Femminella *et al. EURASIP Journal on Wireless Communications and Networking* 2012, **2012**:140
http://jwcn.eurasipjournals.com/content/2012/1/140

EURASIP Journal on
Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

A signaling architecture for multimedia MBS over WiMAX

Mauro Femminella*, Roberto Francescangeli and Gianluca Reali

Abstract

In this article, we design and demonstrate a signaling architecture for multicast and broadcast services over a laboratory-emulated worldwide interoperability for microwave access (WiMAX) network. WiMAX is a broadband wireless access technology which includes different quality of service levels. Currently, a significant research effort focuses on data transmission optimization and mobility support for multicast and broadcast services. Our proposal is fully IP-based, and the relevant signaling architecture is modular in all its entities and extensible to other, non-WiMAX, IP access networks. We specifically address the non-trivial support of multicast services, which has been demonstrated through a prototype implementation of our proposal for an Internet protocol television (IPTV) service. This prototype has been implemented by using open source technologies and its signaling is orchestrated by a JAIN SLEE server. The results obtained show that the implemented system is scalable, can achieve both high signaling throughput and low service latency, and has a signaling overhead lower than similar IMS-based solutions.

Keywords: MBS, JAIN SLEE, WiMAX, signaling, SIP, IPTV

1. Introduction

The worldwide interoperability for microwave access (WiMAX) is a wireless broadband technology. Its architecture includes a set of flexible capabilities which enable the convergence of mobile and fixed broadband networks. By using suitable communication technologies, such as multiple input multiple output smart antennas and orthogonal frequency division multiple access, WiMAX can provide high throughput, efficient data multiplexing, and low transmission latency in both fixed and mobile communications. These features, together with the capability of covering large areas with relatively few base stations (BSs), are fundamental to deploy effective wireless broadband network services able to support many applications, such as data exchange, audio/video streaming, and VoIP services, with different levels of quality of service (QoS) [1]. Leveraging on these network services, service providers can also implement the so-called Multicast Broadcast Service (MBS) over WiMAX, which consists of providing mechanisms for delivering the same contents to multiple users who share radio resources. In particular, a single-frequency operating mode is envisaged, through which a single radio channel is used to distribute information to all users subscribed to the same service [2]. The MBS is typically used for delivering multimedia contents, such as mobile Internet protocol television (IPTV) and audio/video file casting, and for implementing massive software updates [1,2]. Implementing MBS over WiMAX is not trivial for service providers, since customers may also need a return channel for interacting with a management system in order to dynamically subscribe/unsubscribe to any available service. Moreover, multicast services are more complex to be designed and maintained than broadcast services, due to additional signaling, content delivery authorization and management, and required support of multicast streams in WiMAX subscriber stations (SSs) and mobile stations (MSs). Finally, an MBS architecture must be scalable with the number of users, requires a central management system, should both include an easy to use management software application and be implemented through standard technologies so as to simplify MBS service development and maintenance.

In this paper, we illustrate an original design, and the relevant prototype implementation, of a simple and effective end-to-end service architecture that extends

* Correspondence: mauro.femminella@diei.unipg.it
DIEI, University of Perugia, Via G. Duranti 93, 06125 Perugia, Italy

the MBS Network Reference Model (NRM) [3] and provides MBS over WiMAX.

Our aim was to create a flexible and efficient signaling system for MBS architecture, with a low architectural complexity, so as to reduce the initial investment for service providers. In fact, an MBS system has a quite complex set of requirements, such as the need of using a return channel and a central service orchestration point that needs a suitable design. This design has to guarantee scalability, manageability, efficiency and maintainability. Most the recent literature on MBS focuses mainly on data transport over WiMAX networks. Nevertheless, in order to achieve the MBS requirements mentioned above, a suitable signaling support is also essential.

Our proposal addresses these features by resorting to available standard and open interfaces that speed up implementation process and reduce time to market. In fact, the use of development frameworks providing high-level APIs may be extremely helpful to service creation, since suitable APIs can mask the complexity of underlying network and transport layers. In addition, the use of standard open interfaces, rather than proprietary ones, has the additional advantages of both supporting multiple access networks, devices, and protocols, and attracting a larger number of developers. For these reasons, some key implementation issues of our MBS architecture have been faced by resorting to the Service Logic Execution Environment (SLEE), which is a hosting environment for advanced telecom services. The SLEE logic is designed to fulfill the specifications of telecom services, which are typically asynchronous and require high throughput and low signaling latency. Thus, a SLEE server can be useful to effectively orchestrate the overall signaling architecture of MBS over WiMAX.

Our MBS solution is IP-based and adopts standard IP signaling protocols, such as SIP (Session Initiation Protocol) [4] and IGMP (Internet Group Multicast Protocol) [5]. We have implemented the core signaling elements by using an open source JAIN SLEE application server (AS), named Mobicents [6]. Being based on standard protocols and open interfaces, our architecture is not limited to WiMAX, but it may both include other existing networks, and be accessed by most existing terminal devices. As a proof of concept, we have implemented an IPTV multicast service on top of our MBS prototype, which is often indicated as the main application for MBS architectures [2]. We show that this service is scalable with the number of users, flexible in the deliverable contents, modular in all its entities, and centrally manageable. Latency and throughput are evaluated experimentally, and an in-depth analysis of the signaling overhead and its impact on a WiMAX network is presented.

The paper is organized as follows. The next background section shows a brief overview of the SLEE container, including the open source SLEE package that we have used in our implementation; it also illustrates the MBS over WiMAX issues and some recently published results, including a related work about an alternative signaling architectures for IPTV. Section 3 describes the design of our service architecture and the end-to-end signaling flows between network entities. Section 4 illustrates the test-bed implementation. Section 5 shows the achieved performance, including a specific analysis of the signaling overhead. Some final remarks are reported in Section 6.

2. Background and related work

2.1 JAIN SLEE specifications

A SLEE is a service container made of several abstraction layers. Its multi-tier architecture simplifies the development of complex services by providing the non-functional features needed for their execution, such as object lifecycle management, persistence management, thread management, object pooling, and so on.

The Java APIs for Integrated Networks (JAIN) activity has delivered the JAIN SLEE (JSLEE) specifications, which delineate a Java-based, event-oriented, and protocol-agnostic container suitable for hosting carrier-grade telecom services [7]. It includes different Java Enterprise Edition (J2EE) components, adapted to the needs of telecom applications.

In a JSLEE container, a service logic is organized and implemented in system components called Service Building Blocks (SBBs). In operation, a JSLEE server creates a pool of SBB objects and manages them according to a well-defined lifecycle. SBBs operate asynchronously by receiving, processing, and triggering events. They are attached to one or more data streams called Activities, by which they can exchange events and share state variables (Activity Context) regarding the Activity itself with other JSLEE system entities. SBBs may be hierarchically structured according to parent-child relations, in order to enhance service logic modularity. Events are internally managed and routed by a functional entity called Event Router, which delivers each event to the appropriate recipient SBB. With regard to the JSLEE performance, an Event Router plays a critical role, since it processes all system events.

External network events, such as a SIP message arrival, are translated into JSLEE events by means of specific entities called Resource Adaptors (RAs). Thus, the set of available RAs constitutes an abstract interface layer that allows a JSLEE to access external resources. If service interoperability with a particular protocol stack is needed, it can be achieved easily by including the relevant RA interface in service code (e.g. methods to access

SIP header fields) and deploy the appropriate RA Java archive (JAR) file within the JSLEE container.

Our MBS signaling server has been developed by using the Mobicents Communication Platform [6] that is an open source project currently owned by Red Hat. Mobicents includes a JSLEE, a Media server, a Presence server, and a SIP Servlet server. At the time of writing, the Mobicents JSLEE (MSLEE) is under active development and it is the first and only open source implementation of the latest JSLEE v.1.1 specifications [7]. The MSLEE includes several J2EE components, such as Container Managed Persistence (CMP) fields, which enable data persistence for SBB objects, Java Database Connector (JDBC) drivers, Java Management Extensions, SNMP interfaces, and Java Naming and Directory Interface (JNDI), which provides lookup functions for services and variables. In order to leverage the aforementioned J2EE technologies, the MSLEE is deployed and executed within the JBoss AS [8], which plays the role of container of containers, offering advanced capabilities such as JSLEE service deployment, thread pooling, logging, etc.

2.2 MBS over WiMAX

A broadcast service may be defined as a unidirectional point-to-multipoint (PMP) service in which data is transmitted from a single source to all terminals (Subscriber Units, SUs, including either MSs or SSs) included in its broadcast service area. Thus, broadcast services are often referred to as *push*-type services [9]. Multicast services are unidirectional PMP services in which data is transmitted from a single source to a multicast group of SUs located in the multicast service area. Only SUs that have subscribed to a specific multicast service and joined the relevant multicast group can receive data relevant to that service. Whilst a broadcast service can be accessed without any explicit request from customers [10], multicast users need a return channel for interacting with a management system, in order to dynamically subscribe/unsubscribe available services. Thus, designing an end-to-end multicast service architecture is not trivial since it must fulfill all service requirements and be compliant with the underlying network architecture.

The IEEE 802.16-2009 [11] standard specifies the MBS deployment over WiMAX networks. MBS provides an efficient transmission of multimedia streams from one or more BSs to multiple MSs through a shared radio resource [2]. Multiple BSs compose an MBS zone, where BSs synchronously transmit over the air multicast data belonging to the same service flow, by using a common connection identifier (CID) and through the same security association. Since BSs located within the same MBS zone use a common Multicast CID (MCID) for the same MBS multimedia stream transmission, MSs do not require registration to any specific BS and the handover functions are transparent to a MS moving within the same MBS zone.

The MBS service can be provided over the WiMAX air interface [12], since:

- high data rate and large coverage using a single frequency network;
- flexible radio resource allocation that enables dynamic radio resource management in MBS areas;
- low MS power consumption;
- low channel switching time, in order to support multichannel services such as mobile IPTV.

MBS over WiMAX is suitable for applications that need to simultaneously provide large subscriber groups with the same data. Usage of broadcast/multicast bearers allows multiple receiving SUs to share downlink radio resources and receive the same data in parallel. Hence, this technology allows network operators reducing costs for delivering multimedia services, such as mobile IPTV, audio/video file-casting, mass software, and configuration updates. Further, multicast/broadcast transmission allows such services to scale well with the number of users, without modifying the network capacity even in case of extremely high service demand. Multicast services are more complex to be designed and maintained than broadcast services due to additional signaling exchange and content delivery authorization and management. For this reason, the implemented service, shown in what follows as a proof-of-concept of our architecture, focuses on multicast transmission of IPTV to WiMAX users.

2.3 Related studies on MBS over WiMAX

Some of the recent literature on the MBS-enabled WiMAX focuses on the extension of the baseline MBS architecture in order to improve the performance of video broadcasting over the air [2,13]. Other articles are related to video QoS provisioning [14,15] and handover management [16]. Nevertheless, the end-to-end signaling support for IPTV provisioning over WiMAX is not investigated. In [17], the multicast service management is optimized in terms of channel switching time, uplink bandwidth, and power consumption. Nevertheless, this proposal requires SU modifications, which do not operate in a standard way, especially for what concerns IGMP message handling. We believe the need of modifying SUs to be a strong limitation.

Differently, we prefer the solution of using a dedicated service delivery entity [2,13-15,17] and enhance it by adding signaling and authorization capabilities. In addition, the solutions proposed in [2,14,15] can easily be integrated in our architecture, since we do not require any specific procedures for handling multicast data flows.

In regard to signaling for IPTV, the articles [18,19] present an approach based on the IP Multimedia Subsystem (IMS) [20]. In [18], the authors present a detailed SIP exchange for live TV and Video On Demand, both for content request and content switch, and analyze the relevant message delays. In [19], an IMS-based architecture for providing converged IPTV services, compliant with the ETSI TISPAN NGN release 2 specifications, is illustrated. Extensions to IMS session setup for supporting multicast-based many-to-many services are discussed in [21]. Proposals in [22,23] enhance the IMS-based service architecture illustrated in [19], by adding scenarios aiming to a future interactive TV experience, illustrated by a SIP signaling exchange.

The SIP-based IMS architecture is the most largely used in the IPTV signaling proposals [18,19,21-24]. Nevertheless, IMS is a very complex reference architecture, and its complexity often leads service providers to resort to simplified architectures, often proprietary, to provide users with advanced services. Due to its complexity, IMS typically exhibits worse performance than a pure SIP-based service architecture [25]. Our aim is to create a simple and effective architecture that both require a very small initial investment and provides advanced MBS services without changing existing network entities and configurations.

Finally, with reference to IPTV over general network architectures, it is worth to mention a very interesting survey in [24], whereas quality of experience of an IPTV service is analyzed in [26].

