CITY UNIVERSITY OF HONG KONG
香港城市大學

3G AND INTERNET PROTOCOLS INTEGRATION
3G 與互聯網協議整合

Submitted to
Department of Computer Science
電腦科學系
in Partial Fulfillment of the Requirements
for the Degree of Master of Philosophy
哲學碩士學位

By

TSO Fung Po, Posco
曹鳳波

September 2007
二零零七年九月

DECLARATION

I recognize that the copyright of this thesis rests with the author or the university to which it was submitted, and that no quotation from it or information derived from it may be published without the prior consent of the author or university (as it may be appropriate).

TSO Fung Po, Posco

Department of Computer Science

City University of Hong Kong;

83 Tat Chee Avenue, Kowloon Tong

Hong Kong SAR

ABSTACT

In the Next Generation Network (NGN), all networks are speculatively expected to be converged to create a ubiquitous communicating environment that will enable one to avail seamless multimedia communication from any internet connectivity enabled device to another in any network. In this convergence, gateway is the key element in the network to enable ubiquitous communication. This thesis discusses about the special effort for the design and implementation of a unique PC-to-3G video call gateway for universal multimedia communication along with its supporting soft reduced-SIP (Session Initiation Protocol) client. In 3G circuit switched network, 3G-324M umbrella protocol stack has been adopted by 3GPP for supporting video conversation. 3G-324M consists of two core components ITU-T H.245 (for call control) and ITU-T H.223 (for packet multiplexing/demultiplexing). Meanwhile, SIP is the commonly used session initiation protocol for multimedia communication in IP network performing similar functions as H.245.

The thesis particularly aims at designing a 3G-IP gateway which is robust and efficient enough to support both IP and 3G technologies and to handle incredibly high call traffic. Although 3G-324M protocol stack is sufficiently efficient enough for handling a single call session, translating H.245 messages into PER binary stream (and vice versa) and performing media packets multiplexing and demultiplexing consume quite a huge amount of system resources if 3G-324M is merged in a gateway environment (for handling many calls concurrently). These avoidable operations decrease the whole system performance and create a performance bottleneck and also leave rooms for researchers to solve these problems. In order to fulfill the requirements of building a highly robust and efficient gateway and overcoming the mentioned performance bottleneck, some innovative ideas have been proposed and analyzed in the thesis. These ideas are briefly introduced as follow:

- A novel approach uses precompile-store-lookup process based on the nature of H.245 message is proposed for H.245 call control module so as to shorten the call setup time and to reduce the demand of system resources for each call session.

- An efficient direct data dispatching is proposed for H.223 multiplexing/demultiplexing procedure to speed up the message multiplexing and demultiplexing.

- Reduced-SIP is proposed to enable a rapid development of simple SIP-capable communication tools. By integrating the reduced-SIP to the gateway, cross 3G-324M and SIP communication become possible.

The efficient 3G-IP gateway has been successfully designed and tested and is also robust enough to handle ONE Million calls without any problem. It is also experimentally verified that the gateway provides the feature of invariant call setup time (typically 5 to 7s for 3G video call initiation) while the conversation quality and after all the QoS is claimed to be guaranteed. In the meantime, the SIP-based soft client also gives a good interoperability through a series of intensive experimental tests.

Although the whole system is still under development, the demo system has gained a plenty of positive feedbacks in ITU TELECOM WORLD 2006 in Hong Kong. It is widely believed that, by launching this system to the market, the revenue of service provider will increase by providing handy, creative multimedia services (e.g. mobile TV, video tour guide, etc.) to end mobile users.

ACKNOWLEDGMENTS

First of all, I'd like to take the chance to express my great appreciation to my supervisor Dr. Weijia Jia for his kind guidance and help on my study and research in City University of Hong Kong. His valuable experience encouraged me a lot about how to work in the research field. His great knowledge and creative ideas inspired me a lot when I was dealing with difficulties during my research study.

I also would like to thank my MPhil qualify members Prof. Deng Xiaotie and Dr. Wong Shek Duncan, for their guidance for my MPhil study.

Also, this project is a team work and is successfully carried out with help of our whole research and development team. So I acknowledge deeply to all of our group members.

