S. Gundavelli, C. J. Bernardos, "Prefix Delegation for Proxy Mobile IPv6". IETF Draft, March 2012, draft-ietf-netext-pd-pmip-02.txt
- [24] I. Soto, C. J. Bernardos, M. Calderon, A. Banchs, A. Azcorra, "NEMO-Enabled Localized Mobility Support for Internet Access in Automotive Scenarios", IEEE Communications Magazine, Automotive Networking - Technology, Design, and Applications series, Vol.47, No.5, May 2009
- [25] J. C. Zuniga, L. M. Contreras, C. J. Bernardos, S. Jeon, Y. Kim, "Multicat Mobility Routing Optimizations for Proxy Mobile IPv6", IETF Draft, March 2012, draft-ietf-multimob-pmipv6-ropt-00.txt
- [26] S. Figueiredo, S. Jeon, and R. L. Aguiar, "IP Multicast Use Cases Analysis for PMIPv6-based Distributed Mobility Management", IETF draft, March 2012, draft-sfigueiredo-multimob-use-case-dmm-01.txt
- [27] F. Giust, A. de la Oliva, C. J. Bernardos, R. P. Ferreira da Costa, "A Network-based Localized Mobility Solution for Distributed Mobility Management", MMNF 2011 workshop in conjunction with WPMC 2011, October 2011
- [28] A. de la Oliva, M. Calderon, C. J. Bernardos, R. Wakikawa, "Client and Network-based Dual Stack Mobility Management", submitted for publication to the IEEE Wireless Communications Magazine
- [29] H. Soliman (editor), "Mobile IPv6 Support for Dual Stack Hosts and Routers", RFC 5555, June 2009
- [30] R. Wakikawa, S. Gundavelli, "IPv4 Support for Proxy Mobile IPv6", RFC 5844, May 2010
- [31] A. de la Oliva, I. Soto, M. Calderon, and C. J. Bernardos, "The costs and benefits of combining different IP mobility approaches", submitted for publication to Springer Wireless Personal Communications
- [32] A. de la Oliva, C. J. Bernardos; M. Calderon, T. Melia, J. C. Zuniga, "IP Flow Mobility: Smart Traffic Offload for Future Wireless Networks", IEEE Communications Magazine, Feature Topic on Traffic Management for Mobile Broadband Networks, July 2011
- [33] R. Costa, T. Melia, L. Eznarriaga, F. Giust, A. de la Oliva, C. J. Bernardos, "Wireless Multi-Access Delivery for SVC-based Video Applications", MONAMI 2012, submitted in May 2012
- [34] R. Costa, T. Melia, D. Munaretto and M. Zorzi, "When Mobile Networks meet Content Delivery Networks: challenges and possibilities." MobiArch 2012, submitted in May 2012
- [35] D. Munaretto , T. Melia, S. Randriamasy and M. Zorzi, "Online path selection for video delivery over cellular networks", IEEE Globecom 2012 (submitted)
- [36] T. Melia, C. J. Bernardos, A. de la Oliva, F. Giust, M. Calderon, "IP Flow Mobility in PMIPv6 Based Networks: Solution Design and experimental Evaluation", Wireless Personal Communications Journal, Springer, October 2011
- [37] Kwang-deok Seo, Jin-soo Kim, Soon-heung Jung, and JeongJu Yoo, "A Practical RTP Packetization Scheme for SVC Video Transport over IP Networks," ETRI Journal, vol.32, no.2, Apr. 2010, pp.281-291.
- [38] Wenger, Y.-K. Wang, T. Schierl and A. Eleftheriadis, "RTP Payload Format for Scalable Video Coding", RFC 6190, Internet Engineering Task Force, May 2011.

