Signaling for Conferencing in Mobile Ad Hoc Networks

Chunyan Fu

A Thesis
in
The Department
of
Electrical and Computer Engineering

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

March 2008

Your file Votre référence
ISBN: 978-0-494-37751-2
Our file Notre référence
ISBN: 978-0-494-37751-2

ABSTRACT

Signaling for Conferencing in Mobile Ad Hoc Networks

Chunyan Fu, Ph. D.
Concordia University, 2008

Mobile Ad hoc NETworks (MANETs) are networks that do not need to be pre-configured. They are composed of transient nodes connected through wireless interfaces. Due to their flexibility, the ease to build and the associated low cost, they are gaining more and more momentum. They are also seen as part of the fourth generation wireless networks. New applications, such as conferencing, are emerging for such networks. Conferencing enables a set of applications such as audio/video conferencing, debating, distance-learning and multi-party gaming. The implementation of conferencing in MANETs is not an easy task due to scarce network resources, heterogeneous devices, frequently changing topology and unstable wireless connections. It challenges each technical aspect of conferencing: signaling, media handling and conference control. Signaling is the control component of conferencing. It handles the session initiation, modification and termination.

In this work, we focus on signaling for conferencing in MANETs. Two types of MANETs are considered: standalone MANETs and integrated MANETs/3G networks. Background information is provided, requirements are derived and the state of the art, including signaling protocols such as SIP and H.323, are reviewed. Since there is no existing solution that meets all of the derived requirements, we propose a novel cluster-based signaling architecture that meets the requirements of signaling for standalone

MANETs. The clusters are application-layer clusters that are dynamically created and deleted for a conference. We also propose a signaling architectures for integrated MANETs/3G networks. The solution is based on conference gateways. We implement the architectures using SIP extensions. Experimental results are obtained from prototypes and OPNET based simulations. In the prototype, we built the signaling system on a small scale network using IEEE 802.11 ad hoc settings. In the OPNET simulation, we use MANET features. From experiments, we found that clustering is a very promising approach for solving signaling problems in MANETs. Being aware of several performance issues of our signaling systems, we further propose optimization schemes that are based on cross-layer design. We also implement some of these schemes and apply them to our signaling systems. The evaluation shows that the schemes significantly improve the signaling performance.

ACKNOWLEDGEMENTS

Pursing a doctoral degree is a journey that one cannot make alone. First, I would like to thank my supervisor Dr. Ferhat Khendek for his ideas, patience, valuable discussions and guidance. He is always so supportive and he has helped me out in every step of my research. Words are inadequate to express my thanks to my supervisor Dr. Roch Glitho. Without his constructive ideas, I would not have gone this far. His enthusiasm in research truly inspires me and his warm encouragements helped me through the hardest time of my PhD studies.

I would like to thank my supervisory committee members, Dr. Chalin, Dr. Agarwal and Dr. En-nouaary, for their advices and comments at different stages of this research. I would like to thank Dr. Dssouli, she always kindly gives me helpful comments and ideas.

I would like to thank my teammates in TSE lab: Dhafer, Fatna, Joana, May, Rajesh, Slimane, Samir, Mohand, Nuru and Arif for their non-reserved help whenever I needed. I would also like to thanks Ericsson staff: Yves Lemieux, Lucie Robert and other people that I do not recall their names. They helped me to solve so many problems during my internship at Ericsson.

I would like to acknowledge the financial support from Concordia University and Ericsson Canada.

Last but not least, thanks to my family, my husband Xiaoniu and my parents. They always unconditionally support me.

Table of Content

List of Figures

List of Tables

Acronyms and Abbreviations

3G	Third Generation Wireless System
3GSA	3G Signaling Agent
4G	Forth Generation Wireless System
AAPA	Adaptive Application Protocol Agent
AODV	Ad hoc On Demand Vector routing protocol
AP	Access Point
AS	Application Server
BAN/PAN	Body Area Network/Personal Area Network
BTS	Base Transceiver Station
CCCP	Conference Control Channel Protocol
CCP	Connection Control Protocol
CGW	Conference Gateway
CPCP	Conference Policy Control Protocol
DSR	Dynamic Source Routing
GGSN	Gateway GPRS Support Node
GPRS	General Packet Radio Service
IETF	Internet Engineering Task Force
IP	Internet Protocol
ITU	International Telecommunications Union
MAC	Media Access Control
MANET	Mobile Ad hoc NETworks
MC	Multipoint Controller
MCN	Multihop Cellular Network
MCU	Multipoint Control Unit
MEGACO	Media Gateway Control Protocol
MH	Mobile Host
MMCC	Multimedia Connection Management

MP	Multipoint Processor
MRFC	Media Resource Function Controller
MSA	MANET Signaling Agent
MSC	Mobile Switching Center
NIA	Networking Information Agent
OLSR	Optimized Link State Routing
PDA	Personal Digital Assistant
RNC	Radio Network Controller
RTP/RTCP	Real-time Transport Protocol/ RTP Control Protocol
SA	Signaling Agent
SCCP	Simple Conference Control Protocol
SCN	Single-hop Cellular Network
SGSN	Serving GPRS Support Node
SIP	Session Initiation Protocol
SP	Share Space
SUA	Super User Agent
TBRPF	Topology Broadcast Reverse-Path Forwarding
TORA	Temporally-Ordered Routing Algorithm
UA	User Agent
UE	User Equipment
WLAN	Wireless Local Area Network
WMAN	Wireless Metropolitan Wireless Network
WAN	Wide Area Network
XCAP	XML Configuration Access Protocol

Chapter 1 : Introduction

1.1 Motivations

Mobile Ad hoc NETworks (MANETs) are networks that do not need to be pre-configured. They are composed of transient nodes connected through wireless interfaces. MANETs have originated from military scenarios where no infrastructure can be pre-deployed. A typical example is the battle field. They have also been proposed for emergency or disaster situations. In the last decade, with the appearance of low-cost wireless interfaces (e.g. IEEE 802.11 and Bluetooth), MANET gained more momentum outside of the military domain and disaster relief situations. In this section, we first present the expected benefits and new uses of MANETs. We then provide the status of MANET researches and MANET applications before we pinpoint out our research focus.

As a self-organizing network without the need of a backbone or a centralized administration, MANET offers unique benefits and versatility for certain environments and applications [1]. First, since there is no fixed infrastructure, they can be created and used anytime and anywhere. Second, the cost of building an ad hoc network is low as the network only involves end-user devices. Third, such networks can be intrinsically fault resilient because they do not operate under the limitation of a fixed topology. Addition and deletion of nodes occurs only by interactions with other nodes, and no additional agent is involved. Fourth, heterogeneous devices can be involved in such networks. This encourages a wide range of users to participate.

Other than the military and emergency applications, the perceived advantages of MANETs elicited more and more interests among the commercial, educational and even the personal domains. Examples of new applications are the mobile office, the distance-learning, the virtual conference rooms, the transmission of road conditions, news, weather and music in vehicles, the multi-user gaming, the anywhere printing and the shows trading. In the wireless communications domain, MANETs have been envisioned as a part of the fourth generation wireless systems (4G). The 4G system integrates the existing systems, such as the second-generation cellular system (2G) and the third-generation wireless systems (3G), with new access-networks so that advanced services can be provided [1]. As a type of access networks, MANETs contribute to their flexibility and high bandwidth.

The flexibility and convenience of MANETs come at a price [2]. The deployment of a MANET faces many challenges, e.g. the heterogeneous devices, frequently changing network-topologies and unstable wireless connections. Research issues have been identified for each of the protocol layers [3], ranging from physical layer to application layer. These issues led into a large amount of research works in the last decade. However, most of the researches have focused on lower-layer issues such as physical [4], MAC [5] and routing [6]. Application layer issues have been less addressed. The existing research works usually assume a general purpose MANET without considering any specific application. Conti et al. [2] show that the lack of realism is the main reason for the insufficient deployment of MANET in reality even after a decade of research.

New services and applications are the driving-force of MANETs. The gap between MANET theory and practice motivated us to investigate how to deploy an application in a real MANET environment. Conferencing enables a wide range of 'killer' applications such as video conferencing, gaming and distance-learning. It allows participants to exchange multimedia streams in real time. It is composed of three parts: conference control, signaling and media handling.

Conference control focuses on the conference management such as policy, floor control and voting. It is not a mandatory part for conference applications. For example, an ad hoc conference may not have a conference control part. In such a conference, the first two parties create a conference arbitrarily and participants can join, exchange media and leave freely.

Signaling is a control component of conferencing. It is concerned with session management including the session initiation, modification and termination. In a multi-party conference, the signaling is also responsible for coordinating the participant information in all the parties. Media handling is a mandatory part of conferencing. Without it, no media can be transferred. It focuses on the techniques of media transmission, trans-coding and mixing. The characteristics of MANETs challenge each technical aspect of conferencing. Our focus here is about signaling.

1.2 Problem statement and goals

As aforementioned, conferencing enables applications such as multimedia games and public debates. These applications may comprise different media types and involve different numbers of participants. They may be public or private, for pre-selected members only. They may be pre-arranged or established in an ad hoc manner.

Figure 1.1 illustrates an ad hoc conference scenario in an airport. Passengers waiting for their flights wish to play a multiparty multimedia game using their handheld devices (laptop, PDA or cell phone). We assume that the game is pre-loaded on their devices, which are connected through IEEE 802.11 wireless cards. Passenger A starts the game by inviting passenger B and they establish an audio session. Then, passenger B invites passenger C to join the session and then informs passenger A that C has joined. Finally, passengers D, E, F and G are invited to join the game and a seven-party game is established. Various media are exchanged among the parties. In this scenario, the signaling system is responsible for establishing sessions, negotiating media types and propagating participant information. The media handling system is responsible for transmitting the designated type of media among participants in real time and performs trans-coding if necessary.

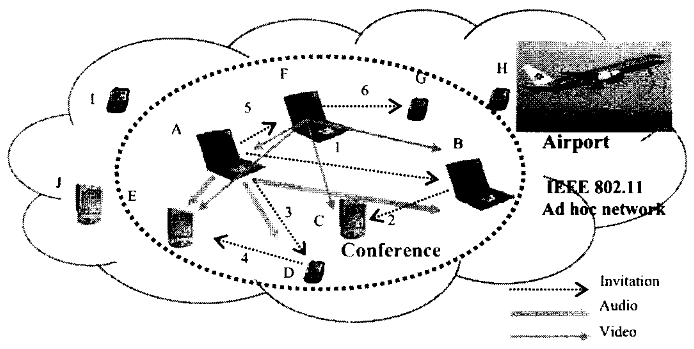


Figure 1.1: An example of conferencing in MANET

From the aforementioned scenario, we can see that deployment of a conferencing-based service in a MANET is not an easy task. First of all, conferencing is resource-demanding. For example, the exchange of multimedia (such as audio and video) streams requires high bandwidth. Meanwhile, the stream transmissions have real-time constraints. Even in an infrastructure-based network, the deployment of conferencing system is not easy. Related issues, such as quality of services, have been discussed for several years.

There is no doubt that the implementation of multimedia conferencing in MANETs is more difficult. The difficulties are presented in two aspects. First, a real-time transmission is not easy because the topologies of MANETs change frequently and the wireless links break frequently. A serious application layer delay happens when there is a delivery failure or there is a re-formation of the network structure in lower layers. Second, the available resources for multimedia conferencing are very limited in MANETs due to the scarce network resources and heterogeneous node capabilities.

As previously introduced, we only focus on the signaling aspect. Our target is to propose signaling solutions that can be used to organize conferences in standalone MANETs and in integrated MANETs/3G networks. We consider these two network architectures because they have a high probability of deployment.

Research challenges related to a signaling system for conferencing can be classified into four categories. The first one is the signaling architecture. In standalone MANETs, the issue is how to set up a signaling architecture in a transient, distributed environment so that a signaling system can support from tens to hundreds and even to thousands of conference participants. In an integrated MANETs/3G networks, the issue is whether the two networks maintain a unified signaling architecture or they maintain different architectures. A further issue is that if different architectures are used, how they inter-operate.

A second category of challenges is associated to session control. The issues are how to handle the frequently changing conference membership (e.g. participants often move out of the network range), how to propagate session-related information (e.g. participant information) and keep its freshness, how to handle the mobility of a conference participant, and how to resume a session when a participant recovers from a temporal link break. In integrated MANETs/3G networks, a further issue is how to setup a session in which participants are located in the two different networks.

A third set of challenges is from the implementation point of view. Several issues have to be considered for the deployment of a signaling system in a real MANET environment. For instance, how signaling entities discover and locate each other, how to find new participants when they are online, and what kind of routing protocol should be selected for the application.

The last category of challenges is related to the performance. We have already seen how performance is important for MANETs. Thus, how to reduce the signaling overhead and how to optimally use network resources and node capabilities are non-trivial issues.

1.3 Contributions of this thesis

The main contributions of the thesis are described as follows, with references to corresponding publications.

- **Signaling requirements for conferencing in MANETs:** Signaling for conferencing in infrastructure-based network cannot be applied to MANETs. In this thesis, we identify the specific requirements for signaling in MANETs (i.e. in standalone MANETs [7] and in integrated MANETs/3G networks [8]). The requirements include those related to the signaling architectures and session management. They also contain the ones related to the optimal use of networks and node resources. Furthermore, the session failure handling, the scale and the performance are considered in requirements.
- **Signaling architecture for conferencing in standalone MANETs ([7],[9],[10],[11], [12]):** We propose a signaling architecture based on clustering. The architecture is different from the traditional, infrastructure-based signaling architectures. It does not rely on permanent central control points. The clusters are created and deleted

dynamically. Related to the architecture, in order to fulfill the requirements, we also propose a solution for session failure detection/recovery and a scheme for node capability exchange.

- **Signaling architecture for conferencing in integrated MANETs/3G networks ([8],[13]):** Based on the signaling solution for standalone MANETs, we propose an architecture for integrated MANETs/3G networks. The architecture is based on conference gateways and it does not require significant upgrade from existing 3G and MANET conferencing entities.
- **Implementation of the architectures**: We selected Session Initiation Protocol (SIP) as an implementation technology for both standalone MANETs ([7],[14]) and integrated MANETs/3G networks ([8],[15]). We propose extensions to SIP so that the signaling entities can support the clustering architecture. We build the proof-of-concept prototype in which we implement every signaling entity that we proposed. We also test the signaling procedures for small scale conferences. In order to further evaluate the architectures, we build up simulations in OPNET [9]. The session delays and session overheads are evaluated for different cases/scenarios. The study shows that the cluster-based architecture is a promising approach for solving the issue and the extended SIP is an appropriate technology to implement the architectures.
- **Optimization of the architectures ([16][17]):** After the identification of several performance issues and weaknesses of the aforementioned signaling architectures, we propose a cross-layer design architecture for optimization. Comparing to other cross-layer design solutions, this architecture is especially appropriate for solving the application-layer issues in MANETs. We define six optimization schemes that are

the examples of applying the architecture. These schemes are independent with each other. Two of them are specific to integrated MANETs/3G networks. By applying part of optimization schemes in the prototype and in the simulation, we found that significant performance gains are obtained without requiring a large footprint overhead.

1.4 Thesis organization

The rest of the thesis is organized as follows.

Chapter 2 presents the necessary background information. We provide an overview of MANETs, including a brief history, technologies and classifications. The solutions of integrated MANETs/3G networks are also presented. Two approaches: clustering and cross-layer design, are often used to solve MANET issues. We introduce them in this chapter because our solutions are related to these approaches. We conclude this chapter, by introducing conferencing and describing conferencing technologies.

Chapter 3 introduces the derived signaling requirements and provides a review of the state of the art. The requirements include signaling requirements for conferencing in both standalone MANETs and integrated MANETs/3G networks. The review of the related work is in light of the requirements. This related work includes the signaling for conferencing solutions from standard bodies (ITU-T and IETF) and other solutions outside of these bodies.

Chapter 4 is devoted to the signaling architecture for standalone MANETs. The architecture is based on application-layer clustering. The general principles, the detailed operational procedures and the signaling scenarios are presented. Special issues such as capability exchange and session recovery are also discussed.

Chapter 5 proposes a signaling architecture for integrated MANETs/3G networks. The architecture integrates the conferencing architectures for standalone MANETs and 3GPP 3G networks. A conference gateway is introduced as a mediator. The principles and signaling scenarios are presented and discussed.

Chapter 6 is devoted to the implementation of the signaling architectures for both standalone and integrated MANETs/3G networks. The implementation is based on SIP and SIP extensions are introduced. In the chapter, we also present the proof-of-concept prototypes and evaluation results.

Chapter 7 evaluates the performance of the signaling architectures through simulations using OPNET. The simulation setups and performance metrics are presented and simulation results are analyzed.

Chapter 8 proposes optimization schemes for the architectures introduced in Chapter 4 and 5. It discusses performance issues, introduces optimization architecture and offers optimization schemes. It also provides some optimization results.

Chapter 9 draws conclusions with a summary of the research contributions and potential future research directions.

Chapter 2 : Mobile Ad Hoc Networks: Architecture, Protocols and Applications

In this chapter, we present the background information that helps to understand the content of this thesis. Four types of information are introduced. First, we introduce the evolution, definition, classification, standards and technical issues of MANETs. Second, a survey of the clustering technologies used in MANETs is provided. Third, we introduce the concept of cross-layer design and its uses in MANETs. Finally, we give an introduction to conferencing and its technical components.

2.1 Mobile Ad Hoc Networks

Mobile ad hoc networks (MANETs) can be defined as transient networks formed dynamically by a collection of arbitrarily located wireless mobile nodes, without the use of existing network infrastructure or centralized administration [3]. They rely on wireless technologies such as IEEE 802.11 and Bluetooth. An important assumption for ad hoc network is the multi-hop routing. Each node in MANETs may play roles of both router and host. Devices in an ad hoc network can be heterogeneous, such as Personal Digital Assistants (PDAs), laptops, palm PCs and cell phones.

2.1.1 A brief history of MANETs

The concept of MANET has originated from military. DARPA Packet Radio Network (PRNet in 1972) [1] can be considered as the first ad hoc network, which features a distributed architecture consisting of network having broadcast radios with minimal central control. In addition, by using multi-hop store-and-forward routing techniques, the radio coverage limitation is overcome. In the following twenty years, ad hoc networking technology has been under sporadic development. The typical scenario was setting up a communication network in a battlefield.

The term "ad hoc network" did not appear until the development of IEEE 802.11 in the 1990s, a standard for Wireless Local Area Networks (WLAN). The IEEE replaced the term "packet radio network" with "ad hoc network". With the adoption of the new name, the IEEE hoped for entirely new scenarios other than traditional battlefield. References [1] and [18] show several examples. New scenarios include disaster-relief networks, construction-site networks, inter-vehicle networks, home-environment networks, emergency networks, networks for visitors at airports, conference centers and shopping malls. The term MANET was first used by IETF as a name of the working group for the researches on IP-based routing protocols for mobile ad hoc networks. It now becomes a widely used term in the domain.

2.1.2 Taxonomy

We introduce two primary classifications for MANETs in this section. The first one is related to coverage area. The second one focuses on the relationships with other networks. In terms of coverage area, ad hoc networks can be classified into four types: Body, Personal, Local, and Wide Area Networks [19] as shown in Figure 2.1. Wide Area ad hoc Networks (WAN) are large-scaled mobile multi-hop wireless networks. They generally interest the military users. On smaller scales, Body Area Network (BAN) is strongly correlated with wearable computers. The components of a wearable computer are distributed on the body (e.g., head-mounted displays, microphones, earphones, etc.), and BAN provides the connectivity among these devices. Personal Area Network (PAN) is a network in the environment around the person. PAN connects mobile devices to other mobile and stationary devices. The typical communication range of a PAN is 10 meters. Wireless LAN has communication range of 100 – 500 meters so it is the solution for home and office automation.

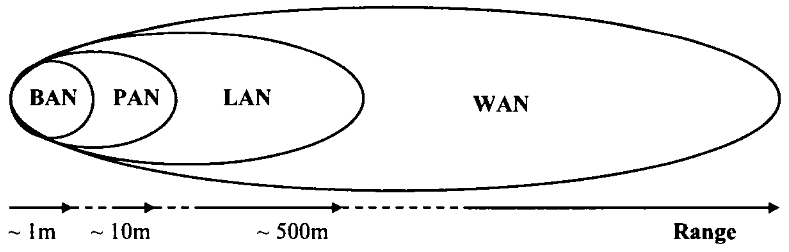


Figure 2.1: Classification of MANETs with respect to coverage

In relation to other networks, mobile ad hoc networks can be classified into standalone ad hoc networks or integrated ad hoc networks [20]. A standalone ad hoc network is a network in which every node only communicates with other nodes in the same networking area. It does not have a connection with other networks, such as Internet. An integrated mobile ad hoc network is a MANET that connects with some infrastructure-based networks, such as 3G networks and Internet. The integration of MANETs and 3G networks is of special interest for us because it is one of the important scenarios in 4G wireless system. The benefit of it includes extending the coverage of the 3G wireless cells and balancing the load between these cells. We introduce the integration technology in the next section.

2.1.3 Integrated MANETs/3G Networks

An integrated MANETs/3G network is also recognized as a type of Multi-hop Cellular Network (MCN) in the wireless network domain. MCN is a concept contrasting with the traditional Single-hop Cellular Network (SCN) such as 2G, 2.5G and 3G wireless networks. Lin et al. [21] first posed the concept. After that, many other research works have contributed to the lower layer connection techniques ([22],[23],[24],[25],[26],[27]), routing strategies ([28],[29],[30]) and mobility management [31].

Cavalcanti et al. [32] summarize the connection alternatives of a MANET and a 3G cellular network. In these alternatives, a general assumption is that users have two wireless interfaces: one to MANET and the other one to 3G. The users may communicate directly through 3G interfaces. They may also communicate directly through MANET interfaces. Furthermore, a user in a MANET may connect to a gateway or a relay, which can establish a connection with another user in 3G network. Another assumption is that MANETs and 3G networks are tightly-connected, i.e. all the users share the same 3G core network. Figure 2.2 shows an example of integrated MANETs/3G networks.

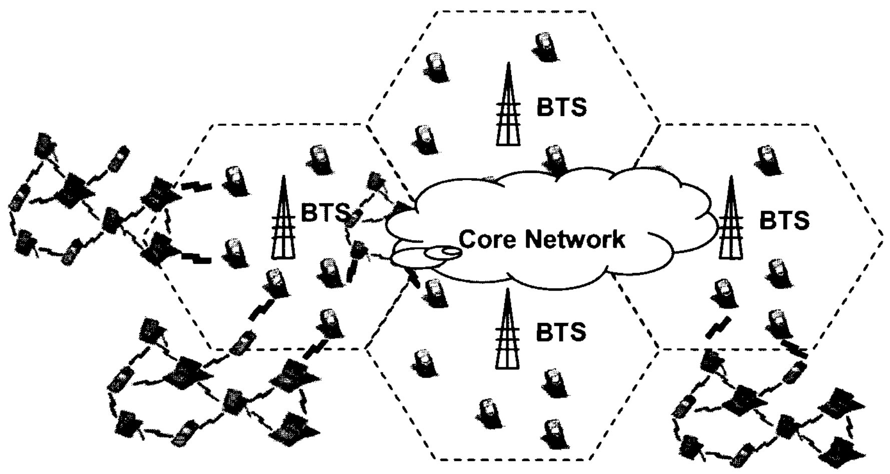


Figure 2.2: An example of Integrated MANETs/3G networks

To illustrate how the integration is concretely done at the lower layers, we use two examples, iCAR [22] and UCAN (a Unified Cellular and Ad-hoc Network) [23]. They take different advantages of the integrated 3G/MANETs and they use different methods for integration.

In iCAR, MANETs are used to balance the traffic load between the wireless cells. An entity, Ad hoc Relaying Station (ARS) is defined to divert the traffic from congested cells to lightly loaded ones. ARSs are wireless devices deployed by the network operator. They have two air interfaces, one to communicate with the cellular Base Transceiver Stations (BTSs) and the other to communicate with Mobile Host (MH) and other ARSs. Three strategies are defined for traffic relaying. First, an ARS directly relays new calls from a congested cell to a neighboring cell. This is called primary relaying. However, if a MH is not close to an ARS, the system will resort the traffics and follow the second strategy, i.e. release the channel from the MHs that are close to the ARSs, relay their traffic to neighboring cells, and allocate the channel to the MH in need. In this case, an MH-to-MH call via ARSs only (i.e. without BTSs involved) is defined. The third relaying strategy called cascaded relaying is the double uses of the second strategy. It covers the situation when both cells, where the calling party and the called party are located, are congested.

In UCAN, MANETs are used to extend the coverage of the wireless cells. The system aims at improving the throughput using multi-hop routing when the quality of the signal in the downlink channel between the BTS and the MH is poor. Dissimilar to iCAR, it does not define the deployed entity. Instead, it uses proxy clients and relay clients to relay packets toward the destination MH. Proxy clients are the MHs that have better downlink signals with the BTS and act as the interface between the MANET and the cellular network. Relay clients are hops to relay the traffic between proxies and destination MHs. In order to find a proxy client, two discovery protocols have been proposed, a proactive greedy scheme and a re-active on-demand protocol. These protocols use the pilot burst information (that reflects the downlink channel condition) collected by the MHs to discover a proper proxy MH.

2.1.4 Technologies and standards

Technologies and standards for MANETs are emerging. In physical and data link layers, there are two standards for wireless PAN and wireless BAN. One is IEEE 802.15.3 [33], which provides high data-rate personal area ad hoc networks. The other is Bluetooth [34], which intends to serve as universal low-cost air interfaces that will replace the plethora of proprietary interconnecting cables between personal devices.

A major WLAN standard is the IEEE 802.11 (or WiFi as marketing term) [35], which is commercially successful and widely used in enterprises and educational organizations. There are two possible settings in the IEEE 802.11. One is infrastructure-based setting in which the Access Point (AP) is defined. The AP is normally connected to a wired network, thus providing Internet access. The other is the ad hoc setting in which nodes are dynamically configured to set up a temporary network. MANETs is only related to this setting. Other standards of WLANs are Digital Enhanced Cordless Telecommunications (DECT), HiperLAN, Infrared WLANs and Ultra Wide Band (UWB). Further details about these technologies can be found in [36].

The standards for WANs are under development. Three of Wireless Metropolitan Area Networks (WMANs) are emerging: IEEE 802.16 (or WiMax as marketing term), ETSI HiperMAN, and WiBro (from South Korea). There is no ad hoc network setting in the current versions of WMAN. However since a MANET built by a WMAN technology especially interests the military users [37], a MANET extension of IEEE 802.16 has been proposed in [38].

At the network layer, IETF MANET working group has standardized four IP routing protocols. They are Ad hoc On-Demand Distance Vector (AODV) routing [39], Optimized Link State Routing Protocol (OLSR) [40] and Topology Dissemination Based on Reverse-Path Forwarding (TBRPF) [41], Dynamic Source Routing Protocol (DSR) [42]. OLSR and TBRPF are proactive link state routing protocols, while AODV and DSR are reactive routing protocols.

2.1.5 Research Challenges of MANETs

Because of the unique characteristics of MANETs, i.e. the absence of an infrastructure, the unreliable network links, the scarce network resources and the mobile, transient and heterogeneous network nodes, research challenges can be found in each of the network layers. Liu et al. [3] summarize these challenges and highlights research issues in each layer as shown in Figure 2.3.

Network Layers	Research issues	Cross-Layer issues
Application Layer Presentation Layer Session Layer	New/Killer Applications Network auto-configuration Location Services Security (authentication, encryption)	Energy conservation, QoS Reliability Scalability Network Simulation Performance Optimization
Transport Layer	TCP Adaptation, Backoff Window	
Network Layer	IP Routing, addressing, Optimization, multicasting	
Data Link Layer	Media Access Control, Error Correction, Optimization	
Physical Layer	Spectrum usage/ allocation	

Figure 2.3: Research issues in MANETs

2.2 Clustering in MANET

Clustering in MANET is a scheme in which MANET nodes are divided into different virtual groups. Each group is called a cluster. Each cluster has a cluster head that acts as a coordinator. Clustering is an important research topic because it makes it possible to guarantee basic levels of system performance, such as throughput and delay, in the presence of both mobility and a large number of mobile terminals [43]. Therefore, it has been widely used for solving different issues such as mobile ad hoc routing [44], topology control [45] and network management [46].

A typical structure of clustering is shown in Figure 2.4 where clusters are formed in accordance with the geographical adjacency of nodes. Cluster-heads serve as local managers and cluster-gateways are non-cluster-heads with inter-cluster links, so they can access neighboring clusters and forward information between clusters. Cluster-members are non-cluster-head node and they do no have inter-cluster links.

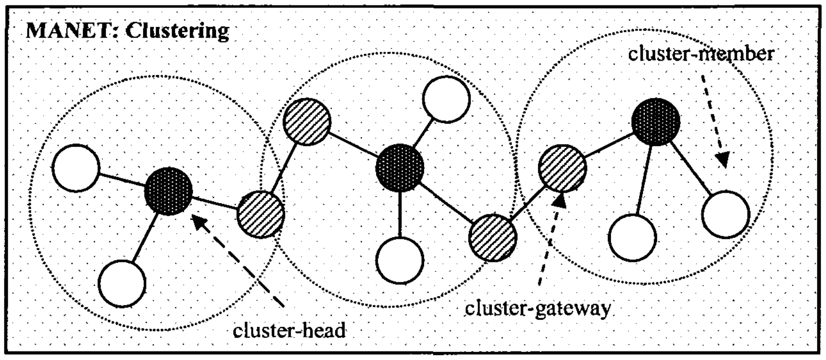


Figure 2.4: An example of clustering in MANET

2.2.1 Classification of clustering schemes

Clustering schemes can be classified according to different criteria. We introduce two of them in this section. The first one is the number of hierarchical levels. A clustering scheme that supports only one level of cluster-head, such as the example shown in Figure2.3, is called flat clustering. A clustering scheme that maintains multi-levels of cluster-heads is called hierarchical clustering. Most of the clustering-based schemes in MANETs, such as those presented in [44], [45] and [46], are flat clustering schemes. Ramanathan et al. [47] present a solution that uses hierarchical clustering. The cluster-heads in the solution are organized into a tree structure. The cost of maintaining a hierarchical clustering scheme is much higher than that of maintaining a flat clustering scheme. However, the hierarchical scheme may significantly enhance the scalability of a system.

Another classification criterion is based on the objectives of clustering schemes. Yu et al. [43] gives a survey to these schemes. In the survey, six types of clustering schemes are presented: DS-based clustering, low-maintenance clustering, mobility-aware clustering, energy-efficient clustering, load-balancing clustering and combined-metrics based clustering. In these schemes, the DS-based clustering is a scheme in which clusters are formed according to geographically adjacent dominating sets. The purpose of DS is to reduce the routing search and routing table maintenance. In mobility-aware clustering, mobile nodes with a relatively low speed are put into a same cluster in order to tighten the connections. The algorithms such as WCA (Weighted Clustering Algorithm) [48] consider multiple metrics in cluster configuration, including node degree, battery energy, cluster size, etc. Moreover, it adjusts its weighting factors for different application scenarios.

2.2.2 Overhead of clustering

Although clustering achieves scalability, its overhead is a non-trivial issue. It may affect the effectiveness and scalability of a MANET. The cost of clustering includes the following aspects. First, explicit messages are required for maintaining a cluster structure. In MANETs, if the network topology changes frequently and nodes are highly mobile, the messages exchanged increase drastically. This will consume a large share of the network bandwidth and device energy. Second, some clustering algorithms (such as the one presented in [49]) may cause ripple effect, i.e. one re-election of a cluster-head brings up the cluster-head elections over the network. Great performance degradations will then happen. Third, the formation of a cluster usually requires a period of time in which no changes to the network structure are made. This enables the exchange of some important cluster parameters. However, the requirement is hard to meet in a realistic scenario.

2.3 Cross-layer design and its uses in MANETs

Cross-layer design refers to protocol design done by actively exploiting the dependence between protocol layers to obtain performance gain. This is unlike layered design, where the protocols at different layers are designed independently [50]. Layered design has obtained great successes in today's infrastructure-based networks such as telephony networks and Internet, while cross-layer design is mainly motivated in wireless networks. Wireless networks have their distinctive characteristics such as user mobility, frequent link failure, limited link capability and limited battery power of mobile devices. The argument is that applying layered design in wireless networks usually causes sub-optimal performances, but a careful cross-layer design helps to solve the issue. A cross-layer design may remove the duplicate functionality or data in layers. It may also optimize the parameters in each layer so that the performance in a single layer or in a whole system is enhanced. In this section, we first introduce different types of cross-layer designs. We then discuss the uses of cross-layer designs in MANETs. We end this section by potential issues in cross-layer design.

2.3.1 Classification of cross-layer design

Srivastava et al. [50] classify the cross-layer designs into four types: interface breaking, merging of adjacent layers, design couplings and vertical calibrations. The interface breaking can further be sorted into three types: upward information flow, downward information flow and back-and-forth. The purpose of vertical calibration is to perform adjustment in each layer in order to achieve a global performance gain. Figure 2.5 illustrate these techniques.

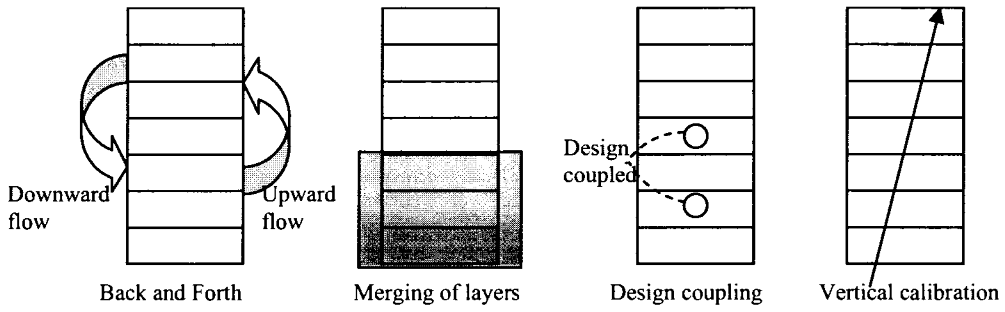


Figure 2.5: Cross-layer design methods

Cross-layer design can also be classified based on how much it is used in system designs. For example, an extreme use of cross-layer design is to discard layers and build a new system [51]. Some proposals apply comprehensive cross-layer design solutions but they still keep layered design in mind. For examples, CLASS [52] provides a 'punch hole' signaling protocol that can be used between any two layers. As another example, MobileMan [53] proposes a share-memory based solution in which each layer contributes to and as well benefits from a shared, network status database. The lowest level of using a cross-layer design is to solve a specific problem in specific layers. For instance, an interaction between routing and middleware is explored in CrossROAD [54].

2.3.2 Cross-layer design in MANETs

Cross-layer is of special interest for MANETs, because the network resources in such networks are very limited and performance optimization is essential. Actually, cross-layer design has been used to optimize performance either globally ([53][55][56][57]) for all the layers or locally for some specific layers ([54][58][59][60][61]) in MANETs. The approach used in each of these proposals can be associated with one of the methods shown in Figure 2.4. We will further discuss some of these proposals in Chapter 8.

2.3.3 Issues in cross-layer design

Cross-layer design is a promising technique that helps to improve performance and reduce overhead. However, unbridled cross-layers can lead to spaghetti design, which can further hinder system innovation and update [62]. We summarize the issues caused by cross-layer design as follows.

The first issue is the design loops. For example, in back and forth interface breaking (see Section 2.3.1), if there is a condition that triggers application layer to call a routing protocol, and meanwhile the routing protocol checks its status and decides to call back the application layer, a loop may then be created. The loop will not end until both the application and the routing functions are blocked. This issue may significantly degrade the system performance. The second issue is that whenever layer independency is broken, it is hard to control the level of dependency between layers. This may introduce the 'ripple effect', i.e. a change to one layer does not only lead to a change to neighboring layers, but also arouse changes to all the other layers. Therefore, it becomes hard to update and add new features to a system. The last issue is related to inter-operability. Since many cross-layer design methods and schemes have been proposed, it is difficult to standardize them and to insure interoperability between the deployed systems.

Because of all these issues, Kawadia and Kumar [62] believe that defining principles, such as keeping a global graph in mind and considering trade offs between system architecture and performance, are essential in cross-layer design.

2.4 Multimedia Conferencing

Multimedia conferencing (also known as multimedia multiparty sessions) can be defined as the conversational exchange of multimedia content between several parties. It consists of three components: conference control, signaling and media-handling. Conference control is related to conference policies, admission control, floor controls and voting. Signaling is used to set up, modify and tear down sessions. It is required before, during and after the media transmission. Media handling is concerned with the transmission and the mixing of the media streams. It also performs the possible trans-coding between different types of media streams.

2.4.1 Classification of conferencing

There are many classification criteria for conferencing. The most commonly used are presented in this section.

Conferences can be with or without floor control. Floor control is a technology that deals with conflicts in shared work spaces [63]. It coordinates the concurrent usage of shared resources and data among participants of a conference. A typical example of floor control is the management of the turn of speaking in a conference, i.e. how and when to allocate the audio channels to involved parties in order to ensure fairness and avoid collisions. Conferences can be pre-arranged or ad hoc. A pre-arranged conference starts at a pre-determined time and is sponsored by specific parties. The duration of a conference may also be pre-defined. An ad hoc conference starts when the first two parties decide to create a session. Parties may join and leave during the conference, and it ends when the last two parties leave.

Another criterion is whether the conference is private (closed) or public (open). A closed or private conference does not welcome parties to join freely. Only the parties who are invited by the conference participants can join. An open or public conference on the other hand publishes its information to all parties in a network. Any party can join the conference if and when it wishes. Yet another criterion is whether the conference has sub-conferencing capabilities. The sub-conferencing capability simulates a conference with different rooms as in the real world. In each room, called a sub-conference, parties can hear/see each other, but they cannot hear/see others that are in different sub-conferences.

The remaining commonly used criterion is the topology used for signaling and media handling. Schulzrinne et al. [64] have discussed four main topologies for signaling and media handling: end-system mixing, full mesh, multicast and centralized. In end-system mixing, one of the participants in the conference does the mixing for all the other participants. In general, due to the limited capability of participants, very few participants can be supported in this type of conferences. In full mesh, every end-system does its own mixing and has a signaling link with every other end-system. Multicast is an enhanced form of full mesh. Every end-system still does its own mixing. However, packets are sent to a multicast address instead of being sent point-to-point. In centralized conferences, a conference bridge is defined to do mixing for all the end systems. Each end system has a direct signaling connection with the bridge. In this model, a participant may either call the bridge to join a conference (dial-in) or be called by the bridge to join (dial-out). A similar but more recent classification has been presented in IETF RFC 4353 [65]. Three models are defined and different names are used, loosely coupled conference (use of multicast), tightly coupled conference (centralized model) and fully distributed conference (full mesh model).

2.4.2 Techniques and standards

IETF and ITU-T have elaborated standards for each aspect of conferencing. We present them one by one.

Conference control has been defined by ITU-T in T.120 Series [66]. The control policies are mainly focused on centralized conferences. From a historical point of view, early conference control contributions in IETF are based on loosely coupled, multicast conferences. Protocols such as MMCC [67], Agreement Protocol [68] and CCCP [69] were defined. SCCP (Simple Conference Control Protocol) [70] was the first draft that tried to map loosely coupled conference into ITU-T T.120 series. However, it still relies on multicast. Recently, IETF has changed its focus from loosely coupled to tightly coupled conference, known as a Common Conference Information Data Model for Centralized Conferencing (XCON [71]). It also defines Conference Policy Control Protocol (CPCP) [72] for centralized conference control.

Signaling protocols have also been defined by both ITU-T and IETF. The most widely applied signaling protocols are H.323 [73] from ITU-T and Session Initiation Protocol (SIP) [74] from IETF. H.323 is a set of specifications. It is actually the very first set of signaling standards created after SS7 (Signaling System 7 that is used for circuit switching networks). Liu et al. [75] provide an overview. H.323 defines four entities: terminal, gateway, gatekeeper and Multipoint Control Unit (MCU). Basic signaling and media handling functions of H.323 are located in terminals. A gateway is a component that bridges H.323 sessions to other types of networks. A gatekeeper, although it is not a mandatory entity, may have many functions such as user admission, zone management, bandwidth control and address translation. The specifications of H.323 cover more than signaling. The H.323 protocols are binary encoded and include three different signaling protocols: Registration Admission and Status (RAS), H.225 and H.245. RAS is used between end-points and gatekeepers. It enables gatekeeper discovery and registration/un-registration of end-points with gatekeepers. H.225 is the protocol for call establishment and teardown. H.245 enables media capability negotiation.

SIP is specifications including a baseline protocol and a set of extensions. In baseline protocol, it defines four entities: *User Agent (UA), proxy server, location server and registrar*. Session control functions are located in UAs. SIP servers are non-mandatory entities that help to route SIP messages and to locate SIP user agents. Schulzrinne et al. [76] give an overview of SIP. SIP is lightweight and extendible and it has been adopted by the two main standards bodies for 3G networks (i.e. 3GPP and 3GPP2) as the sole signaling system for multiparty sessions. It is a text-based request/reply protocol. We will further discuss H.323 and SIP in terms of their uses for conferencing in Chapter 3.

The most widely used media transmission protocols are RTP and RTCP, which are defined by IETF RFC 3550 [77] and RFC 3551 [78] respectively. They are also included in H.323 series as real-time media transport protocols. RTP provides end-to-end network transport functions that are suitable for applications that transmit real-time data, such as audio, video or data, over multicast or unicast network services. It supports the use of translators and mixers. RTCP allows monitoring of the data delivery of RTP.

