EMICOM: Enhanced Media Independent COnnection Manager

André Prata*, Daniel Corujo*, Pedro Gonçalves†, Diogo Gomes*
ESTGA† / DETI*, Universidade de Aveiro
Instituto de Telecomunicações
3810-193 Aveiro, Portugal
{andreprata, dcorujo, pasg, dgomes}@av.it.pt

***Abstract*—With the increasing amount of mobile interfaces combining different kinds of access technologies, ranging from Wi-Fi to 3G and LTE, the integration of flexible and media-independent link control mechanisms becomes of paramount importance. By employing an abstract way of obtaining access link status information and exercising control over the network interface operations, these control mechanisms become able to optimize device connectivity and network attachment. This paper presents EMICOM, an Enhanced Media Independent COnnection Manager framework where a GNU/Linux Network Manager and Link Service Access Points for the IEEE 802.3 and 802.11 technologies were implemented and integrated through cross-layer Media Independent Handover (MIH) mechanisms from the IEEE 802.21 standard. Through an open-source implementation of the framework, the (MIH) command set capabilities are extended, allowing the support of network association and authentication, as well as Layer 3 services such as IP configuration, providing a generic solution for optimal network connectivity management.**

***Index Terms*—802.21, Network Manager, Link layer**

I. Introduction

Current mobile devices offer various interfaces for network connectivity, either for Local, Metropolitan and Cellular networks. This motivates network providers to look at heterogeneous network deployments as a mean of expanding network capacity through the combination of high bandwidth solutions, such as Wi-Fi, with the geographically broader 3G and LTE solutions. User mobility in these scenarios presents an outstanding problem of optimal network technology selection. Current network managing solutions often base selection algorithms on signal quality and user preference policies, while requiring the various radios at the terminal to be enabled, in order to be aware of available alternatives for network connectivity.

This paper presents a framework for controlling and gathering information about different mobile node access interfaces via an abstract interface provided by the IEEE 802.21 standard [1]. This framework enables network selection algorithms to take as input variables such as energy consumption, cost per technology, cost per bit, application and service constraints, as well as network provided information regarding the surrounding environment, allowing for an optimized access interface selection. It also extends the IEEE 802.21 services to provide abstract Layer 2 association and security mechanisms, and Layer 3 static or dynamic IP configuration, enhancing the capabilities envisioned in the standard.

The remainder of this paper is structured as follows. Section II introduces the state of the art on network manager solutions residing in terminals. This is followed by Section IV were our framework is described, alongside extensions done to IEEE 802.21, as well as open-source implementations of 802.3 and 802.11 interfacing modules. This framework was subjected to an evaluation scenario presented in Section V. Finally, we conclude in Section VI.

II. Related work

With the different requirements placed by different existing access technologies being used to access the most diverse kind of services available through the Internet, different attempts for the provision of inter-technology network selection mechanisms have been proposed, at different levels. Most application deployments target one of two scenarios: management based on user preferences, and management based on network enforced policies. In this article, we focus on the solutions that current Operating Systems (OSs) and terminals provide for managing their different connectivity options.

Many network providers offer their own network manager solutions, for seamless integration with their services. These solutions are not available across all existing Operating Systems, and often focus on the largest user base. Network managing functions composing these solutions usually aim at giving users the best possible access to the provider's network solutions, sometimes using proprietary services for hotspot location or traffic management. Nowadays it is typical to find mobile terminal network applications pre-installed by laptop manufacturers, provided by mobile operators as integrated software in USB dongles or even as native applications from Operating Systems. In the following sections, we illustrate some examples.

A. O2 Connection Manager

The O2 Connection Manager[1] is a Windows application for managing Internet connections. It attempts to provide connection to the fastest available networks, including the user's home broadband. It suggests connection to Wi-Fi hotspots from the

[1]O2 Connection Manager, http://www.o2.co.uk/support/broadbandinternet/o2connectionmanager

operator, and integrates with the operator's cellular dongle hardware supporting SMS services in the desktop application.

B. AT&T Communication Manager

The AT&T Communication Manager[2] is a desktop application for taking advantage of the operator's 4G network in the United States. It works in Windows and Mac OS, and also manages Wi-Fi connections, handling the authentication to AT&T Wi-Fi hotspots. The application provides real time data usage management, and allows the creation of mobile hotspots for sharing internet access with multiple Wi-Fi enabled mobile devices. In addition, the application uses the operator's cellular network, or integrated GPS chips on proprietary devices, to provide location services, namely for locating Wi-Fi hotspots.

C. GNU/Linux NetworkManager

The GNU/Linux NetworkManager[3] (NM) application is present in most, if not all, desktop GNU/Linux distributions. It aims to provide constant network connectivity, featuring IPv4 and IPv6 support, WEP, WPA/WPA2 and 802.1X security mechanisms, and the ability to control many network devices including Ethernet, Wi-Fi, WiMAX and 3G modems.

NetworkManager provides a D-Bus interface for Desktop Environment and GUI configuration interfaces. Users provide configurations for each network they want to include, and NetworkManager tries its best to keep the user connected, preferring secure connections first, then selecting Access Points with stronger signals. Moreover, it always attempts to acquire a connection with every available network interface, unless the user explicitly requests disconnection of an interface.