- [39] S. Figueiredo, M. Wetterwald, T. Nguyen, L. Eznarriaga, N. Amram, R. L. Aguiar, "SVC Multicast Video Mobility Support in MEDIEVAL Project", *Future Networks & Mobile Summit 2012*, submitted in December 2011
- [40] L. M. Contreras, C. J. Bernardos, I. Soto, "RAMS: Improved handover performance for multicast traffic in PMIPv6", *JoWUA Vol. 2 No. 2*, 2011.
- [41] D. Corujo, C. Guimarães, B. Santos, and R. L. 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"*, Sep. 2011.
- [42] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, B. Patil, "Proxy Mobile IPv6", RFC 5213, August 2008
- [43] T. Schmidt, M. Waehlich, and S. Krishnan, "Base Deployment for Multicast Listener Support in Proxy Mobile IPv6 (PMIPv6) Domains", RFC 6224, April 2011
- [44] H. Fenner, H. He, B. Haberman, and H. Sandick, "Internet Group Management Protocol (IGMP) / Multicast Listener Discovery (MLD)-Based Multicast Forwarding ("IGMP/MLD Proxying")", RFC 4605, August 2006
- [45] C. Perkins, D. Johnson, J. Arkko, "Mobility Support in IPv6", RFC 6275, July 2011
- [46] S. Figueiredo, S. Jeon, and R. L. Aguiar, "Use-cases Analysis for Multicast Listener Support in Network-based Distributed Mobility Management", Proc. of IEEE PIMRC 2012, to appear.
- [47] S. Jeon, S. Figueiredo, and R. L. Aguiar, "A Channel-Manageable IP Multicast Support Framework for Distributed Mobility Management", Submitted to IEEE GLOBECOM 2012
- [48] 3rd Generation Partnership Project, Technical Specification Group Core Network and Terminals, Access Network Discovery and Selection Function (ANDSF) Management Object, 3GPP TS 24.312, 2011
- [49] "IEEE Standard for Local and Metropolitan Area Networks- Part 21: Media Independent Handover," IEEE Std 802.21-2008 , vol., no., pp.c1-301, Jan. 21 2009
- [50] IETF Distributed Mobility Management WG (DMM), current charter at <http://datatracker.ietf.org/wg/dmm/charter/>
- [51] <http://www.openvpn.net/>
- [52] <http://www.live555.com>
- [53] <http://www.mplayerhq.hu/>
- [54] <http://sourceforge.net/projects/opensvcdecoder/>
- [55] <http://www.bigbuckbunny.org/>
- [56] <http://www.hhi.fraunhofer.de/en/departments/image-processing/image-video-coding/scalable-video-coding/svc-scalable-extension-of-h264avc/>
- [57] "Proactive pull key distribution through target PoS", <https://mentor.ieee.org/802.21/dcn/11/21-11-0157-02-0sec-suggested-remedy-proactive-pull-key-distribution.doc>.
- [58] <https://mentor.ieee.org/802.21/dcn/11/21-11-0186-00-srho-proactive-pull-key-distribution.ppt>, presented during the 47th meeting in Atlanta, USA, November 2011.

- [59] <https://mentor.ieee.org/802.21/dcn/11/21-11-0198-00-srho-merging-proactive-pull-key-dist.ppt>, presented in 802.21c 02/05/2012 Audio Conference.
- [60] Y. Ohba, R. Martin, A. de la Oliva, C. Perkins, “21-12-0020-04-srho-secure-key-distribution” IEEE 802.21c draft, March 2012.
- [61] Open Dot Twenty ONE – <http://atnog.av.it.pt/odtone>
- [62] <http://www.openairinterface.org/components/page1103.en.htm>
- [63] <http://umip.org/>
- [64] <http://helios.av.it.pt/projects/opmip>
- [65] <http://www.netkit.org>
- [66] UML – User Mode Linux, <http://user-mode-linux.sourceforge.net/>
- [67] Debian Linux Distribution, <http://www.debian.org/>
- [68] Vehicular mobility trace of the city of Cologne, Germany, <http://kolntrace.project.citi-lab.fr/>
- [69] The OpenCDN Project, <http://labtel.ing.uniroma1.it/opencdn/>

Annex A Complete Updated Specification

A.1 Internal Interfaces

A.1.1 CM_MIIS_If

The interface between the Connection Manager (CM) and the MIIS is common to the same MIH_SAP provided by MIHF since this interface's primitives are part of the IEEE 802.21 MIH protocol. The interface will reuse the existing primitives and parameters defined in the IEEE 802.21 standard, which should be consulted for further details. Concretely, the CM and MIIS will exchange the following 802.21 primitives.

- **MIH_Push_Information_request (from MIIS to CM)**
- **MIH_Get_Information.request and MIH_Get_Information.response (from CM to MIIS)**

A.1.2 FM_MIIS_If

It is based on IEEE 802.21

- **MIH_N2N_HO_Query_Resources.request and MIH_N2N_HO_Query_Resources.response**
- **MIH_N2N_HO_Commit.request and MIH_N2N_HO_Commit.response**
- **MIH_N2N_HO_Complete.request and MIH_N2N_HO_Complete.response**

A.1.3 FM_FM_If

The interface between two Flow Managers (FM) is common to the same MIH_SAP provided by MIHF since this interface's primitives are part of the IEEE 802.21 MIH protocol. The interface will reuse the existing primitives and parameters defined in the IEEE 802.21 standard, which should be consulted for further details. Concretely, the interface will exchange the following handover-related 802.21 primitives.

