A Layered Infrastructure for Mobility-Aware Best Connectivity in the Heterogeneous Wireless Internet

Paolo Bellavista, Antonio Corradi, Carlo Giannelli
Dip. Elettronica, Informatica e Sistemistica (DEIS) - Università di Bologna
Viale Risorgimento, 2 - 40136 Bologna - ITALY
Ph.: +39-051-2093001; Fax: +39-051-2093073

Email: {pbellavista, acorradi, cgiannelli}@deis.unibo.it

ABSTRACT

The common availability of wireless devices with multiple communication interfaces, e.g., IEEE 802.11, WiMAX, Bluetooth, and/or UMTS, is pushing towards the necessity of novel supports to seamlessly select the proper connectivity technology to exploit at any time. That selection should be context-dependent and consider several aspects, at very different abstraction layers, from application-specific bandwidth requirements to expected client mobility, from connectivity costs and energy consumption to user preferences. We claim the need of effective mobility-aware middleware solutions to relieve application logic from the burden of determining the most suitable interface and connectivity provider for each client at runtime. In particular, we claim that such middleware supports should be structured according to a two-layer architecture: a lower-layer facility to retrieve available interfaces and connectivity providers and to discard unsuitable ones with a per-node decision, e.g., to reduce power consumption; and a higher-layer facility to select the currently most suitable connectivity provider in a per-application way. The paper describes the design and implementation of our novel middleware built according to those architecture guidelines: that permits to clearly differentiate lower-level wireless interface management and connectivity evaluation from higher-level monitoring/selection, thus simplifying the separation between node- and application-specific requirements and the dynamic introduction of new connectivity evaluation metrics. In addition, to take mobility-aware connectivity decisions, our middleware effectively exploits the predicted degree of client node mobility, estimated in a completely autonomous decentralized way. The reported experimental results demonstrate the feasibility of the approach, with accurate estimations of node mobility associated with very limited overhead.

Categories and Subject Descriptors

C.2.1 [Network Architecture and Design]; C.2.4 [Distributed Systems]; D.4.4 [Communications Management]

Keywords

Wireless Computing, Handoff Management, Always Best Connectivity, Middleware, Context Awareness.

1. INTRODUCTION

Two major trends are manifest in the evolution of the mobile computing area in the last decade: the growing availability of processing/memory resources at client nodes and the widespread diffusion of heterogeneous wireless communication technologies. Those trends push towards considering a larger and larger set of services, from traditional Internet applications to novel location-based services, which can be accessed by wireless clients, often equipped with different connectivity technologies, independently of their mobility at provisioning time.

In the following, let us simply call *interfaces* the wireless network interfaces that are available at a client node to connect to the Internet, e.g., IEEE 802.11 and Bluetooth wireless client cards. In addition, we will use the *connector* term to indicate a device that enables client connectivity to the Internet by acting as a bridge between mobile nodes and the traditional fixed network, e.g., an IEEE 802.11 Access Point (AP). Moreover, we will call *channel* any communication entity representing the active usage relationship between a client node and one of its currently available connectors, e.g., the connectivity link activated when an IEEE 802.11 interface associates with a specific AP, after retrieving network configuration via DHCP. In other words, interfaces model the wireless hardware equipment available at the client side, connectors are the non-client-side components providing client access to the fixed Internet via wireless communications with client interfaces, and channels represent the active communication links that client applications may exploit to get connectivity to the Internet. We call this integrated networking scenario including the fixed Internet, client nodes with multiple and heterogeneous wireless interfaces, and heterogeneous connectors in between, as the heterogeneous Wireless Internet (WI).

Even if equipped with several heterogeneous interfaces, nowadays WI client nodes and their applications are able to exploit only one interface at a time. In addition, explicit user effort is required to change the exploited interface at service provisioning time. We claim that in the near future there will be the need to consider also more complex and flexible deployment scenarios where i) client nodes are able to simultaneously exploit several heterogeneous interfaces, and ii) connectors include both infrastructure-based equipment, e.g., IEEE 802.11 or GPRS APs, and client nodes that offer themselves as bridges for Internet connectivity in a peer to peer way, namely, *peer connectors*. In the following we simply indicate the above scenario with the term heterogeneous WI, by intending that in this scenario different wireless technologies and types of interface/connector can be simultaneously involved at any client.

However, the heterogeneous WI requires the development of novel *channel management* solutions to monitor and manage available interfaces/connectors at best, thus providing each application with the most suitable channels at any time. In fact, available interfaces and connectors may provide dramatically different connectivity quality, e.g., IEEE 802.11 provides larger bandwidth while UMTS larger coverage range but at non-negligible economic costs. Novel channel management solutions should manage interfaces/connectors/channels by considering several context data at multiple abstraction levels, e.g., power consumption of an interface, level of trust of a peer connector, and estimated durability of a channel, e.g., how long a peer connector will probably stay within reachability distance. Moreover, such channel management solutions should simultaneously consider even requirements stemming from the overall node, e.g., avoiding power-consuming interfaces when the battery level is low, from the user, e.g., preferring free-of-charge connectors, and from the served applications, e.g., providing higher priority to either channel durability or provided bandwidth. In other words, channel management requires developing specific and flexible metrics to dynamically evaluate available interfaces, connectors, and channels.