The Media Gateway Control Protocol (or Megaco) is a signaling protocol between conference signaling and media handling. It is defined by IETF RFC 3525 [79] and ITU-T H.248.1 [80]. The idea is a separation of call control and media handling logics, adding the control commands between them. This separation introduces more flexibility for deploying multimedia conference services. The 3GPP has also used this protocol for conferencing. We will further discuss the conference model of 3GPP in Chapter 3.

Chapter 3 : Signaling for Conferencing in MANETs: Requirements and Evaluation of Existing Solutions

The particularities of MANETs make signaling for multimedia conferencing very challenging. A signaling scheme does not only need to establish, modify and terminate sessions, but it also has to consider the network statuses such as the lack of infrastructure, the frequently changing participants and the limited resources. In this chapter, we derive signaling requirements for conferencing in both standalone MANETs and integrated MANETs/3G networks. We also review existing signaling solutions in light of these requirements. The related work is organized into two categories: the work from standard bodies (the ITU-T and the IETF) and the proposals from outside of these bodies.

3.1 Signaling requirements for MANETs

Due to the infrastructure-less, the first requirement is that none of the involved entities can be a permanent or static central-control point. The second requirement is that the system should be able to dynamically propagate conference-related information (e.g. who joins, who leaves) to all the involved parties. This is not an easy task, as conferences are normally very dynamic in MANETs. Parties can join and leave at any time and very frequently. A party may leave the conference when it decides to do so or when it is forced to because it has moved out of the coverage area or its battery power is used up. In this thesis, we term the first case (which is general to all networks) "voluntary departure" and the second (which is specific to MANETs) "unintentional departure". A signaling system for multimedia conferencing in MANETs should treat both situations as normal cases and handle them gracefully. If a party in a conference temporally moves out of range or if its link breaks for a very short time, the sessions that it has maintained should be recovered after its connections are recovered.

The third requirement is scalability. A high level of scalability is expected from MANETs, especially in military applications. The signaling system should scale automatically and in the same manner as the network in which it is implemented.

Due to the heterogeneity of MANET nodes and the limited network resources, two more requirements are derived. One is that the system should be lightweight. The other is that the use of the resources available to the system should be optimal.

The last requirement is the independence from the lower layer protocols. There is a plethora of lower layer protocols in MANETs. The signaling system is at the application level and it cannot afford to rely on the features of specific lower layer protocols because these lower layer protocols may not be available in some environments.

3.2 Specific requirements for integrated MANETS/3G networks

In the case of integrated MANETs/3G networks, in addition to the aforementioned requirements, four more requirements are derived. One is related to the basic signaling functionality, i.e. a signaling system should support different conferencing scenarios: all participants in 3G networks, all participants in MANETs, or some participants in 3G networks and some others in MANETs. Another requirement is related to performance issue. That is, when a signaling scheme is used, system performance should not depend on the network (3G or MANET) where the participants are located. For example, participants in the MANET should not be forced to connect to a far away 3G conference control center because this will certainly degrade performance in terms of response time. The third requirement is that signaling scheme should be backward compatible with the existing signaling system in 3G. This means that the nodes that support the new signaling system should still be able to establish sessions with nodes that have not yet been upgraded. In addition, the changes for upgrading the legacy signaling entities in 3G should be minimal. The last requirement is that the system should be able to locate participants in both MANETs and 3G networks. This is necessary to route signaling messages to correct destinations.

Table 3.1 summarizes all the signaling requirements for conferencing in MANETs. In the table, R1 to R6 are general requirement for conferencing in MANETs, and R7 to R10 are extra requirements for integrated MANETs/3G networks.

Signaling requirements for MANETs	
R1	No permanent central entity
R2	Dynamic session control
R3	Scalability
R4	Lightweight
R5	Optimal use of resource
R6	Independent of lower layer protocols
Specific signaling requirements for integrated MANETs/3G networks	
R7	Handling sessions with participants in both MANETs and 3G networks
R8	System performance does not depend on the network where the participants are located
R9	Backward compatibility
R10	Capability of locating participant in both MANETs and 3G networks

Table 3.1: Signaling requirements for conferencing in MANETs

3.3 Related Work

Nine signaling schemes for conferencing are introduced and reviewed in this section. We present them in three subsections: signaling protocol from ITU-T, signaling protocols from IETF and signaling solutions from outside of standard bodies.

3.3.1 Signaling protocol from ITU-T

Multimedia conferencing over packet-switched networks is specified by ITU-T, as a subset of the H.323 series of recommendations [73]. We have introduced the entities of H.323 in Chapter 2. Multimedia conference control in H.323 is done via MCU. An MCU can be further divided up into two entities: Multipoint Controller (MC) and Multipoint Processor (MP). MC handles signaling while MP handles media. MP is an optional entity. It is not required in a decentralized conference model in which media are distributed through multicast. MC is mandatory. It is a central control point for both centralized (i.e. where media mixing is done in a central MP) and decentralized models. The conference models defined in H.323 are shown in Figure 3.1.

H.323 does not meet our first requirement (R1) because it has a central controlling entity – MC. In addition, MC can easily become a bottleneck and hinder scalability. Furthermore, H.323 is complex and heavy. Optimal use of resources is not taken into account. On the other hand, H.323 is independent of the lower layer protocols.

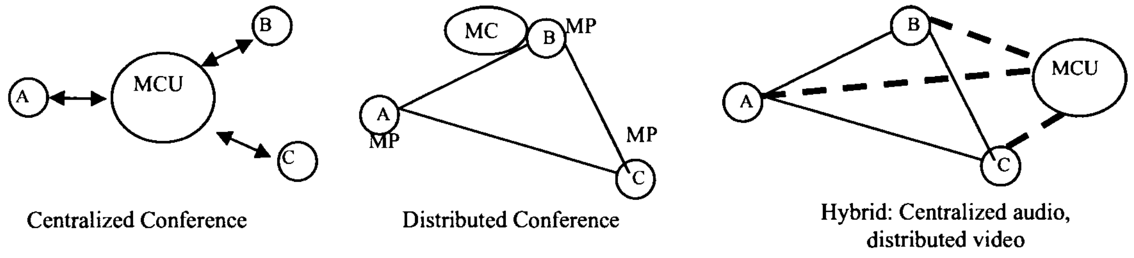


Figure 3.1: Conference models for H.323

3.3.2 Signaling protocols from IETF

In this section, we discuss two signaling protocols from IETF: Connection Control Protocol (CCP) and SIP. For the latter, we further discuss its uses in different conference models: loosely coupled model, tightly-coupled model and fully distributed model.

The first conference approaches proposed by IETF date back to the early 1990s. A common property of these approaches is multicast. These approaches usually do not explicitly use a signaling protocol for session management. However, among them the CCP is an exception. It is a transaction-based protocol that is used to create sessions and refresh participant status. It is used together with the Multimedia Connection Management system (MMCC). Schooler et al. [67] give an introduction to MMCC and CCP. Only one conference scenario is considered by CCP, i.e. the conference initiator creates a conference. The initiator has the responsibility to create a session with each of the conference participants. A list of participants is pre-established via the conference initiator's user interface. The conference initiator is also responsible for propagating conference-related information. If we evaluate CCP in terms of our requirements, we find that there is no permanent centralized entity, but because a conference is controlled by an initiator (an end system), CCP cannot scale to conferences that contain a large number of participants. Other requirements cannot be fulfilled. For example, CCP requires the network layer to support multicast.

IETF has been working on new approaches since early 2000, as part of the work on SIP. SIP has been used for three conference models — loosely coupled, tightly coupled and fully distributed.

A loosely coupled conference is based on multicast. IETF draft [70] describes SCCP, a loosely coupled conference control protocol that uses SIP as the signaling protocol. The signaling architecture is centralized. Signaling messages are exchanged between a controller and a participant through multicast. If we evaluate SIP within SCCP, it does not meet most of our requirements, e.g. there is a permanently central point and multicast is required at the network layer.

A tightly coupled conference is central-server based. Rosenberg et al. [65] defines the SIP usage in this sort of conference model. SIP creates sessions between each participant and a conference focus (i.e. the central server). This conference model is also used by 3GPP [81]. Fig 3.2 shows the 3GPP conference architecture. In the architecture, the conference focus can be implemented in the Media Resource Function Controller (MRFC) and/or in the conferencing Application Server (AS). The MRFC is the functional entity that handles signaling. It considers the Media Resource Function Processor (MRFP) as a default media mixer. The AS hosts conference applications. Any party that wants to participate in a conference should either invite the conference focus, i.e. following the dial-in model, or be invited by the conference focus, i.e. following the dial-out model. In addition to the MRFC and the AS, there is another functional entity, User Equipment

(UE). A UE is a conference participant that has the required conferencing functionality in the end-users' terminals.

This conference model does not meet our first requirement because it has a pre-configured central entity. However, because of SIP, it is lightweight. Another requirement met is that it is independent of lower layer protocols. The remaining three requirements for isolated MANETs are not met. As a 3G conference model, we also review it with respect to the signaling requirements for integrated 3G/MANETs. It does not meet the first requirement because its current version does not support sessions that include MANET participants. A conference focus may be extended to support MANET parties (by adding the functionality of discovery and management of MANET users). This extension does not require a big effort and after the extension, it is possible for the signaling system to be backward compatible. However, the second requirement is not met, i.e. a MANET user is forced to connect to a far away server.

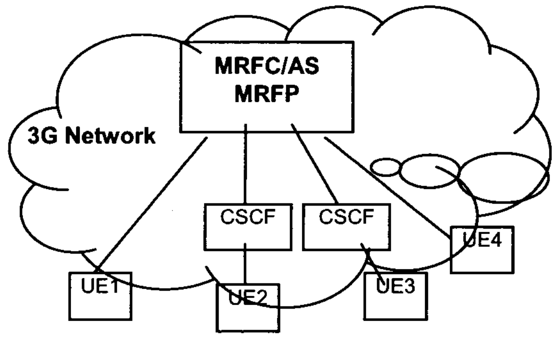


Figure 3.2: 3GPP 3G Conference Architecture

SIP has also been discussed for a fully distributed model. In the model, each participant maintains a SIP session with other participants. Kelley et al. [82] describe this approach in detail. The approach is of special interest for us because it only involves SIP end systems (UAs) and no server is required. Thus, the first requirement - no permanently central point - is met. In addition, the approach is lightweight because there is no extra control message added to the base-line SIP, but the session-related information is carried by the basic SIP messages. However, this approach has several limitations. A first example is the way that the session-related information is dynamically propagated to parties. There is a problem when two (or more) parties are invited to join an on-going session at the same time. There is no general solution to ensure that each of the invited parties is made aware of the other invited parties. The problem is identified in [82] as the 'coincident joins problem', and no solution is provided. A second example is that unintentional departures are not considered and are not handled gracefully. The approach does not scale and the number of signaling connections increases exponentially with the number of participants in the conference. Furthermore, it has not considered the optimal use of network resources.

3.3.3 Approaches from outside of the standard bodies

Several approaches have been proposed from outside of the standards bodies. Some of them target specific issues that are not solved by the standard approaches while others propose more comprehensive solutions. The works proposed in [83] and [84] are examples of the former while GlobalMMCS [85] and ICEBERG [86] are examples of the latter.

The framework defined in [83] is the only work that we can find for conferencing in MANETs. It is a SIP-based solution that addresses the "coincident joins problem" identified in [82]. It proposes a conference leader that propagates session-related information to all participants. The 'coincident joins problem' is then solved. However, it still has other issues such as scalability due to the fully distributed structure. As a solution in MANETs, we also evaluate it in light of the requirements for integrated MANETs/3G networks. We find that the solution does not meet any of our requirements. First, it does not handle the sessions with 3G participants. Second, it is difficult to extend to support a session with 3G participants because its architecture is very different from the existing 3G conference architecture.

The SIP-based conference defined in [84] is another example of work for using SIP for conferencing. It extends the tightly coupled conference model of SIP in order to improve the scalability. Multiple conference focuses are proposed and each focus manages a set of local participants. The conference focuses are inter-connected and form a tree structure. This work solves the scalability issue of the work defined in [65], but it does not solve other issues. For example, the approach relies on pre-configured central entities.

GlobalMMCS [85] is designed to bridge H.323, SIP, Access Grid clients and 2.5G/3G cellular phones in audio-visual collaborative applications. The system makes use of a publication/subscription-based message delivery middleware, the NaradaBroking overlay network. As far as multimedia conferencing is concerned, the system borrows the ideas of MCU, MC and MP from H.323. However, unlike H.323, the MCs can be distributed. There can be several in the same conference, each one managing a sub-set of participants. Although this distribution brings more scalability, our first requirement is still not met, because each MC is a pre-configured central entity for the sub-set of participants it manages.

ICEBERG signaling [86] proposes a signaling system for the management of dynamic and multi-device multiparty sessions. Unlike the other signaling protocols such as SIP, it is a signaling system that is directly designed for multimedia conferencing. Two entities are defined: call agent and call session. They are both dynamic entities created during call session establishment. The call session entity is the control center that manages all the information related to that session. There is one call agent per party. It manages the information related to that party. Changes related to the session are propagated as follows. A designated serving call agent periodically receives a report from each party in the session, and forwards the report to the call session entity. The call session entity maintains the states of all of the parties in a table, and updates the table when it receives the reports. It also propagates the information to each of the call agents. If we review the ICEBERG signaling, we can find that it does not have pre-configured central control entities, thus the first requirement is met. However, the system does not take into account the heterogeneity of the devices. Session-related information is dynamically propagated to the parties. However, it does not consider the unintentional departures. It is also rely on the routing layer multicast. Another advantage, the system scales.

3.4 A review of related work

Table 3.2 shows a review of the existing signaling solutions. We can find that no signaling solution can meet all of our requirements for standalone MANETs. Furthermore, the related signaling schemes in MANETs and in 3G networks cannot meet our requirements for integrated MANETs/3G networks.

	H.323	CCP	SIP (loosely Coupled)	SIP (3G conf-model)	SIP (full-mesh)	ICEBERG	GlobeMMCS	SIP server distribution	SIP framework for MANETs
Signaling requirements for MANETs									
R1	No	Yes	No	No	Yes	Yes	No	No	Yes
R2	No	No unintentional departure detection and session recovery	No unintentional departure detection and session resume	No unintentional departure detection and session resume	No unintentional departure detection and session resume, has concurrent join issues	Yes, but no session resume	No	No unintentional departure detection and session resume	No unintentional departure detection and session resume
R3	middle	small	middle	middle	small	Large	Large	Large	Small
R4	No	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
R5	No	No	No	No	No	No	No	No	No
R6	Yes	No, use multicast	No, use multicast	Yes	Yes	No, use Multicast	No, use Naradabroking	Yes	Yes
Signaling requirements for integrated MANETs/ 3G Networks									
R7	-	-	-	No, no MANET users supported	-	-	-	-	No, no 3G users supported
R8				No, performance will depend on networks					Hard to be extended to support 3G users
R9	-	-	-	Yes, possible. Extension effort is low	-	-	-	-	No, hard to be extended
R10				No, cannot location MANET users					No, cannot locate 3G users

Table 3.2: A critical review of related work

On the other hand, some of the aforementioned signaling solutions do provide us valuable ideas. For examples, the ICEBERG presents an interesting solution for controlling dynamic sessions based on soft states, and the SIP server distribution proposes a division of signaling control into sub-servers. With these ideas in mind, we propose a novel signaling system – a cluster-based signaling architecture, which is not a pre-configure architecture but it is dynamically formed and deleted. We present it in Chapter 4.

Chapter 4 : A Cluster-based Signaling Architecture for Conferencing in Standalone MANETs

Clusters enable scalability without centralization and we believe that they can help in solving the signaling issue in MANETs. In this chapter, we present a cluster-based signaling architecture for conferencing in standalone MANETs. The clusters are formed in the application layer and only when there is a conference. We first present the architectural principles, followed by a description of the clusters' operational procedures. We then discuss two critical issues related to the operational procedures – how to exchange node capabilities and how to handle unintentional departure and session recovery. In the last section, we will analyze the potential of the architecture with respect to our signaling requirements.

4.1 Architectural principles

4.1.1 Network assumptions and conference type

A MANET in this thesis refers to the network in which a node can be reached by any other nodes through a direct wireless connection or through multi-hops. The capability of a MANET node does not refer to any specific capability such as battery power or memory or processing power, but it is a comprehensive capability value. We assume that a node always 'knows' its capability value, and we do not consider what factors are used to calculate the value or how to calculate it.

All the scenario examples and signaling procedures in this chapter are in ad hoc conferences. Such conferences fit quite well in MANET settings and they are the basis of numerous applications such as gaming. It should be noted that the proposed signaling architecture can be applicable for different types of conferences, such as private/public conferences, pre-arranged/ad hoc conferences and conferences with/without floor control, but we do not consider sub-conferencing.

4.1.2 The architecture and general principles

Figure 4.1 gives an overall view of the proposed cluster-based signaling architecture. The only functional entity is the Signaling Agent (SA). There is one per party, or more generally, one per node in a MANET. They are grouped in clusters that we call signaling clusters. These clusters are application-level clusters and are independent of lower layer clusters such as routing clusters. In each cluster, at any given time, there is one and only one cluster head (we refer to as super-member), and all the other members of the cluster are connected to it. A super-member has direct signaling links to the super-members of the neighboring clusters.

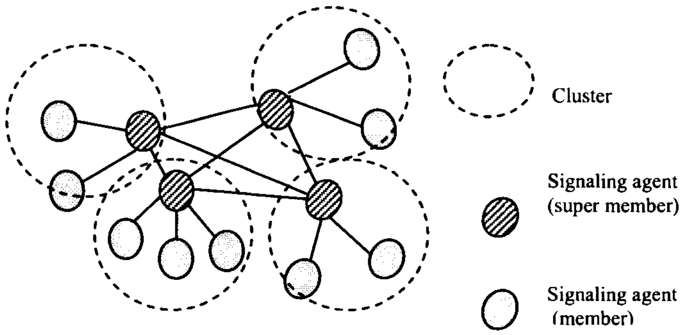


Figure 4.1: A signaling architecture for standalone MANETs

There are two general parameters of a cluster: Split value (Sv) and Merge value (Mv). Every node in a conference maintains the same Sv and Mv. If the size of a cluster reaches

Sv, the cluster will split into two clusters. If it reaches Mv, the cluster will find another cluster to merge with.

A super-member is responsible for keeping track of the information of its members and its neighboring super-members. It also propagates the information when there is a change in membership. In addition, it detects the eventual unintentional departures of the nodes connected to it by sending periodic heartbeat messages.

In our architecture, it is the node with the most capabilities that is elected as the super-member. A participant that initiates a conference is responsible for collecting the capabilities of the called party before the conference is initiated. Super-members keep track of the capability changes of their members and neighboring super-members during the conference.

4.2 Operational procedure of clusters

In our signaling architecture, clusters are dynamically created and deleted for conferencing. The signaling system is responsible for maintaining the state of conference and clusters. Each signaling cluster has a life cycle. The first phase is its creation. A super-member is elected in this phase. After its creation, the cluster moves to an active phase. The membership of the cluster evolves (parties join and leave). These changes may lead a cluster to split into two, or to merge with another cluster. Ongoing activity may also lead to the election of a new super-member, triggered by the departure of the super-member, for example. The life cycle ends with the deletion of the cluster. In this section, we describe the signaling procedures related to each of the phases of the cluster life cycle.

4.2.1 Cluster creation and deletion

The first cluster is created when a conference starts. The creation procedure is as follows: first, the party (called the initiator) that wishes to establish a session collects the capability of the called party. It compares its own capability to the capability of the called party and designates the one with more capability as the super-member. Second, it requests the super-member (itself or the called party) to create a session. The initiator needs to set the Sv and Mv and passes the parameters to the called party. After the first session is set up, the super-member starts to periodically collect the capabilities of its members.

The last cluster is deleted when the last two parties leave the session. All the states and parameters of the cluster are cleared.

4.2.2 Super-member election

An election algorithm is used whenever there is a need to select a new super-member among several candidates. This happens when a new cluster is created or when a cluster merges with another cluster or when a super-member leaves. The basic rule is that the candidate with the most capability is selected as a super-member. We assume that a party that wishes to select a new super-member knows a list of *n* candidates $\{candidate_1 \ldots candidate_n\}$. Assuming that the respective capabilities of these candidates are $\{Cap_1 \ldots Cap_n\}$, the algorithm is given in Table 4.1.

```
1.  $Cap_{max} \leftarrow Cap_1$
2.  *loop* i (from 2 to n)
3.    *if* ($Cap_{max}$ is less than $Cap_i$)
4.      $Cap_{max} \leftarrow Cap_i$
5.      max $\leftarrow$ i
6.    *end*
7.  *end*
8.  return $candidate_{max}$
```

- Cap_{max}: Maximum capability
- $candidate_{max}$: Candidate with most capability
- $\{candidate_1 \ldots candidate_n\}$: n candidates of super-member
- $\{Cap_1 \ldots Cap_n\}$, capability of n candidates

Table 4.1: Super-member election

4.2.3 Member joining and leaving

Both members and super-members can invite parties to join a conference. If it is a super-member that is inviting and it is capable of handling more members, the super-member directly establishes a session with the party. If the super-member cannot handle more members, it may ask a neighboring super-member to do so. If a member invites a party, that member will ask its super-member to establish the session. A new member is then added to the cluster. The super-member of the cluster propagates the membership change to neighboring clusters.

Any participants, including members and super-members, may leave a conference whenever they want to. In the case of a member departure, the member terminates its connection with its super-member and the super-member propagates the membership change to the neighboring clusters. With the departure of a super-member, that super-member designates a new super-member (choosing the member with most capability among the member list) before leaving. It passes its member-list and neighboring super-member list to the new super-member. The new super-member sets up a session with each member and each neighboring super-member and forms a new cluster. After this procedure, the old super-member terminates all its connected sessions. In the case where there is no member in a cluster, the super-member that wishes to leave simply terminates all its connected sessions. An example of super-member departure is shown in Figure 4.2. In the figure, the super-member of *c1* leaves the conference.

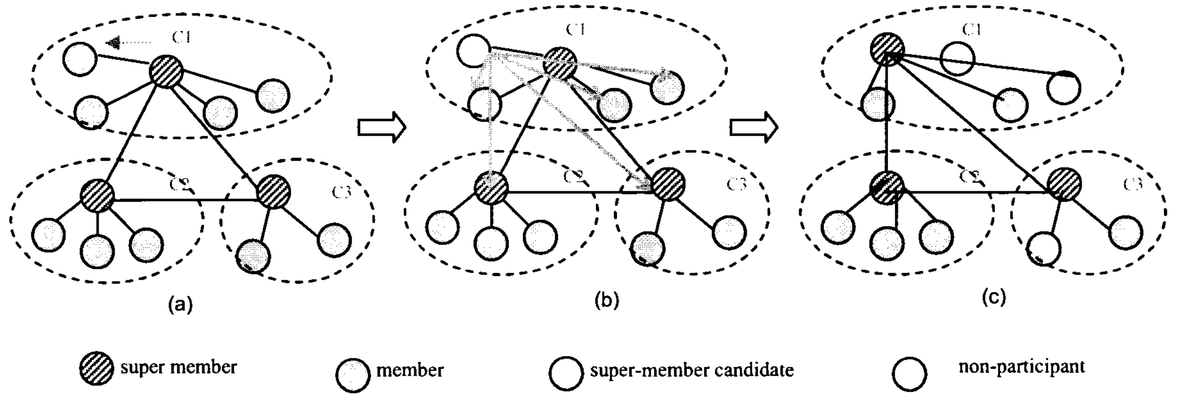


Figure 4.2: Super-member leaving (a) designate a super-member candidate, (b) the candidate creates connections, (c) old super-member leaves, the candidate becomes super-member

Frequent changes of clusters' super members can lead to instability. The clustering algorithm described in [87] has proposed rules to make sure that the cluster head changes as less frequently as possible. Similarly, we set as a rule that when a party joins a cluster, it does not replace the super-member, even if it has more capability. This helps to maintain the stability of the clusters, but does not prevent the super-member from leaving, if it decides to do so.

4.2.4 Splitting

When a new member is added to a cluster, the super-member initiates a split procedure if the size of the cluster reaches *Sv* or if the super-member does not have enough capability to handle more members. A cluster may also split when its super-member does not have enough capability to handle its existing members. This happens, for instance, when the battery power of the super-member decreases. First, the super-member selects a new super-member, based on capabilities. It also selects half of its members that are to become members of the new cluster. The selection may be by random or according to some rules, such as the sequence number of members. Table 4.2 shows a selection procedure based on odd/even sequences of cluster members. We assume that there are *n* cluster members in the cluster member-list *Cm* and the list is denoted as $\{Cm_1, \ldots, Cm_n\}$. A new super-member has been selected using the super-member selection procedure defined in Section 4.2.2 and the new super-member has been removed from the member list *Cm*. The selected members are stored in an empty member-list *Cm'*.

```
if(n<=1)
    return Cm'
else loop i (from 1 to n)
        if(i mod 2 == 0)
            remove $Cm_i$ from Cm
            insert $Cm_i$ into Cm'
        end
    end
return Cm'
```

- **Cm' : Member list to store members that will split out**
- **Cm : Member list to split**
- **$\{Cm_1, \ldots, Cm_n\}$: n cluster members in Cm**

Table 4.2: Selection of cluster members to create a new cluster

The super-member that wishes to split the cluster then asks the new super-member to form a new cluster that contains the selected members, passing the selected member list and neighboring super-member list to the new super-member. The new super-member creates a new cluster by establishing sessions. The super-member then terminates sessions with the selected members. Figure 4.3 shows signaling architectures before and after splitting.

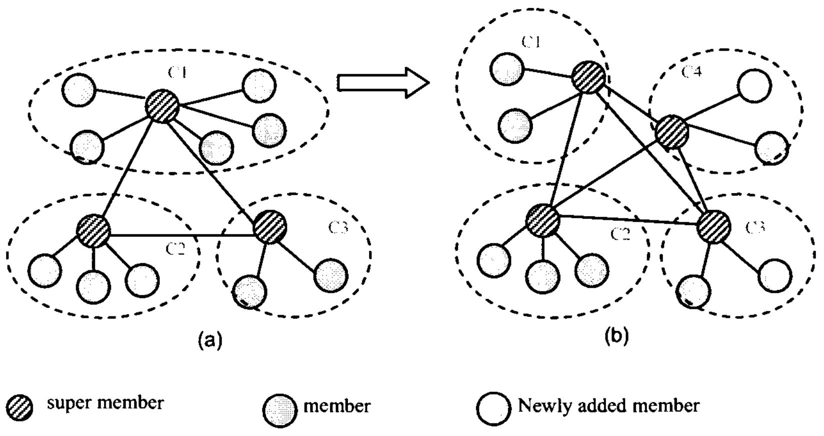


Figure 4.3: Cluster splitting (a) before splitting (b) after splitting

The setting of *Sv* is critical for the signaling system. For example, if the *Sv* too small, the system may exhibit poor performance due to frequent splitting. If the *Sv* is too large, the signaling will have a centralized structure. On one hand, we expect a minimum number of clusters so that the overall signaling overhead is low. On the other hand, a centralized architecture is unacceptable for a large-scale conference. We will further discuss the setting of *Sv* in Chapter 7.

4.2.5 Merging

If the size of the cluster diminishes to *Mv*, the super-member initiates a merger procedure. This procedure starts by a searching for an existing cluster with which to merge, with the constraint that the size after the merger will be less than *Sv*. The searching algorithm is shown in Table 4.3. We denote the cluster that wishes to start a merger as *C* and assume that the cluster *C* has *n* neighboring clusters $\{C_1, \ldots, C_n\}$. The selected cluster C_{select} is initially set to null. The merger begins only when the returned C_{select} is not a null value. In the table, *Smin* is used to store the size of the cluster after merging.

```
1.  Smin ← Sv
2.  if(n<=0)
3.    return C_select
4.  else loop i (from 1 to n)
5.        if(sizeof(C_i)+sizeof(C) ) < Smin)
6.          Smin ← (sizeof(C_i)+sizeof(C)
7.          C_select ← C_i
8.        end
9.      end
10.  return C_select
```

- **Smin : size of the cluster after merging**
- **C : the cluster that wishes to merge**
- **C_{select} : the cluster that C will merge with**
- **{C1,..., Cn}: a list of neighboring cluster of C**

Table 4.3: Searching for a cluster to merge with

A new super-member is elected as soon as the merger begins. The new super-member will be one of the two super-members (the one with more capability) of the two clusters. The procedure continues as follows: the elected super-member establishes sessions with the members of the cluster to merge with. The un-elected super-member then terminates sessions with its members and sets the elected super-member as its super-member, and it becomes a regular member. The merger information will then be propagated to the neighboring super-members.

4.2.6 Information propagation

In order to maintain a signaling cluster system, efficient information propagation is required, i.e. rapid propagation with as little introduced overhead as possible. In our architecture, super-members are responsible for propagating membership and capability information whenever there is a change. The information can be propagated to all the signaling agents in no more than two hops. Figure 4.4 shows an example. An option that can further reduce the propagation overhead is that only part of the information is propagated and only to super-members. This is realistic because it is not necessary for every member to have knowledge of every system change, e.g. a splitting of a cluster.

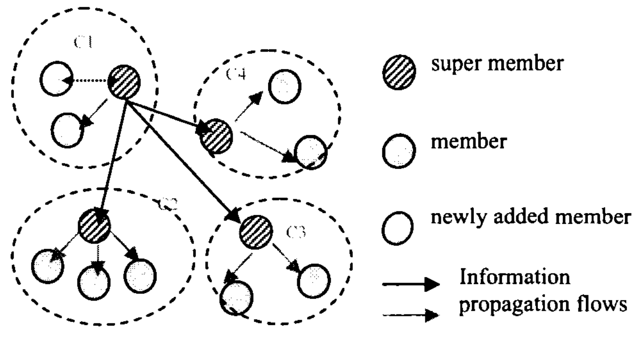


Figure 4.4: Information propagation as a new party joins

4.2.7 The issue of coincident behavior of participants

One issue of a distributed signaling architecture is the state synchronization when there are coincident behaviors of participants in the conference. Such behaviors may cause inconsistent states among participants, e.g. with a coincident joining (defined in [82]), two newly joined parties have no way to know each other and no session will be established between them, and thus the fully distributed signaling architecture cannot be maintained. However, with the information propagation procedure, the cluster scheme defined in this article can handle most coincident behaviors. In some cases, protection mechanisms should be used to prevent inconsistencies. We discuss this issue case by case.

Case I: Coincident join -- two or more parties join a conference at the same time. They may join the same cluster or different clusters. Our scheme can handle this case because the coincidently joined parties do not have a direct session with each other. Instead, they establish sessions with the super-members that are already in the cluster and later they can 'know' each other from their super-member(s).

Case II: Coincident departure – two or more participants leave a conference at the same time. They may leave the same cluster or different clusters. Similar to the first case, our scheme can handle the coincident departure of members and less than two super-members. Our scheme cannot handle the coincident departure of super-members because the newly selected super-members cannot already have knowledge of each other. In order to avoid such a case, we define a protection phase when a super-member leaves. A super-member should reject any session establishment or termination request when it starts to leave. The protection phase ends when the super-member has completed the leaving procedure. Within this protection phase, a super-member leaving procedure will fail if another super-member is leaving at the same time, because the newly selected super-member cannot establish a session with a leaving neighboring super-member. If a super-member fails to leave, it will retry after a random period of time.

Case III: Coincident splitting – two or more clusters split at the same time. With the mesh structure of super-members, the super-members in older clusters maintain a session with every newly split super-member. After a run of the information propagation procedure, a newly split super-member will have knowledge of the other new super-members. The logic to be added in order to handle this case is that if a super-member finds that it has not established a session with a neighboring super-member and if it has a higher address, it will establish a session with that super-member. Figure 4.5 shows an example of the coincident splitting. In the figure, *c1* and *c2* split at the same time. The super-members of *c3* and *c4* 'know' each other after a run of information propagation. The super-member of *c3* then establishes a session with the super-member of *c4*.

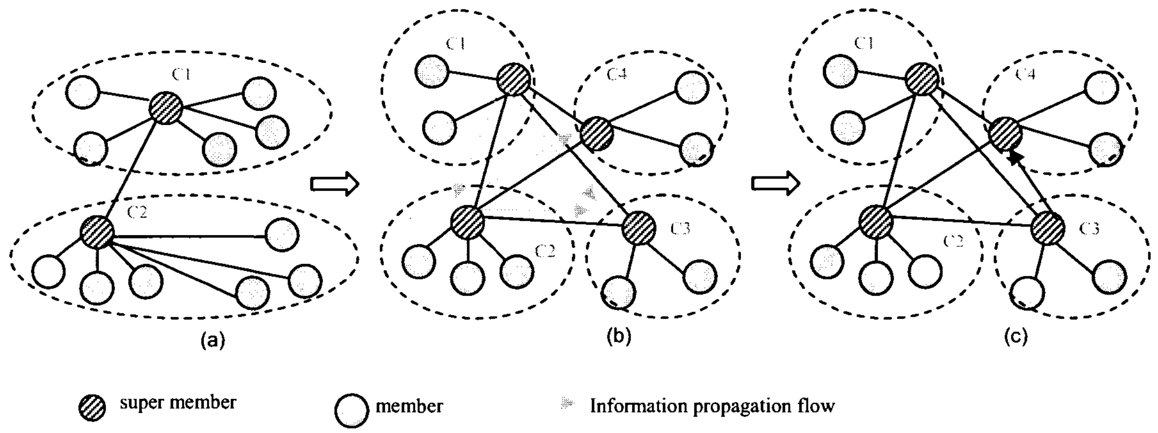


Figure 4.5: Coincident Splitting (a) before splitting (b) after coincident splitting and propagation of cluster information (c) the missing session established

Case IV: Coincident merging – two or more pairs of clusters merge at the same time. Our scheme can handle this case because there is no new super-member elected. The cluster state can be propagated to all neighboring super-members.

There are two other critical issues related to the signaling procedures. The first is the participant capability discovery that is critical to super-member election, and the second is the detection of unintentional departure and session recovery. We will detail the solutions for these issues in the next sections.

4.3 Capability exchange mechanism

The election of a super-member is based on the node capabilities. In this section, we discuss how the capabilities of participants are exchanged and how the capability information is maintained and refreshed. We start by looking into the existing service discovery approaches. Then we propose a simple mechanism that is lightweight and especially suitable for our cluster-based architecture.

4.3.1 Related approaches for capability exchange and discovery

As we have assumed that each participant knows its own capability, there are several ways for a participant to discover the capabilities of other participants. The simplest approach is broadcast, i.e. every participant periodically broadcasts its capability to the network and a participant that receives the information stores it and forwards it. The drawback here is flooding. Therefore, too much network overhead will be introduced.

Another approach is to embed capability information in the routing layer and use the routing protocol for capability publication and discovery (e.g. LSD [88]). This approach saves network resources, but it does not meet our requirement of independence of lower-layer routing protocols. Still another approach is to deploy general publication/discovery protocols (such as Konark [89]) that can provide capability exchange services for signaling protocols. However, this approach is not worthwhile because these protocols are generally loaded with functions that are not required and they may significantly slow down the signaling procedures.

4.3.2 Proposed mechanism for capability publication and discovery

We propose a simple application-level capability publication/discovery mechanism that specifically fits the cluster-based signaling scheme. The entity involved is the SA. We define three types of messages: Cap_subscribe, Cap_notify and Cap_publish. Cap_subscribe is a request message containing a subscription interval. Cap_notify is a response message containing a sequence number and the current capability level, and Cap_publish is a message containing the current capability level.

Cap_subscribe is used in the following three cases:

- When the initiator of a conference establishes the first session, it sends the message to the called party and sets the subscription interval to zero. In this case, the called party sends only one Cap_notify back to the initiator with its current capability loaded in the message.
- A super-member sends a Cap_subscribe message to a member when a session is established. In this case the subscription interval is set to a non-zero value and in each interval period, the member sends back a Cap_notify message loaded with its current capability.
- A super-member sends a Cap_subscribe to a member with a zero subscription interval value. The member sends back a Cap_notify response and stops the periodic Cap_notify messages.

Cap_publish is sent between super-members. When its capability is changed, a super-member sends a Cap_publish to every neighboring super-member.

4.4 Unintentional departure detection and session recovery

In Section 4.2, we have presented the normal cases of user departure, i.e. the voluntary departure. In the normal case, the participant informs the system before leaving. The system can then react and handle the session. In the case of unintentional departure of participants, the signaling system is not informed. The departure may have several causes and it may happen often in MANETs. A timely detection of such departures is necessary for the signaling system. It helps the signaling system in handling the session gracefully, i.e. termination of a failed session, reconstruction of a cluster, update of the cluster state, and update of the participant status.

In this section, we focus on the issue of the unintentional departure of participants. In order to tackle the issue, a failure-detection and a recovery mechanism have to be added to our signaling architecture. This mechanism should meet the following requirements. First, it should be able to work in the cluster environment of our signaling architecture. Second, it should be able to detect every session failure caused by unintentional departure of either a super-member or a member. Third, the failure detection should be fast. Fourth, the mechanism should be lightweight and should not introduce too much overhead in the network. In the next sub-sections, we first introduce the concept of heartbeat and review the approaches in the literature with respect to the requirements. We then propose a session recovery mechanism that is based on a request/reply heartbeat protocol.

4.4.1 Heartbeat and related failure detection approaches

Heartbeat protocols are often used to detect failures in distributed systems (e.g. [92][93][94][95]). Recently, they have also been used in MANETs for node failure detection ([91][96]). A heartbeat mechanism relies on a heartbeat rate and this rate may change according to different conditions. Heartbeat protocols are numerous, and they usually follow three communication modes: 'push' mode (e.g. [91]), request/reply mode (e.g. [92]), and counter-based mode (e.g. [96]). In the push mode, a failure monitor and a monitored node are defined. The monitored node periodically sends a heartbeat message to the monitor. The monitor treats a node as failed when it has not received a pre-defined number of heartbeat messages.

In request/reply mode, a failure monitor is also defined, but it is different from 'push' mode. The monitor sends heartbeat queries to the monitored node and gets responses.

The monitor treats a node as failed when it sends a number of heartbeat requests but does not get any response. This mode is sometimes called 'pinging' (e.g. [95]).

In counter-based mode, every node has a heartbeat counter. It increases the counter value in each period of time and propagates the counter to a sub-set of nodes in the network. A node that receives the counter will update a list of counters. The failure detection scheme is as follows: a node *p* is treated as failed by a node *q* if the *p's* counter in *q*'s list has not been updated for a period of time.

4.4.1.1 Failure detection and recovery in signaling

The unintentional departure detection and session recovery are specific to MANETs. We have not found related work for infrastructure-based signaling systems such as H.323 [73] and SIP [74]. These signaling systems handle failures during the session establishment phase, but they do not have any explicit session failure detection mechanism after the session has been established.

ICEBERG signaling [86] is the only signaling system that has defined an explicit failure detection mechanism. The mechanism is based on a heartbeat protocol, following a two-way 'push' mode. It is illustrated in Figure 4.6.

The iPOP (iceberg Point Of Presence) is a call control point in ICEBERG. It dynamically creates call agent processes. Each call agent handles a session for an end point. The failure of a call agent is detected by the iPOP upon the reception of periodical heartbeat messages from the Iceberg Access Point (IAP). The message contains an updated session status.

The heartbeat message sent from iPOP to IAP has as purpose to detect if the iPOP is still alive. If the heartbeat has not been received after a timeout, the IAP will communicate with another iPOP. There are also heartbeats between two iPOPs in order to detect if each one is still alive.

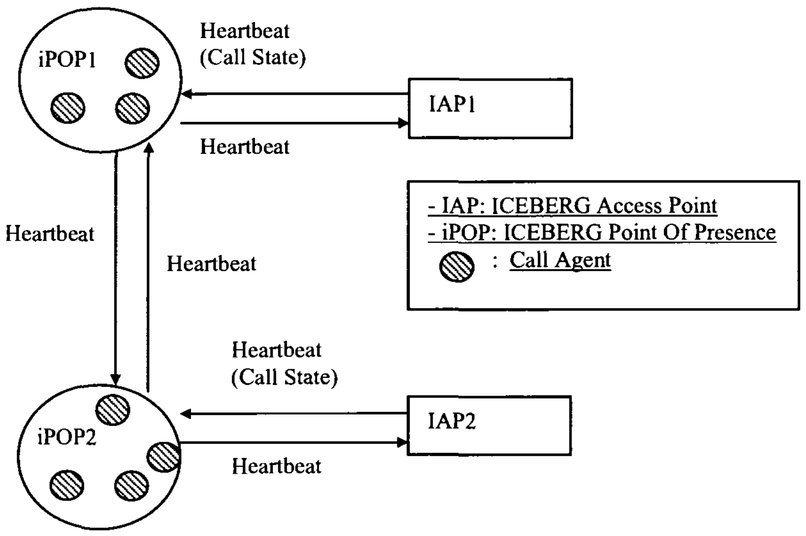


Figure 4.6: Iceberg signaling: failure detection mechanism

This solution cannot be used in our signaling architecture because it assumes that an IAP never fails and the heartbeat message from the IAP can always be received by an iPOP. This is not case for MANETs. Moreover, the heartbeat message from IAP to iPOP is heavily loaded because it carries a full session state. The advantage of this mechanism is that it is able to quickly detect and recover a failed session process.

