MEDIA HANDLING
FOR CONFERENCING IN MANETs

Dhafer Ben Khedher

A Thesis
in
The Department
of
Electrical and Computer Engineering (ECE)

Presented in Partial Fulfillment of the Requirements
For the Degree of Philosophy Doctor at
Concordia University
Montreal, Quebec, Canada

November, 2007

Your file Votre référence
ISBN: 978-0-494-37749-9
Our file Notre référence
ISBN: 978-0-494-37749-9

ABSTRACT

Media handling for conferencing in MANETs

Dhafer Ben Khedher, Ph. D.
Concordia University, 2007

Mobile Ad hoc NETworks (MANETs) are formed by devices set up temporarily to communicate without using a pre-existing network infrastructure. Devices in these networks are disparate in terms of resource capabilities (e.g. processing power, battery energy). Multihop Cellular Networks (MCNs) incorporate multihop mobile ad-hoc paradigms into 3G conventional single-hop cellular networks. Conferencing, an essential category of applications in MANETs and MCNs, includes popular applications such as audio/video conferencing. It is defined as an interactive multimedia service comprising online exchange of multimedia content among several users. Conferencing requires two sessions: a call signaling session and a media handling session. Call signaling is used to set up, modify, and tear down conference sessions. Media handling deals with aspects such as media transportation, media mixing, and trans-coding. In this thesis, we are concerned with media handling for conferencing in MANETs and MCNs.

We propose an architecture based on two overlay networks: one for mixing and one for control. The first overlay is composed of nodes acting as mixers. Each node in the network has a media connection with one mixer in the first overlay. A novel distributed mixing architecture that minimizes the number of mixers in end-to-end paths is proposed as an architectural solution for this first overlay. A sub-network of nodes, called controllers, composes the second overlay. Each controller controls a set of mixers, and collectively, they

manage and control the two-overlay network. The management and control tasks are assured by a media signaling architecture based on an extended version of Megaco/H.248.

The two-overlay network is self-organizing, and thus automatically assigns users to mixers, controls mixers and controllers, and recovers the network from failures. We propose a novel self-organizing scheme that has three components: self-growing, self-shrinking and self-healing. Self-growing and self-shrinking use novel workload balancing schemes that make decisions to enable and disable mixers and controllers. The workload balancing schemes use resources efficiently by balancing the load among the nodes according to their capabilities. Self-healing detects failed nodes and recovers the network when failures of nodes with responsibilities (mixers and controllers) occur. Detection of failed nodes is based on a novel application-level failure detection architecture.

A novel architecture for media handling in MCNs is proposed. We use mediator concepts to connect the media handling entities of a MANET with the media entities of a 3G cellular network. A media mediator assures signaling and media connectivity between the two networks and acts as a translator of the different media handling protocols.

ACKNOWLEDGMENTS

I would like first and for most to express my sincere acknowledge to my advisors, Dr. R. Dssouli and Dr. R. Glitho, for their kind help, guidance, and support to the success of my work and to my maturity as researcher.

I would also like to extend a special thanks to other members of my examining committee for reviewing my thesis work and providing valuable comments.

Thanks are extended to my dear friends and colleagues from the Telecommunications Service Engineering Research Laboratory for their invaluable discussions, support and experiences we have shared.

I have known wonderful friends in Montreal who become family to me. I will not be able to mention all their names, but would like to recognize their valued friendship.

I would like to express my deep gratitude to my parents to whom I state all my respect and acknowledgement; my success is yours. My high regards are addressed to my parents-in-law; thanks for your encouragement.

I would like to express my gratitude to my wife Ibtissem for her support and all the sacrifices she has made.

Finally, I would like to state my affection to my beloved sons Yassine and Rayen; you are the joy of my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

This chapter presents the main concepts, problem statement and objectives of this dissertation, and lists its contributions: it defines the concepts of mobile ad hoc networks and conferencing applications, which specifies the research area. We describe what problems are addressed by this dissertation, and its objectives. We present the principal contributions, detailing how the identified problems have been addressed. As well, this chapter describes the organization of the thesis.

1.1. Main concepts

Mobile Ad hoc NETworks (MANETs) [1][2][3][4] are now emerging and their use is gaining more and more momentum. They can be defined as networks formed by heterogeneous (in terms of resource capabilities) mobile (usually wireless) devices (nodes), set up to communicate temporarily, with no available fixed infrastructure and no pre-determined organization of available connections. MANETs can be used in a wide range of scenarios in private, public, commercial and military settings, either being connected to a backbone (e.g. the Internet, cellular networks) or standalone (i.e. isolated).

To be resource efficient, standalone MANETs (or simply MANETs) distribute the load among powerful nodes in the network to avoid bottlenecks and the vulnerability of centralized models. On the other hand, integrated MANETs with 3G networks [5][6] such as Multi-hop Cellular Networks (MCNs) may rely on centralized entities located in the infrastructure of 3G

networks. MCNs [7][8][9] are the result of the integration of single-hop 3G Cellular Networks (3GCN) with MANETs. They have recently emerged as a promising research area because of the new business opportunities they enable due to their flexibility and power to cater to the requirements of the next generation cellular systems. MCNs can aid in reducing infrastructure costs and in improving network performance, while reducing the path vulnerability encountered in standalone MANETs.

Conferencing [10][11][12] is an essential application category for MANETs that includes well-known applications such as audio/video conferencing, multimedia gaming and multimedia interactive programs. It allows multiple users to exchange media (audio, video, etc.) in real time. Conferencing usually requires opening two sessions: a call signaling session and a media handling session. Call signaling is used to set up, modify and tear down the conference sessions. Media handling is concerned with the transportation of media streams and the control and management of media mixers and media connections. It consists of four tasks: media transportation (e.g. RTP [13]), media mixing (i.e. use of mixers to combine media streams), media negotiation (e.g. SDP [14]), and content adaptation (e.g. media abstraction and media trans-coding). To achieve these tasks, media handling requires a media signaling protocol to define the exchange of information specifically concerned with the establishment, modification and tearing down of media connections, and with the management and control of mixers.

The rest of this chapter presents the problem statement, objectives, contributions, and organization of the thesis.

1.2. Problem statement and objectives

Today, although much work has been done on conferencing using different kinds of networks such as the Internet, there has been little discussion of conferencing in MANETs, especially in terms of the various media handling aspects. In fact, handling media in MANETs entails many potential difficulties, such as: which node is responsible for enabling and disabling mixers since no node remains in a conference indefinitely; if there are changes in the conference (a new connection or disconnection, for instance), how is the information about capabilities propagated to the mixers; and how to deal with the situation if a mixer does not have sufficient resources to handle all of the media connections.

This thesis aims to design, build and evaluate architectures for media handling for conferencing in MANETs and MCNs. The proposed architecture should be self-organizing to automatically assign users (joiners) to mixers, manage and control mixers, and recover the network from failures, because MANETs are infrastructure-less. It needs to address small, medium and even large MANETs (up to hundreds of nodes). An example of an application scenario is multimedia conferencing in military or rescue operations. It is important to note that at any given time only a subset of the nodes will be actively involved (sending media streams) in a conference while the majority of the attendees merely receive media streams. This makes conferencing quite realistic in MANETs despite a scarcity of resources.

The proposed solution for media handling for conferencing in MANETs, called two-overlay architecture, requires two constituent architectures: a media mixing architecture for the first overlay and a media signaling architecture for the second overlay. Moreover, it requires a self-organizing scheme that is composed of three components: self-growing, self-shrinking and self-healing.

To handle media in MCNs, the proposed solution, called Media Handling Architecture for MCNs (MHAM), connects MANETs with 3G cellular networks to support conferencing. The connection requires the introduction of mediators between the multihop mobile ad hoc networks and the single-hop cellular networks. In the rest of this section, we introduce each of these architectures and schemes.

1.2.1. Media mixing architecture

Media mixing is the core component of a media handling system. The main task of media mixing is to reduce the number of media streams in the network when there are several users in the call. This is accomplished through entities called mixers that combine the input streams into a single output stream. In traditional networks (e.g. 3G networks), a centralized bridge acts as a mixer, to which all of the joiners make a direct media connection. Media mixing also contributes to source synchronization so that streams originating at the same time are played at the same time. Mixing functionalities require nodes with high computation power and network link capacity. They also introduce some transmission delays caused by the mixing operation and adapting of the play-out. A media mixing architecture defines the topology of media connections, the location of mixers in the network, and it describes the nodal aspects (e.g. functional entities, information kept in each node) and how media streams are treated (e.g. mixing operation, stream synchronization).

The general requirements for any media mixing architecture deployed in MANETs are very strict (stricter than for cellular networks, for example). A fully decentralized architecture is required, because MANETs are fully decentralized. Furthermore, the system should scale according to the network in which it is implemented. Reduction of the number of media

streams in the network is essential, since multimedia requires a very large network throughput (i.e. delivered data). The architecture should preserve satisfactory media quality (e.g. by reduction of packet loss) throughout the conference. As well, joiners with limited capabilities or with media formats (e.g. codec) different from those used in the current session should not be discarded.

1.2.2. Media signaling architecture

Media signaling is defined as the exchange of information specifically dealing with the establishment of media connections and the management and control of mixers. The traditional approaches for managing and controlling mixers such as Megaco/H.248 [15][16] are obsolete because they are usually designed to manage and control one centralized mixer in the conference. Managing and controlling mixers involve exchanging messages to enable and disable media mixers, assigning tasks to mixers and recovering when mixers suddenly fail (e.g. battery down, shutdown, out-of-order, out-of-range). Assigning a task to a mixer means assigning a conference user to a mixer for the purpose of media mixing. These functionalities require certain control entities in the network to establish media connections and control the mixers in the network. These entities are called controllers. A media signaling architecture should include such controller entities to manage and control the network.

A media signaling architecture defines the topology of the overlay of controllers and the association of controllers with mixers. It also describes the interaction between controllers to achieve the tasks of management and control of the network. Furthermore, it describes the interaction between internal entities (controllers) and external entities (call signaling entities). A media signaling architecture also includes a media signaling protocol that defines the

communication rules that govern exchanging messages dealing with the management of media connections and the management and control of mixers. The protocol also defines format and order of messages sent and received among entities, and actions taken at messages' transmission and messages' reception.

1.2.3. Self-organizing scheme

In MANETs, no node is permanent, paramount and irreplaceable, which means that, no node controls the conference, and thus all of the nodes are concerned with the automatic growth, shrinkage and healing of the network. Subsequently, nodes should collectively exchange information in order to manage the network and make the necessary decisions without manual or external intervention. Therefore, a self-organizing system is required, one that will indicate to joiners which mixers they should connect to. It will also allow an automatic enabling and disabling of mixers and controllers, without involving an external entity. Furthermore, it will include system decision criteria such as when to enable or disable a mixer or a controller. Self-organizing will also recover the network when nodes that have certain roles in the network (e.g. mixers and controllers) become out-of-range or shutdown. To reach these goals, controllers need to exchange messages and update information about mixers and controllers in the conference. This information will also be used to enable recovery from mixer and controller failures. Self-organizing is made up of three components: self-growing, self-shrinkage and self-healing. A description of these components follows.

1.2.3.1. Self-growing and self-shrinking

By self-growing (or self-shrinking), we mean that mixers and controllers are automatically enabled (or disabled) when the network grows (or shrinks). Moreover, self-growing (or self-

shrinking) is necessary to automatically assign and connect (or disconnect) nodes to (or from) the mixers. These functionalities should be performed while preserving efficient usage of resources in the network. By resource efficient, we mean assigning tasks according to node capabilities. MANETs are composed of disparate devices. Situations such as nodes with mixing (or control) capabilities that are not utilized, and giving powerful nodes and nodes with limited capabilities the same workload should be avoided. The self-growing and self-shrinking schemes should ensure an optimized use of resources and make decisions to preserve this optimization as the conference progresses. They should balance the workload among the mixers and controllers, using node capabilities as criteria. Decisions of growth or shrinkage should preserve the workload equilibrium among mixers and controllers, for each decision made.

1.2.3.2. Self-healing

The particular feature of MANETs is that nodes are highly susceptible to failures. A failed node can mean a node with its battery down, or it is shutdown, out-of-order, out-of-range, etc. Another feature of MANETs is they are particularly vulnerable to message loss caused by frequent out-of-range of nodes, by poor channel quality, and by congestion in some intermediate nodes in the routes. Node failure and message loss usually lead to gaps in the flow and thus poor media quality. Self-healing means automatic recovery from the failures of mixers and controllers. To recover from failures, the system requires a failure detection architecture to detect failed nodes that have certain roles, such as mixers and controllers. A failure detection architecture should minimize unreported failure and false detection, which are dependent on the message loss rate and the frequency of nodes that are out-of range in the network. The

failure detection architecture should also be autonomous of the conferencing application to which it provides its service. The active entities in a failure detection architecture should be self-organizing in order to react to the dynamic environment. They have to recover failed nodes, especially those that play a role in the failure detection architecture.

1.2.4. Media handling architecture for conferencing in MCNs

Multihop Cellular Networks (MCNs) incorporate multihop mobile ad-hoc paradigms into 3G conventional single-hop Cellular Networks (3GCN). They increase the network throughput and decrease the power consumption of mobile devices, which makes conferencing more practical and realistic in such networks. In practice, the integration of two different networks usually requires mediators that can communicate with the two networks being integrated. The mediator concept is practical because it leaves the two networks independent, so that the performance of one network is not affected by the network load of the other. Then, media handling architecture for conferencing in MCNs will be based on Media Mediators (MMs) that will bridge the multihop mobile ad hoc world and the single-hop 3G cellular world for media handling purposes. Three main conference scenarios should be supported by an MCN: 3GCN, MANET and MCN conferences. In 3GCN and MANET conferences all users are in 3GCNs and MANETs, respectively. In an MCN conference, some users are in 3GCN and some are in a MANET. Media handling for conferencing in MCNs should consider each of these scenarios.

1.3. Thesis contributions

The main contributions of this thesis are outlined below:

1. A two-overlay architecture is proposed as a global solution to handle media in conferencing applications in MANETs. The first overlay is composed of mixers, each of which has a media connection with a set of nodes. The second overlay is composed of controllers. Each controller has a control connection with a set of mixers. The two-overlay architecture is composed of two constituent architectural solutions: media mixing and media signaling.

2. A novel media mixing approach called Distributed Mixing Architecture (DMA) is proposed as an architectural solution for the first overlay network. DMA significantly reduces the number of media streams in the network and enables rapid delivery because it uses a flat mixer structure that reduces intermediate nodes in the end-to-end paths. This topology is especially well suited for real-time media transportation because in MANETs, media streams may pass through multi-hops between every two nodes. Moreover, DMA facilitates the synchronization of mixers since they are directly connected. This novel architecture is indeed a promising approach for conferencing in MANETs. It can be applied to conferencing in general, and specifically over the Internet.

3. To manage and control the network, a media-signaling architecture is set forth, which permits messages dealing with the management and control of media connections and mixers to be exchanged. This architecture is fully decentralized and scalable in terms of the number of conference joiners. Based on Megaco/H.248, an approach for controlling media mixers in traditional networks, this architecture separates the media signaling done by the Media Gateway Controller (MGC) (the controller) from the mixing done by the Media Gateway (MG) (the mixer). It also separates the Call Signaling Entity (CSE) and the

MGC entity. The CSE makes contact with an MGC using a discovery mechanism. The communication between the CSE and an MGC is defined by a set of primitives.

4. A novel self-organizing scheme is proposed to automatically manage and control the network, with a set of decision criteria that ensures a resource-efficient practice. These aspects are based on novel workload-balancing schemes that equilibrate the workload among the mixers and the controllers. This self-organizing scheme also incorporates a failure detection architecture to detect failed nodes and to recover the network after the failure of nodes that have a control or mixing role in the network. We have proposed a novel overlay-based failure detection architecture that is decentralized, with its own self-organizing scheme, and that provides a probabilistic detection error dependent on message loss probability in the network. This architecture is independent of any lower-layer entity and of the two-overlay media handling architecture. It can be used for detecting failed nodes in any overlay network for application programs.

5. A novel architecture for media handling for conferencing in MCNs is proposed, based on Media Mediators (MMs) that bridge the multihop mobile ad hoc world and the single-hop 3G cellular world. MMs are composed of two functional entities: the Media Gateway Controller Mediator (MGCM) and the Media Gateway Mediator (MGM). The MGCM acts as a controller mediator between a 3GCN and a MANET. It assures connectivity between the networks and acts as a translator of different deployed protocols. The MGCM can be deployed on the 3GCN side, on the MANET side, or in a third party, and it is enabled only when the conference includes users in a 3GCN and a MANET.

1.4. Thesis organization

Chapter 2 presents the basic background of this thesis including MANETs, MCNs, multimedia conferencing, call signaling protocols and media handling components.

Chapter 3 identifies a series of criteria important for any media handling architecture that aims to support conferencing in MANET and MCN environments. Reviews of related work in mixing architectures, media signaling architectures, self-organizing and load balancing schemes, failure detection architectures, and media handling architectures for MCNs in terms of these criteria are then presented.

The two-overlay architecture and the media mixing architecture are presented in Chapter 4. The chapter begins with a description of the two overlays: the first is for mixing and the second is for control purposes -- to control mixers and manage the network. A novel media mixing architecture (called Distributed Mixing architecture -- DMA) is introduced as an architectural solution for the first overlay. The chapter continues with the two-step mixing operation and the synchronization algorithm, along with the mixer core functional architecture.

Charter 5 presents a media signaling architecture based on Megaco/H.248 concepts. It describes the extension made on the protocol and the modifications of commands. This chapter presents the primitives exchanged between the media signaling system and the call signaling system. Furthermore, it presents the events in the network, the semantics of the exchanged messages and the procedures for each event.

Chapter 6 describes the self-organizing mechanism with its three components: self-growing, self-shrinking and self-healing. Self-growing and self-shrinking are based on workload

balancing schemes that use resources efficiently by distributing the load according to node capabilities. Self-healing is composed of both a failure detection architecture and a recovery scheme to react to node failure.

Chapter 7 presents a media handling architecture for MCNs (MHAM) that uses the mediator concept to integrate the different components and to make conferencing possible. This chapter includes the different scenarios of conference initiation, invitation and leaving.

Chapter 8 shows how each component of the two-overlay and MHAM architectures are implemented and evaluated. The chapter begins with a description of the DMA proof-of-concept prototype. The prototype experiments and results, for both small and medium scale, are described next, followed by simulation experiments with a detailed discussion of the results. The chapter then describes a simulation of the media signaling protocol to demonstrate the scalability of the network, followed by a simulation of self-growing and self-shrinking schemes, which shows the ability of the system to grow and shrink when nodes join and leave, and how well the network workload is balanced. A presentation of the failure detection simulation results follows, along with a detailed discussion. The MHAM proof-of-concept prototype implementation environment and test bed are then described. This introduces the media mediator architecture used in the MHAM prototype. Experiment descriptions and results are also included.

Chapter 9 concludes the work presented in this thesis, presents the main contributions and discusses possible future work.

CHAPTER 2

BACKGROUND

Mobile ad hoc networks (MANETs) are an active research area. A significant amount of work has been done on MANET lower layers, but much less work has been done with session and application layers. This chapter is organized as follows. Section 2.1 introduces MANETs. It describes standalone MANETs and MCNs. Section 2.2 describes multimedia conferencing and focuses on the media handling components. Finally, section 2.3 summarizes the chapter.

2.1. Mobile Ad hoc NETworks (MANETs)

A MANET can be defined as a network of wireless mobile disparate terminals (nodes) that is created on demand in order to complete an assigned mission, but that has no available fixed infrastructure [1][2][3][4]. For example, in a battlefield, deployed soldiers, tanks and aircraft fighters must coordinate to perform battle actions either connected to a backbone or operating standalone. MANETs are very useful in the aftermath of a disaster. For example, a temporary MANET can be set up to support the rescue operations after an earthquake. MANETs may be self-configurable and independent of existing infrastructures (standalone), or they may be connected to a pre-existing network infrastructure (e.g. integrated 3G/MANETs). In the rest of this section, we present the different aspects of standalone MANETs and then we describe MCNs.

2.1.1. Standalone MANETs

MANETs that are not connected to a backbone (e.g. the Internet) are called standalone MANETs or simply MANETs. Standalone MANETs operate separately and are usually private. Nodes in standalone MANETs (wireless phones, laptops, palmtops, etc.) are very heterogeneous in terms of resource capabilities (e.g. processing power, memory capacity, storage capacity, device energy, network link capacity) -- some may have a high level of available resources while others may have very limited capabilities. Since capabilities vary considerably between nodes in standalone MANETs, some nodes may contribute more than others in the network. The main feature of standalone MANETs is they can be made resource efficient networks, in which powerful nodes may serve more than less powerful nodes. In the next subsections, we present an overview of MANET lower layers followed by an overview of upper layers, and a description of potential MANET applications.

2.1.1.1. Lower layers

MANETs have evolved with the emergence of broadband wireless technologies, such as IEEE 802.16, IEEE 802.16a and IEEE 802.11a/g. The physical layer in MANETs uses either direct sequence spread spectrum, frequency-hopping spread spectrum, or infrared pulse position modulation to transmit data between nodes. MAC protocols for MANETs are based either on the random access paradigm, such as the Carrier Sense Multiple Access With Collision Avoidance (CSMA/CA) [17], or on controlled access schemes such as the Time Spread Multiple Access (TSMA) [18].

MANETs are multihop networks in which mobile devices set up temporary connectivity communication. The structure of MANETs changes frequently because nodes are capable of

movement. Nodes may serve as mobile routers in the network. They are maintained by different routing protocols such as the Ad Hoc On Demand Vector (AODV) [19] and the Weighted Clustering Algorithm (WCA) [20]. These protocols have their own failure detection mechanisms. For example, in AODV, a link failure between neighbors is detected by three unanswered 'Hello' messages. AODV does not provide periodic updates of information about node failure, and consequently, node failure is only reported when a path is requested.

Routing protocols can be classified as either proactive, where routing tables are proactively maintained, or reactive, where routes are discovered on-demand. Reactive protocols, such as AODV [19], have lower overhead in small and medium networks but the overhead can grow quickly for large networks. To make MANETs more scalable, a number of approaches have been proposed such as the landmark-based technique [21], in which a hierarchy of self-organized landmark nodes maintain a landmark table. Other approaches have been proposed such as the anchor-based technique [22] that uses dedicated fixed anchors for routing. Theses approaches can be used in the design of overlay routing architectures.

2.1.1.2. Upper layers

Nodes in MANETs are highly susceptible to failures. A node failure can be the result of a battery down, an operating system shutdown, an out-of-order, or an out-of-range. Node failure and message loss in the network usually leads to unpredictable behavior of applications, which should therefore be made to be fault-tolerant. Transport protocols such as TCP were originally designed to work in fixed networks (e.g. the Internet), in which node failure and packet loss rates are quite low. Numerous solutions and evaluations have been proposed over the past few years to improve TCP performance over MANETs (e.g. [23][24]).

While the early MANET applications and deployments were military oriented, commercial applications have grown substantially. Many applications in MANETs are very promising, such as multimedia gaming, multimedia interactive programs, streaming and web services. MANETs can be deployed in a number of commercial, academic or military application opportunities categorized as follows:

- Multimedia conferencing applications (e.g. audio/video conferencing, multimedia gaming and distance learning);
- Emergency services in case of disasters (e.g. ambulance, police and firefighters);
- Entertainment and services (e.g. live media streaming, file sharing and web services);
- Embedded computing applications (e.g. wireless sensor networks);
- Military applications (e.g. battlefield actions, coordination and communication).

2.1.2. Multihop Cellular Networks (MCNs)

This subsection presents an overview of the integration of 3G networks with MANETs in general, followed by the presentation of MCNs.

2.1.2.1. Integrated 3G/MANETs

Third generation (3G) Cellular Networks (3GCNs) are being standardized by third Generation Partnership Projects (3GPP/3GPP2) [25]. Standardization of fourth-generation (4G) wireless systems [26], which are made up of both legacy and new networks coexisting and cooperating, has not yet begun. 3G networks are the primary legacy networks, as they continue to be deployed. They are organized in an infrastructure-based mode. 3G Cellular Networks

(3GCNs) have emerged in the last decade. They have evolved from earlier generations (1G and 2G), which provide simple voice services, to a new generation of networks (3G) that provide new integrated data applications and services.

The integrated 3G/MANETs are the result of the integration of 3G networks with MANETs, in which they coexist and cooperate to provide new services and applications. A practical example of integrated 3G/MANETs is Multihop Cellular Networks (MCNs).

2.1.2.2. MCNs

MCNs [7][8][9] are the result of the integration of single-hop 3GCNs with multihop MANETs. MCNs can aid in reducing the cost of infrastructure and in improving performance, while reducing path vulnerability encountered in MANETs. For example, to cover a densely populated metropolitan area to support more users or to increase the throughput, a 3GCN requires a high number of base stations. An MCN reduces the high cost for building a large number of base stations, increases the total throughput in the area [8], and requires less transmission power, which decreases the total power consumption [9]. MCNs are expected to be flexible and powerful, and to support richer and more diverse multimedia services such as multimedia conferencing with higher stream rates. Applications and services in MCNs should benefit from the ubiquitous environment to access information from everywhere, the throughput increase of the network, the area coverage/enlargement of base stations, the network scalability, the efficient use of resources, and the power consumption decrease of mobile devices. The deployment of multimedia applications and services, such as multimedia conferencing, on MCNs has not been thoroughly investigated. Since these applications are usually centralized on 3G networks, their implementation on these networks is a very

challenging task. The few issues tackled at the application-layer are limited to user authentication, service discovery and charging.

Until now, research attention has been focused on lower-layer integration issues such as routing protocols and adaptation of wireless interfaces and networking. Multiple interfaces, such as WiFi/WiMAX and MAC layer interfaces, have been actively explored to provide ad hoc communication capability for MCNs. Examples that illustrate how the integration of 3G networks with MANETs is concretely done at the lower layers includes coverage extension of the wireless cells to increase the throughput (UCAN [27]) and load balancing among the wireless cells (iCAR [28]). UCAN and iCAR assume that mobile devices have two interfaces: one to MANETs and the other to 3G cellular networks. Therefore, MANETs and 3G cellular networks are tightly connected.

2.2. Multimedia conferencing

Multimedia conferencing [10][11][12] (also known as multimedia multiparty sessions) is one of the most challenging category of applications in MANETs and MCNs. It is the basis of a score of applications including audio/video conferences, multimedia gaming and interactive multimedia programs. It is defined as an interactive multimedia service, and it provides the online exchange of multimedia content between several users. In the rest of this section, we present the media quality factors, the different categories of conferencing applications, the conference models and the conferencing application components.

2.2.1. Media quality factors

Conferencing applications are real-time. Media streams are transmitted over a network in a series of packets. In order for the receivers to reproduce the original media streams, they need to receive all the packets along with the timing relationships among them. However, networks such as MANETs do not guarantee when or whether a packet will be delivered to the receivers. Buffering of the packets in the intermediate nodes introduces variable delays and can distort the original timing relationship among the packets. In addition to the delay, jitter that results from the variable buffering times when the buffer limit is exceeded can further impair the media quality caused by packet loss. MANET conferencing is also particularly vulnerable to gaps in the flow caused by frequent out-of-range of nodes, by poor channel quality, and by congestion in some intermediate nodes in the routes. Packet loss, latency (end-to-end delay), jitter, and gaps in the flow constitute the factors that define the media quality. These factors affect the intelligibility of media content (e.g. speech).

2.2.2. Conference categories

There are several schemes for classifying conferences: dial-out and dial-in, prearranged vs. ad hoc, private or public, with or without sub-conferences, and with or without floor control.

Users can attend a conference in two ways: dial-out mode, in which users join when invited by a user who is already in the conference, and dial-in mode, in which joiners call an authority entity to join the conference.

Another scheme is whether the conference is prearranged or ad hoc. A prearranged conference starts at a predetermined time and is sponsored by specific users. The duration of the conference may also be predefined. An ad hoc conference, on the other hand, starts

without any prior booking or reservation when users initiate the conference. Specifically, it starts when the first two users decide to create a session. Users may join by invitation and leave spontaneously during the conference. The conference ends when the last two parties leave.

Conferences can also be categorized by access. They can be private (closed) or public (open). A closed or private conference is not welcome to all users. An open or public conference, on the other hand, publishes its information to all users in a network (e.g. the Internet).

A conference may have sub-conferences that simulate rooms in the real world. In each room, users can hear/see each other, but they cannot see/hear others that are in different sub-conferences. Finally, a conference can be with or without floor control. Floor control is the technology that deals with conflicts in shared work spaces [29].

2.2.3. Conference models

A conference model describes the topology used for signaling and media handling sessions in a conference. Reference [30] discusses four main topologies: full-mesh, loosely coupled, dial-in bridge and ad-hoc bridge. In full-mesh topology, every node has a signaling relationship with every other user in the network. In this model, media streams are distributed by multi-unicast or multicast mechanism. Each end-system does its own media mixing. This model is specific for small conferences. The loosely coupled model is for very large conferences in which there is no relationship between every two nodes. Media is distributed by a multicast mechanism in this model. Every node has to execute a multicast program to route the media streams. Multicast programs can be deployed in the network layer or in the application layer. The third model is the dial-in bridge (centralized model) in which the bridge acts as a centralized control for signaling and media handling. Finally, the ad-hoc bridge is a hybrid model in which a

conference starts as a full mesh model and switches to a dial-in bridge model when the conference becomes unwieldy. Obviously, there are many other models that can be obtained by a combination of the previous models.

A recent classification presented in the IETF RFC [31] proposes three models: fully distributed conferences (full mesh model), loosely coupled conferences (use multicasting), and tightly coupled conferences (centralized model). Full mesh and loosely coupled models have the same description as the models described in the former classification. In centralized conferences, a conference bridge is defined to do the mixing for all of the end systems. Each end system has a direct signaling connection with the bridge. In this model, a user may either call the bridge to join a conference (dial-in) or be called by the bridge to join (dial-out). This centralized model (tightly coupled) is used for conferencing in 3G standards [10]. 3G conferences can be with/without floor control, private/public, ad hoc/prearranged, and with/without sub-conferences. However, conferencing in MANETs cannot be centralized because it is not realistic to assume the existence of a stable conference bridge in such transient and dynamic network environments. On the other hand, like 3G conferences, MANET conferences can be with/without floor control, private/public, ad hoc/prearranged, and with/without sub-conferences.

2.2.4. Conference application components

Usually, multimedia conferencing consists of three components: call signaling, media handling and floor control (Figure 2.1). Call signaling is used to set up, maintain, modify and end the conference sessions. Media handling deals with the transportation of media streams and the management and control of media connections and media mixers. Floor control is concerned

with the management of access rights to shared resources (e.g. video channels). It coordinates the concurrent usage of shared media resources among conference users.

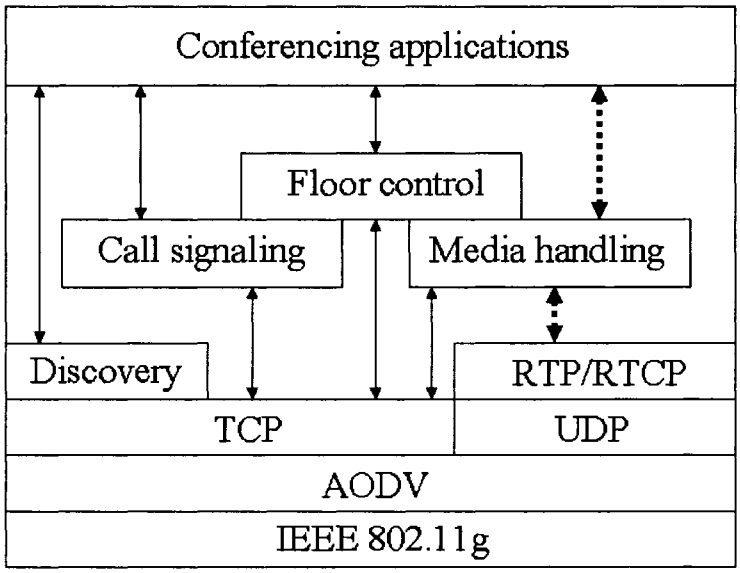

Figure 2.1. Layered architecture for conferencing in MANETs

To establish communication between many users, the conferencing application requires a call signaling session and a media handling session. In the rest of this subsection, we present call signaling and the different tasks of media handling.

2.2.4.1. Call signaling

In general, signaling is defined by the International Telecommunications Union (ITU) as the exchange of information specifically concerned with the establishment and control of connections, and with management of the different entities, in a telecommunication network (ITU-T I.112 [32]). Commonly, a signaling session in a conferencing system consists of two sub sessions. The first, call signaling, includes call setup and call control. The second, called media signaling and part of media handling, deals with establishment and control of media

connections, management and control of media mixers and control of the flow. Distinguishing between the two sub sessions is a delicate task. Thus, the majority of the existing signaling protocols for conferencing combine the two sub sessions when establishing media connections or connecting nodes to mixers. There are many call signaling protocols in the industry for conferencing, usually used on the Internet. The Session Initiation Protocol (SIP) [33][34][35], developed by Internet Engineering Task Force (IETF) [36], is a call signaling protocol for Internet conferencing, telephony, presence, events notification and instant messaging. H.323 [37][38], developed by ITU, allows real-time media exchange and data communication over packet-switched networks such as IP networks. H.323 is designed with multipoint voice and video conferencing capabilities. Another call signaling protocol, called ICEBERG [39][40], developed by Berkeley University, is an open Internet service architecture that enables simple redirection of flows combined with pipelined transformations.

2.2.4.2. Media handling

The media handling session is concerned with the transportation of media streams and the control and management of media connections and media mixers. It consists of five main tasks: media transportation, media mixing, media signaling, media negotiation, and content adaptation.

a. Media transportation

Media transportation deals with the end-to-end real-time delivery of traffic. In fact, real-time traffic is not re-transmittable. The Real-time Transport Protocol (RTP) [13] provides end-to-end network transport functions suitable for applications transmitting real-time data, such as

audio, video or simulation data, over multicast or unicast network services. RTP does not address resource reservation and does not guarantee quality-of-service for real-time services.

b. Media mixing

Media mixing is a critical function for conferencing [41]. Mixing involves receiving media streams from multiple sources, combining them into a single stream, and sending out the mixed result. This operation should send mixed results without returning streams to their sources. Media mixing is carried out by entities called mixers. In conversational conferencing, mixing is necessary to synchronize between the media streams being played. Furthermore, mixing significantly decreases the number of streams in the network. Mixers can also control which sources of streams are passed through and which ones are muted. Nevertheless, the mixing operation requires nodes with high computation power and network link capacity. It also introduces some transmission delays.

c. Media signaling

New conferencing systems, such as Megaco/H.248 [15][16], separate the media signaling done by the Media Gateway Controller entities from the media mixing done by the Media Gateway entities. Media signaling describes the communication rules behind exchanging messages about media connections and mixers' management and control. It describes the interaction of a set of functional entities and a media signaling protocol. The media signaling protocol defines format, order of messages sent and received among entities, and actions taken on message transmission, or on message reception.

d. Media negotiation

Media negotiation supports both session description and capability negotiation. The most widely used protocol for media description is the Session Description Protocol (SDP) [14], which describes multimedia sessions for the purpose of session announcement, session invitation, and other forms of multimedia session initiation.

e. Content adaptation

Content adaptation includes all media transformation necessary to allow certain users to join the conference such as media abstraction [42][43] and media trans-coding [44][45]. For example, MPEG-7 [42] has standardized tools for describing different aspects of multimedia at different levels of abstraction. MPEG-7 includes tools that describe visual and audio features (e.g. color) and user preferences of multimedia. These tools can help in the joining process by selecting the suitable multimedia attributes for the joiner. MPEG-7 does not deal with trans-coding, which is the actual encoding of moving pictures and audio, as MPEG-1, MPEG-2 and MPEG-4 do. Usually, mixers are concerned with content adaptation. Mixers can act as a relay for low-speed link users in a high-speed conference instead of forcing everyone to use a lower-bandwidth or reduced-quality audio trans-coding.