- **MIH_N2N_HO_Commit.request and MIH_N2N_HO_Commit.response**
- **MIH_N2N_HO_Query_Resources.request and MIH_N2N_HO_Query_Resources.response**
- **MIH_N2N_HO_Complete.request and MIH_N2N_HO_Complete.response**

A.1.4 CM_FM_If

The interface between the Connection Manager (CM) and the Flow Manager (FM) is common to the same MIH_SAP provided by MIHF since this interface's primitives are part of the IEEE 802.21 MIH protocol. The interface will reuse the existing primitives and parameters defined in the IEEE 802.21 standard, which should be consulted for further details. Concretely, the CM and FM will exchange the following handover-related 802.21 primitives:

- **MIH_MN_HO_Candidate_Query.request and MIH_MN_HO_Candidate_Query.response (from CM to FM and viceversa)**
- **MIH_NET_HO_Commit.request and MIH_NET_HO_Commit.response (from FM to CM and viceversa)**
- **MIH_MN_HO_Complete.request and MIH_MN_HO_Complete.response (from CM to FM and viceversa)**

A.1.5 CM_UME_If

This is the interface between the Connection Manager (CM) and the Unicast Mobility Engine (UME).

A.1.5.1 CM_UME_ActivateMobility

CM_UME_ActivateMobility.request

Function

This message is used to notify the Unicast Mobility Engine in the terminal, to trigger the mobility protocol.

Semantics of the service primitive

```
CM_UME_ActivateMobility.request (
    Mobile_Terminal_ID,
    Interface_ID,
    HA_Address
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually a NAI)
Interface_ID	STRING	The identifier of the interface where to activate mobility
HA_Address	IN6_ADDRESS	Home Agent Address

Table 2: CM_UME_ActivateMobility.request parameter list

When generated

The message is generated when the CM is notified that it has attached to the new MAR. It forwards this notification to the UME through this message.

Effect on receipt

After receiving this message the UME should start the Mobility Protocol for the mentioned node.

CM_UME_ActivateMobility.response

Function

This message is used to notify the CM that the mobility protocol has finished the setup for the mobile terminal.

Semantics of the service primitive

```
CM_UME_ActivateMobility.response (
    Mobile_Terminal_ID,
    Interface_ID,
    ResultCode
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of the interface where to activate mobility
ResultCode	UINT_8	The result of the protocol setup. (0 on OK, otherwise return an error code)

Table 3: CM_UME_ActivateMobility.response parameter list

When generated

The message is generated by the UME when the mobility procedure of the protocol is over. It contains the result of this process.

Effect on receipt

After receiving this message, the CM should either complete the handover, if it receives an affirmative result, or take appropriate actions otherwise.

A.1.6 FM_UME_If

This is the interface between the Flow Manager (FM) and the Unicast Mobility Engine (UME).

A.1.6.1 FM_UME_ActivateMobility

FM_UME_ActivateMobility.request

Function

This message is used to notify the Unicast Mobility Engine to trigger the mobility protocol.

Semantics of the service primitive

```
FM_UME_ActivateMobility.request (
    Mobile_Terminal_ID,
    Link_ID,
    A-MAR_Address
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually a NAI)
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of a link that is not associated with the peer node. The LINK_ADDR contains the address of this link.
A-MAR_Address	IN6_ADDRESS	Anchoring MAR IPv6 address

Table 4: FM_UME_ActivateMobility.request parameter list

When generated

The message is generated when the FM is notified that a new node has attached to the MAR. It forwards this notification to the UME through this message.

Effect on receipt

After receiving this message the UME should start the Mobility Protocol for the mentioned node.

FM_UME_ActivateMobility.response

Function

This message is used to notify the FM that the mobility protocol has finished the setup for the mobile terminal.

Semantics of the service primitive

```
FM_UME_ActivateMobility.response (
    Mobile_Terminal_ID,
    Link_ID,
    ResultCode
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually a NAI)
Link_ID	SEQUENCE{LINK_TYPE,	The identifier of a link that is not associated with

	LINK_ADDR}	the peer node. The LINK_ADDR contains the address of this link.
ResultCode	UINT_8	The result of the protocol setup. (0 on OK, otherwise return an error code)

Table 5: FM_UME_ActivateMobility.response parameter list

When generated

The message is generated by the UME when the mobility procedure of the protocol is over. It contains the result of this process.