4.4.1.2 Failure detection and recovery for cluster-based architectures in MANETs

Cluster-based routing protocols, such as in [48] and [94], have introduced cluster update and re-affiliation mechanisms that handle link failures between a cluster-head and a cluster member. All the cluster members continuously monitor their signal strength as received from the cluster-head. When the signal strength decreases, a cluster member notifies its current cluster-head that it is no longer able to attach itself to it. The cluster-head tries to hand-over the member to a neighboring cluster. If a cluster-head crashes or moves out of range, a cluster re-formation procedure is performed.

This mechanism is based on the detection of physical wireless link status and it is not applicable at the application layer. It is therefore not applicable in our clustering environment because our clusters are application level clusters.

Authors in [90] have proposed a SIP-based multicast framework for MANETs. They defined a ping-timeout mechanism in order to detect the abrupt disconnections between a cluster head and a cluster member. The mechanism is comparable to a two-way request/reply heartbeat protocol. If a node fails to send or respond to a SIP NOTIFY-like message in a predetermined period, it is assumed to have left the network ungracefully and then the node will be removed from the host list. The removed node may later re-connect to the network by re-broadcasting a request to join the network.

This mechanism is applicable to the cluster-based signaling architecture. It meets the first three requirements that we derived. However, it introduces a significant network overhead, i.e. for each period of time four messages (two requests and two responses) are exchanged between a cluster-head and a cluster-member.

A cluster-based failure detection mechanism has been proposed in [91]. It defines a 'push'-style heartbeat failure detection service, which includes three phases: heartbeat exchange, digest exchange and health-status-update broadcast. A cluster member is determined to be failed if its cluster head has not received its heartbeat or digest message and none of the other members in the cluster is aware of the heartbeat of the member. A Deputy Cluster-Head (DCH) is defined to detect the failure of a cluster-head. A cluster-head is determined to have failed only if its DCH did not receive its heartbeat or digest or the health-status-update.

This mechanism is applicable for the cluster-based signaling system. It is capable of detecting every node failure, but it cannot handle node mobility and link failures. Furthermore, its overhead is non-trivial because it broadcasts heartbeat and digest messages.

We find out that none of the aforementioned solutions meet all our requirements. Therefore, we propose our solution.

4.4.2 Proposed failure detection and session recovery mechanism

We define a *session* as a signaling link between two participants. A conference may consist of two or more sessions. Our basic idea is to use a one-way request/reply heartbeat protocol. It reduces the overhead of the ping scheme defined in [90].

We believe the one-way request and reply is enough to detect a session failure. Authors in [91] have shown that it is impossible to ensure accuracy (i.e. no false detection) for a failure detection mechanism in MANETs. Actually, it is not necessary in our case. When a session has failed, either participant that maintains the session may have unintentionally departed. What we require and believe important is to recover the session as soon as possible, but not to accurately determine which party really has left the conference. Thus, our principle is that a participant is responsible for detecting if a maintained session is still alive. A participant also has the responsibility to recover a maintained session when the later has failed.

We define a *heartbeat* as a periodic exchange of a request and a reply. Each session in a conference maintains a heartbeat. A heartbeat starts when a session is established and it stops when the session is terminated. We define two entities for a heartbeat: a *sender* that sends the heartbeat requests and receives heartbeat replies and a *responder* that receives requests and sends replies. When a new session is created and if the session is between a super-member and a member, the super-member becomes a heartbeat sender and the member, becomes a responder. If it is a session between two super-members, the one with more capability becomes the sender and the other super-member becomes the responder. A heartbeat request message contains a sequence number and a super-member list that includes the identification and resource level of every super-member in the conference. A heartbeat reply messages only contains the sequence number received from the last request. We define three timers: heartbeat rate *Th*, transaction timer *Tt* and recovery timer *Tr*.

The heartbeat mechanism can be described as follows:

- In each period of *Th,* a heartbeat sender sends a request to its responder, and starts a timer *Tt*. If *Tt* fires and no reply received from the responder, the sender re-sends the request and restarts *Tt*. If there is no reply upon a number of requests, the sender will determine that the responder has unintentionally departed, so it will terminate the session and delete the session states. The super-member will then propagate the departure information to the neighboring super-members following the information propagation scheme. The cluster status is updated accordingly.

- When a session is created, a heartbeat responder starts a timer with the value *Th+n*Tt*. If it receives a heartbeat request before the timer fires, it retrieves the super-member list from the request, sends a reply and resets the timer. If the timer fires and no heartbeat request is received from a sender, the responder will terminate the session with the sender. The member then starts a timer *Tr* and tries to establish a session with one of super-members in the super-member list in each period of *Tt,* until a session is successfully established. In the case that the super-member list contains only one super-member (i.e. there is only one cluster in the conference), the member simply waits for the super-member to establish a session with it. If the timer *Tr* fires and no session has been established, the member clears all the conference states and leaves.
- If a super-member finds that it has just terminated the last session it maintains due to the unintentional departures of its members and neighboring super-members, it keeps the conference state and starts the timer *Tr*. In each period of *Tt,* it tries to establish a session with one of the super-members in the neighboring super-member list until one session has successfully established. The super-member then becomes a cluster member and a heartbeat responder. If there is no neighboring super-member, or the timer *Tr* fires and no session has been created, the super-member deletes all the states and leaves the conference.

We further illustrate the mechanism through scenarios.

4.4.3 Scenarios of failure detection and recovery

Figure 4.7 shows a scenario in which Member D has unintentionally departed. When D has recovered later, it joins super-member A's cluster. Figure 4.8 shows a scenario where super-member B has unintentionally departed. Member D then re-joins super-member A's cluster. After a while, when B has recovered, it joins super-member A's cluster and becomes a cluster member.

It should be noted that the mechanism does not always ensure session recovery from transient departures. For example, if all the super-members in a conference unintentionally depart at the same time and none of them is able to re-connect to the network in a short period of time, the conference sessions cannot be recovered. We believe this is fair because super-members play a key role in cluster-based signaling system. In this case, the sessions are at least terminated gracefully.

The three timers that we defined should be set very carefully. The value of *Tt* should be larger than the round-trip transmission delay between two participants. The value of *Th* should be determined by a balance of heartbeat overhead and departure detection time. The value of *Tr* should be less than the duration of a conference.

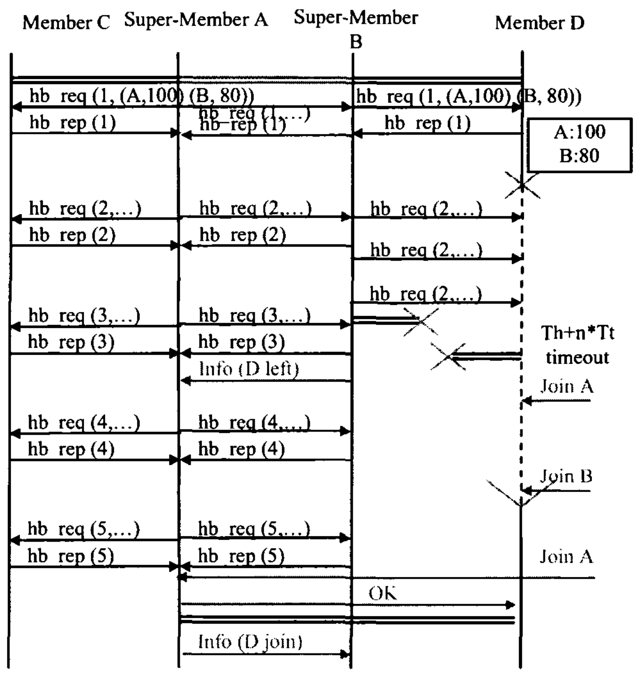


Figure 4.7: Member unintentional departure

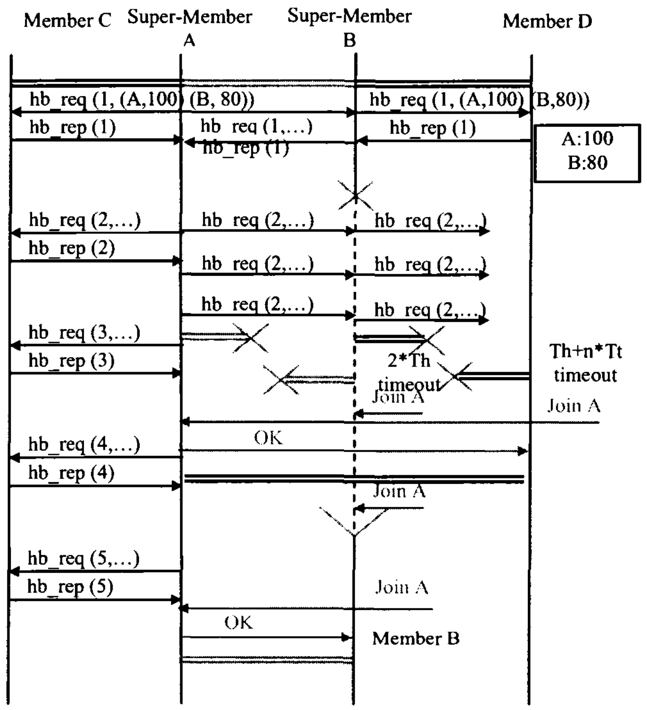


Figure 4.8: Super-member unintentional departure

4.5 Conclusions

In this chapter, we have proposed a cluster-based signaling architecture and shown how the clusters operate during each phase of the signaling procedure. If we analyze the architecture with respect to the signaling requirements, we can find that the architecture is a very promising solution.

First, there is no permanent centralized entity. The signaling agents act as super-members on a transient basis. Second, the sessions are maintained in a distributed and dynamic manner. Session-related information is dynamically propagated to all of the parties. Furthermore, the architecture can handle both voluntary and unintentional departures of participants. A session recovery mechanism is also provided. Third, the signaling system can scale automatically in the same manner as the network in which it is implemented. The clusters can automatically split and merge, fitting the scale of the conference.

Fourth, the architecture can be lightweight if the appropriate technology such as SIP is selected for its implementation. Fifth, resource usage is optimal because the super-member election is based on node capabilities. Finally, our signaling clusters are independent of the lower layer clusters and the signaling system does not rely on any specific lower-layer protocols.

In Chapter 6 and Chapter 7, we will further discuss the implementation and performance measurement of the cluster-based signaling architecture.

Chapter 5 : A Signaling Architecture for Conferencing in Integrated MANETs/3G Networks

Based on the signaling architecture described in Chapter 4, we propose the signaling architecture for integrated MANETs/3G networks. We first present the network structure that we consider. We then introduce the integrated signaling architecture, the entities and the general principles. After that the scenarios of conference establishment, participant-joining and conference termination are illustrated, through which we demonstrate how the architecture work under different cases and how the entities behave. We will conclude by a summary of the architecture and analyze its potentials in meeting the requirements.

5.1 Network assumptions

The integrated MANETs/3G network we consider is shown in Figure 5.1. In this figure, we define a *multihop routing area* as an area in which all the nodes are working on a MANET interface and the nodes can reach each other by direct wireless connections or multihop routing. We also assume that there is more than one multihop routing area in the system. As introduced in Chapter 2, there exist different physical connection techniques. For our work, we do not specify any lower-layer technology, i.e. the signaling architecture will be independent of the lower layers. In fact, from an application-layer point of view, no matter what technique the network uses, the integrated signaling system should provide an identical service to all users. Therefore, the network we assume supports devices with only MANET interfaces, devices with both MANET and 3G interfaces and devices with only 3G interfaces. We define the first two types of devices as the *multihop Mobile Station (MS)* and the third as the *single-hop MS*.

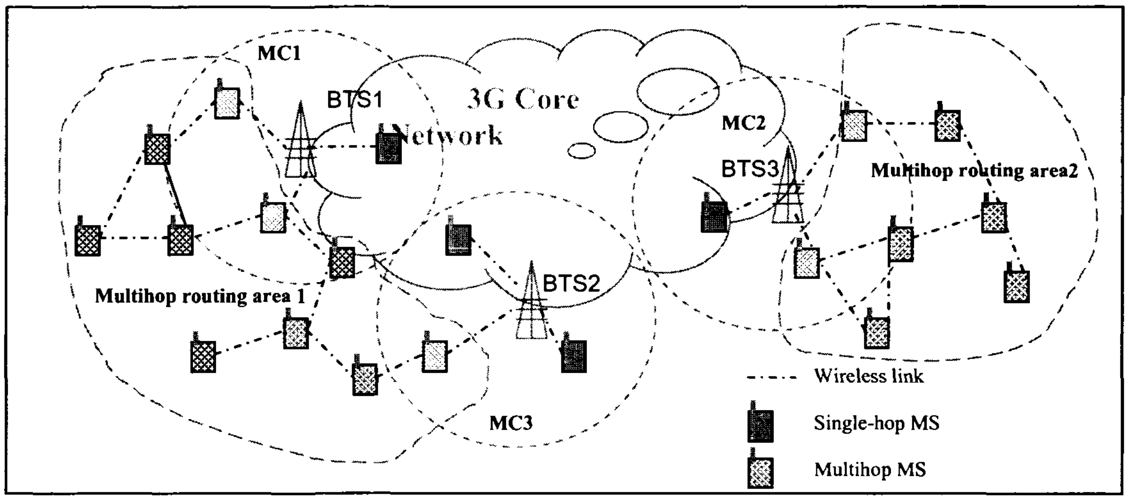


Figure 5.1: Considered Integrated Network

5.2 Architectural principles

We use the solution for signaling in standalone MANETs as basis for the solution in the integrated MANETs/3G system because of two reasons. First, it enables conferences with MANET participants. Second, the architecture is centralized with respect to each cluster, which is applicable for conferencing in 3G networks. We also assume that the cluster-based signaling architecture is the default signaling architecture used by the MANETs' side of the integrated network.

5.2.1 The proposed architecture

The proposed architecture is depicted in Figure 5.2. We define three entities: the Signaling Agent (SA), the Conference Focus (MRFC/AS), and the Conference Gateway (CGW). The MRFC/AS is the entity defined in the 3GPP standard [81]. In our architecture, its functionality as per the 3GPP standard is enhanced. The enhancement is a CGW discovery functionality, i.e. the ability to find a suitable CGW that can handle sessions with the participants in MANETs. The SAs are conference participants. They are either 3GSAs, i.e. participants in 3G, or MSAs, i.e. participants in MANETs. 3GSAs are the signaling parts of the 3G User Equipment defined in [81]. They can establish sessions with the MRFC/AS. MSAs are the same as the SAs defined in chapter 4. An MSA can either be a super member or a simple member. Here also the CGW discovery functionality is needed in addition to the functionality of the MSA defined in chapter 4. It should be noted that 3GSAs and MSAs may run different signaling protocols.

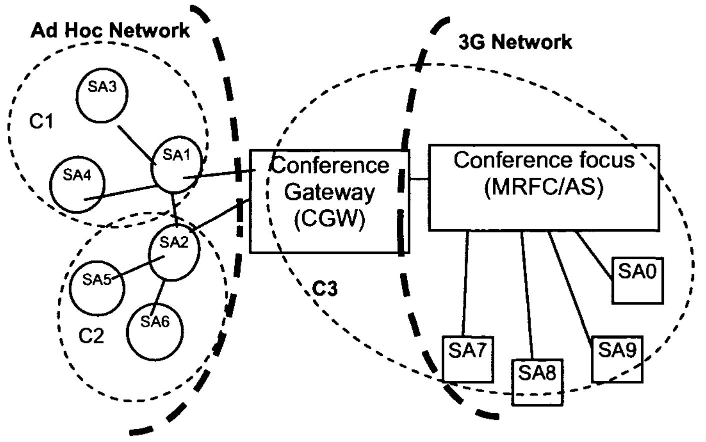


Figure 5.2: Integrated conferencing architecture

The CGW is a new entity that we introduce. It is a mediator deployed by the 3G network operator (or a trusted third party). It has an infrastructure-based interface that is connected to the 3G conference focus. It also has a MANET interface that is connected to MANET SAs. Unlike the client proxy defined in UCAN or the ARS defined in iCAR, the CGW is an application layer entity. Its two interfaces are not physical air interfaces, but application layer interfaces to signaling components. A CGW has six major functions. First, it has the functionality of a signaling agent that is capable of establishing sessions with MSAs and with the conference focus. Second, it understands the signaling protocols for multimedia conferencing in both MANET and 3G, and performs the translation (if required). Third, it understands the conferencing signaling architectures (e.g. centralized versus distributed) used in both MANET and 3G, and does the mapping (if required). Fourth, it collects the membership information in both networks, converts it (if required), and distributes it. Fifth, it provides the functionality of publication and discovery so that MSAs and the conference focus can find and use its services. Sixth, it provides registration functions and manages the repository of MANET participants.

Our architecture relies on three main principles. First, participants in MANET see the CGW as a special super member that never leaves, splits or merges with other super members. Second, participants in 3G (i.e.3GSAs) see the MRFC/AS as a centralized control point to which every participant, in 3G networks or in MANETs, is connected. Third, the MRFC/AS sees the CGW as a sub-focus that aggregates and manages sessions for MANET participants. We assume that participants that are not 3G users are implicitly seen by the 3G conference focus as MANET participants, to which sessions are created through a CGW. The same assumption is made for MSAs.

5.2.2 CGW discovery mechanism

The CGW can be discovered by the 3G conference focus and by the MANET participants. Since the CGW is deployed by either the 3G network operator or by a trusted party, the CGW and the conference focus can be pre-configured to have knowledge of each other. However, a MANET participant has to find and register with an appropriate CGW when it wishes to establish a session with a 3G participant or when it wishes to be invited by 3G participants. A CGW discovery and publication mechanism is therefore needed. It should support two basic scenarios. First, a CGW should be able to periodically publish its location and capability to MANET nodes – necessitated by the frequent membership changes in MANET. The MANET node can cache the CGW information when it first receives it, and registers with the CGW. Second, a MANET node should be able to send a CGW request message that contains CGW capability requirements to the network. A MANET node that has the proper CGW capability information or a CGW that matches the capability requirements can respond. The MANET node that receives the responses then registers with the CGW.

CGW location information contains the CGW's address and listening port. CGW capability information includes parameters such as the *signaling protocols supported*, *conference type supported*, *network information*, *architectures supported* and *encoding supported*. Service discovery and publication in MANET has been extensively investigated. A simple scheme to implement the two scenarios is broadcasting, i.e. a node that has a service periodically broadcasts its service advertisement to all the nodes, and a node that needs a service broadcasts a discovery message to all the nodes and then waits for responses. A major issue in this scheme is the extra load induced on the network by the flooding of messages. To solve this problem, new proposals such as Konark [89] and Allia [97] have emerged. Any of them can be used in our architecture.

In an integrated 3G/MANET network, CGWs can be collocated with the conference focus, or deployed as standalone nodes. The number of CGWs to be deployed is to be determined by network planning and design. We will further discuss the issue of CGW deployment in Chapter 7 and Chapter 8.

5.3 Conferencing scenarios in integrated MANETs/3G networks

Under the considered network environment, the integrated signaling architecture can be applied as a use case shown in Figure 5.3. In this figure, two CGWs are deployed and each of them handles the sessions from a multihop routing area.

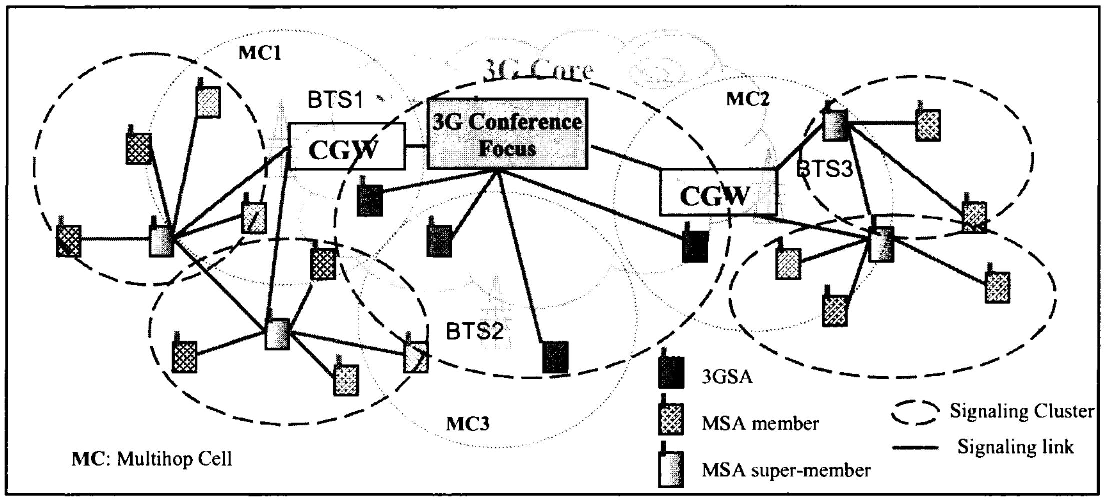


Figure 5.3: A use case for applying the signaling architecture in considered networks

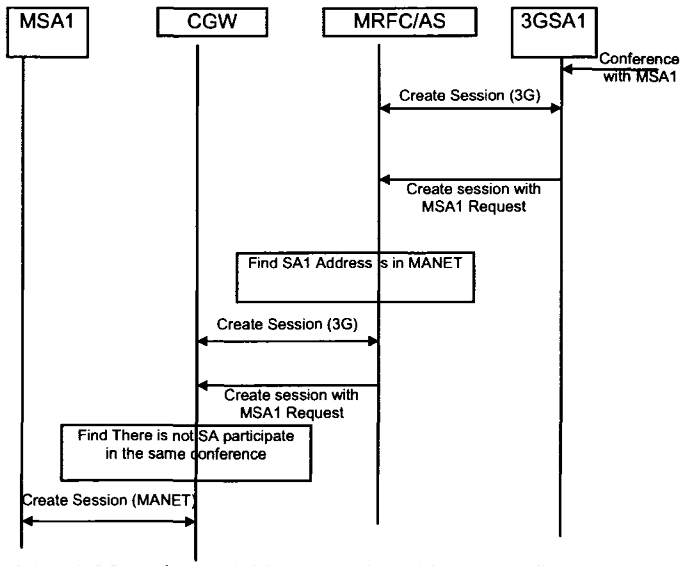
The use case enables four different scenarios: conferencing with 3GSA parties, conferencing with both 3GSA and MSA parties, conferencing with MSA parties in the same multihop routing area and conferencing with MSA parties in different routing areas. The first and the third scenario do not require the use of a CGW. The second scenario requires CGWs to perform protocol translations and signaling routing. The last scenario does not require a protocol translation, but it uses CGWs and the 3G conference focus as signaling routing mediators.

We are only concerned with the new scenarios that require a participation of CGW(s) because the other scenarios are the same as the conference scenarios in standalone MANETs or in pure 3G networks.

5.3.1 Conference initiation

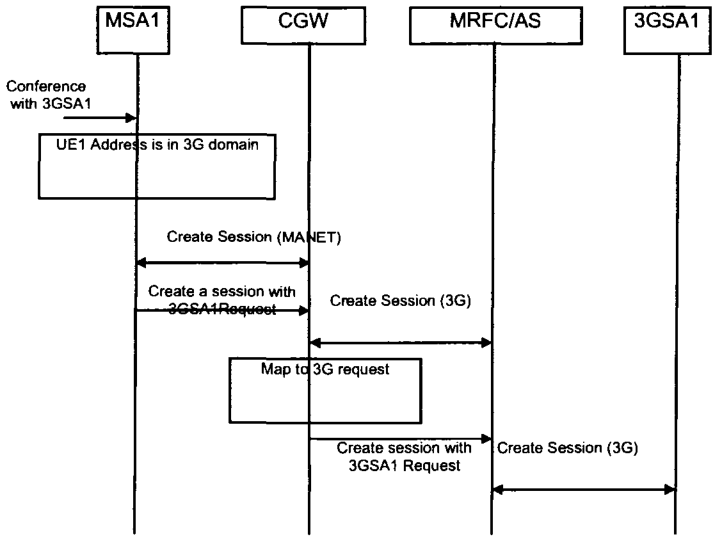
Figure 5.4 depicts conference initiation sequences between an MSA and a 3GSA. In Figure 5.4-a, 3GSA1 initiates a conference. It first creates a session with its MRFC/AS using the 3G signaling protocol. Then it requests the conference focus to create a session with MSA1. The conference focus finds that MSA1 is in a MANET. It then creates a session with the CGW and asks the CGW to create a session with MSA1. Finally, the CGW creates a session with MSA1 using the signaling protocol of MANET and designates MSA1 as a super member. Figure 5.4-b illustrates the case where MANET user MSA1 initiates a session with 3G user 3GSA1. In both cases, the CGW translates protocols and maps the two architectures.

Figure 5.5 shows an example of conference initiation between two MSAs (MSA1 and MSA2) that are located in different multihop routing areas and are registered with different CGWs. The session set up request is first forwarded from the CGW1. Since the CGW1 does not know the location information of MSA2, it forwards the request to the MRFC. The MRFC finds that the MSA2 is registered with CGW2, so it forwards the request to CGW2. The request finally reaches MSA2 and a response message is sent back following the request route. Since the two parties are located in different routing areas, both parties become super-members when the first session is set up. The conference state is stored in both parties and it is stored in both CGWs as well.



MSA1
CGW
MRFC/AS
3GSA1
Conference with MSA1
Create Session (3G)
Create session with MSA1 Request
Find SA1 Address is in MANET
Create Session (3G)
Create session with MSA1 Request
Find There is not SA participate in the same conference
Create Session (MANET)

5.4-a: A 3G participant initiates a session with a MANET party



5.4-b: A MANET participant initiates a session with a 3G party

Figure 5.4: Conference initiations with participants in different networks

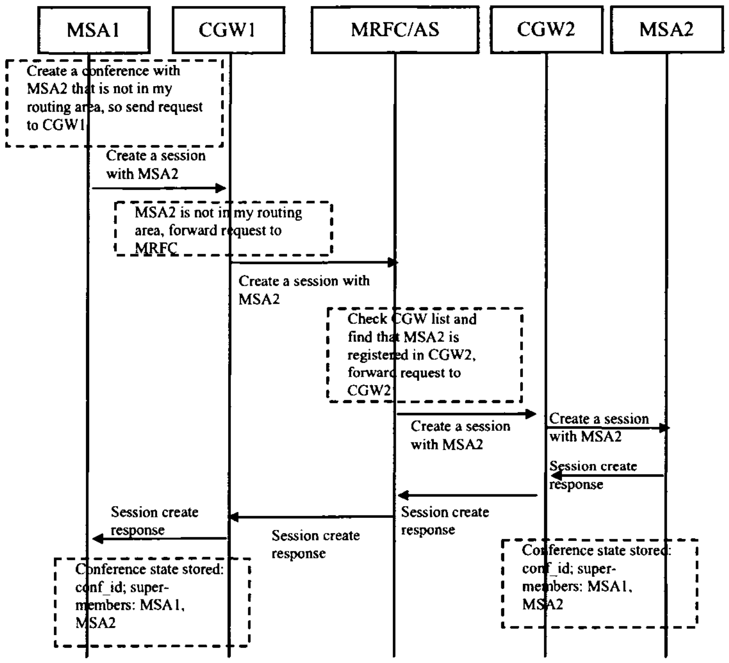
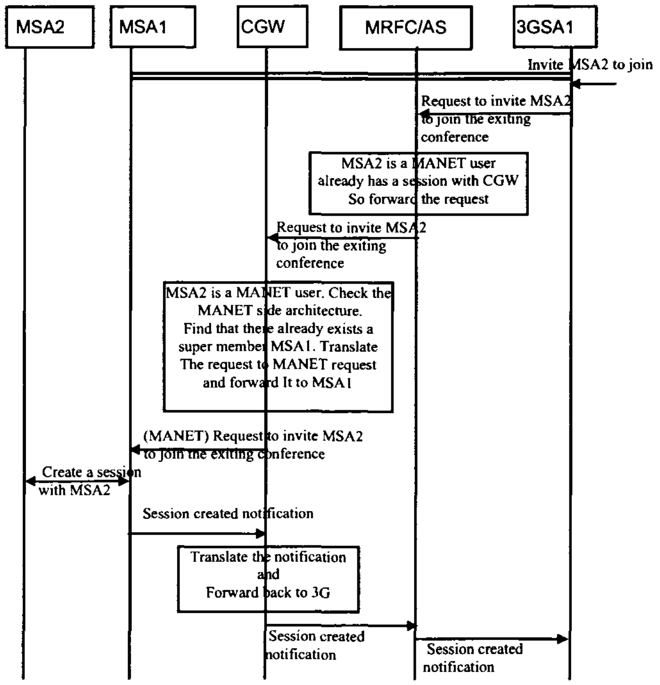

MSA1
CGW1
MRFC/AS
CGW2
MSA2
Create a conference with MSA2 that is not in my routing area, so send request to CGW1
Create a session with MSA2
MSA2 is not in my routing area, forward request to MRFC
Create a session with MSA2
Check CGW list and find that MSA2 is registered in CGW2, forward request to CGW2
Create a session with MSA2
Create a session with MSA2
Session create response
Session create response
Session create response
Session create response
Conference state stored: conf_id; super-members: MSA1, MSA2
Conference state stored: conf_id; super-members: MSA1, MSA2

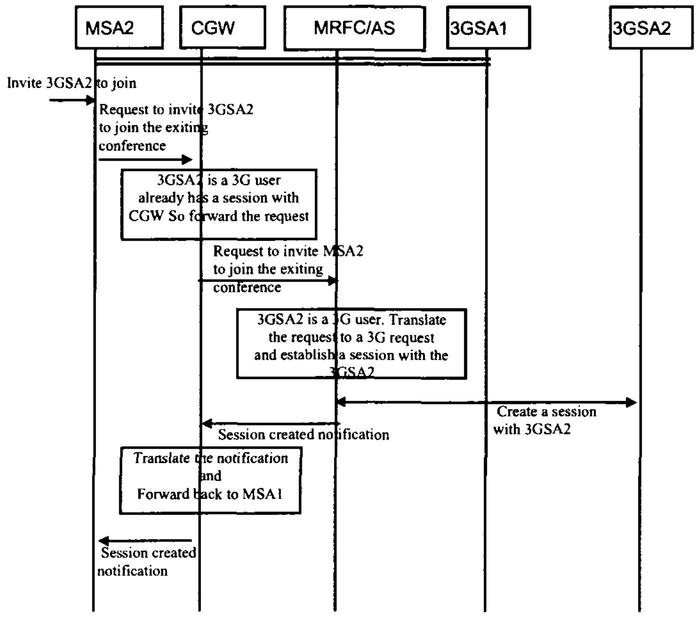
Figure 5.5: Conference initiation with MSAs in different routing areas

5.3.2 Participant joining

3G and MANET users are allowed to join an existing conference when they are invited. A 3GSA that wants to invite a new participant needs to request the conference focus to do so. The conference focus then determines which network the called party is in. If it is in MANET, the conference focus requests the CGW to create a session with the called party. The CGW then decides to which cluster the called party will join. Finally, it asks the super member of the cluster to create a session with the called party. An example of the scenario is illustrated in Figure 5.6-a. If a MSA wants to invite a 3GSA to join the conference and the MSA is a super member, it simply sends a request to the CGW, which will then request the conference focus to invite the 3GSA. If the MSA is not a super member, it has to request its super member to do this. Figure 5.6-b shows an example of this scenario. In the figure, the MSA that invites the 3GSA is a super-member.


MSA2
MSA1
CGW
MRFC/AS
3GSA1
Invite MSA2 to join
Request to invite MSA2 to join the exiting conference
MSA2 is a MANET user already has a session with CGW So forward the request
Request to invite MSA2 to join the exiting conference
MSA2 is a MANET user. Check the MANET side architecture. Find that there already exists a super member MSA1. Translate The request to MANET request and forward It to MSA1
(MANET) Request to invite MSA2 to join the exiting conference
Create a session with MSA2
Session created notification
Translate the notification and Forward back to 3G
Session created notification
Session created notification

5.6-a: A 3G participant invites a MANET party


MSA2
CGW
MRFC/AS
3GSA1
3GSA2
Invite 3GSA2 to join
Request to invite 3GSA2 to join the exiting conference
3GSA2 is a 3G user already has a session with CGW So forward the request
Request to invite MSA2 to join the exiting conference
3GSA2 is a 3G user. Translate the request to a 3G request and establish a session with the 3GSA2
Create a session with 3GSA2
Session created notification
Translate the notification and Forward back to MSA1
Session created notification

5.6-b: A MSA invites a 3GSA

Figure 5.6: Inviting a participant in a different network

Figure 5.7 shows the case that a MSA invites another MSA located in a different multihop routing area. In the case, the MSA3 is located in the same routing area as MSA2 and MSA1 wishes to invite it. The MSA1 that has not found the MSA3 in its routing area does not directly create a session with MSA3. Since it does not know where MSA3 is located, it simply sends a 'request to invite' to its local CGW (CGW1) with the conference ID included. Then, the message is sent to the MRFC and the MRFC finds that MSA3 is managed by the CGW2. It forwards the request to CGW2. The CGW2 finds that the conference exists and the MSA2 is the super-member in the same routing area. It further forwards the request to MSA2. Finally, the MSA2 sets up a local session with the MSA3 and the MSA3's joining information is sent back to MSA1.

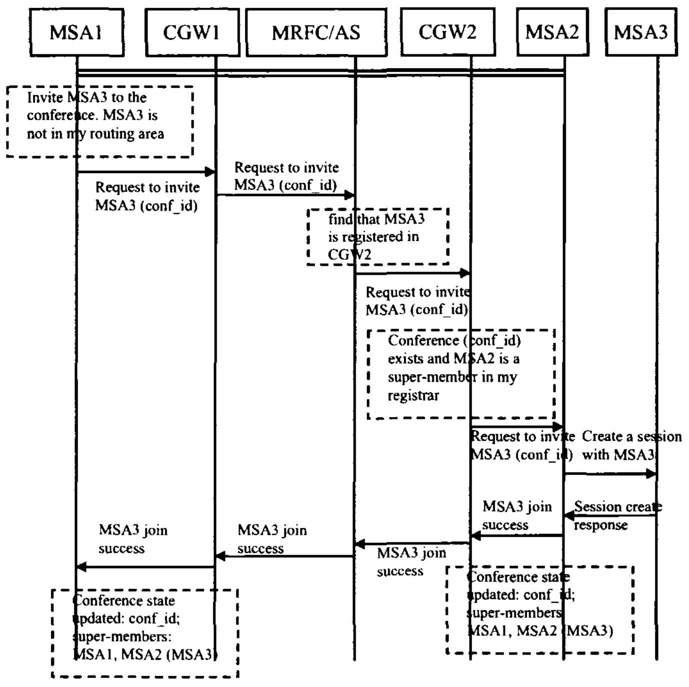

MSA1
CGW1
MRFC/AS
CGW2
MSA2
MSA3
Invite MSA3 to the conference. MSA3 is not in my routing area
Request to invite MSA3 (conf_id)
Request to invite MSA3 (conf_id)
find that MSA3 is registered in CGW2
Request to invite MSA3 (conf_id)
Conference (conf_id) exists and MSA2 is a super-member in my registrar
Request to invite MSA3 (conf_id)
Create a session with MSA3
Session create response
MSA3 join success
MSA3 join success
MSA3 join success
MSA3 join success
Conference state updated: conf_id; super-members: MSA1, MSA2 (MSA3)
Conference state updated: conf_id; super-members: MSA1, MSA2 (MSA3)

Figure 5.7: Inviting an MSA in a different routing area

5.3.3 Conference termination, party leaving and information propagation

The procedure for a 3G or a MANET party leaving has no difference from the party leaving procedures defined in a pure 3G network or in standalone MANETs. The difference is that the conference information has to be propagated correctly between the two networks when parties leave. This requires the translation and transmission of conference events between the 3G conference focus and the MANET super members. This functionality is performed in the CGW. 3GSAs and MSAs (cluster members) can subscribe the information from the conference focus and from super members, respectively, during a conference. When the last two participants terminate a conference, any sessions between the CGW and the conference focus, the MSA and the CGW, and the conference focus and the 3GSA are torn down.

A conference involving both 3G and MANET participants terminates only when the last participant leaves the conference. The relative sessions between CGW and MRFC should be released and all the conference state should be cleared.

5.4 Conclusions

In this chapter, we have proposed a signaling architecture for conferencing in integrated MANETs/3G networks. The architecture combines the cluster-based signaling architecture for standalone MANETs with the centralized signaling architecture for 3G networks. A CGW is proposed as a mediator. The architecture is very promising in meeting the specific signaling requirements for conferencing in integrated MANETs/3G networks. First, it supports all the required conferencing scenarios. Second, a MANET participant can directly create sessions with other MANET participants in the same routing area without a need to connect with a far away conference server. Third, the upgraded signaling entities that support the new architecture can be easily backward compatible with the signaling entities that have not been upgraded. This is due to the changes for upgrading the legacy signaling entities are not significant, i.e. the MSAs and the CF only need to load the CGW discovery functionality and the 3GSAs do not need to upgrade at all. The complexity of the architecture is put into the newly defined entity – CGW. Finally, the system can locate participants in both MANETs and 3G networks. This is also performed in CGW.

In Chapter 6 and Chapter 7, we will discuss the implementation and performance evaluation of the architecture.

Chapter 6 : Proof of Concepts and Evaluations

We have presented the signaling architectures for conferencing in both standalone MANETs and integrated MANETs/3G networks in Chapter 4 and Chapter 5, respectively. The implementation and evaluation of these architectures have to be addressed. In this chapter, we first discuss the implementation of the two architectures. We use SIP as an implementation technology. We then introduce the proof-of-concept prototypes we built. Results are collected from the prototypes. We also discuss how to build a real conference application. The cooperation of our signaling system and the media handling system in MANETs is addressed.

6.1 Implementation of signaling for conferencing in standalone MANETs

6.1.1 Implementation technology

We select SIP as an implementation technology. We also devise extensions to SIP so that the cluster-based architecture can be supported. The choice of SIP is based on the following reasons. First, SIP is a lightweight protocol. This fulfills the lightweight signaling requirement for MANETs. Second, SIP has been designed with extensibility in mind. Actually, there already exist several SIP extensions. Rosenberg [98] summarizes all these extensions. A guideline for the authors of SIP extensions is presented in the IETF draft [99] and it is not difficult to extend the core SIP and existing SIP extensions. Third, an implementation as extensions to SIP will ease inter-working with legacy systems, given that SIP is the sole signaling system used in the 3GPP 3G system. Fourth, extension to an existing signaling protocol reduces the validation workload comparing to development of a new protocol. Next, we introduce our SIP extensions.

6.1.2 Extensions to SIP

We identify our extensions by a new key word "*clustering*" in the "*Supported*" header field. The only functional entity we keep from SIP is the User Agent (UA). In addition to the UA, we introduce a new entity, the Super User Agent (SUA). UA maps naturally onto member and SUA onto super member. All the key concepts of SIP (e.g. dialog, call) remain unchanged. In addition, we introduce two new concepts: cluster and conference.

At any given time a UA or an SUA is in one and only one cluster. The information regarding this cluster is maintained in a UA. It consists of a unique cluster-ID, a set of parameters (*Sv* and *Mv*) related to the cluster, and a unique super-member ID. In the case of a SUA, the information includes in addition, the list of members and the list of neighbors. The newly introduced concept of conference facilitates the sequencing of the messages exchanged in a conference. It is uniquely identified in each UA and SUA with a conference ID. This ID is made up of a conference ID value, a set of dialogs and a cluster-ID.

We propose no new message, but extend the existing messages with new header fields. We use INVITE, ACK, BYE, CANCEL messages from core SIP [74]. We also use REFER [100], INFO [101], and SUBSCRIBE/NOTIFY [102] from existing SIP extensions. The INVITE, ACK, BYE and CANCEL handle the basic session establishment. The INFO and SUBSCRIBE/NOTIFY are used for information propagation. The former is used between SUAs and the latter is used between an SUA and a UA. A UA can subscribe to the conference membership information from its SUA.

We define a new event type: *update-participant* to identify this request event.

The REFER message is used in the following scenarios:

- When a UA wishes to invite a party to join, it needs to send a REFER to its SUA;
- When an SUA decides to leave, it sends a REFER to a newly selected SUA;
- When an SUA splits the cluster in which it locates, it sends a REFER to a new selected SUA;
- When two clusters merge, the SUA that wishes to merge may send a REFER to the SUA in the other cluster. We will illustrate the scenarios in detail in section 6.1.3.

We define four new headers: Conf-ID, Participant-In-Cluster, Neighbor and Cluster-Parameters. *Conf-ID* field acts as a unique identifier for all the messages exchanged during the multiparty session. *Participant-In-Cluster* represents all the members of a cluster and consists of a cluster-ID, a super-member, and a member-list. *Neighbor* represents a neighbor of the cluster represented by *Participant-In-Cluster*. *Cluster-Parameters* represents the parameters of the cluster, such as *Sv* and *Mv*. Table 6.1 shows the relationship between the new header fields and the SIP request methods.

	INVITE	ACK	BYE	CANCEL	REFER	SUBSCRIBE	NOTIFY	INFO
Conf-ID	M	M	M	M	M	M	M	M
Participant-In-Cluster	M	M	M	M	M	M	M	M
Neighbor	C	C	C	C	C	C	C	C
Cluster-Parameters	M	C	C	C	C	-	-	-

"M": Must appear in a method "-": Cannot appear in a method "C": May appear in a method

Table 6.1: Relationship between new header fields and SIP request messages

Implementation of the capability publication and discovery mechanism (see Section 4.3) is still based on SIP. We use the SIP event framework (SUBSCRIBE and NOTIFY) to implement the mechanism of Cap_notify and Cap_subscribe. We use the SIP INFO to implement Cap_publish. There is no extra extension required. We only define a new event type, called *node-capability*, and the event type should be presented in the 'Event' header field of the SIP messages if messages are used to discover or publish node capabilities.