Thanks to my parents for their constant care and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1. INTRODUCTION

The penetration rate of third generation mobile service (3G) has increased explosively worldwide in recent years and as per the reports at 3GSM Congress in Barcelona (Feb 2007), the total number of WCDMA and 3G-CDMA subscribers have hit 104M and 325M respectively [1,2] by Dec 2006. Also in this report [1], it is clearly noted that the WCDMA subscriber sector is particularly growing very fast, i.e., 102% YoY and 16.6% QoQ. This rapid spread of 3G is because of its multi-dimensional features like mobile TV, virtual personal assistance, real time road traffic information and mobile bating etc along with the basic requirement of conversational video calling. Starting from the simple voice call to all these high end mobile services lead to high traffic and eventually the 3G gateway faces the challenge of handling incredibly large number of calls at the same time so as to ensure a good QoS to all the subscribers. Otherwise the gateway faces not only the loss of QoS, but also a severe hazard of performance bottleneck.

The 3rd Generation Partnership Project (3GPP) [3] has adopted the H.324M [4] with some modifications in codec and error handling requirements to create 3G-324M [5] standard for 3G wireless networks. 3G-324M protocol stack consists of a control and signaling channel for call setup and session control during a call and a multiplexing/demultiplexing protocol for correct delivery of control/audio/video data. H.245 [6] recommendation is the specified control and signaling protocol in 3G-324M and it provides a way for informing the remote terminal of the capabilities and limitations of the local terminal and opening the audio/video logical channel and so fourth through a set of predefined procedures during the call setup phase, once a call session has been established, commands and indications may be used to manage and terminate a call session. H.223 [7] recommendation is the specified multiplexing and demultiplexing protocol for 3G-324M. This recommendation particularly regulates the way of packet encapsulation, CRC calculation as well as flag detection according to the chosen error protection level during a call session.

1.1 Motivations

In the Next Generation Network (NGN), all networks are speculatively expected to be converged to create a ubiquitous communicating environment that will enable one to avail seamless multimedia communication from any internet connectivity enabled devices to another in any network. In this convergence, gateway is the key element in the network to enable ubiquitous communication since it acts as a bridge connecting different networks together. This thesis discusses about the special effort for the design and implementation of a unique PC-to-3G video call gateway for universal multimedia communication along with its supporting soft reduced-SIP (Session Initiation Protocol) client.

The 3G video call gateway uses 3G-324M and SIP as two core protocols so as to interface to both 3G circuit switched network and IP network respectively. 3G-324M protocol stack has already been implemented in all kinds of 3G enabled handsets and also the fact shows that it is sufficiently efficient enough for handling a single call session. However, major operations of translating H.245 messages into PER binary stream (and vice versa) and performing media packets multiplexing/demultiplexing consume incredibly a huge amount of system resources if 3G-324M is migrated in a gateway environment (for handling many calls concurrently). These avoidable operations decrease the whole system performance and create a performance bottleneck. There is no doubt that some targeted optimizations of 3G-324M protocol implementation are definitely needed to develop a multi-session-wise highly efficient gateway. This thesis particularly aims at designing a 3G-IP gateway which is robust and efficient enough to support both IP and 3G technologies and to handle incredibly high call traffic. In order to fulfill the requirements of building such a gateway and overcoming the mentioned performance bottleneck, some innovative ideas have been proposed and analyzed in the thesis.

1.2 Contributions

These proposed methods will be discussed in detail in following chapters in this thesis but here mention them briefly: First of all, a novel approach uses precompile-store-lookup process base on the nature of H.245 message is proposed for H.245 call control module so as to shorten the call setup time and to reduce the demand of system resources for each call session. Secondly, separation of H.245 Signaling Entities (SEs, which is the core components for H.245 and are used to communicate with peer SEs in the remote terminal using H.245 messages to execute the procedures) state management is proposed to make system sufficiently scalable to manage multiple call sessions. After that, an efficient direct message dispatching is proposed for H.223 multiplexing/demultiplexing procedure to speed up the message multiplexing and demultiplexing. Finally, an IM client is built on top of the proposed reduced-SIP showing that it enables a rapid development of simple SIP-capable communication tools. The efficient 3G-IP gateway has been successfully designed and tested and is also robust enough to handle one million calls without any problem. It is also experimentally verified that the gateway provides the feature of invariant call setup time while the conversation quality and after all the QoS is claimed to be guaranteed. In the meantime, the SIP-based soft client also gives a good interoperability through a series of intensive experimental tests.

Although the whole system is still under development, the demo system has gained a plenty of positive feedbacks in ITU TELECOM WORLD 2006 and ICT Expo 2007 in Hong Kong. It is widely believed that, by launching this system to the market, the revenue of service provider will increase by providing handy, creative multimedia services (e.g. mobile TV, video tour guide, etc.) to end mobile users.