Effect on receipt

After receiving this message, the FM should either complete the handover, if it receives an affirmative result, or take appropriate actions otherwise.

A.1.7 FM_MUME_If

This is the interface between the FM and the MUME.

A.1.7.1 FM_MUME_ActivateMobility

FM_MUME_ActivateMobility.request

This message is used to notify the MUME to trigger the mobility protocol for a multicast source/listener. As such, this message corresponds to multicast mobility operations using the tunnel-based scheme.

Semantics of the service primitive

```
FM_MUME_ActivateMobility.request (
    Mobile_Terminal_ID,
    Link_ID,
    Multicast_Role,
    A-MAR_Address
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually a Network Address Identifier (NAI))
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of a link that is not associated with the peer node. The LINK_ADDR contains the address of this link.
Multicast_Role	SHORT_INT	The role the MN has in multicast operations

A-MAR_Address	IN6_ADDRESS	Anchoring MAR IPv6 address
---------------	-------------	----------------------------

Table 6. FM_MUME_ActivateMobility.request parameter list

When generated

The message is generated by the Flow Manager, knowing that a multicast source or listener node requiring IP address continuity has / is going to attach.

Effect on receipt

After receiving this message the MUME should start the (multicast-aware) mobility protocol for the mentioned node. If the Multicast_Role parameter is set to 0, the MN has a multicast listener function, and the MUME should initiate the mobility process extended with multicast context (MLD state) transfer, that is, relative to the ExtendedProxyBinding.update message. If the Multicast_Role parameter is set to 1, the MN acts as a multicast source, the MUME block assumes the RP to have the A-MAR address, and triggers a standard PBU message.

FM_MUME_ActivateMobility.response

Function

This message is used to notify the FM that the mobility protocol has finished the tunnel setup for the mobile terminal.

Semantics of the service primitive

```
FM_MUME_ActivateMobility.response (
    Mobile_Terminal_ID,
    Link_ID,
    Multicast_Role,
    ResultCode
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually a NAI)
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of a link that is not associated with the peer node. The LINK_ADDR contains the address of this link.
Multicast_Role	SHORT_INT	The role the MN has in multicast operations
ResultCode	UINT_8	The result of the protocol setup. (0 on OK otherwise return an error code)

Table 7. FM_MUME_ActivateMobility.response parameter list

When generated

The message is generated by the MUME when the mobility procedure of the protocol is over. It contains the result of this process.

Effect on receipt

After receiving this message, the FM should either complete the handover, if it receives an affirmative result, or take appropriate action otherwise.

A.1.7.2 FM_MUME_ActivateSource

FM_MUME_ActivateSource.request

Function

This message is used to setup the local MAR as the incoming end point of the multicast content, sending the content directly to the RP natively.

Semantics of the service primitive

```
FM_MUME_ActivateSource.request (
    Mobile_Terminal_ID,
    Link_ID,
    RP_address
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of a link that is not associated with the peer node. The LINK_ADDR contains the address of this link.
RP_address	IN6_ADDRESS	The IP address of the RP assigned to the IP multicast channel

Table 8. FM_MUME_ActivateSource.request parameter list

When generated

The message is generated when the FM is notified that a MN acting as a multicast source is going to associate to the corresponding MAR.

Effect on receipt

The MUME configures its MLD proxy instance uplink interface towards the RP, and one of its downstream interfaces as the multicast content entrance point. The multicast channel is based on the IP multicast group address and on the link-scope address of the MN. The RP should be obtained similarly to the way A-MAR address is obtained for mobility activation procedures, e.g. by extended N2N_HO_Query_Resources to provide the RP address.

FM_MUME_ActivateSource.response

Function

This message is used to acknowledge the preparation of the MUME for sending the content for the requested multicast channel.

Semantics of the service primitive

```
FM_MUME_ActivateSource.response (
    Mobile_Terminal_ID,
    Link_ID,
    ResultCode
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of a link that is not associated with the peer node. The LINK_ADDR contains the address of this link.
ResultCode	UINT_8	The result of the attachment process (0 on OK otherwise return an error code)

Table 9. FM_MUME_ActivateSource.response parameter list

When generated

This message is generated by the MUME after a source attaches to the corresponding MAR.