6.1.3 Signaling scenarios in standalone MANETs

In this section, we will present six scenarios: conference initiation, participant joining, splitting, super-member leaving, merging and conference termination. All the scenarios are for ad hoc conferences.

Conference initiation – Figure 6.1 depicts conference initiation scenario. Party A starts a conference with party B. Before establishing a session, it collects B's capability and elects itself as an SUA. It then invites UA B and sets up a session. In the INVITE message, it passes the conference ID and the cluster information to UA B. UA B uses the information to build up its conference and cluster states. Whenever a session is established, the heartbeat of the session starts. The heartbeat messages are not shown here as they have been presented in Section 4.4.3. Once the session is established, the SUA A subscribes to the capability of the UA B and SUA A maintains a member capability list. This list may be used for future super-member elections. UA B may also subscribe to the conference membership from SUA A.

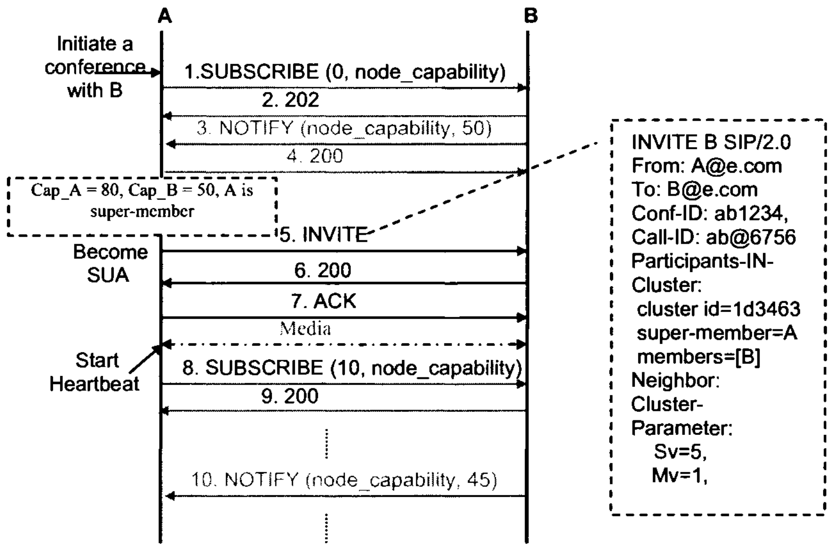


Figure 6.1: Conference initiation scenario for standalone MANETs

Participant joining – Figure 6.2 shows an example of participant joining. Party A and B are already in the conference. Party B wishes to invite a new party C to join the conference. UA B finds that it is not an SUA, so it asks its SUA A to establish a session with C. This is done by sending a REFER message. SUA A accepts this request and establishes a session with UA C. After the session is successfully established, C is added to the cluster member list. SUA A then notifies UA B, in response to the previous REFER messages. UA B receives the message and adds C to its local cluster. Similar to the first scenario, the heartbeat messages and capability subscriptions are exchanged between SUA A and UA C.

If it is a SUA that wishes to invite a new party to join, the SUA directly invites the party if it has enough processing capability. If it cannot handle more sessions, it will find a neighboring SUA, send a REFER message to the SUA and request it to invite the party. If there is no neighboring SUA available, the SUA will invite the party first. A splitting process will be performed right after the session establishment.

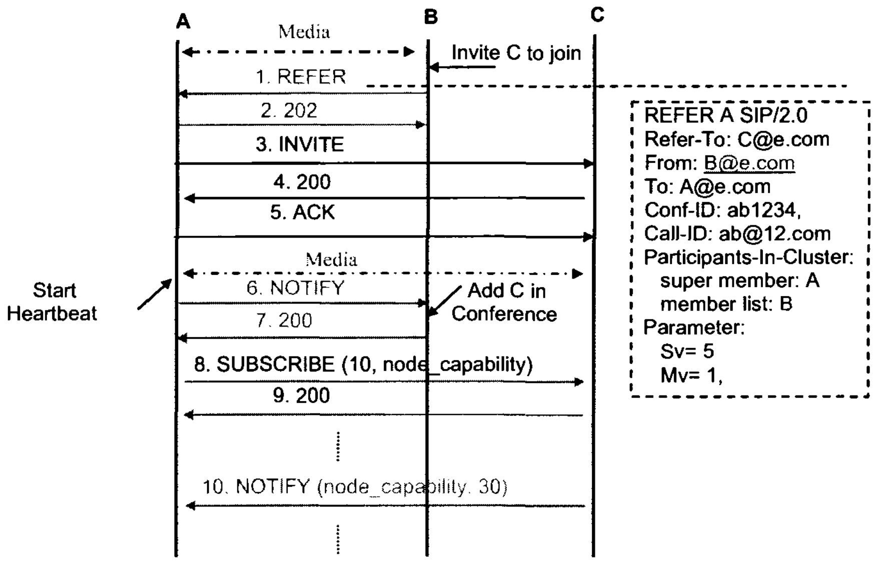
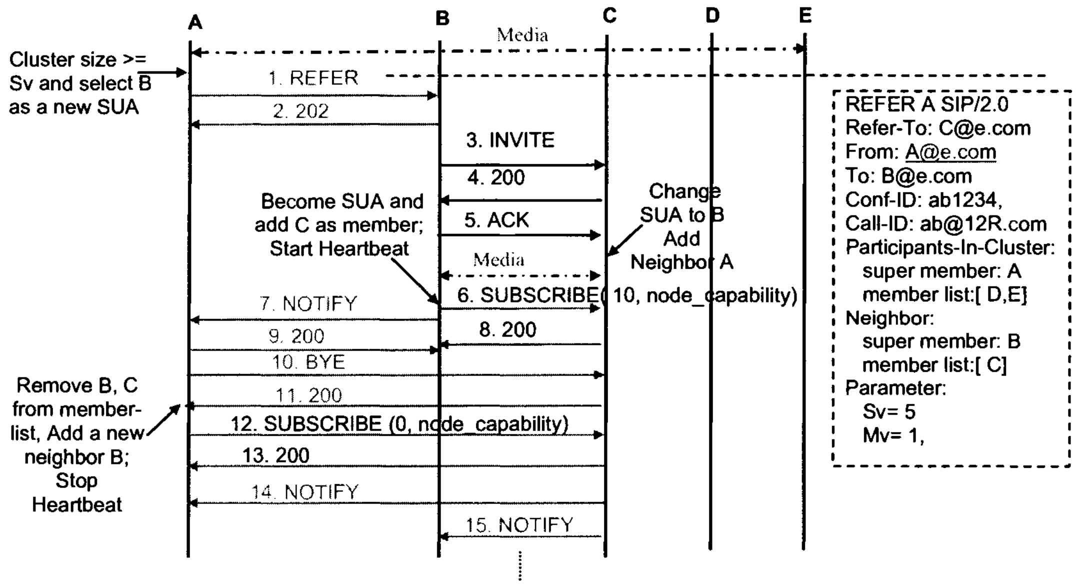


Figure 6.2: Participant joining scenario for standalone MANETs

Splitting – Figure 6.3 shows an example of splitting. Party A, B, C, D and E are already in a conference. Party A is the SUA and it finds that its cluster size has exceeded the *Sv*. It decides to split the cluster into two clusters: {A, D, E} and {B, C} and party B has been selected as the SUA of the new cluster. The splitting starts by sending a REFER message to B. This is to ask party B to invite party C. Party B that receives REFER finds that it will become a new SUA. It establishes a session with C and then starts to subscribe to C's capability. After that, it notifies party A that it has completed the invitation. Party A that receives the NOTIFY terminates the session with party C. This is done by sending a BYE message. It also stops subscribing party C's capability after the session is terminated.

The example shows the splitting of the only cluster in a conference. If there are other clusters in the conference, Party B also needs to create a sessions with the SUAs of these clusters. Party A also needs to send INFO messages to the neighboring SUAs after splitting finishes. The neighboring SUAs that receive the INFO will update their cluster states.


A
B
C
D
E
Media
Cluster size >= Sv and select B as a new SUA
1. REFER
2. 202
3. INVITE
4. 200
Become SUA and add C as member; Start Heartbeat
5. ACK
Change SUA to B Add Neighbor A
Media
7. NOTIFY
6. SUBSCRIBE(10, node_capability)
9. 200
8. 200
10. BYE
Remove B, C from member-list, Add a new neighbor B; Stop Heartbeat
11. 200
12. SUBSCRIBE (0, node_capability)
13. 200
14. NOTIFY
15. NOTIFY

```
REFER A SIP/2.0
Refer-To: C@e.com
From: A@e.com
To: B@e.com
Conf-ID: ab1234,
Call-ID: ab@12R.com
Participants-In-Cluster:
  super member: A
  member list:[ D,E]
Neighbor:
  super member: B
  member list:[ C]
Parameter:
  Sv= 5
  Mv= 1,
```

Figure 6.3: Cluster splitting scenario for standalone MANETs

Super-member leaving – Figure 6.4 shows an example of super-member leaving. Party A, D and E are in one cluster and party A is the SUA of the cluster. Party B and C are in another cluster and party B is the SUA. Party A wishes to leave the conference. It first selects party D as a new SUA and sends a REFER message to party D. Party D that receives the message starts to invite cluster member E and cluster neighbor B. After the invitation, it notifies party A. Party A then terminates sessions with its cluster members D, E and its neighboring SUA B by sending BYE messages. After all the sessions terminated, party A leaves the conference. Now there are still two clusters in the conference, party D, E in one cluster and party B, C in another.

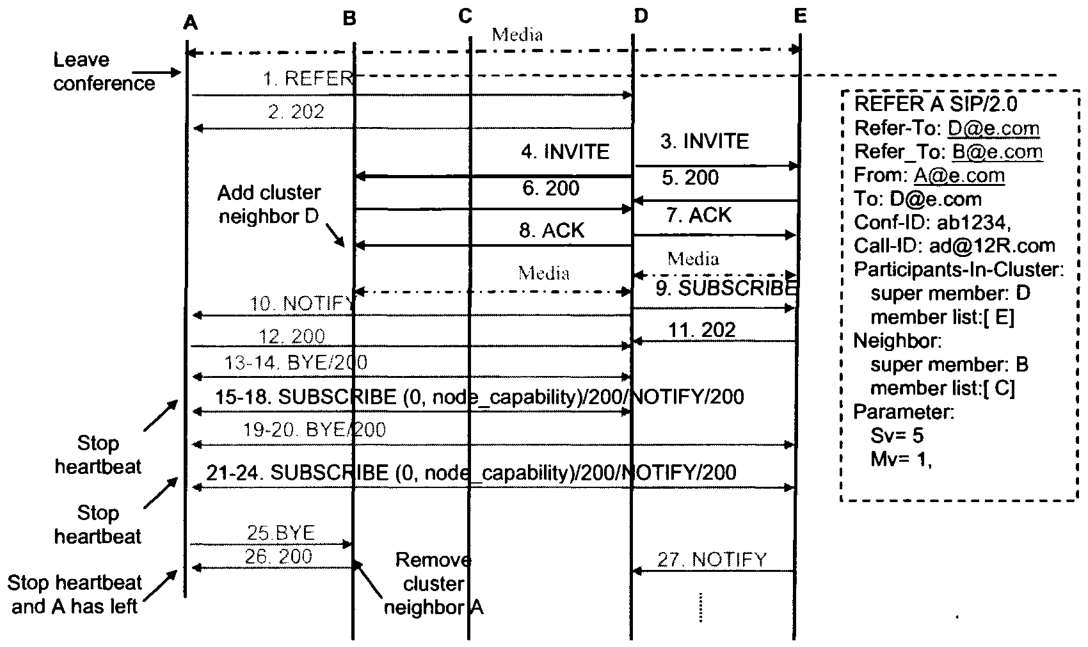
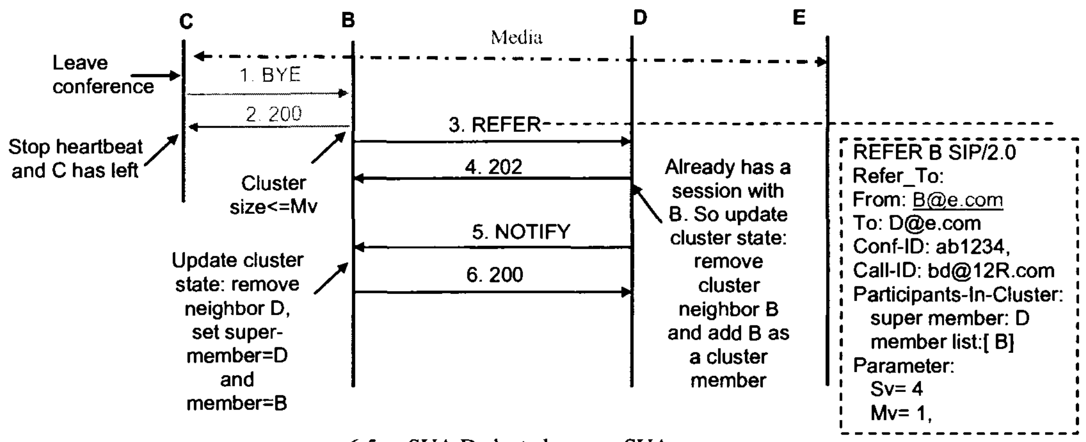


Figure 6.4: Super-member leaving scenario for standalone MANETs

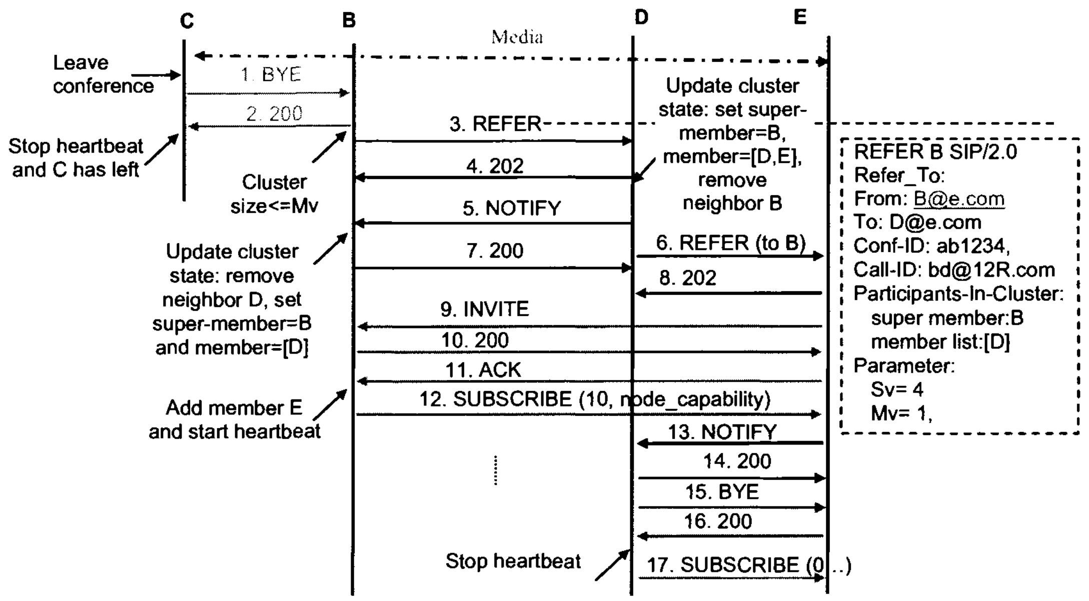
Merging – Figure 6.5 shows examples of cluster mergers. There are two clusters in the conference. Party B (SUA) and C are in one cluster and party D (SUA) and E are in another. The merging procedure is triggered by the leaving of party C. Party B finds that its cluster size has reached *Mv* so it checks if there is a neighboring cluster that it can merge with. It then finds that it can merge with party D's cluster.

Figure 6.5-a depicts that party B elects party D as a new SUA. In this case, no session needs to be established or terminated, but only cluster states updated in both party B and D. A REFER message is used for this merger.

Figure 6.5-b shows that party B elects itself as a new SUA. The merger still starts by a REFER message. Party D that receives the REFER message updates its cluster state and asks its old cluster member E to establish a session with the new SUA B. This is still done by a REFER message. In this message, the 'Refer-To' header field is set to B. Party E that receives this REFER changes its super-member to B and send an INVITE message to party B. After party E and B's session established, party E notifies party D and finally, party D terminates its connection with party E.



6.5-a: SUA D elected as new SUA



6.5-b: SUA B elected as new SUA

Figure 6.5: Cluster merging scenario for standalone MANETs

Conference termination – Figure 6.6 shows an example of conference termination. In the example, the conference termination is triggered by the leaving of SUA A. Party A finds that there is no cluster neighbors and there is only one cluster member left. Instead of electing a new SUA, it directly sends a BYE message to party B. Party B that receives this message responds to the message and then leaves the conference because it has terminated the last session it maintains.

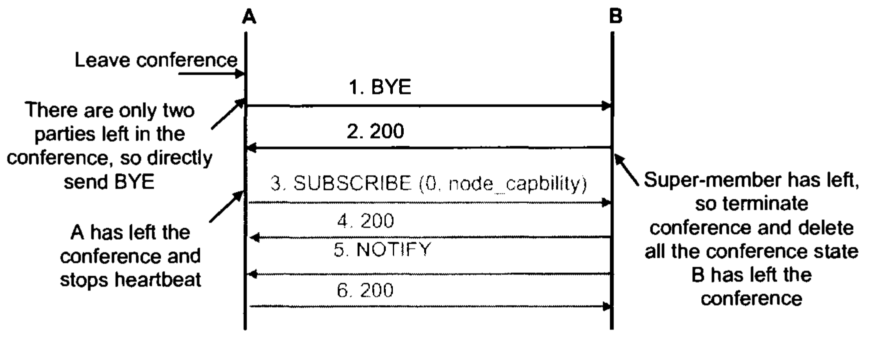

A
B
Leave conference
1. BYE
There are only two parties left in the conference, so directly send BYE
2. 200
3. SUBSCRIBE (0, node_capbility)
Super-member has left, so terminate conference and delete all the conference state
4. 200
A has left the conference and stops heartbeat
5. NOTIFY
B has left the conference
6. 200

Figure 6.6: Conference termination scenario for standalone MANETs

6.1.4 SIP deployment in MANETs

We have shown that the proposed SIP extension can implement the cluster-based signaling architecture in difference conference scenarios. However, when we wish to implement these scenarios in a real MANET environment, we face the problem of SIP deployment, i.e. without fix servers such as proxy servers or registrars deployed, there is no way for SIP UAs to discover and locate each other. This issue can be divided into three sub-issues. First, a UA should be able to become aware of the presence of other UAs in the network. Second, a UA should know the location of a destination UA. Third, a UA should be able to forward the SIP messages to a correct destination.

In the literature, we have found five solutions for deployment of SIP in MANETs. We denote them as S1, S2, S3, S4 and S5. We will present these solutions, analyze their advantages and disadvantages and choose the most appropriate ones for our implementation.

S1 is presented in [83]. It is a very simple mechanism that relies on the lower-layer ad hoc routing protocol. The basic idea is as follows: a participant that wishes to establish a conference broadcasts REGISTER messages. The REGISTER message contains the user name, IP address and listening port. The REGISTER message are flooding, using the underlying ad-hoc routing protocol. This solution can solve all the sub-issues of SIP deployment and it is easy to implement. However, the flooding of REGISTER message may significantly increase the system overhead.

S2 [103] defined a solution for peer-to-peer SIP-based services over wireless ad hoc networks. The solution still relies on lower-layer routing protocol. Its difference from S1 is that it designates a cluster-based ad hoc routing protocol. It then integrates SIP severs with the routing layer clusters. Cluster heads act as SIP proxies and as the forwarding nodes. They also act as registrars for their cluster members. The request messages are sent to the corresponding proxy of the request node. The proxy then sends the messages to neighboring cluster heads in order to discover a route to the destination. One disadvantage of this solution is that SIP UAs have no ways to know each other. Also, the solution relies on specific routing schemes that a MANET might not support.

S3 [104] proposed a decentralized approach in which SIP registrars and proxy servers are distributed in each participant. When a new participant joins the MANET, it broadcasts a REGISTER message to all the participants. The registrar in each party will store the information. It will also respond the message with its own location information. The "broadcast" here refers to a general concept. In more specific, it may be a link-layer broadcast if every two participants are directly linked. It may be multicast if all SIP UAs have registered to pre-defined multicast address. It may also be selective flooding. A user location stored in each registrar has a limited validity time. When the location information expires, a refreshed REGISTER message should be sent. Any request message is forwarded to local proxy. The proxy is responsible for contacting the local registrar, getting the location information of the destination and then sending the message. The solution can solve all the sub-issues of SIP deployment in MANETs. It provides a way to reuse all the SIP entities in SIP layer. However, since the registrar and proxy server are fully distributed in participant, the complexity of end system has increased. Also, the broadcast of REGISTER messages may introduce flooding problem.

Leggio et al. [104] also provide an alternative solution S4, which uses an existing service discovery platform. The platform provides a set of APIs for the SIP UAs to issue service discovery queries. These queries may include a service-query (i.e. finding a user that is online and provides a SIP service) or a location-query (i.e. finding IP address of a user by a user name). The user that provides a SIP service should first register the service with the platform. Examples of such platforms are JXTA [105] and SIP multicast framework [90]. This solution can solve all the sub-issues of SIP deployment in MANETs, but a platform that provides service discovery in MANETs should be deployed first.

The last solution S5 is from IETF P2P SIP [106]. It builds a peer-to-peer overlay layer that provides a distributed resource placement and search for SIP. The overlay layer serves the SIP layer but it is independent of the SIP layer. The DHT (Distributed Hash Table) is used for the P2P overlay layer. DHT algorithms such as Chord [107] and Pastry [108] can be used. Two types of peers are defined: Super Nodes (SNs) and Ordinary Nodes (ONs). Only the SNs participate in building and maintaining the DHT. Each ordinary node is associated with one or more super nodes. Super nodes are not necessarily become proxy or registrar, but they are good candidates to be. The registration procedure follows two steps. First, a UA uses P2P layer API to store its user location information in the P2P overlay network. Second, the UA periodically refreshes the user location record. The session establishment process can be described as follows: upon the receipt of a call establishment command from the application, a UA (ordinary node) sends a request of the location of callee to a super node connected. The super node then starts a search procedure in the overlay network (the procedures is based on what DHT algorithm is used). After that, it sends back a response with the callee's location information included. As the last step, the caller directly sends an INVITE to callee.

S5 is a comprehensive solution that can not only be used for MANETs but also for other types of P2P networks. It can solve all the sub-issues of SIP deployment in MANETs. However, it is not easy to implement an overlay network. In addition to this, the protocols between SN and SN and between ON and SN are yet to be defined.

In Table 6.2, a review of the five SIP deployment solutions is presented. In these solutions, we do not want to choose S1 as our implementation approach because it has flooding problem and it cannot scale. S2 is not suitable because it relies on specific routing protocol and it cannot handle UA discovery. S5 is not easy to implement and the related specifications have not been completed yet.

S3 and S4 are possible solutions to be selected. For our prototype, we choose the solution S4 because a use of existing platform reduces time and workload in the prototype development. We have selected JXTA as a platform. JXTA is a well developed, open-source middleware for peer-to-peer service publication/discovery. It can also run in MANETs. We will discuss it further in Section 6.1.5.2. We choose S3 for simulation because JXTA cannot be directly used in the OPNET simulation environment. In order to overcome the issue of register flooding, we modify the registration procedure of S3 as follows.

Only selected participants in the MANETs act as real registrars. They broadcast a subscription for registration. If a participant receives such a subscription, it will periodically send back a REGISTER message that contains its current location information. Upon receipt of REGISTER messages, a participant will update its local registrar. Periodically, the party that acts as a real registrar publishes the information stored in its registrar, i.e. it sends a message (containing the updated locations of parties) to every participant that has registered. Any party that receives this message will use the information to update its local registrar.

Functionality supported	S1	S2	S3	S4	S5
Registration	Yes	Yes	Yes	Yes	Yes
Discovery	Yes	No	Yes	Yes	Yes
Addressing	Yes	Yes	Yes	Yes	Yes
Other Pros and Cons					
Pros	Simple	performance	Reuse all SIP entities	Reuse the discovery service or middleware that may already exist	Applicable for all the P2P networks (MANET and infrastructure based)
Cons	Flooding, Small scale	Rely on specific routing protocol	End point complexity (UA+Proxy+Registrar), Flooding of REGISTER	The performance of the discovery protocol may be an issue	Complexity of the overlay network deployment, SN-SN, ON-SN protocols are yet to be defined

Table 6.2: Review of solutions for SIP deployment in MANETs

6.1.5 A signaling prototype for standalone MANETs

We have built a prototype for signaling in standalone MANETs. This work allows us to prove the concept. In this section, we will first introduce the setting of the prototype and the tools that we use for implementing the prototype. We will then introduce the design and software architecture. After that, some results will be presented.

6.1.5.1 Environment settings

We have created a MANET environment using 4 PCs and 3 laptops with IEEE 802.11g adaptive cards. These machines are Pentium 4s or mobile Pentium 4s with 512 MB RAMs running Windows XP. The machines are located in two different rooms on the same floor. The settings are shown in Figure 6.7.

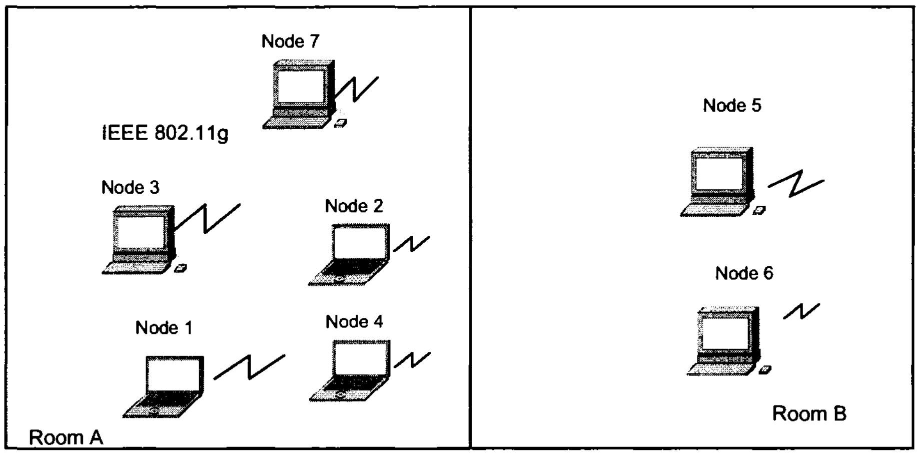


Figure 6.7: Prototype settings

6.1.5.2 Programming language and tools

We use Java as a programming language and two types of Java freeware are used in the prototype. One is JAIN SIP that is the basis for implementing our SIP extensions and the other is JXTA that is used for participant discovery.

JAIN SIP implements SIP stack using Java. Its source code can be found for download at [110] and reference [111] provides a specification. JAIN SIP provides developers a standardized Java interface for SIP services. The latest version of it supports core SIP

(RFC 3261), as well as INFO method (RFC 2976), reliability of provisional responses (RFC 3262), Event Notification Framework (RFC 3265), the UPDATE method (RFC 3311), the Reason Header (RFC 3326), the Message method defined for instant messaging (RFC 3428) and the REFER method (RFC 3515). Since JAIN SIP has implemented all the messages defined in core SIP and major SIP extensions, it is a good choice for us to use as a basis of our signaling system. We use the headers, messages, and dialog that it provides. We also extend it with our new headers and new concepts.

JXTA is an implementation of peer-to-peer middleware. It includes a set of open protocols that have been specifically designed for ad hoc, pervasive and multi-hop peer-to-peer network computing, allowing any connected device (cell phone to PDA, PC to server) on the network to communicate, collaborate, and share resources. JXTA is an open-source project. It provides public APIs for programmer to implement applications on it. The source codes and specifications of it can be found in [105].

JXTA has not been specially designed for MANETs. It cannot handle the frequent changes inherent to MANETs. Some extensions have been proposed (e.g. [112]) for enhance JXTA, but unfortunately, there is no implementation. However the basic discovery and pipe services of JXTA can still be used in MANETs. This is the reason why we choose it as a SIP UA discovery platform. The discovery service is performed in three steps. First, a UA self-configures as a JXTA peer and an XML-based peer advertisement will be generated. Second, a UA publishes itself (its advertisement) onto the network. This is done by calling the JXTA API *remotePublish()*. Third, any other JXTA peer can find a published peer by calling the function *getRemoteAdvertisement()*. This function performs a search for specified type of JXTA peers on the network.

6.1.5.3 Prototype design

We have implemented all the SIP extensions that we proposed. Figure 6.8 shows the software architecture of the prototype. The application layer implements a simple conference application including the graphical interfaces through which user can initiate a conference, invite a party and leave a conference. The signaling layer implements the extended SIP UA. It is an event-driven state machine (part of the state machine is depicted in Figure 6.9). The signaling layer also provides APIs (such as *invite(), bye(), accept()* and *reject()*) for application layer. The middleware layer is based on JXTA and it provides peer discovery and transport service for the signaling layer.

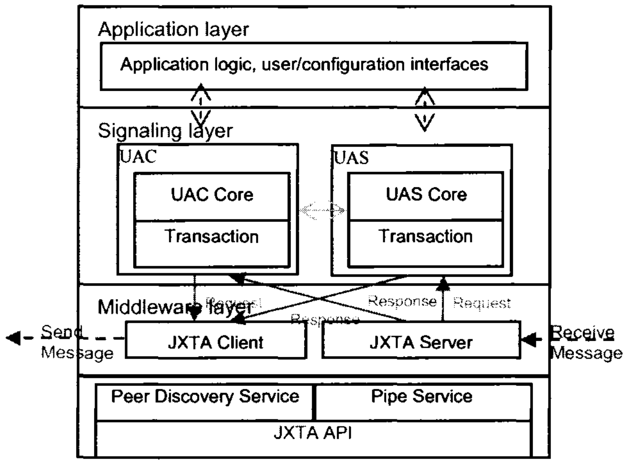


Figure 6.8: Signaling prototype for standalone MANETs – software architecture

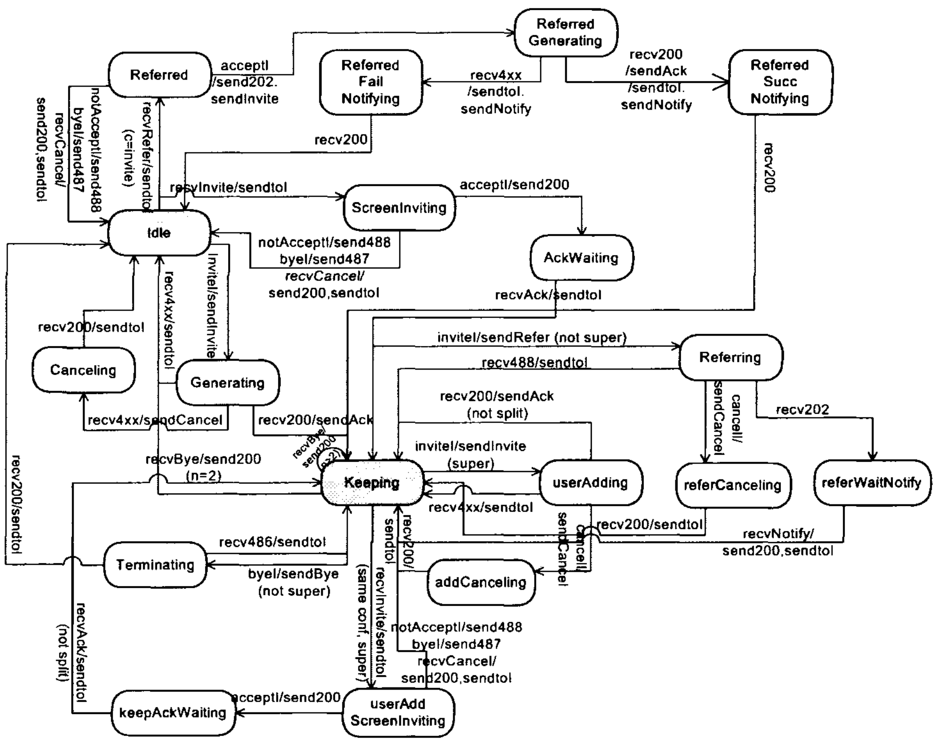


Figure 6.9: Part of the state machine of extended UA/SUA

6.1.5.4 Experiment scenarios and results

We have run the prototype and done experiments on it. The basic scenarios, such as conference establishment, party joining, member leaving, super-member leaving, splitting and conference termination, are tested and they all work well. Table 6.3 shows an example of INVITE and 200 messages exchanged when establishing a session between two users: *concord* and *chunyan.*

```
INVITE sip:concord@142.133.72.76:5060;method=INVITE ;transport=tcp SIP/2.0
Call-ID: 79440813c515a200e862c6e6ca91b683@142.133.72.82
CSeq: 1 INVITE
From: <sip:chunyan@142.133.72.82>;tag=9523050
To: <sip:concord@142.133.72.76>
Via: SIP/2.0/TCP 142.133.72.82:5060;branch=z9hG4bK5d8c20049cda3e16ba5554007fd22912
Max-Forwards: 70
Contact: <sip:chunyan@142.133.72.82:5060>
Conf-ID: this_is_a_conf_id0.5395847102284398
Participant-In-Cluster: cluster-id=This_is_cluster_id0.9028286510059693
          super=sip:chunyan@142.133.72.82 members=[sip:concord@142.133.72.76]
Cluster-Parameters: sv=4 mv=0
Supported: Clustering
Allow: INFO
```

```
SIP/2.0 200 OK
Call-ID: 79440813c515a200e862c6e6ca91b683@142.133.72.82
CSeq: 1 INVITE
From: <sip:chunyan@142.133.72.82>;tag=9523050
Via: SIP/2.0/TCP 142.133.72.82:5060;branch=z9hG4bK5d8c20049eda3e16ba5554007fd22912
Max-Forwards: 70
To: <sip:concord@142.133.72.76>;tag=8140933
Contact: <sip:concord@142.133.72.76:5060>
Conf-ID: this_is_a_conf_id0.5395847102284398
Participant-In-Cluster: cluster-id=This_is_cluster_id0.9028286510059693
          super=sip:chunyan@142.133.72.82 members=[sip:concord@142.133.72.76]
Cluster-Parameters: sv=4 mv=0
```

Table 6.3: Example of signaling messages -- INVITE and 200

We have collected a result on average session establishment delay (shown in Table 6.4). The result is based on the following scenario: Node 7 is an initiator of the conference. It first invites Node 1 to establish a conference. It then invites Node 2 to Node 6 respectively to join the conference. The *Sv* is set to 4 in the tests. The values in the table are in seconds and they are averaged from ten tries. From the results, we can find that conference initiation always takes longer than a party joining. This is reasonable because the first two parties need to exchange their capability before establishing the session. Another result that we find is that session delays are affected by the distance between the two parties that wishes to set up a session. For example, in the table, the nodes located in a different room from the initiator take longer to join the conference.

	Node 1	Node 2	Node 3	Node 4	Node 5*	Node 6*
Average per session delay (seconds)	1.09	0.33	0.41	0.35	0.46	0.53

* A node that does not locate in the same room as the conference initiator

Table 6.4: Average per session setup delay for standalone MANET signaling prototype

We have also evaluated the effect of cluster parameter *Sv*. From two to seven-party conferences have been evaluated with *Sv* settings from 1 to 8. Figure 6.10 shows the result and the signaling overhead (in terms of number of messages) is collected. In the figure, we can see that when the *Sv* is one, signaling system generates most number of messages. In this case, the signaling architecture is actually a full-mesh. When the *Sv* is set to a value larger than the number of participants in the conference, least numbers of messages are generated, but the architecture becomes centralized. It is difficult to find an optimal Sv with which the signaling system generates less number of messages but the system architecture does not become centralized. However, an interesting result shown in Figure 6.10 is that for six and seven-party conferences, if we set the split value as 4, less number of messages is generated than applying other split values (not include the centralized one). Forasmuch, there may exist optimal split values for a conference. We will further investigate this issue in simulation that allows us to test in larger-scaled conferences.

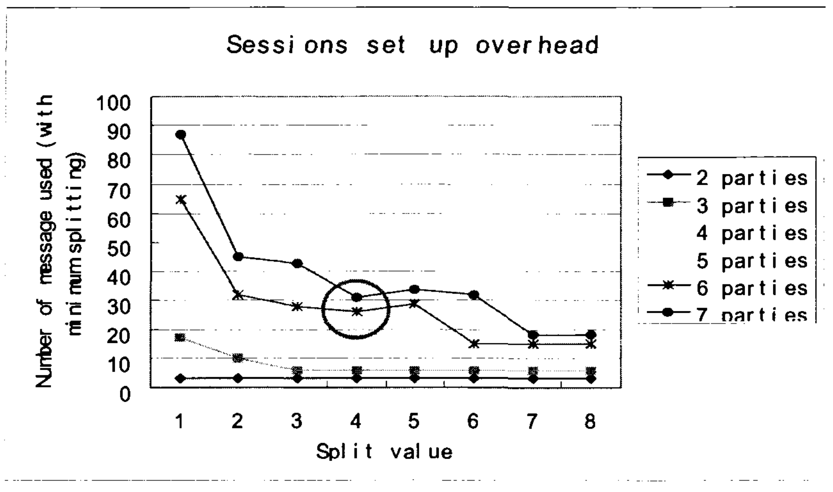


Figure 6.10: Result from standalone MANET prototype – effect of Sv

6.2 An Implementation of signaling for conferencing in integrated MANETs/3G Networks

We have defined SIP extensions for conferencing in standalone MANETs. Since SIP is also the signaling protocol for 3GPP 3G conferencing, using it to implement the integrated architecture is a straightforward solution.

6.2.1 SIP as implementation technology

We do not need to modify the SIP clustering extension defined in Section 6.1.2. It still acts as the signaling protocol in MANETs. Also, we do not need to modify the SIP used in 3G networks. Instead, we enhance the functionality of SUA in MANETs and of MRFC/AS in 3G networks. The enhancements are related to the CGW discovery. We assume that the MRFC/AS is pre-configured with known CGWs. The MANET participants and the CGWs support one of the discovery mechanisms discussed in Section 5.2.2. A SIP REGISTER message is used by the MSAs to register with CGW.

In session management, the CGW plays a key role. It not only maintains conference and cluster status but it also acts as a SIP SUA that sends, responds and processes SIP messages. Signaling translations are also performed in the CGW. The same applies to the mapping of conferencing architecture. Since both 3G and MANET sides use SIP, no entity-address translation is needed. The required translations are conference identifications, some SIP header fields and conference events.

On MANET side, participants use *Conf-ID* to identify a conference, while in 3G networks they use a *conference-URI* generated by the conference focus. *Conf-ID* is a header field in extended SIP while *conference-URI* is a parameter carried in the *Contact* header field of the baseline SIP. The CGW that receives an SIP message from the 3G side needs to look at the *Contact* field, get the *conference-URI*, use it to generate a *Conf-ID* header field and then send the message to MANET side.

The header fields that need to be translated between the two types of networks are clustering header fields, i.e. the *Participant-In-Cluster*, *Neighbor* and *Cluster-Parameters* should be removed by the CGW when a message is forwarded to the 3G conference focus, and they should be added when a message travels in the opposite direction.

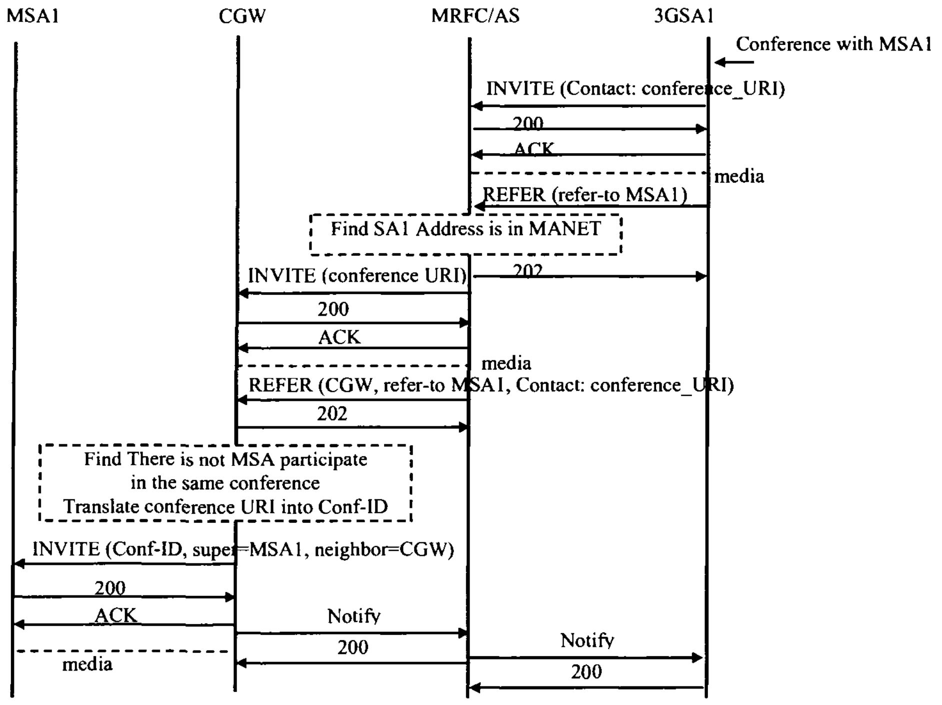
Participants in both MANET and 3G networks may subscribe to conference events using the SIP event notification framework. However, the types of events defined on both sides are different. 3G uses a standard *conference* event package defined in IETF RFC 4575 [109], while clustering-based architecture uses a simplified event *update-participant* that only includes conference membership. Therefore, mappings are necessary. A CGW needs to generate simple events when it passes a conference event to MANETs. It needs to generate XML documents (including information such as *conference_description*, *media* and *joining-method*) when an event is passes to 3G network side.