1.3 Publications

Through the design and implementation of the PC-to-3G video call gateway, I have published following research paper:

1. Bo Han, **Fung Po Tso**, Lidong Ling, Weijia Jia, “Performance Evaluation of Scheduling in IEEE 802.16 Based Wireless Mesh Networks”, Mobile Adhoc and Sensor Sysetems (MASS), 2006 IEEE International Conference on Mobile Ad-hoc and Sensor Systems Oct. 2006, Page(s):789 – 794

2. Weijia Jia, **Fung Po Tso**, and Lizhou Zhang, “Efficient 3G-324M protocol Implementation for Low Bit Rate Multipoint Video Conferencing”, JOURNAL OF NETWORKS, Issue : 5, September 2006, Page(s): 1-8

3. Weijia Jia, **Fung Po Tso**, and Lizhou Zhang, “Video Conferencing for 3G Wireless Network”, Handbook on Mobile Ad Hoc and Pervasive Communications (Accepted)

4. **Fung Po TSO**, Lizhuo Zhang, and Weijia Jia, “Video Surveillance Patrol Robot System in 3G, Internet and Sensor Networks”, SenSys 2007 Demo (Accepted)

5. Pranab Sabitru Naik, **Fung Po TSO**, Weijia Jia, “Design of an Efficient and Robust Multimedia Gateway for Pervasive Communication” PERCOM 2008 (submitted)

1.4 Organization

So far, the motivation of this project has been introduced and the contributions have been briefly concluded. The rest of this thesis is organized as follow.

Chapter two introduces the 3G-324M protocol stack. In this chapter, the evolution of 3G-324M will be discussed; besides that, the core components of this protocol stack, H.245 recommendation for call control and signaling and H.223 recommendation for multiplexing, will also be introduced in detail.

Chapter three introduces an efficient H.245 message processing for multiple call handling in 3G Gateway. This algorithm innovatively uses pre-compile method to pre-compile and reuse a set of frequently used messages so as to reduce run time compilation time. This algorithm is experimentally verified to be suitable for high traffic environment.

Chapter four describes an efficient H.223 data dispatching for packet multiplexing/demultiplexing procedure to speed up the message multiplexing and demultiplexing. Packet multiplexing/demultiplexing based on nested multiplex descriptor processing is very time consuming, in order to tackle with this problem, conversion of nested multiplex entry into direct multiplex string is suggested. The experimental data shows a good performance enhancement by using direct multiplex string.

Chapter five firstly introduces a video instant messaging client based on a novel reduced-sip. After that, a PC to 3G video call gateway, the integration of 3G-324M and SIP communication, will be discussed in detail. The system architecture of this gateway will also be described.

Chapter six presents our innovative research about suggestions for next generation multimedia communication model. The model is about the features of clients to be used in the next generation communication network and the abilities of next generation communication gateway.

Chapter seven concludes this thesis and presents some ideas of future work.

Chapter 2. INTRODUCTION TO 3G-324M PROTOCOL STACK

H.324 is an ITU-T recommendation for low bit rate multimedia communication over regular analog phone lines. H.324 recommendation is often referred to as an “umbrella standard” since it specifies procedures defined in other ITU-T recommendations, for examples, it specifies mandatory and optional video and audio codecs, the message to be used for call setup, control and tear-down and the way that audio, video control and other data are multiplexed and demultiplexed. H.324 and its annex C (Multimedia telephone terminals over error prone channels) are defined as H.324M, which is mainly used for mobile terminals. H.324M terminals offering audio communication will support G.723.1 [8] audio codec. Video communication offered in H.324M terminals will support H.263 [9] and H.261 [10] video codecs. H.324M terminals offering multimedia data conferencing should also support T.120 [11][8]. protocol. In addition, other video and audio codecs and data protocols can optionally be used via negotiation through exchange of the H.245 control messages.

3G-324M is derived from H.324M by specifying mandatory audio and video codecs to be used during multimedia conversation. The whole protocol stack of 3G-324M is shown in Figure 1. The differences between 3G-324M and H.324M mainly lie in codecs, e.g. voice by AMR-Adaptive Multi Rate, and video by H.263 or MPEG-4; and error handling requirements (H.223 Annex A and Annex B as mandatory). Therefore, 3G-324M inherits H.324M but must use AMR for speech codec. The AMR speech coder consists of the multi-rate speech coder, a source controlled rate scheme and an error concealment mechanism to combat the effects of transmission errors and lost packets.