3. System architecture

To facilitate end-to-end interoperability, the WiMAX forum has specified the WiMAX NRM [3], that is, a logical representation of the network architecture. The NRM is based on all-IP core and a packet-switched air interface. The main NRM advantage is that the network entities are agnostic of the IEEE 802.16 radio specifications. In [2], a new entity, called MBS Controller (MBSC), has been included within NRM. However, as mentioned above, the MBSC has been defined only for improving service data delivery, without specifying signaling entities and signaling flow management.

Figure 1 shows our proposed NRM. The top-level scheme includes the following logical entities: client MS/SS, network access provider (NAP), connectivity service network (CSN), and MBS. MS/SS includes the generalized collection of functions used to provide connectivity between mobile/subscriber equipment and BS. The NAP includes access service network (ASN) entities. Each ASN includes an ASN gateway (ASN-GW) and BSs. ASN is a node aggregation defined as a logical boundary in a mobile WiMAX radio access network. Typically, an ASN consists of multiple BSs that perform radio-related functions, and a gateway node (ASN-GW), that both interfaces with a CSN (where the key functional entities for WiMAX operators reside) and provides IP connectivity services to the WiMAX MS/SS. A CSN includes the MBSC, which in turn includes three separate entities, the MBSC enhanced transport layer [15] (MBSC-ETL) and two further entities that we propose to include in this architecture, the MBSC signaling server (MBSC-SS) and the MBSC database (MBSC-DB). The MBSC-ETL is used for scheduling and transmitting

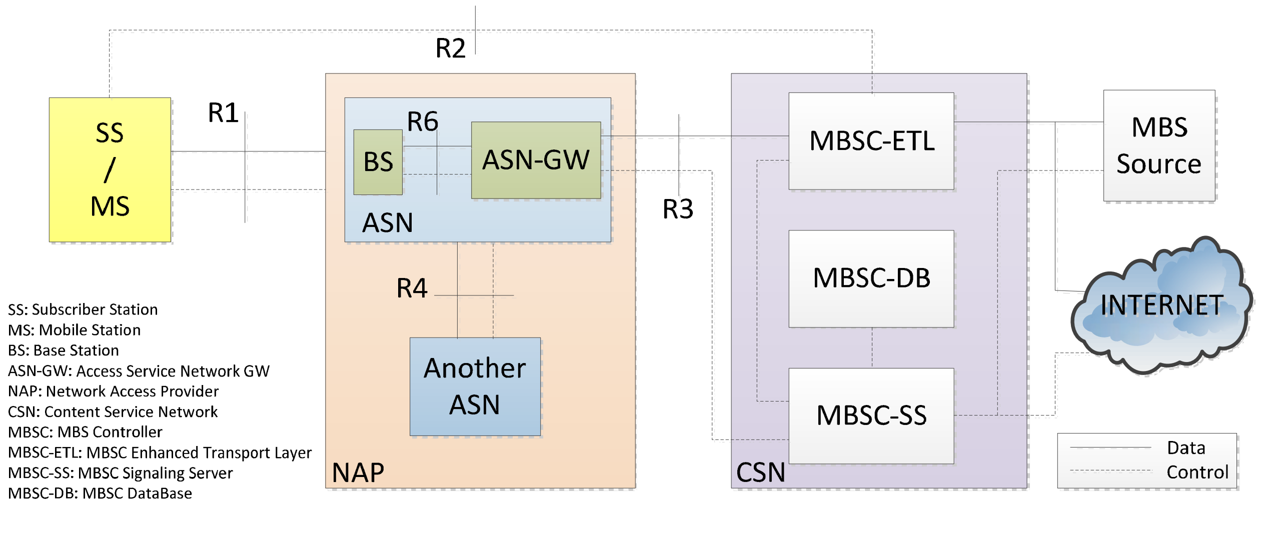

Figure 1 Proposed end-to-end architecture for MBS over WiMAX.

MBS data and for chopping contents of both announcements and data. It is also used to manage MBS zones, MCID, IP addresses, and security at MAC layer [15]. Some key reference points, R1 and R6, also are shown in Figure 1. The MBSC-SS and the MBSC-DB manage service announcements, user subscriptions, security at service level, and session state management for charging and policing. In more detail, the MBSC-SS handles all signaling messages related to providing multicast services to subscribers, whereas the MBSC-DB is the database where the status of each subscriber is stored (e.g., subscribed packages, joined multicast flows, current user activities using multicast services).

The MBS is used to offer data services, including real-time and non-real-time multimedia streaming, together with relevant service announcements. Since the NRM is IP-based, streams can be transmitted from either a dedicated multicast server (e.g., a content provider directly connected to the CSN) or an Internet streaming server.

3.1 Signaling

As mentioned above, providing MBS over WiMAX is not trivial for many reasons, such as the need of using a return channel for interacting with the network entities and dynamically subscribe/unsubscribe services. Thus, in what follows we focus on the design and the evaluation of a suitable signaling architecture for multicast services. A simplified version of our architecture can also be used to provide broadcast or *push*-type services.

Our solution is IP-based and it includes standard IP signaling protocols only, such as SIP and IGMP. This way, the scope of the signaling architecture is not limited to WiMAX access networks, but it includes most of existing data networks and is accessible by most existing terminal devices.

SIP is the main signaling protocol of IP multimedia services [24] due to both its flexibility in supporting different signaling scenarios and its reliability, also when messages are transported by UDP segments. In addition, SIP is the *de facto* standard protocol for VoIP communications and it is central in the 3GPP IMS [20]. However, it is worth noting that in this context the concept of MBS session is very different from the call session typically used in the Internet telephony field. In fact, in our case signaling is used to set up a unidirectional point-to-multipoint media flow, whereas in the Internet telephony it is typically used to set up a bidirectional, point-to-point media flow. Also the pricing scheme is different. In Internet telephony each call is typically charged on a per-second basis [27], and the use of SIP calls handled by SIP back-end servers on the signaling path is necessary to correctly apply charging policies. Instead, in the scenario proposed here, it is much more suitable a per-session charging. Thus, we have decided to use simple, session stateless, SIP MESSAGE requests to wrap data exchanged between MBS entities [4,28], instead of the classic SIP INVITE used in IMS-based systems [18,19,21,29]. This approach simplifies signaling management (e.g., see [21]) and maintains user state consistent by keeping its information updated in the database running in the MBSC-DB.

Finally, IGMP is used to manage multicast groups. Clients use IGMP to report their multicast group memberships to any immediately neighboring multicast router. There are essentially three IGMP message types: Report, Query, and Leave Group. A Report message is used in two situations. When a client wants to receive a specific multicast stream, it sends out an unsolicited Join Report message to the local router in order to join the group. Another situation is when the client passively generates a Membership Report for its interested groups in response to a Query message. A Leave message is sent by a client when it leaves a multicast group. It allows group membership termination to be quickly reported to the router in order to release resources.

Our MBS architecture has been designed according to the service flow shown in Figure 2 (see also [10]). First of all, a user must authenticate its software client with the MBSC-SS by using its credentials, which implies that he has to log in to the system. User's credentials (user name and password or digital certificates) are usually obtained during the initial service registration, which is typically performed through a web page advertising the multicast/broadcast service (i.e., an IPTV web portal). Clearly, the registration step to activate a service has to be executed only once.

When the *first* user access to the MBS system is gained, he is asked to subscribe to categories of interest. For example, in an IPTV service user can subscribe to a package containing his favorite TV channels, whereas in a software update service user can select the software type to be updated (e.g., anti-virus or system updates). Users can add or remove subscriptions at any time, after

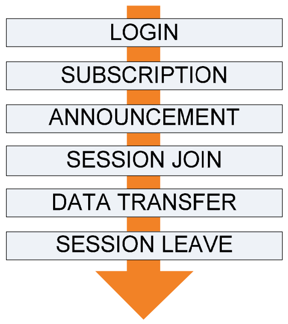

Figure 2 MBS service flow.

being authenticated by the MBS system through the *login* phase described above.

In the *announcement* phase, users are informed about the subscribed MBS sessions, such as list of TV channels and the relevant contents. An announcement can be distributed in either multicast or unicast fashion. We have implemented both approaches. In case of multicast distribution, we have provided a specific multicast channel, for each package or channel offered by the service provider, to send announcements from the MBS source to clients. In case of unicast distribution, a user periodically receives announcements relevant to its subscribed packages or channels from an announcement server, co-located with the MBS source. In both cases, either the multicast IP address of the announcement multicast group, or the IP address of the announcement server, is communicated to the client during his subscription to the relevant channel or package. After receiving an announcement, a user can join the multicast group (*session join*) relevant to the selected content and, after being authorized by the MBSC-SS, the *multicast data transfer* phase from the MBS source starts. Finally, if either a data stream ends or a user leaves the current session, the *session leave* signaling exchange closes the network bearers towards the specific subscriber and, if that user is the last one within an MBS zone, network resources are released.

To implement the aforementioned service phases, our proposal includes two main signaling flows: one for the login-subscription-announcement chain and one for the join/leave a multicast data session. In what follows, when we refer to user credentials, we assume that the considered user has already registered with the service provider prior to use the proposed MBS architecture.

In order to preserve generality, we keep client and WiMAX SU entities separated. For example, the SU could be the WiMAX interface of a vehicular gateway, and the client could run within a notebook/smartphone connected to the gateway through a WiFi connection). Clearly, in case of mobile WiMAX terminals, they are co-located. Note that being the MBSC-ETL entity part of the data plane it is not involved in the following signaling exchanges of the control plane.

3.1.1 Login, subscription, and announcement

The login, subscription, and announcement signaling exchanges are implemented by the message exchange shown in Figure 3. The initial exchange is used to authenticate the user with the MBSC-SS. A normal *SIP Register* request is sent by the client device to the signaling server. In this case, the server acts as a SIP registrar and, upon receiving the SIP request, it issues an authentication challenge to the user by sending a 401 *Unauthorized SIP* response back containing a nonce. At this point, the client can send another SIP Register request including the received nonce and the credentials requested. Since SIP is easily extendable, the second Register request could include additional headers and also a body part to help conveying authentication data to the MBSC-SS. For example, additional information needed could be SU MAC address, service id, public key certificate, and so on. When the server receives the second request, it checks the user identity against the MBSC-DB that contains a list of eligible users (those that have previously subscribed to the service). Finally, it

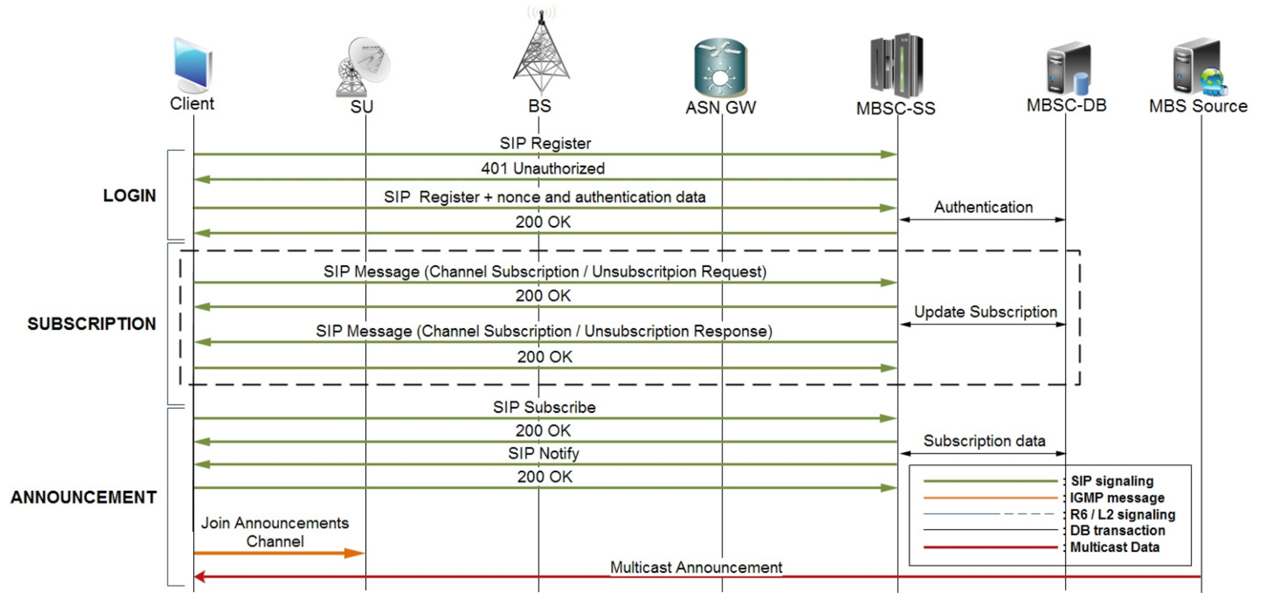

Figure 3 Signaling flow for user login, package subscription, and announcements.

sends a 200 OK response back to the client, so closing the login phase. Once a user has successfully logged in, the subscription phase follows, as depicted in Figure 3. The subscription signaling messages are used to allow customers to subscribe to multiple packages, each containing one or more MBS contents. For example, an IPTV service may allow selecting different channel packages, or a software update service may offer different software types. The user sends a SIP Message request to the MBSC-SS in order to subscribe/unsubscribe to a specific package. The signaling server MBSC-SS checks on the MBSC-DB whether the user is allowed to receive the specified service and responds by sending another SIP Message back to the client. Each SIP Message request is followed by an empty 200 OK response on the opposite direction [28], shown in Figure 3. Clearly, the subscription phase is not executed at each log in, but only when the user wants to change the list of subscribed packages.