Effect on receipt

After receiving this message, the FM should either complete the attachment process, if it receives an affirmative result, or take appropriate action otherwise.

A.1.7.3 FM_MUME_MulticastContextTransfer

FM_MUME_MulticastContextTransfer.request

Function

Prepares MUME for a multicast context transfer process.

Semantics of the service primitive

```
FM_MUME_MulticastContextTransfer.request (
    A_MAR_address,
    Mobile_Terminal_ID,
    Link_ID
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of the interface where the MN will attach to.
A_MAR_address	IN6_ADDRESS	The IP address of the previous MAR

Table 10. FM_MUME_MulticastContextTransfer.request parameter list

When generated

This primitive is triggered when a direct routing scheme is best suited for the multicast listener’s application needs.

Effect on receipt

MUME triggers a multicast context transfer process, without mobility tunnel setup.

FM_MUME_MulticastContextTransfer.response

Function

This message is used by MUME to acknowledge the conclusion of the multicast context transfer process.

Semantics of the service primitive

```
FM_MUME_MulticastContextTransfer.response (
    Mobile_Terminal_ID,
    Link_ID,
    ResultCode
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of the interface where the MN will attach to.
Result_Code	UINT_8	The result of the protocol setup. (0 on OK otherwise return an error code)

Table 11. FM_MUME_MulticastContextTransfer.response parameter list

When generated

The message is generated by the MUME when the context transfer process with the previous router (A-MAR) is over.

Effect on receipt

After receiving this message, the FM should either complete the handover, if it receives an affirmative result, or take appropriate action otherwise.

A.1.7.4 FM_MUME_Trigger_Join

FM_MUME_Trigger_Join.request

Function

This message is used by FM for triggering multicast subscription (using PIM Join) or MRIB update (when existing subscription is received in new downstream interface, i.e. PoA) from local MUME.

Semantics of the service primitive

```
FM_MUME_Trigger_Join.request (
    Mobile_Terminal_ID,
    Multicast_Subscriptions
    Link_ID
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)
Multicast_Subscription	SEQUENCE{SEQUENCE{STRING, IN6_ADDRESS, SEQUENCE{IN6_ADDRES S}}}	List of multicast channels (filter mode, group, source(s))
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of the interface where the MN will attach to.

Table 12. FM_MUME_Trigger_Join.request parameter list

When generated

The message is generated when the FM is aware that a user will start a multicast video session at a PoA of the local MAR or a user which was receiving such a previously established session will move to a PoA of the local MAR.

Effect on receipt

After receiving this message, the MUME should communicate with the MR, so that the multicast functionality of the MAR (PIM-SM protocol) updates its subscription state and joins the multicast tree if needed.

FM_MUME_Trigger_Join.response

Function

This primitive is used by MUME for confirming towards FM the status of its Join operation.

Semantics of the service primitive

```
FM_MUME_Trigger_Join.response (
    Mobile_Terminal_ID,
    Result_Code
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)
Result_Code	UINT_8	The result of the Trigger Join operation

Table 13. FM_MUME_Trigger_Join.response parameter list

When generated

The message is generated when the MUME completed the Join process towards the MR.

Effect on receipt

The FM will proceed with the session initiation or handover completion, depending on the case which triggered the FM_MUME_Trigger_Join.request.

A.1.7.5 FM_MUME_Trigger_Leave.request

Function

This message is used by FM for updating MUME’s MMD regarding the leave of a multicast listener.

Semantics of the service primitive

```
FM_MUME_Trigger_Join.request (
    Mobile_Terminal_ID,
    Multicast_Subscriptions
    Link_ID
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal

		(usually an NAI)
Multicast_Subscriptions	SEQUENCE{SEQUENCE{STRING, IN6_ADDRESS, SEQUENCE{IN6_ADDRES S}}}	List of multicast channels (filter mode, group, source(s))
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of the interface where the MN will attach to.

Table 14. FM_MUME_Trigger_Leave.request parameter list

When generated

The message is generated when the FM is informed that a previously connected multicast video listener handoff'ed to a different MAR.

Effect on receipt

After receiving this message, the MUME will update its MMD, and possibly send a PIM Join/Prune for the case where the listener was the last subscriber of the corresponding multicast channel (s).

FM_MUME_Trigger_Leave.response

Function

This primitive is used by MUME for confirming towards FM the status of the Leave operation.

Semantics of the service primitive

```
FM_MUME_Trigger_Leave.response (
    Mobile_Terminal_ID
    Result_Code
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)
Result_Code	UINT_8	The result of the Trigger Leave operation

Table 15. FM_MUME_Trigger_Leave.response parameter list

When generated

This primitive is used for confirming the success of the MUME Join operation.