D. InterDigital Smart Access Manager

InterDigital's Smart Access Manager[4] (SAM) is a client for mobile devices that makes network selection and traffic management decisions between Wi-Fi, 3G and LTE networks. This is a solution for network providers that aims to provide terminal functions for Wi-Fi offloading and maintaining user Quality of Experience (QoE) on the provider networks via the Access Network Discovery and Selection Function (ANDSF). The client supports dynamic selection between hotspots, giving preference to user-configured Wi-Fi networks. It performs IP traffic routing between networks when both Wi-Fi and cellular access are available, thus enforcing provider policies on terminal devices.

E. Solution Comparison

The previous sections do not attempt at an extensive overview of the available solutions. Solutions from service providers often fail to address multiple network technologies, since they focus mostly on their provided services, and usually Wi-Fi for offloading [2]. Generic solutions, on the other hand, do not provide much information or special features regarding interface and network selection. Of the mentioned solutions, only SAM uses a mechanism for requesting help from the network for attachment decisions. Table I provides a comparison of the various features for each solution.

SAM is considered a Multi-OS solution since it supports various smartphone devices, but it is not intended for desktop usage. Although also listed as non-Multi-Operator, it probably depends on the configuration that each operator requires for the deployment of the software.

	O2	AT&T	NM	SAM
Multi-OS	no	yes	yes	yes
Multi-Technology	yes	yes	yes	yes
Multi-Operator	no	no	yes	no
Dynamic Optimizations	no	no	no	yes
Network decisions	no	no	no	yes
Open Source	no	no	yes	no

Table I
SOLUTION FEATURES COMPARISON.

III. SUPPORTIVE TECHNOLOGIES

Considering the previous shortcomings presented by the network manager solutions illustrated in Section II, we introduce in this section a set of supporting mechanisms which, associated to those network managers, would complement them with the lacking behaviour.

A. IEEE 802.21

The IEEE 802.21 [1] standard enables seamless handover between heterogeneous technologies. This framework is based on a protocol stack implemented in all the devices involved in the handover. It exposes a common L2 interface for IEEE 802 and 3GPP technologies in order to facilitate network handovers, thus a Media Independent Handover (MIH) framework. It considers several network components:

- **Mobile Node (MN):** This is the network attachment candidate itself.
- **Point of Attachment (PoA):** The network endpoint for L2 connection to the MN.
- **Point of Service (PoS):** A network entity that provides information to the MN and takes network configuration actions in order to optimize MN handovers.
- **Non-PoS:** An IEEE 802.21 entity that contains static information regarding network policies and topologies.

These entities are interconnected by an MIH Function (MIHF) through the MIH Network Service Access Point (SAP) interface. It also handles the communication exchanges between the lower layer entities, through the MIH Link SAP and higher layer entities (MIH Users), through the MIH SAP. The MIH SAP provides MIH Users with a large set of technology-independent primitives for interface control and information gathering. The MIH Link SAP provides a mapping between the 802.21 primitives and the technology-specific lower layer interfaces. The MIH primitives are grouped in the Event, Command and Information Services.

[2]AT&T Communication Manager, http://www.wireless.att.com/businesscenter/solutions/wireless-laptop/communication-manager/index.jsp

[3]GNU/Linux Network Manager, http://projects.gnome.org/NetworkManager/

[4]InterDigital Smart Access Manager, http://www.interdigital.com/smart-access-manager

1) Event Service: Events can be initiated either by the MN or by the network. They originate from lower layers or MIHF, at the MN, at the network PoA or the PoS, and are propagated to interested MIH Users, local or remote. MIH Users declare their interest in certain events by subscribing them. MIH events pertain to the following factors:

- **Link State Change events:** This includes events to notify of link layer occurrences such as the loss or establishment of L2 connectivity.
- **Link Parameter events:** These events are generated in response to configured thresholds pertaining to link-layer parameters such as packet loss, signal strength, etc.
- **Predictive events:** Such events convey the likelihood of a change in the link conditions in the near future based on past and present conditions, such as the decay in signal strength in relation to a PoA.
- **Link Transmission events:** These events can be subscribed to receive indication of the link layer transmission status of individual upper layer PDUs.

Events are mostly advisory in nature, which means Users subscribing to a set of events are not required to act on them.

2) Command Service: MIH Users utilize command services to determine the status of links and/or control the device for optimal performance. Commands can be delivered locally or remotely, and are classified in two main categories: MIH Commands and Link Commands.

Link commands are delivered from the MIHF to the Link SAPs, on behalf of the MIH Users, for various control operations, through the following primitives:

- **Capability discovery and event subscription:** A link may be queried in order to determine its supported primitives both for the command and event service. Link events must be subscribed for receiving notifications on a per-link basis.
- **Parameter retrieval:** Various active link parameters can be obtained through Link commands, such as the current bit rate, Quality of Service (QoS) statistics, signal strength, etc.
- **Threshold configuration:** Link parameter events can be configured for future reception over the event service. Thresholds may be configured for obtaining periodic reports or only indications that a parameter value has been crossed.
- **Link control:** A single primitive offers various actions to request a link to perform, including radio scans, changing its operational state or going into power save mode.