The paper claims that, in order to simplify channel management solutions and to leverage the availability of applications exploiting them, there is the need of a context-aware middleware approach that separately considers the requirements affecting the behavior of the whole client node, e.g., power consumption and security concerns, and each application, e.g., channel bandwidth and jitter needs for that specific service. In addition, node requirements should be considered with higher priority, while application-specific requirements should be considered only in a second stage, after having satisfied node-level ones. That separation permits to relevantly reduce the complexity and the overhead of “always best channel” selection in the addressed deployment scenario. To this purpose two different metrics are needed, a lower-layer one to specify which connectors are suitable for channel realization, and a high-layer one to independently select, among the available channels, the best one for each application.

The paper presents the architecture of our Mobility-Aware Connectivity (MAC) middleware which dynamically exploits multiple interfaces and both infrastructure-based and peer connectors, by switching among them in an autonomous and dynamic manner. The focus of the paper is the description of the novel double-layered MAC architecture that clearly separates low-level interface interaction and high-level API provided to upper layers and applications, thus effectively supporting the separate specification of lower/higher-layer evaluation metrics. In addition, while not imposing any peculiar metric, we identify the mobility degree of client nodes and connectors as a crucial context information to consider, since mobility degree has a direct impact on the expected durability of managed channels. For instance, still nodes could get enduring channels by exploiting Bluetooth-based still connectors because channel durability is highly probable in this case. On the contrary, moving client nodes should better consider connectors with larger coverage ranges, e.g., UMTS base stations, or connectors moving in the same direction in order to maximize channel durability. Finally, the paper reports the performance results experimentally obtained while testing our MAC middleware prototype, by demonstrating both the feasibility and effectiveness of the proposed double-layered architecture and the accuracy of MAC estimations of node/connector mobility.

2. BACKGROUND AND MOTIVATIONS

To better understand the motivations behind MAC architecture and solution guidelines, let us rapidly introduce which kind of operations should be taken into account by an advanced support for runtime change of interface/connector (*handover management*). It is possible to identify two major phases in handover procedures: evaluation process and continuity management [1]. The former is in charge of gathering information about the currently accessed interfaces/connectors (and possibly all the available ones) and of evaluating their suitability, e.g., depending on the currently provided quality. The latter is in charge of exploiting the evaluation process result to choose when to perform a handover and to which interface-connector pair (trigger sub-component). Moreover, continuity management should provide support mechanisms for seamless handovers in the case of continuous streaming services, e.g., by temporarily bi-casting packets to both origin and destination connectors to minimize packet loss (switcher sub-component) [2].

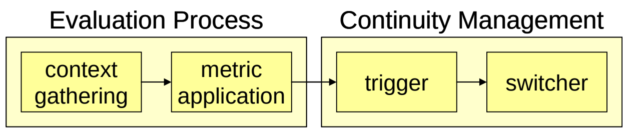

Figure 1. Handover Management facilities include Evaluation Process and Continuity Management.

By considering the notable example of the widespread IEEE 802.11, the evaluation process is embedded in interface firmware and based on Received Signal Strength Indication (RSSI) or Signal to Noise Ratio (SNR), monitored for both origin and destination connectors. The assumption is that lower RSSI and SNR values correspond to limited network performance. Continuity management for intra-horizontal handovers is mainly realized via AP signaling messages to update mobile node location (represented as the currently accessed AP). Continuity management for inter-horizontal handovers is not standardized, but can exploit some partial support mechanisms, either standard such as Mobile IP or special-purpose [3, 4]. In any case, handover is usually triggered when the RSSI or the SNR related to the current AP goes below a fixed threshold [4].

Today the above service provisioning scenario is the most spread one; adopted evaluation solutions are basic and often statically embedded in network interfaces. The separation of concerns depicted in Figure 1 becomes more useful when facing more complex scenarios envisioned for the near future, where there will be client nodes with multiple heterogeneous interfaces, possibly activated simultaneously, and heterogeneous connectors, both infrastructure-based and peer ones. In fact, the increasing capabilities of client nodes suggest new deployment scenarios where clients can also play the role of connectivity providers (peer connectors). For instance, a mobile node with both UMTS and Bluetooth interfaces may decide to play the role of peer connector by exploiting its UMTS interface to connect to the Internet and by offering itself as a Bluetooth modem for neighbors with Bluetooth capabilities. For further information on how MAC models available connectors please refer to [1].

Notwithstanding the complexity of supporting heterogeneous WI scenarios, they can provide several relevant benefits and advantages. First of all, peer connectors can significantly extend the connectivity opportunities for client nodes. For instance, in an area with only cellular connectivity to the Internet, a peer connector with both UMTS and Wi-Fi interfaces can open the Internet access even to nodes with only the IEEE 802.11 interface. In addition, peer connectors can provide alternative connectivity ways, possibly more suitable than directly exploiting infrastructure-based connectors according to specified evaluation metrics. For instance, a peer connector with flat-rate UMTS connectivity can offer itself as a Bluetooth modem for free in the time intervals when it is not accessing any Internet service. Anyway, the exploitation of peer connectors could be far more complex than infrastructure-based usage and requires novel support approaches to effectively tackle newly introduced issues: for instance, peer-based connectivity tends to be less reliable, also in the case of a non-moving client node, since peer connectors can move out of client radio range or abruptly revoke their connectivity offer.

We claim that in such a complex and dynamic scenario the evaluation process cannot be based only on raw monitoring data from the physical layer, such as RSSI and SNR. Moreover, it is unfeasible to statically determine the metric to apply once for all and to embed it in interface firmware. Metrics should depend on the actual deployment environment, e.g., in relation to expected available connector types and target client node capabilities. In addition, the evaluation process should also consider more expressive context information, at different levels of abstraction, to take channel management decisions. Gathered context should include the static/dynamic characteristics of available interfaces and, for each interface, of the available connectors, user preferences, node resource availability (from battery level to available memory), and application-specific quality requirements.