2.3. Summary

This chapter has presented MANETs and MCNs, followed by background on multimedia conferencing including description of media quality factors, conference categories, conference models and conference application components: call signaling and media handling. The

chapter continues with the different tasks of media handling component: media transportation, media mixing, media signaling, media negotiation and content adaptation.

CHAPTER 3

RELATED WORK

The specific characteristics of MANETs complicate the design of a media handling architecture (the two-overlay architecture) because it must be scalable, resource efficient and self-organizing. The design of a Media Handling Architecture for MCNs (MHAM) is also a very challenging task since there is no way to interface MANETs with 3G Cellular Networks (3GCNs). Even though no work has explicitly addressed these issues, there are a wide range of relevant approaches that may be useful in building each of the two-overlay and MHAM architecture components: media mixing architecture, media signaling architecture, self-organizing scheme – composed of self-growing and self-shrinking based on workload balancing schemes, and self-healing -- composed of failure detection architecture and a recovery scheme, and media handling architecture for conferencing in MCNs.

We first introduce, define and motivate a set of criteria used for reviewing existing work. Next, we move to a detailed description and discussion of the most relevant approaches according to this set of criteria. This chapter is organized as follows. The first section presents the criteria that include a set of overall requirements for the two-overlay architecture, requirements specific to each of the components of the two-overlay and MHAM architectures, and then describes the assumptions made for the whole system. The second section describes the existing media mixing approaches and discusses them according to the requirements specific to media mixing architectures. The third section covers media signaling architectures. The fourth section discusses self-organizing schemes and presents workload balancing schemes and

failure detection architectures. The fifth section presents media handling architectures for conferencing in MCNs and discusses them according to the corresponding set of requirements. The sixth section gives a short summary of the chapter.

3.1. Criteria

The peculiarities of MANETs and MCNs put very challenging requirements and assumptions on media handling architectures for conferencing. In the rest of this section, we introduce the overall requirements and the requirements for each of the two-overlay architecture components, followed by the requirements specific to media handling architectures for MCNs. Finally, we present the assumptions made on the whole system.

3.1.1. Requirements

The following set of requirements has been determined as most useful for reviewing related existing approaches to design the two-overlay architecture components and media handling architecture for MCNs, and to evaluate the work presented in this thesis.

3.1.1.1. Overall requirements specific to the two-overlay architecture

The overall requirements are applicable for each of the two-overlay architecture components. The first requirement is that none of the functional entities of the two-overlay architecture should be centralized because MANETs are infrastructure-less and fully decentralized. Thus, to dynamically control and manage mixer and control entities, fully decentralized schemes are required, such as decentralized information management and decentralized decision management. The second requirement is scalability. The system should scale according to the network in which it is implemented. Precisely, the two-overlay architecture should be scalable

in terms of the number of conference users (the network may reach hundreds of users in some applications).

3.1.1.2. Requirements specific to media mixing architectures

Since multimedia requires a very large network bandwidth, the reduction of the number of media streams in the network is a first requirement. Because conferences are in real-time, rapid delivery of media streams is a second requirement. Synchronization is very important in MANETs. It permits end-to-end delivery with proper ordering and timing. Finally, the system should allow participation with different media formats (e.g. codec).

3.1.1.3. Requirements specific to media signaling architectures

To allow scalability, a media signaling architecture should separate the signaling done by the Media Gateway Controller (MGC) from the media trans-coding and mixing done by the Media Gateway (MG). Furthermore, a media signaling architecture should separate the Call Signaling Entity (CSE) and the MGC entity to keep both independent.

3.1.1.4. Requirements specific to self-organizing

The requirements specific to each of the self-organizing components are described in the following subsections.

3.1.1.4.1. Requirements specific to self-growing and self-shrinking

We set three main requirements for self-growing and self-shrinking components that are applicable for self-organizing systems in general. Self-growing and self-shrinking schemes should preserve efficient resource usage, especially control and mixing resources, and be of

low complexity. Furthermore, they have to preserve stability during the conference progress in terms of the number of mixers and controllers in the network.

3.1.1.4.2. Requirements specific to self-healing

Self-healing is ensured by a failure detection architecture and a recovery scheme. The requirements specific to failure detection architectures are presented in the following subsection, followed by the requirements specific to recovery schemes.

a. Requirements specific to failure detection architectures

With respect to layer design architecture, an application-level failure-detection entity should be independent of any lower-layer entity. In addition, the failure detection entity should be autonomous of the application with which it provides its service. Furthermore, when designing a failure detection architecture for MANETs, the designer should give more importance to certain requirements, such as rapid failure detection and low bandwidth usage. These last two properties are inversely related. Thus, if the application requires rapid failure detection then the cost is 'paid' with the bandwidth and vice-versa. In our design, bandwidth usage is a significant concern, while the application is allowed to control the rate of failure detection.

Completeness and accuracy are the key requirements related to the reliability of failure detection architectures. Completeness indicates the report of the failure to non-failed nodes (operational nodes). Accuracy implies no false detection, so that a non-failed node will not be stated as failed by any non-failed node. The work done in [46] demonstrates that it is impossible to design a failure detection system that is both complete and accurate in an asynchronous network. Moreover, message loss is frequent in MANETs, which makes it

difficult to provide deterministic guarantees for completeness and accuracy. Consequently, unreported failure and false detection are allowed probabilistically, depending on the message loss probability in the network. We give more importance in our requirements to the accuracy feature to counter the frequent out-of-range in MANETs that is the main source of false detections.

b. Requirements specific to recovery schemes

Since the recovery scheme should be designed specifically for our system, the only requirement is that this scheme should be of low complexity.

3.1.1.5. Requirements specific to media handling architectures for MCNs

An MCN supports the three conference scenarios: 3GCN, MANET and MCN, because all the users could be in a 3GCN, or in a MANET, or some users could be in a 3GCN and others in a MANET. As a first requirement, it should be possible to handle media when all the users are in a 3GCN, when they are all in a MANET, and when some are in 3GCNs and others are in MANETs. The second requirement deals with performance. Adding more users in one of the two networks (3GCN or MANET) should not impact the performance of the other network. For example, if more users are added to the 3GCN, the performance in the MANET should not degrade. The next requirement concerns the media handling architecture itself. This architecture should be general and flexible enough to allow the integration of several 3GCNs and MANETs. Moreover, it should be independent of the architectures and protocols used in MANETs and 3GCNs.

3.1.2. Assumptions

In this thesis, we use dial-out ad-hoc conferencing as a mode because it is a very natural scheme in MANETs. A dial-out ad-hoc conference begins with two users, and any user can join when invited by a user who is already in the conference. Users can leave whenever they wish and the conference ends when the two last users leave. The conference system does not use any predictive information (e.g. arrival rate, node capabilities, conferencing duration, departure rate).

We assume that there are always sufficient mixing and control capabilities in the network. In addition, we assume that there are three kinds of nodes in the network:

1) **Nodes with no special capability.** This kind of node, called simple node, can only do conferencing by exchanging media streams. These nodes cannot mix and they cannot control.

2) **Nodes with mixing capability.** This kind of node can be enabled to act as a mixer or just act as a simple node (disabled).

3) **Nodes with control capability.** These nodes can be enabled to act as control entities or they can be disabled.

We note that a node can have both mixing and control capabilities at the same time and thus can be both a mixer and a controller simultaneously.

Furthermore, we assume that reorganization of the network during the conference is not suitable. Disconnecting and re-connecting nodes to mixers is time consuming and user disturbing. It also increases the workload of the mixers because they need to set up extra

media connections. Therefore, it is impractical to re-organize the structure of the network by disconnecting nodes from their given mixers and reconnecting them to other mixers.

Finally, conferencing applications cannot rely on routing protocols especially to detect failed nodes. Routing protocols are based on detection of the physical wireless link status to recover routes, so they do not focus on the node status.

3.2. Media mixing architectures

A media mixing architecture defines mixers' locations in the network and describes media connections between different nodes. It also describes how the mixing operation is done. Media mixing consists of receiving media streams from multiple sources, combining them into a single stream and sending the mixed result. This operation should send the mixed results without returning streams to their sources. Mixing reduces the number of streams and contributes in the source synchronization necessary to deliver streams with the correct ordering and timing. On the other hand, mixing demands powerful nodes and causes some delays in stream transmission. There are three principal categories of mixing architectures: centralized mixing, end-point mixing and distributed mixing. We begin by reviewing centralized and end-point mixing architectures. We then review three types of distributed mixing architectures addressed in the literature: hierarchical mixing, distributed partial mixing and distributed video mixing.

3.2.1. Centralized architecture

The centralized mixing approach [47][48] is the architecture most often employed, especially on the Internet. It consists of a centralized mixer that is used by all users (Figure 3.1). Each

user has a media connection with this mixer. The process of mixing is done as follows: the mixer mixes the set of input streams and distributes the results (a set of mixed streams), so that users do not receive their own streams. The first mixed stream is distributed to the set of users that are not currently participating in the mix; each of the other mixed streams is distributed separately to a particular user that is currently contributing to the mix (every contributing user will receive an individualized stream); each of the mixed streams consists of the complete mix minus the input stream of the user to which the mix is distributed. Examples of systems using a centralized mixing architecture follow.

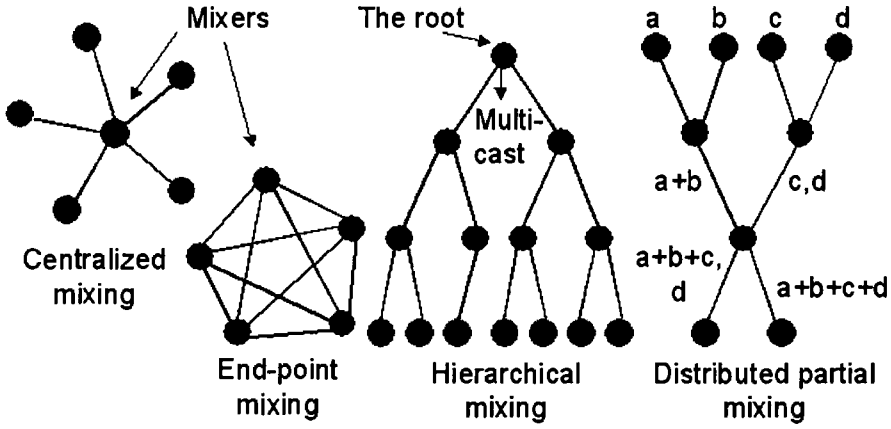

Figure 3.1. Existing mixing architectures

A conferencing system is described in reference [47], which enables heterogeneous multimedia clients to join the same real-time sessions. In this work, a centralized audio mixing architecture is designed to accept different multimedia clients (e.g. SIP and H.323) with many media format. Another work, described in reference [48], describes an algorithm for media mixing, where the centralized mixing approach is used to design a distributed hierarchical mixing architecture. This media-mixing algorithm considerably reduces the streams in the network.

There are some well known conferencing systems that use centralized architecture such as Skype [49] and GlobalMMCS [50]. Skype was trying to address the problems of legacy VoIP (Voice over IP) solutions by improving sound quality, achieving firewall using peer-to-peer (P2P) overlay rather than expensive, centralized infrastructure. It can only support audio conferencing and has no video service. Skype uses a centralized bridge as a mixer for multiparty conferences. It only allows five people in a conference at a time. The collaboration environment called GlobalMMCS [50] is built around the Naradabrokering [51] messaging system to provide a conferencing service. GlobalMMCS uses multiple centralized mixers as the number of topics. Users receive the mixed streams by subscribing to the mixed stream topics.

Finally, the 3GPP [25] media handling architecture uses the centralized mixing approach to provide conferencing services. This architecture will be described later when discussing media handling architectures for conferencing in MCNs.

The centralized mixing architecture does not meet the requirement of de-centralization, making it unsuitable for conferencing in MANETs. The centralized mixer presents a potential bottleneck in the network. It should have enough computation power and bandwidth to serve all nodes in the network. Obviously, it is very difficult to continuously find a node with such strength in a MANET. In addition, this architecture is not scalable since the centralized mixer cannot serve all nodes when the conference becomes very large. However, this architecture considerably reduces media streams in the network. Furthermore, it preserves a low time delivery of streams since there is just one mixer in the end-to-end paths. Moreover, the centralized mixer guarantees synchronization by making it possible to deliver streams with proper timing and ordering. Finally, resources are not used optimally since some of the users

may have mixing capabilities that are not utilized. In addition, self-organizing is not possible because additional mixers will have to be added and configured manually.

3.2.2. End-point architecture

This architecture, in which each user does its own mixing, is easy to implement [52][53] (Figure 3.1). Each mixer receives media streams from other mixers, then mixes and plays the result. Media streams are delivered from source to destination either directly (fully-meshed structure) or by a multicast mechanism. The main relevant work is SPLINE [53] (Scalable Platform for Large Interactive Network Environment). The SPLINE system allows multiple users to access a graphical multi-user virtual world. Users and a set of specialized servers compose the end-point network. Specialized servers are included in a session to support users with slow network connections. They combine audio streams before sending them directly to their supported users. A multicast mechanism is used to exchange audio streams between specialized servers and users with sufficiently fast network connections. A centralized server, the session manager, handles the connection of new users and their eventual disconnection.

The end-point architecture is flexible but it incurs duplication of mixing computation because each node mixes the same input streams. Furthermore, to attend the conference, each user should support the media format used in the current session. Unfortunately, not every node may have mixing capability due to the computation limitations of certain nodes in MANETs. The end-point architecture meets the first requirement of de-centralization. However, MANETs are composed of heterogeneous nodes. Some nodes may not have the capability to do their own mixing, and therefore this architecture is not scalable since nodes with limited

capabilities will not be able to join. Moreover, this architecture produces a significant number of media streams, and so it overloads the network.

On the other hand, streams are delivered rapidly since there are no intermediate mixers in the end-to-end paths. Furthermore, synchronization is easy to implement because there is direct communication among mixers. Unfortunately, this architecture does not use resources efficiently since the most powerful nodes and nodes with limited capabilities have the same workload. However, self-organizing is easily implemented because each new user does its own mixing and connects to each of the nodes in the network.

3.2.3. Hierarchical mixing

A hierarchical mixing architecture using clustering was introduced [48] to support applications that involve large numbers of simultaneously active audio streams. Clusters are composed of closely situated users. Mixers are connected in a hierarchical structure (Figure 3.1). In this mixing hierarchy, mixers constitute internal nodes and users are the leaves. Mixers mix media streams received from their 'children' and send the mixed result to their 'parents'. The only exception is the root mixer, which forwards the final mixed stream to all of the users. Since the mixed packet is common to all users, the root mixer can forward the result by multicasting. Upon receipt of the final mixed stream, each user removes its own contribution before playing the stream. The bandwidth required for transmission at each mixer is only what is needed to send to one parent, whereas the bandwidth for reception is proportional to the number of children belonging to the mixer.

The hierarchical architecture does not meet the first requirement of de-centralization since the root mixer acts as a centralized node. However, it can be highly scalable by increasing only the

height of the hierarchy while putting a limit on the number of children for each mixer. Unfortunately, this case will considerably increase delivery time and decrease audio quality. On the other hand, even though the root mixer presents a potential bottleneck, the architecture is still scalable since media streams are considerably reduced before reaching the root mixer. Mixer synchronization seems difficult to implement in this architecture since the mixers are geographically separated and there is no direct communication between them. Finally, hierarchical architecture does not deal with users who do not have the computation capabilities to remove their own contribution from the mixed stream.

3.2.4. Partial distributed mixing

The work done in [54][55] presents an audio service for large-scale collaborative applications on the Internet that supports many simultaneous speakers. It introduces a distributed partial mixing architecture for audio streams. Unlike total mixing, which is based on mixing the whole set of received streams (e.g. centralized mixing), partial mixing dynamically chooses to mix only a subset of the available streams and forwards the rest. In this architecture, a number of mixers (e.g. servers or clients) are connected in a tree-like structure (or according to another graph-structure) to achieve end-to-end audio stream delivery (Figure 3.1). However, mixers do not mix all of the input streams; instead, they mix a subset and forward the rest depending on the underlying network conditions, such as bandwidth availability, congestion, loss rate values, and link speed capacity.

In Figure 3.1, the picture on the right shows the streams that get mixed at different stages in the network. For example, if the network between mixers is experiencing higher loss rates, then the two streams (*a* and *b*) are mixed into a single mixed stream. However, if the network

between mixers has enough bandwidth available then the two streams (*c* and *d*) are forwarded without mixing.

The partial distributed mixing architecture is de-centralized and highly scalable. However, it presents the same shortcomings as hierarchical mixing architecture, such as the introduction of many mixers in the end-to-end paths -- which considerably increases stream delivery time, degrades the audio quality and makes synchronization difficult or even impossible. Thus, streams sent at the same time may arrive at different times and in different sequence order. This variation is caused by the delays introduced by the mixers to some of the streams. This lack of synchronization makes this architecture impractical for conferencing in MANETs.

3.2.5. Distributed video mixing

In the distributed video mixing service model [56], multiple video streams are delivered in a single video stream by choosing the most informative video segments at any time. A scoring system is used to compare video streams. Mixers in this architecture are stretched as a tree that gives successive mixing from sources to destination. Researchers are working on the tree construction algorithm by assuming that a set of mixers is dedicated for each destination. This model is suitable for video streaming but not for conferencing.

3.2.6. Discussion summary

MANETs are composed of various nodes with different computation power and link capacity. Some nodes in this kind of network may have the strict minimum of CPU power, battery energy and link capacity. Therefore, an end-point mixing architecture in which nodes must deal with receiving, sending, decoding and mixing is difficult to implement in this kind of

network. The centralized mixing architecture presents many issues to be applied in MANET conferencing. The main issue is that the centralized system is not scalable. Scalability is very important in MANET growth. Different distributed mixing architectures provide scalability, reduce streams in the network and allow participation by nodes with different media formats. In hierarchical mixing architecture, a higher height of the hierarchy considerably increases the media delivery time and makes synchronization difficult. Distributed partial mixing architecture does not permit rapid delivery since the number of mixers in the end-to-end paths is high. Furthermore, in a tree-structure, synchronization is difficult between mixers since they do not exchange information directly. Table 3.1 summarizes the previously discussed mixing architectures.

	Centralized mixing	End-point mixing	Hierarchical mixing	Partial mixing
De-centralization	No	Yes	Yes	Yes
Scalability	No	No	Yes	Yes
Reduce streams	Yes	No	Yes	Yes
Rapid delivery	Yes	Yes	No	No
Synchronization	Yes	No	No	No
Attendance with different media formats	Yes	No	Yes	Yes

Table 3.1. Comparison between existing mixing architectures

3.3. Media signaling architectures

We define media signaling as the exchange of information, especially dealing with management and control of media connections, with allocation, management and control of mixer and control entities. Existing signaling protocols, such as SIP [33], H.323 [38] and ICEBERG [39], control and manage media connections without any support for mixer and control entities. We have not found a signaling protocol dealing with the management of mixers since there are no existing conferencing systems with dynamic allocation and management of multiple mixers. However, there are three main media signaling approaches in the state-of-the-art that control static media mixers: VoiceXML, Megaco/H.248 and an evolved tree-based Megaco/H.248.

3.3.1. VoiceXML

VoiceXML [57] is a standard language used to develop voice Web applications based on XML specifications. This language was created to solve the problem of the incompatibility of applications developed by different providers. VoiceXML requires a call signaling support to set up, modify and tear down the conferences. The most commonly used call signaling supports are the Call Control eXtensible Markup Language (CCXML) [58] and the SIP-based-VoiceXML browser [59].

When a user wants to join a conference he/she dials the number of the VoiceXML browser (Figure 3.2). The VoiceXML browser is the controller. It authenticates the user and identifies the conference the user wants to join. Then, it refers the user to the conference server, which is the mixer that supports the conference. The user then asks the conference server to join the conference, and establishes a media connection with it. When the user wants to leave, it simply disconnects from the conference server.

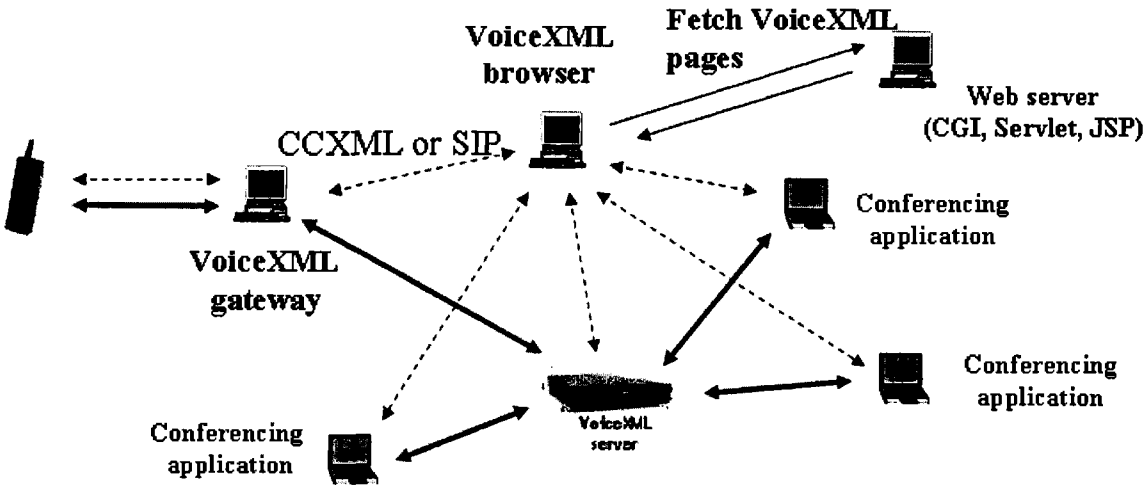

Figure 3.2. Conferencing with VoiceXML

The VoiceXML browser can also act as a media bridge between the user and the conference server. In this case, after authentication of the user and identification of the conference, the browser asks the conference server if the user can join, then establishes media connections with both the user and the conference server. When leaving, the user informs the browser, and then the browser disconnects the media connections.

The VoiceXML browser does not perform any control over the conference server. The latter is explicitly centralized. Therefore, this architecture is not scalable since the server presents a bottleneck for the network. In addition, VoiceXML does not allow the dynamic allocation of mixers.

3.3.2. Megaco/H.248

Megaco/H.248 [62][61] is a protocol that has evolved from the Media Gateway Control Protocol (MGCP). It separates the signaling done by the Media Gateway Controller (MGC) from the media trans-coding and mixing done by the Media Gateway (MG). Point-to-point

connection paths are defined by two main concepts: Termination and Context. A Termination is the source or destination of one or more streams. It identifies an end-point for media flows, applies signals and generates events. A Termination may be established permanently or ephemerally. A Context is an association with a collection of Terminations (Figure 3.3) that defines communication between these Terminations, and acts as a mixing bridge. A Context contains one or more Terminations but a Termination can be in only one Context. A null Context is a special type of Context that contains all the separated Terminations. We map each of the Megaco MGCs to a controller and each of the MGs to a mixer. The Terminations are the users and the Context forms the conference.

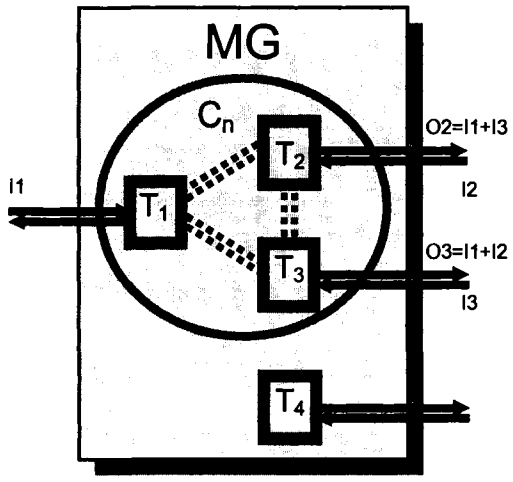

Figure 3.3. Context and Termination concepts in Megaco/H.248

The information that flows between the MGs (mixers) and the MGCs (controllers) is defined by messages containing commands. Reply responses are required for commands. Eight commands are supported by the protocol: Add, Subtract, Move, Modify, Notify, AuditValue, AuditCapability and ServiceChange. The Add and Subtract commands are used to add and remove Terminations (users) to a Context (a conference). The Move command is for moving

Terminations from one Context to another. The Modify command is used to change a Termination's characteristics and the Notify command reports the detection of events. AuditValue and AuditCapability are used to determine the characteristics and the characteristics' values of Terminations and MGs. Lastly, the ServiceChange command is for the negotiation of a new control session or a change in control. Figure 3.4 describes how to make conferences with Megaco/H248 using SIP or H.323 for call signaling. We note that conferencing with two MGs is not defined in the Megaco/H.248 standard and so designers have to implement the necessary functionalities to make it realizable.

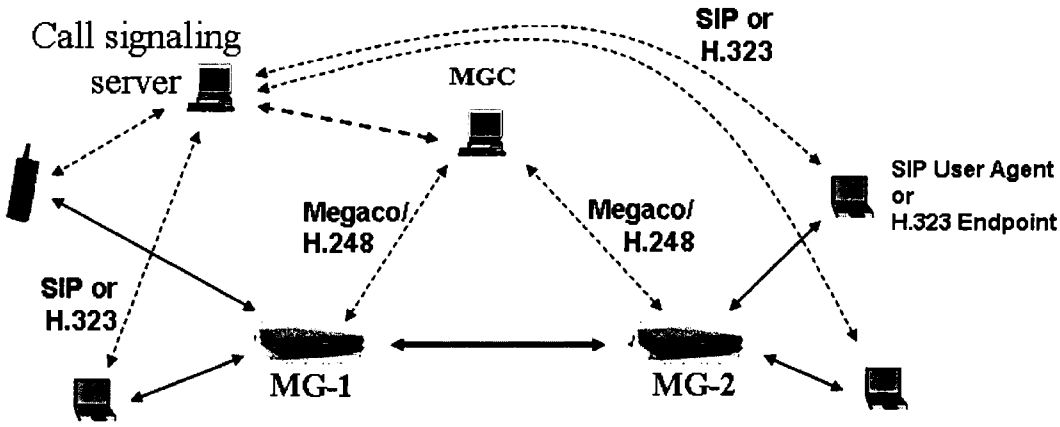

Figure 3.4. Conferencing with Megaco/H.248

In Megaco/H.248 there is no explicit centralization of controllers. However, the specifications do not define the communication mechanisms between the controllers. Moreover, Megaco/H.248 does not include mechanisms for mobile and dynamic environments such as the automatic enabling (disabling) of mixers and controllers. On the positive side, Megaco/H.248 is scalable since media traffic can be distributed among several mixers.

3.3.3. Evolved tree-based Megaco/H.248

The work done in [62] describes an MG tree-based solution for multimedia multipoint conferences on the Internet using Megaco/H.248. Each conference has a root-MG where the conference initiator is connected. The main goal of this tree-based approach is to reduce the system resources' usage compared to the MCU (Multipoint Control Unit) approach, where a centralized unit takes control of conference calls and performs the media mixing and trans-coding (e.g. H.323).

In a tree-based architecture (Figure 3.5), users can perform four operations: creating, terminating, joining and leaving the conference. A conference starts when an initiator adds a Termination in a null Context to its connected MG. A conference is terminated when the initiator leaves the conference. When a user joins a conference, the connected MG adds a Termination to the Context associated with the conference and then the controller MGC makes a point-to-point call between the connected MG and an outgoing MG in the tree by determining the shortest path. When a user leaves the conference, the connected MG notifies its MGC of the departure event. Then, the MGC may apply a local reshaping mechanism to avoid a skewed tree development.

In this architecture, MGs are fully decentralized, which permits scalability -- but the MGC-MG relationship is predefined and does not allow dynamic allocation of controllers. Self-organizing is not applicable in this architecture since the point-to-point MG calls are the only dynamic operations. In addition, the MG tree-based architecture uses a restrictive conference mode such as termination of the conference when the initiator leaves.

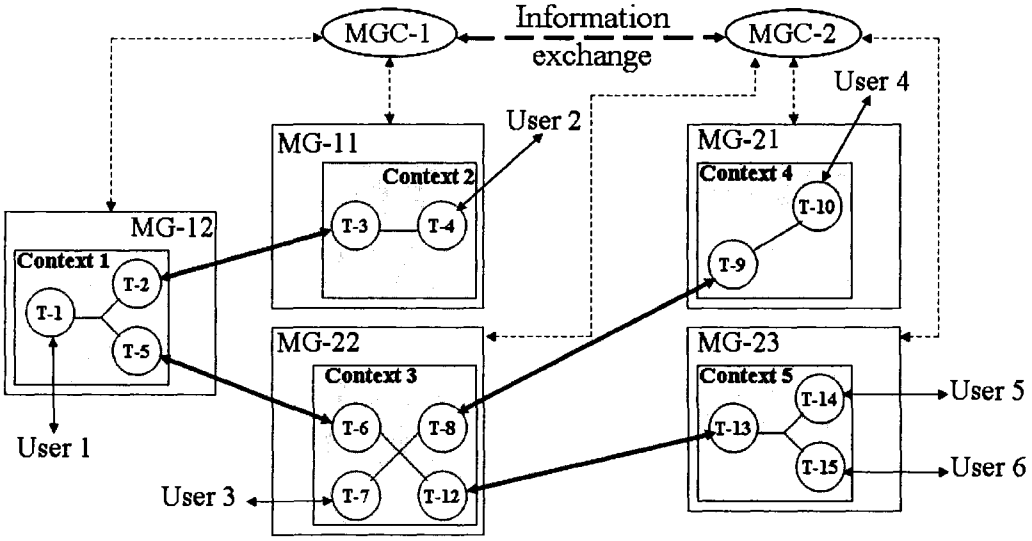

Figure 3.5. Conferencing with evolved tree-based Megaco/H.248

3.3.4. Discussion summary

Various media signaling systems were discussed. Since these systems are not designed for MANETs, they are either not applicable (e.g. VoiceXML, Evolved tree-based Megaco/H.248) or they need enhancement to meet the requirements (e.g. Megaco/H.248). As described above, Megaco/H.248 can be extended to be decentralized by defining the communication between MGCs. It can be made self-organizing by adding mechanisms to MGCs such as self-growing, self-shrinking and self-healing. As well, Megaco/H.248 is standardized, and thus suitable to enhance and to use as a model.

3.4. Self-organizing schemes

The self-organizing paradigm is based on an autonomic biological notion. These systems keep themselves in equilibrium without a higher authority [63][64]. Usually, these self-organizing systems are self-healing and so recover from failures. Self-organizing conferencing systems

(e.g. on the Internet) react to the dynamic changes in the environments where they operate without human intervention. They will indicate to conference joiners the servers (e.g. mixers) which they should connect to. Self-organizing media handling will permit an automatic enabling and disabling of mixers and controllers without external authority. They should include a decision scheme such as deciding when to enable or disable a mixer or a controller. There is a significant body of knowledge about self-organizing on the application layer of peer-to-peer networks [65][66] and on traditional networks [54][67]. In the work done in [54], a self-organized approach for conferencing on the Internet was proposed that describes a dynamic placement of distributed partial mixing servers. The placement is demand-driven by clients. The clients elect the servers (clients that offer to be servers) by choosing the most suitable server offer. Servers are placed on clients' machines. This self-placement is ensured by the full collaboration of clients. Unfortunately, this approach does not permit complete self-organizing since media connections between the clients are statically assigned at the beginning of the session. Finally, there has been very little work done on self-organizing for conferencing in MANETs, especially the media handling aspects. Usually, these self-organizing systems are composed of three components: self-growing, self-shrinking and self-healing.

3.4.1. Self-growing and self-shrinking

Self-growing and self-shrinking allow to a self-organizing system to automatically grow and shrink when nodes join and leave. Usually, self-growing and self-shrinking are based on load balancing schemes that assure resource efficiency by distributing the load among the nodes [64].

3.4.1.1. Load balancing schemes

Load balancing techniques improve network performance by distributing load equitably among nodes. It has been one of the major research interests in both distributed systems and traffic engineering. There are two approaches based on the load information (e.g. load arrival rate, execution rate) kept by the system: static and dynamic. Unlike the dynamic scheme, the static approach does not use any load information [68]. The dynamic approach involves much higher system overhead [69]. It usually outperforms the static scheme.

Heterogeneity of devices in MANETs makes the process of finding a balanced network more complex. The work done in [70] proposes a load-balancing algorithm in MANETs for finding the best-suited node for load sharing. The algorithm relies on lower layer information such as the number of hops. The majority of work done in load balancing in MANETs assumes that loads are used only for communication purposes, especially for routing [71] and clustering [72].

Load balancing is certainly an essential part of an optimal peer-to-peer network. The major objective of these systems is the management of large amounts of distributed data. Usually, these systems try to distribute loads in a self-organizing manner. Many approaches have been proposed for balancing stored data between peers [73][74] or for balancing data retrieval, such as a lookup tree [75][76].

As we have presented above, all load balancing schemes are either information-oriented for application level (e.g. data sharing, data retrieval) or structure-oriented for lower layer networks (e.g. routing, clustering). Till now, there have been almost no studies done on structure-oriented load balancing for application level networks.

3.4.2. Self-healing

A significant feature of MANETs is that nodes are highly susceptible to failure. A failed node can mean a node with its battery down, or it is shutdown, out-of-order, out-of-range, etc. Another specific feature of MANETs is they are particularly vulnerable to message loss caused by frequent out-of-range of nodes, by poor channel quality, and by congestion in some intermediate nodes in the routes. Node failure and message loss usually lead to unpredictable behavior of applications, which should be made to be fault-tolerant. To make these applications fault tolerant in MANETs, they need a mechanism for detecting failed nodes. This mechanism should detect failed nodes and then inform all the other nodes to collectively recover the network. We do not present a related work here for a recovery scheme, since it would be designed specifically for our system. In the rest of this section, we will discuss the existing failure detection architectures, followed by a discussion summary.

3.4.2.1. Failure detection architectures

In the literature, three basic approaches are used for detecting failed nodes in communication networks: heartbeating architectures, pinging architectures and gossiping architectures. In the rest of this section, we present these approaches and then we discuss them according to the requirements for failure detection architectures.