MIH Commands are sent by the higher layers to the MIHF. They may originate locally, through the MIH SAP, or remotely, through the MIH Network SAP. MIH Commands may translate into specific Link Commands operations, or aid in handover decision procedures by the following requests:

- **Handover candidate query:** Both the MN and the Network may issue this command in order to exchange suggested networks and associated points of attachment information for possible handovers.
- **Query resources:** This command is used to assess or prepare network resources in a target network for MN handover.
- **Handover commit:** For MN controlled handovers, this is used to inform the serving network of the target decision. For Network controlled handovers, this command informs the MN of which network to attach to.
- **Handover completion:** This command allows both serving and target networks to indicate the completion status of a handover operation.

3) Information Service: The Information service is a collection of mostly static information elements about networks and operators. These elements provide information essential to the network selection algorithm to make a successful handover across heterogeneous networks and technologies. A Mobile Node benefits mostly from this service by being able to query geographically surrounding networks, as well as their service capabilities before making a handover decision or even powering other radio interfaces. For example, information about a nearby Wi-Fi hotspot could be obtained using a 3G interface without the need to power the Wi-Fi radio. It may also be used to query target network information regarding security or QoS mechanisms, thus influencing the target selection.

4) Media Specific Mappings for SAPs: The MIHF aggregates disparate interfaces with respective media dependent lower layer instances into a single abstract interface for MIH Users, reducing the inter-media differences to the extent possible. For the most part, existing primitives and functionality provided by different access technology standards are used. Amendments to existing standards are proposed when deemed necessary to fulfil the MIHF capabilities.

The Link Service Access Point (LSAP), defined in the IEEE 802.2 [3] standard, provides the interface between the MIHF and the Logical Link Control (LLC) sublayer across both IEEE 802.3 and 802.11 networks. However, the complete MIH_LINK_SAP set of primitives mappings for 802.11 requires an enhancement proposal defined in IEEE 802.11u [4], in the form of the MAC State Generic Convergence Function (MSGCF).

The interface for 802.16 networks depends on the C_SAP and M_SAP, both defined in IEEE 802.16g [5].

A special SAP, MIH_3GLINK_SAP, is defined for interfacing with the MIHF and the different protocol elements of the 3G system.

B. Converging decision processes

With different kinds of access technologies available to multi-interface mobile terminals, achieving an optimal handover decision depends on multiple parameters [6], ranging from:

1) The dynamics of the wireless strata (e.g., signal-noise ratio, available bandwidth, cell load);
2) Requirements placed by the service content being accessed (e.g., minimum latency);
3) Requirements placed by the user (e.g., perceived video and/or audio quality, cost);

4) Network conditions (e.g., cell load, requested service, policies).

The different criteria involved must not only take into consideration the capabilities of the service being provided, but also the resources available in the network and, ultimately, the user satisfaction. As such, optimal handover decisions have the need to assess different objectives, from different layers of the network stack, in order to achieve an Always Best Connected [7] solution.

In this sense, different schema are possible, varying between mobile terminal centric decisions [8], and network controlled decisions [9], or even combinations of both where the perspective of the terminal and network come together to optimize the handover decision to a broader set of requirements [10].

However, a common requirement for optimized handover processes relies on the flexibility and simplicity of network manager application design, contributing for the facilitated deployment of such mechanisms in different kinds of mobile terminals using a broad spectrum of access technologies. As such, in this work, we consider the integration of Media Independent mechanisms within the fabric of network manager applications, aiming at mainly two things: abstracting information and control capabilities from different kinds of access interfaces; and empowering such applications with the means to disseminate that information and control with local decision entities, as well as network-controlled remote entities when available. This constitutes the setting for our Enhanced Media Independent COnnection Manager framework, EMICOM, presented next.

IV. Framework

The developed framework is based on an open source MIHF implementation, ODTONE[5], supporting the whole range of MIH services. It provides a set of Application Programmable Interfaces (APIs), enabling the development of custom MIH Users and Links to interface with the provided MIHF. These APIs are based on socket message transport, which enables the reuse of the solutions between multiple Operating Systems.

The mechanisms presented by the IEEE 802.21 standard can be considered as an abstraction for Media Independent interface management. Not only this common interface provides easier interface control, it also exposes common information enabling network managers to employ a plethora of new network selection algorithms based on application and service requirements, power constraints, and other requirements. However, the 802.21 services do not provide all the necessary primitives for interface management. For example, the procedures for network association are outside of the scope of the standard. Also, since it aims to provide an abstraction only at the link layer, there is no support for network layer configuration.

This section introduces our network manager framework by detailing how such concepts can be integrated with 802.21, as well as identifying key enhancements done over the base standard, enhancing its functionality.

[5]ODTONE, Open Dot Twenty ONE, http://atnog.av.it.pt/odtone

A. MIHF Cross-layer

The open-source nature of the ODTONE software collection allowed an extension to the 802.21 protocol in order to keep the advantages of interface agnostic control, and thus not requiring specific hardware or OS support for operation. The proposed extension refers to operations within the Mobile Node only.

Association and security procedures require link layer support from supplicant software. IP configuration requires assigning IP addresses to interfaces, as well as default gateways, custom routes, and DNS servers. As such, for these tasks, two additional Command Service primitives were integrated into the 802.21 mechanics, for both the MIH SAP and MIH Link SAP, providing the following functionality:

- **Link configuration:** Attach to a given network, providing the necessary authentication, association and security information.
- **L3 configuration:** Configure a set of networking properties on an interface, such as IP address, static routes and list of DNS servers.