Once logged in and subscribed to a service package, the client device needs to discover the multicast address of the announcement stream in order to receive the relevant service announcements. This is achieved by sending a *SIP Subscribe* request to the MBSC-SS. The signaling server sends a 200 OK response back and checks on the DB the appropriate announcement channel(s) for that user. In fact, the announcement channel could vary depending on the content of the service being advertised and on the user type or permissions. Once the multicast IP addresses and ports used to transport announcements are selected, the signaling server can send them to the client within a *SIP Notify* message. The client then acknowledges the notification by sending a 200 OK response back and then proceeds in joining the announcement multicast groups. When the user requests to join an announcement multicast stream, the client device sends an unsolicited IGMP report, shown as an orange arrow in Figure 3, in order to join the multicast group and receive announcement data. More details on how to join a multicast group in our MBS over WiMAX architecture will be given in the next section and illustrated in Figure 4. Announcement data can also be made available in a unicast fashion, for instance by downloading a file from a web server using HTTP; in this case the *SIP Notify* request sent by the MBSC-SS to the client includes a set of URLs used by the client to access the announcement file. At this point the client is able to receive the announcement data of the service provider from the MBS source. Thus, the user has all the information needed to join a specific MBS session.

In our solution, we have encapsulated signaling messages into the body of *SIP Message* requests. The reason of this approach is that SIP Message requests can be issued at any time and they do not require any SIP session to be established and managed by a signaling server, thus decreasing the overall MBSC-SS load. In addition, a SIP Message can contain different payload types, including contents encoded also in binary format and not only in the text-based format typical of SIP, even though the original design purpose of SIP messages was to carry instant messages in VoIP communications [28].

3.1.2 Session join/leave and session refresh

The signaling flow for a client joining or leaving an MBS session over WiMAX is shown in Figure 4. Before going into details, we point out that all the SIP protocol messages used in Figure 4 are *SIP Message* requests. For

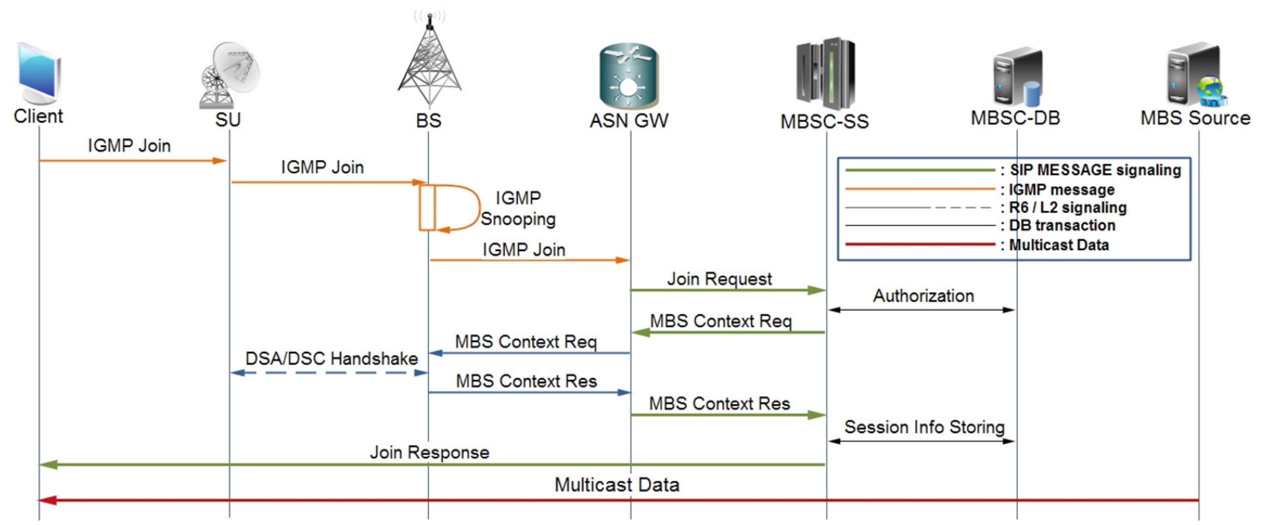

Figure 4 Signaling flow for session join.

the sake of neatness, all SIP 200 OK responses are not shown in Figure 4.

In order to join a multicast group, the client becomes aware of the multicast address and port through the *announcement* phase and issues an unsolicited IGMP report. This report is forwarded by the SU to the BS, and from the BS to the closest multicast capable router which, in our scenario, is the ASN-GW. In order to optimize traffic delivery, the BS *can* "snoop" the IGMP report [30] and send it to the ASN-GW only. Once the latter receives the IGMP report, a new SIP *Message* request, called *Join/Leave Request*, is created and sent to the MBSC-SS. The message body of the *Join/Leave* request contains both user information and the IGMP report that has triggered it. The signaling server parses such an information and uses it to check whether the user is allowed to join the desired session. For example, the MBSC-SS can check on the MBSC-DB if the user has subscribed to that session. If the authorization check is successful, the MBSC-SS issues an MBS Context Request towards the ASN-GW in order to either trigger the creation of a new multicast context or reuse an already existing one for the new client. When the ASN-GW receives the MBS *Context* request, a message exchange between ASN-GW and BS is initiated through the R6 reference point. If this signaling exchange is successful, it triggers a BS-initiated three-way Dynamic Service Addition handshake, carried out at MAC layer, between the BS and the SU of the client. Once the multicast context is created, the BS notifies the ASN-GW by sending it another R6 message, called MBS Context Response. The ASN-GW, in turn, creates a SIP Message to notify the MBSC-SS that it can save the session information on the MBSC-DB.

The final step of the procedure consists of another SIP message, called *Join/Leave Response*, sent by the MBSC-SS to the client (see Figure 4), and used to notify it whether his request was successfully processed by the signaling entities. It is worth noting that this message is not mandatory, and could be avoided in order to save bandwidth, unless it is used to transport some information needed by the application to receive the content. For example, it could transport a cryptographic key needed by the client software to decode the streamed data, or custom data enabling more options for service providers, such as live advertisements or notifications. When the packets of the joined session are routed to the BS, they are delivered to the SU through the established multicast connection over the air.

When the client leaves the multicast group, the same signaling flow as shown in Figure 4 is triggered by sending an IGMP *Leave* message to the neighboring multicast router (ASN-GW). This signaling might cause the deletion of the over-the-air connection by means of BS-initiated *Dynamic Service Deletion* messages, transmitted on the Primary Management Connection of the MS/SS, if the leaving SU is the last multicast user for that multicast connection on the BS, and the consequent release of the unused wireless resources.

To refresh an already established session, we make use of the IGMP soft-state refresh, as shown in Figure 5. By default, every 125 s, the first multicast-enabled router on path (in our case the ASN-GW) issues a multicast IGMP query message. All users interested to the advertised multicast group, must respond to this message. Every IGMP report sent by the users is in turn encapsulated by the ASN-GW into a SIP Message containing a *Join* request and sent to the MBSC-SS. When the message arrives at the MBSC-SS, the signaling server becomes aware that it represents a session refresh by checking the MBSC-DB, and it updates the session information accordingly. Finally, the refresh may be acknowledged to the user by a *Join* response, if this message is requested by the application.

Another option consists of defining a timer inside the MBS client that sends unsolicited IGMP messages towards the ASN-GW. In this second case, the timer for the IGMP query sent by the ASN-GW could be relaxed or disabled at all, since clients autonomously refresh their status. It is easy to show that the second solution does not imply additional overhead with respect to the first one, which is analyzed in detail in the article. In this second option, when the ASN-GW receives the IGMP message, and encapsulates it inside the payload of a SIP message towards the MBSC-SS, it includes additional information such as the BS serving the MBS client, which, in turn, has an its own identifier. When this information is received by the MBSC-SS, it checks the data stored in the MBSC-DB to determine if the previous status of the MBS client is changed. This step allows the MBSC-SS to realize if the client has changed the serving BS, and thus to trigger, if necessary, a signaling procedure to update the system by using exactly the same messages already defined (i.e., MBS Context Req and MBS Context Res). Thus, this way of performing session refresh could allow an easy way to maintain active the multicast session in case of client mobility.

Differently from [17], where the BS infers its subordinate interest of SU in a multicast group from the MAC-layer management messages, in order to avoid periodic exchange of IGMP reports, we prefer to not include additional, mandatory functions in the BS. Thus, the IGMP exchange occurs on the wireless channel in a standard way, since the overhead of exchanging IGMP message is small (see Section 5.2).

If either a user leaves the session abnormally, thus without issuing an IGMP leave message, or the client software crashes, such a situation is handled at the

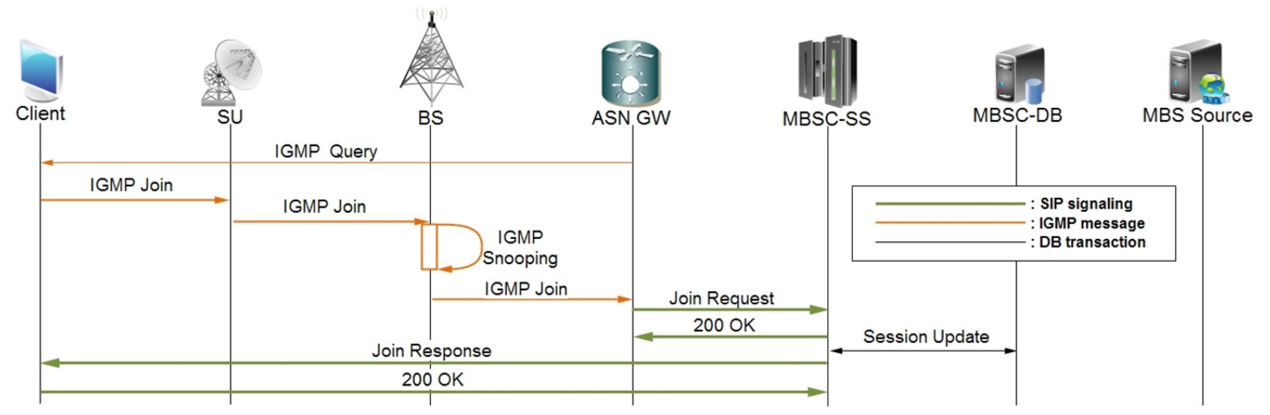

Figure 5 Signaling flow for session refresh.

MBSC-SS level. Within the signaling server, we have designed a further simple service, which periodically checks the MBSC-DB to make sure that there are no users with active sessions that have not been refreshed for more than a fixed amount of time. If it happens, the MBSC-SS issues an MBS Context Request that acts as an unsolicited, MBSC-SS-initiated leave procedure (see Figure 4). This periodical check is very useful to release resources on the wireless interface when the user that got stuck was the last one in his multicast session. A good choice for the value of this timer is an integer multiple of the IGMP refresh period (the default value is 125 s).

4. Prototype implementation

In this section, we illustrate the implementation of the testbed used for validating the proposed service architecture, in particular the end-to-end signaling architecture for MBS over WiMAX. Thus, we focus on the entities involved in the signaling process only and exclude those belonging to the data path, such as the MBSC-ETL, that has already been described in the literature [2,14,15].

The testbed has been designed to provide a multicast IPTV service as an instance of a typical MBS over WiMAX service. In what follows, “TV channel” and “multicast session” share the same meaning.

Five main entities are involved: MBSC-SS, which is the core of the proposed signaling service architecture, MBSC-DB, ASN-GW, client, and MBS Source.

4.1 MBSC-SS

The MBSC-SS is the key functional entity that orchestrates all the components used to enable, create, and manage the MBS IPTV streaming service. In order to fulfill the requirements illustrated in Section 2.3, the Mobicents JSLEE (MSLEE) has been selected to implement the central signaling server. It handles all the incoming SIP signaling messages from the MBS clients and the ASN-GW. In addition, it performs user authentication, service subscription, session join/leave, periodic check of active users, and their authorizations.

The MSLEE service, running in the same MBSC-SS where the signaling logic resides, has been implemented in a single SBB, called MessageSbb. As mentioned above, all the SIP sessions used in our architecture are stateless, differently from the typical INVITE-initiated VoIP call dialogs [31]. Thus, the SBB has been configured to process all incoming SIP events fired by the SIP RA as a root SBB. Specifically, each incoming SIP request is processed by a different MessageSbb entity, which is identified by the Call-ID value of the originating SIP request. As a design choice, requests belonging to the same transaction (e.g., MBS Context REQ–RES) share the same Call-ID value and are processed by the same SBB entity, thus increasing the MSLEE efficiency.