3.4.2.1.1. Heartbeating architectures

Heartbeat protocols are widely used for failure detection in network systems (e.g. [77][78][79]). In these protocols, a node periodically sends a heartbeat message ("I am alive") to a detector node. If the time between consecutive heartbeat messages exceeds a timeout value, then the

node is considered failed. Heartbeat architectures are used in many areas: system diagnosis, network protocols, reaching agreement, and failure detection in computer networks. Many variants of heartbeating architectures are found in the literature, depending on the network topology: centralized, all-to-all, ring-based, and cluster-based heartbeating.

a. Centralized heartbeating

A centralized entity senses the arriving heartbeat messages. Each node periodically sends a heartbeat message to the centralized entity. A node is declared failed if the centralized entity does not hear from it for a defined timeout. This variant is simple to implement. It is often used in devices that control servers to insure that they are running. When the devices miss a user-defined number of heartbeat intervals, they will reboot the servers. Unfortunately, the centralized entity presents a single point of failure and potential bottleneck when the network scales upwards. This architecture is not appropriate for MANETs since they are inherently fully decentralized.

b. All-to-all heartbeating

Every node in the network periodically sends heartbeat messages to every other node [77]. If a node does not receive a heartbeat message from a node after a certain period, it declares that node failed. This variant demands a high bandwidth and consequently it does not scale well.

c. Ring-based heartbeating

In this variant, the nodes are connected logically or physically in a ring. Each node sends a heartbeat message to its successor neighbor when it receives a heartbeat message from its previous neighbor. A node is determined failed if it does not receive a heartbeat message from

its neighbor after a timeout. This variant was used in IBM SP-2 [78]. It presents a high detection delay and it is unpredictable for simultaneous multiple failures.

d. Cluster-based heartbeating

In this variant, the network is partitioned into clusters. Each cluster is maintained by a cluster-head. Different heartbeat styles can be applicable inside clusters and between cluster-heads. For example, it can be all-to-all heartbeating inside clusters and ring-based between cluster-heads. Cluster-based heartbeating is a compromise between centralized heartbeating and all-to-all heartbeating. By nature, it is decentralized and can be easily made scalable. Moreover, it minimizes the system throughput. Studies of cluster-based failure detection issues for MANET applications are still in the very early stages.

The work in [79] presents a cluster-based Failure Detection Service (FDS) for applications that are made up of large and dense populations of lightweight system resources. Applications are built over ad hoc wireless networks. The FDS exploits message broadcasting in wireless networks to build a heartbeat failure-detection service. A cluster-head and the nodes in its range constitute a cluster. Nodes in the range of two clusters may act as gateways for inter-cluster failure forwarding. The heartbeat style is composed of three phases: heartbeat exchange, digest exchange and health-status-update diffusion. In the first phase, every node in the cluster broadcasts a heartbeat message to its cluster-head while the cluster-head broadcasts a heartbeat message to its members. In the second phase, every node in a cluster sends the cluster-head a digest message, which enumerates the nodes that it heard in the first phase. The cluster-head broadcasts its own digest messages to its members. Finally, in the third phase, the cluster-head analyzes the information collected in the two previous phases, identifies the failed

nodes according to a set of failure detection rules and then diffuses an update message to all the nodes. A cluster-formation algorithm ensures the election of cluster-heads and connectivity between the clusters. It does not support routing stability when nodes move. In fact, the current cluster-based heartbeating solution is designed for stationary hosts and does not address node mobility at all, which is critical for MANETs. Consequently, the clusters will disappear once the nodes move. The FDS will then give unpredictable results since it relies on clusters that may no longer be there.

Although the authors claim that their framework can be extended to accommodate host migration, we believe that their framework is still inappropriate for MANETs. First, the iterative cluster-formation algorithm is not designed for mobility, since cluster-head election and gateways' assignment for cluster connectivity are presumed stationary during the execution of the process. Second, when the cluster-heads move out of the range of their members during the heartbeat and digest phases, broadcasted messages will be lost and then failed nodes will not be detected until the stabilization of the clusters' formation, which is very costly in terms of time.

3.4.2.1.2. Pinging architectures

In this approach, a node sends a ping message ("Are you alive?") to another node. The receiving node replies with an acknowledgement message ("I am alive"). Two strategies are used to detect failed nodes. A node is considered failed if it does not send an acknowledgement within a timeout or fails to respond to a defined number of ping messages. Pinging architectures are more vulnerable to message loss than heartbeating architectures because of their acknowledgement feature, which increases message loss.

A variant of pinging architectures, randomized pinging, is described in [88]. Each node randomly picks another node to ping. If there is no response, k other nodes are randomly chosen to ping the suspected node. If an acknowledgement is received by one of the k nodes, the acknowledgement is forwarded to the original node. If no acknowledgement is received by the original node, the suspected node is considered failed. This variant considerably decreases the bandwidth but it presents a high detection delay.

3.4.2.1.3. Gossiping heartbeating

Gossiping architecture was presented for the first time in [81]. Subsequently, some new versions of the gossiping architecture have been proposed (e.g. [88]). In this architecture, each node in the network maintains a list of $<M_i, H_i, T_{last}>$ such that M_i is the address of node i, H_i is the heartbeat count and T_{last} is the last time of the heartbeat increase. Every T_{gossip} time, each node increments its heartbeat, selects a random target node (from its list) and sends to it a constant number of $<M_i, H_i>$ entries. A node, upon receiving a gossip message, merges its list with the list received (taking the maximum of heartbeats). If the sum of T_{last} of M_i and a predefined T_{fail} is less than that of the current system time, then node M_i is considered as failed. These architectures are resilient against message loss but they use a large bandwidth because of the message length.

3.4.2.1.4. Discussion summary

Pinging architectures are not appropriate for MANETs because of their large bandwidth and high detection delay. Gossiping is interesting for MANETs but large messages are not suitable, especially in large-scale networks. Heartbeating can provide rapid failure detection, which is very important in MANETs since messages pass through multi-hop nodes. Unfortunately, the

existing cluster-based heartbeating solution for MANET applications is designed for stationary hosts.

Heartbeating in general is more appropriate for MANETs than the other approaches. Because heartbeating should reside in the application level, an overlay network should be deployed on the physical layer. In addition, this overlay requires its own self-organizing scheme to manage the overlay network formation.

3.5. Media handling architectures for MCNs (MHAM)

There is no work that has explicitly addressed media handling for conferencing in MCNs. However, there is a relevant architecture that may be useful in designing the MHAM. In this section, we describe the 3GPP architecture for media handling for conferencing in 3GCNs, and then we discuss this architecture according to the requirements specific to media handling architectures for conferencing in MCNs.

3.5.1. Media handling for conferencing in 3GCNs

Until now, the 3GPP has focused on the infrastructure-based mode when a 3GCN is integrated with wireless networks. The standard architecture for media handling in 3GCNs is described in [83]. In this architecture, the media handling functionalities are assured by a component called the Multimedia Resource Function (MRF) described in Figure 3.6. The MRF is composed of two functional entities: the Multimedia Resource Function Controller (MRFC) and the Multimedia Resource Function Processor (MRFP). The MRFC controls the media stream resources provided by the MRFP and interprets external information from the Application Server (AS) or from a Serving-Call Session Control Function (S-CSCF) and controls the MRFP accordingly.

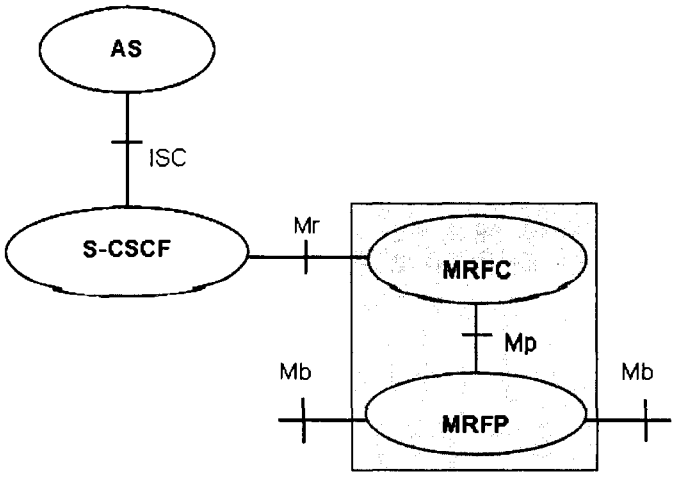

Figure 3.6. Architecture of the MRF

The AS hosts the conferencing application. It can have many tasks, including conference booking, management of booking information (e.g. start time, duration, list of users). Any user that wants to attend the conference should establish call signaling with the AS. Consequently, the AS triggers the MRFC to create connection resources for the user. The S-CSCF is the centralized server that handles call signaling. It is a SIP server [33] located in the home network that provides routing services and decides to which AS the SIP messages will be forwarded.

Examples of external information manipulated by the external interface include session identifier and joiner address. The external interface may be specified within the Mr interface between the MRFC and an S-CSCF, or within the AS in which the MRFC function resides. The protocol used for the Mr reference point is SIP. The Mb reference point defines the interface between the MRFP and the other bearer entity, if any.

The MRFP provides media stream resources to the MRFC, such as mixing incoming media streams (e.g. for multiple users), media stream processing (e.g. audio trans-coding, media

analysis), and floor control (i.e. managing access rights to shared resources).

The protocol on the Mp interface between the MRFC and the MRFP is defined to comply with the standard of ITU-T H.248.1 Gateway Control Protocol (part of Megaco/H.248 [60]). This protocol defines the media signaling protocol between the MRFP (the MG) that establishes, terminates and manipulates media streams and the MRFC (the MGC) that controls the MRFP. It uses the concepts of Termination and Context as described in Megaco/H.248 and given previously (subsection 3.3.2).

The 3GCN media handling architecture cannot meet most of our requirements. It cannot handle conferences with users located in a MANET. If we were to connect MANET users, all the media streams would be sent to the MRFP. The scale of the conference will be constrained by the capability of the resources of the MRFP. Moreover, the performance of the 3GCN will then depend on the number of users in the MANET. However, 3GCN media handling architecture can be enhanced to support MANET users. In this case, 3GCN will have to identify and locate MANET users. Unfortunately, media streams of MANET users located far from the MRFP may experience much longer delays than those experienced by streams of 3GCN users and users who are close. Consequently, performance for MANET users will be very dependent on the users' location, making the 3GCN media handling architecture not suitable for the MCN environment.

3.6. Summary

In this chapter, we gave an overview of work related to media mixing architectures, media signaling architectures, self-organizing schemes, load balancing schemes, failure detection architectures, and media handling architectures for MCNs. A series of review criteria was

selected, and evaluated for its importance to handle media for conferencing in MANETs and MCNs. Each of the reviewed architectures and schemes was discussed according to its specific set of criteria.

CHAPTER 4

TWO-OVERLAY AND MEDIA MIXING ARCHITECTURES

As we have described in the introduction, the two-overlay architecture is composed of two constituent architectural solutions for each overlay network. The first overlay network is overlayed over the set of nodes that have mixing capabilities for mixing purposes. It is based on a novel media mixing architecture called distributed mixing architecture and presented in this chapter. The second overlay network is overlayed over nodes with control capabilities for control purposes, is based on a novel media signaling architecture described in the next chapter.

This chapter presents the two-overlay architecture and the media mixing architecture for media handling for conferencing in MANETs. It is divided into three sections. The first section introduces the two-overlay architecture. It presents the overall architecture including architectural principles and an illustrative scenario, and then describes nodal aspects in terms of node life cycle, node functional architecture, and the information kept in each node. The second section presents the novel media mixing architecture, namely the Distributed Mixing Architecture (DMA). It describes the DMA principles followed by a two-step mixing operation and a synchronization algorithm. Finally, the third section gives a short summary of the chapter.

Results from this chapter have been published in [92], [93], and [95].

4.1. Two-overlay architecture

This section is divided into two parts: overall architecture and nodal aspects.

4.1.1. Overall architecture

This subsection presents the architecture principles and an illustrative scenario.

4.1.1.1. Architectural principles

As we described in the system assumptions, there are three kinds of nodes in the network: nodes with no special capabilities, nodes with mixing capabilities and nodes with control capabilities. Nodes with mixing (or control) capabilities can be enabled to act as mixers (or controllers), or can be disabled to be simple nodes. Nodes with no special capabilities are usually called simple nodes. We note that some nodes can have mixing and control capabilities at the same time and can be both controllers and mixers at the same time. So, mixers and controllers located in the same nodes can exchange messages internally if they have to communicate.

We propose a two-overlay architecture for media handling for conferencing in MANETs (Figure 4.1). The first overlay network is overlayed on the set of nodes that have mixing capabilities. This overlay network is composed of mixers (enabled mixers). Every simple node has a media connection with a mixer in the first overlay. The first overlay network can also be described using level concepts as follows: nodes in the network that form the conference on top of the devices' network constitutes the first level, and a set of nodes acting as mixers forms the second level, which provides the service of mixing to the simple nodes.

The overlay of mixers is highly dynamic in MANETs. Disabled nodes with mixing capabilities can be enabled anytime and become part of the first overlay, when more nodes join the conference. Conversely, when nodes leave the conference, mixers may be disabled and be moved to the set of simple nodes.

To control the first overlay network (both levels), we overlayed the set of nodes that have control capabilities in the network by a second overlay network. The second overlay network is composed of controllers (enabled controllers) (Figure 4.1). Every controller has a control connection with a set of controlled mixers. Each mixer is controlled by one controller and a controller may control one or many mixers.

Controllers are fully mesh connected. This means that each controller has control connections with all the other controllers. Full mesh topology is motivated by the fact that control connections are lightweight connections, compared to media connections that require physical allocation of resources (e.g. channels, buffers). This topology facilitates the failure detection and recovery of controllers.

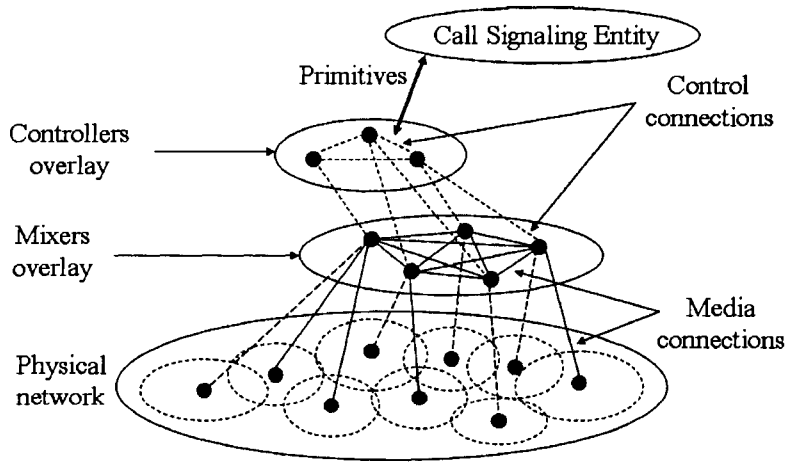

Figure 4.1. Two-overlay architecture

The Call Signaling Entity (CSE) (Figure 4.1) in each node handles call sessions. It also provides discovery of controllers. Communication between the CSE and controllers is defined by a set of primitives (e.g. connect_request, disconnect_request, modify_request). After call signaling (i.e. of an invitation process) is established between a joiner and the node that invites the joiner (called the host), the latter needs to discover a controller (called the contact controller) that will make the necessary media connections. This invitation process is handled by the corresponding CSEs and is inherent to the dial-out ad-hoc conferencing mode used in this thesis. The discovery functionality consists of the ability to find a suitable controller that can connect the joiner. Furthermore, the discovery concept preserves the independence between call signaling and media handling. The discovery of a contact controller can be done by different methods such as multicast or anycast. However, we designed a simple scheme of contact controller discovery, described as follows: If the node that invites the joiner is a controller, then it will be the contact controller; otherwise, the contact controller is that controller to which the node that has invited the joiner is already connected to (directly or via a mixer).

The most critical feature of the overlay of controllers is that it is highly dynamic in MANETs. Disabled nodes with control capabilities can become enabled anytime and part of the overlay of controllers, when more mixers are enabled. Conversely, when mixers are disabled, controllers may become disabled and be moved out of the overlay of controllers.

The two-overlay architecture can also be described in terms of network levels. Simple nodes (nodes that do not do mixing) form the first level, the set of nodes acting as mixers is the second level, and the set of controllers composes the third level. The three levels of the architecture can be made self-organizing by simply adding automatic mechanisms for decisions

and organization. Moreover, this architecture can be easily deployed in a resource efficient manner by assigning the role of mixers and controllers to the most powerful nodes in the network. Therefore, the self-organizing system should ensure an optimized use of resources and make decisions to preserve this optimization throughout the conference progress.

4.1.1.2. Mixer enabling scenario

When a joiner is invited to the conference and then has to be connected to a mixer, the host first discovers a contact controller and sends it a connection request primitive. Then, the contact controller determines if the joiner should be connected to an existing mixer or to a new mixer by evaluating a decision algorithm. If the contact controller decides to enable a disabled node, denoted by *NewMixer-2* (Figure 4.2) (e.g. it will be controlled by the contact controller), then it sends an *'Enable_request'* message to *NewMixer-2* requesting it to enable mixing. Subsequently, *NewMixer-2* enables as a mixer, establishes media connections with current mixers and then sends a confirmation to the contact controller. Finally, upon receiving the confirmation, the contact controller sends a message to *NewMixer-2* requesting it to connect the joiner. If the contact controller does not receive a confirmation from *NewMixer-2* after a given time, for example (e.g. *NewMixer-2* may become out-of-range), then it repeats the process described above to reconnect the joiner.

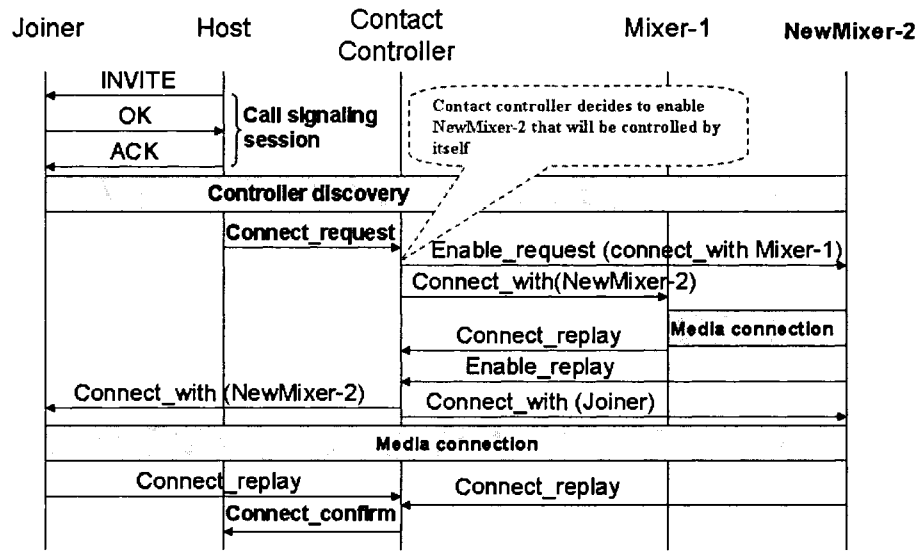

Figure 4.2. Mixer enabling scenario

4.1.2. Nodal aspects

The nodal aspects are presented in terms of node life cycle, functional entities and interaction between the different entities, and the information kept in each node.

4.1.2.1. Node life cycle

The node life cycle describes the different node states from joining to leaving the conference (Figure 4.3). When joining the conference, nodes publish their capabilities. Nodes with mixing (and/or control) capabilities then start conferencing and in the disabled state, and they may be enabled during the conference. The two-overlay architecture should include schemes to decide if nodes having mixing (or control) capabilities become mixers (or controllers) or remain disabled nodes. System decisions come up as external events to a node, launching transition from one status to another.

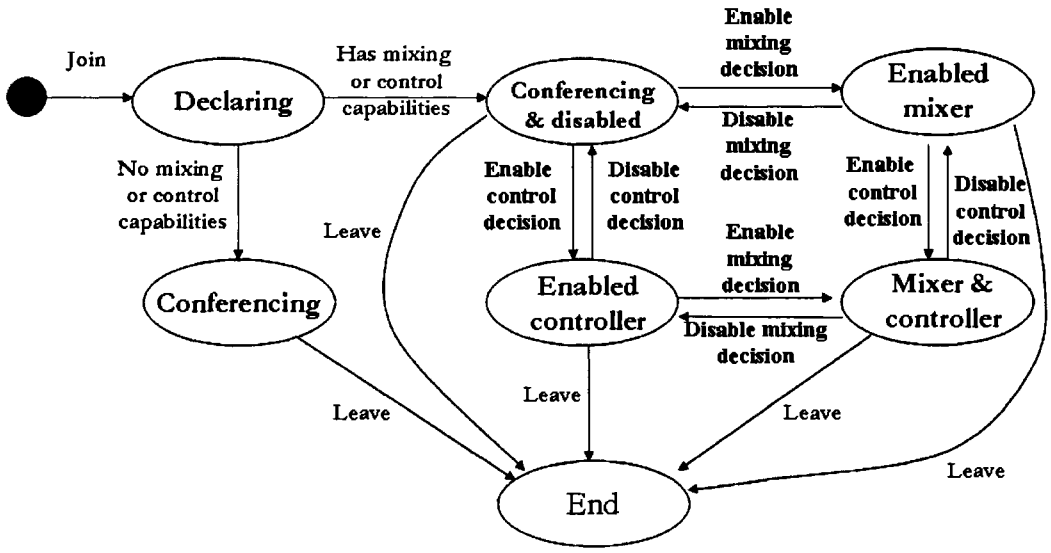

Figure 4.3. Node life cycle

4.1.2.2. Node functional architecture

There are three user agents in the node: User Conferencing Agent (UCA), User Mixing Agent (UMA) and User Automatic Control Agent (UACA) (Figure 4.4). The UCA is always activated. It contains conferencing and media connection management entities. The UMA includes entities that deal with mixing. It may be deactivated when the node does not do mixing. The UACA can be activated when the node is enabled to act as a controller or deactivated. The automatic manager entity is the core entity that assigns joiners to mixers and enables and disables mixers and controllers. It uses the service of media signaling handler entity, information manager entity and decision maker entity. Decision maker entities of the different controllers coordinate together to make decisions. To achieve this goal they use the service of information manager entities that maintain the information kept in each node. The media signaling handler entities of the different nodes implement the media signaling protocol that provides the service of exchanging messages between nodes in the network.

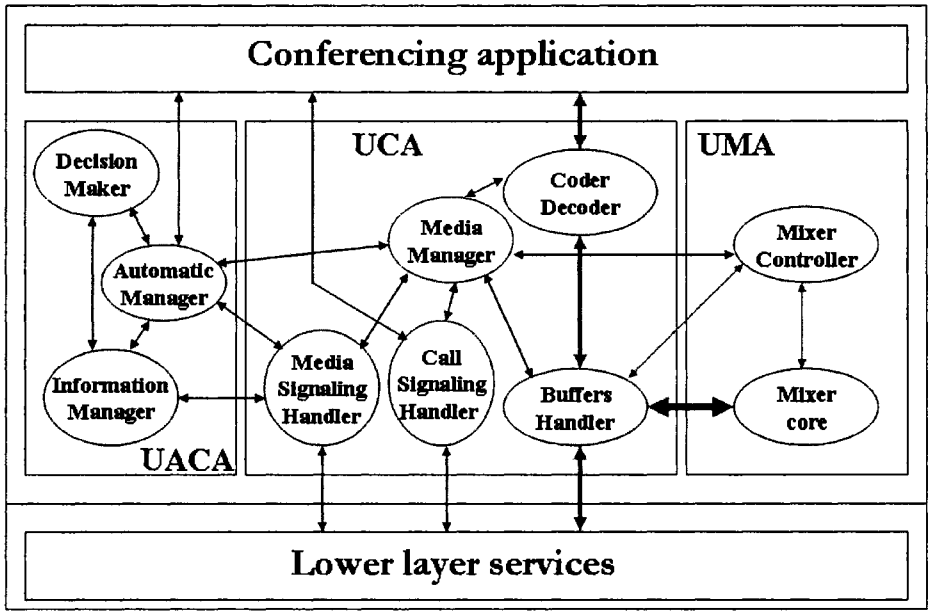

Figure 4.4. Functional entities

4.1.2.3. Information kept in each node

There is no information stored in simple nodes and mixers concerning the structure of the two-overlay network. The mixers only maintain information about their served nodes for management purposes. On the other hand, the controllers have to maintain information about the two-overlay network. In addition, they have the responsibilities to enable and disable mixers, enable and disable controllers and recover from mixers' and controllers' failures. Furthermore, controllers assign and connect conference joiners to mixers and disconnect leaving nodes. To reach these goals, each controller needs to store and update information about active (enabled) controllers and mixers, potential (disabled) controllers and mixers, and simple nodes. Every controller maintains two different lists of information described as follows:

- **Shared information:** decentralized and duplicated information used to maintain the full mesh structure of controllers and the structure of controllers with their controlled mixers. It consists of the list of the information of all of the controllers with their controlled mixers. Shared information also includes the list of the information about potential (disabled) mixers and controllers in the network. The shared information is updated by the exchange of messages between controllers after each event concerning mixers and controllers. The replication of mixers' information is useful when recovering from controllers' failures.

- **Local information:** information specific to each controller and is not shared with the other controllers. It consists of the list of the information of all of the nodes served by mixers controlled by the controller in question. Updating this information is done after the occurrence of node joining and leaving events. This list is used to maintain structure information about controllers and their controlled mixers with their related nodes, and it is also used to recover from mixers' failures.

4.2. Media mixing architecture - Distributed Mixing Architecture (DMA)

In related work, we argued that the distributed mixing approach is the most suitable architecture for conferencing in MANETs from scalability, reduction of streams in the network and attendance with different media formats, viewpoints. The challenges of the foreseen architecture are rapid delivery and good synchronization to deliver media streams with correct ordering and timing. To use resources in an efficient manner, powerful nodes act as mixers by playing the additional role of mixing. The choice of mixers from the set of nodes is fixed by some criteria based on specific capabilities (e.g. processing power, memory capacity,

battery energy and link capacity). The determination of capability values based on devices' features is not the scope of this thesis. The proposed solution is a novel media mixing architecture called Distributed Mixing Architecture (DMA). In the rest of this section, we describe the DMA principles followed by the mixing operation, synchronization algorithm and mixer core functional architecture.

4.2.1. DMA principles

The main requirements of the DMA are rapid delivery and along with an associated synchronization mechanism. To tackle these challenges, we need an architecture in which the number of mixers in end-to-end path does not depend on the number of nodes in the network. Furthermore, to make synchronization easy, mixers should communicate directly with each other. In the DMA, we divide the nodes in the network into two levels (Figure 4.5). The first level is a flat structure of enabled nodes performing mixing (mixers) and forming a sub network. Nodes that do not perform mixing (simple nodes) are part of the second level. Each simple node is served by one mixer (its dedicated mixer) in the first level. Each mixer acts as a centralized mixer for a set of second-level nodes, called mixer's related nodes. As with the two-overlay architecture described earlier, the sub network of mixers forms the first overlay and all the nodes (mixers and simple nodes) form the physical network.

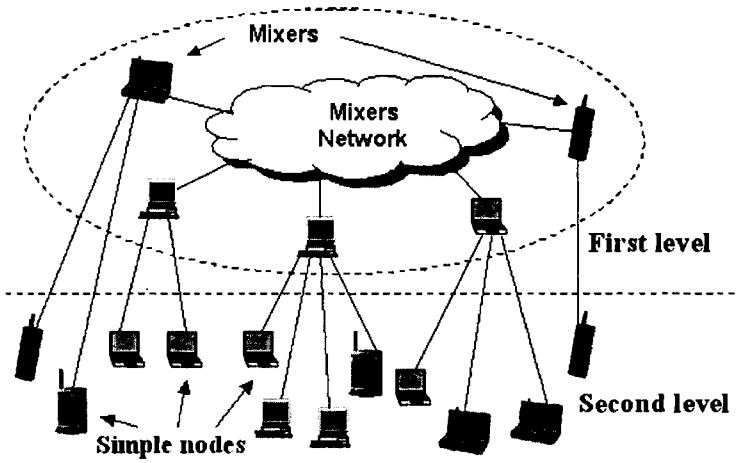

Figure 4.5. Two-level overlay network for mixing

The topology of the mixer level can be fully or partially meshed, but a partially meshed topology is not suited for real-time media transportation in MANETs since it increases the media delivery time. Therefore, we have used a fully meshed topology for the mixers' sub-network, since it reduces intermediate nodes in the end-to-end paths. Subsequently, the maximum number of mixers in the end-to-end paths will be two. If the two nodes are connected to the same mixer it will be one otherwise it will be two. This topology is especially well suited for real-time media transportation because in MANETs media streams may pass through multi-hops between every two nodes. Furthermore, fully meshed topology facilitates the synchronization of mixers since they are directly connected.

Mixers can send media streams to other mixers by various communication modes such as flooding, multi-unicasting, or multicasting. Flooding increases the waste of resources, especially bandwidth, by duplicating media streams in the network. On the other hand, multi-unicasting may guarantee synchronization between mixers but it needs power computation capabilities and bandwidth resources. Multicasting provides a solution to avoid wasting network resources,

but it includes transmission delays introduced by routing operations. The choice of mixers' topology and a communication mode usually is guided by the underlying network characteristics and the network resources (e.g. nodes' computation power, network bandwidth).

As media streams will be subject to different delays and possibly a different quality of service, streams that were sent at the same time by one source may not reach the receiver at the same time. Inter-flow synchronization can be achieved by adapting the play-out so that streams originating at the same time are played at the same time. This requires that the times at which different flows from the same mixer were captured are available to the other mixers. We restrict this synchronization to mixers only because it is impractical to synchronize all of the nodes in the network. The two-level structure facilitates this synchronization of mixers since they are directly connected.

This two-level structure significantly reduces the number of media streams in the network and enables rapid delivery because of the flat structure of the mixers. Obviously, the maximum reduction is obtained with one centralized mixer in the network. Finally, the architecture meets the requirement of rapid stream delivery since there are no more than two mixers in the end-to-end paths. In the case of streams exchanged between a mixer's related nodes, there is just one intermediate mixer, which is their dedicated mixer. Finally, network resource efficiency can be improved by allowing powerful nodes to contribute more by playing the additional role of mixing.

4.2.2. Two-step mixing operation

In the proposed distributed mixing architecture, mixers follow specific rules to exchange media streams with the nodes, which are connected to them, and also with the other mixers. Obviously, mixers should not return media streams to their sources. On the other hand, mixers should not mix the entire input in one step as in centralized mixing. Otherwise, streams could loop indefinitely between mixers.

We propose mixing in two steps. The first step is called mixing for external sending. In this step, streams coming from related nodes are mixed and immediately sent to the other mixers. In the second step, called mixing for internal sending, buffered streams from the first step are mixed with the streams coming from the other mixers. The different mixed results are sent to related nodes. However, this step consists of many mixing operations. For each node connected to the mixer, the mixer mixes the streams coming from the other mixers with the buffered streams (except the stream sent by the node, if any), and then sends the result to the node. This step takes advantage of the fact that only a few nodes are typically simultaneously sending streams to the mixer. The mixer itself can send and receive media streams to and from other nodes. Its streams are treated just like the streams from the nodes, which are connected to it. In this architecture, every stream is mixed twice, at most.

Figure 4.6 illustrates a scenario in which node F sends a stream to its dedicated mixer E and at the same time nodes I and J send streams to their dedicated mixer H. The figure shows the streams that are mixed at different steps in the system.

For example, in the first step, streams i and j are mixed by H and sent immediately to A and E. At the same time stream f is just forwarded by E to the other mixers. Upon receipt of the

streams, *A* mixes all the input and sends the result to the nodes which are connected to it. However, mixer *H* operates differently. It mixes stream *i+j* with stream *f* to deliver stream *i+j+f* to *K*. After that, it mixes and delivers streams *j+f* and *i+f* to *I* and *J*, respectively.

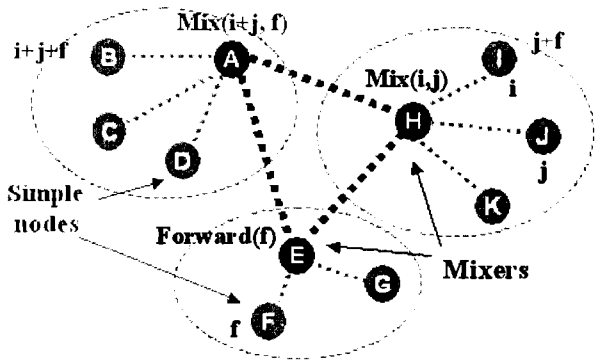

Figure 4.6. A scenario of a mixing operation

4.2.3. Synchronization algorithm

Media streams may be received with different delivery times, streams that were grabbed at the same time at the sender may not arrive at the receiver at the same time. Inter-flow synchronization can be accomplished by adapting the play-out so that streams that originated at the same time are played at the same time. This requires that the times that different flows from the same sender were captured are available at the receivers. Synchronization in distributed systems is a classical problem. In addressing this problem, Lamport has proposed a distributed algorithm based on logical clocks [84][85]. We can use this as a basis because we can specify the distributed mixing problem without referring to real time. Logical clocks substitute real-time clocks in the system, simulating the potential causality between events. However, a problem arises when logical clocks can be used instead of real-time clocks. In general, it depends on the system specification and whether it makes references to real time or

not. For example, there is no reason to mention real time in the specification of the distributed election problem or the mutual exclusion problem. Therefore, we can specify the distributed mixing problem without referring to real time. The specification can be made in terms of the observable events in the system. We assume that our system is composed of a collection of mixers. We also assume that sending/receiving media messages are the only observable events. Every mixer is dedicated to a number of nodes. Mixers exchange messages between them. Every mixer is dedicated to a number of nodes, which receive and send messages via the mixer in question. We assume that there are no message delays between mixers and their related nodes. Therefore, we do not consider sending and receiving between mixers and their related nodes to be events. The algorithm for this system must meet the two following conditions: (I) Messages coming from nodes must be delivered in the order in which they were sent. (II) Messages that arrive at the same time at different mixers from their related nodes must be delivered at the same time. We note that our system specification does not refer to real time. Therefore, Lamport's logical clocks can solve our synchronization problem. Logical clocks may be implemented by counters with no actual timing mechanism, and they monotonically increase with varying granularity. Then, counters can implement logical clocks with no actual timing mechanism and we have implemented a counter in each mixer. We add to each message m a timestamp T_m that equals the time at which it was sent.

Lamport defines two implementation rules that guarantee a correct system of logical clocks: (IR1) each mixer increments its logical clock between successive events (sending or receiving messages to or from other mixers); (IR2) upon the receipt of a message m time-stamped T_m, a mixer must advance its logical clock C to be higher than T_m. More precisely, the logical clock C is set to the maximum of its value and T_m+1. We consider that the event of receiving the

message m occurs after the setting of C by the first rule. In Figure 4.7 we present the distributed synchronization algorithm.

```
clock = 0;
while true do {RUN}
        event = event();
        if event = EXIT then
           exit RUN;
        else if event = "send m" then
              T_m = clock;
              send [m, T_m] to Mixers;
              clock = clock + 1;
        else if event = "receive [m', T_m'] from Mixer" then
              clock = clock + 1;
              clock = maximum(clock, T_m' +1);
        end if;
end; {RUN}
```

Figure 4.7. Distributed synchronization algorithm

4.2.4. Mixer core functional architecture

This section describes the mixer core functional architecture (Figure 4.8). The mixer acts both as a source and as a receiver. It consists of six principal components: Receiver, Buffers, Mixing component, Synchronizer, Control unit and Transmitter. Before arriving at the mixer, streams are decoded according to their source media format (e.g. codec), and then received by the receiver component, which removes jitter and routes them to buffers. Streams from related nodes are immediately passed to the mixing component. The mixed result is sent to different mixers by the transmitter component. The latter operation ends the first step by buffering the mixed result. Furthermore, a copy of previous streams stays in the buffers. Buffers are

controlled by the control unit component. In the second step, buffered streams and the mixed result obtained from the first step are mixed with streams from the other mixers. The synchronizer component prepares streams to be mixed in the second step. Mixing is done on streams that have the same timestamp. This second-step mixing operation is done without returning streams to their sources. The control unit component controls the mixing operation and updates the logical clock of the mixer.