While the literature includes several design and implementation efforts to exploit multiple interfaces simultaneously [5], only some recent research contributions have focused on evaluation metrics for heterogeneous wireless technologies. The evaluation process function in Terminal Management System (TMS) defines quality and cost indicators for eligible connectors depending on a user-specified priority order among connectivity providers and interfaces [6]. TMS applies its evaluation function to any available connector: each function term is weighted according to a weight set based on user-specified priorities, which can change at service provisioning time. Also the Vertical Handoff Decision Function (VHDF) provides users with the capability to specify a priority order among different network characteristics, by defining a proper weight set [7, 8]. The VHDF evaluation function exploits the weights to calculate a linear combination of current network conditions, network performance, service cost, power requirements, security, and pro-activity of the exploited handover process. Both TMS and VHDF neither consider peer connectors nor take into account client/connector mobility as a crucial element to decide channel suitability.

In fact, we claim that, among all the information characterizing the provisioning context, the mobility degree of both client nodes and peer connectors is of primary relevance. Client mobility degree relates to the mobility of the user and refers to a fixed reference system. A client node is *still* when its distance from fixed APs does not change, it is in *motion* otherwise. Peer connector mobility degree relates to the mobility of peer connectors in relation to a reference system centered on their client nodes. We define a peer connector as *joint* if it moves together with the client node having a channel with it (with the same direction and speed), *transient* otherwise. The main idea is to exploit mobility degree information to reduce the set (and management complexity) of available connectors. For instance, while a Bluetooth connector may be suitable to reduce power consumption, it should be discarded in the case of a rapidly moving client with strict requirements of channel durability.

In addition to mobility degree, to evaluate which is expected to be the most suitable channel, an evaluation process should consider several other context data, related to different entities and at different abstraction levels. It should take into account capabilities of client nodes, connector state, e.g., quality and overload conditions, and channel performance, e.g., the estimated time durability. In addition, the evaluation metric should consider requirements related to the user, e.g., the minimum trust level requested to exploited connectors, to the operating system, e.g., either high performance or limited power consumption depending on battery level, and running applications, e.g., bandwidth requirements. The number of involved context information coupled with the heterogeneity of involved entities and abstraction levels makes difficult the development and deployment of novel context-aware metrics and their evolution to consider newly available context data, wireless interfaces, or goals to achieve. In addition, requirements from different entities could bring to conflicting interface management actions, e.g., requiring the same IEEE 802.11 device to simultaneously associate with different APs.

For these reasons, we have designed and implemented the simple but effective MAC evaluation process based on a double-layered metric: a bottom one to evaluate interfaces and connectors considering the whole client node requirements, a top one to evaluate the channels identified by the bottom layer by considering per-application needs. We claim the division of the evaluation metric into two layers greatly simplifies the development of novel metrics. At the same time it is easy to impose greater priority to user and operating system requirements. In fact, in this way the bottom layer metric has the role of discarding certainly unsuitable connectors; for example, a connector is discarded because it belongs to a blacklist of untrusted connectors or because it is a highly transient mobile peer expected to provide connectivity only for a short time interval. The top layer metric has the role of performing the final channel selection, i.e., it selects the channel that an application will exploit during a service session. In any case, applications are impeded to behave harmfully for the client node, e.g., the usage of power-consuming interfaces is inhibited when battery level is low. In other words, the bottom layer metric performs a general-purpose coarse-grained selection, while the top one a fine-grained decision; the former is based on the whole mobile node context information and requirements, the latter depend on application-specific requirements, separately considered for each channel to be established.

3. MAC PRIMARY DESIGN CHOICES AND ARCHITECTURE

Figure 2 represents how the Mobility-Aware Connectivity (MAC) middleware is logically organized. At the current stage of prototype implementation, we have primarily focused on the evaluation process phase, while the continuity management layer is part of our current work.

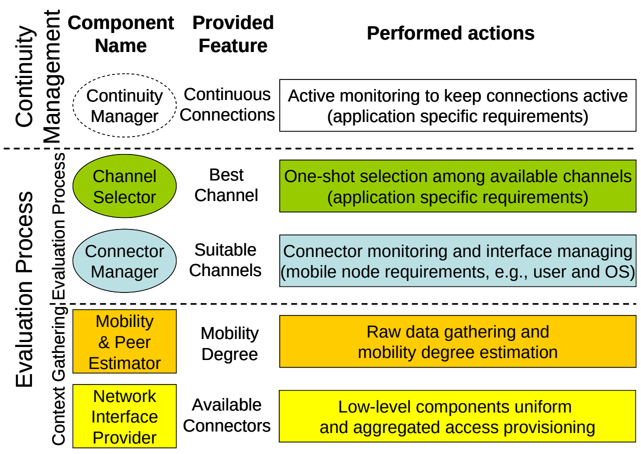
Continuity Management
Evaluation Process
Component Name
Provided Feature
Performed actions
Continuity Manager
Continuous Connections
Active monitoring to keep connections active (application specific requirements)
Evaluation Process
Channel Selector
Best Channel
One-shot selection among available channels (application specific requirements)
Connector Manager
Suitable Channels
Connector monitoring and interface managing (mobile node requirements, e.g., user and OS)
Context Gathering
Mobility & Peer Estimator
Mobility Degree
Raw data gathering and mobility degree estimation
Network Interface Provider
Available Connectors
Low-level components uniform and aggregated access provisioning

Figure 2. MAC logical organization.