The format of data encapsulated into the body part of the SIP Message requests is based on the Cisco® R6 interface [32], which consists of a set of control and bearer plane communication protocols between BS and ASN-GW [33]. In short, in our implementation each data packet is encoded as a binary Type-Length-Value (TLV) data structure and located within the body field of a SIP Message request. After receiving such a request, the recipient replies with a final SIP response with an empty body (i.e., 200 OK) [28]. The ASN-GW encapsulates the R6 message into the body of SIP Message requests. In this way, the modifications needed in the ASN-GW are marginal, since it only has to encapsulate the relevant R6 and IGMP protocol messages into a SIP message body. Thus, they affect the implementation effort and the complexity of the signaling architecture on the MBSC-SS only. If needed, given the SLEE flexibility and extendibility, the MBSC-SS can easily be modified, so as to support other R6 protocols, different from the Cisco® version.

When a MESSAGE is received, its content is parsed by using the *parser* Java class that decodes and creates Java objects from each TLV (e.g., BsMacTLV, BsIpTLV, SessionIdTLV, etc.). Each TLV field and each packet exchanged have been modeled by using a different Java class, as shown in the UML class diagram shown in Figure 6. The parsed data, wrapped into a Java object, are then processed inside the SBB class, and it is used to both query/update the database and send subsequent requests.

The signaling management service runs on top of Mobicents JSLEE v.1.2.4.GA, which is executed in the JBoss AS v.4.2.2. The designed service makes an extensive use of the external database (MBSC-DB), hence the MySQL Connector/J v.5.1.6 JDBC, which is the official JDBC driver for MySQL, has been deployed onto JBoss and configured to access MBSC-DB from the MBSC-SS service class.

The *MessageSbbclass* has also been equipped with a management interface by means of a Management Bean (MBean) class called MessageSbbConfigurator (Figure 6). Through the management interface, an administrator can change configuration parameters of the MBS service at runtime without stopping or even pausing the JSLEE server. The MBean exploits the Java Management eXtension (JMX) interface provided by JBoss, thus the parameters are easily accessible through a simple JMX web interface.

4.2 MBSC-DB

All the service-related information, such as user subscription and registration data, channels, packages, and configuration variables, are stored and maintained in the MBSC-DB.

The MBSC-DB is implemented by a Linux box running the open source MySQL database server v.5.0.75 as our Data Base Management System (DBMS). This database stores user data, package composition, channel information, ongoing sessions, subscriptions, and scheduled streams by using different tables. Each table has been designed by optimizing the memory space occupied by each field. The database can be managed using, e.g., MySQL Administrator or the phpMyAdmin free software tool [34].

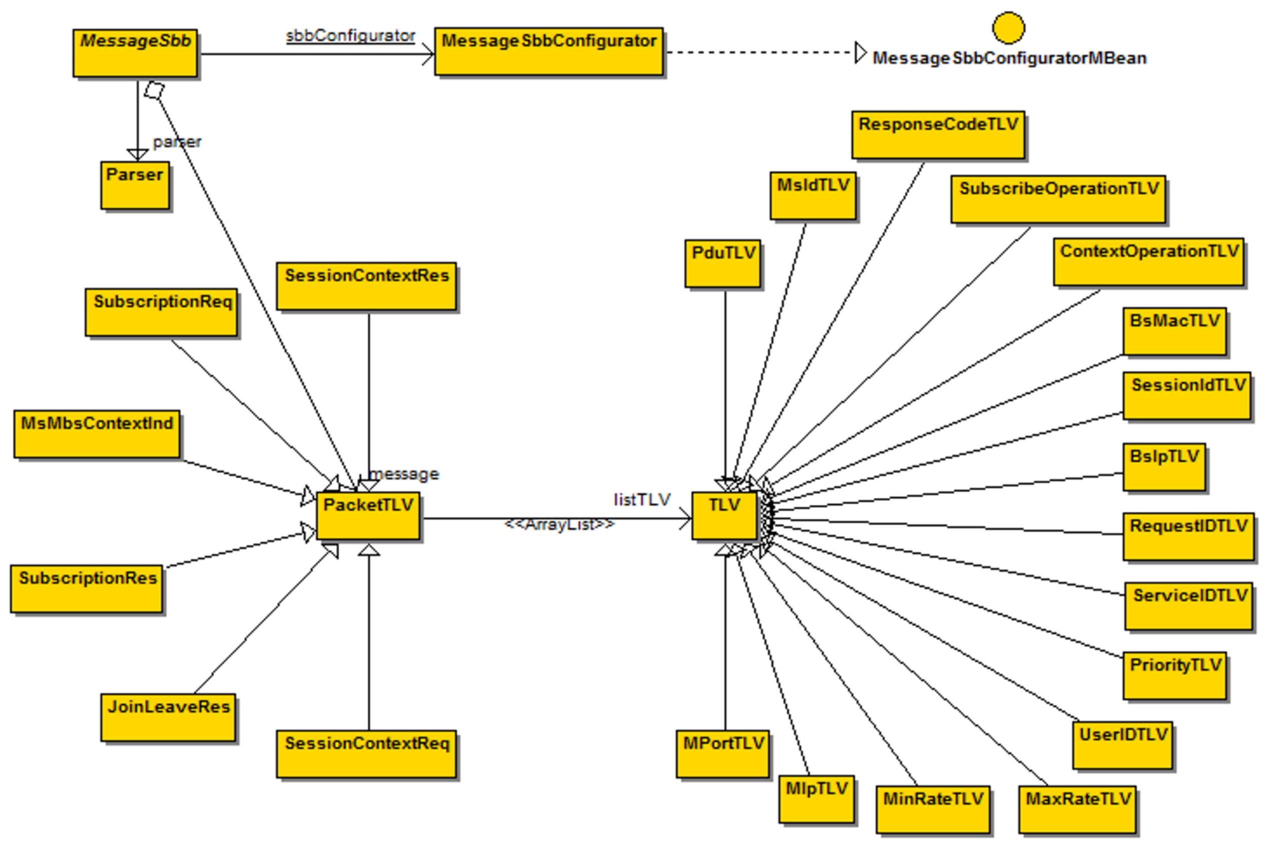

Figure 6 UML class diagram of the JSLEE service.

4.3 ASN-GW

ASN-GW, which is placed at the edge of the ASN, is the entity that connects the ASN to the CSN. The ASN-GW assists mobility and security in the control plane and handles IP forwarding. In our architecture, the ASN-GW also connects the WiMAX entities with the MBS service entities. It translates the protocol messages from the R6 protocol to the SIP protocol.

The testbed design includes a custom-built C++ software module, the WiMAX Access Network Emulator (WANE) that emulates the functions of the WiMAX access network entities (including the SUs, the BSs, and ASN-GW) which are significant for our use case. In more detail, our WANE is in charge of (i) capturing IGMP reports coming from the client by using a raw socket, (ii) encapsulating them into R6 messages, and (iii) encapsulating the latter messages together with some additional information within SIP message requests, which are then sent to the MBSC-SS. Conversely, the WANE generates IGMP queries and sends them to clients. Upon receiving MBS Context Request messages from the MBSC-SS, it replies with MBS Context Response messages emulating the overall ASN. Finally, it is in charge of handling and generating 200 OK SIP messages as specified in [28]. Figure 7 shows the role of the WANE in the WiMAX access network.

The MSLEE server providing the MBSC-SS is then in charge of parsing and interpreting encapsulated data. Thus, the service running in the MSLEE is always aware of the protocol being used within the WiMAX network [32].

Given our focus on the IP part of the MBS signaling architecture, we treat the WiMAX network as a black box. Thus, mobility effects, modulation parameters, and wireless loss rates and retransmissions are not emulated by our WANE module and are not included in our performance evaluation results.

Finally, it is worth citing that we also executed functional tests of the overall system in the laboratories of a WiMAX manufacturer, running our software on a real ASN-GW, able to dialogite with real WiMAX BSs and our MBSC.

4.4 MBS software client

The MBS software client handles the signaling messages at the client side and provides a graphical user interface (GUI) of the implemented IPTV service. The system architecture that we have designed is modular and allows developers to implement an MBS client as both a standalone application and a Web 2.0 client.

The standalone version consists of a standard Java application using the open source JAIN SIP stack v.1.2 [35] to enable SIP communication with the MBSC-SS. The same SIP stack is used also internally by the SIP RA in the MSLEE signaling server. The client is also equipped with a multimedia player compliant with the used multimedia flow formats and able to send the IGMP unsolicited report message to join a multicast group. The join message triggers all the signaling exchange described in Section 3.

The Web 2.0 version of the MBS client consists of a multi-tier Web-based application developed by using Microsoft Silverlight™ 3.0 together with PHP, Javascript, ActiveX, NPAPI, and CSS, all embedded in a Web page accessible through the Web portal of the MBS over WiMAX service (an additional entity in the CSN beyond those depicted in Figure 1). In this case, the announcements are transferred in a unicast fashion, and they are retrieved by periodically downloading a Really Simple Syndication (RSS) file through a simple HTTP unicast connection. An ActiveX library, for Internet Explorer®, and an NPAPI plug-in, for Mozilla Firefox®, have been included to handle SIP and IGMP messages and fire events towards the Silverlight application.

In both standalone and web cases, the GUI displays user information about subscribed and available channels and packages, announcements, and multicast streaming contents.

4.5 MBS source

The MBS Source is one of the key entities for deploying an MBS over WiMAX architecture. Nonetheless, thanks to our modular architecture, the service administrator is allowed to select the technology that better fits his

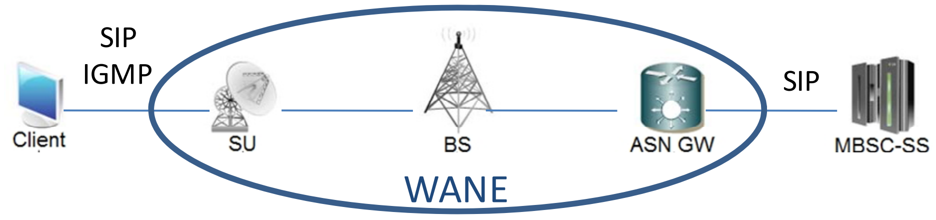

Figure 7 WANE: emulated network entities and supported protocols.

interests, without modifying the other service entities. In our IPTV experiment, this is achieved by isolating the actual streaming server technology from the service logic. The only requirement for the MBS Source is to support multicast streaming endpoints. We have used a simple VideoLAN client (VLC), running in server mode in a Linux machine, and configured some multicast endpoints to stream a sample video. Then, we have recorded the URLs of the created endpoints in the appropriate database table of the MBSC-DB. We have also created channel packages that the user can subscribe to, such as sport, movies, etc. Users can subscribe to packages and/or to a single channel. The MBS Source has also to create and send multicast announcement data on a dedicated multicast group when a standalone client is used. This is done by using the Session Announcement Protocol (SAP, [36]) functionality of the mini-SAP server bundled with the VLC software. On the other hand, if a Web 2.0 MBS client is used, then a Web server is also necessary. The Web server is used to provide users with the Web portal through which they can access the MBS Service. The Web server has been deployed by installing the open source LAMP (Linux Apache, MySQL, PHP) package on a Linux machine executing Ubuntu Server 9.1 x86. The Web portal also provides an administrative Web interface by which service providers can add, remove, and manage channels, users, and subscriptions. The Web server also dynamically creates and sends the unicast announcement data for the channels subscribed by a user. In this case, data are used to update the web page of the service with real-time information. In fact, each user, once subscribed and authorized by the MSLEE server, periodically polls the web server for announcements encoded as an RSS news file. The client GUI is then in charge of parsing XML files and visualizing data in a user-friendly way. Even if not implemented in our prototype yet, the web client could have the option of using multicast announcements distributed through the SAP protocol, as it happens in the standalone client approach.

5. Performance evaluation

In this section, we show the performance metrics used to evaluate the MBS signaling. A number of experiments have been realized by using the system testbed described in Section 4. For this purpose, we have deployed the Mobicents JAIN SIP RA v.1.2.4.GA in the MSLEE server that acts as MBSC-SS. We have optimized the SIP RA load by increasing the number of threads, in order to maximize the number of simultaneous processed events. During test execution, we have stopped recording logs in order to improve MSLEE and JBoss AS performance. For what concerns the operating system, we have used the 32-bit Ubuntu server 10.04, running in a virtual machine (VM) equipped by three CPU cores and 3 GB of RAM, and managed by the ESXi 4.1 VMWare hypervisor [37]. The server running the hypervisor is equipped with a dual Intel Xeon E5410 with clock frequency equal to 2.33 GHz and 16 GB of RAM. The MSLEE has been configured with 2.5 GB Java heap, using the Parallel Garbage Collector scheme to periodically clean the memory allocated to the Java server. The choice of deploying the MBSC-SS in a VM allows using computing resources efficiently, implementing energy efficient policies, simplifying replication of ASs along with their failover management, and so on. An interesting reading about virtualization advantages can be found in [38]. The DBMS has been deployed in a different VM having the same characteristics of the MBSC-SS VM and executed in the same server.