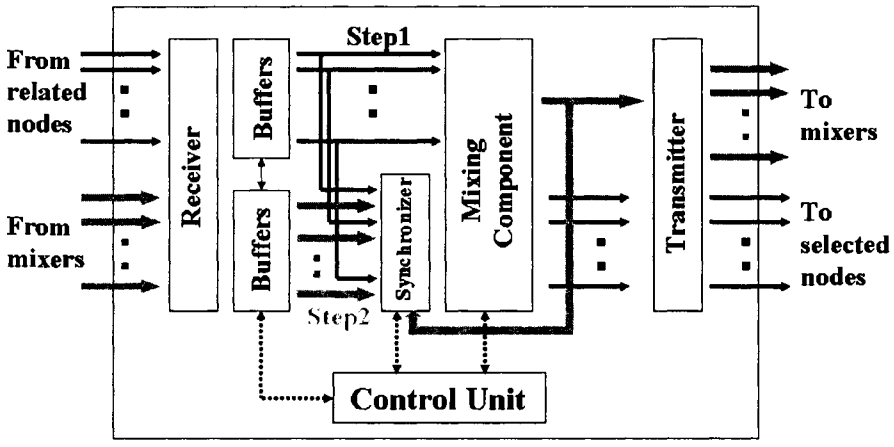

Figure 4.8. Mixer core functional architecture

4.3. Summary

This chapter has introduced two-overlay and media-mixing architectures as approaches for media handling for conferencing in MANETs. In the first part of the chapter, the first overlay network for mixing purposes and the second overlay network for control purposes are described as a solution for the MANET environment. The nodal aspects are also described, including node life cycle, node functional architecture and the information kept in each node. The second part of the chapter presented the novel approach for distributed mixing, called Distributed Mixing Architecture (DMA), used by the first overlay network of the two-overlay

architecture. A two-step mixing operation is introduced to describe rules and operations on media streams. After that, a synchronization algorithm is proposed to synchronize between mixers in order to ensure that media streams that were grabbed at the same time at the senders will be delivered to the receivers at the same time. Finally, the mixer core functional architecture is presented with a description of each component and of streams flows between components.

The next chapter describes the media signaling architecture used by the second overlay network of the two-overlay architecture. This architecture is necessary to manage and control media connections, media mixers and controllers. The next chapter presents the different procedures occurring after each event without referring to how decisions are made.

CHAPTER 5

MEDIA SIGNALING ARCHITECTURE

This chapter focuses on the media signaling aspects of media handling for conferencing in MANETs. It presents a novel architecture for managing media connections and controlling media mixers. Control of media mixers, performed by controllers, consists of enabling/disabling media mixers, assigning tasks to mixers and recovering when mixers become suddenly unavailable or out-of-range (i.e. failed). Controllers also control themselves by enabling and disabling controllers. The proposed architecture is based on an extended version of Megaco/H.248, an approach for controlling media mixers in traditional networks. We describe this architecture in terms of principles, primitives, events, messages and procedures.

This chapter is divided into three sections. The first introduces the overall architecture. It presents the architectural principles, primitives, events and the semantics of messages. The second section illustrates the procedures for different events that occur during the conference. The third section gives a short summary of the chapter.

5.1. Overall architecture

Megaco/H.248 [61] separates the media signaling done by the Media Gateway Controller (MGC) (the controller) from the media mixing done by the Media Gateway (MG) (the mixer). Node-to-node media connections are defined by two main concepts: Termination and

---

Results from this chapter have been published in [95].

Context. A Termination is the source or destination of one or more media connections. A Context is an association with a collection of Terminations that defines communication between these Terminations, and acts as a mixing bridge. A Context contains one or more Terminations but a Termination can be in only one Context. We map mixers in the two-overlay architecture as MGs in Megaco/H.248, and controllers as MGCs. Each simple node is affiliated with a Terminal in an MG.

5.1.1. Architectural principles

As described in the related work chapter, Megaco/H.248 can be decentralized by defining the communication between MGCs. It can be made self-organizing by adding mechanisms to the MGCs such as self-growing, self-shrinking and self-healing. As well, it is standardized, and therefore, Megaco/H.248 is suitable to use as a basis.

We define a local Context as the Context associated with the conference in an MG. A local Context defines the association between a mixer and the nodes that are connected to it for mixing. An MGC (controller) can control one or more MGs (mixers) but each MG is controlled by only one MGC (Figure 5.1). The affiliated MGC of an MG is its controller (e.g. MGC-1 is the affiliated MGC of MG-11 and MG-12).

For example, if the call signaling session for inviting a user or a user leaving a conference succeeds, the CSE discovers an MGC, called a contact-MGC (it may or may not be the same MGC that controls the MG to which the conference joiner will finally be connected or the leaving user is already connected), and invokes the corresponding primitive. The discovery of a contact-MGC can be done by different methods such as multicast or anycast. Next, the contact-MGC executes the joining or leaving procedure to add a Termination in an MG to the

local-Context associated with the conference or remove the Termination from the local-Context in the MG that connects the leaving user, respectively.

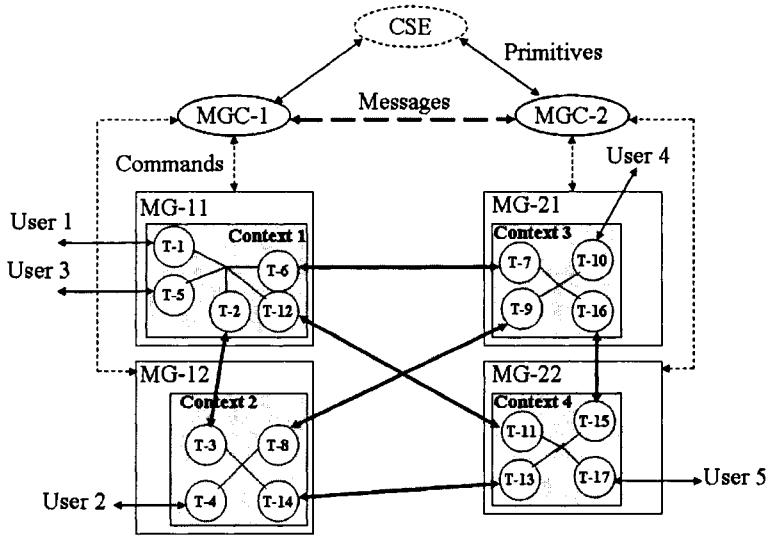

Figure 5.1. Overall architecture

We still use the same master/slave approach between MGCs as is used between MGCs and MGs as defined in Megaco/H.248. The sender of the command assumes the role of master, and the receiver assumes the role of slave. Commands encapsulated in messages are sent from a sender-MGC to a receiver-MGC. Typically, the sender-MGC is the contact-MGC and the receiver-MGC is the target-MGC that controls the target-MG specified in the command. A typical scenario is described next.

Upon receiving an invoke primitive from a CSE, the contact-MGC determines the target-MG that will perform the operation. It then sends a command to the target-MGC that controls the target-MG. The target-MGC forwards the command to the target-MG. When the operation is performed, the target-MG sends a Reply response to the target-MGC, which in turn relays the

response to the contact-MGC. Finally, the contact-MGC replies by invoking the corresponding confirm primitive.

5.1.2. Primitives

We totally separate the Call Signaling Entity (CSE) and the MGC entity. Communication between them is defined by a set of primitives described as follows.

- *Connect_request primitive.* This primitive is used to connect a joiner to the local Context associated with the conference in an MG. If the call signaling session for inviting a user succeeds, the CSE discovers a contact-MGC, and invokes this primitive. If the media connection is established successfully, the contact-MGC replies with a Connect_confirm primitive.

- *Disconnect_request primitive.* This primitive permits a leaving user to disconnect. If the call signaling session for a leaving user is successful, the CSE discovers a contact-MGC and invokes this primitive. The contact-MGC replies using the Disconnect_confirm primitive.

- *Media_connection_modify_request primitive.* This primitive is used to change the characteristics of a media connection such as the media type (audio or video), the sampling rate or the bit rate. The CSE discovers a contact-MGC and invokes this primitive. When the modification has been made, the contact-MGC replies to the CSE by the Media_connection_modify_confirm primitive.

5.1.3. Events

Two events are generated by the CSE: joining a conference and leaving a conference. These two events may trigger another two events generated by MGCs: enabling an MG and disabling an MG, which in turn may trigger another two events generated by MGCs: enabling an MGC and disabling an MGC. Furthermore, two other events: failed MG and failed MGC are triggered by MG and MGC failures, respectively.

The shared information maintained by each MGC contains information about MGCs (enabled and disabled) and MGs (enabled and disabled) and their relationships. Updates are made by a multicast of the ServiceChange command after each event involving MGs and MGCs (enabling an MG, disabling an MG, enabling an MGC, disabling an MGC, failed MG and failed MGC). Moreover, to allow for failure recovery, each MGC uses local information about the nodes connected to its connected MGs. Updates of local information are made after node joining and leaving events.

5.1.4. Semantics of messages

We do not modify the semantics of messages exchanged between MGCs and MGs defined in Megaco/H.248. We have used the same vocabulary and concepts used by Megaco/H.248 to define exchanged messages between MGCs. In the rest of this subsection, we present the semantics of messages exchanged between MGCs.

The commands supported between MGCs are shown in Table 5.1. The address carried by a command can be a unicast or a multicast address (e.g. multicast IP address). It should be noted that commands still carry Megaco's parameters such as Termination Descriptors. In Table 5.1, we describe only the extra parameters of commands between MGCs.

Command	Purpose
Add	Add a Termination to a local-Context. Address: Unicast. Parameters: The target-MG to which the conference joiner will be connected. The list of MGCs if the receiver-MGC is disabled. Preconditions: The target-MGC may or may not be enabled. The target-MG may or may not be enabled. The Context may or may not exist. The Termination does not exist. Post-conditions: If the command succeeds, the target-MGC and target-MG will be enabled and the Context and Termination will both exist.
Subtract	Remove a Termination from an existing Context. Address: Multicast. Precondition: The Termination may or may not exist. Post-conditions: If no Terminations remain in the Context, this one will be destroyed.
Modify	Change the characteristics of an existing Termination. Address: Multicast. Precondition: The Termination may or may not exist.
Notify	Report that one or more events have occurred at the target-MGC or target-MG. Address: Unicast or Multicast.
Audit-Capability	Determine the possible values supported for the characteristics of a given MG or MGC. Address: Unicast.
Service-Change	Declare the MG or MGC in- or out-of-service. Address: Multicast.

Table 5.1. Commands between MGCs

5.2. Procedures

Members participate in a conference through the use of such procedures as "joining" or "leaving" conferences, which in turn may trigger procedures (by generation of events) such as "Enabling an MG", "Disabling an MG", "Enabling an MGC" or "Disabling an MGC". Furthermore, two other procedures: "Recovering an MG" and "Recovering an MGC" are triggered by failed MG and failed MGC events, respectively. These conference procedures are described in detail in this section.

5.2.1. Joining a conference

When a node joins a conference, the CSE discovers a contact-MGC and invokes the Connect_request primitive. Next, the contact-MGC sends an AuditCapability command to the joiner. Simultaneously, the contact-MGC determines the target-MG (or enables one if necessary) and then identifies the target-MGC (enabling one if necessary) that controls the target-MG. Next, the Contact-MGC sends an Add command to the target-MGC, which in turn sends an Add command to the target-MG. The target-MG then adds a Termination to the local-Context associated with the conference, and replies to the target-MGC of the connection being created. The target-MGC replies to the contact-MGC, which in turn replies to the CSE using the Connect-confirm primitive. After sending an AuditCapability command, the contact-MGC waits for a reply for a specified time after which it decides that the joiner is a simple node. If this is not the case, it will receive a Reply message indicating the joiner's capabilities and it will notify the other MGCs in the network. Figure 5.2 illustrates the exchange of messages between nodes involved in connecting a joiner to the conference.

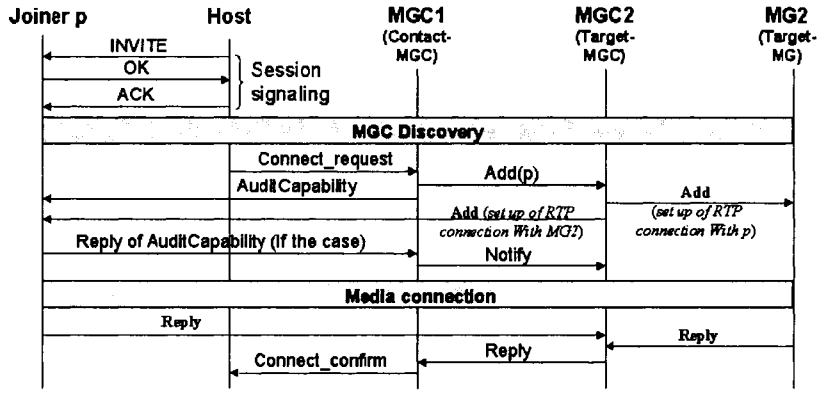

Figure 5.2. Connecting a joiner to the conference

5.2.2. Leaving a conference

A node may dynamically join or leave a conference. There are three types of nodes in the network determined by their network roles: simple nodes, MGs enabled and MGCs enabled. Upon detecting a Disconnect-request primitive, a contact-MGC determines the type of the leaving node and acts accordingly. The three types of nodes and the corresponding leaving procedures are described next.

a) Leaving of a simple node: If the leaving node is a simple node, the contact-MGC checks if it is connected to an MG that it controls. If it is not, then the contact-MGC multicasts a Subtract command to all the MGCs in the network. Upon receiving the command, each target-MGC determines if the leaving node is connected to a target-MG that it controls. When it is identified, the target-MGC sends a Subtract command to the target-MG where a departing Termination is located. From here, the target-MG simply uses the normal Megaco/H.248 departing procedure to clear the call and release the allocated resources. Figure 5.3 shows the message sequences required to disconnect a leaving simple node.

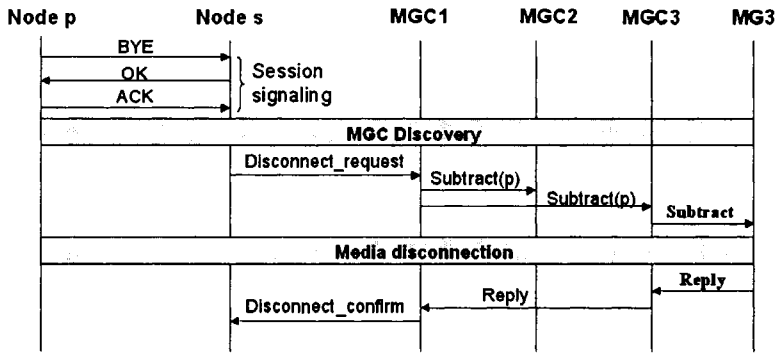

Figure 5.3. Disconnecting a simple node

b) Leaving of an MG: If the leaving node is MG-enabled then all the connected nodes must be disconnected and re-connected to other MGs. Therefore, the Contact-MGC sends a Subtract command to the affiliated MGC of the leaving-MG. Then, for each node connected to the leaving-MG, the affiliated MGC disconnects the node and, similar to the joining node process, assigns an MG and re-connects the node. Finally, the affiliated-MGC disconnects the leaving-MG from the other MGs and notifies the other MGCs that the event has occurred.

In Figure 5.4, we depict the use case of an MG leaving (MG1). The contact-MGC (MGC1) begins by sending a Subtract command to node *p* and MGC2 (MGC1 should disconnect MG1 from all of its related nodes and all of the MGs). Then, MGC2 sends a Subtract command to MG2 to disconnect MG1. Upon receiving a reply of the successful disconnection of MG1, MGC1 sends an Add command to MGC2 to add the node *p* (*p* will be connected to MG2).

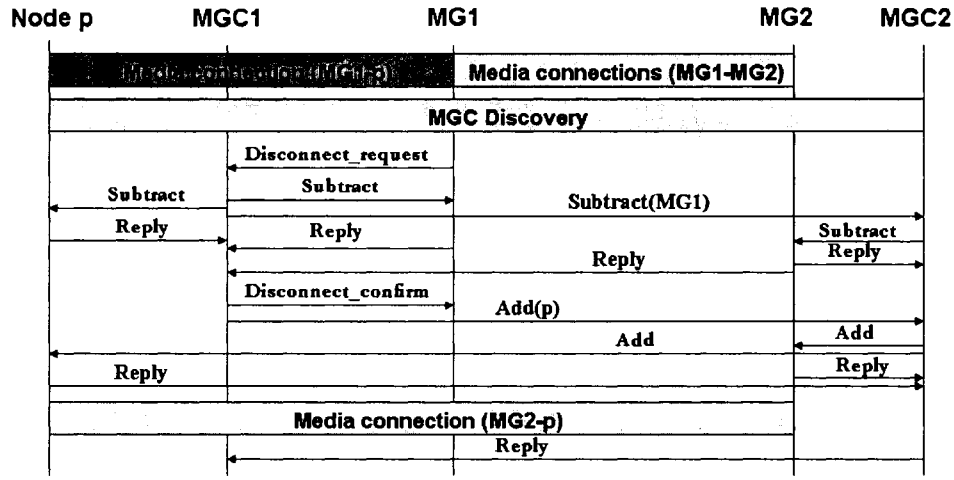

Figure 5.4. Leaving of an MG

c) **Leaving of an MGC:** Each MG controlled by the leaving-MGC is assigned to an enabled MGC (or a new MGC is enabled if necessary) by the contact-MGC (Figure 5.5). Next, the contact-MGC (MGC1) notifies the other MGCs about the MGC leaving (MGC2). Finally, the contact-MGC disconnects the media connection of the leaving MGC (MGC2), as in the case when a simple node leaves.

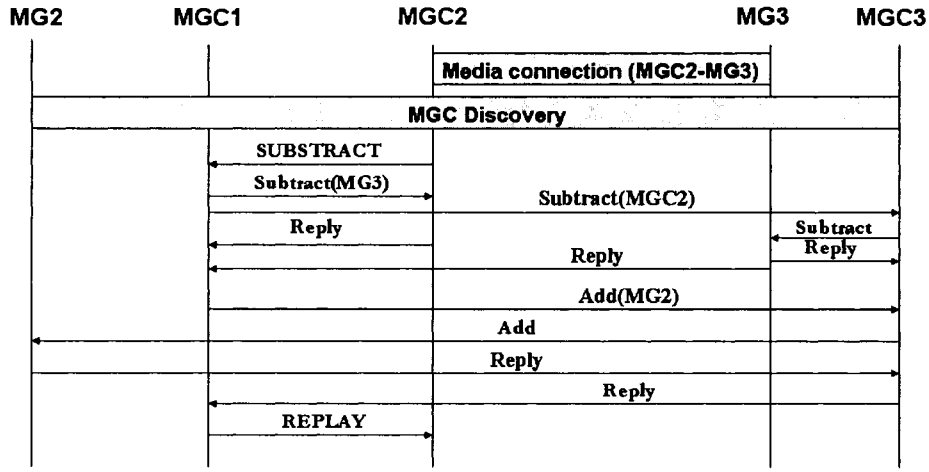

Figure 5.5. Leaving of an MGC

5.2.3. Enabling an MG

When a disabled MG receives an Add command from an MGC, it becomes enabled. It creates a local Context related to the conference, and then adds the corresponding Termination specified in the command to the local Context. The MGC sets up all the necessary media connections with the other MGs with a set of Add commands.

In Figure 5.6, we depict the use case of adding a joiner that enables a new MG (MG2). The contact-MGC (MGC1) begins by sending an Add command to MGC2, which simultaneously sends Add commands to MG2 and MG1 to connect to each other (MGC2 should connect MG2 to all the enabled MGs). Upon receiving a reply about the successful enabling of MG2, MGC1 sends an Add command to MGC2 to add the joiner.

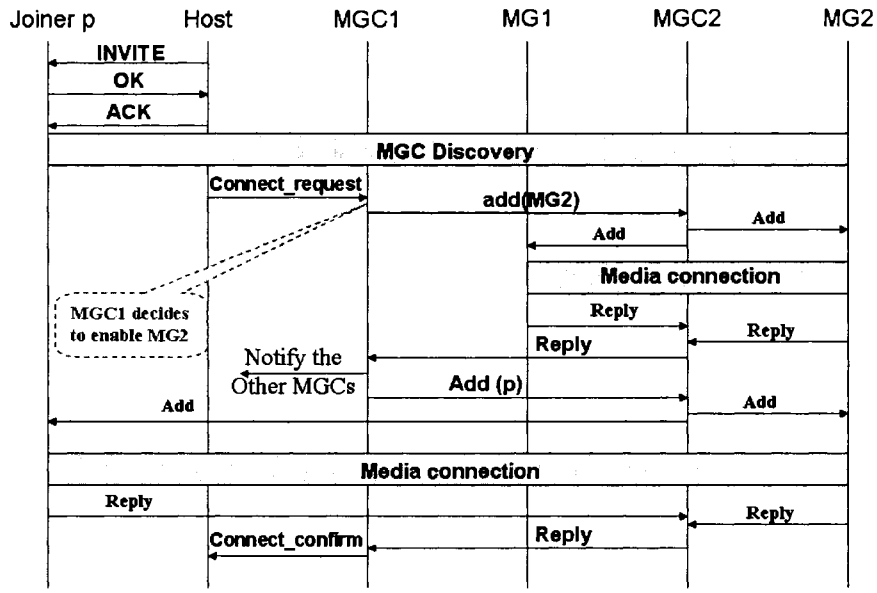

Figure 5.6. Enabling a new MG

5.2.4. Disabling an MG

When the last connected node leaves an MG, then the affiliated MGC (or a contact-MGC) may decide to disable it by applying a decision algorithm. If the decision to disable is made, the affiliated MGC selects an MG to maintain the connection with the disabled MG and disconnects it from the other MGs. Finally, the affiliated MGC notifies the other MGCs about the event. Figure 5.7 shows the message sequences to disable an MG when its last node leaves. In this figure, MGC-1 selects MG2 to maintain the connection with the disable MG (MG1).

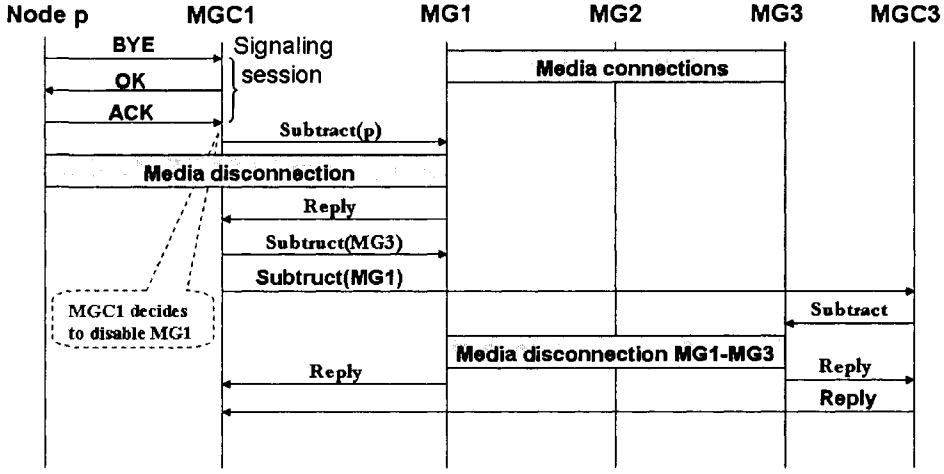

Figure 5.7. Disabling an MG

5.2.5. Enabling an MGC

When a disabled target-MGC receives an Add command from a sender-MGC, it becomes enabled and begins controlling the target-MG specified in the command. If the target-MG is not enabled, then the target-MGC enables it as shown above. Next, the target-MGC replies to the sender-MGC, which in turn notifies the other MGCs. Note that the Add command includes the shared information that should be updated as the conference progresses.

In figure 5.8, we depict the use case of adding a joiner that enables a new MG, which in turn enables a new MGC. The contact-MGC (MGC-1) begins by sending an Add command to MGC-2, which simultaneously sends Add commands to MG-2 and MG-1 to connect to each other (MGC-2 should connect MG-2 to all the enabled MGs). Upon receiving a reply of the success of enabling MGC-2 and MG-2, MGC-1 sends an Add command to MGC-2 to add the joiner.

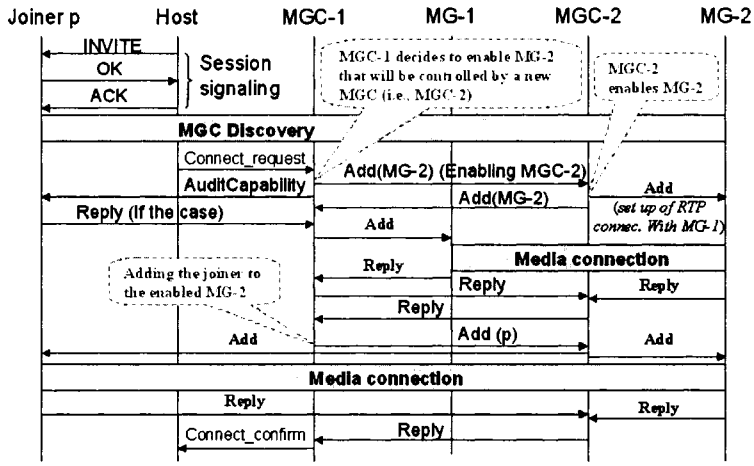

Figure 5.8. Enabling a new MGC and a new MG

5.2.6. Disabling an MGC

When the last connected MG leaves an MGC, then this MGC may stop acting as a controller. The MGC applies a decision algorithm to determine if it will disable. If the decision is made to disable then it stops acting as a controller and notifies the other MGCs.

5.2.7. Recovering an MG

The Notify command is used for failure detection messages to sense the MG live status. When an MG is detected as failed by a contact MGC (it can be its affiliated MGC or another one),

this MGC multicasts a ServiceChange command to all the MGCs. Upon receiving the command, each MGC updates the shared information. The affiliated MGC of the failed MG, using the local information, tries to re-connect each failed-MG related node using the joining procedure.

5.2.8. Recovering an MGC

The MGCs use Notify commands as failure detection messages to detect each other's failures. When an MGC fails, a contact-MGC (e.g. the first MGC that detects the event) multicasts a ServiceChange command to all the MGCs. Next, based on the shared information about the connected MGs of the failed-MGC, the contact MGC proceeds to assign and connect each MG to an MGC.

5.3. Summary

Conferencing in MANETs requires a media signaling architecture for managing media connections and controlling media mixers and controllers. We have proposed a Megaco/H.248-based architecture to manage and control the conference for media handling purposes. We then presented the architecture in terms of principles, primitives, events, messages and procedures. In this chapter, we have focused only on the architecture that defines the exchange of messages between different entities in the system. The self-organizing aspects that are behind the decisions for growing and shrinking and the rules of detecting failures are solved in the next chapter.

CHAPTER 6

SELF-ORGANIZING

Self-organizing media handling systems automatically enable and disable controllers and mixers when the network grows and shrinks, with no external authority. They also indicate to conference joiners which mixers they should connect to. To accomplish this, a self-organizing system has to include a decision scheme, such as to decide when to enable or disable a controller or a mixer. Finally, self-organizing systems include schemes to recover from failures.

As we have stated in the requirements specific to self-organizing for media handling in MANETs, the latter should preserve efficient resource usage during the conference, be of low complexity, and not significantly increase the number and length of exchanged messages in the network. Moreover, as the conference progresses, the number of nodes in the network will increase considerably and then shrink, all while the number of mixers and controllers should remain stable. This stabilization is essential to keep the number of exchanged messages low when the conference grows and shrinks.

We divide the self-organizing scheme for media handling for conferencing into three main components: self-growing, self-shrinking, and self-healing. Self-growing and self-shrinking are based on workload balancing schemes that provide decisions for growing and shrinking while respecting the efficient usage of resources. To conform to the decentralization requirement, these components use the shared information, which is decentralized information, to support

---

Results from this chapter have been published in [93], [94], and [96].

decentralized decisions. Self-healing is based on a failure detection architecture to detect failed nodes, and it employs a recovery scheme that uses the shared and local information kept in each controller (see two-overlay architecture nodal aspects) to stabilize the network.

This chapter is divided into four sections. The first section introduces the self-growing component, which is based on a novel workload-balancing scheme for growing. The second section presents the self-shrinking component that is also based on a novel workload balancing scheme -- for shrinking. The third section explores the self-healing component, including a novel failure detection architecture and a recovery scheme. The fourth section summarizes the chapter.

6.1. Self-growing

When nodes join the conference, the network grows according to the two-overlay architecture topology described in chapter 4. Figure 6.1 describes the growth of a conference. The first image describes a conference with three nodes. A joiner is connected to the mixer in the second image. In the third image, the joiner triggers the enabling of a new mixer. In the fourth image, the joiner is connected to the newly enabled mixer. The last image shows a joiner that triggers the enabling of a new mixer, which in turn triggers the enabling of a new controller.

To make the growth autonomic and automatic, we propose a self-growth mechanism based on a novel workload-balancing scheme that ensures an efficient usage of the mixing and control resources in the network. The scheme balances the control workload among the controllers and the mixing workload among the mixers, using their capabilities as criteria.

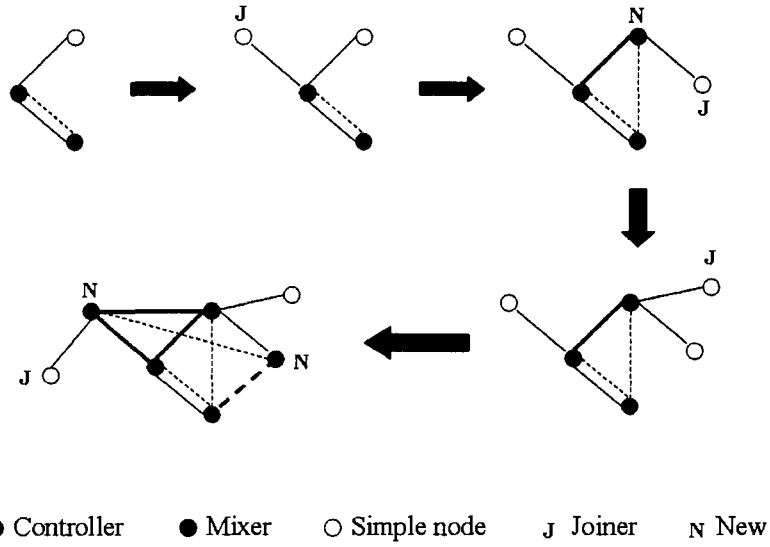

Figure 6.1. Network growth

Since the two overlays have the same topology, we apply the same workload-balancing scheme for growing to the two overlay networks. Then, we focus our analysis on the first overlay network composed of mixers and their related nodes. The same reasoning is applicable for controllers and their controlled mixers. In the rest of this section, we first study the workload-balancing scheme for growing by presenting the problem statement, and then proposing a solution and studying its complexity. Next, we present the system decisions for growing.

6.1.1. Workload balancing scheme for growing

To use resources efficiently, it is important to find an optimized network structure whenever a node joins the network. This best structure is found by solving an optimization problem. The problem's objective is to balance the workload among the mixers, based on their capabilities. In the following subsections, we first introduce the notation. Then, we present the problem statement and solution with a complexity study.

6.1.1.1. Notation

We denote by P the set of nodes and n their number. M denotes the subset of mixers and m their number. The capability of node p, C_p, is defined as the maximum number of nodes that can be connected to p for mixing purposes. Or, C_p expresses the number of nodes that p can serve as a mixer. The degree of node p, D_p, represents the number of actual nodes that are connected to p for mixing purposes. The ratio $w_p = D_p/C_p$ constitutes the unique performance measure by which we characterize the workload of node p. In practice, this workload has to be expressed in terms of the usability of computation capability, network link capacity, device energy, etc. Table 6.1 summarizes the notation meaning.

Notation	Meaning
P	set of nodes
M	set of mixers
$P \setminus M$	set of simple nodes
n	number of nodes
m	number of mixers
C_p	number of nodes that node p can serve as a mixer
D_p	number of actual nodes connected to node p
$w_p = D_p/C_p$	workload of node p

Table 6.1. Notation meaning

6.1.1.2. Problem statement

There are four events that can affect the workload of the mixers in the network: a node joins the session, a node leaves the session, a mixer is enabled, and a mixer is disabled. To use resources efficiently, it is important to find an optimized network structure whenever any of these events happen. The target structure is determined by solving an optimization problem, with the objective of balancing the workload among the mixers, using the nodes' capabilities as criteria.

To balance the workload, we successively consider three events: node joining, mixer enabling and mixer disabling. The node-leaving event can be disregarded, based on the assumptions we have made on the system, because it is impractical to re-organize the structure of the network after simple nodes leave.

6.1.1.3. Problem solution

Our problem of allocating nodes to mixers is fairly similar to an optimization problem of maximizing the attendance of joiners and where the constraints are the capabilities of mixers. We proposed a preliminary workload-balancing scheme that assigns joiners to the more powerful mixers. However, we have proven that this does not provide the best performance at high loads. We demonstrate that it is beneficial to assign joiners to the more powerful mixers at low loads, while at high loads, it is better to keep the mixers more balanced.

6.1.1.3.1. Balancing when connecting a joiner

There are two ways to connect a joiner to the network of mixers: connect the joiner to an existing mixer or enable a new mixer and connect the joiner to it.

a. Connecting the joiner to an existing mixer

With an obvious strategy, an optimized structure is obtained if we connect the joiner to the mixer that has the lowest workload. Hence,

$$\min \{w_i\}\ i \in M \quad \text{where } (D_i + 1)/C_i \leq 1 \qquad [6.1]$$

Formula [6.1] expresses the equilibrium among nodes without taking into account the mixers' capabilities. To take into account the capabilities of mixers, then, for the same workload, it is better to add the joiner to the mixer that has the highest available capability.

As we see in Figure 6.2, all mixers have the same workload value (w = number of connections by mixer capability); but, it is better to connect the joiner to the mixer who has the capability rated as '6', since the difference in the workloads of mixers in this state is less than that in the alternate state where the joiner is added to the mixer with capability three.