Effect on receipt

FM will proceed with the handover completion.

A.1.8 UME_UME_If

This interface is used to exchange IP mobility signaling. With respect to the type of mobility event, two different classes of messages are transmitted:

1. When the mobility event takes place within the MEDIEVAL DMM domain, the signaling occurs between UMEs residing in the MARs using the Proxy Binding Update and Proxy Binding Acknowledgment messages defined in RFC 5213. The rules specified in RFC 5213 apply to create and append the mobility header and options, except for the Home Network Prefix option in the PBU which is always set blank. The Home Network Prefix option in the PBA is filled with the HNP(s) that the MAR that is sending the PBU assigned to the MN when it was directly attached to it.
2. When the mobility event takes place outside the MEDIEVAL DMM domain, the signaling occurs between the UME in the MN and UME(s) residing in the MAR(s) using the Binding Update and Binding Acknowledgment messages defined in RFC 6275.

No extra options are defined for these messages, nor their format is modified. However, the rationale to fill the values in the options and to process them might be subjected to little variations with respect to the corresponding RFC, as specified in Section 4.1.3.

A.1.9 MUME_MR_If

A.1.9.1 MUME_MR_Join

Function

This message is used by MUME for triggering multicast trees subscription updates.

Semantics of the service primitive

```
MUME_MR_Join {
    Multicast_Subscriptions,
    Mobile_Terminal_Address
}
```

Parameter	Type	Description
Multicast_Subscriptions	SEQUENCE{SEQUENCE{STRING, IN6_ADDRESS, SEQUENCE{IN6_ADDRES S}}}	List of multicast channels (filter mode, group, source(s))
Mobile_Terminal_Address	IN6_ADDRESS	The IPv6 address of the multicast subscriber

Table 16. MUME_MR_Join parameter list

When generated

The message is generated when MUME was triggered by FM for joining multicast tree(s).

Effect on receipt

After receiving this message, the MR should join the specified multicast tree(s).

A.1.9.2 MUME_MR_Leave

Function

This message is used by MUME for triggering the prune of necessary multicast tree(s), avoiding leave latency problems.

Semantics of the service primitive

```
MUME_MR_Leave {
    Multicast_Subscriptions,
    Mobile_Terminal_Address
}
```

Parameter	Type	Description
Multicast_Subscriptions	SEQUENCE{SEQUENCE{STRING, IN6_ADDRESS, SEQUENCE{IN6_ADDRES S}}}	List of multicast channels (filter mode, group, source(s))
Mobile_Terminal_Address	IN6_ADDRESS	The IPv6 address of the multicast subscriber

Table 17. MUME_MR_Leave parameter list

When generated

The message is generated when the MUME, after checking its MMD, verifies the need to leave multicast tree(s) – i.e. the last MN subscribing a specific group or groups has left the local MAR.

Effect on receipt

After receiving this message, the MR should leave the specified multicast tree(s), by sending a PIM Prune message towards the corresponding upstream router(s).

A.1.10 MUME_MUME_If

This interface is necessary for exchanging either multicast or mobility-related information between two MARs, depending on the used scheme and on the multicast node role.

A.1.10.1 MUME_MUME_ExtendedProxyBinding

MUME_MUME_ExtendedProxyBinding.update

Function

This message is sent by the S-MAR to the A-MAR to request the MN’s multicast group state information and the establishment of a tunnel between them.

Semantics of the service primitive

```
MUME_MUME_ExtendedProxyBinding.update (
    Mobile_Terminal_ID,
    Link_ID,
    S_MAR_address
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of the interface where to receive multicast traffic.
S_MAR_address	IN6_ADDRESS	S-MAR address, one of 2 tunnel endpoints

Table 18. MUME_MUME_ExtendedProxyBinding.update parameter list

When generated

The message is generated by the S-MAR when it wants to request the MN’s multicast group state and to establish the tunnel for multicast traffic from the A-MAR (after receiving the FM_MUME_ActivateMobility.request from the FM with the MulticastRole parameter set to 0).

Effect on receipt

The A-MAR sends MLD Query(s) to get the MN’s MLD state. After receiving the MN’s MLD Report, it will reply to the S-MAR with a MUME_MUME_ExtendedProxyBinding.acknowledge with corresponding MN’s MLD state. The A-MAR also sets up its endpoint of the bi-directional tunnel to the S-MAR.

MUME_MUME_ExtendedProxyBinding.acknowledge

Function

This message is used to transfer the multicast state relative to the MN and indicate the S-MAR to complete the tunnel establishment.

Semantics of the service primitive