Network authentication protocols such as 802.1X [11] provide many authentication mechanisms, some requiring an indefinite number of exchanges between the supplicant and the authentication server (i.e., depending on deployment scenarios), and possibly demanding user input. This can be interpreted as a network request for the user, and can be implemented through the 802.21 Event Service. The following primitive was added to the 802.21 mechanics:

- **Link configuration required:** Indicates a network request for authentication material from the supplicant.

Using these commands, MIH Users are abstracted from specific protocols such as DHCP or stateless auto-configuration. An MIH User requesting Layer 3 configuration may request attribution of specific IPv4 or IPv6 addresses, but it may as well ask the framework to attempt DHCP configuration, for example. The same is true for the Link configuration primitive; the required parameters for the specific network are provided immediately in the Link configuration request. The framework then implements the architecture of Figure 1.

In this sense, our Network Manager assumes the role of MIH User, interfacing with an enhanced MIHF implemented over the ODTONE open-source software, thus empowering it to not only access different kinds of link layer technologies in a media-independent way, but also to abstract security association and address requesting processes.

B. Link Interfaces Service Access Points

Supporting this framework requires implementing Link SAPs for various network device technologies. These components require direct communication with the OS software or interface drivers, whose implementation is platform dependent. For the specific case of GNU/Linux, two Link SAP implementations were developed, and added as open-source software to the ODTONE project, for the IEEE 802.3 and 802.11 technologies.

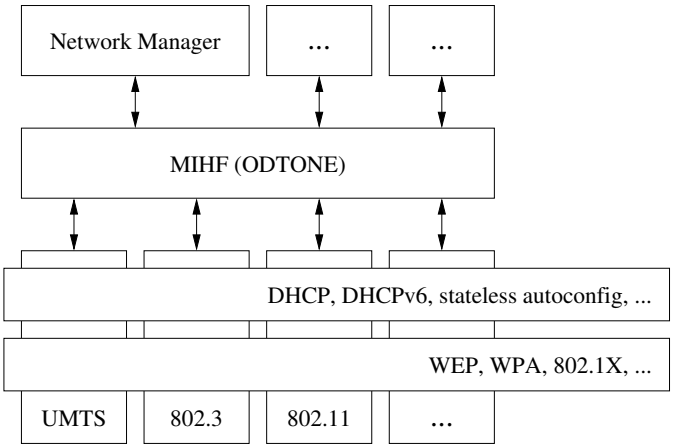

Figure 1. EMICOM framework architecture.

The Linux kernel offers two main interfaces for network device control, both over the Netlink [12] protocol: Route Netlink and the *nl80211* family over the Generic Netlink protocol. These interfaces provide some simple link control functionalities exposed by the Linux kernel, which were exploited in our work to produce the LINK SAPs. The Route Netlink interface enables access to the networking subsystem of the Linux kernel by providing a set of primitives for adding, getting and removing information from its internal table-oriented data structures. For security mechanisms support, and dynamic IP configuration, external tools are also used.

1) Route Netlink: This interface provides the full mapping for 802.3, and a partial mapping for the 802.11 Link SAP, according to 802.21 primitives and parameters. It offers three groups of messages of particular interest, for manipulation of different aspects common to all Linux network interfaces:

- **LINK messages:** Allow a user of the Route Netlink protocol to manipulate information about network interfaces on the system. These messages contain attributes for indicating link state and device power changes.
- **ADDR messages:** Used for manipulation of IP addresses of the network interfaces. These messages support IPv4 and IPv6 addresses, and an interface can be assigned multiple IP addresses.
- **ROUTE messages:** These messages baptise the whole protocol name and, as it points out, they refer to IP routing table management.

These messages are used to get or modify attributes of a link, but can also be subscribed in order to receive notifications when the kernel signals changes to these objects.

2) nl80211: This protocol offers an interface for the Linux *mac80211* framework, for Wi-Fi specific operations. Among others, the following mechanisms are exposed:

- **Power control:** In addition to the various interface states offered by the Route Netlink protocol, the nl80211 interface allows controlling a Wi-Fi interface's power save mode for sleeping between Access Point (AP) beacons, as defined in the 802.11 standard.
- **Scanning:** The interface supports triggering scans at any moment as well as the configuration of scheduled scans at regular intervals. The scan result provides a listing of the detected APs containing basic BSS attributes, as well as the full listing of the beacon Information Elements.
- **Association and Authentication:** Linux supports various 802.11 authentication methods. For security mechanisms, there are facilities for userspace applications to transmit raw packets as well as subscribing received packets by means of matching the first few bytes of the desired frames.
- **QoS:** The *mac80211* framework supports the 802.11e [13] extension, and will assign frames to specific hardware queues based on the Type of Service (TOS) field of the packet IPv4 header. IPv6 does not support the TOS field and uses Traffic Class codes instead. Unfortunately, current versions of the Linux kernel do not provide statistics such as minimum, average and maximum packet delay for individual Classes of Service, which is one of the features of the 802.21 framework.

Similarly to the Route Netlink interface, *nl80211* allows the subscription of certain occurrences. This includes events for L2 connection and disconnection, but also the signalling of new scan results and the crossing of a given signal strength threshold.

For easier Netlink message parsing and program memory management, both the Route Netlink and *nl80211* access are mediated by a custom C++ wrapper on top of the *libnl*[6] library.