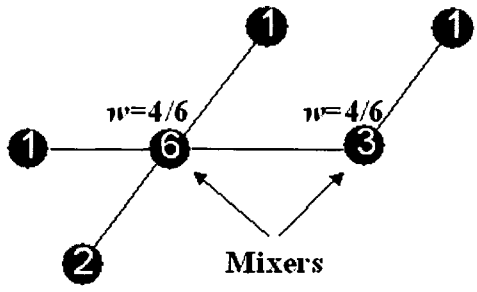

Figure 6.2. Example of a balanced network

To take this case into account, we compare the workloads of the mixers after adding the workload induced by the joiner. We denote w_i' the workload of mixer i after adding the

workload of the joiner. Then, we minimize w_i' where i belongs to M. However, w_i' is equal to $(D_i+1)/C_i$ and w_i is equal to D_i/C_i. Hence, the expression becomes:

$$\begin{array}{l} \min \ \{w_i + (1/C_i)\} \ i \in M \\ where \ \ (D_i + 1)/C_i \le 1 \end{array} \quad [6.2]$$

We enhance Expression [6.2] by keeping mixers more balanced at high loads. Then, nodes are distributed equally among mixers depending of their mixers' degrees. We define high loads when all the mixers reach 50% of their capabilities. Hence, the expression at high loads becomes:

$$\begin{array}{l} \min \ \{D_i\} \ i \in M \\ where \ \ (D_i + 1)/C_i \le 1 \end{array} \quad [6.3]$$

Then, in high loads, Expressions [6.3] will be used instead of Expression [6.2].

b. Connecting the joiner to a new mixer

In this case, we should take into account that the new mixer should connect with $(m-1)$ mixers (there is one connection that already exists which relates the newly enabled mixer with its previous dedicated mixer) and then, we connect the joiner to the enabled mixer. Hence,

$$\begin{array}{l} \min \ \{w_i + (m/C_i)\} \ i \in P \setminus M \\ where \ \ (D_i + m)/C_i \le 1 \end{array} \quad [6.4]$$

Continued enabling of mixers at high loads will lead to a deadlock because of the overload of connections that will be made between mixers. So, at high loads, the focus of the scheme will become giving priority to the distribution of the joiners among the existing mixers.

6.1.1.3.2. Complexity study

When connecting a joiner, the optimization of the system is assured by Expressions [6.2] and [6.4]. The complexity is therefore $O(n)$. To reduce this complexity we introduce the theorem below.

Theorem: Given -- a structure s and a joiner p. If we have to enable a new mixer, then Expression [6.4] is minimized by selecting the disabled node that has the highest capability.

Proof: Let disabled node u be the node with the highest resource capability C_u. Let disabled node v be less powerful than node u with resource capability C_v, where $C_v < C_u$. So, we have $C_v < C_u$ that it is equal to $1/C_u < 1/C_v$. However, $(1+m)$ is greater than zero, so the expression is equivalent to $(1+m)/C_u < (1+m)/C_v$. Hence, the expression becomes: $(1/C_u)+(m/C_u) < (1/C_v)+(m/C_v)$, and then: $w_u+(m/C_u) < w_v+(m/C_v)$ because $D_u = D_v = 1$. So node u minimizes Expression [6.4] and has the smallest workload after the connection of a joiner.

By the previous theorem, we demonstrate that to enable a new mixer, we do not have to evaluate all of the disabled nodes in the network. The smallest workload is guaranteed by choosing the disabled node that has the highest capability. Then, the complexity to connect a joiner by resolving Expressions [6.2] and [6.4] becomes $O(m)$.

6.1.2. System decisions for growing

The system's decisions should respect the efficient use of network resources by preserving workload equilibrium among mixers for each decision made. Figure 6.3 depicts three system decisions relating to a node-joining event.

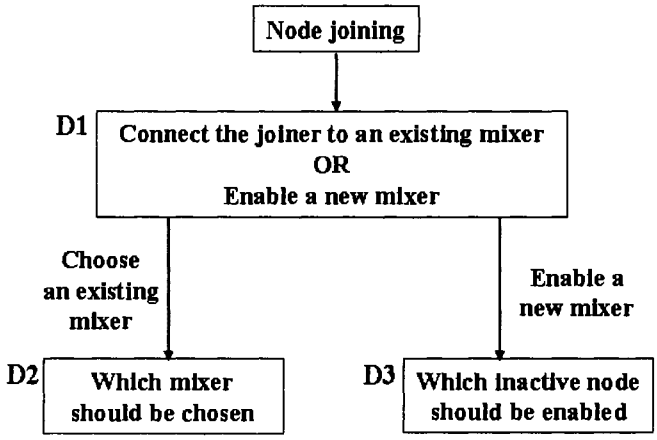

Figure 6.3. Decisions relating to a node joining

Based on the proposed workload-balancing scheme we provide the criteria for each decision as follows.

a. Enable a new mixer or not? (D1)

Based on the resource efficient scheme described above, the criterion to enable a new mixer can be described as: If enabling a new mixer balances the network more than choosing one of the existing mixers then we enable a new mixer - otherwise we choose an existing mixer. A typical scenario is described next.

When a joiner is invited to the conference and then has to be connected to a mixer, the host first discovers, and sends a Connect_request primitive to, a contact controller. To determine if the joiner should be connected to an existing mixer or to a new mixer, the contact controller evaluates Expressions [6.2] and [6.4] (At high loads only Expression [6.3] is evaluated to determine an existing mixer to connect the joiner to). If Expression [6.4] minimizes the workload more than Expression [6.2] for a disabled node, denoted by m_h (e.g. controlled by the contact controller), then the contact controller sends an 'Add' message with a list of existing

mixers to node m_h requesting it to enable mixing. Subsequently, node m_h enables as a mixer, establishes media connections with current mixers and then sends a confirmation to the contact controller. Finally, upon receiving the confirmation, the contact controller sends a message to the new mixer m_h requesting it to connect the joiner. If the contact controller does not receive a confirmation from mixer m_h after a given time, for example (e.g. mixer m_h may be out-of-range), then it repeats the process described above to reconnect the joiner.

Figure 6.4 illustrates the specific scenario where a new mixer, *Mixer-2*, is enabled, but suddenly fails. We assume that Expression [6.4] minimizes the workload more than Expression [6.2], which is why the new mixers are enabled. It should be noted that the new mixers establish media connections with all the current mixers, although the figure shows only the media connection with *Mixer-1*.

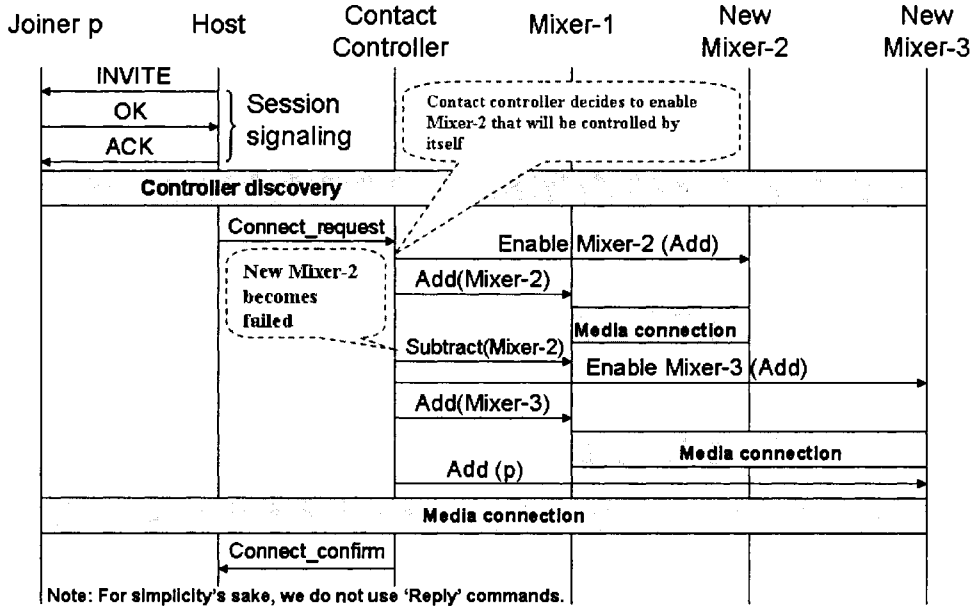

Figure 6.4. Enabling an alternate new mixer when a newly enabled mixer suddenly fails

b. Which existing mixer should be chosen ? (D2)

As presented above in the workload-balancing scheme for growing, adding the joiner to the existing mixer that has the lowest workload provides the best structure. Therefore, the contact controller chooses the mixer that minimizes the workload in the network by evaluating Expression [6.2] (or Expression [6.3] at high loads). When the contact controller selects an existing mixer, denoted by m_i (e.g. controlled by the contact controller), it sends it a message to connect the joiner. Upon receiving this message, mixer m_i establishes a media connection with the joiner. Finally, mixer m_i sends confirmation of adding the joiner to the contact controller.

c. Which disabled node should be enabled ? (D3)

Instead of comparing all of the disabled nodes to compute the lowest workload, the disabled node that has the greatest capabilities guarantees the smallest workload. Furthermore, the choice of the most powerful disabled node certifies the best path for expansion of the network since each new mixer is chosen for its strength. Consequently, controllers only exchange and maintain updated information about powerful disabled nodes in the network.

6.2. Self-shrinking

When nodes leave, the network shrinks. The leaving nodes can have an impact on the network depending on their roles: simple nodes, mixers or controllers. When a simple node leaves, there is no re-arrangement required, even though this event can unbalance the workload among the mixers. Reorganization of media connections is time costly. A simple node-leaving event may trigger a disabling mixer event that in turn may trigger a disabling controller event. When a mixer leaves the conference, it may also trigger a disabling

controller event. Figure 6.5 describes these different events by examples. We can see in the second image that a disabling mixer leads to a disabling controller in the third picture. The fourth picture describes a leaving mixer, in which all of its related nodes will be re-connected to the other mixer (in the fifth picture).

We apply the same workload-balancing scheme for shrinking to the two overlay networks, since they have the same topology. Subsequently, we focus our analysis on the first overlay network composed of mixers and their related nodes. The same reasoning is applicable for controllers and their controlled mixers. In the rest of this section, we first study the workload-balancing scheme for shrinking, and then present the system decisions for shrinking.

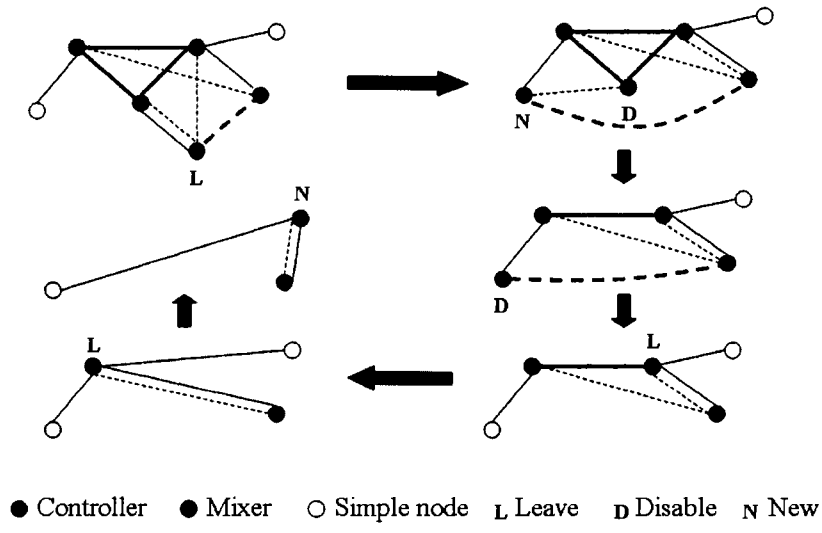

Figure 6.5. Network shrinkage

6.2.1. Workload balancing scheme for shrinking

We use the same notation as used in the subsection for the workload balancing scheme for growing described above (subsection 6.1.1.1).

6.2.1.1. Problem statement

When a mixer has no simple node to serve, it can be disabled by a contact controller (it can be its controller or another one). In fact, a mixer is disabled if its workload is low compared to the powerful disabled mixer in the network. This allows for large scalability since the less powerful mixers are replaced by more powerful mixers, which increases the overall capability while the conference progresses. The disabled mixer remains in the conference as a simple node and needs to be served by another mixer. The structure of the network should remain balanced after it disables. The disabled mixer becomes a potential joiner after disconnecting from other mixers. A mixer has to be selected to reconnect with the disabled mixer/joiner.

6.2.1.2. Problem solution

The chosen mixer that should reconnect with a disabled mixer should minimize Expression [6.2] after removing the connection of the disabling mixer and adding it again as a joiner. Thus, we minimize $(w_i - (1/P_i)) + 1/P_i$, which is equal to w_i, so that

$$\begin{array}{l} \min \ \{w_i\}\ i \in M \\ where \quad D_i \geq 1 \end{array} \qquad [6.5]$$

The complexity of the algorithm implementing Expression [6.5] is $O(m)$. Along the same lines, controllers can be disabled if they have no mixers to serve. It is better to leave a powerful controller enabled without mixers because if it is disabled it will probably be re-

enabled after a short time. However, when a controller is disabled, a connection load is subtracted for each controller, preserving the workload-balance among controllers.

When a mixer has to leave, the contact controller tries to reconnect all the nodes served by that mixer, then disconnects the mixer and informs the other controllers. When a controller has to leave, it begins by re-connecting controlled mixers to other controllers, disables itself, and informs the other controllers.

6.2.2. System decisions for shrinking

A system decision relating to a node-leaving event is depicted in Figure 6.6.

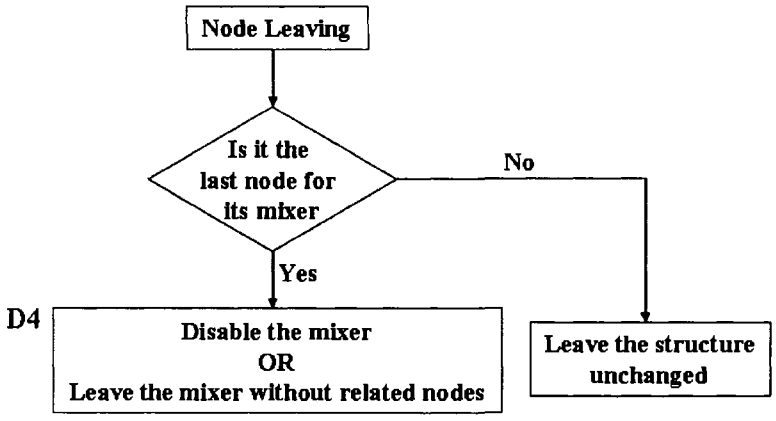

Figure 6.6. Decisions relating to a node leaving

a. Disable a mixer or not ? (D4)

When the last related node leaves a mixer, then it may stop being a mixer. It stops if its capabilities are less than those of the most powerful disabled node in the network. This condition is motivated by the fact that the disabled mixer can be re-enabled immediately if it is

more powerful than the most powerful disabled node. On the other hand, disabling a mixer minimizes the number of connections in the network and does not disturb communication with the disabled mixer. A usual scenario is described next.

When a mixer, denoted by m_s, has to be disabled, the contact controller (the controller that controlled the leaving operation of the last node of mixer m_s) should select one mixer to stay connected to mixer m_s, and disconnect it from the other mixers. Then, the contact controller evaluates Expression [6.5], which provides the mixer, denoted by m_t, to stay connected to mixer m_s. Subsequently, the contact controller sends a message to the other mixers (directly if they are controlled by itself or via their controllers) to inform them that mixer m_s will disable itself and disconnect from them (except for mixer m_t).

6.3. Self-healing

A self-organizing system for conferencing includes a self-healing component. Self-healing in media handling assures re-establishment of media connections in case of failures. In our system, we distinguish critical failures, such as controllers' and mixers' failures. Simple nodes' failures are discarded because no other nodes rely on them. Connection failures are presumed non-existent. The controllers have the responsibility of detecting failures and recovery of the network if necessary. Self-healing is supported by a failure detection architecture and a recovery scheme presented in the following subsections.

6.3.1. Failure detection architecture

As we have shown in the related work chapter, a heartbeating approach is appropriate for MANETs. Therefore, we propose an overlay architecture using heartbeating for detecting

failed nodes in MANETs. In the rest of this subsection, the failure detection architecture is presented in terms of assumptions, principles and procedures.

6.3.1.1. Architectural assumptions

The system we consider is a MANET composed of a set of nodes. The failure detection procedure is applied for a group of nodes in the network, called the critical group, which can be all of the nodes in the network or a sub-group of nodes. The critical group should be informed when one of its nodes fails. The nodes are all assigned with unique identifiers (*IDs*). We assume that node failure is permanent, or equivalently, that a node that recovers from a failure will get a new identity (*ID*) as a new joiner. Temporary out-of-range is not considered as a failure if it does not exceed a timeout fixed by the application requirements. This timeout, denoted by *TH*, is the frequency of heartbeat messages.

Communications in MANETs are inherently unreliable: messages may be delayed or dropped by multi-hop links. However, we assume that in most cases, messages are delivered with some propagation delay, within a reasonable time, denoted by *TD*, as specified by the application. We note that *TH* is much longer than *TD*. Finally, we assume that message loss is the only factor that leads nodes to make a false detection or not report a failure.

We chose multicasting to report failure information to the critical group. Multicasting reduces the number of messages and provides a reasonable delivery delay [86]. Moreover, it can be built to be reliable, and specifically, to counter the vulnerability of message loss in MANETs (e.g. using broadcasting). In the remainder of this section, for simplicity's sake, we only use one critical group, which is all of the nodes in the network.

6.3.1.2. Architectural principles

We overlayed the physical network with a self-organizing two-level overlay network (Figure 6.7). The first level is a flat structure of detectors that form a sub-network. The detectors periodically send heartbeat messages to each other. Nodes that do not act as detectors are part of the second level. Each node in the second level periodically sends a heartbeat message to a detector in the first level. Each detector senses a set of second-level nodes, called detector's affiliated nodes. Overlay network-formation and detector-assignment are assured by the self-organizing for the failure detection scheme presented next. Every detector incorporates a logical clock (a counter), which is included as a timestamp in all sent messages. The clock is incremented by the events of sending and receiving heartbeat messages. When a detector sends heartbeats to all the other detectors, it is treated as one event. It is the same for receiving -- it is considered one event when all the heartbeat messages arrive with the same timestamp. The timestamp is used to distinguish the heartbeat messages from the previous period, which arrived after time *TD*, from the current heartbeat messages.

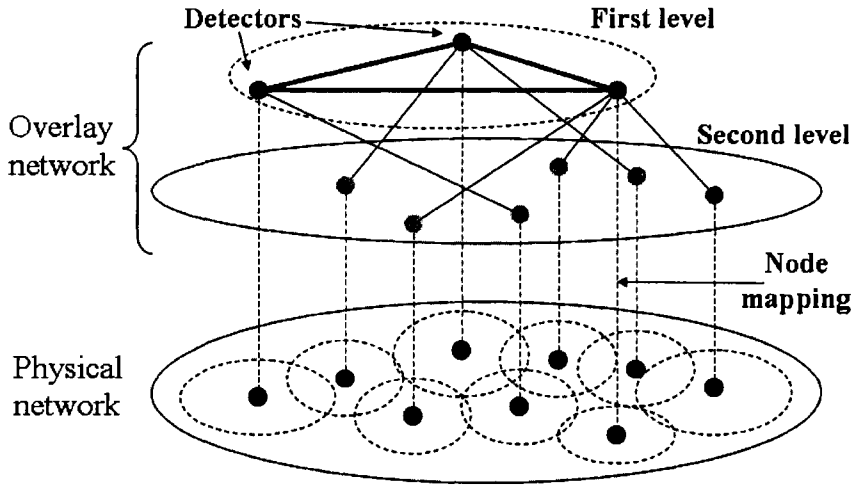

Figure 6.7. Overlay-based failure detection architecture

Only detectors are responsible for failure detection and for reporting that information to all of the nodes of the critical group. Detectors identify failed nodes according to the failure rules described below:

- *Affiliated node failure detection rule.* When a detector does not receive a heartbeat message from one of its affiliated nodes, it sends it a 'warning' message. If this node is still alive and receives the warning message, it sends a 'correct' message to its detector to specify that it is alive. After a timeout, if the detector of this node does not receive either the 'correct' message or the delayed heartbeat (if any), it multicasts a 'failed' message to the critical group.

- *Detector failure detection rule.* When a detector does not receive a heartbeat message from one detector, it sends a 'warning' message to all the detectors. If the detector in question is still alive and receives the warning message, it sends a 'correct' message to all of the detectors. A detector is declared failed by another detector if and only if neither a heartbeat nor the 'correct' message has been received, and it has been sent at least one warning message. One of the detectors (e.g. the detector with the smaller *ID*) multicasts a 'failed' message. If one of the detectors does not receive the 'failed' message after a timeout, it re-multicasts the information about the failed node. If there are only two detectors in the network, they apply the affiliated node failure detection rule described above between them.

6.3.1.3. Procedures - Self-organizing for failure detection

Three events may change the network's formation: node joining, node leaving and node failing. For each event, nodes may exchange messages to respond to the new formation. The network begins its formation with two nodes, which are automatically assigned as detectors, and additional nodes can join subsequently. Every detector determines its service threshold, depending on its capabilities (processor power, memory capacity, link capacity, etc.), which specifies the maximum number of affiliated nodes that it can serve. The algorithm of computing the service threshold is not in the scope of this thesis. Every detector maintains the information of all of the detectors and of their affiliated nodes. This information is updated via the exchange of messages between the detectors.

The self-organizing scheme attempts to distribute the joining nodes among the detectors until the thresholds are reached in every detector. When all of the detectors are full, the next joining node will be assigned as a new detector. This formation is not affected when an affiliated node leaves, unless it is the last node of a detector. In this case, this detector stops being a detector, informs the other detectors and chooses the detector that has the highest threshold value to continue sending its heartbeats to.

In addition to the timestamp, the heartbeat message includes three bits: *DH*, *FH* and *LH*. The *DH* bit is used to specify if the sender is a detector or not. The *FH* and *LH* bits specify that it is the first and the last heartbeat message, respectively.

a. Node joining

When a node joins the network, it multicasts a JOIN message to inform all of the nodes of the critical group about its arrival. All the detectors reply with a BID message, which includes their

thresholds and their affiliated nodes' information. The joiner chooses the detector with the minimum load (number of affiliated nodes/threshold ratio) and starts sending heartbeats. Then, the chosen detector informs the other detectors to update their information. If the thresholds of all detectors have been reached, the joiner begins as a detector and starts sending heartbeats to all the detectors. Consequently, the detectors will determine if the joiner started as a detector by the *DH* bit in the first heartbeat message. To achieve maximum growth, we recommend that a joiner starts as a detector if its threshold is greater than the thresholds of the network's detectors. Note that detectors continue bidding after they have reached their thresholds since they have to pass on their information and their affiliated nodes' information to the joiner so that it will be informed about the nodes in the critical group.

b. Node leaving

If the leaving node is an affiliated node, it sets the *LH* bit to the 'true' value of its last heartbeat, multicasts a LEAVE message to inform all of the critical group nodes, and quits. When a detector is leaving, it begins by multicasting a LEAVE message to inform all of the nodes of the critical group. Upon receiving the multicast message, the affiliated nodes of the leaving detector have to repeat the joining procedure (see above). The leaving detector will control the joining operation of its affiliated nodes. Finally, it sets the *LH* bit to the 'true' value of its last heartbeat, and quits.

c. Node failing

If an affiliated node fails, no action is required since nodes in the second level have no obligation in the management of the network. We note that all of the critical group nodes have already been informed about the affiliated node failure (see the affiliated node failure detection

rule). In the case of a detector failure, its affiliated nodes are already informed about their detector's failure (see the detector failure detection rule). Therefore, these affiliated nodes become new joiners and then perform the node joining procedure described above. With the information about the affiliated nodes of the failed detector, one of the detectors (e.g. the detector with the following *ID*) will check if the joining operations are complete for all the affiliated nodes. If one of the affiliated nodes is missing, the detector will report it as failed.

6.3.2. Recovery scheme

We distinguish critical failures such as controllers' failures and mixers' failures. Simple nodes' failures are discarded because there are no other nodes that rely on them in the system. Failure recovery is supported by the shared and local information. All the controllers store and maintain the shared information, which consists of the list of the information about controllers and their mixers and information about potential mixers and controllers in the network. In addition, every controller stores the information about nodes that are served by mixers that it controls (the local information). The controllers are responsible for recovering the network if necessary.

6.3.2.1. Controller recovery

We applied our proposed failure detection architecture to the set of controllers, with controllers forming the critical group. If a controller fails, one of the controllers (e.g. the one that informs the other controllers about the failure), using the shared information, tries to recover the set of mixers controlled by the failed controller. It will repeat the mixer enabling procedure for each mixer (see subsection 5.2.3).

6.3.2.2. Mixer recovery

With the local information, a controller can reconnect the simple nodes served by its failed mixer. We applied our proposed failure detection architecture to the set of mixers. The set of mixers forms the critical group. When a detector-mixer detects a failure, it informs its controller (or discover a contact controller), which in turn informs all the controllers to update their shared information. Subsequently, the affiliated controller of the failed mixer tries to recover the set of simple nodes by repeating the joining procedure for each node (see subsection 5.2.1).

6.4. Summary

This chapter has presented a self-organizing scheme. It is divided in three parts, each presenting a component: self-growing, self-shrinking and self-healing. Workload-balancing schemes were presented as base for self-growing and self-shrinking schemes for supporting growing and shrinking decisions. Decision situations were presented and criteria determined. Enhancements were provided to reduce complexity and to allow a large number of users to attend the conference. The third part of the chapter presented a self-healing scheme to ensure that if a failure occurs, the network recovers. Self-healing is composed of two distinct components: a failure detection architecture and a recovery scheme. The failure detection architecture ensures that if a node fails, all of the other nodes in a critical group (e.g. group of controllers) will be notified. This architecture is generic, independent of lower layers and decentralized. The recovery scheme assures mixers' and controllers' recovery, using access to the shared and local information.

CHAPTER 7

MEDIA HANDLING ARCHITECTURE FOR MCNs

Multihop Cellular Networks (MCNs) are the result of the integration of 3G Cellular Networks (3GCNs) with MANETs. A media handling architecture for conferencing in MCNs allows conferences in which some users are in 3GCNs and some in MANETs. MANET media handling architecture is scalable, but it only supports conferencing when all of the users are in a MANET. On the other hand, as we have shown in the related work chapter, 3GCN media handling architecture cannot meet our requirements to handle media for conferencing in MCNs, especially because it cannot handle conferences with users located in a MANET. However, 3GCN and MANET media handling architectures are considered as the basis for our solution for media handling for conferencing in MCNs. The proposed architecture is based on Media Mediators (MMs) that will bridge 3GCNs with MANETs for the purpose of media handling.

This chapter is divided into three sections. The first section presents the proposed media handling architecture for MCNs. It begins with the architectural principles, and then describes specific extensions to 3GCNs so that they can interface with MANETs, followed by the functional entities. The second section illustrates the scenarios of session management for initiation, invitation and leaving cases. Finally, the last section gives a short summary.

Results from this chapter have been published in [97] and [98].

7.1. Media Handling Architecture for MCNs (MHAM)

3GCN users cannot be directly connected to a MANET conference for media handling because the 3G MRFP (the mixer in 3Gs) has no interface with MANET MGs (mixers in MANETs). It is of course possible to enhance the MANET media handling architecture and allow it to also support 3GCN users via direct connections with MANET MGs without passing through an MRFP. Unfortunately, this enhancement will considerably increase the throughput of a MANET since all the media streams of the 3GCN users will go through the MANET. Moreover, the intermediate nodes between the 3GCN and the MANET may be overloaded by the large number of streams. This enhancement is still not practical to support 3GCN users. However, the MANET media handling architecture can be enhanced by other means. A MANET can be seen as a set of sub-networks, each controlled by a Media Gateway Controller (MGC). A MANET can then view a 3GCN as a sub-network and thus part of its network. In this enhanced architecture, a limited number of media streams will be delivered to the MANET since they are mixed by the MRFP before delivery. Moreover, users' performance will be independent of their location in the network. Media streams of MANET users will not need to be sent to a distant MRFP. Enhancement of the MANET and 3GCN media handling architectures to permit an interface with each other is feasible and can potentially meet all the requirements. This section introduces the architectural principles and the extensions made to 3GCNs so they can interface with MANETs, and the functional entities of the proposed architecture.

7.1.1. Architectural principles

In practice, the integration of two different networks usually requires mediators that can communicate with the two networks. The mediator concept is practical because it leaves the two networks independent, so that the performance of one network is not affected by the network load of the other.

From the MANET point of view, MRFC (3G controller) and MRFP (3G mixer) entities in 3GCN can be seen as MGCs and MGs, respectively, because they have the same structure (a set of users are connected to a mixer that is controlled by one controller). The 3GCN can then be integrated with the MANET by a mediator that connects the two networks and translates the deployed protocols. However, MRFCs do not have interfaces with peer entities as per the 3GPP specification. The same is true for MRFPs that cannot interface together. To allow integration of the two networks for media handling purposes, MRFC and MRFP entities must have a horizontal interface to permit communication between peer entities.

As for the benefits of integration, these new interfaces will also allow the decentralization of conferencing in 3GCNs and thereby allow conferencing with several MRFCs and MRFPs in one or more 3GCNs. In fact, conferences can already be performed in more than one 3GCN. However, all the users need to connect to the same MRFC/MRFP, and thus the conference performance will decrease when additional users join. By fully decentralizing MRFC/MRFP networks, we resolve the scalability issue and permit interfacing with other types of networks. Consequently, we will extend 3GCNs and then describe the media handling architecture for MCNs in terms of functional entities.

7.1.2. Extended 3GCNs

In extended 3GCNs, MRFCs and MRFPs have horizontal interfaces that allow them to communicate with peer entities (Figure 7.1). The Mc reference point defines the interface between two MRFCs. This interface allows two MRFC peers to exchange messages to establish, control and end media connections between MRFPs. The media signaling protocol proposed in Chapter 5 can be used for the Mc interface between MRFC peers. Thus, it can be applied for extended 3GCNs in general by discarding the dynamic aspects. This media signaling protocol uses the same vocabulary and concepts that were used with Megaco/H.248 to define exchanged messages, and which are already used between MRFCs and MRFPs. For example, when an MRFC has to make a media connection between its MRFP and a target MRFP, it simply sends an Add command to the target MRFC (which controls the target MRFP), which in turn commands the target MRFP to add the media connection. Finally, reply messages are exchanged to finalize the operation.

The Mm interface allows two MRFP peers to provide resources for the media connections between them. In fact, this interface is used for media stream transportation (e.g. RTP packets) between MRFP peers. Therefore, in decentralized environments (more than one MRFP), the media mixing operation is different than that deployed on conventional 3GCNs. The two-step mixing operation proposed in Subsection 4.2.2 can be used for decentralized MRFPs. An illustration of how the mixing operation is done can be described as follows: users UE-1 and UE-2 send media streams to MRFP-1 and users UE-3 and UE-4 send media streams to MRFP-2 (Figure 7.1). Upon receiving entries, MRFP-1 immediately mixes the streams and sends the result to MRFP-2, which in turn does the same with its entries. When receiving the mixed result, MRFP-2 performs a first mix by mixing the entry with the stream of UE-3 and

sends the result to UE-4, and does a second mix by mixing the entry with the stream of UE-4 and sends that result to UE-3. MRFP-1 performs the same process with its users. Finally, the interface Mm can also be used between MRFPs, for the purpose of synchronization. The interface between MRFCs and MRFPs is defined by the Mp reference point. This interface remains the same as in the MRF architecture described in Chapter 3.

In the proposed extended 3GCNs, conferences can be made in several MRFC/MRFPs. Users are still connecting to their MRFC/MRFP in the same manner as in conventional 3GCNs. The messages exchanged over the interfaces Mc and Mp are only used to set up, modify and tear down media connections between MRFPs.

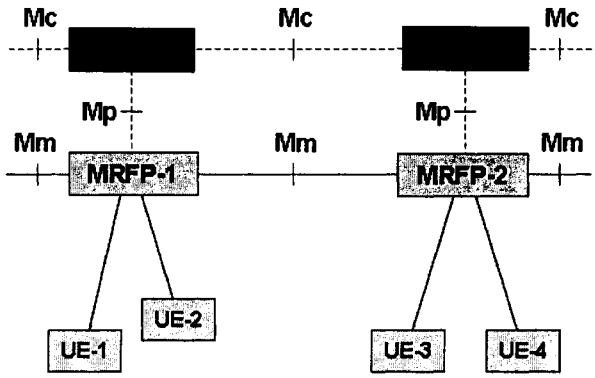

Figure 7.1. Extended 3GCNs

In this subsection, we discussed in detail the specific extensions required to allow integration between 3GCNs and MANETs. In the next subsection, we describe the proposed media handling architecture for MCNs in terms of functional entities.

7.1.3. Functional entities

Figure 7.2 describes the proposed media handling architecture for MCNs in terms of functional entities. We define a new entity between a 3GCN and a MANET called the Media

Mediator (MM). The MM assures the independence of 3GCNs from MANETs' topologies and protocols. It is composed of two entities: the Media Gateway Controller Mediator (MGCM) and the Media Gateway Mediator (MGM). The MGCM acts as a controller mediator between a 3GCN and a MANET. It will assure connectivity between the networks and act as a translator of different deployed protocols. The MGCM can be deployed on the 3GCN side, on the MANET side, or in a third party, and it is enabled only when the conference includes users in a 3GCN and a MANET. It is seen by MANET entities as an MGC that controls the MRFP via the MRFC. It has all the functionalities of a MANET MGC and can add MANET MGs in the Context of the conference in MRFP. From the 3GCN point of view, the MGCM entity is perceived by the MRFC as an MRFC peer with which it can communicate, via the Mc reference point. The MGM entity is necessary when MRFP and MANET MGs use different media stream transport protocols or when some specific media processing should be done when interfacing with MANETs. The interface between the MGCM and the MGM is still the same as that deployed between the MGCs and MGs of the MANET. When media processing is required, the MGCM enables the MGM and sets up the necessary media connections between the MRFP and the MGM, and between the MANET MGs and the MGM.

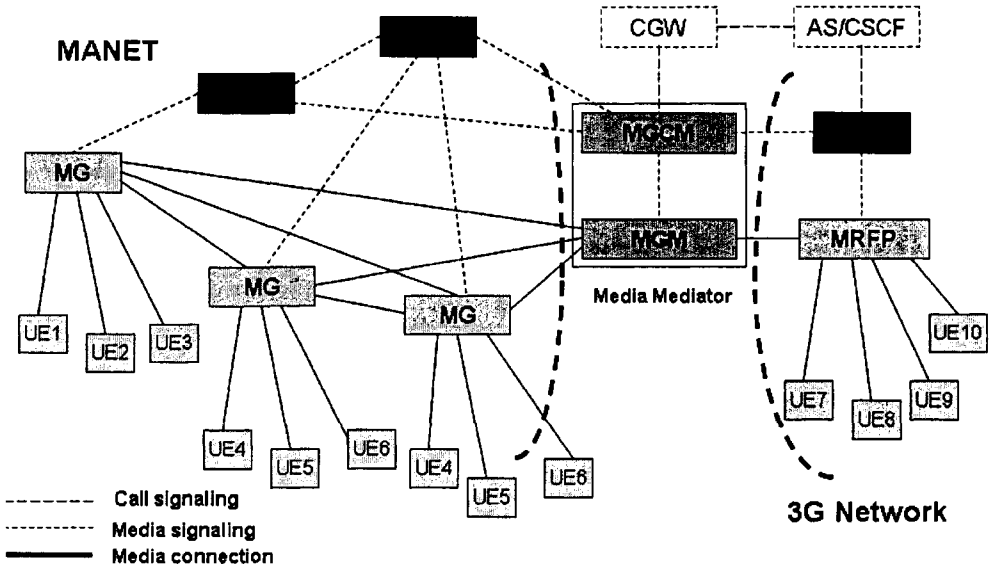

Figure 7.2. Media Handling Architecture for MCNs

The proposed architecture does not limit the topology of MANETs' MGs to a full-mesh structure as described in Figure 7.2. The topology of MGs-MGM sub network can have any kind of structure (e.g. hierarchic, tree). To set up call sessions, we can use any MCN call-signaling system, such as described in [88]. This call signaling uses a Conference Gateway (CGW) entity that handles call sessions for both sides. The CGW, acting as a mediator deployed by the 3GCN operator, has an infrastructure-based interface with the 3GCN and an interface with selected user agents in the MANET. It can comprehend signaling architectures and protocols, and performs the required mapping and translation. The CGW also provides publication and discovery services. It periodically sends its service advertisement to all the nodes, and a user that needs a service sends a discovery message through the network and waits for a response. Finally, the CGW provides registration functions and manages the repository of MANET users.