In the next section, we will show how SIP works in integrated MANETs/3G networks through sequence diagrams.

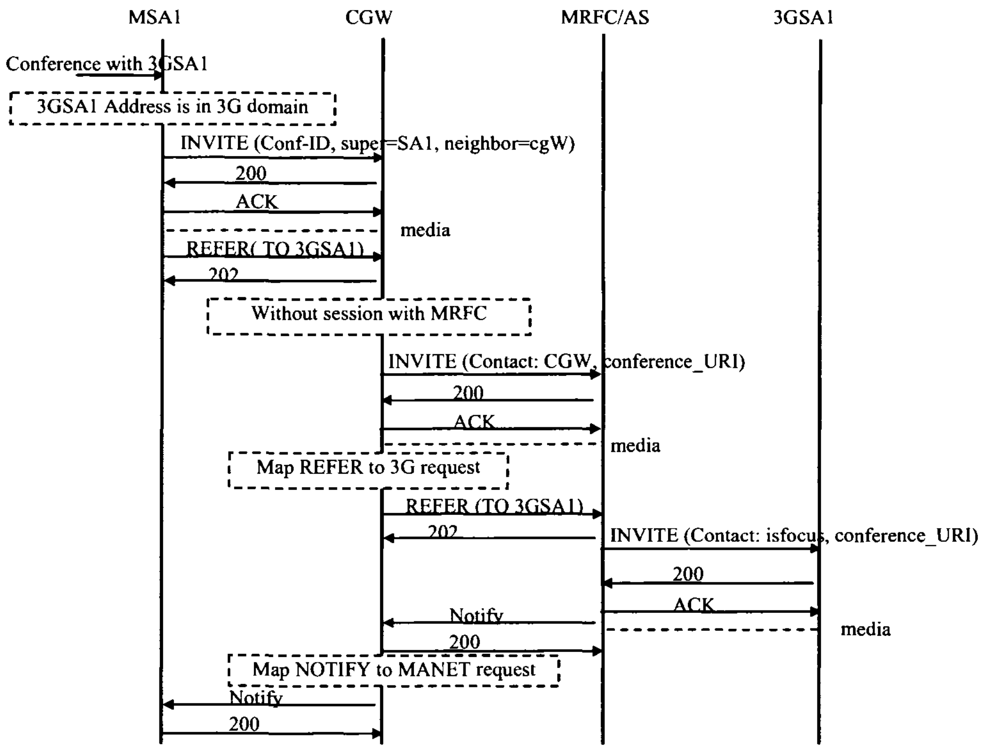
6.2.2 Signaling scenarios in integrated MANETs/3G networks

Four types of scenarios are illustrated: conference initiation, participant joining, event subscription/notification and conference termination.

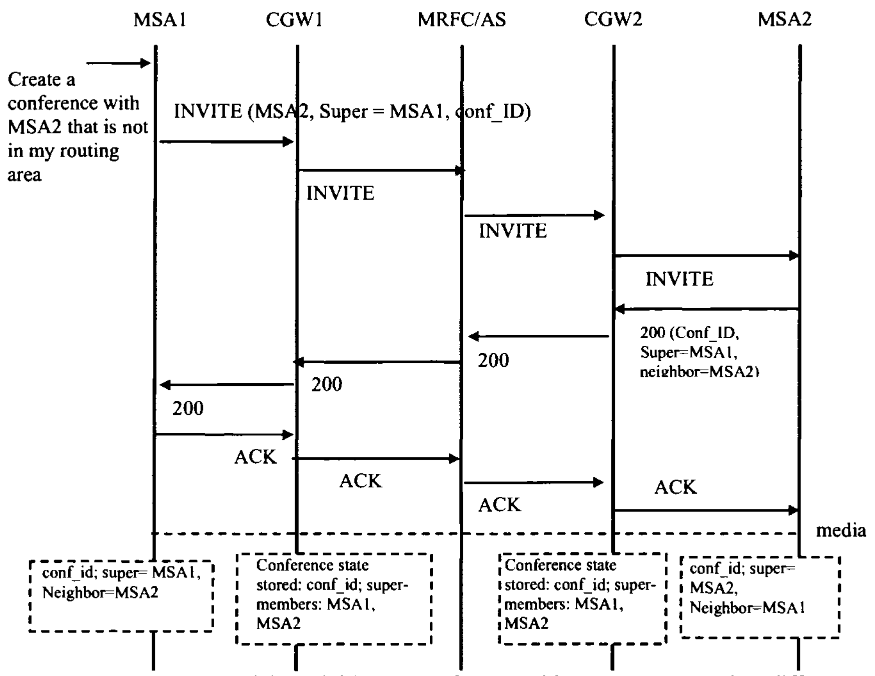
Conference initiation – Figure 6.11 shows three scenarios of conference initiation: a 3G party initiates a conference with a MANET party (Figure 6.11-a), a MANET party initiates a conference with a 3G party (Figure 6.11-b) and two MANET parties located in different routing areas establish a conference (Figure 6.11-c). It presents the SIP implementation of the scenarios shown in Section 5.3.1.


MSA1
CGW
MRFC/AS
3GSA1
Conference with MSA1
INVITE (Contact: conference_URI)
200
ACK
media
REFER (refer-to MSA1)
Find SA1 Address is in MANET
INVITE (conference URI)
202
200
ACK
media
REFER (CGW, refer-to MSA1, Contact: conference_URI)
202
Find There is not MSA participate in the same conference Translate conference URI into Conf-ID
INVITE (Conf-ID, super=MSA1, neighbor=CGW)
200
ACK
Notify
200
Notify
200
media

6.11-a: A 3G participant initiates a conference with a MANET party


MSA1
CGW
MRFC/AS
3GSA1
Conference with 3GSA1
3GSA1 Address is in 3G domain
INVITE (Conf-ID, super=SA1, neighbor=cgW)
200
ACK
media
REFER(TO 3GSA1)
202
Without session with MRFC
INVITE (Contact: CGW, conference_URI)
200
ACK
media
Map REFER to 3G request
REFER (TO 3GSA1)
202
INVITE (Contact: isfocus, conference_URI)
200
ACK
media
Notify
200
Map NOTIFY to MANET request
Notify
200

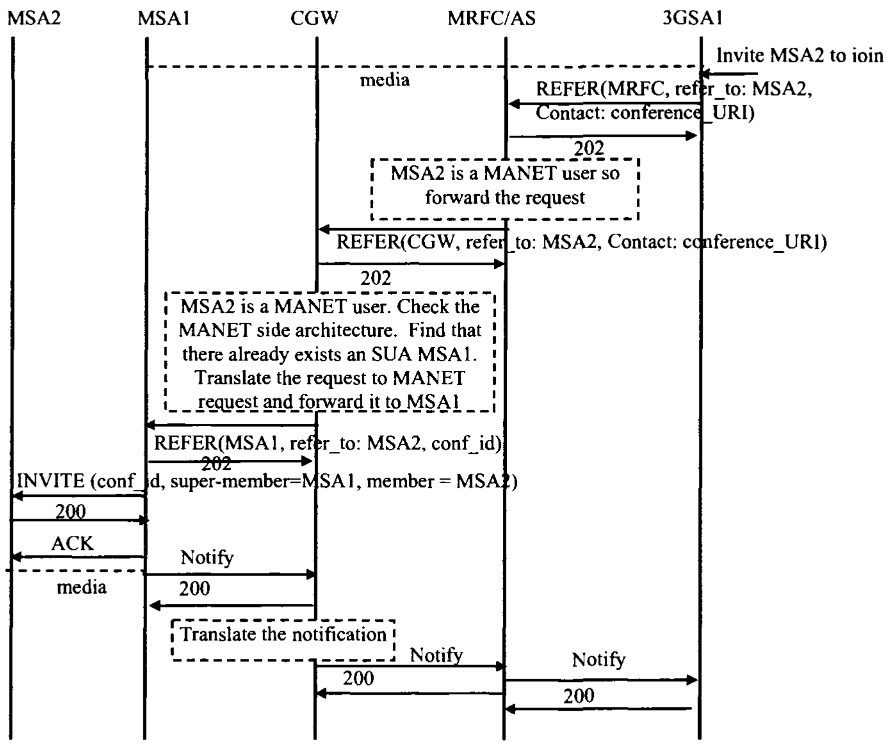
6.11-b: A MANET participant initiates a conference with a 3G party


MSA1
CGW1
MRFC/AS
CGW2
MSA2
Create a conference with MSA2 that is not in my routing area
INVITE (MSA2, Super = MSA1, conf_ID)
INVITE
INVITE
INVITE
200 (Conf_ID, Super=MSA1, neighbor=MSA2)
200
200
200
ACK
ACK
ACK
ACK
media
conf_id; super= MSA1, Neighbor=MSA2
Conference state stored: conf_id; super-members: MSA1, MSA2
Conference state stored: conf_id; super-members: MSA1, MSA2
conf_id; super= MSA2, Neighbor=MSA1

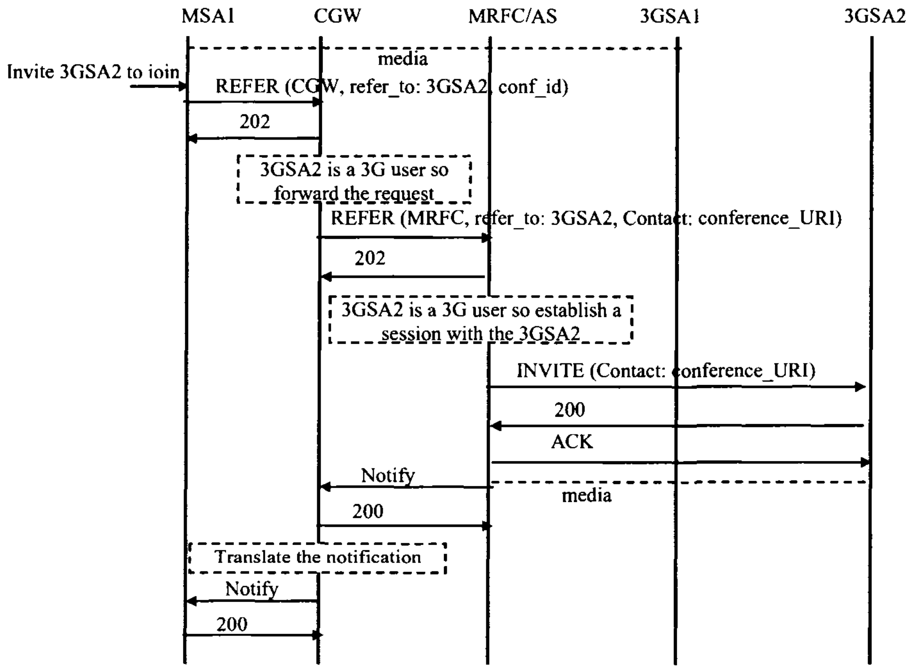
6.11-c: A MANET participant initiates a conference with a MANET party in a different routing area

Figure 6.11: Conference initiation scenarios for integrated MANETs/3G networks

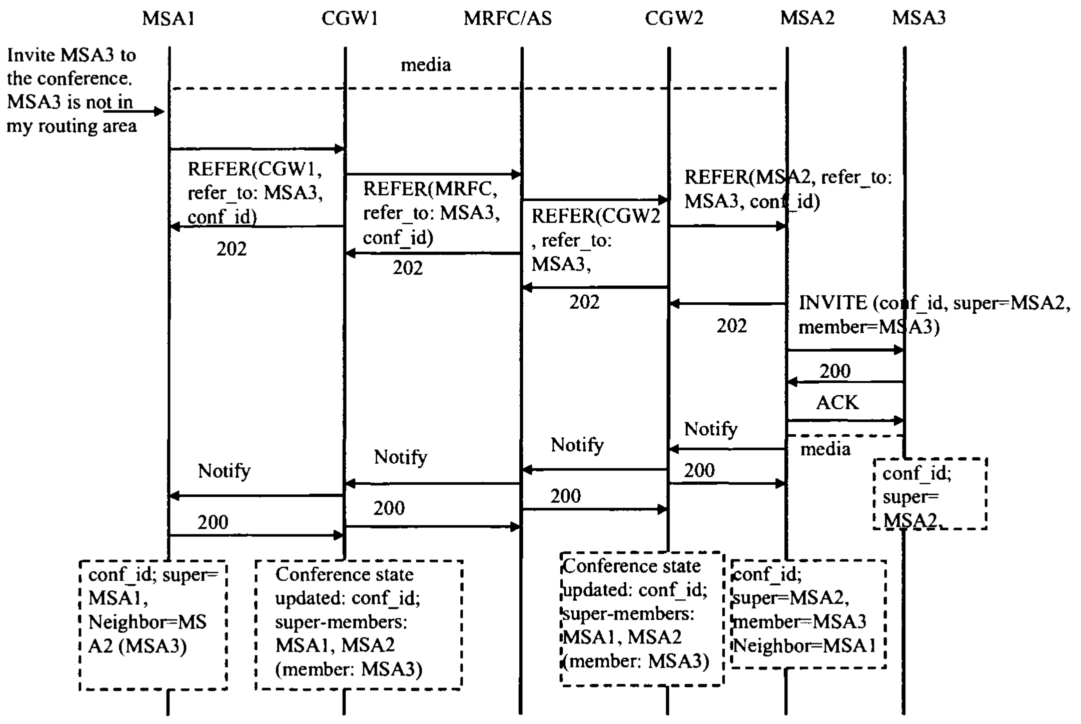
Participant joining – Figure 6.12 shows three scenarios of participant joining: a 3G party invites a MANET party to join (Figure 6.12-a), a MANET party invites a 3G party to join (Figure 6.12-b) and a MANET party invites another MANET party that is located in different routing areas to join (Figure 6.12-c). It presents the SIP implementation of the scenarios shown in Section 5.3.2. From the figure, we can find that when inviting a 3G party, it is always the MRFC/AS that is responsible for session setup. When inviting a MANET party, it may be the CGW that sets up the session (if it is the first MANET party to be invited to the conference) or may be an existing MSA (SUA) that sets up the session. In the latter case, the CGW should select an SUA with the most capability to establish the session if there are more than one SUA.


MSA2
MSA1
CGW
MRFC/AS
3GSA1
Invite MSA2 to join
media
REFER(MRFC, refer_to: MSA2, Contact: conference_URI)
202
MSA2 is a MANET user so forward the request
REFER(CGW, refer_to: MSA2, Contact: conference_URI)
202
MSA2 is a MANET user. Check the MANET side architecture. Find that there already exists an SUA MSA1. Translate the request to MANET request and forward it to MSA1
REFER(MSA1, refer_to: MSA2, conf_id)
202
INVITE (conf_id, super-member=MSA1, member = MSA2)
200
ACK
media
Notify
200
Translate the notification
Notify
200
Notify
200

6.12-a: A 3G participant invites a MANET party


MSA1
CGW
MRFC/AS
3GSA1
3GSA2
media
Invite 3GSA2 to join
REFER (CGW, refer_to: 3GSA2, conf_id)
202
3GSA2 is a 3G user so forward the request
REFER (MRFC, refer_to: 3GSA2, Contact: conference_URI)
202
3GSA2 is a 3G user so establish a session with the 3GSA2
INVITE (Contact: conference_URI)
200
ACK
media
Notify
200
Translate the notification
Notify
200

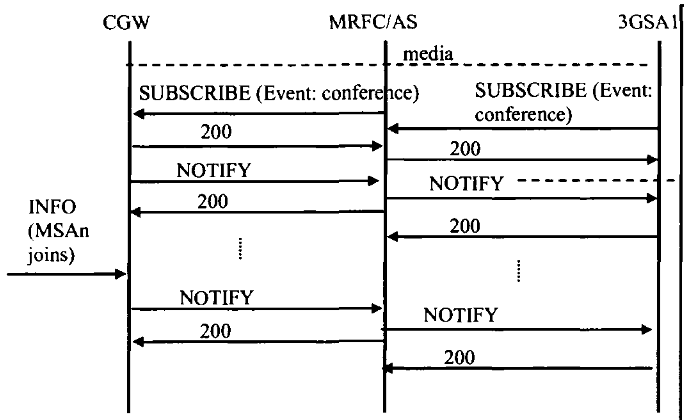
6.12-b: A MANET party invites a 3G party


MSA1
CGW1
MRFC/AS
CGW2
MSA2
MSA3
Invite MSA3 to the conference. MSA3 is not in my routing area
media
REFER(CGW1, refer_to: MSA3, conf_id)
202
REFER(MRFC, refer_to: MSA3, conf_id)
202
REFER(CGW2, refer_to: MSA3,
202
REFER(MSA2, refer_to: MSA3, conf_id)
202
INVITE (conf_id, super=MSA2, member=MSA3)
200
ACK
media
conf_id; super= MSA2
Notify
200
Notify
200
Notify
200
Notify
200
conf_id; super= MSA1, Neighbor=MSA2 (MSA3)
Conference state updated: conf_id; super-members: MSA1, MSA2 (member: MSA3)
Conference state updated: conf_id; super-members: MSA1, MSA2 (member: MSA3)
conf_id; super=MSA2, member=MSA3 Neighbor=MSA1

6.12-c: A MANET party invites a MANET party in different routing areas

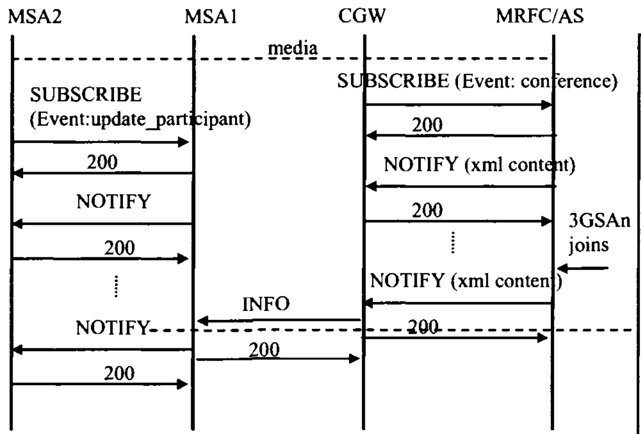
Figure 6.12: Participant joining scenarios for integrated MANETs/3G networks

Conference event subscription/notification – Figure 6.13 shows how 3G parties and MANET parties subscribe conference-related information. Figure 6.13-a depicts the case of a 3G party event subscription. Figure 6.13-b shows the case of a MANET party (UA) event subscription. It should be noted that CGWs and MRFC/AS always maintain the last update of the conference status. This is done by SIP event notification framework, i.e. when a session between CGW and MRFC/AS is established, a CGW should subscribe *conference* event from MRFC/AS and MRFC/AS also subscribe *conference* event from CGWs. When there is a participant or media change, the CGW or the MRFC/AS that is aware of the change will send NOTIFY to the other. For example, in Figure 6.13-a, when CGW is informed that a new MSA has joined, it notifies MRFC/AS, and the MRFC/AS then notifies 3GSA1 because the 3GSA1 has subscribed the *conference* event. The format of the NOTIFY message is shown in the figure. In Figure 6.13-b, when a new 3G party has joined conference, MRFC/AS notifies CGW and the CGW that acts as an SUA informs its neighboring SUA MSA1 by an INFO message. The MSA1 then notifies the MSA2 that has subscribed the *update-participant* event. The format of the NOTIFY message shown in the figure is different from that in depicted Figure 6.13-a.



6.13-a: A 3G party subscribes event

NOTIFY
From: focus@e.com
To:3GSA1@e.com
Subscription-State: active ;expires=7200
Event: conference
Contact: conference_URI@
Content-Type: application/conference-info+xml
Content-Length: (...)
<?xml version="1.0" encoding="UTF-8"?>
<conference-info version="0" state="full", entity= "conference1@e.com" xmlns="urn:ietf:params:xml:ns:conference-info"> <user uri="msan@e.com",display-name="Alice"><status>connected</status>< media-stream media-type ="audio"> <proto> RTP/AVP </proto> </media-stream> </user>... </conference-info>



NOTIFY
From: msa1@e.com
To: msa2@e.com
Subscription-State:
active ;expires=7200
Event: update_participant
Conf_ID: conference_URI@
Cluster_in_participant : super-member: msa1, member: msa2
Neighbor: super-member: cgw, member:[mrfc/as, 3gsa1, 3gsa2, 3gsan]
Content-Type:
application/participant_info
Content-Length: (...)
participant_info = [msa1, msa2, 3gsa1, 3gsan]

6.13-b: A MANET party subscribes event

Figure 6.13: Event subscription scenarios for integrated MANETs/3G networks

Conference termination – Figure 6.14 shows an example of conference termination. The termination is triggered by the leaving of MSA1. The CGW finds that there is only one party left in the conference, so it terminates its session with MRFC/AS. The MRFC/AS finally tears down the last session with 3GSA1 and the conference is then terminated.

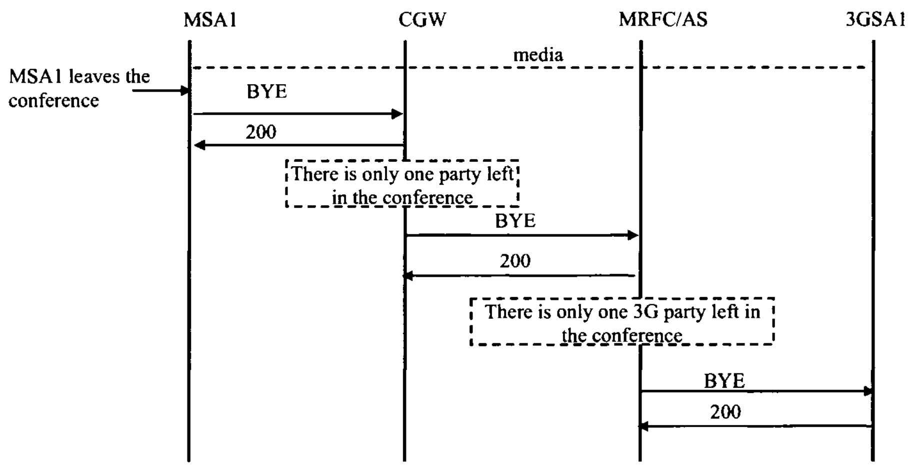

MSA1
CGW
MRFC/AS
3GSA1
media
MSA1 leaves the conference
BYE
200
There is only one party left in the conference
BYE
200
There is only one 3G party left in the conference
BYE
200

Figure 6.14: Conference termination scenario for integrated MANETs/3G

6.2.3 A Prototype for integrated MANETs/3G networks

Based on the signaling prototype for standalone MANET, we have also built a prototype for integrated MANETs/3G networks. In this section, we will first introduce the network setup. Then, we will present the prototype design and results.

6.2.3.1 The network setups

Setting up a real integrated MANET/3G environment is not easy because we do not have a 3G network environment. What we only have is the small-scaled MANET introduced in section 6.1.5.1. So we try to deploy all the signaling entities for integrated MANET/3G networks on this MANET. For example, we decided to put CGW, 3G MRFC/AS and 3GSAs into one MANET node in order to overcome the lack of a device running both

MANET interface and 3G wireless interface. Although we cannot obtain very realistic results from this network setting, it is sufficient for proof-of-concept.

6.2.3.2 Prototype design and software architecture

We have implemented all the signaling entities defined for integrated MANETs/3G networks. The MSA is developed based on the extended SIP UA for standalone MANETs. We implemented a simple application-layer CGW/MSA discovery protocol on the UA and the discovery is based on broadcast, i.e. the CGW periodically broadcasts its address and listening point. An MSA that receives this address will register with the CGW. Periodically, the MSAs update their location information by sending a new REGISTER message.

The 3GSAs and conference focus (MRFC/AS) is simplified and only signaling aspects are implemented. A 3GSA includes a SIP UA that is configured to know the conference focus, and a conference application (with graphic user interface) that can invite parties and leave a conference.

The conference focus includes a conference manager that generates a conference room for each conference. Each conference room includes a SIP UA that can handle sessions. This design allows the conference focus to handle many conferences simultaneously. The conference manager also distributes conference events to a right conference room. The conference focus is configured to know the address of CGW.

The CGW is a key entity that we implement. The software architecture of it is depicted in Figure 6.15. Functional blocks such as CGW service discovery and publication, protocol translator, architecture translator and SIP user agents are implemented. The MANET user registrar maintains the current locations of MANET participants. The conference focus interface implements a simplified message transport interface (based on TCP/IP) to the conference focus. The 3G configuration provides a user interface to configure the address of the conference focus. The CGW also has a MANET interface that is responsible for sending messages to MANET side.

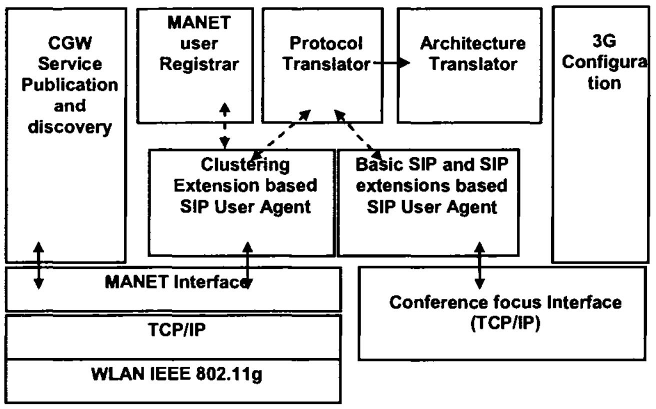

CGW Service Publication and discovery
MANET user Registrar
Protocol Translator
Architecture Translator
3G Configuration
Clustering Extension based SIP User Agent
Basic SIP and SIP extensions based SIP User Agent
MANET Interface
TCP/IP
WLAN IEEE 802.11g
Conference focus Interface (TCP/IP)

Figure 6.15: CGW software architecture

6.2.3.3 Experiment scenarios and results

We have tested the basic conference initiation, participant joining/leaving and conference termination cases using this prototype. Each case is tested with a conference including parties from both MANET and 3G network sides. It shows that the proposed architecture works well.

We have collected a result on average session establishment delay (shown in Table 6.5). The result is based on the following scenario: a 3GSA is the initiator of the conference. It first invites Node 1 to establish a conference. It then invites Node 2 to Node 6, respectively, to join the conference. It should be noted that 3GSAs, conference focus and CGW are deployed on Node 7. Comparing the result with that shown in Table 6.4, we can find that it always takes longer to setup a session between a 3G party and a MANET party. This is reasonable because any invitation request needs to traverse conference focus and CGW before it can reach the destination. In this result, we can still find that session delays are affected by the distance between the two parties that wishes to set up a session.

	Node 1	Node 2	Node 3	Node 4	Node 5*	Node 6*
Average per session delay (seconds)	1.646	1.255	1.348	1.645	1.797	2.239

* A node that does not locate in the same room as the CGW

Table 6.5: Average per session setup delay for integrated MANET/3G signaling prototype

6.3 Building a conference in MANETs

So far, we have discussed the implementations of our signaling architectures. However, signaling is not the only component of a conference. In this section, we discuss how our signaling system inter-works with other components in order to set up a real conference application in MANETs. We first introduce the composition of a conference and the interfaces related to signaling. The general implementation methods of these interfaces are discussed. We then describe the work on media handling in MANETs and show how it works with our signaling system.

6.3.1 Conference components and interfaces

In Chapter 2, we have introduced three technical components of conferencing: signaling, media handling and conference control. The boundary between signaling and conference control are somehow vague. For example, the conference membership management can be implemented in either signaling or conference control and we have considered it as a part of signaling tasks in this thesis. A common view is that conference control handles conference policy and floor control while signaling handles session establishment, modification and termination. Since not every conference requires policy and floor control, a conference may only includes signaling and media handling. The division of signaling and media handling is much clearer. Media handling deals with the media transmission, mixing and possible trans-coding. Besides of the technical components, in order to build a conference application, a user application that reacts with user's commands and triggers the technical components is always necessary.

A general view of the relationships between signaling and other conference components is shown in Figure 6.16. There are three interfaces related to signaling. The I_1 is the interface between Signaling and User Application. It is usually implemented through APIs. The APIs may be local signaling APIs (e.g. what we have implemented in our prototypes and simulations) or web APIs based on Web Service (e.g. Parlay X [114]). The web APIs are of a higher level of abstraction and with which a user does not need to understand the signaling details.

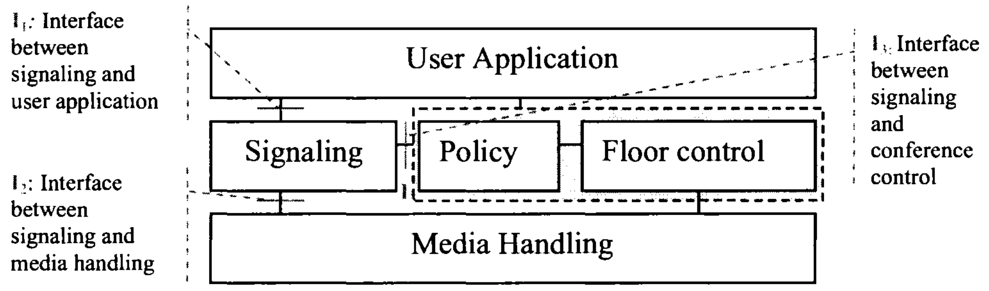

I1: Interface between signaling and user application
User Application
I3: Interface between signaling and conference control
Signaling
Policy
Floor control
I2: Interface between signaling and media handling
Media Handling

Figure 6.16: A general view of the relationships between signaling and other conference components

The I_2 is the interface between Signaling and Media Handling. It can be implemented using three methods. The first is that signaling directly calls media handling. Byrne et al. [115] show an example of the method in which SIP user agent directly calls RTP media processing functions and media stream functions. This method requires that the signaling knows details of the media handling. The second is that media handling system provides APIs so that signaling can call. An example of the method presented in [116]. The third method is that signaling and media handling communicate through primitives or protocols, such as Megaco/H.248. An example is the 3GPP conference defined in [81]. In the specification, the MRFC (signaling entity) in conference focus uses H.248 to add media streams to MRFP (mixer) after a SIP session is established. Cho et al. [117] show another example of the third approach. It defines a SIP-like protocol between signaling and media handling in order to activate and remove mixers. Still another example is shown in [118]. It uses the concept of Megaco, but it separates MGC from signaling entity and defines primitives between them.

The I_3 is the interface between conference control and signaling. This interface can also been implemented through APIs and protocols. An example of implementing this interface is shown in IETF CPCP [72] for centralized SIP conference, where an XML (Extensible Markup Language [119]) based document is described. The document contains conference policies and it is understandable by the SIP conference focus. Figure 6.17 shows a use of CPCP message between a UE and MRFC. In the figure, UE1 informs MRFC to add a new user to the conference. In the example, the CPCP XML document is created by XML Configuration Access Protocol (XCAP [120]), a protocol that allows a client to read, write and modify application configuration data, stored in XML format and directly accessed by HTTP.

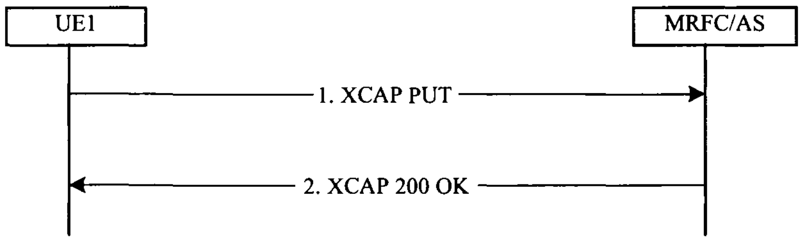


Figure 6.17: Policy control using CPCP

The implementation of interfaces between signaling and other components in MANETs may not go beyond the methods that we introduced. However, we wish to specify the implementation of our signaling system with media handling and conference control proposals in MANETs. Unfortunately, so far we cannot find related work for conference control in MANETs, so we will not further investigate this. The work on media handling in MANETs is slowly emerging. In the next sub-section, it will be introduced and we also will detail how it communicates with signaling.

6.3.2 Media handling in MANETs and inter-working with signaling

The work in [116] presents a distributed media mixing architecture for MANETs. In the architecture, the media handling nodes are divided into two levels. The first level is composed of nodes with mixing capabilities (called mixers) and these mixers form a fully-meshed network, i.e. every mixer has a media connection with every other mixer in the network. The second level is composed of nodes without mixing capabilities (called inactive peers) and each inactive peer in this level connects to one and only one of the mixers in the first level. The overview of the architecture is shown in Figure 6.18. This architecture is self-organized and mixers are designated and removed during conferencing.

A media core is defined and located in each node. It is responsible for controlling mixers. It is also responsible for communicating with signaling system. It hides the complexities of media handling and provides standard APIs, such as “open media connection”, “close media connection”, “start mixer” and “stop mixer”. In this API-based method, the mixers are decided by signaling. If we integrate the cluster-based signaling system (our proposal) with the media handling system, the signaling super-member can be designated as mixers. This facilitates the formation of the media handling architecture. On the other hand, signaling should have the knowledge of whether a node has a mixer capability or not. It should also add a condition to the super-member election schemes, i.e. if a node does not have a mixer capability, it will not be selected as a super-member. This introduces dependency between signaling and media handling architectures, which we do not wish to see.

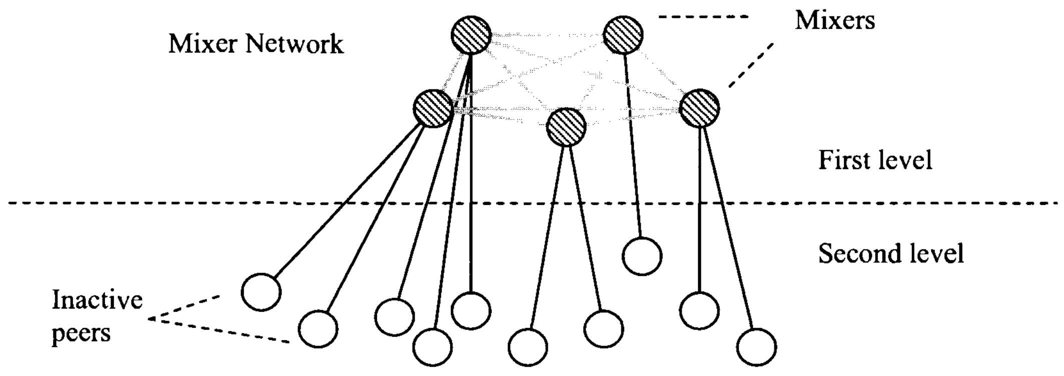


Figure 6.18: Two-level media handling architecture

Kheder et al. [118] defines a media mixer control method that improves the work defined in [116]. The method applies the concept of Megaco and it adds a controller overlay on top of the mixer network. The controller overlay consists of MGCs and the mixers are MGs. It assumes that only a set of nodes support MGC functionality and these nodes are responsible for adding and removing MGs. The work also defines that signaling entities and MGCs are totally separated and they communicate with each other through primitives. After a session is established, a signaling entity will discover a MGC and then use primitives such as *connection_request, modify_request* and *disconnection_request* to connect, modify and disconnect MGCs. In this work, signaling system does not need to decide or to know where is the mixers. If we integrate our signaling system with this architecture, we can find that there is no more dependency between the two architectures, i.e. a signaling super-member may not be a mixer of the media handling. The signaling system only needs to add a function – MGC discovery. Service discovery in MANETs is not a new task. In the section of CGW discovery in Chapter 5, we have introduced some methods such as broadcast and usage of middleware, for service discovery. Similarly, the MGC discovery can also be implemented using these approaches.

6.4 Conclusions

In this chapter, we have proposed a SIP extension for implementing the cluster-based signaling architecture for standalone MANETs. Based on the extension, we also addressed implementation techniques for signaling in integrated MANETs/3G networks. Proof-of-concept prototypes are built on a small-scale, IEEE 802.11-based ad hoc network. We have tested the basic conference initiation, participant joining/leaving, termination and membership subscription cases using the prototypes. It shows that the proposed architectures work well in each of these cases. In this chapter, we have also discussed how to build a conference application in MANETs, in which the implementation of the interfaces between signaling and other conference components are investigated.

Chapter 7 : Performance Evaluation Using OPNET

We evaluate the performance of the signaling architectures for both standalone MANETs and integrated MANETs/3G networks through simulation studies using OPNET. In this chapter, we will present simulation setups, performance metrics, simulation scenarios and result analysis. We will first present the evaluation for signaling in standalone MANETs and then we will demonstrate a measurement for signaling in integrate MANETs /3G networks.

7.1 Performance evaluation for signaling in standalone MANETs

7.1.1 OPNET and settings

We set up the simulations in OPNET V.11.5.A, which provides a comprehensive development environment supporting the modeling of communication networks and distributed systems. It contains many standard models including MANETs. It supports four MANET routing protocols: AODV [39], DSR [42], OLSR [40] and Temporally Order Routing Algorithm (TORA) [113]. A user can configure to use one of these protocols. The IP address of each node can be manually configured or auto-assigned. The link layer transmission parameters such as transmission range can also be set by users. The OPNET standard application model provides a set of client/server based applications, such as email, FTP and SIP voice call. There is no existing model for peer-to-peer based applications. OPNET provides an environment for user development. The environment is based on Finite State Machine (FSM). The programming language is Proto-C – a combination of C, C++ and OPNET Event Simulation APIs.

In this simulation, we have set a simulation area of 1 km by 1 km. We use the node model IEEE 802.11 WLAN mobile workstation and the link transmission distance is defined as

500 meters. We use AODV as the default routing protocol and the IP address is auto-assigned. Different conference scales are evaluated, e.g. small-scale conferences with 3, 7 and 15 nodes, medium-scale conferences with 25, 40 and 50 nodes, and large-scale conferences with 75 and 100 nodes. These nodes are randomly distributed in the simulation area and we define the node mobility model as a random waypoint.

7.1.2 System design and software architecture

Figure 7.1 shows an overall view of simulation setup. It also demonstrates the modular structure of a MANET node. The system provides standard lower-layer modules such as WLAN wireless transmitter and receiver, ARP, IP and TCP. We build our application module: *Conf_Application* directly on top of the TCP module.

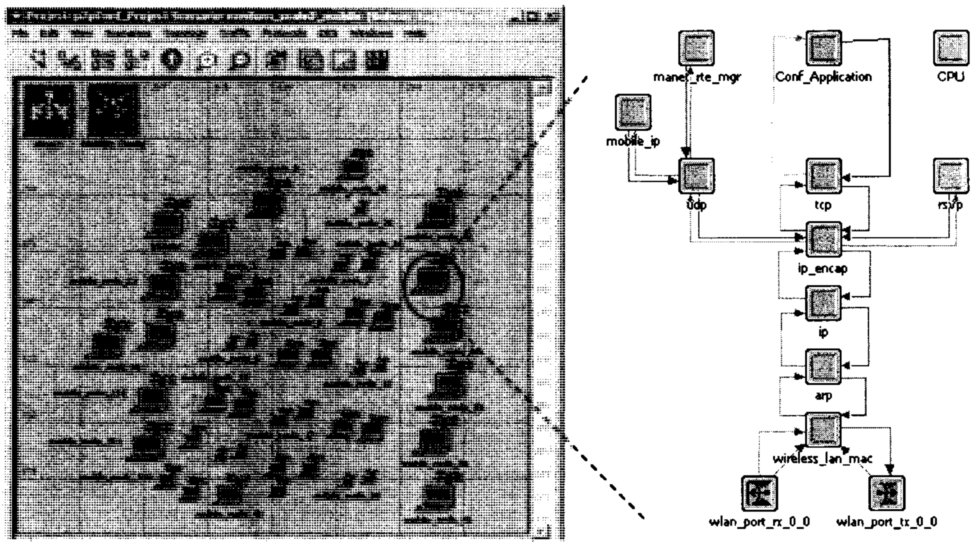


Figure 7.1: Simulation setup and MANET node structure

The software architecture of the *Conf_Application* module is shown in Figure 7.2. There are two layers in this architecture. The application logic defines simulation scenarios. It uses the SIP services provided by the signaling. It should be noted that it is impossible to build the same dynamic ad hoc conference scenario in OPNET as what we have built in prototypes. This is because whenever an OPNET simulation starts, we cannot input any events. Thus, if we need to change a scenario, a different logic should be developed.

The signaling contains an extended SIP UA, a SIP Registrar and a SIP Proxy Server. The Extended SIP UA consists of a UAC and a UAS. It generates and processes the SIP requests and responses. It also maintains the conference, dialog and cluster states. In order to monitor the status of a session, a UA calls a *heartbeat* process and the state diagram of *heartbeat* is shown in Figure 7.3. This process is responsible for sending session heartbeats and reporting session errors.