3) Security Mechanisms: Support for network authentication mechanisms such as WPA2 and 802.1X is achieved with the *wpa_supplicant*[7] software. *wpa_supplicant* provides a text-based socket interface as well as a D-Bus API. The Link SAPs interface with *wpa_supplicant* via the D-Bus API, which supports adding network configurations and keying material at runtime. Requesting authentication to a network can be performed immediately or later. If the authentication mechanism requires further parameters, the daemon signals the occurrence of network requests indicating the required fields.

4) IP Configuration: Dynamic IP configuration is supported via the Internet Systems Consortium's DHCP[8] software bundle. It provides a DHCP client, *dhclient*, supporting DHCPv4 and DHCPv6, as well as IPv6 Stateless Address Autoconfiguration. Unfortunately, this program does not offer any sort of API, and its integration is done via scripted calls to the command line daemon, providing the appropriate parameters as command line options each time. DNS servers are configured by direct manipulation of system configuration files.

C. Network Manager

The aim of the Network Manager development was to replace the existing NetworkManager software which is inte-

[6]libnl libraries, http://www.infradead.org/~tgr/libnl/
[7]WPA Supplicant daemon, http://hostap.epitest.fi/wpa_supplicant/
[8]DHCP Client, https://www.isc.org/software/dhcp

grated with most desktop environments via network configuration GUIs and panel applets via a freedesktop.org[9] endorsed D-Bus API. This API includes the following interfaces for Desktop applications:

- **NetworkManager:** A main central object that offers an interface for overall management tasks. This object holds the network device structure, the overall machine state and individual connection states.
- **Device:** This is an abstract interface to represent common attributes of all network devices. An object implementing this interface is associated to a unique network device in the system. Each technology extends this interface for providing technology-specific features. The **Device.Wireless** interface, for example, provides a D-Bus API for requesting scans, list of known APs, etc.
- **AccessPoint:** An AccessPoint object exposes properties to identify networks and determine how to properly prepare the association to the BSS, such as the SSID, Frequency, Maximum bitrate, etc.
- **Settings:** The Settings interface may be regarded as the system-global repository for the various configured networks. It exposes methods to add and retrieve configuration objects, containing the necessary information for each network, including authentication material, IP configurations (whether dynamic or static), etc.

The services exposed by the NetworkManager D-Bus interface, otherwise implemented directly for each specific hardware implementation, must be converted to the MIH Services primitives. In essence, regardless of the Device type exposed by the D-Bus API, the underlying object for interfacing through the MIHF is the same for every type. This base object uses a reduced set of primitives for media independent device control:

- **Link actions:** Used to initiate scanning, disconnecting a link, and setting the device power state between on, off and power save mode.
- **Link configure thresholds:** Used to configure threshold parameters for future event notifications.
- **Link get parameters:** Retrieve certain parameters of the currently established link.
- **Link configuration:** Used to request association and setup of security mechanisms to a network.
- **L3 configuration:** Used to request IP configuration on a link.

Similarly, reaction to device events is supported via the set of primitives of the MIH Event service:

- **Link up:** L2 connection is established on a link.
- **Link down:** L2 connection is lost on a link.
- **Link detected:** A PoA of a new network was detected following a scan.
- **Link parameters report:** Events generated regarding configured thresholds.
- **Link configuration required:** Additional procedures are required for security mechanisms.

[9]freedesktop.org, http://www.freedesktop.org/

The MIH User, and the D-Bus interface, should be extended with further intelligence and algorithms for network selection decisions, depending on the target environment.

V. Evaluation

The first aspect that needs to be evaluated from this abstract access to the interfaces is the framework footprint in a system. Since it is a multi process solution that relies on Inter Process Communication (IPC) mechanisms, there is an obvious overhead in communication. It is also relevant to compare other aspects such as the amount of code that composes the framework, in comparison with media dependent solutions, as well as the memory consumption and other aspects such as battery drainage, which are important in mobile devices. The following sections provide a comparison with the existing GNU/Linux NetworkManager application, as well as view of various different scenarios that distinguish the EMICOM framework from other solutions.

A. Test Setup

The tests were run in a laptop computer with the specifications defined in Table II. The testbed for network experiments contains two wireless Linksys WRT54G Access Points with the *DD-WRT*[10] firmware, connected by Ethernet to a video server machine, as depicted in Figure 2.

Component	Value
Operating System	Archlinux
Kernel version	3.5.4
Processor	Intel Core i7 M620 ($2 \times 2.67GHz$)
Memory	$4GB$ at $1333MHz$
Ethernet card	Intel PRO/1000 CT
Wi-Fi card #1	Intel Centrino Advanced-N 6200
Wi-Fi card #2	ASUSTeK WL-167g

Table II
Computer attributes.

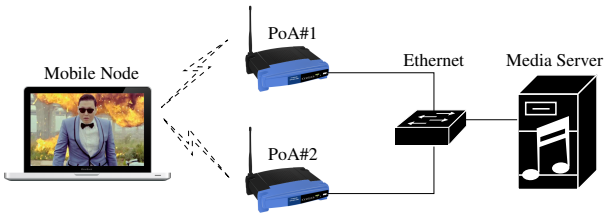

Figure 2. Testbed architecture.

B. Inter process overhead

In practice, the proposed solutions adds a layer to the existing kernel interfaces for network management, which abstracts the media dependent control. Communicating through this layer introduces an overhead that causes delays between operations, and implies data transmission between processes, at a cost.