After call signaling is established between two users on two sides, the CGW must discover a media gateway controller to make the necessary media connections. The discovery functionality consists of the ability to find a suitable media gateway controller that can connect the joiner. In our case, the media gateway controller is usually the MGCM, but this is not the case if the network is composed of several MANETs with many mediators. Furthermore, the discovery concept preserves the independence between the call signaling protocol and the media handling protocol. The interface between the CGW and the MGCM is described by a set of primitives: create_session_request, create_session_confirm, connect_request, connect_confirm, disconnect_request, disconnect_confirm, modify_request and modify_confirm.

7.2. Scenarios for media handling session management

In our architecture, both 3GCN and MANET users are allowed to initiate the conference. 3GCN and MANET conferences are initiated, managed and torn down as in 3GPP [83] standard and MANETs, described in this thesis. In an MCN conference, some users are in a 3GCN and some are in a MANET.

Four scenarios are possible for starting an MCN conference: a 3GCN user initiates the conference with a MANET user, a MANET user initiates the conference with a 3GCN user, a 3GCN user invites a MANET user, and a MANET user invites a 3GCN user. Figure 7.3 depicts the MCN conference initiation sequence, described in the next subsection. Figure 7.4 depicts the MCN conference invitation sequence, and is described in the subsection after. The last subsection describes the MCN conference-leaving scenario.

7.2.1. MCN conference initiation

In Figure 7.3.a, 3GCN-User initiates an MCN conference with MANET-User. It first creates a 3GCN call session. Then the MRFC is triggered to create a new local Context and to add a new Termination to the local Context, and it returns the Local Connection Address to the 3GCN-User as in the 3GPP standard. After this step, 3GCN-User requests the creation of a call signaling session with MANET-User. Then, a call signaling session with the CGW is made, which in turn creates a call signaling session with MANET-User. Consequently, the CGW will discover MGCM, which is the gateway mediator to the MANET users, and then sends a Connect_request primitive to connect MANET-User to the conference. The MGCM will request MANET-User to start an MG for itself by creating a new local Context, adding a Termination for itself, and then connecting with the MRFP. Figure 7.3.b illustrates the case where a MANET user initiates a conference with a 3GCN user. In both cases, the MGCM translates protocols and maps the two networks. It acts as an MGC for MANET users and as a mediator for 3GCN users.

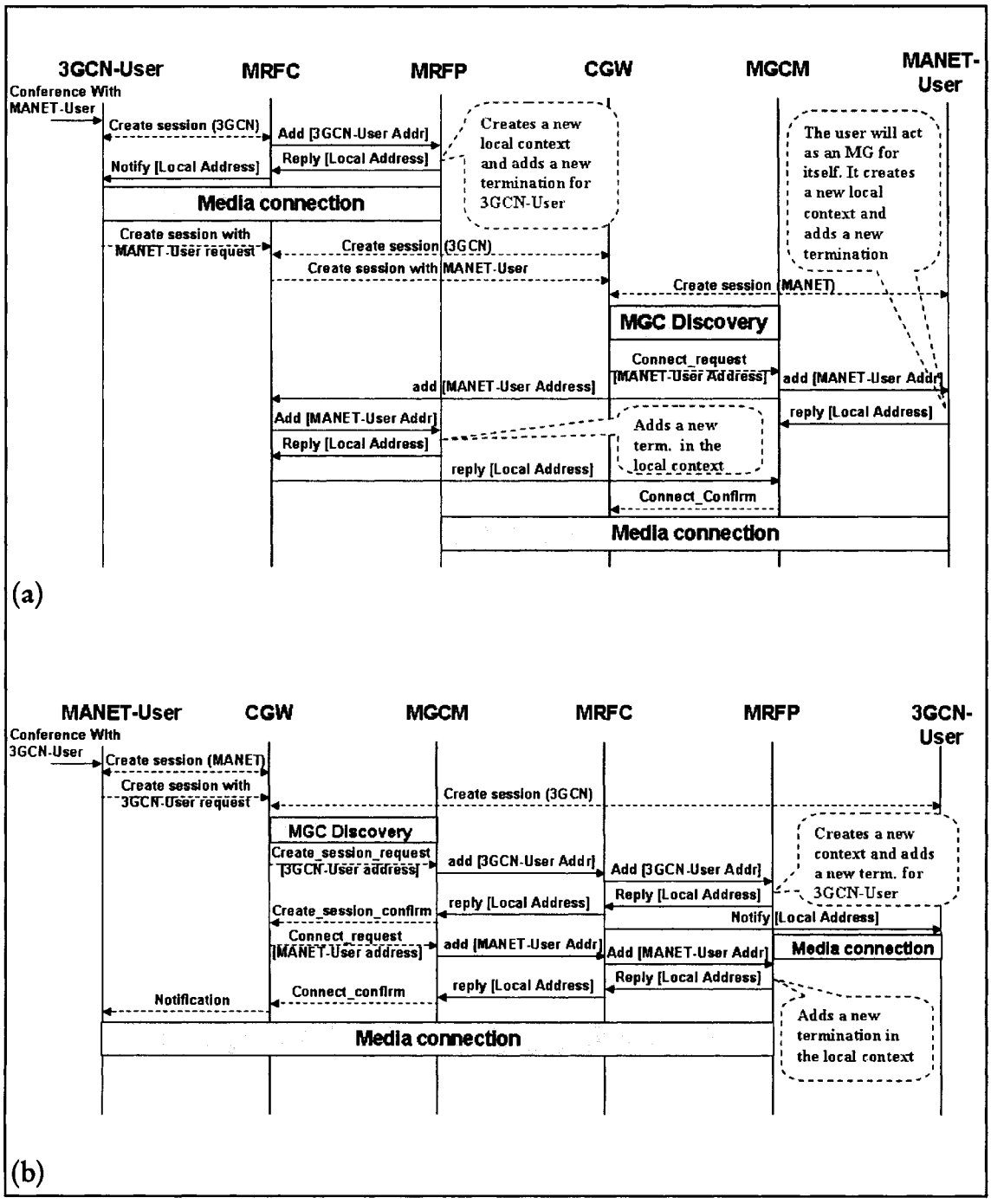

Figure 7.3. MCN conference initiation: a) a 3GCN user initiates the conference with a MANET user; b) a MANET user initiates the conference with a 3GCN user.

7.2.2. MCN conference invitation

3GCN and MANET users can join an existing MCN conference when they are invited. A 3GCN user that wants to invite a MANET user needs to first set up a call signaling session with it. Consequently, the MRFC will be triggered to add the MANET user. Figure 7.4.a illustrates the scenario of a 3GCN user inviting a MANET user, and Figure 7.4.b shows how a MANET user invites a 3GCN user. In the second scenario, MANET-User sets up the call signaling with 3GCN-User and then discovers an MGC to perform the Connect_request primitive to connect 3GCN-User to the conference. The discovered MGC will determine that 3GCN-User is in the 3GCN and then request the MGCM to add 3GCN-User to the conference. The MGCM will relay the request to the MRFC that will request the MRFP to add a new Termination for 3GCN-User in the local Context of the conference.

7.2.3. MCN conference leaving

Leaving 3GCN-users follow the scenarios defined in 3GPP [83] and leaving MANET-users follow the scenarios defined in MANETs presented in Chapter 5. Conference information should be propagated correctly between the two networks. When the last two users tear down the call session, all the local Contexts are deleted and the Media Mediator is torn down.

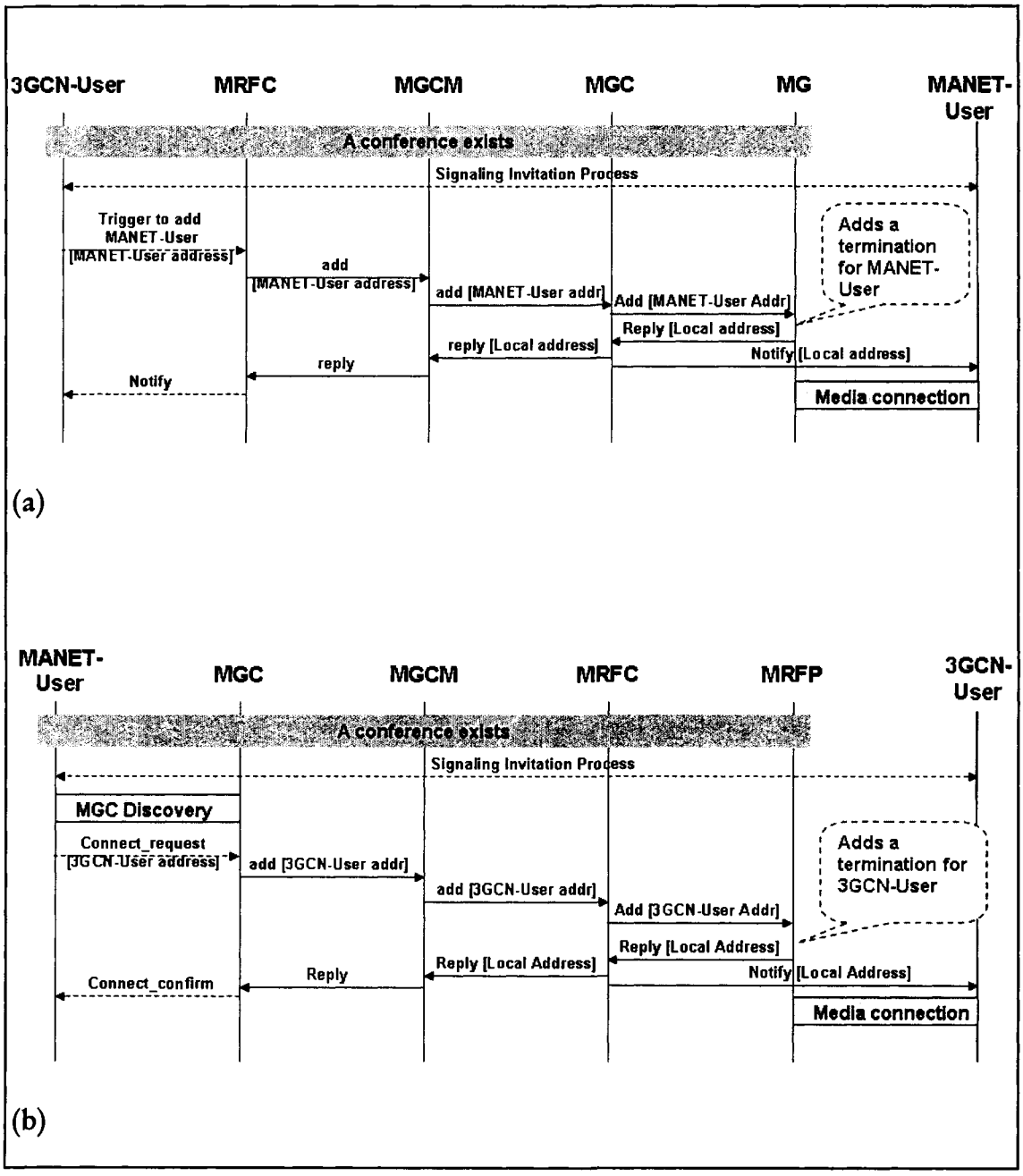

Figure 7.4. Procedure to add users in MCN conference: a) a 3GCN user invites a MANET user; b) a MANET user invites a 3GCN user.

7.3. Summary

This chapter has proposed a new architecture for media handling in MCNs. It extended the 3GCN media handling architecture so that it can interface with MANETs. Then, it used Media Mediators (MMs) to bridge MANETs and 3GCNs for media handling. The MM is composed of two functional entities: the Media Gateway Controller Mediator (MGCM) and the Media Gateway Mediator (MGM). Four scenarios were described for starting MCN conferences: when a 3GCN user initiates the conference, when a MANET user initiates the conference, when a 3GCN user invites a MANET user, and when a MANET user invites a 3GCN user. The leaving scenario is also described.

CHAPTER 8

IMPLEMENTATION AND EVALUATION

This chapter describes the proof-of-concept prototypes designed and implemented to evaluate the Distributed Mixing Architecture (DMA) and Media Handling Architecture for MCNs (MHAM). It also presents simulation experiments and the results of media signaling protocol, self-growing and self-shrinking schemes, and failure detection protocol.

This chapter is organized as follows. The first section presents the evaluation of the novel media mixing architecture (DMA). It describes the proof-of-concept prototype implementation environment and architecture, and the results from small- and medium-scale experiments, followed by simulation experiments and results. The second section describes the media signaling protocol simulations and results. The self-organizing evaluation is presented in the third section, using simulation experiments and results to demonstrate network scalability and resource efficiency, and is followed by experiments to show automatic stabilization when the network grows and shrinks, resource efficiency as the conference progresses, and performance of an enhancement made on the scheme. This section continues with the presentation of the results of failure detection protocol experiments. In the fourth section, the media mediator architecture is presented first, followed by a proof-of-concept prototype of the MHAM with experiments and results. The last section gives a short summary of the chapter.

Results from this chapter have been published in [92], [93], [94], [95], [96] and [97].

8.1. Distributed Mixing Architecture (DMA)

8.1.1. Prototype implementation environment and architecture

We have developed a proof-of-concept prototype of the distributed mixing architecture for an audio conferencing system, using Java as the programming language in an MS Windows environment. For media transportation we used the Real Time Protocol (RTP) [13], which is provided by the Java Media Framework API (JMF) [87]. Node-to-node physical connections are made up of broadband IEEE 802.11g wireless connections. For network routing, we used an implementation of AODV (WinAODV, Intel Corporation, V.0.1.14).

The prototype implements the two-level structure, in which mixers exchange media streams by multi-unicasting. Multi-unicasting is used for the sake of simplicity, when compared to the cost of implementing application-level multicasting. The execution environment requires a significant amount of computation time (especially the Java Virtual Machine and the JMF), making synchronization impossible. Subsequently, mixers process as follows: streams from related nodes are merged and immediately sent to other mixers, and then, only during the time required for buffering, they are merged with the streams from the other mixers and the results are sent to the related nodes. A simplified version of the self-organizing system is implemented. The control of mixers relies on a cluster-based signaling system for MANETs developed in our lab [89]. This clustering approach has the same structure as that of DMA. In this approach, the clusters are fully meshed -- each is ruled by a super-member. Each member has a signaling relation with the super-member of its cluster. The growth and shrinkage of the network are done by split and merge functionalities. When a split threshold is reached, a cluster is divided into clusters with the election of a new super-member. A reverse process is performed for merging, with a merge threshold value fixed in advance. In the implementation,

for simplicity's sake, it is assumed that all nodes have the same level of resources. Next, super-members are randomly selected. We map a cluster's super-member and members as a mixer and its related nodes. Then, decisions such as enabling and disabling of mixers rely on splitting and merging decisions.

The prototype architecture is described in Figure 8.1. Each node has two functional modules: conferencing and mixing. Depending on the node status (mixer or disabled node), the media path either passes through the mixing module or through the conferencing module. When joining the conference, a node turns on the conferencing module. Then, it starts sending and receiving media streams through the conferencing module. This module is composed of call signaling and media core components. These components interact via an Application Programmer Interface (API). The API includes functions such as "open media connection", "close media connection", "enable mixer" and "disable mixer". The media core is dedicated to media handling. It establishes, maintains and tears down media connections. Furthermore, it enables the mixing module to act as a mixer when the node is selected. Subsequently, media streams will be switched through the mixing module and then mixed by the mixer core.

To illustrate how the prototype works we provide the following scenarios. Suppose there are three nodes: *A*, *B*, and *C*, in communication within a cluster, where *A* is the super-member. *A* acts as a mixer as well, as per our assumptions. When node *B* wants to invite node *D*, it refers *D* to its super-member *A*. Next, node *A* establishes a call signaling link with node *D* and then invokes the "open media connection" function of the API. Finally, the mixer core of node *A* establishes a media connection with the media core of node *D*.

Now, if the cluster has to split into two, super-member *A* has to choose a node to be the super-member of the new cluster, for example node *C*. When node *C* becomes a super-

member, it invokes the "enable mixer" function of the API. Then, the media core of node *C* enables the mixing module and switches the media connection with node *A* from its media core to its mixer core. Node *C* is now ready for joiners.

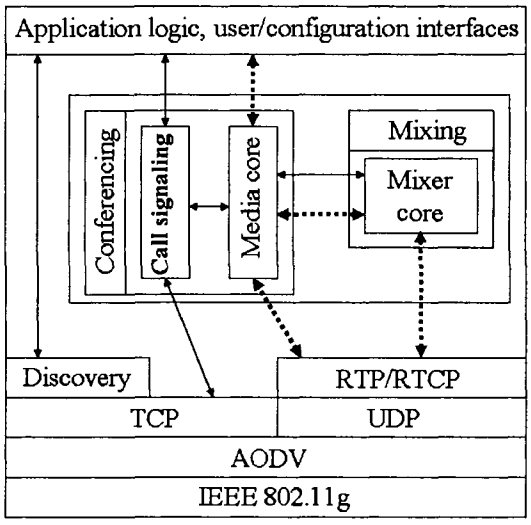

Figure 8.1. DMA prototype architecture

8.1.2. Prototype experiments and results

We make the comparison with hierarchical media mixing architecture, which we implement using the same prototype architecture. In these experiments the mixers simply merge and forward the media streams (i.e. do not combine the media streams to reduce their number), and since this is the case for both architectures, it does not impact the conclusions drawn from the test results.

8.1.2.1. Low scale testing - Real conference

As a low scale real test we built a voice conference system with seven machines, each of which has a 2.2 GHz Intel Pentium IV, 512 MB RAM, and Windows XP. The audio is ULAW code,

8 bits, 8000 Hz, mono. The seven machines are connected to a 100 Mbps local area network. We compared our proposed distributed architecture to the hierarchical architecture. In both cases, we set three mixers, each of which is connected to four nodes.

As observed in Table 8.1, the latency (end-to-end delay) in our proposed distributed architecture is significantly lower than in the hierarchical architecture, where the media streams go through three mixers instead of only two. This leads to more delays, which in turn increases the latency.

	Distributed	Hierarchical
Average latency (ms)	51.49	65.31
Average jitter (ms)	15.33	21.92
Average packet loss (%)	6.8	9.1
Delays in mixers (ms)	21.83	29.74
Total bandwidth (Mbps)	5.9	6.5

Table 8.1. Low scale testing results

Jitter is the difference in the latency between stream's packets. When the jitter reaches a level higher than what can be seamlessly handled by the jitter buffers, packets are dropped. In Table 8.1, the higher jitter in the hierarchical architecture induces a higher packet loss.

The packet loss in the hierarchical architecture is 9.1 %, as opposed to 6.8 % in the distributed version. In both architectures, this loss annoys users but the speech is still intelligible. The voice quality, defined by latency, jitter, and packet loss, will decrease as more mixers are added

to the hierarchical architecture while it should remain virtually the same in the distributed architecture because there will never be more than two mixers in end-to-end paths.

8.1.2.2. Medium scale testing - Simulation

We ran a medium scale test by simulating thirty nodes. We used six machines with the same features as above. We set three machines as mixers and use the others to simulate nodes (nine per machine). We compared the results for distributed and hierarchical architectures.

As inferred from Table 8.2, latency in the hierarchical setup exceeds the one-way end-to-end delay recommended for high-quality real-time traffic [90] (more than 150 ms). In addition, packet loss is considerably higher. Hence, voice quality does greatly decrease in the hierarchical architecture while remaining acceptable in our proposed architecture.

We can conclude that real-time conferencing does not tolerate more than two mixers in an end-to-end path. This further justifies our choice of a flat structure of mixers. It also justifies our choice of a full meshed structure for mixers.

	Distributed	Hierarchical
Average latency (ms)	123.2	175.27
Average jitter (ms)	33.99	46.08
Average loss (%)	13	19
Total bandwidth (Mbps)	74.93	80.56

Table 8.2. Medium-scale testing results

8.1.3. Simulation experiments and results

We modeled the distributed mixing architecture in OPNET and we conducted simulations. In this experiment, we assume that all nodes have a transmission range of 1000 meters and are randomly distributed over a 4 km x 4 km area. Nodes move in defined trajectories and may be out-of-range. For audio streams we used G723.1 code of 6.3 Kbit/s (ITU-T standard [91]). The G.723.1 standard is mostly used in Voice over IP (VoIP) applications for its low bandwidth requirement. It compresses voice audio in chunks of 30 milliseconds, each coded in 24 bytes. We simulated conferences with only one speaker at any given time without silence suppression. We measured end-to-end delay, number of delivered messages, throughput (amount of delivered data) and message loss caused by exceeded buffers, triggered by increasing the number of nodes in the network.

Figures 8.2 and 8.3 show the end-to-end delays during conferences with low (under 40 nodes) and high (more than 40 nodes) number of nodes, respectively. They demonstrate the high value of end-to-end delay when the number of nodes exceeds 48. Because nodes are mobile in the network, the number of hops increases with time progress, which in turn increases the end-to-end delays. Except for the beginning of the conference in which the delays are high because of hop discovery, the peaks in the curves (Fig. 8.2) are explained by time consumption for the discovery of new paths caused by node mobility. The simulation time (10 seconds) is related to the limitation of the OPNET tool performance; however, it does not impact the conclusions drawn from the experiment results because of the restriction of node mobility area. The same conclusions as below are confirmed by Figure 8.4, which measures the average of end-to-end delays with different numbers of nodes. Beginning from 36 nodes, the delay increases linearly until it exceeds the limits of media quality requirements (more than 150 ms).

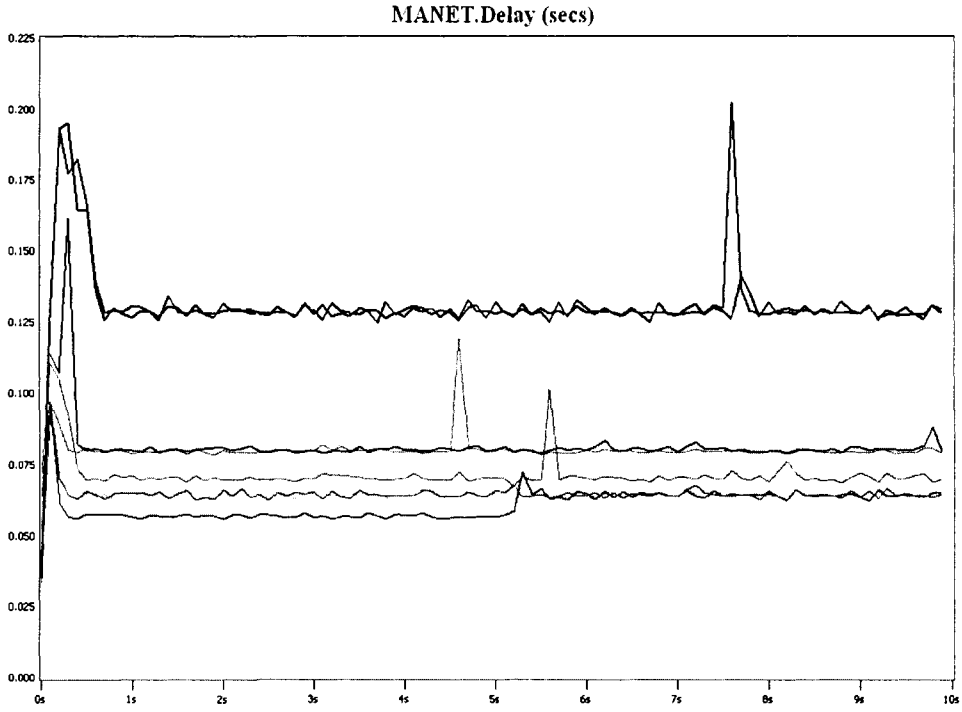

Figure 8.2. Delays during conferences with a low number of nodes (<40)

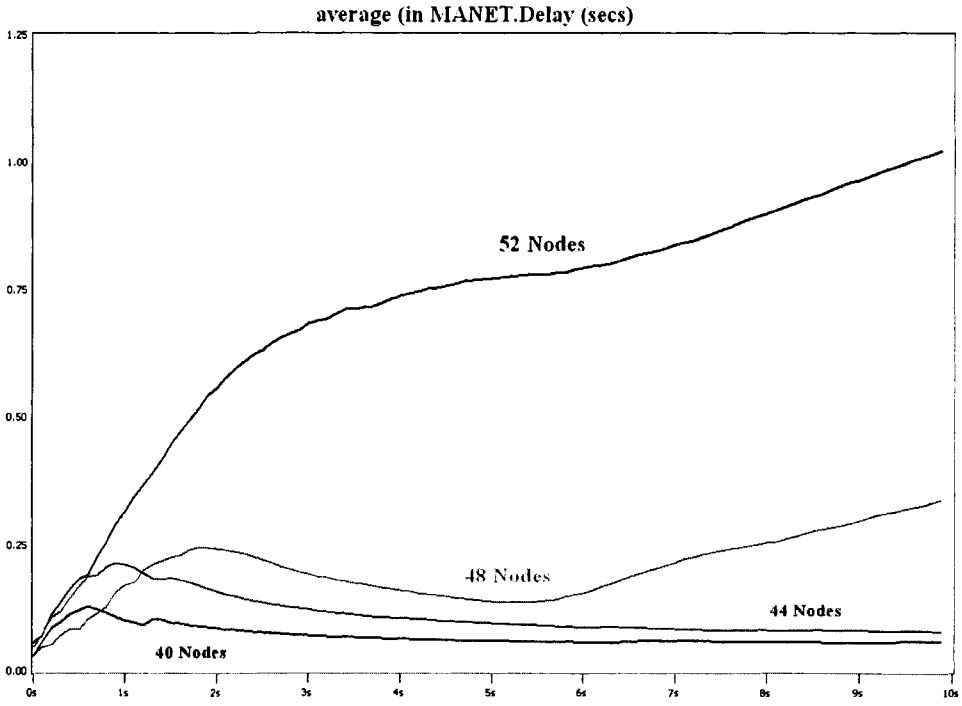

Figure 8.3. Delays average during conferences with a high number of nodes (>40)

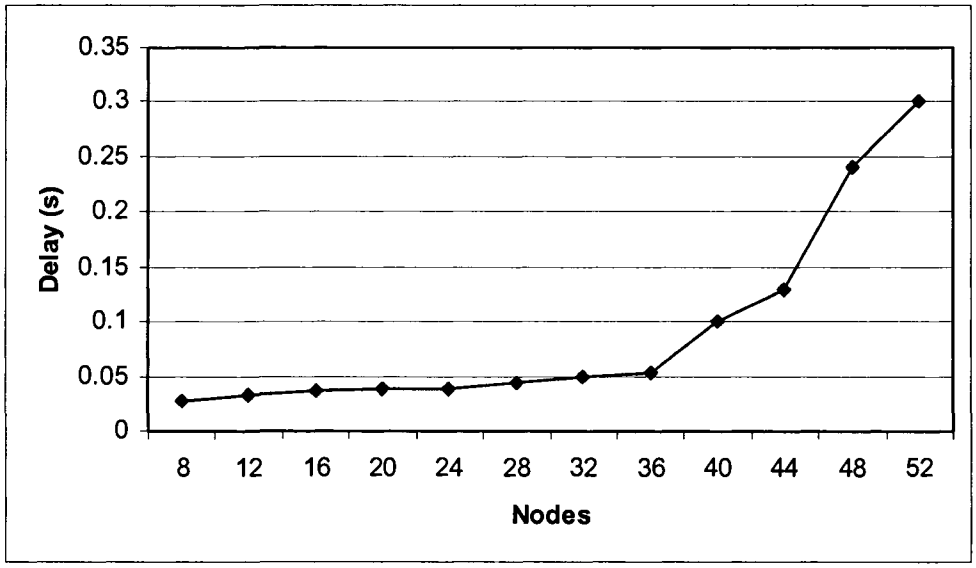

Figure 8.4. Average delays with different number of nodes

Figure 8.5 shows the number of delivered messages. In this figure, beginning from 40 nodes the slope of the curve significantly decreases till 48 nodes. Figure 8.6 shows the throughput of the network and demonstrate the same conclusions. This is explained by message loss caused by the high number of hops between each two nodes (around five hops). As we see in Figure 8.7, the conference with 48 nodes presents a significant loss of data at the end of the conference. With high number of nodes in the network, buffers of intermediate nodes are overflowed as the time progresses and then message loss significantly increases in the network. Conferences with more than 48 nodes are unpractical with high delays and a huge amount of data losses. This result is specific to the application configuration that we set, the lower layers' configuration (e.g. active route timeout, timeout buffer) and the OPNET tool performance.

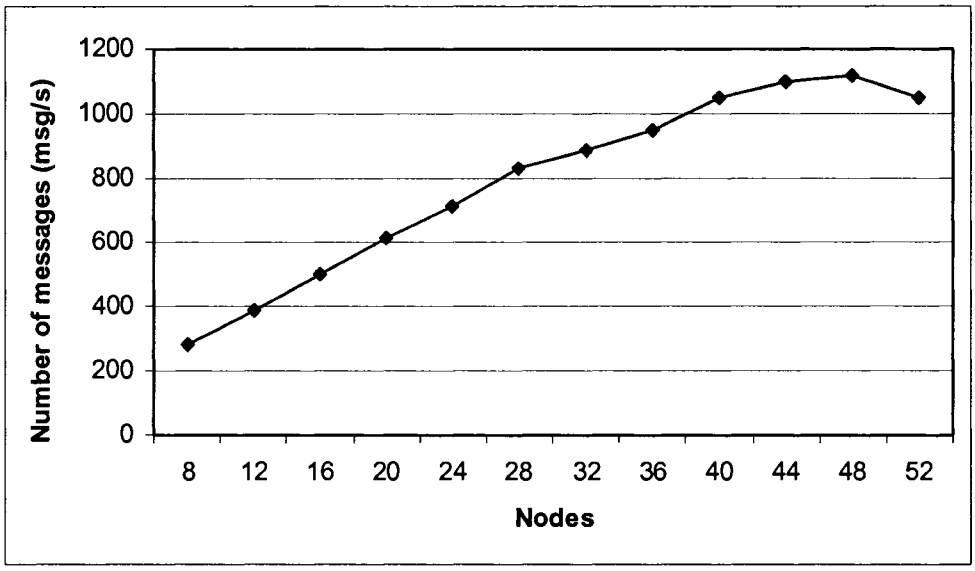

Figure 8.5. Number of delivered messages

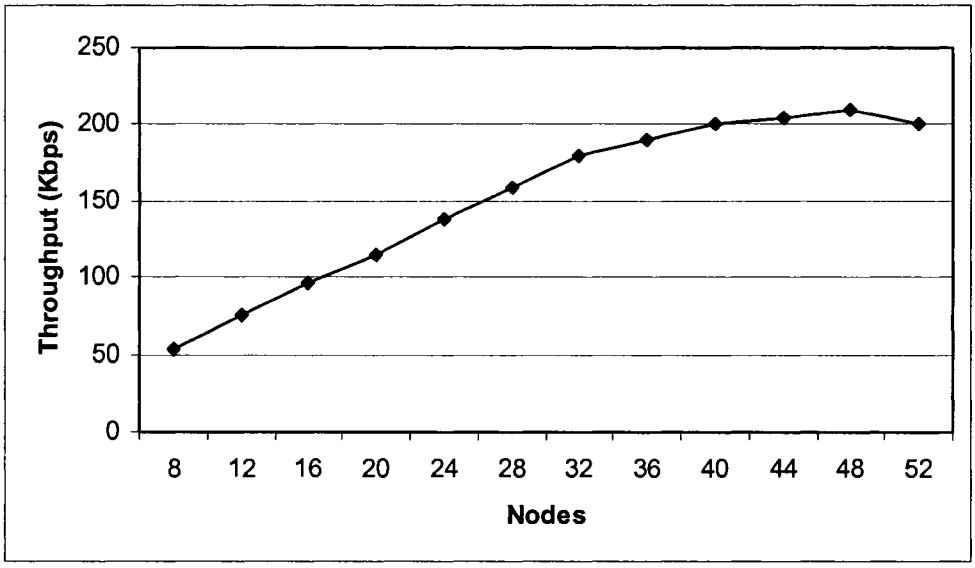

Figure 8.6. Throughput of delivered data

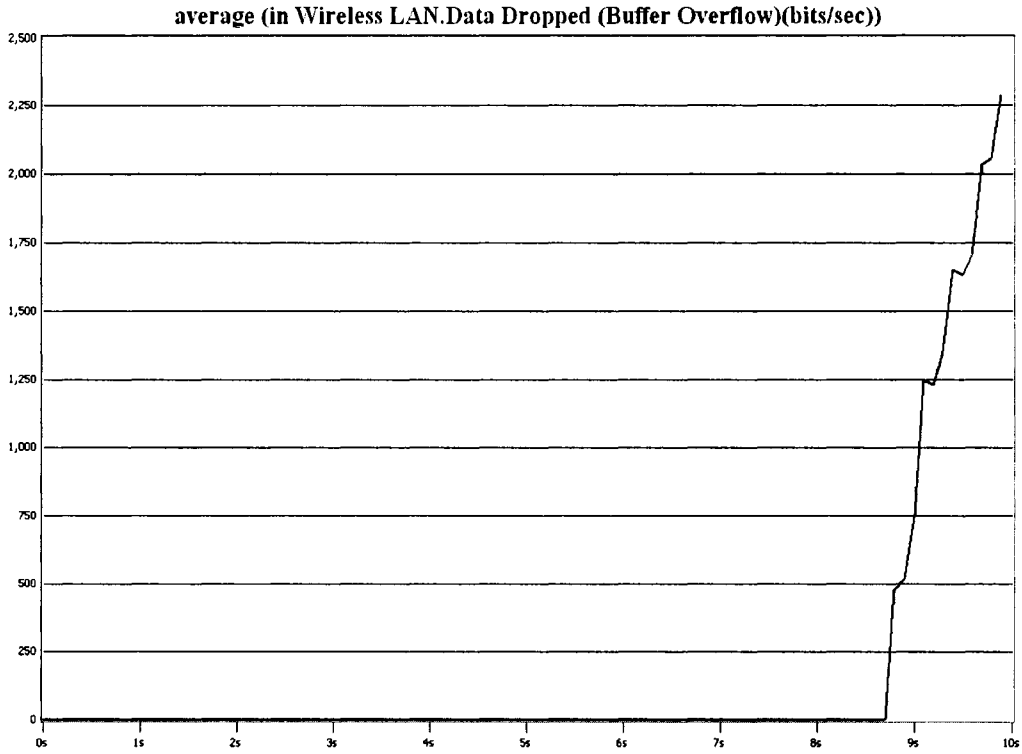

Figure 8.7. Data loss during the conference with 48 nodes

In this section, we evaluate DMA by prototype and simulation experiments. We demonstrate the limit of the number of nodes that can conference in a MANET in a specific application and network configuration.