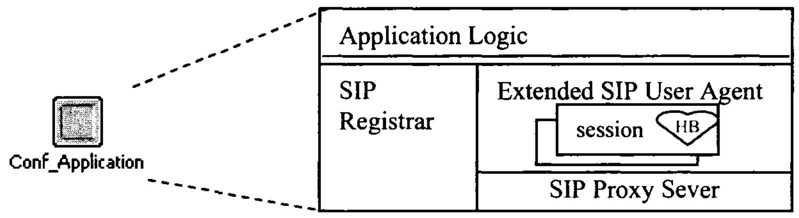


Figure 7.2: Software architecture of module Conf_Application

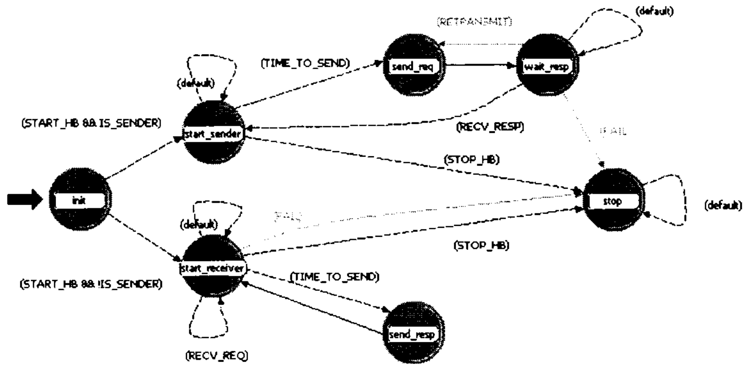
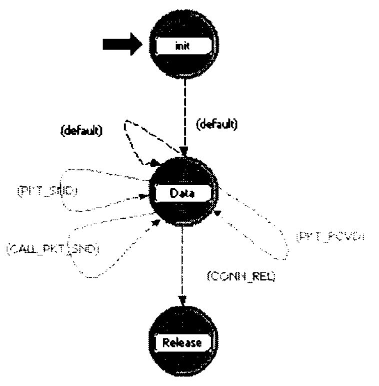
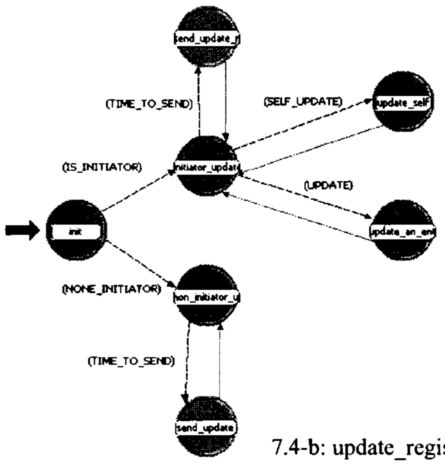


Figure 7.3: State diagram: heartbeat

The Proxy Server is responsible for sending SIP messages to the right destination. It contacts the Registrar in order to obtain the locations of destinations. It then communicates with the TCP module in order to send messages. This is done by a *tcp_agent* process. The state diagram of the process is shown in Figure 7.4-a. The SIP Registrar stores the updated location information of participants. In this simulation, we define that the initiator of a conference plays the role of a real registrar. Every other participant maintains a copy of this registrar. The location information of each participant should be updated periodically. The task is performed by the process *update_registrar* and the state diagram is shown in Figure 7.4-b. The SIP Registrar invokes this process whenever the first run of registration has been finished.



7.4-a: tcp_agent



7.4-b: update_registrar

Figure 7.4: State diagrams: tcp_agent and update_registrar

Different from the prototype, we cannot use existing tools such as JAIN SIP stack in OPNET simulation. We have to re-define SIP messages. Messages are transmitted by packet in OPNET and OPNET provides a tool to build packet. Figure 7.5 shows the SIP packet that we built. The packet is simple and it contains three fields: Msg_name, Msg_type and Msg_info. When generating a SIP message, all these fields should be set. An example shown in Figure 7.5 is the setting of a SIP INVITE message. We can see that the real header fields of the INVITE message are pointed by the *Invite_info*.

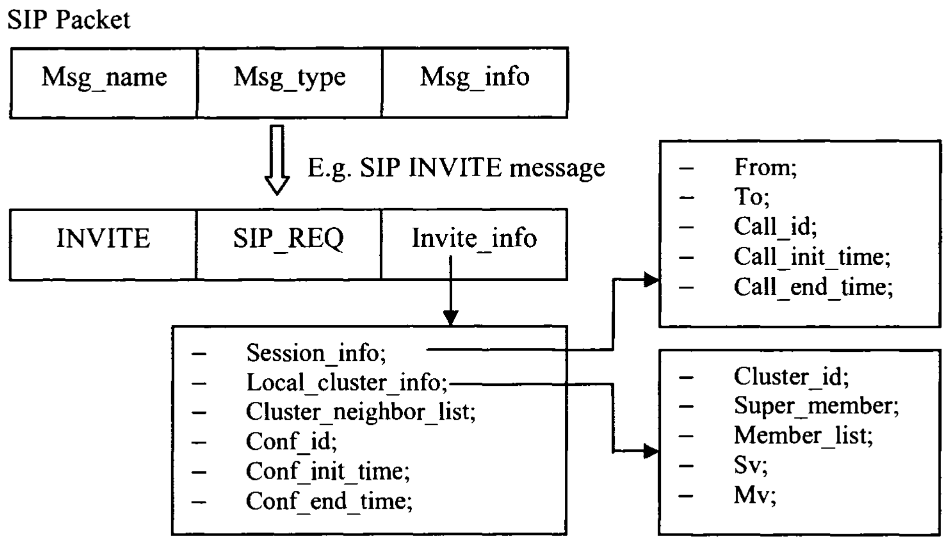


Figure 7.5: Design of SIP Packet in OPNET

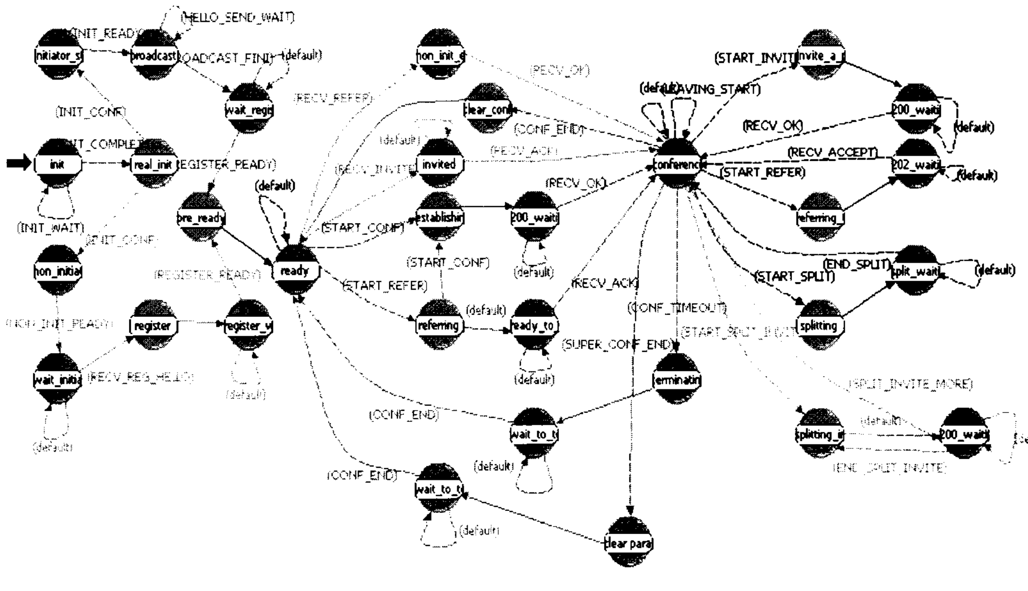
7.1.3 Simulation scenarios

In order to evaluate the performances of different aspects of the signaling system, we have implemented three conference scenarios. The first is the conference setup and termination with 'member-leaving-first'. The second is the conference setup and termination with 'super-member-leaving-first'. The last scenario is the conference setup with node/link failures.

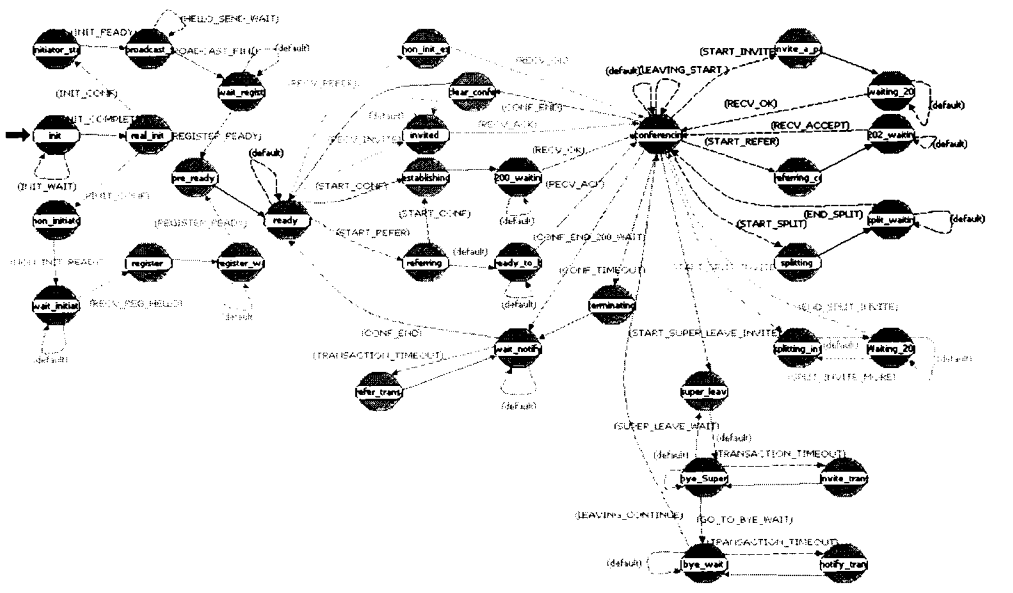
The conference setup procedure is defined as follows: A conference is always started by an initiator. The first elected super-member is responsible for inviting participants to join the conference. When a cluster reaches its split value or the super-member of the cluster reaches its processing limit, a new super-member is elected and it will continue to invite participants. The procedure continues until every participant has added into the conference. The conference then persists for a period of time before the termination procedure starts. The member-leaving-first termination means that the cluster members have a higher priority to leave a conference. A super-member leaves only when its cluster members have left. The state diagram of this scenario is shown in Figure 7.6-a.

In the second scenario, the same conference setup procedure is performed. The super-member-leaving-first termination means that the super-members have a higher priority to leave a conference. In order to avoid coincident super-member leaving, we define that the super-member with a higher address leaves first. The state diagram of this scenario is shown in Figure 7.6-b.

In the last scenario, node failure/recovery is simulated using the interrupt mechanism defined by OPNET. Since OPNET does not provide a link failure/recovery configuration for MANETs, we manually inject such failure/recovery by defining appropriate mobile trajectories for nodes. The state diagram of this scenario is shown in Figure 7.6-c.



7.6-a: Scenario of conference setup and termination with member-leaving-first


7.6-b: Scenario of conference setup and termination with super-member-leaving-first

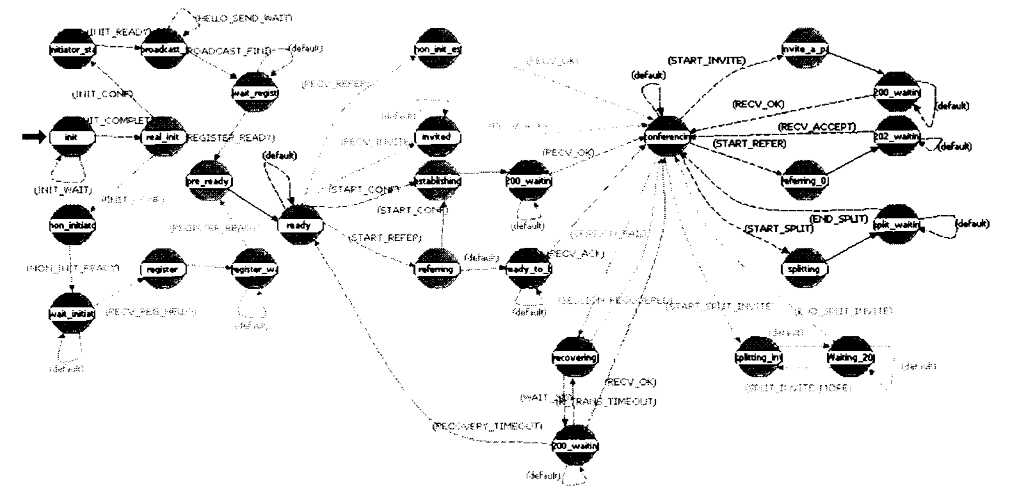

7.6-c: Scenario of conference setup with node/link failures

Figure 7.6: State diagrams for different conference scenarios

7.1.4 Measurements and analysis

In this section, we will present the performance results collected from the simulations. The performance metrics are presented first, followed by four set of results.

7.1.4.1 Metrics

We use the following metrics to measure and evaluate our signaling system:

- **Overhead:** the amount of signaling messages transmitted. The amounts are in bytes. The types of signaling overheads measured include participant registration and capability exchange, session establishment, information propagation and session termination.
- **Delay:** the delays are calculated in seconds. We measured the following five types of delays: The conference setup delay is the delay from the establishment of the first session to the joining of the last participant. The average session setup delay is an average value for the delay of every session during the conference establishment procedure. The conference termination delay is the delay from the first party departure to the termination of the last session in the conference. The session failure Detection deLay (DL) is the delay from the last successful heartbeat message received to a failure detected. The Recovery deLay (RL) is the delay from a failed session detected to a recovered session established.
- **Load:** the ratio of the number of bits transmitted per second. There are two types of loads measured. The signaling load presents the signaling traffic introduced to the network, which includes the session setup load, information propagation load and the registration/capability exchange load. The network load represents the total network traffic per second.

7.1.4.2 Session setup and termination

The first set of experiments is to evaluate the performance of basic conference setup and termination. The cluster split value is set to 10. Figure 7.7 depicts three types of signaling loads with different conference scales. For each conference scale, the simulation time is 20 minutes and the result is an average of three runs with different seeds. In the figure, the signaling load increases with the growth of the conference scale. The one with the most significant increment is the information propagation load, which is less than the session setup load when the conference scale is small but it increases to more than twice the setup load when the conference scale becomes 100 nodes. This is due to the mesh structure of the super-members. With a split value of 10, there are more than ten clusters in a 100-node conference. Whenever a new member joins a cluster, the super-member of that cluster needs to propagate the information to all the neighboring super-members. This significantly increases the network load. There is no way to avoid this traffic in our cluster architecture, but a possible way to reduce the traffic is to redesign the propagation message and make it lighter.

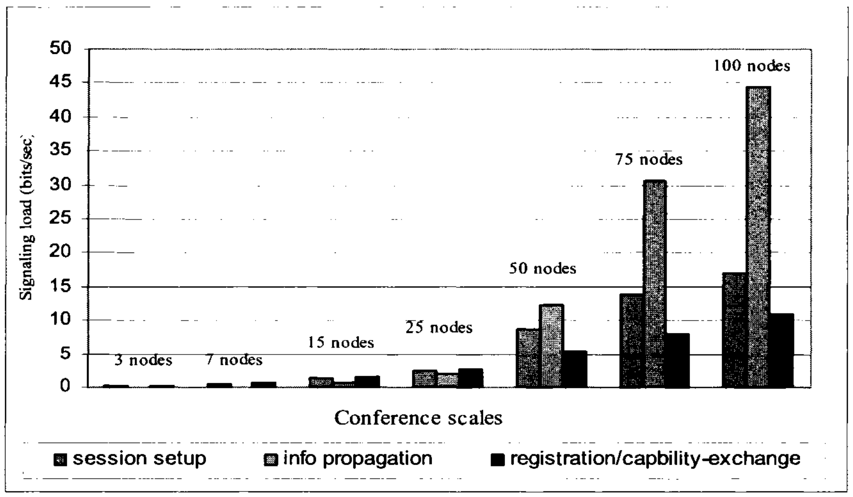


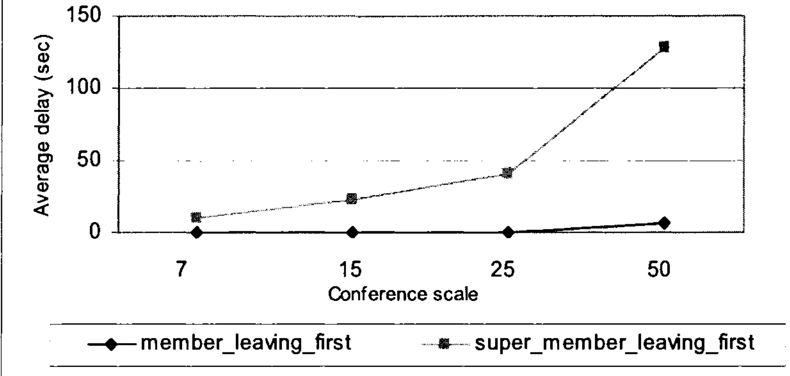
Figure 7.7: Signaling load introduced by conference initiation

Table 7.1 shows the results of setup delays from the same experiments. The values are more experimental but less realistic compared to the values that we obtained from the prototype, where the average session setup delay for the 7-node conference was about 0.5 second. However, the trend that a larger-scaled conference introduces a larger setup delay helps us to predict the delay in real larger-scaled conferences. For example, the average per session setup delay increases about 20 times from a 7-node conference to a 100-node conference. If this were absolutely the case in reality, then a session setup in a large-scale conference with 100 nodes would be 10 seconds. This is an acceptable delay for SIP in a wireless system. Fathi et al. [121] claim that a SIP session up delay can be up to 7 seconds with TCP in 3G wireless network.

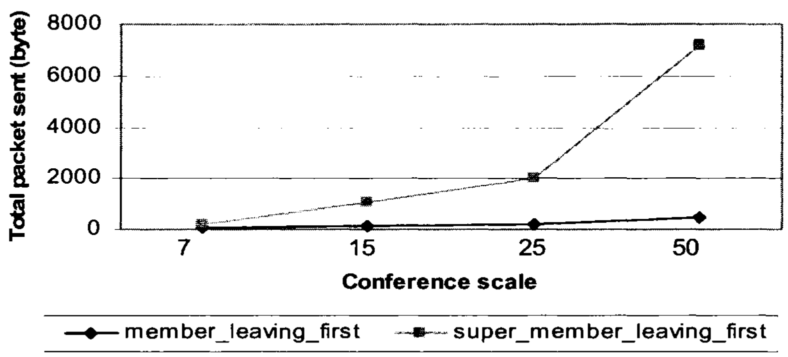
Conf-scale/ Delay (sec)	3 nodes	7 nodes	15 nodes	25 nodes	50 nodes	75 nodes	100 nodes
Conference setup delay	0.014	0.021	0.079	0.426	1.268	4.887	9.027
Average per session delay	0.002	0.003	0.005	0.017	0.016	0.038	0.062

Table 7.1: Conference setup delay and Average per session setup delay

Figure 7.8 illustrates comparative results for the two conference termination scenarios: member-leaving-first and super-member-leaving-first, in terms of termination delay and overhead. Actually, the two scenarios represent the best and the worst cases of conference termination. A super-member departure introduces a much higher overhead than a member departure, because when a super-member leaves a cluster must be re-formed. The 'cost' of the reformation is very expensive, and includes the election of a new super-member, session establishments between the new super-member and members and the session terminations between the leaving super-member and members. In the figure, we find that with conference participant growth, the gap between the two extreme cases becomes even bigger. For a 50-node conference, the super-member-leaving-first scenario costs 15 times the overhead of the member-leaving-first, and it introduces 18 times the delay of the case of member-leaving-first.



7.8-a: Delay of conference teremination



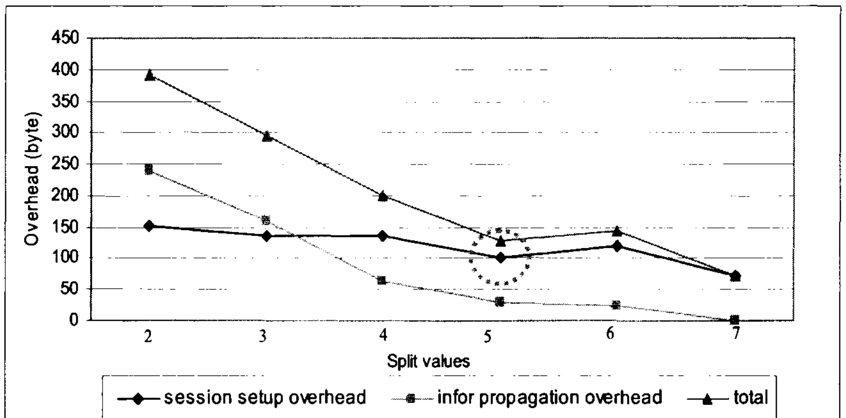
7.8-b: Overhead of conference termination

Figure 7.8: Performance results for conference termination

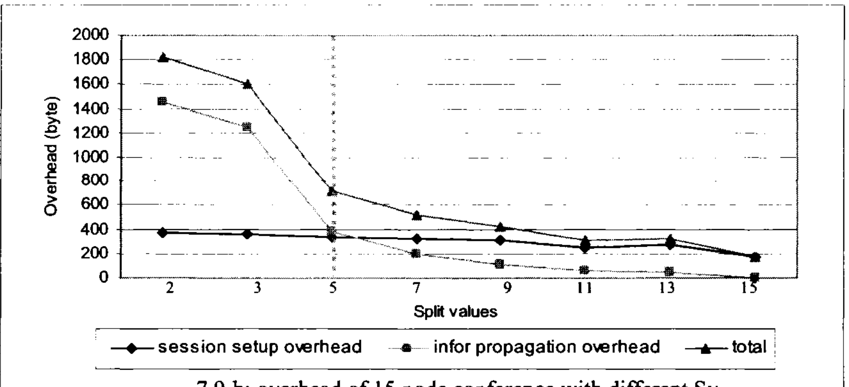
7.1.4.3 Signaling performance and split values

The split value is a very important parameter that directly affects the signaling performance. The second set of experiments is to find out if there is an optimal split value for the whole system, i.e. a minimum signaling overhead with a minimum session delay and the system architecture does not become centralized. We have tested four different scales of conferences: 7, 15, 25 and 50 nodes. In each conference scale we selected a set of split values. For example, we select *Sv* = {2, 3, 5, 7, 9, 11, 13, 15} for the 15-node conference.

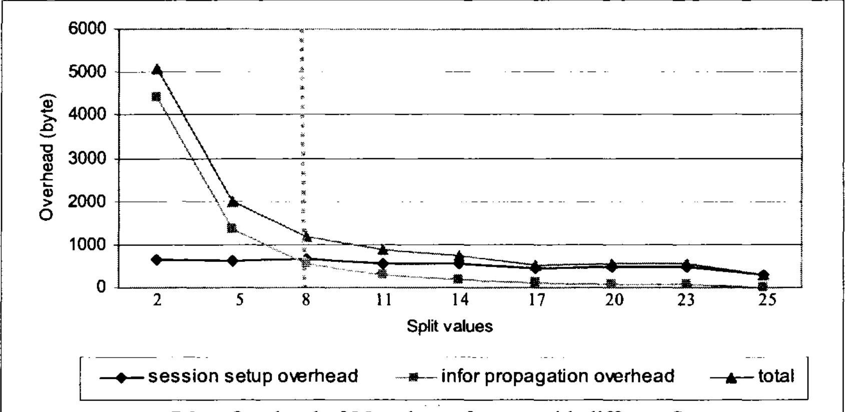
Figure 7.9 depicts the overhead of the session setup and information propagation. In general, the larger the *Sv* is, the smaller the signaling overhead will be. In each conference scale, there is an *Sv* where the session setup overhead and the information propagation overhead are the closest. The *Sv*s are 3, 5, 8 and 11 for the 7, 15, 25 and 50-node conferences, respectively. If we set the split value equal to or larger than these values, the signaling system will not introduce too much overhead. This value increases with the growth of the conference scale.



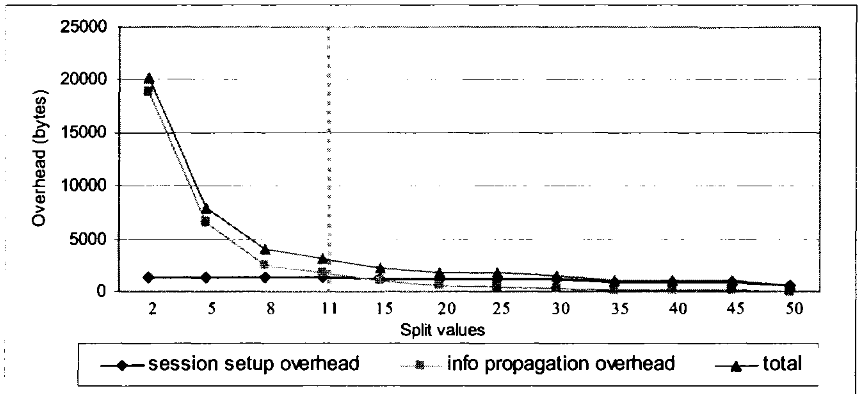
7.9-a: Overhead of 7 node conference with different Sv



7.9-b: overhead of 15 node conference with different Sv



7.9-c: Overhead of 25 node conference with different Sv

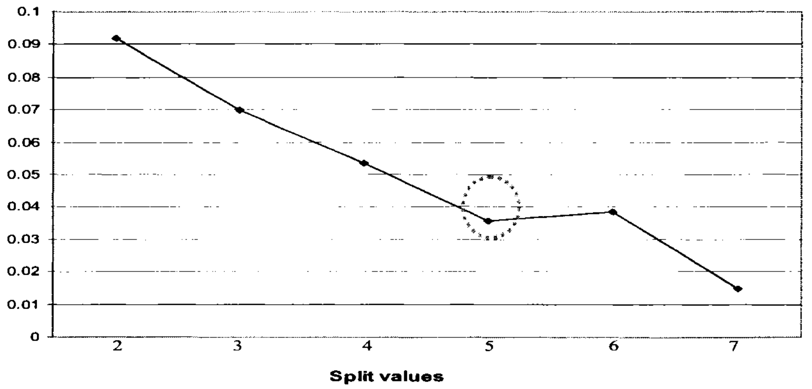


7.9-d: Overhead of 50 node conference with different Sv

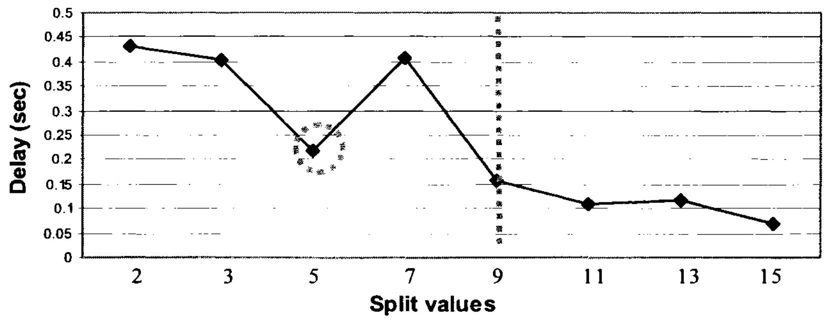
Figure 7.9: Conference setup overhead with different split values

Figure 7.10 shows the conference setup delays. The general trend is still the larger the split value, the smaller the delay. However, the delays do not strictly follow the trend. For example, with an *Sv* = 7, it takes longer to set up a 15-node conference than that does with *Sv* = 5. As another example, for a 50-node conference, the setup delays fluctuate with the split values set from 11 to 40.

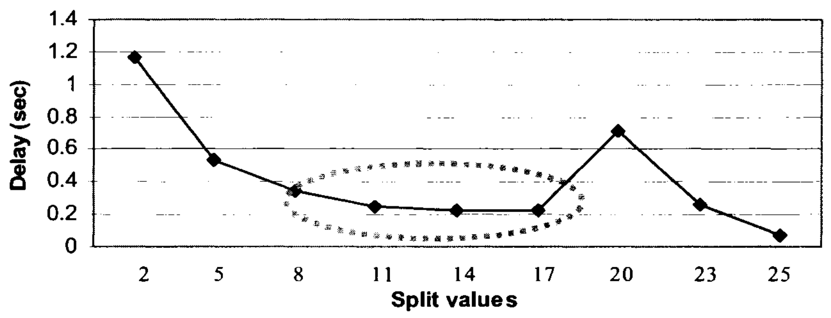
From the experiment, we can find that there is no split value that makes the system performance optimal. Trade offs must be made between system overhead, delay and the number of clusters, and different scales of conferences should be analyzed case by case.



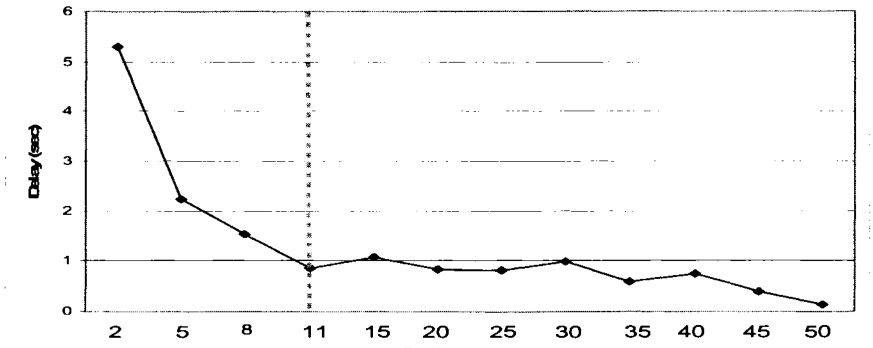
7.10-a: Delay of 7 nodes conference with different Sv



7.10-b: Delay of 15 nodes conference with different Sv



7.10-c: Delay of 25 node conference with different Sv



7.10-d: Delay of 50 node conference with different Sv

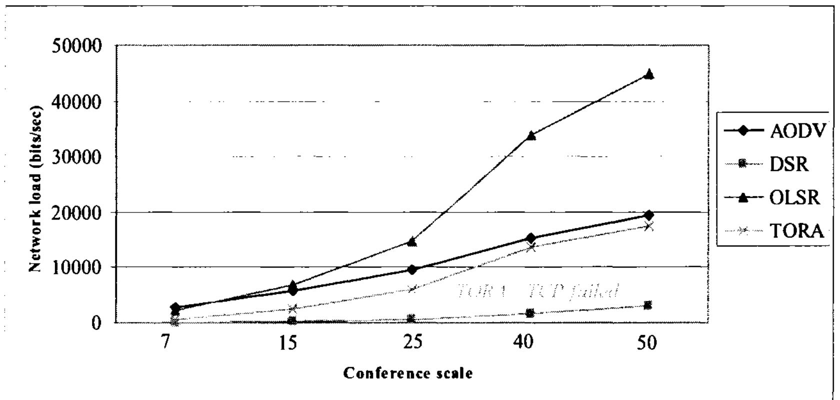
Figure 7.10: Conference setup delay with different split values

7.1.4.4 Running on top of different routing protocols

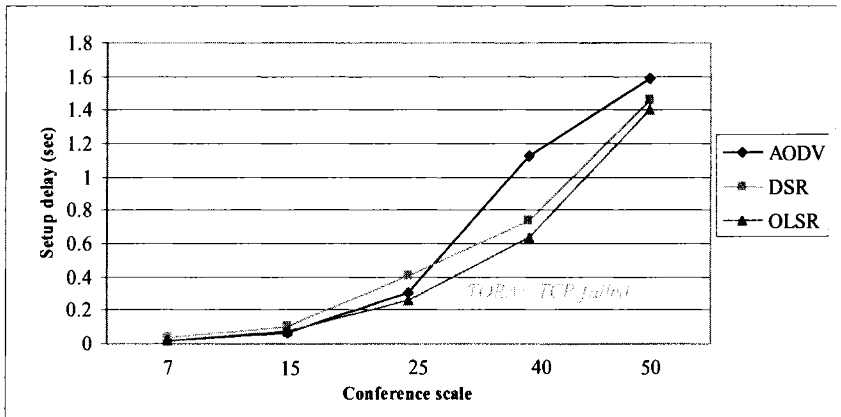
The third set of experiments is to test the performance of the signaling system running on different routing protocols. This helps us to analyze how the routing layer settings affect the performance of the application layer. Four routing protocols are evaluated: AODV, DSR, OLSR and TORA. The former two are reactive routing protocols, i.e. forming a route on-demand. OLSR is a proactive protocol and it needs to continuously calculate routes. TORA is a mix of reactive and proactive. We have measured the network load with conference setups and the setup delays. The experiments are still done with different conference scales: 7, 15, 25, 40 and 50 nodes.

Figure 7.11 shows the results. We found that the signaling procedures cannot complete if we deploy TORA as the routing protocol. It always generates too many TCP transmission errors. This is probably because TORA does not maintain routes between every source/destination pair at all times and it is more suitable for relatively sparse traffic patterns. In our case the signaling traffic is very dense during the conference setup and termination phases.

The signaling works well with the other three protocols. In terms of network load, the signaling system performs best with DSR and it performs worst with OLSR. The AODV is in the middle. With respect to the setup delay, the signaling performs best with OLSR and it performs worst with the DSR when the conference scales are set to 7, 15 and 25 nodes. It performs worst with AODV when the scales become larger. It can be found that the differences among the setup delays are not very significant compared to the performance differences among network load. Thus, we can conclude that the reactive routing protocols (DSR and AODV) perform better under our simulation environment.



7.11-a: Network load



7.11-b: Setup delay

Figure 7.11: Conference setup on top of different routing protocols

7.1.4.5 Session failure detection and recovery

The last set of experiments is to evaluate the performance of the session failure detection and recovery mechanism. We set the heartbeat timer Th = 60 sec (Th should be larger than n*Tt), the transmission timer Tt = 10 sec (Tt should be larger than Round Trip Time (RTT) of SIP message), the recovery timer Tr = 180 sec (Tr should be larger than Th and the value we set is identical to the SIP proxy transaction timeout value), and the retransmission time n is 3. Session failures caused by different cases were tested. We also considered different conference scales. The results are shown in Figure 7.12, Table 7.2 and Table 7.3.

In Figure 7.12, we evaluate the network load under a 25-node conference environment. The first two columns demonstrate the network load introduced by the signaling system with and without using the heartbeat mechanism. It is clear that the heartbeat mechanism introduces a 4.25% increment of network load. The other six columns show the cases of cluster member failure, member permanent moving out-of-range (or departure), member temporal moving out-of-range, super-member failure, super-member permanent moving out-of-range and super-member temporal moving out-of-range. For the cases of node failure and permanent moving out-of-range, the failed session cannot be recovered but it can be detected and removed from the conference. The figure shows that a super-member failure or departure always introduces more network load than a member failure or departure. For example, the network load increases 2.06% in the case of a temporal member departure while it increases 8.21% in the case of a temporal super-member departure. This is because a super-member departure always causes a set of participants to lose session connections. It is costly to recover all these sessions. It should be noted that the cases of node failure always introduce less network load than for node departures. The reason is that a failed node does not send or receive any messages, i.e. no network load is introduced by it. On the contrary, a departed node generates traffic to the network. The failure detection overheads of the two cases are the same.

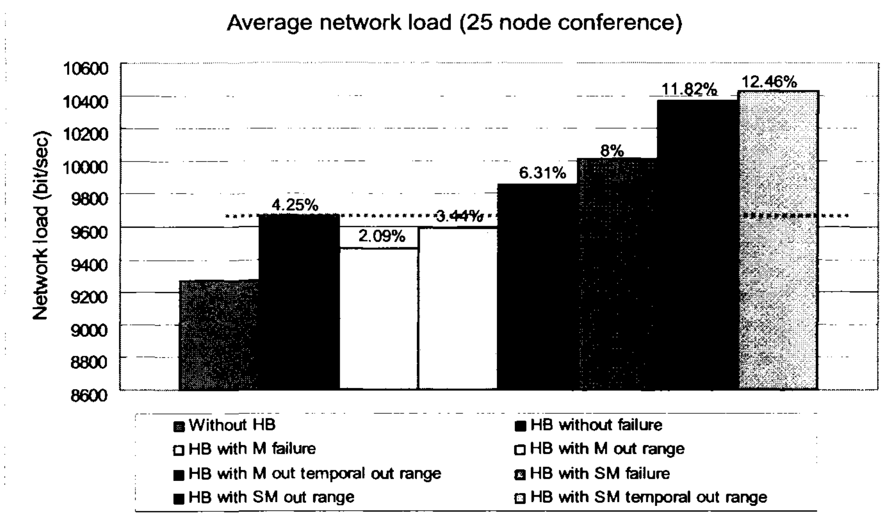


Figure 7.12: Network load with different types of party failures

Table 7.2 shows the result for session failure detection delay – DL. The cases of session failures (i.e. node failure and node moving out-of-range) are tested. The 'M' in the table denotes the member and 'SM' the super-member. With the timeout parameters that we set, the ideal DL value for a super-member is 30 seconds (n*Tt) and the ideal value for a member is 90 seconds (Th+n*Tt). In the table, it can be found that the DL values for all of the cases are close to the ideal values. Thus, the detection delays are not much affected by the different failure cases or by different conference scales.

Table 7.3 shows the result for session recovery delay – RL. In addition to the session re-establishment procedures, the recovery delay is also affected by the time that a node stays physically unconnected. In this experiment, we set the time to 5 seconds. In the table, the first line of results shows member's average RL values when a super-member fails. For a 3-node conference, since there is only one cluster, a super-member failure leads to the complete conference failure. There is no RL value. For the other conference scales, the members can reconnect to other super-members and the RL is about 30 seconds. The second and third lines of results show the cases of temporal member departure and temporal super-member departure. It can be found that with a similar session failure, the larger the conference scale, the lower the average RL. If we define that an error ratio is the number of error nodes divided by the total number of nodes, we can conclude that the session's RL value increases with the growth of the error ratio.

Conference scales/ DL (second)	3 nodes	7 nodes	15 nodes	25 nodes	50 nodes
M failure (SM DL)	29.98	29.96	29.96	29.87	29.41
SM failure (SM DL)	----	29.98	29.998	29.95	29.75
SM failure (M DL)	90	90	90	90	90
M out range (SM DL)	30	29.98	30	29.89	30.03
M out range (M DL)	89.99	90	90	90	90
SM out range(SM DL)	29.996	29.997	30.8	30.24	33.24
SM out range (M DL)	89.998	89.99	88.42	89.39	89.39

Table 7.2: Average detection delay with different types of session failures

Conference scales/ RL (second)	3 nodes	7 nodes	15 nodes	25 nodes	50 nodes
SM failure	---	30	30.02	30	30.06
M out range	98.46	73.47	73.3	60.33	46.99
SM out range	109.63	79.29	55.33	36.48	29.41

Table 7.3: Average recovery delay with different types of session failures

7.2 Performance evaluation for signaling in integrated MANETs/3G Networks

7.2.1 Simulation settings

OPNET provides a standard UMTS 3G feature. We build a one-cell UMTS System using the existing node models in the feature. The system includes a Node-b (the WCDMA base transceiver station), a RNC (Radio Network Controller), an SGSN (Serving GPRS Support Point) and a GGSN (Gateway GPRS Support Point). An IEEE 802.11 based MANET is deployed at the edge of the cell and we configured it to run the AODV routing protocol. We simulated a gateway node that can forward a packet from MANET side to 3G destinations and vice versa. The gateway node keeps the routing topology of the MANET nodes and it has a wireless connection with the Node-b as well. A conference server is connected to the UMTS core network through a hub. A simplified version of conference focus has been implemented in the server. A set of 3G UEs have been deployed randomly in the cell and every UE runs a 3GSA that can establish or join a conference. Every MANET node runs an MSA loaded with CGW discovery functionality. The CGW can be optionally deployed on the gateway node, in 3G core network or in MANET. The conference server and the CGW are configured to 'know' each other. Figure 7.13 shows a general view of the simulation, in which the modular structure of a 3GSA is illustrated. The module that we develop in the structure is the *simple_conf_application.*

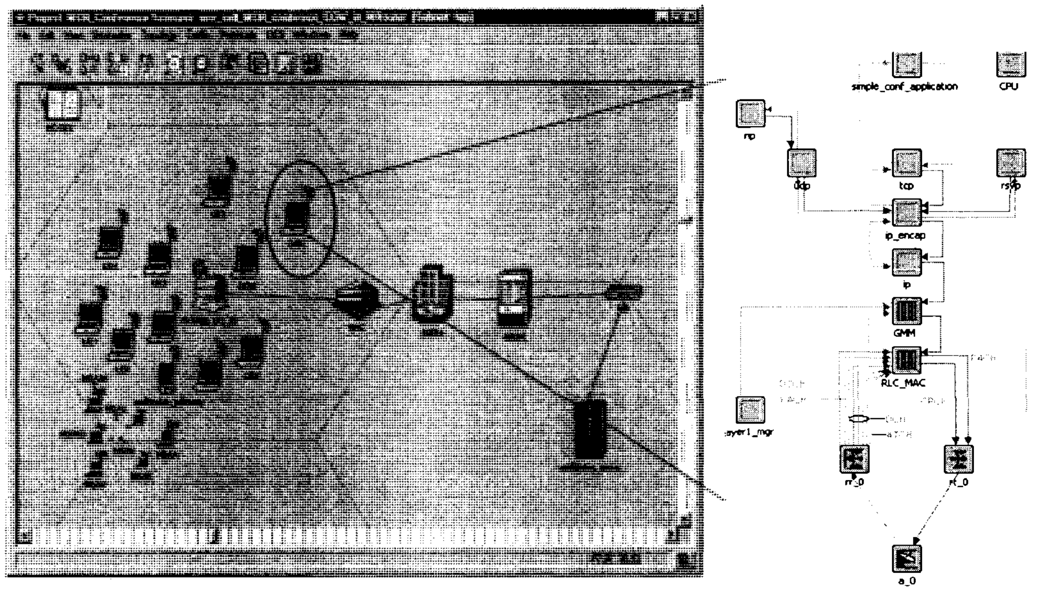
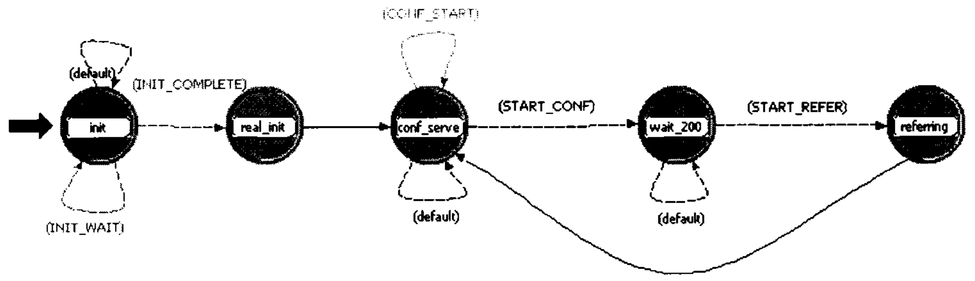
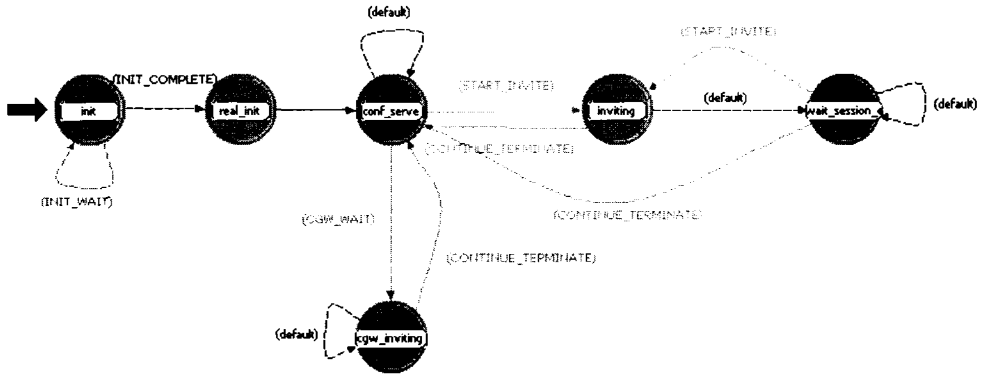


Figure 7.13: Simulation setup and 3GSA node structure

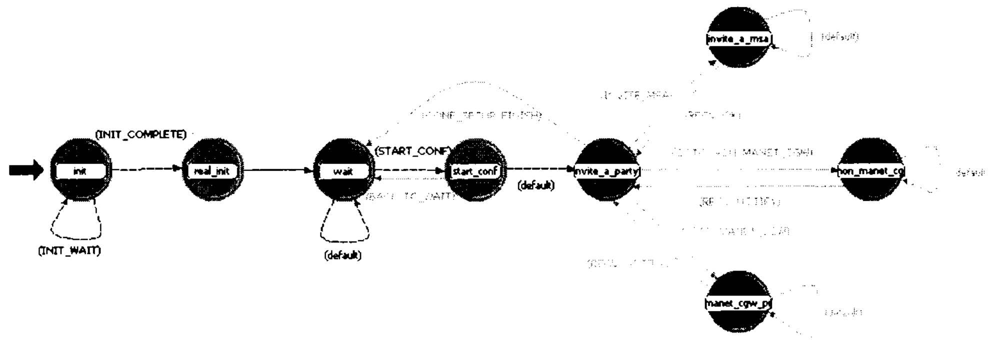
As above mentioned, we developed four signaling entities: 3GSA, MSA, CGW and conference focus. The functionalities of the 3GSA, the conference focus and the CGW are similar to those described in prototype, but the difference is that the SIP message exchanged among these entities are SIP packets defined in Figure 7.5. It should be noted that the SIP messages exchanged in 3G side do not include the cluster-related header fields. Message translations are performed in CGW. Figure 7.14 demonstrates the state diagrams of the four entities 3GSA, conference focus, MSA and CGW respectively. For simplicity, the current version that we developed only supports conference establishment and participant joining. These scenarios have allowed us to evaluate the most important aspects of the signaling system for integrated MAENT/3G networks.