[10]DD-WRT firmware, http://www.dd-wrt.com/

The 802.21 standard defines data transmission between remote entities via transport protocols such as UDP and TCP, encoding the necessary information in the Type-Length-Value (TLV) format. ODTONE uses this format for local transmission over transport sockets as well, allowing it to achieve the ability of being OS-independent, but at an obvious cost. Communication between the higher and lower layers require transmitting from the MIH User to the MIHF, then from the MIHF to the Links; answers traverse the inverse path. Messages from the event service travel only from the Link SAP to the MIH User, via the MIHF as well. Communication with remote entities is a necessary overhead and does not account for IPC analysis. Table III shows the number of MIH messages exchanged for various situations, including the total payload of transmitted data.

Operation	# of Messages	Total payload (bytes)
Power DOWN	4	162
Power UP	4	164
L2 Connect (simple)	4	197
L2 Connect (WPA + EAP)	4	508
L3 Configure (DHCP)	4	193
Disconnect	4	162
Link Detected event	2	144
Link Down event	2	80
Link Up event	2	102

Table III
MIH MESSAGE SIZES.

From these values it is visible that Link events take the fewest amount of bytes required (between 80 and 144 bytes). Link commands demand more information (between 162 and 197 bytes), but it is the new more complex commands, where the DHCP and security association and capabilities have been added to the standard 802.21 behaviour, that require the most amount of information (between 193 and 508 bytes, respectively).

The delay for the transmission of these messages is not analyzed, since the performance is internal and highly dependent on the load and capacity of each system. However, [14] shows that, for message sizes of up to 512 bytes, a low end (by today's standards) Linux machine delivers a rate of over 70 000 messages per second, translating to just $14\mu s$ per message. Furthermore, [15] shows a great performance improvement in transfer rates for Linux IPC by using UNIX domain sockets instead of UDP. ODTONE does not support UNIX sockets yet but, given its open source nature and the message-oriented IPC mechanism, the support should be implemented with little effort.

C. Code base

When compared to a native solution, the extra layer for media abstraction requires a greater amount of code for translating operations. However, the higher layers require less code for controlling individual interfaces, since the procedures are reused across device technologies.

Table IV offers a direct comparison between the provided framework and the Linux NetworkManager (version 0.9.6.0) solution, using the *SLOCCount*[11] tool. Despite being developed in different programming languages (C vs. C++), the number of code lines is nonetheless an acceptable measure of development effort. It should be noted that not the entire codebase of NetworkManager is considered; the counting excludes NetworkManager components not yet included in the EMICOM framework such as automatic VPN setup, Bluetooth and WiMAX support, etc.

NetworkManager	EMICOM		
Total: 70 815	MIHF (ODTONE)	11 606	Total: 21 960
	MIH User	5 277	
	802.11 Link SAP	1 610	
	802.3 Link SAP	897	
	libnl wrapper	1 347	
	dhclient wrapper	77	
	wpa_supplicant wrapper	1 146	

Table IV
CODE BASE COMPARISON, IN NUMBER OF SOURCE CODE LINES.

It is clear that the whole EMICOM framework, providing the same features as the considered for the NetworkManager software, requires less than a third of the source code. Several reasons contribute to this fact, the most prominent being the different programming languages. Other factors include the used libraries. ODTONE, and EMICOM, are highly dependent on the *boost* libraries[12] for data manipulation, which could also be a relevant contribution to the decrease in code size.

D. Memory usage

Process memory usage is a common limiting factor in some deployment scenarios. Embedded systems usually have limited memory. Even in desktop computers, it is desirable that resident applications account for a small impact on the overall system capacity.

Measuring process memory usage in modern Operating Systems is a complex task. Processes commonly make use of system libraries that, once loaded into memory, can be reused several times by several processes, and thus the system does not allocate multiple instances of the library. These libraries can be considered components of a program, but the program may not be the sole responsible for loading the library into the memory.

The *Valgrind*[13] utilities allow developers to track memory allocations of individual processes. This utility can accurately report the memory that each process allocates both in the Heap and Stack memory segments. Table V shows the size of a snapshot of the combined Heap and Stack memory allocated by each solution, captured after an initial launch, after the attachment to both an 802.3 and 802.11 network (for this specific test, the computer is also attached by Ethernet, not represented in Figure 2).

Again, comparing similar situations for both solutions, the EMICOM framework shows a great benefit, compared to

[11] SLOCCount tool, http://www.dwheeler.com/sloccount/
[12] Boost Libraries, http://www.boost.org
[13] Valgrind utilities, http://valgrind.org/

NetworkManager	EMICOM		
Total: 967 064	MIHF (ODTONE)	35 688	Total: 553 496
	MIH User	401 192	
	802.11 Link SAP	52 200	
	802.3 Link SAP	64 416	

Table V
MEMORY USAGE BY EACH SOLUTION, IN BYTES.

the NetworkManager software. NetworkManager is openly developed, and has existed for a long time. Apart from the base ODTONE library, the EMICOM software has not been reviewed by external developers, and has not been submitted to optimization procedures, which means there could still be improvements in this area. It should be noted that the *boost* libraries do not directly contribute to this factor, since they are mostly header-only, thus not loaded as shared system libraries.