8.2. Media signaling protocol

We conducted experiments to measure the number of messages generated by the media signaling protocol under different network situations. We simulated conferences using an implementation built with Java. Nodes in conferences randomly join and leave until the last node has left. Moreover, nodes' mixing and control capabilities are generated randomly. A conference is simulated as follows: first, we increase the probability of joining and decrease that of leaving until equilibrium is reached, then we start decreasing the probability of joining

while increasing that of leaving until the last node has left the conference. In the implementation, for simplicity's sake, we exclude the case of nodes that can have both mixing and control capabilities at the same time and thus can be both controller and mixer simultaneously.

The implementation simulates MANETs by abstracting wireless and mobile network features such as: out-of-range, time-out messages and group multicasting. In MANETs, nodes are randomly moving in unpredictable directions that frequently make nodes out of range for a certain time (sufficient to detect a node's unresponsiveness) and consequently unavailable for conferencing. When the node comes back in range it is treated as a new joiner by the conference. We abstract the out-of-range feature by making nodes in the conference randomly unavailable for a random period of time. We note the probability for a node to become failed during the conference by Pu.

Moreover, by inserting a random loss of messages we abstract the problem of time-out messages in MANETs routing protocols. For each experiment we define the rate of loss of messages, noted by Lm. Furthermore, the implementation simulates multicasts between MGCs that abstract multicast trees constructed by routing protocols to reduce the traffic. Finally, all of these abstractions do not affect the results, since we compare many scenarios under the same conditions.

In a series of experiments, we compare conferences with different node failure probabilities. These experiments are necessary to observe the reaction of the system under different network conditions. In one such network condition, nodes are rapidly moving in oscillating directions, which leads to a high node failure level in the network. To evaluate the system under these conditions, we measure the number of messages exchanged in the network by different node

failure probabilities. We fixed the rate of loss of messages Lm to five percent for all of the experiments, which is reasonable for conferencing. The conferences connect three hundred nodes.

Figure 8.8 shows four curves, each representing the number of messages while a conference progresses with a fixed node failure probability value. We observe that all the curves are linearly ascendant with different ascent coefficients. The differences between these coefficients appear to be equivalent. This result leads us to conclude that the system ensures no significant expansion of the number of messages for Pu values from 0 to 0.3. We believe that a node failure probability of 30 percent is already high (it doubles the number of messages from Pu = 0), and any higher value would make the network impractical for conferencing.

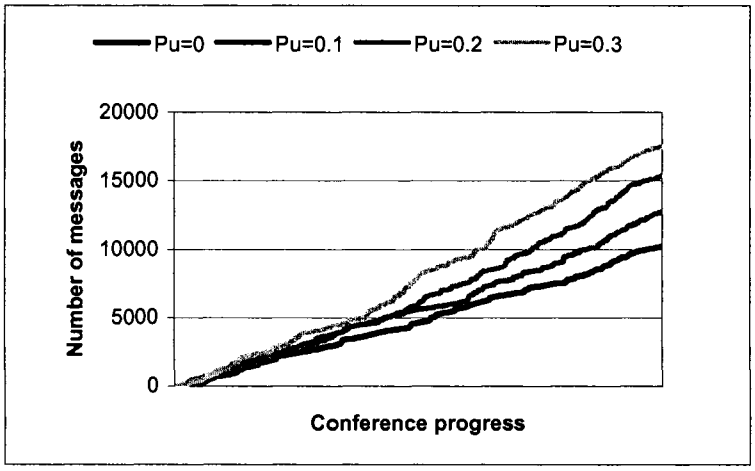

Figure 8.8. Media signaling protocol reaction to different network situations

In this section, we demonstrate the number of messages generated by the media signaling protocol under different node failure probabilities and show the limit of 30 percent as the upper value for realistic conferencing.

8.3. Self-organizing

We conducted experiments to simulate conferences and observe scalability, resource efficiency and how mixers and controllers are automatically enabled and disabled. Furthermore, we evaluated the scalability and reliability of the failure detection protocol by simulation. This section is divided into two parts: self-growing and self-shrinking, and failure detection.

8.3.1. Self-growing and Self-shrinking

To begin, we simulate the expansion of conferences to demonstrate how well the two-overlay networks grow and how well the workload is balanced among the mixers and controllers. Next, we performed experiments to demonstrate the concepts and to evaluate to what degree the self-growing and self-shrinking schemes ensure an efficient use of resources. To that end, we measured the workload among mixers and controllers over a whole conference (expansion and contraction). In addition, we have compared a former workload-balancing scheme with an enhanced one in terms of the number of nodes that can attend the conference. A simulator was built using Java to perform the experiments.

8.3.1.1. Network scalability and workload measurements and results

We have experimented with the growth of the system and have observed the variation of the number of mixers in the network. (The same results are applicable for controllers since they use the same growth scheme). Moreover, we have conducted experiments to demonstrate that the workload is balanced among the mixers when the network grows.

The first series of experiments simulate a conference in which nodes join till there is no mixing capability left in the network. In this simulation, nodes do not leave the conference. The event of disabling a mixer is triggered when the mixer does not perform mixing for any node. Nodes' capabilities are generated randomly between 1 and a maximum value, which is defined in each experiment. A node's capability is defined as the maximum number of nodes that can connect with it. We fixed the average node capability within the range of 1 to 25. Each experiment is simulated 1000 times and the average is measured.

Figure 8.9 shows a considerable network growth while the number of mixers goes linear with the network average node capability. We observe a considerable expansion of the network while the number of mixers remains significantly low. This stabilization is crucial to avoid an explosion of the number of mixers in the network, which would lead to a rapid network overload.

This phenomenon is due to the fact that expressions [6.2] and [6.4] favor the connection of joiners to mixers with high capabilities at the beginning of the conference. These expressions also serve to maintain the equilibrium of the number of mixers in the network.

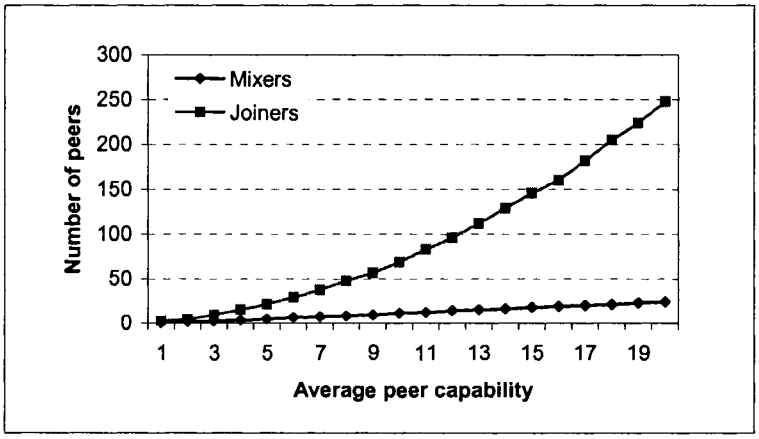

Figure 8.9. A considerable network growth while the number of mixers remains low

To quantitatively measure how well the workload of the mixers is balanced when the network grows, we use the variance of the mixers' workload. Then, we compute the average mixers' network workload, which is:

$$AvgW = \frac{1}{m} \sum_{p \in M} w_p \qquad [8.1]$$

M the set of mixers
m the number of mixers
w_p is the workload of node p

Hence, the variance of the mixers' workload, denoted by V, becomes:

$$V = \frac{1}{m} \sum_{p \in M} (AvgW - w_p)^2 \qquad [8.2]$$

M the set of mixers
m the number of mixers
w_p is the workload of node p

We measure the variance by increasing the number of nodes in the network. Obviously, a low value of V means a better balance of workload among mixers. This converges to zero for a perfectly balanced network.

As shown in Figure 8.10, it is difficult to maintain a perfectly balanced workload at all times. The three curves (one per network node capability average) have two parts: an upward part and a downward part.

In the upward portion, the network is constructing the network of mixers. Each enabled mixer in this phase increases the variance. In the downward portion, the network of mixers has already been constructed and nodes are simply connected to mixers. This phase decreases the variance till the network is blocked -- where all the mixers are overloaded and the variance is equal to zero.

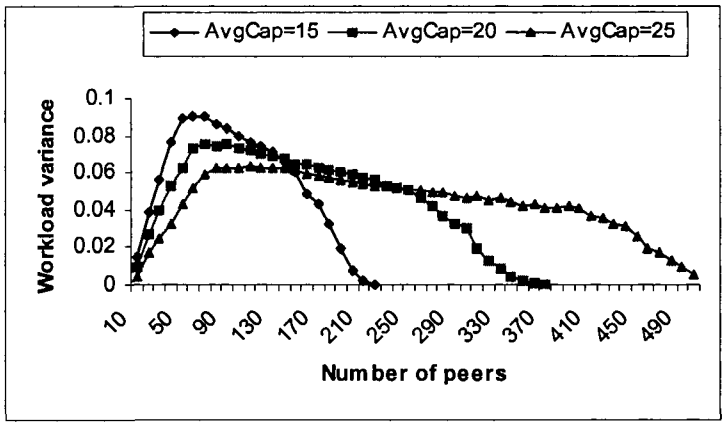

Figure 8.10. Mixers' workload variance when the network grows

8.3.1.2. Automatic stabilization, resource efficiency and enhancement performance experiments and results

We conducted experiments to prove the concept by showing how the system of controllers and mixers automatically evolves when the conference grows and shrinks. The experiments simulate a conference in which nodes randomly join and leave until the last node has left. We assume that the wireless ad-hoc network provides a peer-to-peer communication service to the conference application, and so wireless features are abstracted by the out-of-range of nodes and the messages' time-out and loss properties. In simulations, node capabilities are generated randomly between one and a maximum value defined in each experiment. In the experiments, we increase the probability of joining and decrease that of leaving until equilibrium is achieved; then we decrease the probability of joining while increasing that of leaving.

As shown in Figure 8.11, the beginning of the conference demonstrates considerable network growth while the number of mixers and controllers stays linear. We observe the phenomenon of a marked expansion of the network while the number of mixers and controllers remains

significantly low. This stabilization is crucial to avoid an explosion in the number of mixers and controllers in the network, which would quickly lead to network overload. The end of the conference shows the mixers' and controllers' sub networks shrinking.

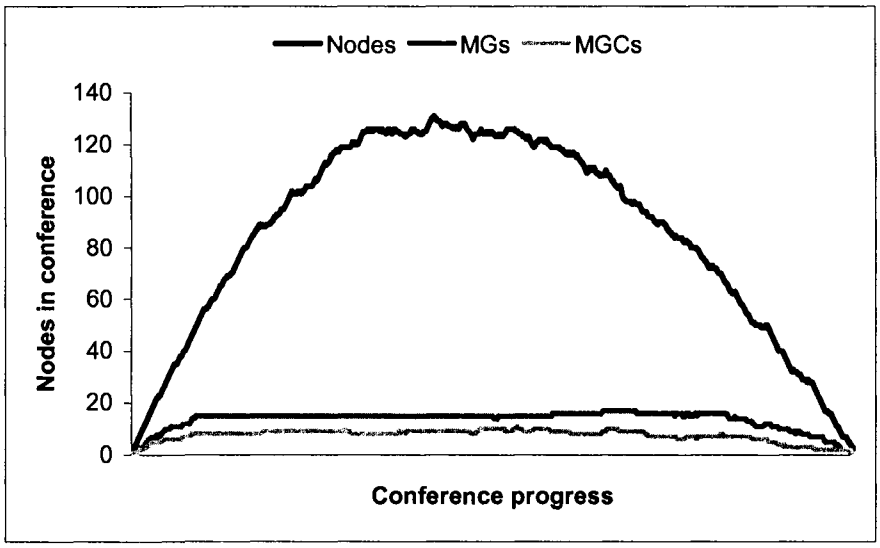

Figure 8.11. Automatic stabilization when the network grows and shrinks

In the second series of experiments, we use the variance of the mixers' workload to quantitatively measure how well resource usage is optimized during the conference. (The same is applicable for controllers). Then, we compute the average network workload as the conference progresses. Obviously, a lower variance value means a better workload balance among mixers, and the variance converges to zero for a perfectly balanced network.

As shown in Figure 8.12, the workload variance is high at the beginning of the conference, and low at the end. Every peak in the curve represents an enabled mixer that temporarily increases the workload variance. At the beginning of the conference the system enables several mixers, which explains the presence of several peaks. As the conference progresses, the sub network of mixers has already been constructed and nodes are simply connected to mixers. When there

is a need for new mixers, the system enables them as shown in the middle of the conference. Finally, when the number of leavers is equal to or more than the number of joiners, the variance becomes stable and near zero because there is a balanced distribution of nodes among mixers.

This scheme favors the powerful mixers in the network in order to achieve maximum growth in the conference. Moreover, joiners are distributed to mixers in an optimized manner that balances the workload among mixers.

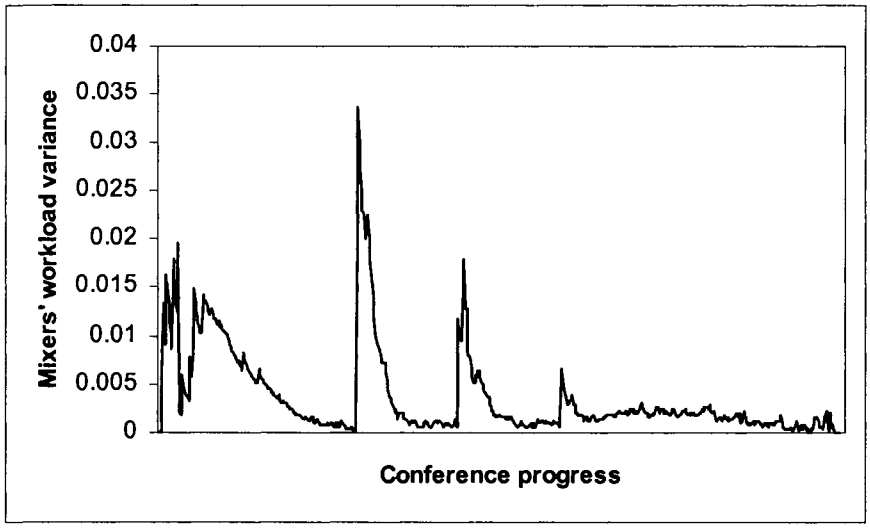

Figure 8.12. Mixers' workload variance observed during the conference progress

In the third series of experiments, we compare the enhanced scheme described by Expression [6.3], only used at high loads, with the former scheme described by Expression [6.2], used at all loads, in terms of conference attendance. As shown in Figure 8.13, the beginning of the conference demonstrates the same network growth for the two schemes. While the former scheme continues conference growth at high loads, the enhanced scheme

tries to postpone the peak of the conference to avoid a premature deadlock. We observe that the enhanced scheme allows more nodes to attend the conference.

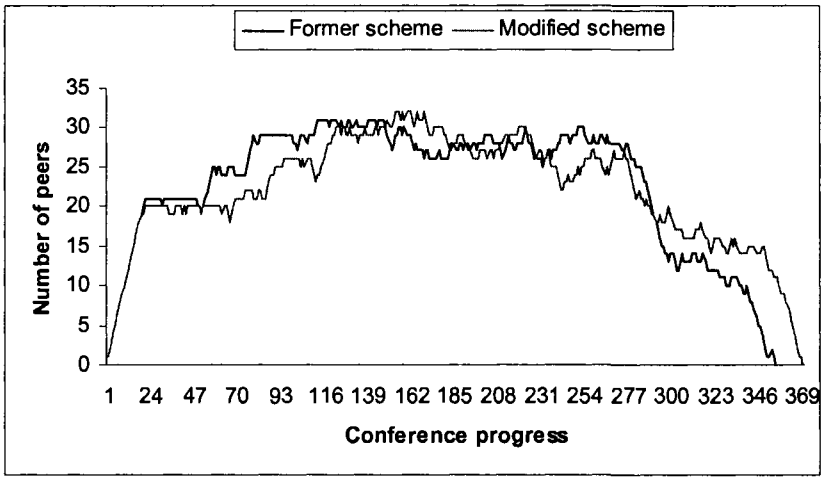

Figure 8.13. Comparison between the former and the enhanced workload balancing schemes

8.3.2. Failure detection

The quantitative study presented in this section evaluates the scalability and reliability of the failure detection protocol in a MANET environment. We modeled the failure detection protocol in OPNET and we conducted simulations. The experiment simulates a network of 70 nodes (constructed at the beginning with five detectors) in which nodes randomly fail and recover. We assume that all nodes have a transmission range of 1000 meters and are randomly distributed over a 4 km x 4 km area. Nodes move randomly and may be out of range. We set the frequency of heartbeats to 5 seconds. The message loss probability was fixed at 5 percent, which can increase by about 2 percent due to nodes being out of range during the simulation. We use a simple multicasting scheme whose reliability is only affected by message loss.

8.3.2.1. Scalability measurements and results

We define scalability by two criteria: the number and the delay of messages during the simulation. These two criteria should maintain a horizontal linear behavior to assure scalability. As we see in Figure 8.14, the average of the exchanged messages is stable during the simulation, which guarantees scalability. We estimate the overhead of failure detection self-organizing as four messages per second (256 bps). In Figure 8.15, the curve of the delay of messages is composed of two parts. The first part maintains a delay of 75 ms and the second part shows some increase in the delay. During the second part, we observe an important drop of wireless packets -- around five packets per second. We note that the average of the number of hops per route is four.

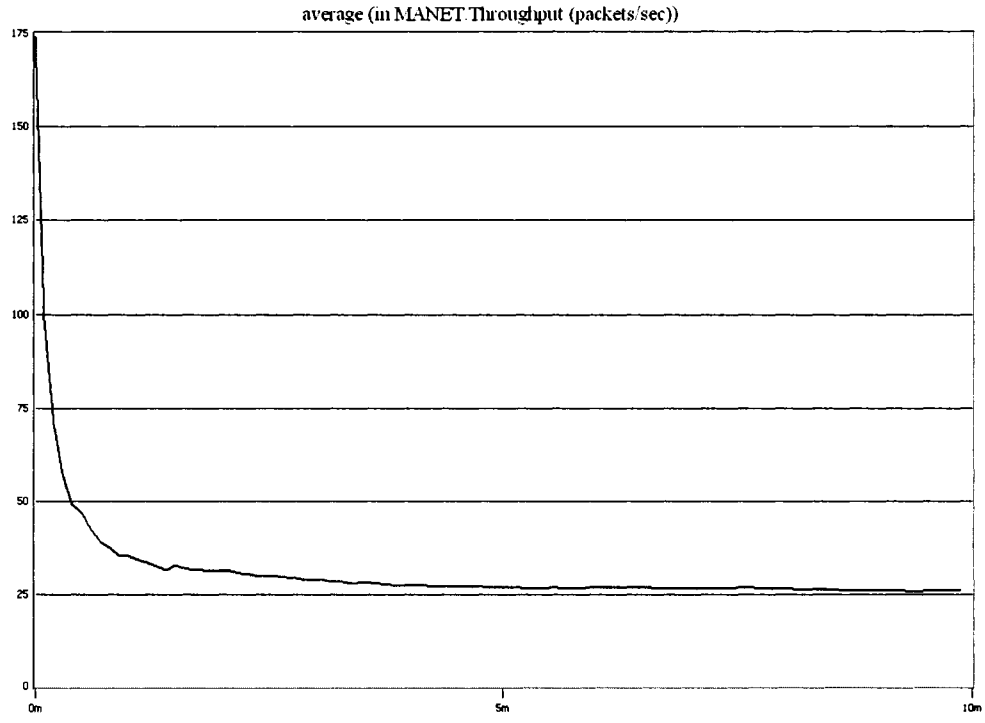

Figure 8.14. Number of messages during the simulation

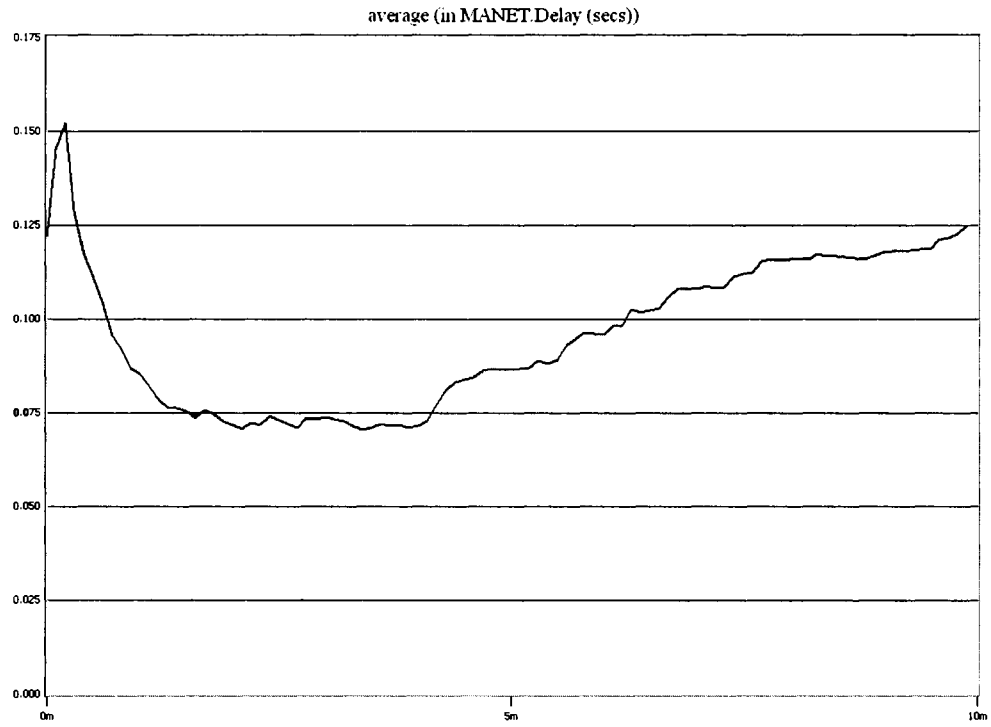

Figure 8.15. Delay of messages during the simulation

8.3.2.2. Reliability measurements and results

We define reliability by completeness and accuracy. Completeness indicates the report of the failure to non-failed nodes. Accuracy implies no false detection, so that a non-failed node will not be stated failed by any non-failed node. Completeness is closely related to the multicasting reliability, which in turn is related to message loss in our experiments. Accuracy is only affected by false detection reported by certain non-failed nodes. We can identify the conditions under which a false detection may happen:

- **C1: Condition for an affiliated node:** The detector receives neither the heartbeat message from the affiliated node nor the correct message.

- **C2: Condition for a detector:** A detector receives neither the heartbeat message from the detector nor the correct message, and receives at least one warning message concerning the failed detector from another detector.

We made a series of simulations and we collected the number of false detections and failures not reported. Figure 8.16 shows that the number of false detections (the curve on the bottom) and the number of failures not reported (the curve in the middle) demonstrate a small increase during the simulation, but they are still very few compared to the number of checked nodes (the curve at the top). This shows that the architecture probabilistically (considering message loss probability) maintains the two criteria, completeness and accuracy.

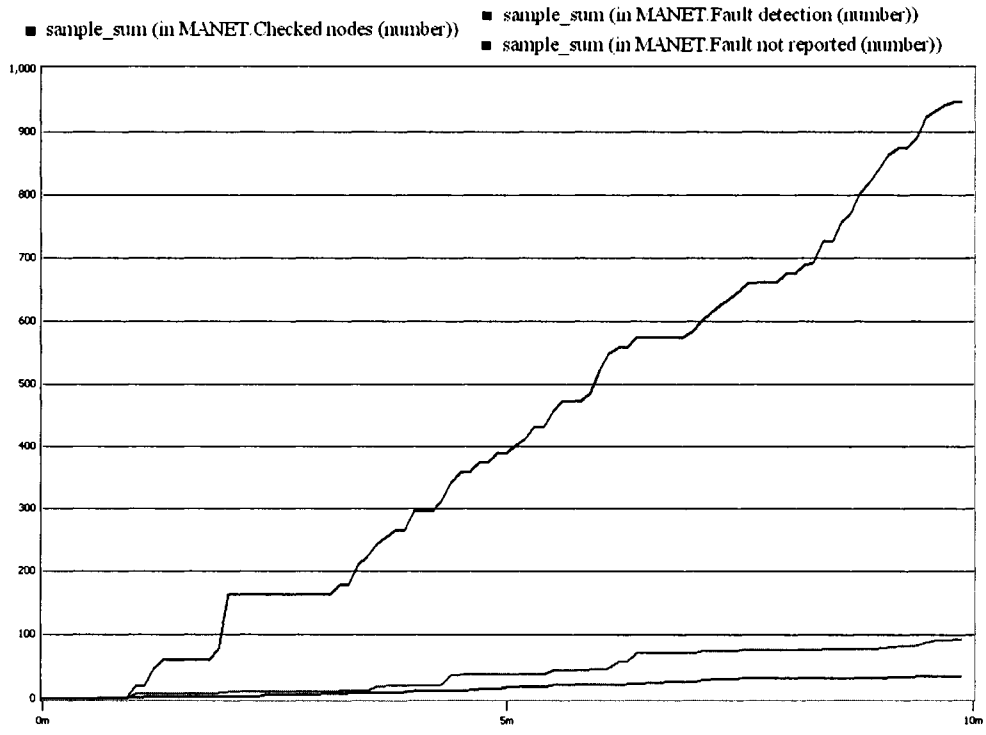

Figure 8.16. False detections and failures not reported during the simulation

The failure detection architecture provides a probabilistic detection error dependent on message loss probability in the network. Of course, false detection and failures not reported will affect the normal operation of conferences. In addition to taking some nodes out of the conference, false detection can lead to an extra load of reconnection especially for false-detected mixers and controllers. On the other hand, failures not reported will defer the failure detection and then disturb conferencing until the failures are reported.

8.4. Media Handling Architecture for MCNs (MHAM)

We have implemented a proof-of-concept prototype of MHAM with all of the functional blocks to represent the interconnection of the two networks (3GCN and MANET) via the MM entity. However, we have implemented an elementary version of the functional blocks of the MM. We have enhanced the functionality of the MGCs in MANETs. The enhancements are related to the MGC interface with MGCM. We have performed experiments to prove the architecture's concepts. The media mediator architecture is presented first, followed by the proof-of-concept prototype implementation environment, and prototype measurements and results.

8.4.1. Media mediator architecture

The MM is the new functional entity that we have introduced. Its software architecture is depicted in Figure 8.17. Functional blocks such as MGCM core, MGM core, protocol translator, MGC logic and different interfaces implement the basic functionality of the MM. The 3GCN configuration entity provides a user interface to configure the address of the 3GCN access point. The 3GCN users registry functional block maintains the local addresses

of 3GCN users. The MGCM interface with 3GCN implements the extended 3GCN protocol. It also has a MANET interface that implements the MANET protocol. A protocol translator is needed if a MANET uses a protocol different than the extended 3GCN protocol. The MGCM uses an MGC logic entity to make decisions and actions concerning the management of the MANET. Finally, MGCM has two interfaces with the CGW. The first interface is for the exchange of primitives. The second interface is used by the CGW to discover an MGC in the network or a specific MGC that controls a specific user. In fact, the MGC discovery entity is independent of the MM and it is deployed in each MANET user function.

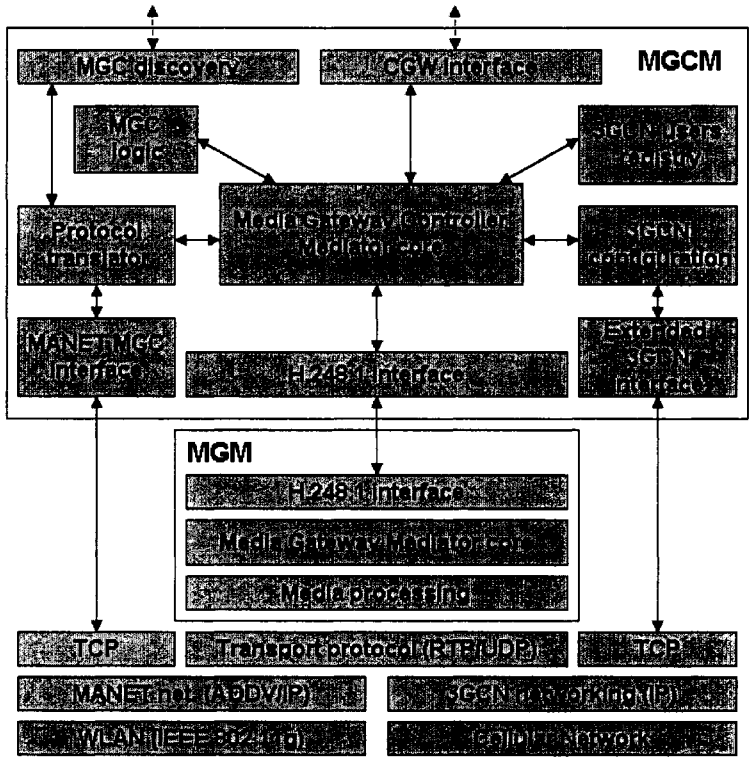

Figure 8.17. Media Mediator architecture

8.4.2. Prototype implementation environment

We built a proof-of-concept prototype of an audio conference in MCN. We used Java as the programming language in an MS Windows environment. For the MANET routing protocol, we used an implementation of AODV [19] (Intel Corporation, winaodv, V 0.1.14). We created a MANET network using three laptops (1.6 GHz Intel Pentium mobile, 504 MB RAM) and four desktops (2.2 GHz Intel Pentium IV, 512 MB RAM) with IEEE 802.11g adaptive cards (Figure 8.18). We have implemented simplified versions of all of the functional blocks of the MM. The creation of a full MCN environment is not feasible in our current experimental setup. However, this is not required for a proof-of-concept of the media handling architecture. Thus, we have implemented a very simplified MRFC and MRFP, which only contains the creation of local Context, addition of Terminations, and the interface with the MGCM configuration.

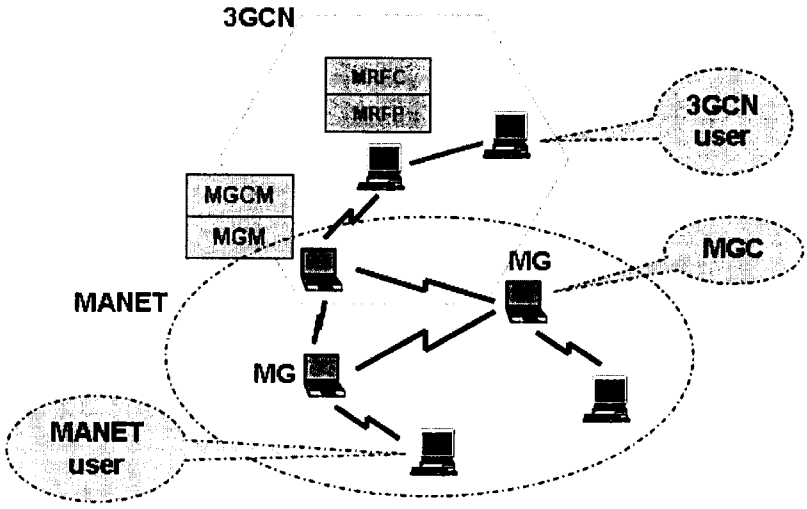

Figure 8.18. MCN test bed architecture

We have used the call signaling system developed in our lab as described in [87]. The protocol that defines interfaces between networks and the mediator is implemented by a set of

Megaco/H.248 based Java APIs that assure the exchange of messages between the different entities. The set of Java APIs are: add(user_address), add_reply(local_address), subtract(user_address), subtract_reply(). The set of primitives triggered between the CGW and the MM are also implemented by Java APIs.

We deployed the CGW and the MM on the 3GCN side. This will simplify the discovery of the MGCM by the CGW. We assume that the 3GCN is pre-configured with a known CGW and MM. Then, the discovery mechanism of MM by CGW will be a direct API access point. Since both sides use the same transport protocol, no MGM entity is needed.

8.4.3. Prototype experiments and results

The experiments included interconnecting the two networks via the MGCM entity. We have tested conference initiation with two users on the two sides and with the addition of users from both sides. Our results show that the proposed architecture works well in each of these cases. Furthermore, we did not observe any difference in performance compared to the standalone MANET experiments.

8.5. Summary

In this chapter, we presented experiments and results with discussions of each of the architectures, schemes and protocols of the media handling architectures for MANETs and MCNs, which are: distributed mixing architecture (DMA), media signaling protocol, self-growing and self-shrinking schemes, failure detection protocol, and media handling architecture for MCNs (MHAM).

CHAPTER 9

CONCLUSION

This chapter is divided into three sections, starting with a thesis summary. The second section then presents the main contributions of this work organized in terms of its philosophy, theory, realization and use. The third section provides reflections on the work presented here and identifies areas for future research.

9.1. Thesis summary

This thesis focuses on the design and evaluation of media handling architectures for conferencing in Mobile Ad hoc NETworks (MANETs) and Multi-hop Cellular Networks (MCNs).

Chapter 1 provides the framework by showing that media handling architectures for conferencing in MANETs and MCNs are becoming a necessity, and details the specific constraints to observe. With the rapid increase in the deployment of MANETs involving lower layers, there is a real need for real-time and multiparty applications, and especially conferencing applications. It is therefore important to understand these issues and build architectures that can handle media and control media connections and media mixers in MANETs and MCNs.

Existing media handling approaches may be used for conferencing in MANETs. However, they do not explicitly consider the specific feature of MANETs, e.g. they do not address decentralization. Current approaches in media handling do not include self-organizing to

automatically react to events such as a mixer leaving or node failure. There are currently no schemes to handle media for conferencing in MANETs, and that is a serious lack as it is the only way to deal gracefully with media streams in multiparty applications.

Media handling architectures that do not consider separation of the media signaling done by the Media Gateway Controller (MGC) from the media mixing done by the Media Gateway (MG) have a number of problems. These problems can include, but are not limited to: difficulty to decentralize, impossibility to standardize, and extreme complexity to integrate MANETs with 3GCNs for media handling purposes. Even a self-organizing scheme that separately manages MGCs and MGs will suffer from these problems.

The work in this thesis therefore focuses on the design of a novel media handling architecture for conferencing in MANETs, based on novel architectures: media mixing and media signaling, and a novel scheme: self-organizing -- composed of self-growing and self-shrinking based on novel workload balancing schemes, and self-healing – composed of a novel failure detection architecture and a recovery scheme. This work also focuses on the design of a novel architecture that handles media for conferencing in MCNs based on Media Mediator (MM) concepts. This thesis suggests the distribution of media handling tasks among the nodes according to their capabilities as a basic principle for conferencing in MANETs.

Chapter 2 presented an overview of MANETs and MCNs, followed by the background of multimedia conferencing including a description of media quality factors and conference categories, models and application components: call signaling and media handling.