2.1 Introduction to H.324

H.324 recommendation describes terminals utilizing V.34 modems (regular 33,600 bit/s) operating over the General Switched Telephone Network (GSTN) for low bit-rate multimedia communication. H.324 terminals usually carry real-time voice, data, and video, or any combination, including video telephony.

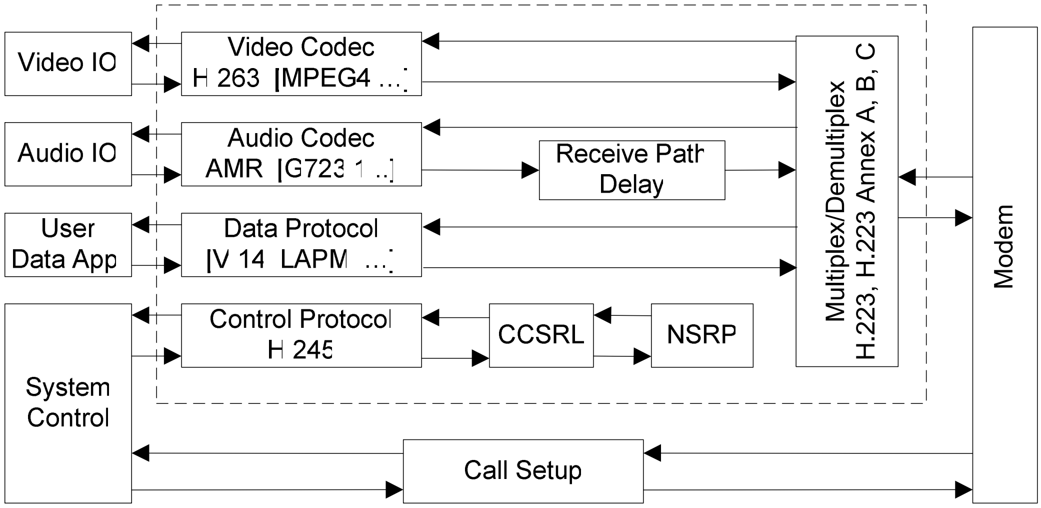


Figure 1: A typical 3G-324M terminal and its components

H.324 is an "umbrella standard". Other ITU-T recommendations in this recommendation include the H.245 control and signaling protocol, H.223 multiplexing/demultiplexing protocol, H.263/H.261 video codec, and G.723.1 audio codec etc. H.324 enabled terminals are allowed to communicate to each other by exchanging their terminal capabilities through procedures defined in H.245. The terminal may reject a communication request from a remote terminal once any inconsistency is detected. For an example, there are two terminals want to setup a video conversation session, terminal A supports only H.263 video codec while terminal B supports H.261 only. This will eventually lead to the video codec inconsistency between terminal A and B and thus video logical channel can not be opened and the call will be rejected.

Mobile communication is always an error-prone communication, therefore, in order to support error-prone low bit rate multimedia communication, annex C is added to H.324 to form H.324M. And later, H.234M is adopted by 3GPP as 3G-324M protocol stack to support video telephony over 3G circuit switched (CS) network.

2.1.1 Functional elements of H.324

A typical H.324 terminal consists of following functional elements working together to provide multimedia communication services:

- The Video codec (H.263 or H.261) carries out redundancy reduction coding and decoding for video streams

- The Audio codec (G.723.1) encodes the audio signal from the microphone for transmission, and decodes the audio code which is output to the speaker. Optional delay in the receiving audio path compensates for the video delay, so as to maintain audio and video synchronization.

- The Data protocols support data applications such as electronic whiteboards, still image transfer, file exchange, database access, audiovisual conferencing, remote device control, network protocols, etc. Other applications and protocols may also be used via H.245 negotiation.

- The Control protocol (H.245) provides end-to-end signaling for proper operation of the H.324 terminal, and signals all other end-to-end system functions including reversion to analogue speech-only telephony mode. It provides for capability exchange, signaling of commands and indications, and messages to open and fully describe the content of logical channels.

- The Multiplex protocol (H.223) multiplexes transmitted video, audio, data and control streams into a single bit stream, and demultiplexes a received bit stream into various multimedia streams. In addition, it performs logical framing, sequence numbering, error detection, and error correction by means of retransmission, as appropriate to each media type.

- The Modem (V.34) converts the H.223 synchronous multiplexed bit stream into an analogue signal that can be transmitted over the GSTN, and converts the received analogue signal into a synchronous bit stream that is sent to the Multiplex/Demultiplex protocol unit.