In our experiments, clients have been emulated by using the well-known SIPp traffic generator [39]. This choice is due to the unavailability of high performance IGMP message generators. Thus, by using SIPp, the client generates SIP messages with the IGMP report information already encoded in the payload, thus emulating the output of the ASN-GW towards the MBSC-SS. In turn, the WANE has been modified with respect to its base version, in order to simply forward the SIPp-generated messages to the MBSC-SS, without modifying them. By using this configuration, we have collected significant statistics, such as the number of lost SIP messages and the signaling latency, by using the built-in logging capabilities of the SIPp. In particular, we have measured the throughput and latency of the core component of our architecture, the MBSC-SS. We have checked that the WANE was never the system performance bottleneck in our testbed implementation. In addition, we have measured the average signaling protocol overhead of each message exchanged between the MBS entities, and compared it with a typical IMS-based system. In all service lifecycle phases, per-user and per-BS frequency overhead have been evaluated. Finally, we have evaluated the maximum number of users that our testbed can support while keeping a good QoS level in terms of signaling latency and service availability. In particular, we consider a maximum tolerable channel switching time of 100 ms and service availability equal to, at least, 99%. As mentioned in Section 4.3, it is worth to remember that our focus is the performance of the IP part of the MBS signaling architecture, thus the WiMAX network may be simply regarded as a "black box" emulated by our WANE module.

5.1 Throughput and latency

We define a *service transaction* as the signaling exchange which starts with an IGMP message sent by the client, and ends when a SIP reply is sent by the

MBSC-SS back to the client (see Figure 4). Therefore, this signaling exchange corresponds to a Join or a Leave operation. We use the service transaction as the basic message exchange to evaluate performance in terms of signaling throughput and latency. Since it is the most frequent and complex signaling exchange of our signaling architecture, it is a meaningful benchmark for evaluating the processing capability of the MBSC-SS. As for the service transaction latency, this is a significant parameter, since it represents the contribution of the core elements of the signaling architecture to the channel switching time. It is defined as the delay between the IGMP message generation (directly encapsulated within a SIP message) at the client side and the reception of the Join Response message.

The SIPp client, which emulates the MBS client, has been configured to generate SIP Messages according to a deterministic arrival process, at a fixed rate λ. Each test has been executed by using the same λ value for 60 min. Both TCP and UDP protocols have been used for transmitting SIP messages. In case of TCP usage, SIPp is configured to work in a mono socket mode.

Figure 8 shows the achieved throughput versus offered load, both expressed in terms of service transactions/s. The first observation is that TCP outperforms UDP significantly. In particular, TCP can reach 210 service transactions/s with a loss rate lower than 1%, whereas UDP can reach only 70 service transactions/s with the same desired maximum loss performance (75 service transactions/s if the loss target is increased by 5%). This large difference is due to the fact that, by using TCP, retransmissions do not occur at the application layer (i. e., SIP) as it happens by using UDP. In fact, if UDP is used, retransmissions are managed by the SIP RA of the MSLEE server, and this can cause avalanche restart effects immediately after a full garbage collection (i.e., after the Java heap cleanup). Note that this operation is quite different from the classical SIP call management in IP telephony where the three way SIP handshake (INVITE-200 OK-ACK) is used, since in that case the SIP layer manages the 200 OK retransmissions directly, even if TCP is used (see also [31]). When using TCP, retransmissions are managed more efficiently by the transport layer at the kernel level and not by the SIP layer in user space. This means that when TCP is used, if the MBSC-SS server get overloaded, it experiences packet losses in the internal queues of the JSLEE application server (e.g., during the garbage collection process). However, since TCP is used, SIP disables retransmission timers, thus the sender knows, from the TCP ACK, that the packet has correctly been received and will not retransmit it. Nevertheless the receiver, which is overloaded, has lost the packet and there is no means of recovering it, and the service transaction fails. Thus, when this process begins, a large number of service transactions will fail and the harsh behavior shown in Figure 8 happens. Note that we have assumed a perfect wireless network channel, meaning that there are no packet losses at the MAC layer, so as to evaluate the MBSC-SS worst-case signaling scenario.

In UDP, a packet lost in the internal queues of the JSLEE application server will always be retransmitted with increased timer value. Thus, overall throughput is much lower, but the decrease is smoother.

Figure 9 illustrates the average latency, expressed in seconds, associated with a service transaction, as a function of the offered load (service transactions/s). The ordinate axis has a logarithmic scale. Again, TCP exhibits superior performance, since it is possible to maintain the latency, which affects the IPTV channel switching time, below 100 ms even if the offered load grows up to 210 service transactions/s. This latency value is compliant with the specifications provided in [26]. On the other hand, by using UDP, a similar latency performance can be obtained only if the offered load is lower than 60 service transactions/s. This is due to the need of managing SIP message retransmissions by the MSLEE RA. The slope for UDP is much steeper than that of TCP since, when the MBSC-SS becomes overloaded; it is not able to process all SIP messages before timer expiration. This implies an increased number of SIP packets to be managed (original packets and retransmissions), which first experience higher delays (in the order of hundreds of ms, since first retransmission occur after at least 500 ms), and then are discarded (service transaction failure), thus having a nearly constant delay independently on the offered load. This almost flat delay of about 20 s is due to a maximum value of SIP timer equal to 32 s. When TCP is used, retransmissions are managed at a lower layer and the process is much more efficient. However, this is a special case relevant to SIP messages only, since in case of SIP sessions with INVITE messages they are managed differently.

Consider also that the additional latency due to the over-the-air data transmission is not included in our tests, thus it is really important to keep the measured latency value as small as possible in order to have an acceptable service in operation.

By using the maximum throughput values, expressed in service transactions/s, obtained by using either TCP or UDP, we have calculated the maximum number of clients that our MBSC-SS can simultaneously support. We assume that all users are already logged in. This means that they need to issue one service transaction in order to be authorized to join the multicast data flow, an additional one to notify their intention to leave the multicast stream, and a number of service transactions are used to refresh the current multicast session, by

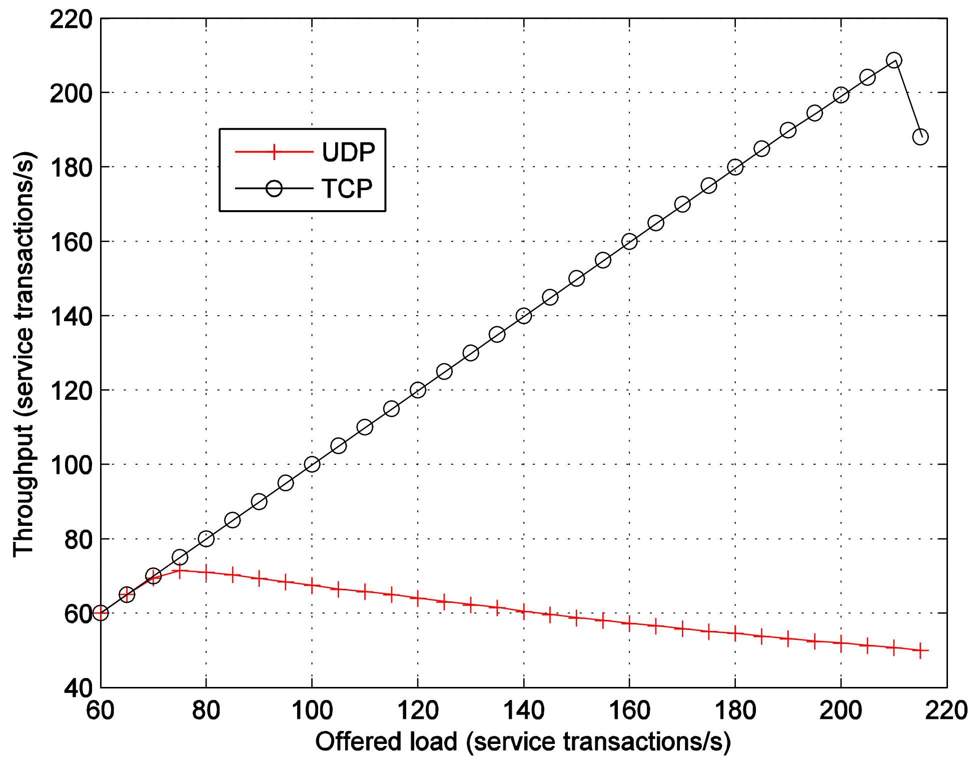

Figure 8 Throughput versus offered load.

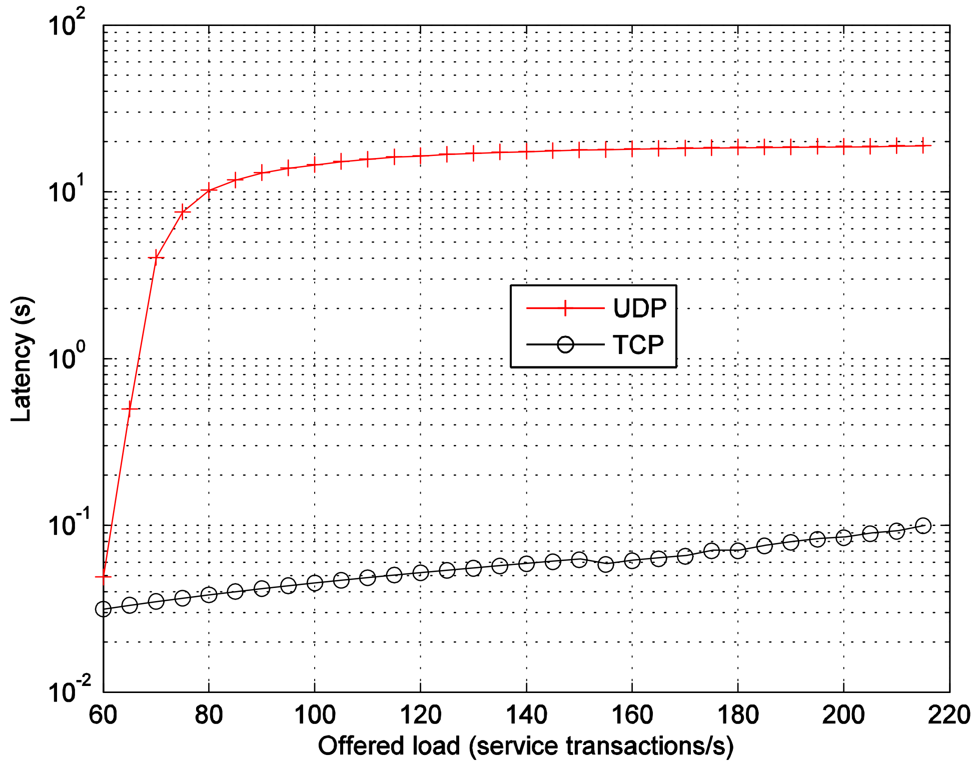

Figure 9 Latency versus offered load.

answering the Query sent by the ASN-GW with an IGMP Join message (the relevant signaling exchange is shown in Figure 5). Clearly, in this case the only parameter that can influence the number of supported users is the IGMP query period value, since it determines the number of service transactions per user. Note that during a session refresh (see Figure 5), the number of SIP messages exchanged by the MBSC-SS is lower than those exchanged during a service transaction (see Figure 4). Thus, our evaluation is a lower bound of the real capacity of the proposed implementation of the signaling server.

To calculate the maximum number of users N_a supported by our system, and considering movie TV channels as use case, we assume an average flow duration FD = 5400 s (i.e., one and a half hour, which is a typical movie duration). The maximum number of session refresh transactions (*NSTF*) for a multicast session is given by Equation (1). Table 1 reports all the parameters used in the following performance evaluation equations.