The Context Gathering layer consists of the Network Interface Provider (NIP) and the Mobility & Peer Estimator (MPE) components, which are the primary context sources in MAC. NIP provides a uniform and aggregated access to underlying network interfaces, e.g., by providing the set of available connectors for each interface; MPE determines mobility state for client nodes and connectors, e.g., whether a client is currently still or mobile. The Evaluation Process layer is composed by Connector Manager (CoM) and Channel Selector (ChaS) that respectively evaluate connectors and channels. CoM identifies the list of suitable channels depending on the whole client node requirements; ChaS selects the most suitable channel in the CoM-provided list by additionally considering application-specific requirements. Let us note that Figure 2 not only points out each MAC component role, but also delineates the associated abstraction layer: MIP provides information about available connectors; MPE is at a slightly higher abstraction level and exploits NIP output to dynamically evaluate client node and connector mobility degree; CoM interacts with connectors to estimate their suitability for realizing reliable and durable channels; ChaS evaluates CoM-provided channels to select the best one for each application.

Figure 3 presents how we have implemented the model of Figure 2 in the actual MAC architecture. Differently from the purely layered model of Figure 2, MAC sometimes adopt a cross-layer solution for the sake of performance: NIP behaves as input for both MPE and CoM; MPE is configured by CoM in a feedback fashion; moreover, each MAC component not only provides its features and information to other middleware components, but also to the application layer. In this manner, MAC is able to behave both as an autonomous and self-contained system providing the most suitable channel and as a support component useful for other support infrastructure, e.g., our future middleware for continuity management built on top of MAC. In any case, a clear distinction between context gathering and metric application is ensured. The rest of the paper provides details about the primary components of the current MAC middleware prototype: NIP and MPE for context gathering, CoM and ChaS for metric application.

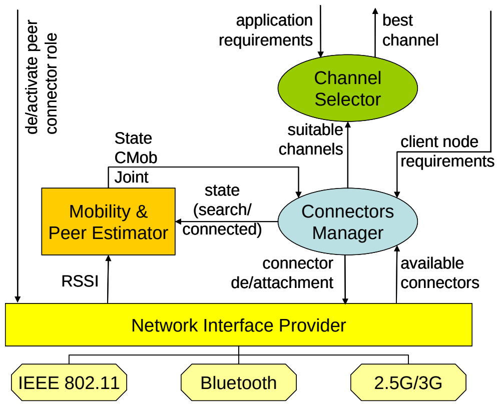

Figure 3. MAC architecture.

4. CONTEXT GATHERING

To correctly estimate each connector suitability degree, MAC has to gather several context data at different levels of abstraction. To that purpose, MAC requests users and applications to express their requirements related to the whole mobile node. User requirements are assumed not to change frequently and may include energy consumption (power saving or maximum performance), maximum affordable cost, and required level of trust. Application requirements are assumed not to change during a service session and may include bandwidth, channel endurance, and other channel related requirements.

While it is possible to provide and exploit many context sources, such as an external positioning system such as GPS to get current geographical location, the current MAC prototype focuses only on locally available information, thus making it rapidly and immediately deployable in current wireless scenarios. However, it is easy to extend the MAC prototype to consider also infrastructure-side context information, such as network load level provided by connectors to avoid the exploitation of overloaded networks.

4.1 Network Interface Provider (NIP)

NIP is the component in charge of actively interacting with network interfaces. NIP provides upper layers with a transparent access to interface capabilities, by completely hiding low-level details related to underlying interface drivers and operating system. In fact, to simplify interface interaction, NIP offers a uniform API to heterogeneous interfaces while preserving peculiar characteristics an interface could be able to provide, as better detailed in the following.

NIP is structured in two layers: feature and wrapper. At middleware initiation time the feature layer considers the underlying operating system and loads the right wrappers to communicate with interface drivers. In addition, it exposes an API to upper layers to access interfaces without knowledge of low-level and interface-specific implementation details. The wrapper layer is in charge of directly interacting with interface drivers to perform required commands, possibly in an operating system-dependent way. Note that the upper layer is developed once for every interface, while the lower layer once for every exploited operating system. In this manner, NIP facilitates the introduction and exploitation of new interfaces over different operating systems.

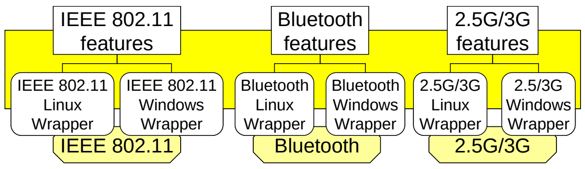

Figure 4. Network Interface Provider.

Delving into finer details, the feature component provides a set of capabilities common to any interface:

- *get available connectors*, i.e., the set of connectors and related information that an interface is currently able to access;
- *connect to a connector*, requiring the interface to connect to a particular connector and establishing the related channel;
- *perform as peer connector*, starting to offer connectivity with a specific interface in a peer-to-peer way.

Not any interface could be able to provide the above features and, in any case, the same feature applied to different interface types could behave in a slightly different manner, depending on the capabilities offered by the underlying wrapper. For instance, peer-to-peer connectivity in Bluetooth could be offered via the Personal Area Network (PAN) service, in IEEE 802.11 by creating a new ad-hoc network, while it is not possible for UMTS devices. In addition, some interfaces may provide additional capabilities: for instance, the Bluetooth interface can obtain the set of currently connected remote devices, while IEEE 802.11 can connect to a specific AP (via BSSID identification) and even to a specific target network (via ESSID identification).