7.14-a: 3GSA state diagram



7.14-b: Conference focus state diagram



7.14-c: MSA state diagram

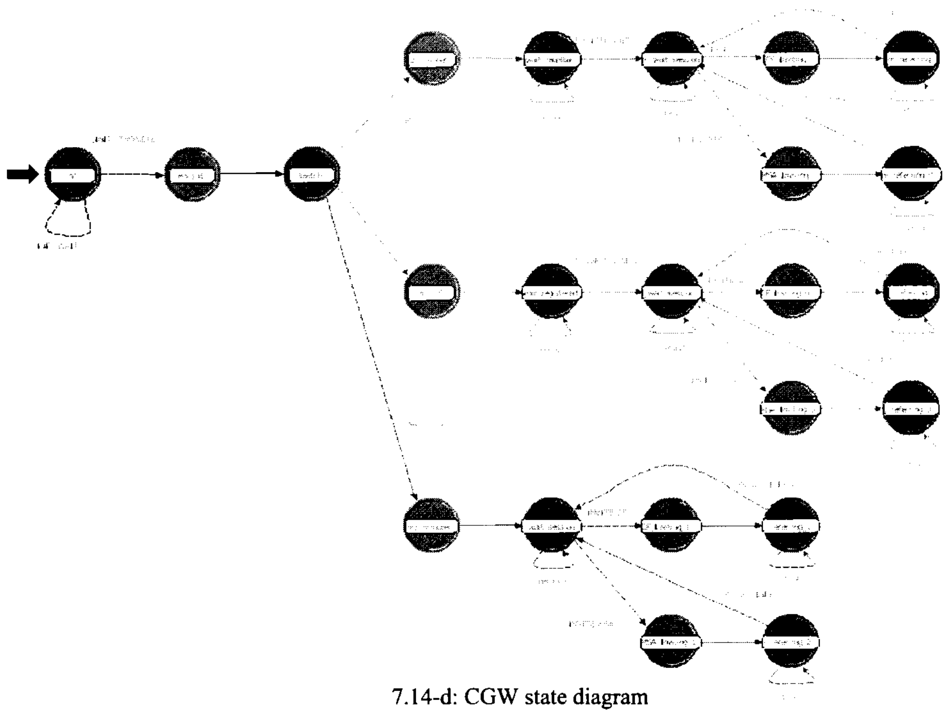
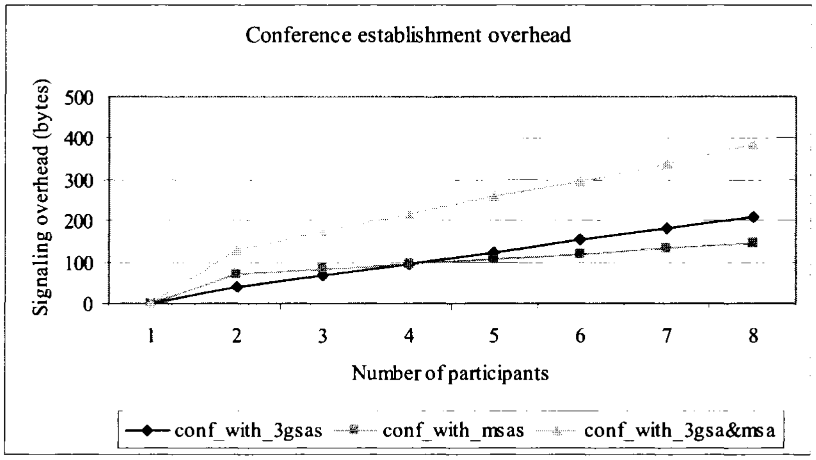

7.14-d: CGW state diagram

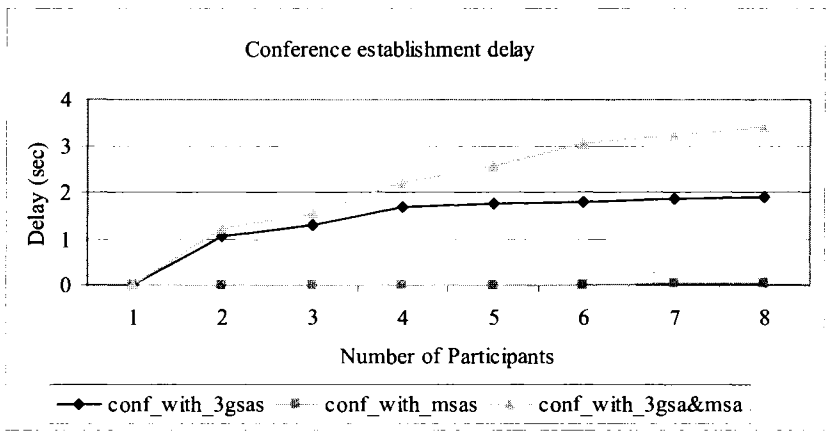
Figure 7.14: State diagrams for integrated MANETs/3G signaling entities

7.2.2 Measurements and analysis

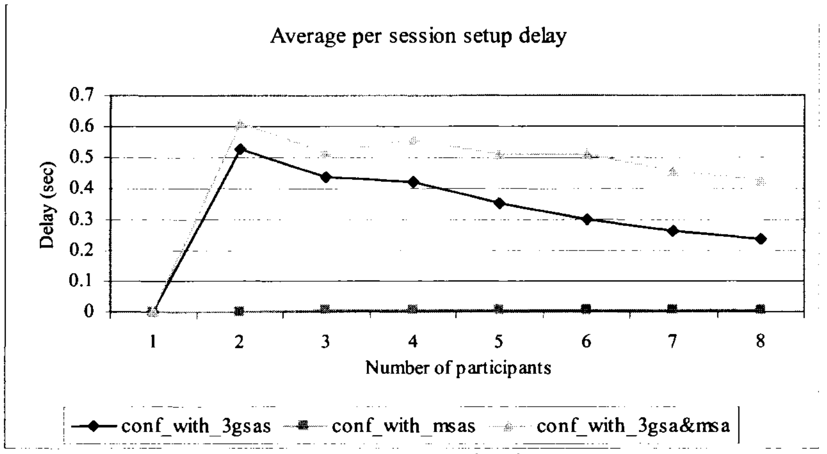
We have done two set of experiments. In the first experiments, we compare the conference setup overheads and delays among three types of conferences: conferences with only 3G parties, with only MANET parties and with both 3G and MANET parties. The overheads are measured in bytes and they are signaling overheads including session setup and possible CGW discovery overheads. Figure 7.15 shows the result. From the figure, we can find that a conference with both 3G parties and MANET parties always takes longer to establish and it introduce most signaling overhead. The overhead difference between a conference set up with only 3G parties and that set up with only MANET parties is not significant, but the conference setup delays for both cases are very different. A conference with only MANET parties takes much less time to establish. This is reasonable because MANET parties can establish sessions locally, while 3G participants always need to connect to the conference focus. The delay of a signaling message may accumulate while it traverses the 3G wireless network and each node of the 3G core network. Furthermore, the conference focus itself may be a bottle neck for processing signaling messages.



7.15-a: Signaling overhead



7.15-b: Conference setup delay



7.15-c: Average per session delay

Figure 7.15 Performance results for different conference setup scenarios in integrated MANETs/3G networks

In the second set of experiments, we deploy the CGW on different network nodes (i.e. on a MANET node, on a 3G core network node and on a 3G wireless node) and compare the conference setup delays. Two scenarios are investigated: an MSA invites a 3GSA and a 3GSA invites an MSA. Table 7.4 shows the results. We can find that the signaling system always takes longer to set up a conference when the CGW is located in 3G wireless network. It should be noted that there is no difference between signaling procedures when deploying a CGW in different location, but we observed that the delay is caused by the lower-layer TCP re-transmissions between CGW and conference focus. This may be caused by the instability of the 3G wireless links. Thus, we can conclude that in our simulation environment, deploying a CGW in 3G wireless network is inappropriate.

	MSA invites 3GSA			3GSA invites MSA		
	CGW in MANET	CGW in 3G core	CGW in 3G wireless	CGW in MANET	CGW in 3G core	CGW in 3G wireless
Average conference setup delay (second)	1.224	1.280	4.455	1.175	1.175	6.955

Table 7.4: Conference set up delay with CGW located on different network nodes

7.3 Conclusions

In this chapter, we have simulated the signaling architectures for standalone MANETs and for integrated MANETs/3G networks using OPNET. Through the experiments on scenarios in a standalone MANET, we have found that the clustering architecture is very promising and it can manage sessions for from small to large-scale conferences. Through the experiments on signaling in integrated MANETs/3G network, we have found that all the required scenarios (i.e. all participants in MANETs, all participants in 3G network or some participants in MANETs and some others in 3G network) work well. Moreover, in these scenarios the MANET participants benefit a lot from local session establishments, while the 3G participants take much longer to establish sessions with the centralized conference server.

An issue that we found from the simulation studies is that the capability exchange mechanism and the heartbeat detection/recovery mechanism introduced a much heavy load to the signaling system. This is because both mechanisms require periodical message exchanges in application layer. In order to solve this issue, we will introduce a new method — cross-layer design, in Chapter 8. A cross-layer design architecture will be demonstrated and cross-layer optimization schemes proposed, followed by the experimental results that will show how much benefit we can obtain from the new method.

Chapter 8 : Optimization of Signaling Systems: Using Cross-layer Design

Performance is one of the key issues for MANETs. In this chapter, we propose performance optimization schemes for the signaling systems, for both standalone and integrated MANETs/3G networks. The optimization is based on cross-layer design. We first pinpoint the performance issues in the proposed the signaling architectures. We then present the related work on cross-layer design for MANETs. After that, we present our cross-layer design architecture, followed by a set of optimization schemes. We evaluate and analyze some of these schemes through prototyping or simulation.

8.1 Performance issues of proposed signaling architectures

Application-layer heartbeat – The heartbeat mechanism we propose deals with the unintentional departures of nodes. Although we have considered to reduce the overhead introduced by the heartbeat messages, e.g. we have used a one-way request/replay scheme and the heartbeat messages are simple and lightweight, we found that the heartbeat does introduce significant overhead. Especially, when the heartbeat rate is high, the overhead can exceed the session establishment overhead. On the other hand, we cannot remove this overhead because the detection and handling of unintentional departures is important for conferencing in MANETs.

Application-layer capability exchange – The application-layer capability exchange scheme is used for meeting the requirement of optimal use of resources. It has been proposed to exchange node capabilities among parties. The capability information is maintained by each super-member and it is used for super-member elections. We have used SIP to implement this scheme and the SIP implementation is lightweight. Similar to the heartbeat message, the overhead introduced by application-layer capability exchange is non-trivial because of periodical messages.

The aforementioned mechanisms help to meet the signaling requirements that are not related to session management, but are necessary for handling the particularities of MANETs. For example, the unintentional departure is often caused by link break, node moving out-of-range or node crash. These issues are common to every layer of MANETs and they have been discussed frequently in relation to lower layers. A proof is that the AODV [39] routing protocol has used three non-responded "Hello" messages to detect a link break. The same thing happens to the optimal use of resources. The optimal use of resources is a common target for all MANET layers because resources in MANETs are scarce. Actually, some of the lower layer protocols have considered this requirement in their design. For example, WCA [48] uses the capability of nodes as one of the criteria to decide cluster heads. Thus, we argue that the use of heartbeat and capability exchange mechanisms in the application layer is not very efficient, but cross-layer design helps to share information among layers and to improve the overall performance.

CGW deployment and CGW publication/discovery – CGW is a key entity in the integrated signaling architecture. The proper deployment of a CGW can improve performance. There are two questions to be answered. The first is where to deploy a conference gateway. The second is how many conference gateways should be deployed. In Chapter 7, we have partially addressed this issue and we found that if we deployed the CGW in 3G wireless network, a longer session-setup delay would be introduced. However, this does not answer the question of the proper deployment of CGWs, i.e. the proper location of CGW deployment and the number of CGWs in the network. Furthermore, the CGW publication/discovery is yet a task that requires periodical exchange of messages at the application layer. Are there opportunities to reduce this overhead?

Sub-optimal routing – This is an issue caused by application-layer clusters. A new cluster member joins a cluster when the super-member or a member of the cluster invites it. The scheme has not considered whether the joiner is physically close to the cluster. This may introduce serious performance problems. In Figure 8.1, for instance, the shortest path between party A and party B for the routing layer is 3 hops. If signaling clusters in the application layer have been formed, the real path distance between party A and party B will be 11 hops: 4 hops between A and its super-member H, 5 hops between B and its super-member K and 2 more hops between the two super-members H and K. This issue can be somehow avoided if the application layer 'knows' physical location of nodes, e.g. if super-member K knows that party B is close to super-member C, it may not invite B but asks super-member C to do so. We believe that cross-layer design can help in this situation.

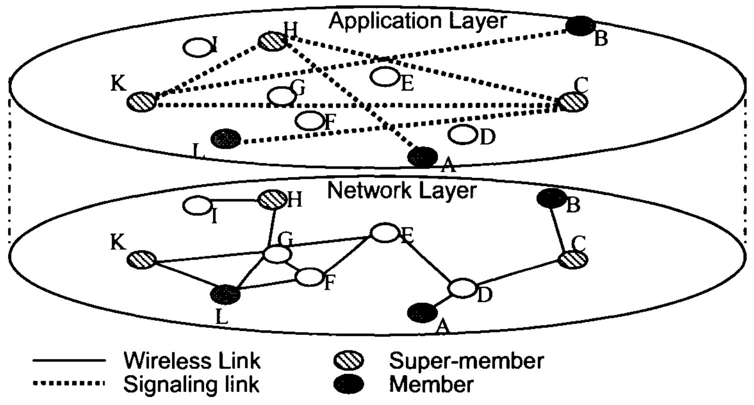


Figure 8.1: Issue of sub-optimal routing

From the above analysis, a need for cross-layer design is identified. Two of the performance issues are directly related to lower layer problems such as routing and link-break detection. The issue of optimal resource usage is common to every layer and the CGW deployment is related to lower layer network architectures. We consider that a research on the lower layer architectures of integrated MANETs/3G networks and a cross-layer design may help us to answer these questions.

8.2 Cross-layer design in MANETs: related work

8.2.1 Cross-layer design requirements for signaling optimization

The cross-layer design for signaling optimization should meet the following requirements. First, it should be easy to implement and does not introduce too much network overhead. Second, it should respect the cautionary aspects of cross-layer design, e.g. no design loop and an ease of upgrade. Third, the cross-layer information should be time-sensitive. This is important for the signaling system because conferencing is a real-time application. Fourth, the interoperability should be considered for the integrated signaling system. For example, an SA with a cross-layer design should be capable of interacting with an SA designed in traditional manner.

8.2.2 Related work in standalone MANETs

In Chapter 2, we have briefly introduced cross-layer design proposals or MANET and we have divided the proposals into two categories: global versus local solutions. The former usually presents new cross-layer design methods that are applicable for different architectures and benefit different layers. The latter consists of local solutions for specific architectures and requirements. For example, the work defined in [61] uses cross-layer design for the optimization of real-time video transmission. Local solutions are difficult to adapt for other applications or situations. Therefore, we will discuss them further.

Four global cross-layer design solutions are examined. CLASS [52] follows the method of direct communication between layers. It proposes a general mobility management architecture that allows for direct signaling flows between non-neighboring layers. This signaling interaction enables efficient mobility management for the whole system. However, it does not meet our requirements. For example, it requires logical changes in each layer, which is complicated to implement and upgrade. In addition, it can lead to conflicts and loops if the signaling has not been designed very carefully.

MobileMAN [53] considers the cautionary aspects of cross-layer design and it makes use of a shared database. It defines a repository called *Network Status* from which each layer can write and read information. It provides optimization for all the network functions and improves local and global adaptations. However, it is not easy to implement because it requires the redesign of protocols in each layer. Furthermore, the expiration of the data in the repository has not been discussed so the optimization scheme may not be efficient for time-sensitive applications. References [55] and [56] present some other drawbacks of MobileMAN, e.g. it may be cumbersome for protocols that neither write nor read the Network Status.

CrossTalk [55] extends the vision of MobileMAN. It introduces a global view of the network status while specifying the *Network Status* defined in MobileMAN as a local view. It is capable of providing real-time information for the optimization processes.

However, the global view is collected through a data dissemination procedure that incurs a significant overhead.

With the design goals of rapid prototyping, minimum intrusion, portability and efficiency in mind, ECLAIR [56] is proposed for the optimization of mobile device protocol stacks. It uses an approach similar to that of MobileMAN and CrossTalk. The difference is that it not only collects data from layers and stores them in a repository, but it also develops the optimization processes outside of the protocol stack. This abstraction makes the design more flexible and ensures a fast deployment. Similar to MobileMAN, it does not consider expiration of data, so the real-timeliness requirement is not fulfilled.

8.2.3 Related work in integrated MANETs

In the context of integrated MANETs (or MCNs), cross-layer design has been considered recently (e.g. [122] and [123]). The work in references [122] provides a cross-layer design for BTS routing discovery. It adjusts the functionality of physical, MAC and network layers. It also collects information from the three layers so that an efficient route can be discovered. The performance evaluation shows that the proposed schemes can achieve faster route discovery and more reliable route setup. However, the proposal is a specific method that cannot be used by our signaling system. Furthermore, it has not considered interoperability issue. Similar to what presented [122], Kannan et al. [123] present another MCN routing protocol using cross-layer design. It considers a set of constraints (e.g. interference level) when discovering a route. These constraints are collected from physical or MAC layer. Like [122], this work cannot meet our requirements. To the best of our knowledge, there is no global cross-layer design solution proposed for integrated MANETs.

After a review of existing cross-layer solutions for both standalone and integrated MANETs, we cannot find a solution that meets all our requirements. In next section, we propose a novel cross-layer architecture that is especially suitable for application-layer optimizations.

8.3 A cross-layer optimization architecture

The cross-layer optimization architecture is shown in Figure 8.2. In this architecture, we apply the shared database method and consider the cautionary aspects of cross-layer design. The general principles are summarized as follows: An application protocol defines optimization schemes and specifies the types of information that it wants to acquire from lower layers. The lower layers provide the information, which is stored in and retrieved from a shared database.

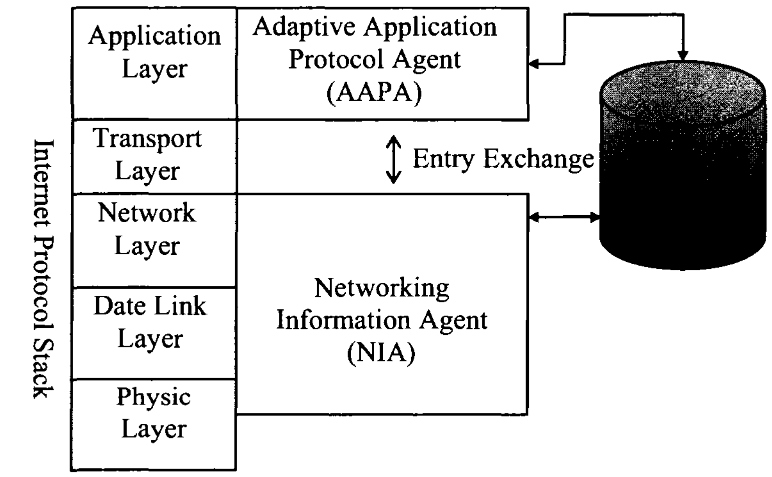


Figure 8.2: Cross-layer optimization architecture

8.3.1 Entities

We define three entities: Share sPace (SP), Adaptive Application Protocol Agent (AAPA) and Networking Information Agent (NIA). SP is a repository from which application protocols can retrieve lower-layer information. A share space contains a set of entries. Each entry represents one type of information. It contains an entry type, an entry value and a keep-fresh timestamp, which is used to ensure the freshness of the information. If the information has not been updated for a given period of time, it will expire.

An entry is initially defined by an AAPA, which is responsible for translating the information types that are requested by application protocols into common entry types. These common entry types are exchanged and understood by AAPAs and NIAs. An AAPA is also responsible for creating entries in the SP. An NIA is responsible for collecting information from the lower layers and updates the relative entries in the SP. When there is a new update, the SP will inform the AAPA and the AAPA will then cache the updated information, reformat it and send it to the application protocol.

The purpose of using AAPA and NIA is to mask the complexity of the cross-layer design so that existing protocols in single layers do not need to have major upgrades to support cross-layer optimization. The protocols in the lower layers only need to open their data structures to the NIA, and the application protocol simply sends requests to and gets responses from its AAPA. Any functional upgrade or version change in a layer is still independent of other layers. However, the AAPA and NIA should be upgraded accordingly.

8.3.2 Benefits of the optimization architecture

This architecture benefits from the advantages of both layered design and cross-layer design. Compared to the existing proposals, the architecture is more application-layer oriented. Although shared database-based proposals such as MobileMAN and ECLAIR have defined some standard parameters for each layer, the shared data cannot really adapt to the different requirements of diverse applications. For example, routing cluster information is useful for optimizing a clustering-based application but it may not be considered as a standard parameter. Our architecture makes it possible to exchange the data entries, and the new entries can be negotiated before performing an optimization. This allows space for customization of cross-layer parameters.

In our solution, the lower-layer information is retrieved locally. Timestamps are used to ensure the freshness of the information. There is no extra spreading overhead introduced in the network. On the other hand, the SP may not include as much of the information as that included in the global view defined in CrossTalk, but it includes necessary information required by application. Similar to ECLAIR, our architecture is flexible and quickly deployable. The application optimization schemes are designed independent of basic application logics. Thus, it can simply return to the basic logic if an optimization is not performed.

8.4 Optimization schemes

We propose six optimization schemes: link-break handling, capability usage, sub-optimal routing, super-member election based on clustering, super member election based on signal power and CGW deployment. The last two schemes are specific to the signaling for an integrated MANET/3G network. As a cross-layer design approach, each scheme uses a particular type of information from lower layers. There is no specific lower layer protocol required for the architecture. The general principle is that an optimization scheme is only performed when the lower-layer protocol(s) can provide the related information. This principle ensures that the optimized signaling scheme can run on different lower-layer routing protocols. It should be noted that the six optimization schemes are independent of each other. These schemes are examples of using the proposed architecture, but they are not exhaustive list of possible schemes.

8.4.1 Common entries

Six common entries are defined using the format <type, value, timestamp>. These entries are the *routing table* <routing_table, route_list, timestamp>, the *node capability list* <node_capability, capability_info_list, timestamp>, the *neighbor list* <neighbor_hop_list, address_list, timestamp>, the *routing-layer clustering* <routing_cluster, cluster_info, timestamp>, the *routing topology* <routing_topo, route_list, timestamp>, and the *signal power* <signal_power, power_value, timestamp>. In the *routing table* entry, each route in the route_list consists of a source address, a destination address, a path, a hop count, an active status and a timestamp. In the *node capability list* entry, each capability_info consists of a node address, capability types and corresponding capability values. In the *neighbor list* entry, a list of neighboring nodes' addresses is stored. In the *routing layer clustering* entry, cluster_info contains a cluster-head. If the node is a cluster-head, it also contains a list of cluster members. This entry may not contain all of the nodes' capabilities. The *routing topology* entry stores the routes to reach every MANET node in an integrated architecture. In the *signal power* entry, a signal power value is stored. The signal power of a node is its wireless signal strength to the BTS.

8.4.2 General optimization schemes for signaling in MANETs

Link break handling optimization scheme: performed when the routing table entry is updated. Every routing protocol provides this information. A super-member checks the route status for each of its connected members and super-members. If it discovers a route failure, it will terminate the session. This scheme helps the signaling system to handle a forced departure gracefully without using a heartbeat mechanism.

Capability usage optimization scheme: when the first super-member is elected, participants check their local node capability list. If there is a fresh capability list, the participant will copy and use this list for super-member election, i.e. the participant with the highest level of capability becomes the super-member. After the establishment of the first session, the super-member checks its local capability list only when it needs to elect a new super-member. This scheme can only be invoked when the lower layer protocol takes node capabilities into consideration. Using this scheme, we can avoid invoking the application-layer capability exchange mechanism.

Sub-optimal routing optimization scheme: uses the entry of the neighbor list. Some of the routing protocols (e.g. proactive routing protocols) can provide this information. The scheme can be described as follows.

- When a new member joins the conference, all the super-members are informed.

- Each super-member checks its neighbor list. If the information is fresh and if it contains the address of the new joiner, it will ask the super-member that has invited the joiner (which we call the *original super-member*) to switch the new joiner to its cluster. As there may be more than one switching request, the original super-member will choose the one with the most available capabilities to switch the session.
- Periodically, a super-member checks its neighbor list. If there is a conference member that is its neighbor but is not its member, it will ask the super-member that has connected to the member to switch the session. The super-member may initiate a switch or refuse the switch request if it is also adjacent to the member.

This scheme ensures that a member joins a cluster when the cluster head has a direct link with the member. However, it does not ensure the shortest path. We do not recommend an obligatory shortest path optimization because it may lead to very frequent session switches. In addition, it may seriously increase the network overhead. For example, super-members would be required to exchange their routing information frequently.

Super-member election based on clustering scheme: it is performed when the routing cluster entry is updated. Node capability is still the major criterion for super-member election. However, if all the candidates have a similar capability level, the cluster heads in the routing layer will have priority to be chosen as super-members in the application layer. This helps to further optimize the signaling route. This scheme can only be invoked when the routing protocol uses a clustering scheme.

8.4.3 Specific optimization schemes for integrated MANETs/3G networks

CGW deployment optimization scheme: uses the entry of route topology. In most of the routing protocols of MCNs, there is a route topology stored in an entity of the infrastructure side of the network. The entity may be a BTS (e.g. in [122]) or a Mobile Switching Center (MSC) (e.g. in [25]), depending on what type of lower layer routing protocol is deployed. The idea is that we collocate a CGW with the entity where there is a routing topology. In this case, the CGW can periodically acquire the MANET nodes' information from the routing topology and update its participant location repository. The overhead related to the frequent application-layer location update can then be avoided. For the question of how many CGWs should be deployed, in the conditions of our considered environment, we propose that one CGW be deployed per multihop routing area. This will facilitate the signaling routing procedure.

Super-member selection based on signal power scheme: the signal power entry is used. A node with the highest signal power has a higher priority to be a super-member. The signal power value is provided by the physical layer. In most cases, signal power is influenced by the distance between a BTS and an MS, and thus a higher signal power may reflect a short path between a super-member and the CGW. Also, we consider that when a node is physically closer to a BTS, it is less prone to move out of range. This helps to optimize the signaling route and improve cluster stability.

8.4.4 Interoperability analysis

The interaction between parties with and without cross-layer optimizations is a complex issue. Some of our optimizations may cause *interoperability* problems while others may not, depending on local versus global effect of an optimization scheme. A local effect does not cause an *interoperability* issue, while the global effect causes an issue if both parties make a decision at the same time and their decisions are in conflict. Within the optimization schemes introduced thus far, those that may cause serious *interoperability* problems are the super-member election optimization schemes. In a cluster, more than one party may be selected as a super-member based on different rules. This is not allowed in our cluster scheme. One solution is that whenever a super-member detects another super-member in the same cluster, it checks if its cross-layer information is active. The super-member without active cross-layer information changes itself to a cluster member.

8.5 Evaluation of the optimization schemes

In this section, we evaluate the impact of three schemes: capability usage, link break handling and CGW deployment through prototype or simulation experiments. The first three sub-sections are devoted to these evaluations. In the last subsection, we discuss the other three schemes: sub-optimal routing, super-member election based on clustering and super-member election based on signal power. It can be found that these schemes are proposed to solve the issue of sub-optimal routing, so we call them sub-optimal routing related schemes.

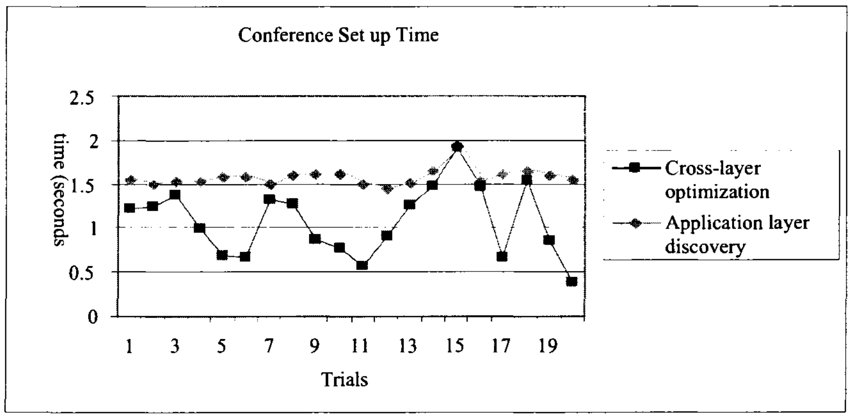
8.5.1 Capability usage optimization

The scheme for optimizing capability usage has been evaluated through prototyping. We implement the scheme directly in the signaling prototype for standalone MANET (see Section 6.1.5). We have implemented the AAPA, the NIA and the SP in each SA. For simplicity, the SP is implemented as a local file. The AAPA and NIA read and write the file. The *node capability list* entry has been implemented. We also simulate a simplified node capability-aware routing protocol. In order to evaluate the efficiency of the capability usage optimization scheme, we compare the two schemes (the application-layer capability discovery and the cross-layer capability usage optimization) with respect to three metrics: session set up time, network overhead, and program footprint.

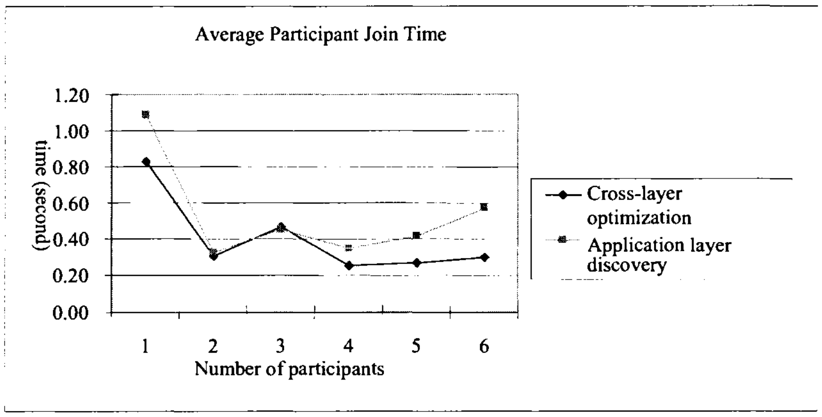
Figs. 8.3-a and 8.3-b show the results of the conference set up time and the participant-join time. It shows that a conference set up always takes less time (0.5 second in the average) with the cross-layer scheme. A similar result is observed in most of the participant joining cases. For the second and the third party joining, it does not show significant difference between the two schemes. This is due to the procedure of a participant joining, i.e. the participant first creates a session with a super-member. The capability discovery is performed as a second step. This differs from the procedure of a conference set up, where the capability discovery is performed before a session is set up. The delay of the first capability exchange procedure has thus not been calculated for participant joining situations. This result can only be found in small-scale cases. The benefits of the cross-layer design will be more significant with a larger number of participants because the capability exchange delays caused by the cluster splitting will be calculated with a larger number of participants. For example, in Fig 8.3-b the split value is set to four and the benefit of cross-layer design starts showing up when the fourth party joins.

In terms of overhead, although the application layer discovery protocol that we use is very simple and lightweight, Figs. 8.3-c and 8.3-d shows that the cross-layer scheme introduces much less network overhead for both participant joining and the setting up of a conference. In order to implement the two schemes, we added about 800 lines of code for the cross-layer scheme, and we added about 600 lines of code for the application layer discovery protocol. The footprint difference between the two schemes is not significant.

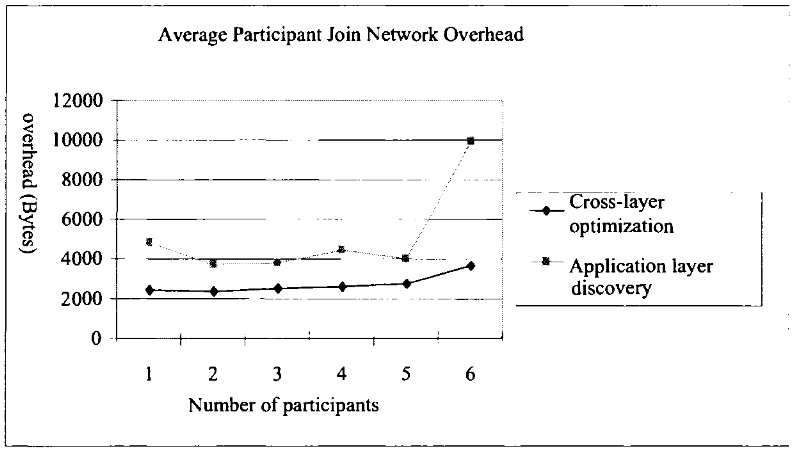
In the figures, some curves show significant variations while others are rather stable. This is due to the instability of the wireless connections. We observed that wireless links were weak in some trials, leading to longer destination discovery delays. This also introduced more network overheads, such as routing traffics. It should be noted that the wireless link instability affects the performances of both architectures, i.e. the architecture with as well as the architecture without the capability usage optimization. However, in the average, the signaling system performs better with the optimization.



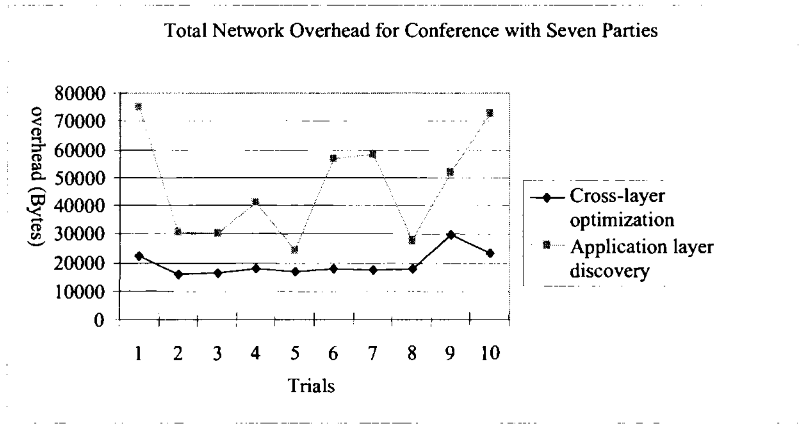
8.3-a: Conference set up time



8.3-b: Participant join time



8.3-c: Average party join overhead



8.3-d: A seven party conference overhead

Figure 8.3: Capability usage optimization results

8.5.2 Link break handling optimization

We evaluate the link break handling optimization scheme through simulation experiments. The simulation is based on the simulation that we built for evaluating the signaling for standalone MANET (see Section 7.1). We add the AAPA, NIA and SP entities in each MANET nodes. The node model with cross-layer design is demonstrated in Figure 8.4.

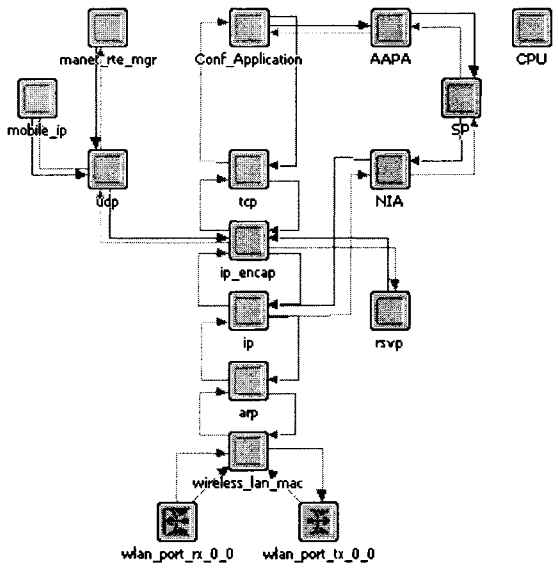


Figure 8.4: MANET node model with cross-layer design

In the simulation, we implemented the entry of *routing table*. We added a simple function “notify_NIA()” in the AODV process, which opens the routing table to NIA. The function is responsible for sending an internal message to NIA. The NIA that receives this message will create a routing table entry in SP. When routing layer detects a link failure, the AODV process will call a function “inform_NIA_conn_loss()” and this function also sends a message to NIA. The NIA that receives the message will update the *route status* of the *routing table entry* in SP. Figure 8.5 shows where the AODV calls the function.

```
static void aodv_rte_neighbor_conn_loss_handle (void* neighbor_conn_info_ptr1, int PRG_ARG_UNUSED
(code))
        {
        AodvT_Conn_Info*            neighbor_conn_info_ptr;
        char                        neighbor_addr_str [INETC_ADDR_STR_LEN];

        /** Connection to the neighbor node has been          **/
        /** lost. Handle the loss appropriately                   **/
        FIN (aodv_rte_neighbor_conn_loss_handle (<args>));
                neighbor_conn_info_ptr = (AodvT_Conn_Info*) neighbor_conn_info_ptr1;
                aodv_packet_queue_all_pkts_to_dest_delete (pkt_queue_ptr, neighbor_conn_info_ptr-
>neighbor_address);
                /* Send out route error messages */
                aodv_rte_route_error_process (neighbor_conn_info_ptr->neighbor_address, OPC_NIL,
AodvC_Link_Break_Detect, OPC_FALSE);
                /* Create the neighbor address string        */
                inet_address_print (neighbor_addr_str, neighbor_conn_info_ptr->neighbor_address);

                /* to inform NIA about the connection loss*/
                inform_NIA_conn_loss (neighbor_conn_info_ptr->neighbor_address, op_sim_time());

                /* Remove the entry for this node from the neighbor connectivity hash table */
                prg_string_hash_table_item_remove (neighbor_connectivity_table, neighbor_addr_str);
                inet_address_destroy (neighbor_conn_info_ptr->neighbor_address);
                op_prg_mem_free (neighbor_conn_info_ptr);
        FOUT;
        }
```

Figure 8.5: AODV informs NIA a connection loss

Through experiments on the scenario of a seven party conference with one link failure introduced, we find that the session failure detection delay is about 6 seconds when using the optimization scheme. In order to further evaluate the scheme, we compare the scheme with the application-layer, heartbeat-based session failure detection mechanism. In Figure 8.6, we compare two schemes and two sets of heartbeat parameters are evaluated. For the first set of parameters (*Th*=60s, *Tt*=10s and *n*=3), the network load introduced is not much higher than the load of using the cross-layer optimization scheme, but the detection delay introduced is much longer (60s vs. 6s). For the second set of parameters (*Th*=6s, *Tt*=1s and *n*=3), the detection delay is about 6 seconds. That is close to the detection delay when using the cross-layer optimization scheme. However, a much higher network load is introduced in the case. Therefore, we can conclude that the optimization scheme promises a rapid failure-detection without introducing a high network overhead. Like in the prototype, we did not introduce significant footprint in order to implement cross-layer optimizations in the simulation.