By definition, the *NSTF* value depends only on the timer of the IGMP queries and it is equal to

$$\mathrm{NSTF} = 1 + \left\lceil \frac{\mathrm{FD}}{\mathrm{IGMP}_{T1}} \right\rceil. \tag{1}$$

Given the *NSTF* value and the maximum throughput of the signaling server ($\mathrm{MAX_{STS}}$) for both TCP and UDP (see Figure 8), the maximum number of users supported by our system is given by

$$N_a = \mathrm{MAX_{STS}} \cdot \frac{\mathrm{FD}}{\mathrm{NSTF}}. \tag{2}$$

Figure 10 shows the N_a values for different IGMP query interval values ($\mathrm{IGMP_{T1}}$). Note that both the abscissa and the ordinate values are expressed in a logarithmic scale. We considered $\mathrm{IGMP_{T1}}$ values ranging from 10 s (about 10 times lower than the default value 125 s) to 15 min (about 7 times larger than the default value). It appears that by changing the value of the IGMP query period, the maximum number of users supported by the system varies in a nearly linear fashion, spanning more than two orders of magnitude. In the worst case, by using UDP with $\mathrm{IGMP_{T1}}$ = 10 s, only about 800 users are supported. Nevertheless, by using the IGMP default value, $\mathrm{IGMP_{T1}}$ = 125 s, and the TCP transport protocol, the number of supported users increases significantly, up to about 26,000 (about 8,000 when UDP is used). With a lower query rate, it is possible to reach a number of users very high, up to 300,000 for TCP and 90,000 for UDP.

In this evaluation, we have done the realistic assumption that each user joins only one multicast session. In addition, we have assumed that the ASN-GW sends Group-Specific multicast Queries, which are queries relevant to the membership to a *single, specific* multicast group [5]. In this way, we avoid handling messages from users not involved in such a multicast group, i.e., in the specific case, from inactive users.

In the above evaluation, we have neglected the effect of the queries needed to maintain the membership to multicast groups used to transport announcement data. Clearly, this effect is present only when announcements are multi-casted. We suggest two possible approaches to take this effect into account and limit the performance

Table 1 Definition of symbols used in overhead formulas

Symbol	Definition	Value
$\mathrm{MAX_{STS}}$	Maximum number of service transactions per second supported by an MBSC-SS instance	210 (TCP) 70 (UDP)
FD	Average multicast flow duration, expressed in seconds	5400
$\mathrm{IGMP_{T1}}$	IGMP timer value, in seconds, for queries refreshing membership to multicast groups transmitting data flows	125
$\mathrm{IGMP_{T2}}$	IGMP timer value, expressed in seconds, for queries refreshing membership to multicast groups transmitting announcement data	1250
R_t	SIP Registration expiration value, expressed in seconds	3600
NSTF	Number of service transactions per user needed to maintain the membership to a multicast data flow of duration FD	-
N_p	Number of available subscription packages (each one with likely more than one channel)	20
N_a	Maximum number of users receiving a multicast data flow supported by a MBSC-SS instance	-
N_{BS}	Number of users served by a BS (or BS sector) on a single WiMAX channel	200
R_{OH}	Overhead per user evaluated at the IP layer associated to a SIP registration	-
A_{OH}	Overhead per user evaluated at the IP layer for maintaining the membership to N_p multicast announcement groups (one for each subscribed package of TV channels)	-
S_{OH}	Overhead per user evaluated at the IP layer for maintaining the membership to a multicast data flow	-
C_{TCP}	TCP connection setup/teardown overhead, expressed in bytes	268 (TCP) 0 (UDP)

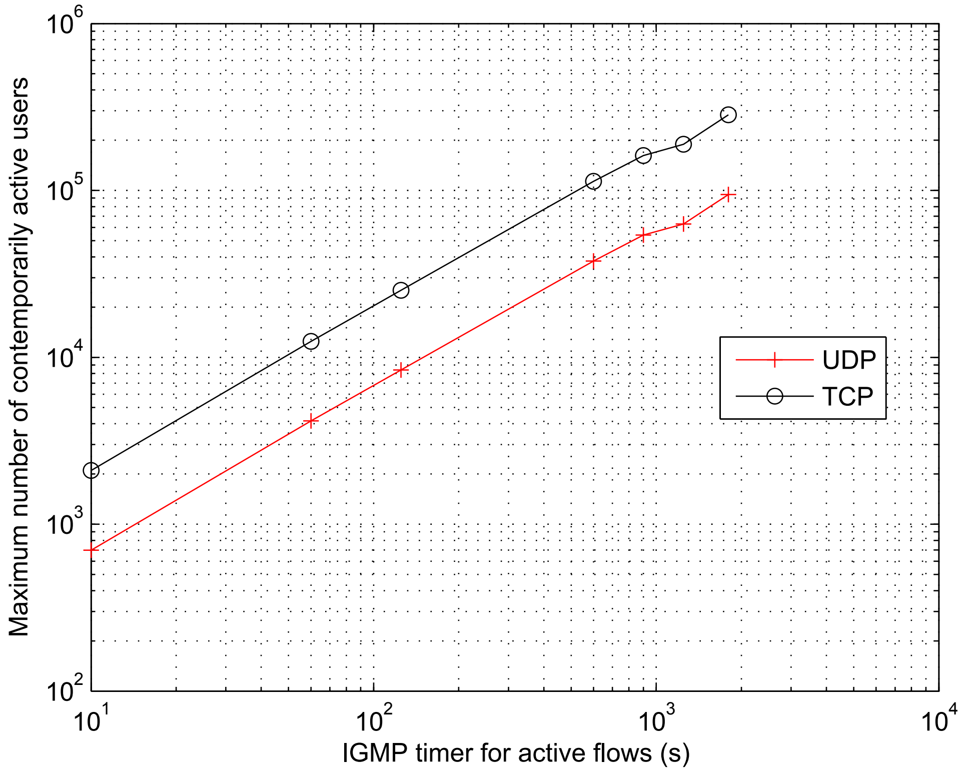

Figure 10 **Max number of users versus IGMP session refresh timer**.

penalty to a negligible level. The first one is to use an IGMP_{T2} timer value larger than IGMP_{T1} (e.g., 10 times larger), so as to have a lower rate, and thus a lower impact of the responses to queries on the signaling server. The other approach consists of both using IGMP_{T2} > > IGMP_{T1} and sending a General Query upon expiration of the IGMP_{T2} timer at ASN-GW. In this way, irrespective of the number of the multicast groups the client has joined, it will answer by means of a single Report message, listing all multicast groups [5]. In both approaches, the additional resource consumption is negligible.

5.2 Signaling overhead

Each signaling protocol produces some overhead due to control messages. Transmission of such messages requires an additional bandwidth consumption and a processing load in all involved entities. In this section, we determine the average signaling protocol overhead of our proposed MBS over WiMAX architecture. Table 2 summarizes the overall message overhead, for all messages described in Section 3.1, when no compression is used. Since we have proposed an IP-based signaling, we will evaluate the overhead at the IP layer only, thus the additional MAC overhead calculation (see [40] for details) at the R1 interface is not considered. It can be observed that the largest contribution to the signaling overhead is given by the message exchange needed by the join/leave process. In fact, when a user joins a new multicast service session, the procedure requires an exchange of five different messages: an IGMP *join* and a Join *response* between the user and the MBSC-SS on the wireless interface, a Join *request*, an MBS *context request*, and an MBS *context response* between the ASN-GW and the MBSC-SS.

Therefore, the total average amount of signaling overhead for a session setup is $I_r + J_{\text{req}} + MC_{\text{req}} + MC_{\text{res}} + J_{\text{res}} = 36 + 410 + 410 + 400 + 388 = 1644$ bytes, plus any possible retransmissions when UDP is used to deliver SIP messages. If only TCP is used, there are no SIP retransmissions at the application layer, but the average overhead increases by 12 bytes per packet plus the TCP handshake overhead when the connection is not yet established. Note that only the first *join* and the final *leave* messages produce this amount of overhead, whereas the periodic *join* messages, triggered by the IGMP queries, used to refresh data or announcement multicast sessions, only use $I_q + I_r + J_{\text{req}} + J_{\text{res}} = 40 + 36 + 410 + 388 = 874$ bytes (886 bytes in TCP) including the IGMP query sent by the ASN-GW.

Similarly, the registration process involves four different messages, even if their size depends on the type of

Table 2 Message overhead by protocol layer (expressed in bytes)

Message type	Label	IP layer	Transport layer	Signaling overhead	Overall overhead
IGMP query	I_q	24	-	16	40
IGMP report	I_r	20	-	16	36
Register	$R1$	20	8 (UDP), 20 (TCP)	320	348 (UDP), 360 (TCP)
401–unauthorized	$R2$	20	8 (UDP), 20 (TCP)	450	478 (UDP), 490 (TCP)
Register w/auth data	$R3$	20	8 (UDP), 20 (TCP)	500	528 (UDP), 540 (TCP)
200 OK	OK	20	8 (UDP), 20 (TCP)	250	278 (UDP), 290 (TCP)
Subscription req	S_{req}	20	8 (UDP), 20 (TCP)	350	378 (UDP), 390 (TCP)
Subscription res	S_{res}	20	8 (UDP), 20 (TCP)	350	378 (UDP), 390 (TCP)
Subscribe	$S1$	20	8 (UDP), 20 (TCP)	250	378 (UDP), 290 (TCP)
Notify[a]	$S2$	20	8 (UDP), 20 (TCP)	350[a]	378[a] (UDP), 390[a] (TCP)
Join/leave req	J_{req}	20	20	370	410
Join/leave res	J_{res}	20	8 (UDP), 20 (TCP)	360	388 (UDP), 400 (TCP)
MBS context req	MC_{req}	20	20	370	410
MBS context res	MC_{res}	20	20	360	400

[a]Additional 20 bytes for each subscribed package are necessary.

security support implemented in the selected user authentication procedure. In fact, if public key encryption is used, then the size of the Register messages can increase significantly. Even though the registration could easily become the largest contribution to the signaling overhead, in an operational service scenario we believe that the session *join* and *refresh* exchanges are those to be kept under control in terms of size, latency, and rate. To this aim, we will refer to the quantities reported in Tables 1 and 2.

Since the wireless link is typically the bandwidth bottleneck, in what follows we analyze the overhead only in the radio interface.

We start evaluating the average overhead rate for each user as follows. The user registration overhead, expressed in bits, is composed of five components, as follows:

$$R_{\mathrm{OH}} = (R1 + R2 + R3 + OK + C_{\mathrm{TCP}}) \cdot 8. \tag{3}$$

The initial four quantities in Equation (3) are the size of the signaling messages used in the registration phase (see Table 2) and C_{TCP} is the TCP connection setup and teardown contribution to the overhead (see Table 1).

Similarly, the overhead of both refreshing active data and announcing multicast sessions is given by

$$A_{\mathrm{OH}} = S_{\mathrm{OH}} = (I_q + I_r + J_{\mathrm{req}} + J_{\mathrm{res}} + C_{\mathrm{TCP}}) \cdot 8. \tag{4}$$

The overhead rate of the registration phase for each user is given by

$$Rr_{\mathrm{user}} = \frac{R_{\mathrm{OH}}}{R_t}, \tag{5}$$

where R_t represents the expiration value of the SIP registration, which is typically equal to 3600 s, and R_{OH} is the registration overhead calculated by Equation (3).

The overhead rates for each user needed to maintain the membership to both announcement and existing data sessions in steady state, are given by Ar_{user} and Sr_{user}, respectively

$$Ar_{\mathrm{user}} = \frac{A_{\mathrm{OH}}}{\mathrm{IGMP}_{T2}}, \tag{6}$$

$$Sr_{\mathrm{user}} = \frac{S_{\mathrm{OH}}}{\mathrm{IGMP}_{T1}}. \tag{7}$$

In fact, as shown in Figures 4 and 5, the announcement multicast session and the data multicast session are both joined and refreshed by similar procedures, which start with an IGMP message. Thus, each time they are performed, they produce the same amount of overhead on the wireless interface, even if membership refresh requires a lower number of signaling messages for the signaling server. Given the greater importance of data sessions queries than announcement ones,[a] two different IGMP query periods can be used in the ASN-GW, $\mathrm{IGMP}_{\mathrm{T1}}$, and $\mathrm{IGMP}_{\mathrm{T2}}$, which represent the refresh period of both the data and the announcement multicast sessions, respectively. Their values used in subsequent calculations are reported in Table 1. We have used a value for $\mathrm{IGMP}_{\mathrm{T2}}$ 10 times larger than the value of $\mathrm{IGMP}_{\mathrm{T1}}$. These values can be adjusted for different multicast groups in the ASN-GW configuration, and the overhead rate can therefore be optimized.

Figure 11 shows the average per-user signaling overhead rate in the wireless interface. All service phases are evaluated and compared from a transport protocol point of view. When a user watches a TV channel, the largest contribution is given by the membership refresh process for the active multicast data flow. In this case, a rate of 48 bit/s is reached by using UDP, whereas the TCP usage provides 67 bit/s, which is 40% higher. A value of IGMP_{T1} = 125 s has been used as IGMP query interval for TV channel multicast groups. The announcement and registration overhead is much smaller than the active flow management one, and is lower than 10 bit/s per user in both cases. In fact, for the former we have defined IGMP_{T2} = 1250 s and for the latter we use a registration refresh time R_t = 3600 s. The overall amount of per-user overhead is thus about 80 and 60 bit/s, by using TCP and UDP, respectively. If we assume that a user receives a multicast video streaming flow in low resolution (e.g., 500 kbit/s), the ratio between the overall signaling overhead and the data traffic is about 1.6×10^{-4} for TCP and 1.2×10^{-4} for UDP, respectively. The overall comment is that the per-user overhead required by our solution is definitely affordable in a WiMAX network. Also increasing the rate of queries, the overhead is very low.