E. Benefits

The real world benefit for this framework is the plethora of scenarios where it may be considered for network selection and handover optimization. One of the main aspects that benefit a system with an 802.21-based networking solution is the Information Service, that will allow obtaining information about neighboring networks without powering additional radios. This service also allows the exchange of many network configurations and policies that will allow decision algorithms to take into account variables such as the cost for each network, the throughput or delay requirements for each application, and much more.

1) Battery life: This is increasingly relevant, as more and more mobile devices provide multiple radios for multiple network technologies. Figure 3 shows two different test runs, where a single laptop computer is retrieving a video stream via a Wi-Fi interface at a fixed rate of $500KB/s$. In one test run, the laptop is running the GNU/Linux NetworkManager, and the second is using the EMICOM framework. Both have one Ethernet interface and two Wi-Fi interfaces. This need not be the case, as there should be little benefit in having two similar radio interfaces, but it serves to compare the impact of having more than one wireless device in the same system. Moreover, due to EMICOM's usage of 802.21 abstraction mechanisms, the same events and commands used in this scenario would still be valid in scenarios featuring other technologies such as 802.16 and 3GPP links.

A regular NetworkManager typically employs very basic and static connectivity strategies, which implies having all the devices active at all times. In this specific scenario, we enhanced the base behaviour by allowing the second device to not be active at all times, but instead waking up at regular intervals of 30 seconds (halving in frequency every half hour) and performing a scan. A 802.21 solution, however, does not need to power the secondary device for learning about neighbouring networks, because it can rely on the Information service for the task of finding neighbouring networks, so in this scenario the MIH User always keeps the second device off, only activating it when an optimized handover opportunity occurs.

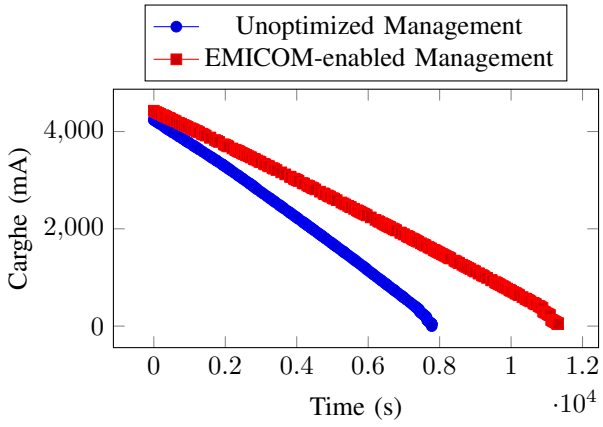

Figure 3. Battery drain comparison.

In the collected results, it is clear that the EMICOM run starts with higher initial capacity. This is common, as batteries often report different maximum capacity values at each charge cycle. The impact of powering an additional interface and performing scans is nonetheless noticeable and very significant, reducing the total autonomy of the device by a factor of 30%. Interestingly, though, the chart does not portray the increase in scan intervals after each 30 minute period. The initial interval of 30 seconds is perhaps a low initial value, resulting in maximum scan interval of 8 minutes in the experiment, although this would be an acceptable value for connectivity in a high mobility scenario.

2) Optimal selection: Figure 4 provides a 60 second test scenario where the EMICOM tool benefits from information from an MIH User in the Network, assisting with the handover decision. In this specific test, two Wi-Fi APs offer access to the same network. The computer is trying to maximize its TCP throughput by retrieving an on-demand video from the server. One AP has a stronger signal than the other ($-23dBm$ versus $-39dBm$), but it offers a lower throughput. This could be because the stronger AP is serving a greater amount of users, or has a low downlink, or many other reasons. In the specific test case, the rate was throttled at the AP on purpose. The GNU/Linux NetworkManager always stays connected to the strongest AP, since it only bases its connectivity decision on signal level. EMICOM however, using 802.21, receives information pushed by the network (e.g., via an *MIH Net Handover Commit request* command) to the user after 30 seconds, suggesting a handover to the other AP. This enables better load balancing on the network, while directly benefiting the user service.

A period with total loss of connectivity is noticeable. This is the occurrence of the hard handover, since the integration with IP mobility management engines achieving seamless mobility are out of scope from this paper. However, EMICOM, is able to leverage from mobility management primitives provided by 802.21, when such IP mobility schemes are employed.

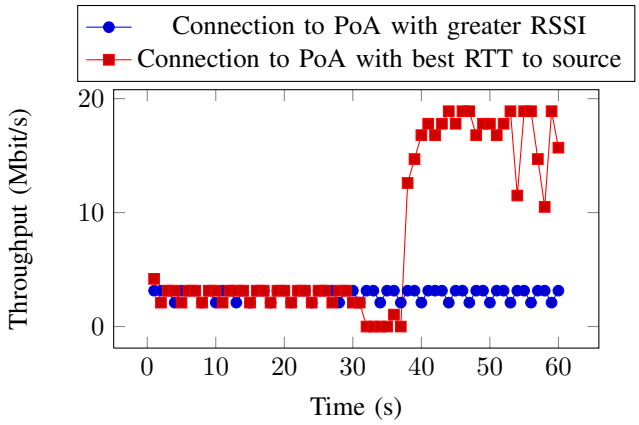

Figure 4. Optimal AP selection for throughput.

Nonetheless, via the stimuli provided by our 802.21-enabled Network Manager, the EMICOM framework was able to enhance the throughput of the video reception by performing a link switch to an AP with better downlink connectivity.