```
MUME_MUME_ExtendedProxyBinding.acknowledge (
    Mobile_Terminal_ID,
    Link_ID,
    Multicast_Subscriptions
    ResultCode
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of the interface where to receive multicast traffic.
Multicast_Subscriptions	SEQUENCE (SHORT_INT IN6_ADDRESS UINT_8 SEQUENCE (IN6_ADDRESS))	List of multicast records (filter mode, group address, number of sources, source(s) address(es))
ResultCode	UINT_8	The result of the protocol setup. (0 on OK otherwise return an error code)

Table 19. MUME_MUME_ExtendedProxyBinding.acknowledge parameter list

When generated

The message is generated by A-MAR after setting up its endpoint of bi-directional tunnel.

Effect on receipt

After receiving this message, S-MAR will set up the multicast status of the point-to-point link between the S-MAR and the MN as well as join the content identified by Multicast Address Records on behalf of the MN (in case the S-MAR is not receiving already the multicast traffic). It also sets up its endpoint of the bi-directional tunnel to the A-MAR.

A.1.10.2 MUME_MUME_Context_Transfer

The primitives supported by this interface are used for exchanging multicast group state information between MARs.

MUME_MUME_ContextTransfer.request

Function

Context transfer is used between the previous and current access routers for multicast flows involved in the video delivery, providing a fast multicast tree reconstruction and allowing an uninterrupted service.

Semantics of the service primitive

```
MUME_MUME_Context_Transfer.request (
    Mobile_Terminal_ID,
    Link_ID
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)

Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of the interface where to receive multicast traffic.
---------	-----------------------------------	---

Table 20. MUME_MUME_ContextTransfer.request parameter list

When generated

When a MAG-based scheme is used for supporting a multicast listener, such as no mobility is actually activated, the S-MAR communicates with A-MAR through means of context transfer protocol for exchanging the multicast profile information.

Effect on receipt

The receiving MUME sends MLD Query to get the MN's MLD state entries, and embeds it in a message to be sent to the requesting MUME.

MUME_MUME_ContextTransfer.response

Function

This message is used to transfer the multicast state relative to the MN.

Semantics of the service primitive