The current MAC prototype supports IEEE 802.11 and Bluetooth interfaces, by including wrappers for both Windows XP/Vista and Linux. The former interface is accessed on Linux client nodes via the Linux Wireless Extensions, on Windows XP/Vista client nodes via the Microsoft Network Driver Interface Specification User-mode I/O (NDISUIO), which is platform-dependent but portable among different wireless interface implementations. For instance, MAC exploits the NDISUIO function `DeviceIOControl()` to query the `OID_802_11_BSSID_LIST_ SCAN` object to retrieve the complete list of currently reachable connectors, either IEEE 802.11 APs or peer nodes in ad-hoc configuration. The latter interface is accessed on Linux client nodes via the standard API provided by the BlueZ protocol stack, on Windows XP/Vista client nodes via API provided by the Windows Driver Kit and the Software Development Kit tools. For example, MAC becomes aware of the set of available Bluetooth devices close to a client by invoking `BluetoothFindFirstDevice` and `BluetoothFindNextDevice` functions.

4.2 Mobility & Peer Estimator (MPE)

While NIP provides raw information and access to interfaces, MPE provides context information at a higher abstraction level. It provides a dynamic estimation of the client node movement degree, namely *CMob*, and, for each peer connector, its mobility degree in relation to the client node, namely *Joint*. To estimate these values, MAC monitors the execution environment and collects RSSI data about any eligible connector.

Delving into finer details, for each interface MAC determines the list of available connectors and collects RSSI sequences for each connector. Then, for each fixed (mobile) connector *CMob* (*Joint*) is set linearly depending on the variability of the RSSI sequence for that connector. To estimate RSSI sequence variability, first of all MPE low-pass filters RSSI fluctuations due to signal noise, in order to identify only RSSI modifications due to actual client node movements. RSSI low-pass filtering is achieved applying the Discrete Fourier Transform (DFT) to 4s-long RSSI sequences and regenerating the RSSI sequence via the Inverse Discrete Fourier Transform (IDFT) exploiting only the first harmonic, thus discarding high frequency signal components. In particular, when evaluating IEEE 802.11 (Bluetooth) connectors, MPE gathers 4 (1) RSSI values per second, thus applying the DFT to 16 (4) values. We exploit different RSSI sequence lengths since IEEE 802.11 RSSI values show greater noise if compared with Bluetooth ones, thus requiring more aggressive RSSI low-pass filtering. Then, mobility degree indicators are computed via the linearization in the [0, 1] range of the first harmonic module of the low pass filtered RSSI sequence. We have experimentally validated how *CMob* and *Joint* depend on RSSI variability and the values used in MAC are the result of these experimental evaluations. Additional implementation details and performance results are presented in Section 6.

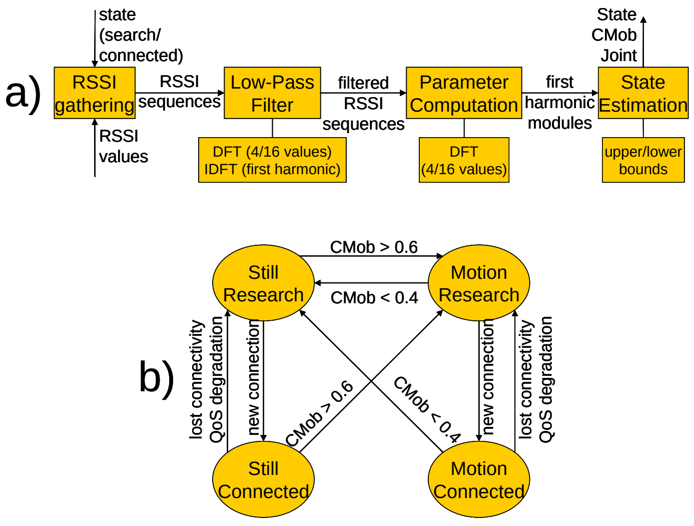

Figure 5. Mobility and Peer Estimator: a) processing chain and b) still/motion state diagram.

Let us note that MAC only performs local monitoring at client nodes. In that way, it achieves a twofold benefit. First, MAC exploits only local information that is available despite clients are currently connected to the heterogeneous WI. Then, it does not require any external special-purpose support component, e.g., monitoring components working on the infrastructure side, thus enabling the potentially immediate MAC adoption in any heterogeneous WI scenario by only deploying MAC components at client nodes. However, even the only local monitoring of network interfaces is a power consuming process [9]. Therefore, to minimize power consumption, MAC performs "aggressive" context gathering only when required (client in research state), while performing "lazy" monitoring otherwise (client in connected state) [1]. In addition, MAC considers even the client node mobility state, either still or motion, by performing an aggressive monitoring whenever client state changes because each connector suitability degree may vary dramatically, as better detailed in the following section. To understand whether a client is in still/motion states, MAC exploits CMob monitoring and its time evolution according to the state diagram in Figure 5. MAC switches the client state from still to motion whenever CMob becomes greater than 0.6, while it performs the inverse switch when CMob passes below the 0.4 threshold. The adoption of different thresholds for the two state transitions has been decided to prevent from bouncing effects. In fact, frequent switching between still and motion states would impose repeated perturbations in connector/channel selection, by possibly causing frequent and expensive connector/channel changes and by consequently degrading connection quality.