2.1.2 Multimedia Data Streams in H.324

H.324 functional elements collect and generate data streams according to their roles. These data streams are called multimedia data stream and can be classified as audio streams, video streams, data streams as well as control streams

- Video streams are produced by video codecs such as H.263 or MPEG4. The video streams carrying moving color pictures present the major new feature of a 3G video call. During the video call, the video streams consume a very large part of the whole bandwidth (about 40 Kbps of the 64 Kbps in total). And the bit rates available for video streams can be changed according to the needs of the audio and other channels below.

- Audio streams are generated by audio codecs. They are real-time, but may optionally be delayed in the receiver processing path to maintain lip synchronization with the video streams.

- Data streams may represent still pictures, facsimile, documents, computer files, computer application data, undefined user data, and other data streams.. Standardized data applications include T.120 for real-time audiovisual conferencing, T.84 simple point-point still image file transfer, T.434 simple point-point file transfer, etc

- Control streams pass control commands and indications between remote counterparts. Terminal-to-modem control conforms to ITU-T Recommendation V.250, for terminals using external modems connected by a separate physical interface. Terminal-to-terminal control is according to ITU-T Recommendation H.245.

2.1.3 H.324 Terminal Procedures

H.324 terminals need to go through a series of procedures to create a communication session. The provision of the communication is made in the following steps:

- Phase A: Call set-up of voiceband channel;
- Phase B: Initial analogue telephony communication;
- Phase C: Establishment of digital communication, modem training;
- Phase D: Initialization;
- Phase E: Communication;
- Phase F: End of session;
- Phase G: Supplementary services and call clearing.

Phase A – Call set-up of voice band channel

The calling terminal shall request the connection according to procedures for analogue telephony, according to national standards.

When a call is initiated by a terminal which is external to the modem (a separate physical item connected by an interface) the procedures of ITU-T Rec. V.250 (*ex*-V.25 *ter*) shall be used. Upon successful completion of call set-up, the H.324 terminal shall proceed to phase B.

In our gateway, we innovatively do this by issuing AT commands to a 3G modem. First of all, once the gateway detects any incoming call requests from IP network, it extracts the number of called party (Figure 2) followed by initiating an AT command (Figure 3) to the 3G modem (MERLIN U630) to open a call session.

```
void CallRequest()
{
  PVIM_PACKET_HDR pPacket_Hdr;
  pPacket_Hdr = (PVIM_PACKET_HDR)lParam;
  switch(pPacket_Hdr->packet_type)
  {
          case VIM_STARTPHONECALL_REQ:
           //check to see if server is busy or not
           ……
   }
}
```

Figure 2: Incoming 3G call request from IP network

```
void  Dial(int phone_no)
{
   strcpy(at_cmd, "AT+CBST=134,1,0");
   SendATCmd(at_cmd);
   GetCmdResponse(at_resp);

   sprintf(at_cmd,"ATDT %d",phone_no);
   SendATCmd(at_cmd);
   GetCmdResponse(at_resp);

   while ( strstr(at_resp, "CONNECT") == NULL )
   {
          ……
   }
          ……
}
```

Figure 3: Dial using AT commands

Phase B – Initial analogue telephony communication

A. V.8 procedure

When the procedures of ITU-T Rec. V.8 are in use, phase B shall be bypassed, proceeding directly to phase C.

B. V.8 *bis* procedure

When the procedures of ITU-T Rec. V.8 *bis* are in use, an optional phase B begins when the called party has answered. Phase B is normal analogue telephony voice mode. In this mode users have the opportunity to speak before proceeding to multimedia telephony.

If the terminal is conditioned to go directly into digital communication mode, phase B shall be bypassed, proceeding directly to phase C. If the terminal is conditioned for initial analogue telephony voice mode, the terminal shall proceed to phase C when the user manually causes the terminal to initiate a V.8 *bis* transaction; or the terminal detects an initiation signal from the distant terminal.

Phase C – Establishment of digital communication, modem training

A V.8 procedure

The terminal shall follow the call start-up procedure described in ITU-T Rec. V.8. The calling terminal should not transmit V.8 calling tones CT or CNG, and should transmit calling tone CI. The answer terminal shall support V.8 CM/JM exchanges, and shall transmit answer tone without waiting for call signals. H.324 terminals should signal the "H.324" V.8 call function (value 0x21), and shall not signal a V.8 protocol category.

If the V.8 start-up procedure detects a V.34 modem, the start-up procedure for that modem shall be followed. Upon completion of the modem start-up procedure and establishment of digital communication, the terminal shall proceed to phase D – Initialization.