```
MUME_MUME_Context_Transfer.response (
    Mobile_Terminal_ID,
    Link_ID,
    Multicast_Subscriptions
    ResultCode
)
```

Parameter	Type	Description
Mobile_Terminal_ID	STRING	The identifier of the Mobile Terminal (usually an NAI)
Link_ID	SEQUENCE{LINK_TYPE, LINK_ADDR}	The identifier of the interface where to receive multicast traffic.
Multicast_Subscriptions	SEQUENCE (SHORT_INT IN6_ADDRESS UINT_8 SEQUENCE (IN6_ADDRESS))	List of multicast records (filter mode, group address, number of sources, source(s) address(es))
ResultCode	UINT_8	The result of Context Transfer. (0 on OK otherwise return an error code)

Table 21. MUME_MUME_ContextTransfer.response parameter list

When generated

After receiving the MN's MLD Report, the MUME at the A-MAR embeds MN's MLD state entries in this message and sends it to the S-MAR.

Effect on receipt

The receiving MUME joins the content identified by Multicast Address Records on behalf of the MNs (in case the S-MAR is not receiving already the multicast traffic).

B.2 MEDIEVAL demo poster used during the 83rd IETF meeting

Network-based Distributed Mobility Management demo



Universidad Carlos III de Madrid
www.uc3m.es





http://www.ict-medieval.eu/





Background and Motivation

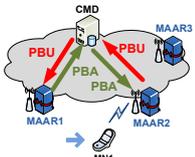
- The number of mobile users and their traffic demand is expected to be ever-increasing in future years
- This growth can represent a limitation for deploying current mobility management schemes that are intrinsically centralized, e.g., Mobile IPv6 and Proxy Mobile IPv6
- Distributed and dynamic mobility management (DMM) approaches aim at reducing operators' burdens, evolving to a cheaper and more efficient architecture
- This demo shows a full functional real-life prototype of a network-based DMM solution, based on:
 - Partially distributed DMM approach as described in draft-bernardos-dmm-pmip [1]
 - Distributed Logical Interface concept as described in draft-bernardos-dmm-distributed-anchoring [2]

Demo Description

- MN1 moves between the three different MAARs
- A different (locally anchored) prefix is delegated by each MAAR
- MN1 initiates new communications at each MAAR, while keeping old ones
- PMIPv6-based mobility signaling (CMD as MAAR proxy) as specified in [1]
- MAARs implement DLIF concept as specified in [2]
- Legacy IPv6 MN
- Legacy applications, enjoying IPv6 address continuity
- Simultaneous anchoring of flows at different MAARs
 - MN1 attaches to MAAR1 and starts a TCP video flow
 - MN1 moves to MAAR1, video keeps playing (data is tunneled between MAAR1 and MAAR2)
 - MN1 starts a new application (web browsing), using the IPv6 address anchored at MAAR2
 - MN1 moves to MAAR3, video keeps playing (tunnel endpoint is updated)

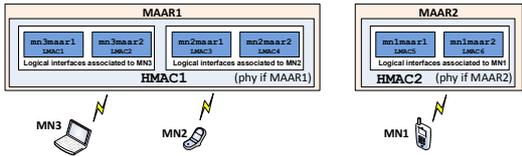
A PMIPv6-based solution for DMM

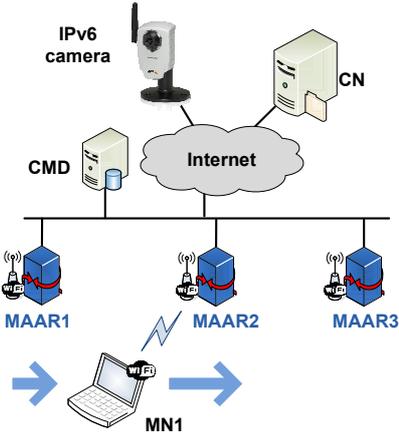
- Partially distributed approach for network-based DMM
 - Only data forwarding is distributed
- Entities
 - MAAR: Mobility Anchor and Access Router
 - First IP hop seen by the MNs connected on its access links. It runs LMA and MAG functionalities
 - CMD: Central Mobility Database
 - It stores the mobility sessions for the MNs in the domain
- 3 different signaling modes:
 - CMD as PBU/PBA relay
 - CMD as MAAR locator
 - CMD as MAAR proxy
 - CMD serves as a proxy
 - PBU/PBA signaling between MAARs and CMD (no direct MAAR-to-MAAR signaling)



The Distributed Logical Interface concept

- The Distributed Logical Interface (DLIF) is a software construct allowing to hide the change of anchor from the MN
- Each serving MAAR exposes itself towards a given MN as multiple routers, one per active anchoring MAAR associated to the MN
 - This is achieved is by the serving MAAR configuring different logical interfaces
 - From the point of view of the MN, the anchoring MAARs are portrayed as different routers, although the MN is physically attached to a single interface of the serving MAAR





- MAARs and CMD are Asus WL-500gP routers, running Linux (OpenWRT backfire) kernel 3.0.12 with mobility support
- CN and MN are Laptops/netbooks, running standard Linux (Ubuntu 10.04) with no mobility support
- An IPv6 camera is also used as CN
- DMM PMIPv6 implementation based on OpenAirInterface Proxy Mobile IPv6 (OAI PMIPv6) partially developed within the framework of the MEDIEVAL EU project
- DLIF concept implemented using `macvlan` support
- Source-based IPv6 routing, as well as tunnels (in case of handover) provided by the Linux Advanced Routing and Traffic Control features
- IEEE 802.11abg wireless access, without layer-2 handoff specific optimizations

References

[1] "A PMIPv6-based solution for Distributed Mobility Management", Carlos J. Bernardos, A. de la Oliva, Fabio Giust, Telemaco Melia, Rui Costa, draft-bernardos-dmm-pmip-01, March 2012, <http://tools.ietf.org/html/draft-bernardos-dmm-pmip-01>

[2] "PMIPv6-based distributed anchoring", Carlos J. Bernardos, Juan Carlos Zúñiga, draft-bernardos-dmm-distributed-anchoring-00, March 2012, <http://tools.ietf.org/html/draft-bernardos-dmm-distributed-anchoring-00>