By using the single user overhead rate, we can estimate the amount of aggregate overhead rate, consumed by a group of 200 users being served by a single MBS frequency under a single BS. We assume that each user is already registered and is watching a TV channel. In addition, all users are subscribed to N_p = 20 different packages and receive the relevant announcement data.

The aggregate overhead bit rate, relevant to all users served by a single BS on an MBS frequency required for keeping the registration state active within the MBSC-SS, is given by

$$Rr_{\text{freq}} = \frac{R_{\text{OH}} \cdot N_{\text{BS}}}{R_t}, \tag{8}$$

where N_{BS} is the number of users served by a single BS on an MBS frequency.

The aggregate overhead required for maintaining the membership to both announcement and active sessions in steady state, considering all users served by a single BS with an MBS frequency, is given by, respectively

$$Ar_{\text{user}} = \frac{(A_{\text{OH}} - I_q \cdot 8) \cdot N_{\text{BS}} + I_q \cdot 8}{\text{IGMP}_{T2}} \cdot N_p, \tag{9}$$

$$Sr_{\text{freq}} = \frac{(S_{\text{OH}} - I_q \cdot 8) \cdot N_{\text{BS}} + I_q \cdot 8}{\text{IGMP}_{T1}}. \tag{10}$$

In Equations (9) and (10), we consider a single IGMP query triggering a number N_{BS} of IGMP reports, as in the standard IGMP settings. This happens N_p times in the case of announcements (we assume, as worst case,

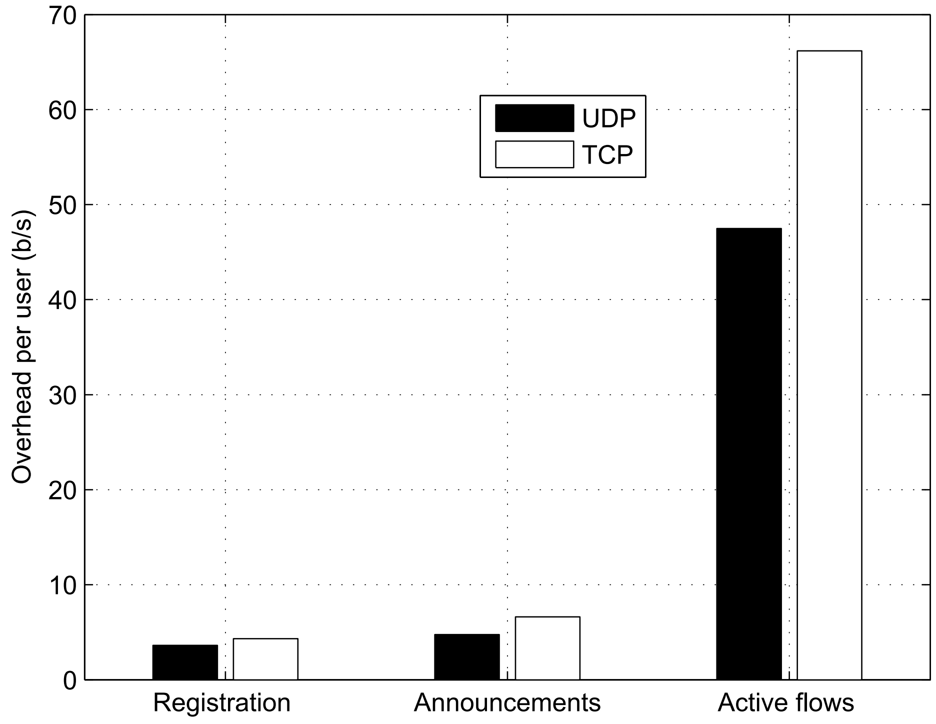

Figure 11 Per-user signaling overhead in bit/s using TCP or UDP for each service phase with R_t = 3600 s, IGMP_{T1} = 125 s, and IGMP_{T2} = 1250 s.

that all users subscribed all N_p packages) and only once for the TV channel that the user is currently watching. Similar to Equations (6) and (7), also in Equations (9) and (10), we assume that the IGMP query periods of announcement multicast channels are longer than for those of normal data channels ($\mathrm{IGMP}_{T2} = 10 \times \mathrm{IGMP}_{T1}$).

In Figure 12, we can observe that when a large group of users is considered, the largest contribution to signaling is the membership update of the announcement channel sessions. In fact, by using UDP this contribution is equal to 18 kbit/s, whereas using TCP it grows up to 25 kbit/s, i.e., 40% more than using UDP. In both cases, it is quite low. Keeping the TV channel session open for all users under the same BS and frequency requires less than 10 kbit/s for UDP usage, and 13 kbit/s for TCP. For this reason, it is very important to differentiate the IGMP query interval timer. For updating a registration, the aggregate overhead rate is below 2 kbit/s for both TCP and UDP.

The overall aggregate overhead rate for 200 users is then around 40 kbit/s for TCP and 28 kbit/s for UDP. If we consider that each announcement channel, using the SAP protocol, is limited to a maximum of 4 kbit/s [36], and that the streaming TV channel can easily use more than 1 Mbit/s, then the overall overhead percentage of our signaling architecture is very low. In more detail, assuming the *worst* case for the signaling overhead, in which 200 users subscribed to N_p packages, all watching the same channel with a data rate equal to 500 kbit/s (e.g., low-quality video), the ratio between the maximum overhead (if TCP is used) and aggregate data rate is given by the ratio 40/500 = 0.08, without any compression. Clearly, this value is definitely affordable.

As already mentioned in Section 2, in the IMS-based architectures the use of the SIP INVITE message is mandatory to create a session state and to provide an IPTV service [18,19,22,23]. Even other proposals [21] try to extend the IMS session setup for multipoint-to-multipoint service type, using again INVITE messages. Conversely, the signaling overhead of our architecture is significantly lower than that of an IMS-based one, due to the usage of simple Message requests. In [29], the overhead of the IMS architecture is analyzed together with the signaling latency, and in both cases our architecture proves to save bandwidth and provides lower latency values for initiating a service session.

5.3 A deployment scenario

In designing our solution, in order to minimize the signaling architecture complexity, we have used only messages that do not require a SIP session to be established. In fact, by both using simple SIP Message requests and by saving and recovering the transaction state on the MBSC-DB, our architecture is easily deployable in a carrier-grade scenario, as shown in Figure 13.

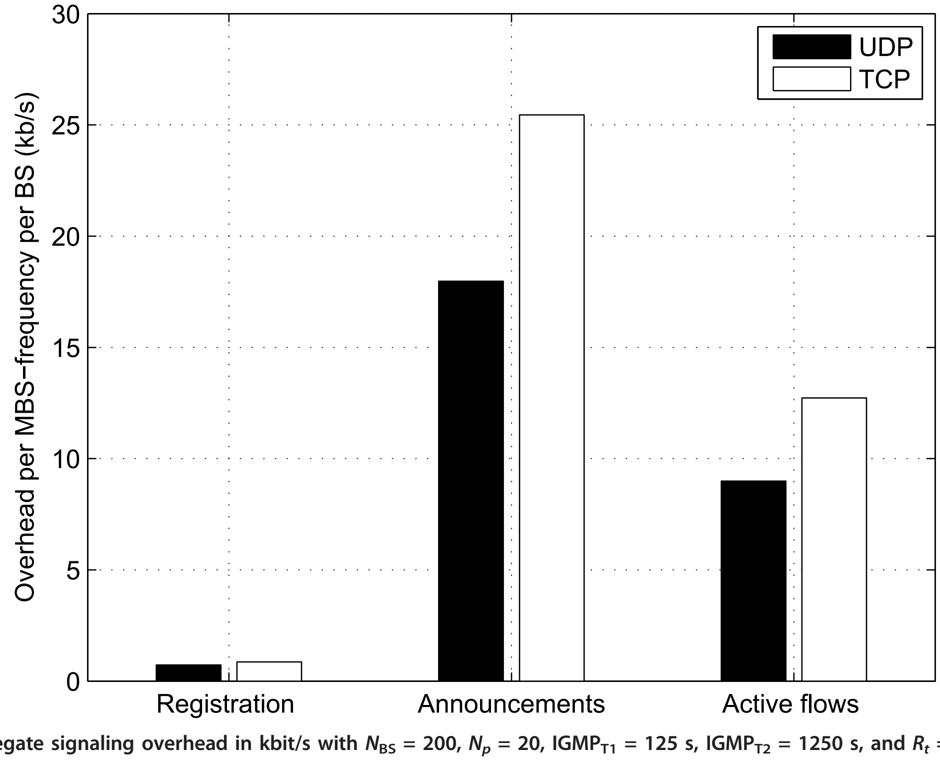

Figure 12 Aggregate signaling overhead in kbit/s with N_{BS} = 200, N_p = 20, IGMP_{T1} = 125 s, IGMP_{T2} = 1250 s, and R_t = 3600 s.

The MBSC-SS is an AS and thus it is usually deployed on a cluster of servers in the CSN. It serves a number of ASN-GWs, each providing Internet connectivity to a set of BS. By using a cluster of servers in the MBSC-SS, critical situations, such as failures and overload, can easily be managed without compromising the user's perceived grade of service. Moreover, it is also possible to increase throughput and decrease latency by using different small servers in parallel rather than using a single powerful node processing all messages. In fact, recent studies (see [41] and references therein) show that Java-based ASs, especially SIP servers, perform better if they are executed in a VM with a limited number of CPU cores (e.g., two or three CPU cores), and scalability is efficiently obtained by deploying multiple VMs on the same physical server, each one hosting a replica of the AS. In this deployment, the entry point of all SIP signaling messages in the MBSC-SS is a SIP proxy load balancer, in charge of dispatching SIP messages among AS replicas. This task can easily be accomplished by using a SIP proxy such as SIP Express Router, which can provide high throughput also on low end hardware [42]. In addition, it can also act as a SIP registrar during the registration phase of the service [42].

In a cluster scenario, like the one shown in Figure 13, the session-oriented nature of SIP has important implications for load balancing. In a typical VoIP scenario, all transactions relevant to the same call must be routed to the same server; otherwise, the server will not recognize the call. Session-aware request assignment is the process through which a system allocates requests to servers such that sessions are properly recognized by that server, and subsequent requests, corresponding to the same session, are allocated to the same server [43]. In our case, in which no SIP INVITEs are used, there is no need of complex algorithms. On the contrary, a simple static approach to load balancing (such as a call-id hash algorithm [43]) is enough to ensure that messages with the same call-id, hence belonging to the same MBS service transaction (as defined in Section 5.1), are routed to the same server within the cluster. In addition, when the MBS service transaction (e.g., the Join procedure) is completed, in our architecture any future update of the user state can be processed by another replica of the MBSC-SS, since the session status of each user is stored in the DB. This allows managing computing resources very efficiently by employing simple load-balancing and failover techniques, since ASs do not have to maintain session states and can be powered off without any problem.

For what concerns the use of UDP or TCP for exchanging SIP messages, the best solution is to use UDP between the MBS clients and the SIP load balancer, so as to minimize the overhead on the wireless interface, and to use TCP for all other exchanges, that is between the ASN-GW and the SIP load balancer and

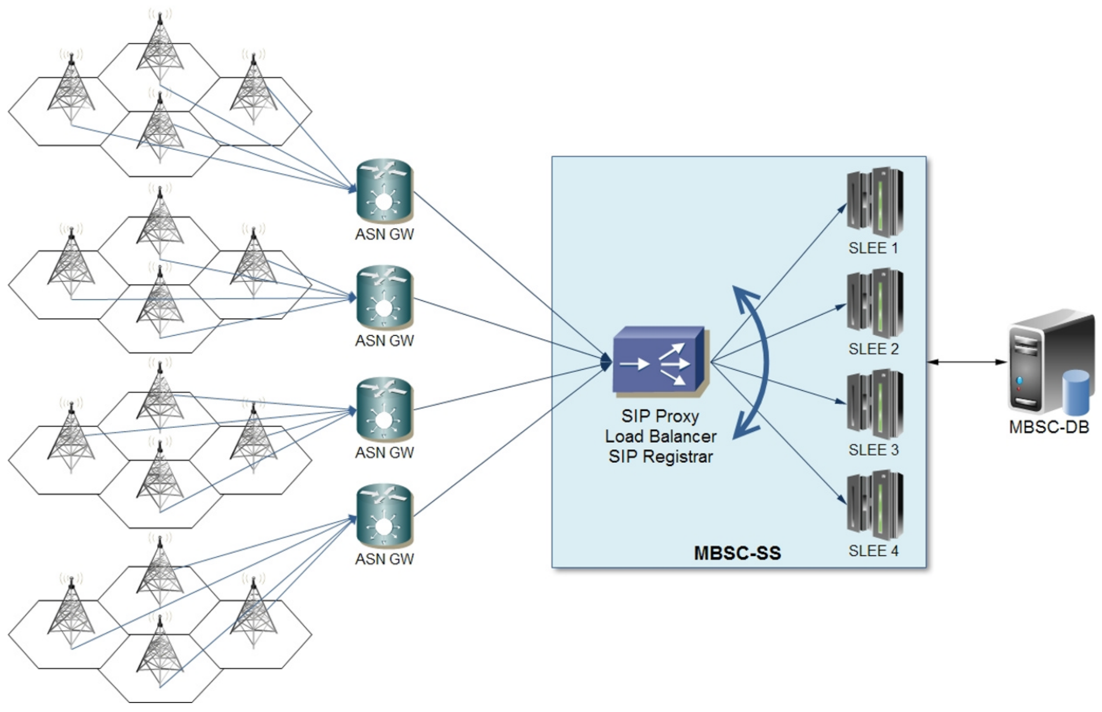

Figure 13 Service deployment in a carrier-grade scenario.

between the SIP load balancer and the MSLEE instances (see Figure 13). In this way, it is possible to maximize the achievable throughput and to control the system latency.