VI. Conclusion

The increase in connectivity opportunities, in terms of access technologies, to reach for different kinds of on-line content, places stringent requirements in terms of optimized connectivity and interface selection. Network Manager applications are becoming standard in different Operating Systems, but lacks the flexible capabilities for abstract access technology interfacing as well as disseminating and receiving handover optimization information from other sources, both local to the node or remotely available in network controlling entities.

This paper overthrew those shortcomings by integrating the Media Independent mechanisms of the IEEE 802.21 standard into Network Manager application concepts, defining EMICOM, an Enhanced Media Independent COnnection Manager framework.

This framework, more than just integrating 802.21 with Network Manager functionality, extended the first with the support for security association and address negotiation procedures, which are invaluable in today's seamless connectivity procedures in mobile networks. Moreover, it contributed to the open-source community by making available such extensions over ODTONE, an open-source implementation of the IEEE 802.21 standard. To that end, a set of Link Service Access Points, enabling ODTONE to interact with 802.3 and 802.11 interfaces in the GNU/Linux operating system, were also contributed to the project. Indeed, the 802.11 Link Service Access Point has been adopted by the FP7 ICT MEDIEVAL project[14], increasing the derived project with media independent control to Wi-Fi interfaces, as well as providing a complex testing environment for the work presented here.

The implementation of EMICOM also allowed the realization of an extensive evaluation effort, providing insight on the benefits of using Media Independent information and control capabilities to assist optimized handover and interface selection. Obtained results were compared to a popular NetworkManager from the GNU/Linux Operating System, showing a reduced code base, better battery consumption and allowing optimized handover procedures for opportunistic network attachment.

As future work, the evaluation of the proposed framework does not attempt to provide a performance analysis on handover operations, or even provide a full featured solution for the network management problem. Currently, these mechanisms are being evaluated under the scope of the MEDIEVAL project, where much other information can be used by a Network Manager MIH User in order to achieve the best connectivity, depending on user policies.

[14]MEDIEVAL, http://www.ict-medieval.eu/

Acknowledgment

This work has been partially funded by the European Community's Seventh Framework Programme (FP7-ICT-2009-5) under grant agreement n. 258053 (MEDIEVAL project).

References

[1] IEEE Standard for Local and Metropolitan Area Networks- Part 21: Media Independent Handover. *IEEE Std 802.21-2008*, 2009.

[2] de la Oliva, A. and Bernardos, C.J. and Calderon, M. and Melia, T. and Zuniga, J.C. IP flow mobility: smart traffic offload for future wireless networks. *Communications Magazine, IEEE*, 49(10):124 –132, oct. 2011.

[3] Logical Link Control. *ANSI/IEEE Std 802.2-1985*, 1984.

[4] IEEE Draft Std 802.11u WLAN MAC and PHY Amendment 7: Interworking with External Networks. *IEEE Unapproved Draft Std P802.11u/D5.0, Feb 2009*, 2009.

[5] IEEE 802 Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems - Amendment 3: Management PLANe Procedure and Services. *IEEE Std 802.16g 2007 (Amendment to IEEE Std 802.16-2004)*, 2007.

[6] Abid, M. and Yahiya, T.A. and Pujolle, G. A Utility-based Handover Decision Scheme for Heterogeneous Wireless Networks. In *Consumer Communications and Networking Conference (CCNC), 2012 IEEE*, volume , pages 650 –654, jan. 2012.

[7] Gustafsson, E. and Jonsson, A. Always best connected. *Wireless Communications, IEEE*, 10(1): 49 – 55, feb. 2003.

[8] Kassar, M. and Achour, A. and Kervella, B. A mobile-controlled handover management scheme in a loosely-coupled 3G-WLAN interworking architecture. In *Wireless Days, 2008. WD '08. 1st IFIP*, volume , pages 1 –5, nov. 2008.

[9] Bertin, P. and Guillouard, K. and Rault, J.-C. IP based network controlled handover management in WLAN access networks. In *Communications, 2004 IEEE International Conference on*, volume 7, pages 3906 – 3910 Vol.7, june 2004.

[10] Corujo, D. and Guimaraes, C. and Santos, B. and Aguiar, R.L. Using an open-source IEEE 802.21 implementation for network-based localized mobility management. *Communications Magazine, IEEE*, 49(9):114 – 123, september 2011.

[11] IEEE Standard for Local and metropolitan area networks - Port-Based Network Access Control. *IEEE Std 802.1X-2010 (Revision of IEEE Std 802.1X-2004)*, 2010.

[12] J. Salim, H. Khosravi, A. Kleen, and A. Kuznetsov. Linux Netlink as an IP Services Protocol. RFC 3549 (Informational), July 2003.

[13] IEEE 802 Part 11: WLAN MAC and PHY Amendment 8: MAC QoS Enhancements. *IEEE Std 802.11e-2005 (Amendment to IEEE Std 802.11, 1999 Edition (Reaff 2003)*, 2005.

[14] Brian F. G. Bidulock. STREAMS vs. Sockets Performance Comparison for UDP. *OpenSS7*, 2007. Accessed Oct. 2012, http://www.openss7.org/papers/strinet/testresults.pdf.

[15] Kwame Wright and Kartik Gopalan and Hui Kang. Performance Analysis of Various Mechanisms for Inter-process Communication, 2007. Accessed Oct. 2012, http://osnet.cs.binghamton.edu/publications/TR-20070820.pdf.