5. METRIC APPLICATION

As already stated, the evaluation process involves multiple steps and considers several entities, i.e., interfaces, connectors, and channels. In particular, Connector Manager (CoM) is the MAC component that evaluates connectors and actively interacts with interfaces; it actually changes the client node behavior, e.g., by establishing a channel with a given connector or by switching interfaces on/off. Instead, Channel Selector (ChaS) simply evaluates CoM-provided channels to estimate which is the most suitable one for a given application. Coupling CoM and ChaS makes possible to provide a flexible, context-aware, and effective evaluation process, by clearly separating connector/channel evaluation and system/user/application requirements.

5.1 Connector Manager (CoM)

CoM is a crucial component of the MAC middleware because it directly affects the client node channel decisions. In fact, it interacts with the underlying interfaces to change their configuration. Due to the criticality of the actions it performs, CoM cannot be directly set by applications: indeed, applications could be selfish, requiring always as much performance as possible, even if their requirements may affect other applications. Fort these reasons, CoM provides applications with a limited set of channel possibilities, i.e., only with the channels suitable for the entire client node with "no risks" for other running applications. While this may decrease the potential capabilities of applications, it ensures the safety of the whole client. In order to correctly estimate whether a connector is suitable for establishing a channel, CoM has to gather and consider many client-related context data, since channel realization may affect the capabilities of the whole client node. For instance, preferring Bluetooth connectors could become compulsory in the case of battery shortage, while accessing an untrusted peer connector may affect client node security.

The primary CoM sub-components are Metric Provider, Connector Evaluator, and Connector Configurator. Metric Provider permits to add new evaluation metrics and to change the actually adopted one. Each metric has only to provide a specific interface with a method to quantitatively evaluate a provided connector; it may be based on different context sources and requirements; MAC middleware administrators are in charge of checking the availability of any required external support component. Connector Evaluator is the component actually retrieving the currently selected metric and applying it on available connectors. Connector Configurator establishes channels by using the most suitable connectors, i.e., the ones with estimated value greater then a threshold or always the best *n* connectors (threshold and *n* values can be specified at runtime by users or system administrators). Note that Connector Configurator is the only sub-component that actively manages interface behavior, e.g., by associating an IEEE 802.11 device to a given AP or connecting a Bluetooth device to suitable Bluetooth peer connectors. In addition, Connector Configurator associates established channels with related context, e.g., provided nominal bandwidth (depending on underlying interface technology) and channel durability (estimated by Connector Evaluator).

Note that Figure 6 explicitly considers mobility indicators as metric inputs because we claim their crucial role in the definition of suitable evaluation metric for the heterogeneous WI. We have decided that CoM considers channel durability as a primary goal. For instance, CoM automatically excludes transient mobile peers as possible connectors because their channels are shortly durable due to the fact that these connectors remain in the client node connectivity range only for a short period.

In particular, CoM quantitatively evaluates every connector by determining a ConnectorValue in the [0, 1] range (0=worse choice, 1=best choice) for each of them. To that purpose, CoM exploits the evaluation function below:

$$ConnectorValue = EnduranceValue + MetricSpecificValue$$

where EnduranceValue estimates the expected connector durability and MetricSpecificValue its expected quality in terms of other context information. The evaluation of each addend dynamically changes depending on the fact that CoM considers the evaluated connector either fixed connector or mobile peer connector. In the case of a fixed connector:

$$EnduranceValue = CMob \bullet Range$$

$$MetricSpecificValue = (1 - CMob) \bullet \left((1 - \alpha - \beta) + \alpha \cdot Energy + \beta \cdot Trust \right)$$

while in the case of a mobile peer connector:

$$EnduranceValue = (1 - Joint) \bullet Range$$

$$MetricSpecificValue = Joint \bullet \left((1 - \alpha - \beta) + \alpha \cdot Energy + \beta \cdot Trust \right)$$

CMob and *Joint* are values in the [0, 1] range. The Range parameter (always in the [0, 1] interval) concisely models the radio coverage of a connector and only depends on the associated interface. For instance, CoM associates IEEE 802.11 AP connectors with a Range value of 0.7 while Range for Bluetooth peer connectors is 0.3. MetricSpecificValue considers for both fixed and mobile connectors the Energy and Trust parameters (always in the [0, 1] interval), which model the ratio between available/required energy and level of trust. α and β (α, β ≥ 0 and α + β ≤ 1) concisely model OS-/user-specified requirements, thus adapting the relative relevance of OS/user preferences, e.g., in relation to the dynamically varying battery level, thus adapting the relative relevance between energy and level of trust preferences.

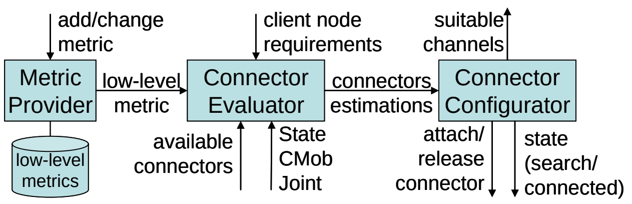

Figure 6. Connector Manager.

To better understand the proposed simple metric, let us take into consideration the two following cases. On the one hand, when the mobile node state is motion or the peer connector transient, i.e., CMob ≈ 1 or Joint ≈ 0, the proposed metric considers the endurance of the connection as the only goal to pursue. On the other hand, when the mobile node state is still or the mobile node joint, i.e., CMob ≈ 0 or Joint ≈ 1, the metric considers even other parameters, e.g., the ones specified by the adopted metric. The rationale is that if a user requires a reliable connectivity while moving, a potentially difficult task, the metric has to partially ignore her requirements in terms of power consumption or level of trust. Instead, a reliable connection is easily achievable when the user state is still/joint, by allowing to consider additional requirements such as power consumption and level of trust.