Finally, we point out that the solution proposed, although based on Java, has not been tailored exclusively to the JAIN SLEE service framework. In fact, another very interesting approach to the deployment of our MBSC-SS entails the usage of the Java-based NSF NetServ in network container [44]. By using NetServ, a dynamic approach, in which MBSC-SS replicas can be created and installed closer to the ASN-GW lowering signaling latencies and dynamically adapting the number of servers to the load, is made possible with no significant development effort.

6. Conclusion

In this article, we have illustrated the novel design and implementation of a modular, open source, and IP-based MBS over WiMAX service signaling architecture. The modularity of the proposed architecture allows service providers to select the best technology for each service entity without any need of modifying the other service entities or the signaling interfaces. The presented service architecture is service independent, and allows creating services such as file-casting, media streaming, mass software updates, and data push services without any need of modifying the signaling messages or network entities. Any change of the service type requires only the change of the user GUI of the MBS client and the MBS source content provider technology, both outside the core network elements.

Being full IP-based, our service architecture is independent from the access network technology, thus it is possible to extend it to other fixed or mobile access networks beyond WiMAX.

The signaling messages exchanged between system entities are based on encoded binary data encapsulated within the payload of SIP messages. In the proposed solution, we have chosen to shift implementation complexity from the ASN-GW towards the JSLEE server orchestrating all signaling. Thanks to the open nature of the JSLEE specifications and the Java capability to ease the development process of telecommunication services, our approach allows supporting a number of different vendors/protocols without major changes.

To test the performance of the service architecture, we have focused on the non-trivial case of a multicast IPTV service offered through an MBS over WiMAX architecture. The signaling server has been implemented by using a single *service building block* deployed over the open source Mobicents JSLEE server. Also the other signaling entities have been implemented by using open source technologies, thus reducing costs and leveraging the community of developers. Our results show that a simple virtual machine hosting the MBSC signaling server, which can be regarded as a common mid-level machine in terms of memory size and CPU processing power can provide up to 210 service transaction/s (and thus supporting tens of thousands of users) and a signaling latency as low as 100 ms when the TCP transport protocol is used by the MBSC-SS.

Although the experiments shown in this article and in some referenced articles are quite different, it could be interesting to compare session set-up times reported by Munir and Gordon-Ross [29] (well above 1 s in the best case) and our switching time (below 100 ms by using TCP as transport protocol). This should be enough to show an order of magnitude on the advantage of deploying our system with respect to IMS-based ones, which are much more complex. As for the comparison with other MBS systems, let us consider the one defined for the UMTS illustrated in [45]. Comparing that proposal with our solution, two differences emerge. The first is that the solution proposed in [45] is tailored exclusively to the UMTS system, whereas our proposal is quite general and could be adapted to other systems with minimal software changes in the MBSC-SS service logic, maintaining the same protocols. The second is that, analyzing the complexity of the signaling exchange, our solution is definitely simpler.

A future work goal is the definition of a standard high level interface to be used between the ASN-GW and the MBSC-SS based on the emerging Open WiMAX standard [46]. Using such a standard interface, the JSLEE service logic becomes independent from the specific protocol implementation used in the WiMAX network, and thus portable over different vendor devices.

Endnote

[a]As explained in last part of Section 3.1.2, missing replies to IGMP query refresh messages could cause service interruption, thus IGMP queries relevant to data sessions are much more important than those relevant to announcements.

Competing interests
The authors declare that they have no competing interests.

Received: 20 September 2011 Accepted: 11 April 2012
Published: 11 April 2012

References

1. Mobile WiMAX–Part I: A Technical Overview and Performance Evaluation http://www.wimaxforum.org/technology/downloads/Mobile_WiMAX_Part1_Overview_and_Performance.pdf
2. T Jiang, W Xiang, Multicast broadcast services support in OFDMA-based WiMAX system. IEEE Commun Mag. **45**(8), 78–86 (2007)

3. WiMAX Forum, WiMAX end-to-end network systems architecture, Stage 2: architecture tenets, reference model and reference points http://www.wimaxforum.org/ (August 2006)
4. J Rosenberg, H Schulzrinne, G Camarillo, A Johnston, J Peterson, R Sparks, M Handley, E Schooler, SIP: session initiation protocol. IETF RFC 3261 (June 2002)
5. B Cain, S Deering, I Kouvelas, B Fenner, A Thyagarajan, Internet group management protocol, Version 3. IETF RFC 3376 (October 2002)
6. Mobicents Web Site http://www.mobicents.org
7. JSR 240, JAIN SLEE v1.1. Web Site http://jcp.org/en/jsr/detail?id=240
8. J Boss, Web Site, available at: www.jboss.com
9. K Etemad, L Wang, Multicast and broadcast multimedia services in mobile WiMAX networks. IEEE Commun Mag. **47**(10), 84–91 (2009)
10. 3GPP TS 22.146. Multimedia Broadcast/Multicast Service (MBMS); Stage 1 (Release 9)
11. IEEE 802.16-2009, IEEE Standard for Local and metropolitan area networks, Part 16: Air Interface for Broadband Wireless Access Systems (May 2009)
12. WiMAX Forum, Mobile WiMAX–Part II: A Comparative Analysis http://www.wimaxforum.org/ (May 2006)
13. S Bhatnagar, Multicast data delivery in WiMAX by reusing L2 handoff procedure. Annu Rev Commun. **61**, 277–283 (2008)
14. J Wang, M Venkatachalam, Y Fang, System architecture and cross-layer optimization of video broadcast over WiMAX. IEEE J Sel Areas Commun. **25**(4), 712–723 (2007)
15. JM Lee, H-J Park, SG Choi, JK Choi, Adaptive hybrid transmission mechanism for on-demand mobile IPTV over WiMAX. IEEE Trans Broadcast. **55**(2), 468–477 (2009)
16. JH Lee, TT Kwon, Y Choi, S Pack, Location management area (LMA)-based MBS handover in mobile WiMAX systems, in *IEEE International Conference on Communications System Software and Middleware (COMSWARE) 2008*, Bangalore, India, pp. 341–348 (January 2008)
17. N Liao, Y Shi, J Chen, J Li, Optimized multicast service management in a mobile WiMAX TV system, in *6th IEEE CCNC*, Las Vegas, NV (USA), pp. 1–5. February 2009 (January 2009)
18. C Riede, A Al-Hezmi, T Magedanz, Session and media signaling for IPTV via IMS, in *Mobilware '08*, Innsbruck, Austria, (February 2008)
19. E Mikoczy, D Sivchenko, B Xu, V Rakocevic, IMS based IPTV services–architecture and implementation, in *ACM MobiMedia'07*, Nafpaktos, Greece, (August, 2007)
20. ETSI ES 282 007. Telecommunications and Internet converged services and protocols for advanced networking (TISPAN); IP multimedia subsystem (IMS); functional architecture, V2.0.0, 2008-05 (2008)
21. I Vidal, J Garcia-Reinoso, I Soto, F Valera, Evaluating extensions to IMS session setup for multicast-based many-to-many services. Comput Netw. **55**(3), 600–621 (2011). doi:10.1016/j.comnet.2010.09.011
22. C Riede, O Friederich, R Seeliger, S Arbanowski, Interactive IMS-based IPTV–plunge into the reality game show, in *IEEE IMSAA 2008*, Bangalore, India, pp. 1–6 (December 2008)
23. N Hung, N Thanh, Intelligent features for IMS-based IPTV, in *IEEE ATNAC 2010*, Auckland, New Zealand, pp. 130–134 (November 2010)
24. Y Xiao, X Du, J Zhang, F Hu, S Guizani, Internet protocol television (IPTV): the killer application for the next-generation Internet. IEEE Commun Mag 126–134 (2007)
25. J Kellokoski, E Tukia, E Wallenius, T Hamalainen, J Naarmala, Registration performance comparison between IP multimedia subsystem and session initiation protocol networks, in *International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2010*, Moscow, Russia, pp. 20–25. 18-20 October 2010
26. J Lloret, M Garcia, M Atenas, A Canovas, A QoE management system to improve the IPTV network. Int J Commun Syst. **24**(1), 118–138 (2011). doi:10.1002/dac.1145
27. D Di Sorte, M Femminella, G Reali, A QoS index for IP services to effectively support usage-based charging. IEEE Commun Lett. **8**(11), 686–688 (2004). doi:10.1109/LCOMM.2004.837661
28. B Campbell, J Rosenberg, H Schulzrinne, C Huitema, D Gurle, Session initiation protocol (SIP) extension for instant messaging. IETF RFC 3428 (December 2002)
29. A Munir, A Gordon-Ross, SIP-based IMS signaling analysis for WiMAX-3G interworking architectures. IEEE Trans Mob Comput. **9**(5), 733–750 (2010)
30. M Christensen, K Kimball, F Solensky, Considerations for Internet group management protocol (IGMP) and multicast listener discovery (MLD) snooping switches. IETF RFC 4541 (May 2006)
31. M Femminella, R Francescangeli, F Giacinti, E Maccherani, A Parisi, G Reali, Scalability and performance evaluation of a JAIN SLEE-based platform for VoIP services, ITC-21 Paris, France, (September 2009)
32. Cisco Broadband Wireless Gateway http://www.cisco.com/en/US/prod/collateral/wireless/wirelssw/ps8738/product_data_sheet0900aecd806cdb72.html
33. B Li, Y Qin, CP Low, CL Gwee, A survey on Mobile WiMAX. IEEE Communications Magazine. **45**(12), 70–75 (2007)
34. phpMyAdmin web page http://www.phpmyadmin.net/home_page/index.php
35. JAIN SIP stack Web Site http://jsip.java.net
36. M Handley, C Perkins, E Whelan, Session announcement protocol. IETF RFC 2974 (2000)
37. VMware ESXi Web Site http://www.vmware.com/products/esxi/
38. A Singh, An introduction to virtualization. http://www.kernelthread.com/publications/virtualization/ (2004)
39. SIPp Traffic Generator http://sipp.sourceforge.net/
40. D Pareit, M Deruyck, E Tangle, W Joseph, I Moerman, L Martens, P Demeester, Detailed modeling of MAC throughput and ranges for mobile WiMAX. IEEE Commun Lett. **158**(8), 839–841 (2011)
41. M Femminella, E Maccherani, G Reali, Performance management of java-based SIP application servers, in *IEEE/IFIP IM'11*, Dublin, Ireland, pp. 493–500. (23-27 May 2010)
42. EM Nahum, J Tracey, CP Wright, Evaluating SIP proxy server performance, in *NOSSDAV '07*, Urbana, IL, (2007)
43. H Jiang, A Iyengar, E Nahum, W Segmuller, A Tantawi, CP Wright, Load balancing for SIP server clusters, in *IEEE INFOCOM 2009*, Rio de Janeiro, Brazil, pp. 2286–2294. 19-25 April 2009
44. M Femminella, R Francescangeli, G Reali, J Lee, H Schulzrinne, An enabling platform for autonomic management of the future internet. IEEE Netw. **25**(6), 24–32 (2011)
45. G Xylomenos, V Vogkas, G Thanos, The multimedia broadcast/multicast service. Wirel Commun Mob Comput. **8**(2), 255–265 (2008). doi:10.1002/wcm.463
46. Open WiMAX whitepaper, The disruptive approach of open WiMAX http://www.wimax-industry.com/sp/wcm/avr/dl/wp/Disruptive_Approach_wp.pdf

doi:10.1186/1687-1499-2012-140
Cite this article as: Femminella *et al.*: A signaling architecture for multimedia MBS over WiMAX. *EURASIP Journal on Wireless Communications and Networking* 2012 **2012**:140.