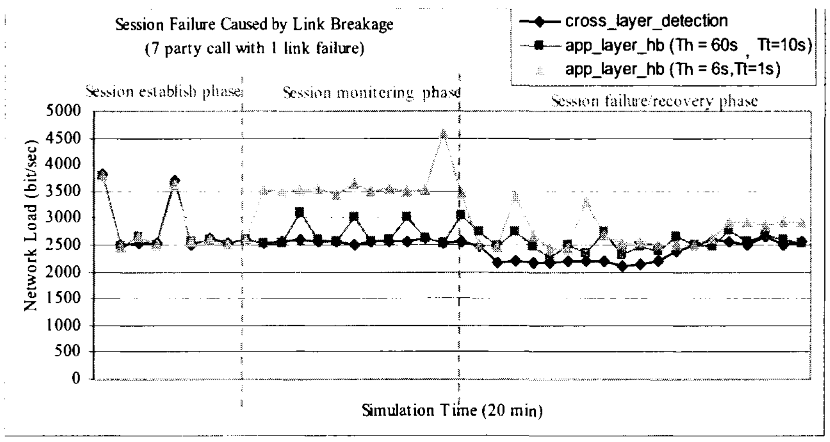


Figure 8.6: Simulation results for link break handling optimization

8.5.3 CGW deployment optimization

We also evaluate the CGW deployment optimization scheme through simulation experiments. This simulation is based on the simulation that we built for evaluating the signaling for integrated MANETs/3G networks (see Section 7.2). We implement the *routing topology* entry and we assume that the routing topology of MANET nodes is maintained by the gateway node (the node that connects the two wireless networks: MANET and 3G wireless). The AAPA, NIA and SP are also implemented inside in the gateway node. It should be noted that the routing topology of MANET nodes can also be implemented in Node-B or RNC or SGSN. The reason we implement it in the gateway node is simplicity.

In order to evaluate CGW optimization scheme, we compare the signaling overhead of using the simple CGW/MSA discovery protocol (see Section 6.2.3) with that of using the

CGW optimization scheme where the CGW finds MSAs location by retrieving the local routing topology information. It should be noted that in the optimization scheme the CGW directly publishes its address and listening port to MSAs but not uses a broadcast. Figure 8.7 shows that the CGW optimization scheme introduces much less overhead than the usage of MSA discovery protocol in application layer. In the figure, the signaling overhead represents the total conference establishment overhead including the overheads of CGW/MSA discovery and the session establishments. From two to ten-party conferences have been evaluated. Each conference includes parties from both 3G and MANET sides. We have also performed experiments in order to evaluate the session setup delays and we observe that the CGW deployment optimization scheme does not reduce or increase the session setup delay, but the location of the CGW deployment affects the session setup delay.

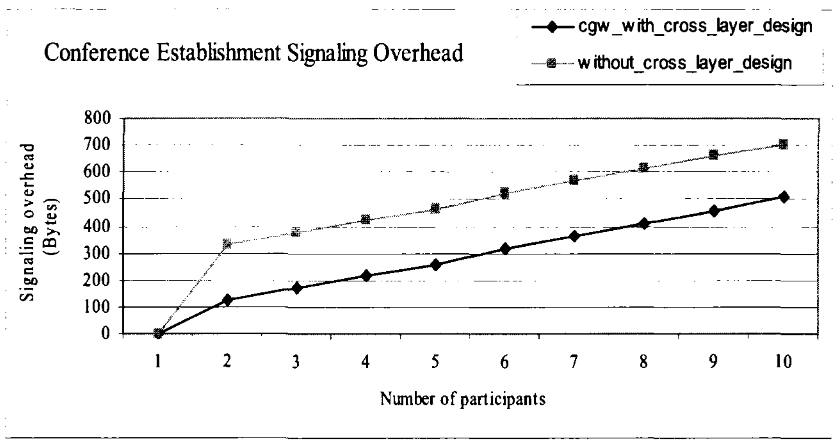


Figure 8.7: Simulation results for CGW deployment optimization

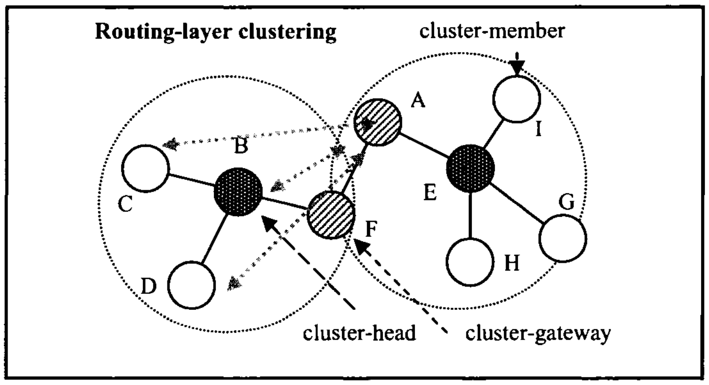
8.5.4 The sub-optimal routing related optimization schemes

We have implemented the most important three optimization schemes and we have shown that these schemes can help to solve the performance issues described in Section

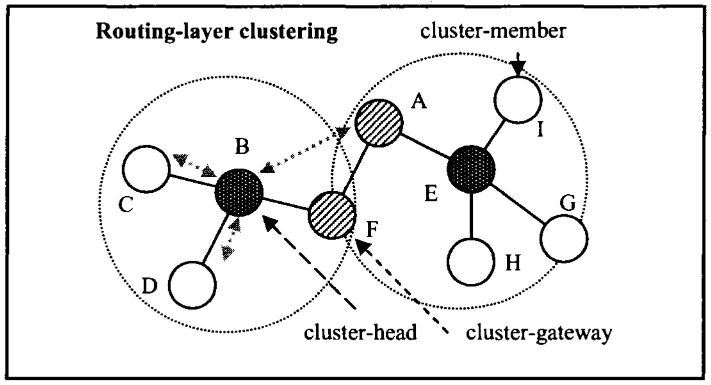
8.1. In this section, we discuss the remaining optimization schemes used for solving the issue of sub-optimal routing.

Unlike the three optimization schemes that we have measured, the sub-optimal routing related optimization schemes only show its benefit when there is a sub-optimal route. Figure 8.8 shows an example that the "super-member election based on clustering" scheme is used for optimizing the signaling routes. Figure 8.8-a shows the case in which no optimization is applied. Party A initiates a conference with party B and party A and B's capabilities are the same. Party A is elected as super-member because A is an initiator. Party A then invites party C and party D to join the conference. Three sessions are therefore established and we denote them as AB, AC and AD. We can find that the route distances of the three sessions are AB with 2 hops, AC with 3 hops and AH 3 with hops. If we compare the total signaling route distance with those shown in Figure 8.8-b where the party B is elected as the super-member because it is a lower-layer cluster head, we can find that the signaling route in the latter case is optimized, i.e. BA with 2 hops, BC with 1 hop and BD with 1 hop.

By application of the sub-optimal routing related optimization schemes, we expect to reduce the session setup delay and to reduce the overall network load. However, there is a price to pay. For example, with a usage of the "sub-optimal routing optimization" scheme, the super-members have to frequently check if their physical neighbors are in the conference and are their cluster members. Furthermore, a session switch may happen, which will introduce extra signaling overhead. In order to evaluate the actual impact of these schemes, a comprehensive comparison between cost and effect should be made. A possible method to do this comparison is through simulation but the design of simulation scenarios should be careful. Scenarios with sub-optimal signaling routes should be implemented.



8.8-a: Before optimization: party A elected as super-member



8.8-b: After optimization: party B elected as super-member

Figure 8.8: A case of using super-member election based on clustering optimization scheme

8.6 Conclusions

In this chapter, we have addressed several performance issues of our signaling systems and we proposed a cross-layer optimization architecture. The architecture is more application-layer oriented compared to the existing cross-layer proposals. Six optimization schemes have been proposed and three of them prototyped or simulated. The results from the experiments have been analyzed and discussed. They show that cross-layer optimizations are very promising for enhancing the performance of our signaling architecture. In addition, these optimizations do not necessitate a large footprint overhead. The remaining three optimization schemes are all proposed for solving the sub-optimal routing issue. We have briefly discussed them and will carry out these schemes in future work.

Chapter 9 : Conclusions and Future Work

In this chapter, we summarize the contribution of this thesis. We also discuss the remaining issues and directions for future work.

9.1 Summary of contributions

MANETs have been an active research area for years. A key motivation is the possibility of novel application scenarios. Conferencing enables a set of attractive applications. It is very challenging to deploy a conference in MANETs. In this thesis, we have addressed an important conferencing aspect: signaling. We have investigated signaling issues in two types of MANETs: standalone and integrated MANETs/3G networks.

The major contributions of this thesis are as follows.

- **Identified the issue of signaling for conferencing in MANETs** – We have seen that the signaling for conferencing in MANETs is very challenging due to the particularity of MANET. The absence of a pre-configured infrastructure makes it difficult to build a fixed signaling architecture. The frequently changing network nodes and topology introduce complexity for session management. In addition, it has to take into account the scarce network resources and the unstable wireless links. The challenges make the signaling issue a very interesting research topic.
- **Derived signaling requirements for conferencing in MANETs** – We have derived signaling requirements for conferencing in both standalone MANETs and integrated MANETs/3G networks. We have also reviewed the state-of-the-art signaling protocols such as SIP, H.323 and ICEBERG with respect to these requirements. We found that none of the existing signaling protocols meets all the requirements. Most

of the solutions are only suitable for infrastructure-based networks. A few others are too to handle the complexity of signaling in MANETs.

- **Proposed a cluster-based signaling architecture for standalone MANETs** – The cluster-based signaling architecture is a core contribution of this thesis. It has been proposed to meet all the derived signaling requirements for standalone MANETs. The architecture is very different from the existing signaling solutions. First, it is based on dynamic clusters that are created and deleted only by conference participants. Second, a cluster-head election is based on participant capabilities (e.g. energy power and processing capability). Third, signaling sessions are monitored by a heartbeat mechanism so that a session failure can be detected and the signaling architecture can be recovered.
- **Proposed a signaling architecture for integrated MANETs/3G networks** – In the literature, there is no solution for signaling for conferencing in integrated MANETs/3G networks. Our signaling architecture has filled this blank. It uses a conference gateway as a mediator to integrate both signaling architectures: the cluster-based architecture and the 3GPP 3G architecture. It enables all the required signaling scenarios but does not necessitate a significant upgrade on signaling systems from both sides.
- **Implemented the signaling architectures, proposed an extension to SIP and built proof-of-concept prototypes** -- SIP is a lightweight and extendible signaling protocol. We have selected it as our implementation technology and proposed a SIP extension that supports the concept of 'clustering'. In order to implement the extension in a real MANET environment, we also discussed the SIP deployment

approaches and built prototypes for both standalone MANETs and integrated MANETs/3G networks. The prototypes ran under a small-scaled IEEE 802.11 ad hoc network environment and they have allowed us to test different signaling scenarios and collect results. From the experiments, we found that clustering is promising for the establishment of small-scaled conferences and SIP as an implementation technology works well in every scenario.

- **Evaluated performance of the signaling architectures using OPNET** – We have simulated the cluster-based signaling system using MANET features of OPNET. We have built different scenarios and measured different aspects of the signaling system. It has shown that the signaling system can establish and terminate sessions for from small to large scales of conferences. It can handle the unintentional departure of nodes and recovery sessions. In addition, it meets other requirements such as independence of lower layer protocols. We have also simulated the signaling architecture for integrated MANETs/3G networks. A one-cell integrated MANET/3G network was built and different conference scenarios were tested. It shows that our signaling system does support the required scenarios and it meets all the specific signaling requirements for integrated MANETs/3G networks.
- **Optimized the signaling schemes using a cross-layer design method** – Observing several performance issues of our signaling systems, we have proposed a cross-layer design based optimization solution. The solution includes a new cross-layer design architecture that is especially suitable for application-layer optimizations, and a set of optimization schemes. We have also evaluated some of these schemes through

prototyping and simulation. It shows that the schemes can significantly improve signaling performance.

9.2 Future work

This thesis provides solutions for signaling in MANETs and it validates them using prototypes and simulation. However, much work remains to be done. We organize it into three categories: architectures, implementation and validation, and other remaining open issues.

9.2.1 Signaling architecture

We have shown that the signaling architecture for standalone MANETs is based on clustering. Different from lower layer clusters, the signaling clusters are formed during session establishments. Thus, the scheme does not require a period of network stable time as that required by the lower-layer clustering schemes (seen Section 2.2.2). However, it still has a cluster stability issue. For example, frequent super-member leavings lead to re-formation of clusters, which consumes a large amount of the network bandwidth. It is impossible to avoid this issue because any participant in a conference is free to leave and furthermore any super-members may unintentionally leave (e.g. they move out of range). A possible solution for improving cluster stability is to select deputy cluster heads (as defined in [91]). It will be interesting to investigate whether we can use or how we can use deputy super-members in our signaling architecture.

The signaling architecture is based on flat clustering with all the super-members interconnected with each other. The purpose of the architecture is to reduce to the distance of signaling paths, but when a conference grows to a very large scale, i.e. thousands of participants, the cost of building the super-member mesh would be very expensive. A possible solution is to introduce a hierarchy. An example of hierarchical clustering is presented in [47], in which a cluster has a 'level' parameter. A higher level cluster head controls a cluster of cluster heads at lower level. In future work, we would like to examine how to build a hierarchical clustering-based signaling architecture.

9.2.2 Implementation and validation

Conti et al. [124] show that a key issue that slows down the adoption of MANET is the lack of real world implementation and industrial deployment. In this thesis, we have discussed the implementation and deployment of our signaling system. However, the work suffers from the lack of a sophisticated MANET environment. The testbed for prototypes only includes several nodes and we have used open-source AODV software that does not work well when setting up a multihop route. Consequently, we only test our prototype in a small-scaled, meshed ad hoc network. Experiments on a larger scaled, multihop MANET is foreseen as future work.

Middleware plays an important role in our prototype. We have used JXTA for participant discovery, but JXTA is not designed for MANETs. It is not only hard to configure but also performs badly in MANETs. Thus, development of a middleware that is especially suitable for MANETs (e.g. Expeerience [112]) is an interesting task to tackle in the future.

In the simulation, we have evaluated conference scales up to 100 nodes. It would be interesting to know the behaviors of the signaling scheme with hundreds and even thousands of nodes. However the current MANET features in OPNET only support up to 200 nodes when running an AODV routing protocol. Since we also used TCP on top of AODV, we found that TCP transmission errors and re-transmission becomes more frequent when we set up a conference with more than 50 nodes. It is possible for the OPNET MANET feature to support more nodes when running on a dual-core machine. Thus, it would be interesting to investigate this possibility and test our signaling system in larger scaled conferences.

The evaluation of the signaling system in integrated MANET/3G networks is. As next step, it would be interesting to build a simulation including multiple cells and multiple routing areas.

9.2.3 Open issues and future directions

There are some issues that we have not tackled in this thesis. These issues are either not too close to the signaling issue or less critical to basic signaling functionality. However, they are important for real deployment of conferencing applications in MANETs and consequently they require further development and study.

The first issue is billing and charging. This issue is less relevant to standalone MANETs because it is not clear whether there is a network operator that provides the standalone MANET services, but it is import for integrated MANETs (e.g. [125], [126]). In this case, the billing point is typically put in the access point or in the infrastructure network side. Charging a MANET user is a complex issue. It is hard to evaluate how many resources a user has actually used because the user may also contribute to relaying traffic of other users. This is also related to another issue in MANETs: selfishness.

Selfishness means that a user may conserve its processing capability and refuse to provide a service to other users (e.g. store and forward a packet). The selfishness is a common issue for all MANET users and it has been discussed in routing protocols (e.g. [127]). In our signaling system, it may happen when a participant refuses to act as a super-member. Lamparter et al. [125] proposed a charging policy that encourages users to contribute. When a user forwards packets, it gains credits. Preventing the selfishness is still an open issue and more research needs to be done.

Still another issue is security. The issue was not addressed in this thesis but it is partially considered in SIP, which provides basic security mechanism. It suggests the potential use of email security mechanisms like PGP (Pretty Good Privacy) [128] and a SIP URI can try to build up a secure transport layer tunnel using Transport Layer Security (TLS) [129]. IP security (IPsec) [130] can also be used as a general purpose mechanism to encrypt all the SIP communications. However, these security mechanisms do not work well in MANETs and they face new challenges. Yang et al.[131] demonstrates these challenges and proposes new security mechanisms. How SIP works with these MANET security mechanisms is an interesting problem to look at in the future.

References

[1] George Aggelou, “Mobile Ad Hoc Networks – From Wireless LANs to 4G Networks”, a book published by McGraw-Hill Companies, Inc., 2005

[2] Marco Conti and Silvia Giordano, “Multihop Ad Hoc Networking: The Theory”, IEEE Communications Magazine, April 2007, Volume 45, Issue 4, Page(s):78-86

[3] Jennifer J. –N. Liu and Imrich Chlamtac, “Mobile Ad hoc Networking with a view of 4G Wireless: Imperatives and Challenges”, Mobile Ad Hoc Networking, Wiley-IEEE Press, July 2004

[4] Ram Ramanathan, “Antenna Beamforming and Power Control for Ad Hoc Networks”, Mobile Ad Hoc Networking, Wiley-IEEE Press, July 2004

[5] Sunil Kumar, Vineet S. Raghavan and Jing Deng, “Medium Access Control protocols for ad hoc wireless networks: A survey”, Elsevier, Ad Hoc Networks, Volume 4, Issue 3, May 2006, Pages 326-358

[6] Elizabeth M. Belding-Royer, “Routing Approaches in Mobile Ad Hoc Networks”, Mobile Ad Hoc Networking, Wiley-IEEE Press, July 2004

[7] C. Fu, R. H. Glitho and R. Dssouli, “A Novel Signaling System for Multiparty Sessions in Peer-to-Peer Ad Hoc Networks”, Proceedings of IEEE Wireless Communications and Networking Conference, Volume 4, March 2005, Page(s):2287 - 2292

[8] C. Fu, R. Glitho, F. Khendek, “Signaling for Conferencing in Integrated 3G/Mobile Ad Hoc Networks”, IEEE symposium on Computers and Communications’ 06. Proceedings, June 2006, Page(s):838 - 843

[9] C. Fu, R. Glitho and F. Khendek,, “Signaling for Multimedia Conferencing in Standalone Mobile Ad Hoc Network”, submitted to IEEE Transactions on Mobile Computing on September 2007

[10] C. Fu, R. Glitho and F. Khendek, “A Novel Session Recovery Mechanism for Cluster-based Signaling Architecture for Conferencing in MANETs”, Distributed Computing Systems Workshops, IEEE International Conference on Distributed Computing System '07, Workshop, June 2007, Page(s):19-19

[11] C. Fu, R. Glitho and R. Dssouli, “Methods for cluster-based multi-party conferencing in ad-hoc networks”, United State Patent Application, Serial No.: 999920, Series Code: 10, June 1, 2006

[12] C. Fu, R. Glitho and R. Dssouli, “Cluster of terminals and ad-hoc network for cluster-based multi-party conferencing”, United State Patent Application, Serial No.: 999944, Series Code: 10, June 1, 2006

[13] C. Fu, F. Khendek, R. Glitho, “Signaling for multi-media conferencing in 4G: the Case of Integrated 3G/MANETs”, IEEE Communications Magazine, Volume 44, Issue 8, Aug. 2006 Page(s):90 - 99

[14] C. Fu, R. Glitho and R. Dssouli, “User agent and super user agent for cluster-based multi-party conferencing in ad-hoc networks”, United State Patent Application, Serial No.: 999955, Series Code: 10, June 1, 2006

[15] C. Fu, R. Glitho and R. Dssouli, “Method and conference controller for cluster-based conferencing”, United State Patent Application, Serial No.: 280206, Series Code: 11, May 17, 2007

[16] C. Fu, R. Glitho and F. Khendek, “Cross-Layer Design for Optimizing the Performance of Clusters-Based Application Layer Schemes in Mobile Ad Hoc Networks”, IEEE CCNC, Jan. 2007, Page(s):239 - 243

[17] C. Fu, R. Glitho and F. Khendek, “A Cross-layer Architecture for Signaling in Multihop Cellular Networks”, accepted for publication, to appear in IEEE Communications Magazine

[18] Magnus Frodigh, Per Johansson and Peter Larsson, “Wireless ad hoc networking – the art of networking without a network”, Ericsson Review No.4, 2000

[19] Marco Conti, “Body, Personal, and Local Ad Hoc Wireless Networks”, in: M. Ilyas (Ed.), Handbook of Ad Hoc Networks, CRC Press, New York, 2003 (Chapter I)

[20] Bangnan Xu, Sven Hischks and Bernhard Walke, “The Role of Ad Hoc Networking in Future Wireless Communications”, Proceedings of International Conference on Communication Technology 2003, Volume 2, Page(s):1353-1358

[21] Y. Lin and Y. Hsu, “Multihop Cellular: A New Architecture for Wireless Communication”, IEEE INFOCOM 2002, Vol. 3, Page(s): 1273 - 1282

[22] H. Wu, C. Qiao, S. De, and O. Tonguz, "Integrated Cellular and Ad Hoc Relaying Systems: iCAR", IEEE JSAC, vol.19, 2001, pp. 2105-15

[23] H.Luo et al., "UCAN: A Unified Cellular and Ad-Hoc network Architecture", proc. ACM International Conference on Mobile Computing and Networking, Sept. 2003.

[24] Z. Dawy, S. Davidovic and I. Oikonomidis, "Coverage and Capacity Enhancement of CDMA Cellular System via Multihop transmission", IEEE Globecom, Page (s) 1147-1151, 2003

[25] H. Lee and C. Lee, "An Integrated Multi-hop Cellular Data Network", IEEE Vehicle Technology Conference, 2003, Vol. 4, Page(s) 2232-2236

[26] Y. Liu et al., "Integrated Radio Resource Allocation for Multihop Cellular Networks with Fixed Relay Stations", IEEE JSAC, Vol. 24, Page(s) 2137-2146, Nov. 2006

[27] G. Kannan, S. N. Merchant and U. B Desai, "Access Mechanism for Multihop Cellular Networks", IEEE VTC-2007, Page(s) 279-283, 2007

[28] A. Kusuma, L. Andrew and S. V. Hanly, "On Routing in CDMA Multihop Cellular Networks", IEEE Globecom, Page(s) 3063-3067, 2004

[29] Y. Wu, K. Yang and J. Zhang, "An Adaptive Routing Protocol for an Integrated Cellular and Ad-hoc Network with Flexible Access", ACM International Wireless Communication and Mobile Computing Conference'06, Page(s) 263-268, Vancouver, 2006

[30] A. R. Wolff and C. Lee, "Large Scale Routing in a Multi-Hop Cellular Network Using a Radial Geometric Approach" IEEE Wireless Communications and Networking Conference, Page(s) 4446 - 4451, 2007

[31] P. P. Lam and S. C. Liew, "Nested Network Mobility on the Multihop Cellular Network" IEEE Communications Magazine, Volume 45, Issue 9, Page(s) 100 – 104, September 2007

[32] 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, Volume 12, Issue 3, Page (s): 30-41

[33] IEEE 802.15.3 TM, "Part 15.3: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for High Rate Wireless Personal Area Networks (WPANs)", IEEE Computer Society, Sep. 2003

[34] C.Bisdikian, "An Overview of Bluetooth Wireless Technology", IEEE Communications Magazine, Dec. 2001, page 86-94

[35] IEEE std 802.11, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", IEEE Communication Society, 1999

[36] G. V. Zaruba and S. K. Das, "Off-the-Shelf Enablers of Ad Hoc Networks", Mobile Ad Hoc Networking, Wiley-IEEE Press, July 2004

[37] J.L. Burbank and W.T. Kash, "IEEE 802.16 broadband wireless technology and its application to the military problem space", IEEE MILCOM 2005, 17-20 Oct. 2005 Page(s):1905 - 1911 Vol. 3

[38] M. Sherman, et al., "A PMP-Friendly MANET Networking Approach for WiMAX/IEEE 802.16", IEEE Military Communications Conference 2006, 23-25 Oct. 2006 Page(s):1 – 7

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

[40] T.Clausen, Ed. and P.Jacquet, Ed., "Optimized Link State Protocol (OLSR)", RFC 3626, Oct. 2003

[41] R.Ofier, F.Templin, M. Lewis, "Topology Dissemination Based on Reserve-Path Forwarding (TBRPF)", RFC 3684, Feb. 2004.

[42] D. Johnson,Y. Hu and D. Maltz,"The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4", RFC 4728, Feb. 2007

[43] J. Y. Yu and P. H. J. Chong, "A Survey of Clustering Schemes for Mobile Ad Hoc Networks", IEEE Communications Surveys and Turorials, First Quarter 2005V.

[44] Hwa-Chun Lin, Yung-Hua Chu, "A clustering technique for large multihop mobile wireless networks", Vehicular Technology Conference Proceedings, 2000 IEEE 51st, Volume: 2, Page (s): 15-18, May 2000

[45] Chien-Chung Shen et al., "CLTC: a cluster-based topology control for ad hoc networks", Mobile Computing, IEEE Transactions on, Volume: 3, Issue: 1, Jan.-March 2004, Page (s): 18-32

[46] W. Chen, N. Jain and S. Singh, "ANMP: Ad Hoc Nework Management Protocol", IEEE Joural on Selected Areas in Communications, Vol. 17, No. 8, August 1999

[47] Ram Ramanathan, Martha Steenstrup, "Hierarchically -organized, multi-hop mobile wireless networks for quality-of-service support" Mobile Networks and Applications 3, 1998, Volume 3, Issue 1, Page (s): 101-119

[48] M. Chatterjee, S. K. Das and D. Turgut, "WCA: A Weighted Clustering Algorithm for Mobile Ad hoc Networks", Cluster Computing, Springer Netherlands, Volume 5, Number 2, April 2002, Page(s): 193-204

[49] C.-C. Ciang et al., "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel", Proc. IEEE Singapore International Conference on Networks' 97, 1997

[50] Srivastava and M. Motani, "Cross-Layer Design: A Survey and the Road Ahead", IEEE Communications Magazine, Volume 43, Issue 12, Page (s): 112-119, December 2005.

[51] R. Braden, T. Faber and M. Handley, "From Protocol Stack to Protocol Heap – Role-Based Architecture", Pro. Hot Topics in Net. Princeton, MJ, Mar. 2002

[52] Q. Wang and M.A. Abu-Rgheff, "Cross-Layer Signaling for Next-Generation Wireless Systems", Proc. IEEE WCNC, Volume 2, Page (s): 1084-1089, March, 2003

[53] M. Conti, G. Maselli, G. Turi and S. Giordano, "Cross-Layering in Mobile Ad Hoc Network Design", IEEE Computer, Volume 37, Issue 2, Feb 2004, Page:48 - 51

[54] E. Borgia, M. Conti, F. Delmastro and E. Gregori, "Experimental Comparison of Routing and Middleware solutions for Mobile Ad Hoc Networks: Legacy vs Cross-Layer Approach", ACM SIGCOMM' 05 Workshop, August 2005

[55] R. Winter, J.H. Schiller, N. Nikaein and C. Bonnet, "CrossTalk: Cross-Layer Decision Support Based on Global Knowledge", IEEE Communications Magazine, January 2006, Page(s): 93-99

[56] R.T. Raisinghani and S. Lyer, "Cross-Layer Feedback Architecture for Mobile Device Protocol Stacks", IEEE Communications Magazine, January 2006, Page(s): 85-92

[57] L. XU and B. Zhang, "Study on Cross-layer Design and Power Conservation in Ad Hoc Networks", IEEE PDCAT'2003, August 2003, Pages: 324-328

[58] M. Madueno and J. Vidal, "Joint PHY-MAC Layer Design of the Broadcast Protocol in ad-hoc networks", IEEE ICASSP, 2004, Pages: 557-560

[59] M. Tarique, K. E. Tepe and M. Naserian, "A Cross-Layer Design for Passive Forwarding Node Selection in Wireless Ad Hoc Networks", IEEE International Conference on Wireless Networks, Communications and Mobile Computing, Vol.1, Pages: 82-87, 2005

[60] Li Li and Louise Lamont, "A Lightweight Service Discovery Mechanism for Mobile Ad Hoc Pervasive Environment Using Cross-layer Design", IEEE PerCom. 2005, Pages: 55-59

[61] E. Setton et al., "Cross-layer Design of Ad hoc Networks for Real-time Video Streaming", IEEE Wireless Communications, Volume 12, Issue 4, Pages: 59-65, August 2005

[62] V. Kawadia and P.R. Kumar, "A Cautionary Perspective on Cross-Layer Design", IEEE Wireless Communications, Volume 12, Issue 1, Pages: 3-11, February 2005

[63] Dommel, H. and Aceves, J., "Floor Control for Multimedia Conferencing and Collaboration", ACM Multimedia Systems Magazine, 1997, Vol. 5, No. 1, pp. 23-38.

[64] Schulzrinne, H. and Rosenberg, J., "Signaling for Internet Telephony", Proc. 6th International Conference on Network Protocols, IEEE press style, IEEE computer society, 1998, pp. 298-307.

[65] J. Rosenberg, "A Framework for Conferencing with the Session Initiation Protocol (SIP)", IETF RFC 4353, February 2006

[66] ITU-T Recommendation T.120, "Data Protocols for Multimedia Conferencing", ITU-T, July, 1997

[67] E. M. Schooler and S. L. Casner, "An Architecture for Multimedia Connection Management", Proceedings IEEE 4th Comsoc International Workshop on Multimedia Communications, MM'92, Page(s) 271-274, Monterey

[68] S. Shenker, A Weinrib, and E. Schooler, "Managing Shared Ephemeral Teleconferencing State: Policy and Mechanism", IETF Internet Draft, July 1995

[69] M. Handley, I. Wakeman, and J. Crowcroft, The Conference Control Channel Protocol (CCCP): A scalable base for building conference control applications, In Proceedings of ACM SIGCOMM'95, Boston, USA, August 1995

[70] C. Bormann, J. Ott, C. Reichert, "Simple Conference Control Protocol", IETF Internet draft, December 1996

[71] O. Novo, G. Gamarillo, D. Morgan and R. Even, "A Common Conference Information Data Model for Centralized Conferencing (XCON)", IETF Internet Draft, April, 2006, Expires: October, 2006

[72] H. Khartabil, P.Koskelainen, and A. Niemi, "The conference policy control protocol," IETF, Internet draft < draft-ietf-xcon-cpcp-01 >, 2004

[73] H.323 series , ITU-T recommendations, Geneva 2003

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

[75] H. Liu and P. Mouchtaris, "Voice Over IP Signaling: H.323 and Beyond", IEEE Communications Magazine, October 2000, Vol. 38 No10, pp. 142-148

[76] H. Schulzrinne and J. Rosenberg, "The Session Initiation Protocol: Internet Centric Signaling", IEEE Communications Magazine, October 2000, Vol. 38 No10, pp. 134-141

[77] H. Schulzrinne et. al., "RTP: A Transport Protocol for Real-Time Applications", IETF RFC3550, July, 2003

[78] H. Schulzrinne and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", IETF RFC 3551, July, 2003

[79] C. Groves el al., "Gateway Control Protocol Version 1", IETF RFC 3525, June, 2003

[80] H.248.1, "Gateway Control Protocol: Version 3", ITU-T, Sep., 2005

[81] 3GPP TS 24.147, "Conferencing using the IP Multimedia (IM) Core Network (CN) subsystem", June 2005

[82] Mark Kelley, "Distributed Multipoint Conferences using SIP", IETF Internet Draft, March 8, 2000, <draft-mark-sip-dmcs-00>

[83] Khlifi, H., Agarwal, A., and Gregoire, J.-C., "A framework to use SIP in ad-hoc networks", Electrical and Computer Engineering, IEEE CCECE 2003, Canadian Conference on 4-7 May 2003, Page(s):985 - 988 vol.2

[84] P. Koskelainen, H. Schulzrinne, and X. Wu, "A SIP-based Conference Control Framework", ACM Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video, May 2002, Pages: 53-61

[85] Uyar, A., Wu, W., Bulut, H. and Fox, G., "Service-oriented architecture for a scalable videoconferencing system", IEEE International Conference on Pervasive Services '05 on 11-14 July 2005, Page(s): 445 – 448

[86] Helen J. Wang, et al., "Iceberg: An Internet Core Network Architecture for Integrated Communications", IEEE Personal Communications, Volume 7, Issue 4, Aug. 2000, Page(s): 10 – 19

[87] M. Jiang, J. Li and Y.C. Tay, "Cluster Based Routing Protocol (CBRP)", IETF Internet Draft, 14 August 1999

[88] Li Li and Louise Lamont, "A Lightweight Service Discovery Mechanism for Mobile Ad Hoc Pervasive Environment Using Cross-layer Design", IEEE PerCom 2005 Workshops, Third IEEE International Conference on 8-12 March 2005, Page(s): 55 - 59

[89] S. Helal, N.Desai, V. Verma, and C. Lee, "Konark—A Service Discovery and Delivery Protocol for Ad-Hoc Networks", IEEE WCNC 2003 on 16-20 March 2003, Page(s): 2107 - 2113 vol.3

[90] Ying Yu, Agarwal, A., "A SIP-based multicast framework in MANET", IEEE WiMob, Volume 3, 22-24, Aug. 2005, Page(s): 229-236

[91] Tai, A.T., Tso, K.S. and Sanders, W.H., "Cluster-based failure detection service for large-scale ad hoc wireless network applications", IEEE Dependable Systems and Networks International Conference 2004, Page(s):805 – 814

[92] Gouda, M.G. and McGuire, T.M., "Accelerated heartbeat protocols", IEEE Distributed Computing Systems International Conference 1998, Page(s):202 – 209

[93] Paul Stelling et al., "A fault detection service for wide area distributed computations", SpringerLink, Cluster Computing, Volume 2, Number 2 / September, 1999. Pages: 117-128

[94] Hayashibara, N.; Cherif and A.; Katayama, T., "Failure detectors for large-scale distributed systems", IEEE Symposium on Reliable Distributed Systems, 2002. Page(s):404 – 409

[95] Bertier, M. Marin, O. and Sens, P., "Implementation and performance evaluation of an adaptable failure detector", IEEE Dependable Systems and Networks International Conference 2002, Page(s):354 – 363

[96] Szu-Chi Wang and Sy-Yen Kuo, "Communication strategies for heartbeat-style failure detectors in wireless ad hoc networks", IEEE Dependable Systems and Networks International Conference 2003, Page(s):361 – 370

[97] O. Ratsimor et al., "Allia: alliance-based service discovery for ad-hoc environments", ACM International Workshop on Mobile Commerce, 2002, Page(s): 1-9

[98] J. Rosenberg, "A Hitchhiker's Guide to the Session Initiation Protocol (SIP)", IETF Internet Draft <draft-ietf-sip-hitchhikers-guide-04>, November 2007

[99] J. Rosenberg and H. Schulzrinne, "Guidelines for Authors of Extensions to the Session Initiation Protocol (SIP)", IETF Internet draft <draft-ietf-sip-guidelines-09>, Feb., 2005

[100] R. Sparks, "The Session Initiation Protocol (SIP) Refer Method", IETF RFC 3515, April 2003

[101] S. Donovan, "The Session Initiation Protocol (SIP) INFO Method", IETF RFC 2976, October 2000

[102] A. B. Roach "Session Initiation Protocol (SIP) –Specific Event Notification", IETF RFC 3265, June 2002

[103] N. Banerjee, A. Acharya, S. K. Das, "Peer-to-peer SIP-Based Services over Wireless Ad Hoc Networks", BROADNETs 2004, Workshop on Broadband Wireless Multimedia

[104] Simone Leggio, Jukka Manner, Antti Hulkkonen and Kimmo Raatikainen: "Session Initiation Protocol Deployment in Ad-Hoc Networks: a Decentralized Approach", In proceedings of the International Workshop on Wireless Ad-Hoc Networks (IWWAN2005), May 23 - 26, 2005, London, UK.

[105] JXTA, on line at https://jxta.dev.java.net/

[106] David A. Bryan, Bruce B. Lowekamp, and Cullen Jennings, "A P2P Approach to SIP Registration and Resource Location", IETF Draft, Oct. 2006. The online link can be found at: http://www.p2psip.org/ietf.php

[107] Stoica et al., "Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications", IEEE/ACM Transactions on Networking, Vol. 11, No. 1, pp. 17-32, February 2003

[108]A. Rowstron and P. Druschel, “Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems”, IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany, pages 329-350, November, 2001

[109]J. Rosenberg, H. Schulzrinne and O. Levin, “A Session Initiation Protocol (SIP) Event Package for Conference State”, IETF RFC 4575, August 2006

[110]JAIN SIP Sources online at: https://jain-sip.dev.java.net/

[111]JSR-000032 JAIN SIP Specification (final release), online at: http://jcp.org/aboutJava/communityprocess/final/jsr032/

[112]Bisignano, M., Calvagna, A.; Modica, G.D. and Tomarchio, O., “Expeerience: A Jxta middleware for mobile ad-hoc networks”, Proceedings of the Third International Conference on Peer-to-Peer Computing, IEEE 2003, Page(s):214-215

[113]V. Park and M. S. Corson, “A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks”, Proc. IEEE INFOCOM ‘97, Kobe, Japan (1997), Volume 3, Pages: 1405-1413

[114]H. Lofthouse, M. Yates and R. Stretch, “Parlay X Web Services”, BT Technology Journal, Volume 22, Number 1, January 2004 , pp. 81-86(6), published by Springer

[115]G. Byrne and D. Barber, “Developing multiparty conferencing services for the NGN: towards a service creation framework”, Proceedings of the ISICT, Las Vegas, Nevada, USA, June 16-18, 2004. ACM International Conference Proceeding Series 90, Page(s): 50-55

[116]D. Ben Kheder, R. Glitho, and R. Dssouli, “Media Handling Aspects of Conferencing in Broadband Wireless Ad Hoc Networks”, IEEE Network, Volume 20, Issue 2, March-April 2006, Page(s):42 – 49

[117]Y. Cho et al., “Policy-Based Distributed Management Architecture for Large-Scale Enterprise Conferencing Service Using SIP”, IEEE JSAC, VOL. 23, NO. 10, OCTOBER 2005, Page(s): 1934-1949

[118]Dhafer Ben Kheder, Roch Glitho and Rachida Dssouli, “A Megaco Based-Architecture for Controlling Media Mixers When Conferencing in Mobile Ad Hoc Networks”, In the proceedings of the IEEE CCNC’07, January, 2007, Page(s):696 - 700

[119]Extensible Markup Language (XML) 1.0 (Forth Edition), W3C Recommendation, September, 2006, on line at: http://www.w3.org/TR/REC-xml/

[120]J. Rosenberg, "The Extensible Markup Language (XML) Configuration Access Protocol(XCAP)", IETF Internet draft < draft-ietf-simple-xcap-02>, August 2004

[121] H. Fathi, S.S. Chakraborty, and R. Prasad, "Optimization of SIP Session Setup Delay for VoIP in 3G Wireless Networks", IEEE Transactions on Mobile Computing, Volume 5, Issue 9, Sept. 2006, Page(s): 1121-1132

[122]H. Choi and D. Cho, "Fast and Reliable Route Discovery Protocol Considering Mobility in Multihop Cellular Networks", IEEE Vehicle Technology Conference, 2006, Vol. 2 Page(s): 886-890

[123]G. Kannan, S.N. Merchant and U.B.Desai, "Cross Layer Routing for Multihop Cellular Networks", Advanced Information Networking and Applications Workshops, Volume 2, May 2007, Page(s):165 – 170

[124]Marco Conti and Silvia Giordano, "Multihop Ad Hoc Networking: The Reality", IEEE Communications Magazine, April 2007, Page(s):88-102

[125]B. Lamparter, K. Paul and D. Westhoff , "Charging support for ad hoc stub networks", Computer Communications, Volume 26, Issue 13, 15 August 2003, Pages 1504-1514

[126]C. Makaya and S. Pierre, "An Architecture for Seamless Mobility Support in IP-Based Next-Generation Wireless Networks", IEEE Transactions on Vehicular Technology, Accepted for future publication, Volume PP, Issue 99, 2007

[127]H. Miranda, and L. Rodrigues, "Friends and foes: preventing selfishness in open mobile ad hoc networks", IEEE ICDCSW, May 2003 Page(s):440 – 445

[128]M. Elkins, "MIME Security with Pretty Good Privacy (PGP)", IETF RFC 2015, 1996

[129]T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", IETF RFC 4346, 2006

[130]S. Kent and R. Atkinson, "Security Architecture for the Internet Protocol", IETF RFC 2401,1998

[131]H. Yang, H. Luo, F. Ye, S. Lu and L. Zhang, “Security in mobile ad hoc networks: challenges and solutions”, IEEE Wireless Communications, Volume 11, Issue 1, February 2004, Page(s): 38- 47