System/user requirements behave in a slightly different manner. Considering a rather still client node (CMob ≈ 0) or a rather joint peer connector (Joint ≈ 1), if α, β = 0, MetricSpecificValue ≈ 1, thus connectors become similar the one to the other, by completely neglecting power consumption and level of trust. CoM may connect to every connector, delegating any choice to the following top layer metric. Instead, if α, β = 0.5, the bottom layer metric considers power consumption and level of trust equally.

5.2 Channel Selector (ChaS)

Since every channel provided by CoM is considered suitable for connectivity provisioning, ChaS gives the possibility to applications to specify their specific requirements and then accordingly selects the most suitable channel. ChaS evaluation metric considers the only context information related to the available channels, e.g., channel bandwidth and durability, thus relevantly reducing the optimal channel selection complexity. Note that ChaS scope is rather limited: it cannot either interact with interface or change client node configuration.

In other words, CoM and ChaS behave greatly differently. While the former interacts with interfaces, actively changing their configuration, the latter simply monitors available channels to select the most suitable one for each application separately. ChaS does not change the behavior of underlying components; it simply provides each application with the best channel considering application specific requirements. Another relevant difference is that while CoM actively monitors available connectors and determines potential channels despite application-level connectivity requirements, ChaS evaluates provided channels only as consequence of application channel requests. Once an application has obtained a channel from ChaS and started its session, ChaS performs neither channel monitoring nor connection re-establishment in case of lost channel; the application (or the Continuity Manager component on top of ChaS) has to explicit require channel re-establishment whenever its channel does not fit its requirements anymore. ChaS includes a Channel Evaluator sub-component, in charge of evaluating each available channel applying a specific metric and application requirements. Similarly to Connector Evaluator, Channel Evaluator is able to apply novel metric dynamically; the only requirement is that any metric function offers a method to invoke for the quantitative evaluation of a channel.

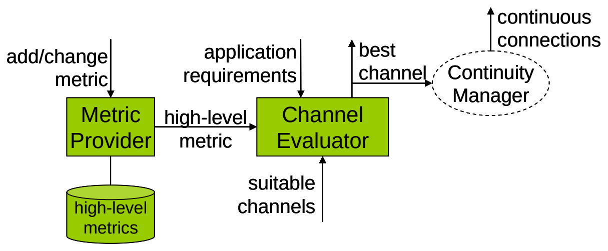

Figure 7. Channel Selector.

Even if ChaS does not rely on any particular metric, in the current MAC prototype we identify the channel endurance as the most crucial context data it has to consider. The endurance estimation for each available channel is provided by the underlying CoM. While even the bottom layer metric considers the channel endurance as an important evaluation parameter, the top layer metric could take into consideration more stringent requirements, e.g., application-specific requirements on channel durability. In particular, ChaS quantitatively evaluates channels as follows:

$$ChannelValue = x \bullet EnduranceValue + y \bullet MetricSpecificValue$$

where EnduranceValue is the estimated durability of the channel, MetricSpecificValue may consider network-layer channel conditions (bandwidth, BER, delay, jitter), and x and y (x, y ≥ 0 and x + y ≤ 1) represent application requirements, i.e., the relative relevance between channel durability and other requirements.

To better understand top-layer metric behavior, consider two applications, one downloading a file via FTP, the other providing interactive monitoring capabilities to the user. The former application may benefit from a channel with large bandwidth, even if expected to last for a short time; for this reason its requirements are modeled as x=0 and y=1, thus considering only network-level capabilities. Instead, the latter application requires an enduring connection, even if with limited bandwidth; for this reason it specifies x=1 and y=0, thus forcing to consider the most durable channel.

6. PERFORMANCE CONSIDERATIONS

We have performed several experiments to accurately establish effective configurations of MPE and of metric parameters fitting most common deployment scenarios. For instance, only to mention some practical configuration details, our experiments in IEEE 802.11 (Bluetooth) testbed environments suggested us to set *CMob* (*Joint*) to 0 (1) when the first harmonic module is <= 0.35 (0.15), to 1 (0) when the module is >= 0.85 (0.60), to a linearly dependent intermediate value otherwise. Additional information about MAC implementation and the downloadable code of the mobility estimator prototype are available at:
`http://lia.deis.unibo.it/Research/MAC/`

Here, for the sake of briefness, the main purpose of this section is to point out the robustness of MPE in relation to several different wireless environments. The reported experimental results are mainly focused on the *CMob* parameter gathered via an IEEE 802.11 simulated environment (simulations permit to achieve several performance results in different deployment environments with easily controllable configurations). However, our experience has demonstrated that very similar performance results may be achieved also in actual IEEE 802.11/Bluetooth environments for both *CMob* and *Joint* indicators. In particular, to evaluate the MPE performance, we have defined the following indicators:

$hitRate_{\%} = (Correct / Total) * 100$

where Correct is the number of correctly estimated client node mobility states, either still or mobile, and Total is the total amount of sampled states (sample frequency = 1 Hz);

Responsiveness = [∑ (correctly perceived state change time - actual state change time)] / correctly perceived state changes

where Responsiveness models how quickly MPE is able to perceive the state change from still to mobile or vice versa;

longTermHitRate$_{\%}$

the same as hitRate but without considering samples in a 5s-long time window after any mobility state change.