Chapter 3 gave an overview of work related to media mixing architectures, media signaling architectures, self-organizing schemes, load balancing schemes, failure detection architectures,

and media handling architectures for MCNs. A series of review criteria was selected, evaluated for its importance for designing decentralized, scalable and resource-efficient architectures and schemes. The challenge is to investigate if it is possible to find the optimal tradeoff between the requirements and the assumptions. Maintaining a pre-determined media quality over multihop networks could be achieved only through a highly adaptive topology, since in MANETs we cannot avoid message loss and out-of-range of nodes. These criteria were applied to various existing architectures and schemes in order to understand how well existing approaches could support them. Even though each of the existing approaches has its own significant contribution to media handling networking research, none of them addresses all of the chosen criteria. For example, even though there are existing media mixing architectures that are decentralized and scalable in terms of the number of users in the conference, none of them achieve satisfactory media quality. Current approaches that guarantee satisfactory media quality are not decentralized nor can they be self-organizing. In chapter 3 certain features of interest from the reviewed work are highlighted, such as the use of distributed mixing for load decentralization and the separation of the media signaling done by the controllers from the mixing done by the mixers. These ideas were selected and then developed as part of the two main approaches presented in this thesis, namely the two-overlay media handling architecture for MANETs and the Media Handling Architecture for MCNs (MHAM).

Chapter 4 introduced a two-overlay media handling architecture as a novel approach to handle media for conferencing in MANETs. The first overlay network is composed of mixers that are fully meshed media connected. Every node in the network has a media connection with a mixer in the first overlay. The second overlay network is composed of controllers. Each controller controls one or more mixers. The controllers maintain control connections between

each other so they can exchange messages in order to maintain information on the two-overlay network. This information is used to make decisions concerning the growth and shrinkage of the conference and to assign joiners to mixers. It is also used when recovery is performed in the case of node failure. In the first part of the chapter, an overall architecture, a two-overlay architecture, was proposed as a global solution in MANET environments, followed by the nodal aspects including node life cycle, node functional architecture and the information kept in each node. The second part introduced Distributed Mixing Architecture (DMA) as a novel approach for media mixing in MANETs. Comprising the first overlay of the general architecture, it is made up of a set of mixers; each of which serves a number of related nodes. DMA uses a two-step mixing operation -- a new technique for mixing. In the first step, or mixing for external sending, streams sent from related nodes are mixed and immediately sent to the other mixers, and in the second step, called mixing for internal sending, buffered streams from the first step are mixed with the streams from the other mixers and are sent to related nodes. The chapter includes an algorithm for synchronization between mixers to deliver streams with the correct ordering and timing. The mixer core functional architecture is also described here.

Chapter 5 proposed a Megaco/H.248 based-architecture to manage and control the media connections and the media mixers, presented in terms of architectural principles, primitives, events, messages and procedures. The procedures that occur after each event are detailed by scenarios. The events covered here are: joining a conference, leaving a conference, enabling an MG, disabling an MG, enabling an MGC, disabling an MGC, recovering an MG, and recovering an MGC. This chapter is limited to the architecture that defines the exchange of messages between different entities and actions taken on messages transmission, or on

messages reception. The self-organizing aspects that govern decisions for growing and shrinking are solved in the next chapter.

Chapter 6 introduced a self-organizing scheme, which indicates to joiners which mixers they should connect to, and automatically enables and disables mixers and controllers when the network grows and shrinks. Self-organizing includes two resource efficient components: self-growing and self-shrinking, based on novel workload-balancing schemes, which equilibrate the workload among the mixers and controllers. Self-growing and self-shrinking include system decision criteria such as when to enable or disable a mixer or a controller.

The third component of the self-organizing scheme, self-healing, was also presented in chapter 6. Self-healing both detects failed nodes and acts to recover the network after failures. It has two components: a failure detection architecture and a recovery scheme. A novel application-level failure detection architecture is proposed, which is independent of lower layers and autonomous of the two-overlay architecture. This architecture can be applied to conferencing applications in general, and specifically to those in MANETs. Nodes form a self-organizing overlay network that is overlayed on the network that we need to be informed about when one of its nodes fails (e.g. the sub-network of controllers). In this overlay network, a set of nodes, called detectors, periodically sense each other's heartbeat messages. Every detector also senses the heartbeat messages of a set of nodes. If a detector does not hear from a node after a timeout (possibly due to node failure or message loss), it identifies the failure of the node according to a set of failure detection rules and then reports the failure to all of the other nodes. The self-organizing proposed in this architecture is different from that used by the two-overlay architecture. This one is based on the thresholds of detectors, and can enable new detectors in the overlay network. Thus, if the thresholds of all of the detectors have been

reached, a joiner starts acting as a new detector. The recovery scheme uses the information kept in the controllers to re-stabilize the network after node failures.

Chapter 7 proposed a media handling architecture for MCNs that integrates 3G Cellular Networks (3GCNs) with MANETs for conferencing based on Media Mediator (MM) concepts. The MM acts as a bridge to connect the controllers (MGCs) and mixers (MGs) of a MANET to the MRFC (the 3G controller) and the MRFP (the 3G mixer) of a 3GCN, respectively. The MM is composed of two functional entities: the Media Gateway Controller Mediator (MGCM) and the Media Gateway Mediator (MGM). MGCM acts as a controller mediator and MGM acts as a mixer mediator. MGCM assures connectivity between the networks and acts as a translator of different deployed protocols. The MGM is only necessary when the MRFP and MANET MGs use different media stream transportation protocols or when some specific media processing should be done when interfacing with MANETs, otherwise media connections will be made directly between the MRFP and MANET MGs. This architecture assures the independence of 3GCNs from MANET topologies and protocols and does not limit the topology of MANET MGs. Various scenarios for media handling session management are described, such as: a 3GCN user initiates a conference with a MANET user, a MANET user initiates a conference with a 3GCN user, a 3GCN user invites a MANET user, a MANET user invites a 3GCN user, a 3GCN user leaves a conference, and a MANET user leaves a conference.

Chapter 8 described the implementation or simulation environment, experiments and results of each of the constituents (architectures, schemes and protocols) of the media handling architectures for MANETs and MCNs. The evaluated constituents are: Distributed Mixing Architecture (DMA), media signaling protocol, self-growing and self-shrinking schemes, failure

detection protocol and media handling architecture for MCNs (MHAM). The first part of the chapter presented the DMA proof-of-concept prototype used for small and medium scale experiments. The experimental environment and the results compared to hierarchical mixing were presented, which show that DMA outperforms hierarchical mixing by maintaining satisfactory media quality at the small and medium scale. Large-scale experiments were performed by simulation, using the OPNET tool, to show the limitation of conferencing in MANETs. Obviously, this result depends on the simulation environment (e.g. simulation area, antenna strength, speed and trajectory of mobile nodes) and the conference feature (e.g. number of speakers at a time, silence suppression, audio coding).

The second part of the chapter simulated the media signaling protocol using a Java application to evaluate the overhead. The number of messages generated by this protocol was measured under different network situations, such as different node failure probabilities. The results showed that the system ensures no significant expansion of the number of messages up to a 30 percent value of node failure probability.

In the third part of the chapter, the self-growing and self-shrinking scheme was simulated to evaluate the scalability, resource efficiency, stability, and performance of the enhancement. We observed the phenomenon of a considerable expansion of the network while the number of mixers and controllers remains significantly low. This stabilization is crucial to avoid an explosion in the number of mixers or controllers in the network that would quickly lead to a network overload. The end of the conference showed the automatic mixers' and controllers' sub-networks shrinking. The experiments, using the variance of the mixers' workload, quantitatively demonstrated how well resource usage is optimized. They showed that, as the conference progresses, the variance becomes stable and near-zero, which means there is a

good distribution of workload among mixers and controllers. The enhancement made at high load aspect demonstrates better performance in terms of the number of nodes that can attend the conference.

The fourth part of the chapter evaluated the scalability and reliability of the failure detection protocol with OPNET simulations. The scalability is defined by two criteria: the number and the delay of messages during the simulation. The experiments confirmed a guarantee of scalability with a stable average of the number of exchanged messages and acceptable delivery delays during the simulations. Reliability is also defined by two criteria: the number of false detections and of failures not reported during the conference. The experiments showed that the two criteria are maintained probabilistically dependent along with the message loss probability in the network.

The last part of the chapter presented a proof-of-concept prototype for media handling in MCNs, with the media mediator architecture and implementation environment. The results showed that the proposed architecture works well in each of the following cases: initiation with two users on the two sides and the addition of users from both sides. Furthermore, no difference was observed in the media quality compared to the standalone MANET experiments.

9.2. Thesis contributions

This section elaborates the main original contributions of this thesis. As noted at the beginning of the previous section, the central interests of this thesis revolve around media handling for conferencing in MANETs and MCNs.

- The work contributes to the philosophy of handling media for conferencing applications in MANETs and MCNs by recognizing the importance of decentralizing, ensuring scalability and self-organizing. In this way, the conferencing applications are more robust against bottlenecks, automatically allow a maximum number of users in the conference, and fault tolerant to failures caused by out-of-range and node failure. This work presents a novel media handling architecture for MANETs, called a two-overlay architecture, which is composed of a set of new concepts, architectures, schemes and protocols. Furthermore, it presents a novel media handling architecture for MCNs based on media mediator concepts to integrate MANETs with 3GCNs for media handling purposes.

- This work presents a new mixing architecture, Distributed Mixing Architecture (DMA), which extends the traditional concept of mixing by making it distributed and responsive both to a MANET environment and to application requirements. DMA is well suited for real-time media transportation because there will never be more than two mixers in end-to-end paths. It is also very appropriate for MANETs since media streams may pass through multi-hops between every two nodes. Furthermore, DMA can be easily deployed in a resource-efficient manner by assigning the role of mixer to the most powerful nodes. The work includes a prototype used for evaluation experiments on both a small and a medium scale. The large scale was evaluated by simulation with the OPNET tool. Part of this work was published in the Proceedings of the 10th IEEE Symposium on Computers and Communications (ISCC 2005) [92].

- The work presents a self-organizing scheme that provides decisions for self-growing and self-shrinkage. The scheme maintains a stable number of mixers and controllers during the conference, which avoids an explosion of their number that would lead to a quick network

overload. This was published in IEEE Network Magazine [93]. The self-growing and self-shrinkage schemes are based on workload balancing schemes that balance the load among the mixers and controllers according to their capabilities. These workload-balancing schemes maintain the networks' resource efficiency throughout the conference and allow a maximum number of joiners. Simulation by a Java application is done to demonstrate network scalability, resource efficiency, and stability. An enhanced workload-balancing scheme was published in the Proceeding of the 7th International Conference on New Technologies of Distributed Systems (NOTERE 2007) [94].

- This work presents a media signaling architecture for topologies composed of distributed mixers and controllers. The architecture is based on Megaco/H.248 concepts. The enhancements made on command semantics was presented in detail, and extensively supported by different scenarios. Experiments with a Java application evaluated the overhead generated by the architecture as it controls a conference. This was published in the Proceedings of the 4th IEEE Consumer Communications and Networking Conference (CCNC 2007) [95].

- This work presents a novel application-layer failure detection architecture for MANET conferencing. The architecture is generic enough to be used in any other MANET application. The structure of the overlay network of this architecture was presented in detail, and a specific self-organizing scheme was described. Simulations performed with the OPNET tool demonstrated scalability and reliability of the architecture. This was published in the Proceedings of the 15th IEEE International Conference on Networks (ICON2007) [96].

- The work successfully achieved the integration of the two-overlay architecture with 3GCN architecture for media handling purposes. Using media mediator concepts, the integration maintains the independence of the two networks and sustains media quality. A proof-of-concept prototype to handle media in an MCN environment was used to demonstrate the connection of users on the different sides. This work was submitted to IEEE Network Magazine [97]. It is also the subject of a patent application to the United States Patent and Trademark Office (USPTO) [98].

9.3. Reflections and future work

The results from the experiments presented in this thesis are encouraging, evaluating the effectiveness of the two-overlay architecture in terms of scalability, resource efficiency, stability and preserving satisfactory media quality during a conference. The same is true for the media handling architecture for MCNs in terms of integrating MANETs with 3GCNs. However, there are still some issues that have not been fully addressed, which are identified below.

9.3.1. MANET media handling issues

While the motivational focus of this work has been on the media handling aspects for conferencing in MANETs, it has not been possible to perform large-scale trials for the general architecture within the temporal constraints of this work. This is a clear area for ongoing work and is necessary to address the efficiency of the two-overlay architecture and its philosophy in general as a media handling technique for multiparty applications over MANETs and the new generation of networks.

A number of real-world experiments should be conducted to examine distributed mixing, self-organizing and failure detection behaviors in real networks. These experiments should help to uncover some of the real-life issues that exist in MANETs and that might not be seen in the simulated environments. For example:

- Stricter measurements of the delay and noise that each mixer introduces should be done in order to make stronger recommendations about the optimal number of mixers and controllers allowed in the network.
- The overhead of the general architecture should be measured in real world experiments to study how realistic conferencing is in MANETs and to pinpoint the gourmand components.
- Delays concerning failure detection and recovery should be measured in order to test the responsiveness of the topology to failures in the network.
- In addition, other criteria besides message loss and node failure patterns could be used for controlling the dynamics of the two-overlay network topology. For example, users with similar capabilities might be connected to the same mixer.

Future work plans bring many challenges, some of which are not straightforward extensions of the basic architectures and schemes. However, we believe that the two-overlay media handling design philosophy will remain applicable and effective for conferencing in MANETs.

9.3.2. MCN media handling issues

Multimedia applications and services in MCNs will create diverse new business opportunities in the near future. Conferencing in MCNs is an example of a useful and promising application.

Nevertheless, work remains to be done to enhance the integrated architecture to accommodate more conference types, modes and scenarios. Performance issues should be tackled, especially in terms of media quality. Thus far, we believe that the proposed media handling architecture for MCNs allows media handling for conferencing in several 3GCNs and MANETs. Questions like how many MMs should be deployed and how and where they should be deployed are still unclear. Issues such as can an MM be shared conceptually between many 3GCNs and MANETs are still unanswered. Certainly, the application-level in MCNs, especially MCN conferencing, is a very promising research area and much more remains to be done to solve issues such as:

- Both 3GCN and MANET have different traffic patterns and system capacity. The bottleneck of media transportation under heavy load may be occurring at the MM due to a lack of resources for bridging traffic flow. Admission control may be considered in advance.
- For 3GCN carrier operators, the security issue is very important, as unauthorized users must not utilize the 3GCN resources. Security in MANETs should be developed to safely connect MANET users with 3GCN users in the conference.
- The scale of the conference will be limited by the capabilities and the resources of the MM. How does the MM deal with admission control?
- Most of the situations considered in this work are ideal cases. However, in the real-life situation where network resources could be busy or unavailable, how would the MM respond?

REFERENCES

[1] I. Chlamtac, M. Conti, J.-N. Liu, "Mobile ad hoc networking: imperatives and challenges", *Elsevier, Ad Hoc Networks* 1, pp. 13-64, 2003.

[2] Ram Ramanathan and Jason Redi, "A Brief Overview of Ad hoc Networks: Challenges and Directions", *IEEE Communications Magazine*, vol. 40, Issue: 5, May 2002.

[3] D. Remondo, I. G. Niemegeers, "Ad hoc networking in future wireless communications", *Computer communications*, vol. 26, pp. 36-40, February 2002.

[4] D. Groten and J.R. Schimdt, "Bluetooth-based Mobile Ad Hoc Networks: Opportunities and Challenges for a Telecommunications Operator", *In IEEE VTS 53rd Vehicular Technology Conference (VTS 2001)*, pp. 1134-1138, May 2001.

[5] J. Liu and I. Chlamtac, "Mobile Ad Hoc Networking with a View of 4G Wireless: Imperatives and Challenges", *Mobile Ad Hoc Networking, ch. 1*, New York: Wiley-IEEE Press, July 2004.

[6] R. Pichna, "Interworking Architecture between 3GPP and WLAN System", *IEEE Communication Magazine*, vol. 41, no. 11 Nov. 2003.

[7] D. Cavalcanti, D. Agrawal, C. Cordeiro, B. Xie and A. Kumar, "Issues in Integrating Cellular Networks, WLANs and MANETs: A Futuristic heterogeneous Wireless Network", *IEEE Wireless Communications*, June 2005.

[8] Y.-D. Lin and Y.-C. Hsu, "Multihop Cellular: A New Architecture for Wireless Communication", *IEEE INFOCOM*, vol. 3, pp. 1273-1282, 2002.

[9] A. A. N. Ananda Kusuma and L. L. H. Andrew, "Minimum Power Routing for Multihop Cellular Networks", *In Proc. IEEE GLOBECOM'02*, pp. 37-41, Taipei, Taiwan, 2002.

[10] 3GPP TS 24.147, "Conferencing Using the IP Multimedia (IM) Core Network (CN) Subsystem", June 2005.

[11] A. Uyar, W. Wu, H. Bulut, G. Fox, "An Integrated Videoconferencing System for Heterogeneous Multimedia Collaboration", *7th IASTED International Conference on Internet and Multimedia Systems and Applications*, Honolulu, Hawaii, Aug. 2003.

[12] M. Handley, J. Crowcroft, C. Bormann, J. Ott, "Very large conferences on the Internet: the Internet multimedia conferencing architecture", *Computer Networks*, vol. 31, pp. 191-204, 1999.

[13] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP A Transport Protocol for Real-Time Applications", *IETF RFC 3550*, July 2003.

[14] M. Handley et al., "SDP: Session Description Protocol", *IETF RFC 2327*, April 1998.

[15] F. Cuervo, N. Greene, A. Rayhan, C. Huitema, B. Rosen, and J. Segers, "Megaco Protocol Version 1.0", *IETF RFC 3015/ITU-T H.248*, Nov. 2000.

[16] T. Taylor, "Megaco/H.248: A New Standard for Media Gateway Control", *IEEE Communications Magazine*, Oct. 2000.

[17] G. Bianchi, L. Fratta, M. Oliveri, "Performance evaluation and enhancement of the CSMA/CA MAC protocol for 802.11 wireless LANs", *Proceedings of PIMRC 1996*, Taipei, Taiwan, pp.392-396, Oct. 1996.

[18] I. Chlamtac, A. Farago, "Making transmission schedules immune to topology changes in multi-hop packet radio networks", *IEEE/ACM Transactions on Networking 2 (1)*, pp. 23-29, 1994.

[19] C. Perkins, E. Belding-Royer and S. Das, "Ad Hoc On Demand Distance Vector (AODV) Routing", *IETF RFC 3561*, July 2003.

[20] M. Chatterjee, S. K. Das and D. Turgut, "WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc Networks", *Cluster Computing, Springer Netherlands*, vol. 5, n. 2, pp. 193-204, Apr. 2002.

[21] P.F. Tsuchiya, "The landmark hierarchy: A new hierarchy for routing in very large networks", *Computer Communication Review*, vol. 18, no. 4, pp. 35-42, Aug. 1988.

[22] L. Blasevic, S. Giordano and J.-Y. Le Boudec, "Anchored Path Discovery in Terminode Routing", *Proceedings of the second IFIP-TC6 Networking Conference (Networking 2002)*, Pisa, May 2002.

[23] A. Ahuja et al., "Performance of TCP over different routing protocols in mobile ad-hoc networks", *Proceedings of IEEE Vehicular Technology Conference (VTC 2000)*, Tokyo, Japan, May 2000.

[24] G. Anastasia et al., "A power saving network architecture for accessing the Internet from mobile computers: design, implementation and measurements", *The Computer Journal 46 (1)*, pp. 3-15, 2003.

[25] 3GPP TS 23.002, "Network architecture", Oct. 2005.

[26] R. Tafazolli (Ed.), "WWRF, Technologies for the Wireless Future", Hoboken, NJ: Wiley, 2004.

[27] H. Wu, C. Qiao, S. De, and O. Tonguz, "Integrated Cellular and Ad Hoc Relaying Systems: iCAR", *IEEE JSAC*, vol. 19, 2001.

[28] H. Luo et al., "UCAN: A Unified Cellular and Ad-Hoc Network Architecture", proc. ACM Mobicom, Sept. 2003.

[29] H. Dommel and J. Aceves, "Floor Control for Multimedia Conferencing and Collaboration", *ACM Multimedia Sys.*, vol. 5, no. 1 pp. 23-38, 1997.

[30] H. Schulzrinne and J. Rosenberg, "Signaling for Internet Telephony", *Proc. 6th Int'l. Conf. Network Protocols*, pp. 298-307, 1998.

[31] J. Rosenberg, "A Framework Using Conferencing with the Session Initiation Protocol (SIP)", *IETF RFC 4353*, Feb. 2006.

[32] International Telecommunications Union http://www.itu.int/home/

[33] J. Rosenberg et al., "SIP: Session Initiation Protocol", IETF RFC 3261, June 2002.

[34] H. Schulzrinne, J. Rosenberg, "The Session Initiation Protocol: Internet-Centric Signaling", *IEEE Communications Magazine*, October 2000.

[35] Session Initiation Protocol (SIP), http://www.cs.columbia.edu/sip/

[36] Internet Engineering Task Force (IETF), http://www.ietf.org/

[37] H. Liu, P. Mouchtaris, "Voice over IP Signaling: H.323 and Beyond", IEEE Communications Magazine, October 2000.

[38] H.323, http://www.itu.int/rec/T-REC-H.323/en

[39] H. J. Wang et al., "ICEBERG: An Internet-core Network Architecture for Integrated Communications", *IEEE Personal Communications*, Aug. 2000.

[40] ICEBERG project, http://iceberg.cs.berkeley.edu/

[41] P. Venkat Rangan, H. M. Vin, S. Ramanathan, "Communication Architectures and Algorithms for Media Mixing in Multimedia Conferences", *IEEE/ACM Transcations on Networking*, vol. 1 (1), pp. 20-30, Feb. 1993.

[42] MPEG requirements group, "MPEG-7: Contexts, Objectives and Technical Roadmap, V.12", *ISO/IEC*, Vancouver, July 1999.

[43] A. B. Benitez et al., "Semantics of Multimedia in MPEG-7", *Proc. of International Conference on Image Processing*, vol. 1, pp. 137-140, 2002.

[44] C.-Y. Lin, A. Natsev, B. L. Tseng, M. Hill, J. R. Smith, C.-S. Li "Content transcoding middleware for pervasive geospatial intelligence access", *ICME*, pp. 2139-2142, 2004

[45] V. Balasubramanian and N. Venkatasubramanian, "Server Transcoding of Multimedia Data for Cross-Disability Access", *Proceedings of SPIE*, vol. 5019, pp. 45-56, Jan. 2003.

[46] T. D. Chandra and S. Toueg, "Unreliable Failure Detectors for Reliable Distributed Systems", *Journal of the ACM*, 43(2), pp. 225-267, Mar. 1996.

[47] A. Uyar, W. Wu, H. Bulut, G. Fox, "An Integrated Videoconferencing System for Heterogeneous Multimedia Collaboration", 7th IASTED International Conference on Internet and Multimedia Systems and applications, Honolulu, Hawaii, Aug. 2003.

[48] P. V. Rangan, H. M. Vin, S. Ramanathan, "Communication Architectures and Algorithms for Media Mixing in Multimedia Conferences", *IEEE/ACM Transactions on Networking*, vol. 1, pp. 20-30, Feb. 1993.

[49] S. A. Baset, H. G. Schulzrinne, An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol, *Infocom 2006*, Apr. 2006.

[50] A. Uyar, W. Wu, H. Bulut, G. Fox, "Service-Oriented Architecture for a Scalable Videoconferencing System", *IEEE International Conference on Perusive Services (ICPS'05)*, Santorini, Greece, July 2005.

[51] G. Fox and S. Pallickara, "The Narada Event Brokering System: Overview and Extensions", *proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA '02)*, 2002.

[52] S. Yang, S. Yu, J. Zhou and Q. Han, "Multipoint communications with speech mixing over IP network", *Computer Communication*, vol. 25, pp. 46-55, 2002.

[53] R. Waters et al. "Diamond Park and SPLINE: A Social Virtual Reality System with 3D Animation, Spoken Interaction, and Runtime Modifiability", *In Presence: Teleoperators and Virtual Environments*, vol. 6, No. 4, pp. 461-480, Aug. 1997.

[54] M. Radenkovic, "A Framework for Building and Deploying the Multiparty Audio Service for Collaborative Environments", *In Presence: Teleoperators and Virtual Environments*, Special Issue: VRST ACM 2002, vol. 13, pp. 708-725, Dec. 2004.

[55] M. Radenkovic, C. Greenhalgh, S. Benford, "A Scaleable Audio Service for Collaborative Virtual Environments Over Internet", *TELSIKS 2001*, vol. 2, pp. 455-458, Sept. 2001.

[56] B. Yu, K. Nahrstedt, "A Scalable Overlay Video Mixing Service Model", *in Proceedings of the eleventh ACM international conference on Multimedia: Doctoral Symposium*, pp. 646-647, Berkeley, CA, USA, Nov. 2003.

[57] S. McGlashan, D. C. Burnett, J. Carter, et al., "Voice Extensible Markup Language Version 2.0", *W3C Recomm*, http://www.w3.org/TR/voicexml20/, Mar. 2004.

[58] R. J. Auburn, "Call Control eXtensible Markup Language (CCXML) Version 1.0", *W3C Working Draft 29*, http://www.w3.org/TR/ccxml/, Jun. 2005.

[59] K. Singh, A. Nambi, anf H. Schulzrinne, "Integrating VoiceXML with SIP services", *IEEE International Conference on Comm. 2003*, vol. 2, pp. 784-788, May 2003.

[60] ITU-T Recommendation H.248.1 (05/2002), Gateway control protocol: Version 2 + Corrigendum 1 (03/2004).

[61] F. Cuervo, N. Greene, A. Rayhan, C. Huitema, B. Rosen, and J. Segers, "Megaco Protocol Version 1.0", *IETF RFC 3015/ITU-T H.248*, Nov. 2000.

[62] W. Liao, J.-C. Chang, and V. O. K. Li, "Application-Layer Conference Trees for Multimedia Multipoint Conferences Using Megaco/H.248", *IEEE Transactions on Multimedia*, vol. 7, no. 5, pp. 951-959, Oct 2005.

[63] C. Prehofer and C. Bettstetter, "Self-Organizing in Communication Networks: Principles and Design Paradigms", *IEEE Communications Magazine*, Jul. 2005.

[64] D. Alderson and W. Willinger, "A Contrasting Look at Self-Organization in the Internet and Next-Generation Communication Networks", *IEEE Communications Magazine*, Jul. 2005.

[65] K. Kojima, "Grouped Peer-to-Peer Networks and Self-Organizing Algorithm", *IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, pp. 2970-2976, Oct. 2003.

[66] W. Dou, Y. Jia, H. M. Wang, W. Q. Song, P, Zou, "A P2P Approach for Global Computing", *Parallel and Distributed Processing Symposium*, Apr. 2003.

[67] T. Eymann, M. Reinicke, O. Ardaiz, A. Artigas, F. Freitag, L. Navarro, "Self-Organizing Resource Allocation for Autonomic Networks", *Proc. of 14th International Workshop on Database and Expert Systems Applications*, pp. 656-660, Sep. 2003.

[68] X. Tang and S. T. Chanson, "Optimizing Static Job Scheduling in a Network of Heterogeneous Computers", *International Conference on Parallel Processing*, pp. 373-382, Aug. 2000.

[69] I. Park, Y. Lee and Y. Choi, "Stable Load Control with Load Prediction in Multipath Packet Forwarding,", *15th International Conference on Information Networking*, pp. 437-444, Jan. 2001.

[70] D. Turgut *et al.*, "Balancing Loads in Mobile ad hoc Networks", *10th International Conference on Telecommunications, ICT'03*, vol. 1, pp. 490-495, Feb. 2003.

[71] A. Zhou and H. Hassanein, "Load-Balanced Wireless Ad Hoc Routing", *Canadian Conference on Electrical and Computer Engineering*, vol. 2, pp. 1157-1161, May 2001.

[72] A. Amis and R. Prakash, "Load-Balancing Clusters in Wireless Ad Hoc Networks", *Proc. of ASSET 2000*, Richardson, Texas, pp. 25-32, Mar. 2000.

[73] S. Rieche, L. Petrak and K. Wehrle, "A Thermal-Dissipation-based Approach for Balancing Data Load in Distributed Hash Tables", *29th Annual IEEE International Conference on Local Computer Networks*, pp. 15-23, Nov. 2004.

[74] Y. Zhu and Y. Hu, "Towards Efficient Load Balancing in Structured P2P Systems", *Proc. of 18th International on Parallel and Distributed Processing Symposium*, pp. 20-26, Apr. 2004.

[75] B. Godfrey *et al.*, "Load Balancing in Dynamic Structured P2P Systems", *INFOCOM 2004*, vol. 4, pp. 2253-2262, Mar. 2004.

[76] J. C.Y Chou., "SCALLOP: A Scalable and Load-Balanced Peer-to-peer Lookup Protocol for High-Performance Distributed Systems", *IEEE International Symposium on Cluster Computing and the Grid, CCGrid'04*, pp. 19-26, Apr. 2004.

[77] Shen, T. Yang, L. Chu, J. L. Holliday, D. A. Kuschner, and H. Zhu, "Neptune: Scalable Replica Management and Programming Support for Cluster-based Network Services", *USITS'01*, pp. 197-208, Mar. 2001.

[78] T. Agerwala , J. L. Martin , J. H. Mirza , D. C. Sadler , D. M. Dias, "SP2 system architecture", *IBM Systems Journal*, vol. 34, no. 2, pp. 152-184, 1995.

[79] A. T. Tai, K. S. Tso, W. H. Sanders, "Cluster-Based Failure Detection Service for Large-Scale Ad Hoc Wireless Network Applications", *International Conference on Dependable Systems and Networks*, pp. 805-814, 2004.

[80] Chandra T. D., Goldszmidt G. S., Gupta I., "On Scalable and Efficient Distributed Failure Detectors", *Proc of the twentieth annual ACM symposium on Principles of distributed computing*, Newport, Rhode Island, USA, 2001.

[81] R. van Renesse, Y. Minsky, and M. Hayden, "A Gossip-Style Failure Detection Service", *Proc of Middleware'98*, pp. 55-70, Sep. 1998.

[82] S. M. Hedetniemi, S.T. Hedetniemi, amd A. Liestman, "A Survey of Gossiping and Broadcasting in Communication Networks", *IEEE Networks*, vol. 18, pp. 319-349, 1988.

[83] 3GPP TS 23.333, "Multimedia Resource Function Controller (MRFC) - Resource Function Processor (MRFP) Mp interface", Dec. 2006.

[84] L. Lamport, "Time, clocks, and the ordering of events in a distributed system", *Communications of the ACM*, vol. 21(7), pp. 558-565, Jul. 1978.

[85] G. Neiger, S. Toueg, "Substituting for Real Time and Common Knowledge in Asynchronous Distributed Systems", *ACM*, pp. 281-293, 1987.

[86] T. Omari, G. Franks, M. Woodside, "On the effect of traffic model to the performance evaluation of multicast protocols in MANET", *Canadian Conference on Electrical and Computer Engineering*, pp. 404-407, May 2005.

[87] C. Fu, F. Khendek and R. Glitho, "Signaling for Multimedia Conferencing in 4G: The Case of Integrated 3G/MANETs", *IEEE Communications Magazine*, Aug. 2006.

[88] Java Media Framework API (JMF) home page, http://java.sun.com/products/java-media/jmf/

[89] C. Fu, R. Glitho, R. Dssouli, "A Novel Signaling System for Multiparty Sessions in Peer-to-Peer Ad Hoc Networks", *IEEE Wireless Communications & Networking Conference, WCNC'05*, New Orleans, USA, Mar. 2005.

[90] International Telecommunications Union, ITU-T G.114 Standard, May 2001.

[91] International Telecommunications Union, ITU-T G. 723.1 Standard, 1995.

[92] D. Ben Khedher, R. Glitho and R. Dssouli, "Media Handling for Multiparty Sessions in Ad-hoc Peer-to-Peer Networks: A Novel Distributed Approach", *In the Proc. of the 10th IEEE Symposium on Computers and Communications (ISCC 2005)*, pp. 131-136, Jun. 2005.

[93] D. Ben Khedher, R. Glitho and R. Dssouli, "Media Handling Aspects of Multimedia Conferencing in Broadband Wireless Ad hoc Networks", *IEEE Network Magazine*, Apr. 2006.

[94] D. Ben Khedher, R. Glitho and R. Dssouli, "An enhanced workload-balancing scheme for conferencing in ad-hoc peer-to-peer networks", *NOTERE'07*, Jun. 2007.

[95] D. Ben Khedher, R. Glitho, R. Dssouli, "A Megaco Based-Architecture for Controlling Media Mixers When Conferencing in Mobile Ad Hoc Networks", *In the Proc. of the 4th IEEE CCNC*, Jan. 2007.

[96] D. Ben Khedher, R. Glitho and R. Dssouli, "An Application-Level Cluster-based Heartbeat Failure Detection Architecture for MANETs", *IEEE ICON*, 2007.

[97] D. Ben Khedher, R. Glitho and R. Dssouli, "Media Handling for Multimedia Conferencing in Multihop Cellular Networks", *Submitted to IEEE Network Magazine.*

[98] D. Ben Khedher, R. Glitho and R. Dssouli, "Media Handling for Multimedia Conferencing in Multihop Cellular Networks", *United States Patent and Trademark Office (USPTO)*, Patent Application Serial No 11/840,088, Aug. 2007.

ACRONYMS AND ABBREVIATIONS

API	Application Programmer Interface
3G	Third generation
3GCN	Third generation Cellular Network
3GPP	Third Generation Partnership Project
AODV	Ad Hoc On Demand Vector
API	Application Programming Interface
AS	Application Server
CCXML	Call Control eXtensible Markup Language
CPU	Central Processing Unit
CSE	Call Signaling Entity
CSMA/CA	Carrier Sense Multiple Access With Collision Avoidance
DMA	Distributed Mixing Architecture
FDS	Failure Detection Service
iCAR	integrated Cellular and Ad Hoc Relaying
IETF	Internet Engineering Task Force
IP	Internet Protocol
IRTF	Internet Research Task Force
ITU	International Telecommunications Union
JMF	Java Media Framework
MAC	Media Access Control

MANET	Mobile Ad hoc NETwork
MCN	Multi-hop Cellular Network
MCU	Multipoint Control Unit
MG	Media Gateway
MGC	Media Gateway Controller
MGCM	Media Gateway Controller Mediator
MGCP	Media Gateway Control Protocol
MGM	Media Gateway Mediator
MHAM	Media Handling Architecture for MCNs
MM	Media Mediator
MRF	Multimedia Resource Function
MRFC	Multimedia Resource Function Controller
MRFP	Multimedia Resource Function Processor
P2P	Peer-to-Peer
RTCP	Real-time Transport Control Protocol
RTP	Real-time Transport Protocol
S-CSCF	Serving-Call Session Control Function
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SPLINE	Scalable Platform for Large Interactive Network Environment
TCP	Transmission Control Protocol
TSMA	Time Spread Multiple Access
UACA	User Automatic Control Agent

UCA	User Conferencing Agent
UCAN	Unified Cellular and Ad hoc Network
UDP	User Datagram Protocol
UMA	User Mixing Agent
VoIP	Voice over IP
WCA	Weighted Clustering Algorithm
WiMAX	Worldwide Interoperability for Microwave Access
XML	eXtensible Markup Language