Main environment characteristics may affect MPE performance are the RSSI noise, strongly dependent on concrete walls disposal and human presence, and client node mobility pattern, e.g., maximum user speed. We have compared MPE performance in a simulated environment with 17 APs deployed in a hexagonal grid, adopting the following parameters: RSSI with a noise standard deviation of 1, 3 or 5 dBs; a waypoint mobility pattern with a speed in the [0.5, 1.5], [1.5, 2.5] or [2.5, 3.5] m/s range.

Table 1. MPE performance results.

RSSI Std. Dev. (dB)		1			3			5		
Average Speed (m/s)		1.0	2.0	3.0	1.0	2.0	3.0	1.0	2.0	3.0
Hit Rate (%)		72	73	73	70	73	67	65	61	53
Respon-siveness (s)	average	13.5	4.7	4.3	12.8	5.2	5.1	9.6	9.9	9.3
	std. dev.	12.7	1.3	1.9	10.0	3.2	2.9	7.5	6.4	6.0
Long Time Hit Rate (%)		84	99	97	85	96	94	78	74	65

In general, MPE has shown to correctly evaluate client node mobility state; in particular, after the 5-s transition period following still/mobile state change, MPE achieves great performance. As Table 1 shows, only imposing very relevant RSSI noise, i.e., with a 5dB standard deviation, the achieved performance starts to decrease because RSSI fluctuations due to signal noise are more frequently evaluated as client movements. Another interesting aspect to underline is that MPE usually behaves better when client node speed is relatively high. In fact, MPE is less effective in recognizing slow movements and, in any case, it requires a non-negligible time interval to recognize mobility state changes. Finally, let us stress that RSSI gathering and *CMob*/*Joint* estimation are performed in a completely autonomous and decentralized manner, thus introducing a limited overhead [4]. The reported performance results, coupled with the low overhead imposed, demonstrates the MPE capability to provide mobility-related context information in an effective manner, by actually permitting to compute and exploit channel durability in the evaluation process for mobility-aware always best connectivity.

7. CONCLUSIONS

The advances in device miniaturization and wireless communications are pushing the migration towards the heterogeneous WI, where a flexible and context-dependent evaluation process is crucial to fully take advantage of the novel opportunities offered by this challenging deployment scenario. The paper demonstrates the need for novel middleware supports that evaluate interface and connector suitability considering not only traditional parameters but also more expressive context information. MAC shows the viability of the adoption of middleware solutions by pointing out the need to separately consider low-level user/system requirements, related to the whole client node context, and high-level application ones, individually specified by each application requiring connectivity. In addition, MAC demonstrates the suitability of gathering and exploiting mobility indicators as crucial context data that evaluation metrics should exploit for taking decisions about channel selection and establishment.

The encouraging results obtained by the first MAC prototype are stimulating our on-going research activities. In particular, we are currently evaluating the MAC performance over a wide deployment scenario with dozens of WiFi/Bluetooth infrastructure-based and peer connectors, to validate our middleware capability to support continuous services with even strict handover requirements, such as multimedia streaming. In addition, we are extending the current MAC prototype to include the support to additional interfaces such as UMTS and WiMAX.

8. ACKNOWLEDGMENTS

Work supported by the MIUR PRIN MOMA, the MIUR FIRB TOCAI, and the CNR Strategic IS-MANET projects.

9. REFERENCES

[1] P. Bellavista, A. Corradi, C. Giannelli, "Mobility-Aware Connectivity for Seamless Multimedia Delivery in the Heterogeneous Wireless Internet", 2nd Work. on multiMedia Applications over Wireless Networks (MediaWiN 07), Aveiro, Portugal, July 2007.

[2] N. Shenoy, R. Montalvo, "A Framework for Seamless Roaming across Cellular and Wireless Local Area Networks", IEEE Wireless Communications, Vol. 12, No. 3, pp. 50-57, June 2005.

[3] P. Bellavista, A. Corradi, C. Giannelli, "Mobile Proxies for Proactive Buffering in Wireless Internet Multimedia Streaming", IEEE Int. Conf. Distributed Computing Systems (ICDCS) Workshops, pp. 297-304, June 2005.

[4] P. Bellavista, A. Corradi, C. Giannelli, "Adaptive Buffering based on Handoff Prediction for Wireless Internet Continuous Services", Int. Conf. High Performance Computing and Communications (HPCC), LNCS 3726, pp. 1021-1032, 2005.

[5] P. Bahl, A. Adya, J. Padhye, A. Wolman, "Reconsidering Wireless Systems with Multiple Radios", ACM SIGCOMM Computer Communications Review (CCR), Vol. 34, No. 5, pp. 39-46, Oct. 2004.

[6] E. Adamopoulou, K. Demestichas, A. Koutsorodi, M. Theologou, "Intelligent Access Network Selection in Heterogeneous Networks - Simulation Results", Int. Symp. Wireless Communication Systems (ISWCS), pp. 279-283, Sept. 2005.

[7] A. Hasswa, N. Nasser, H. Hossanein, "Generic Vertical Handoff Decision Function for Heterogeneous Wireless", IFIP Int. Conf. Wireless and Optical Communications Networks (WOCN), pp. 239-243, Mar. 2005.

[8] N. Nasser, A. Hasswa, H. Hassanein, "Handoffs in Fourth Generation Heterogeneous Networks", IEEE Communications Magazine, Vol. 44, No. 10, pp. 96-103, Oct. 2006.

[9] E. Ferro, F. Potorti, "Bluetooth and Wi-Fi Wireless Protocols: a Survey and a Comparison", IEEE Wireless Communications, Vol. 12, No. 1, pp. 12-26, Feb. 2005.