If the V.8 procedure fails to detect a V.34 modem, or the handshake and the establishment of the digital connection is not successful after a suitable period the calling terminal may, depending on predetermined configuration, go to telephony mode, disconnect the line, or go to another operating mode more suitable for the detected modem. Such other modes are outside the scope of H.324 Recommendation.

The terminal shall wait for a suitable call set-up period, in addition to processing, signal detection, and maximum round-trip delays, before deciding on further action.

B V.8 *bis* procedure

The terminal shall follow the call start-up procedure described in ITU-T Rec. V.8 *bis*. If the V.8 *bis* procedure detects that the distant terminal is not capable of V.8 *bis*, but is capable of V.8, the Phase C procedure for V.8 (above) shall be followed. If the V.8 *bis* procedure detects a distant H.324 terminal supporting the capabilities desired for this call, the V.34 start-up procedure shall be followed.

Upon completion of the V.8 *bis* procedures and establishment of digital communication, the terminal shall proceed to phase D – Initialization.

If the V.8 *bis* procedure fails, results in a return to analogue telephony, or the handshake and the establishment of the digital connection is not successful after the period specified in ITU-T Rec. V.8 *bis*, the calling terminal may, depending on predetermined configuration, go to telephony mode, disconnect the line, or go to another operating mode more suitable for the detected modem. Such other modes are outside the scope of this Recommendation.

Phase D – Initialization

After digital communication has been established, a minimum of 16 HDLC flags shall be transmitted in order to ensure synchronization and to agree on the highest mutually supported flag protection level. Following this, system-to-system communication shall be initiated using the H.245 control channel. Since no multiplex table entries have yet been sent to the receiver, initial control messages shall be sent using multiplex table entry 0.

Terminal system capabilities are exchanged by transmission of the H.245 TerminalCapabilitySet message. This capability PDU shall be the first message sent. The H.245 MasterSlaveDetermination message shall also be sent at this time, in which the terminals exchange random numbers, according to the procedure in ITU-T Rec. H.245, to determine the master and slave terminals. H.324 terminals shall be capable of operating in both master and slave modes, and shall set terminalType to 128 and set statusDeterminationNumber to a random number in the range 0 to $2^{24} - 1$. Only one random number shall be chosen by the terminal for each call, except in the case of identical random numbers, as described in ITU-T Rec. H.245.

If the initial capability exchange or master/slave determination procedures fail, these should be retried at least two additional times before the terminal abandons the connection attempt and proceeds to phase G.

The range of terminalTypes from 0 to 127 is reserved for possible use by MCUs or other non-terminal devices which may need to be slave at all times, and the range 129 to 255 is reserved for possible use by MCUs or other non-terminal devices which may need to be master at all times.

After these procedures are complete, and the far-end's capabilities have been received, the procedures of ITU-T Rec. H.245 may then be used to open logical channels for various information streams. Multiplex table entries may be sent before or after logical channels are opened, but information shall not be transmitted over a logical channel until the channel is open, and an appropriate H.223 multiplex table entry has been defined.

The indication videoIndicateReadyToActivate, "Video Indicate Ready-to-activate", is defined in ITU-T Rec. H.245. Its use is optional, but when used the procedure shall be as follows:

Terminal X has been set so that video is not transmitted unless, and until, the remote terminal has also indicated readiness to transmit video. Terminal X shall send the indication videoIndicateReadyToActivate when the initial capability exchange has been completed, but shall not transmit a video signal until it has received either videoIndicateReadyToActivate or incoming video.

A terminal which has not been set in this optional way is not obliged to wait until receipt of videoIndicateReadyToActivate or video before initiating its video transmission.

Phase E – Communication

During this session, the procedures for changing logical channel attributes, capability, receive mode etc. shall be carried out as defined in ITU-T Rec. H.245.

A. Rate changes and retrains

During phase E communication, the modem may retrain or alter its rate of data transmission, with or without momentary disruption of data transmission and loss of data. Upon any such momentary disruption of data transfer, the terminal shall not restart phase D, but shall remain in phase E and execute the normal H.324 error recovery procedures according to ITU-T Rec. H.223.

B. Involuntary disconnection

Should the terminal detect involuntary, unrecoverable loss of modem communication, or of the underlying GSTN connection, the terminal shall immediately proceed to phase G, analogue telephony mode or line disconnection, bypassing phase F.

Phase F – End of session

Either terminal may initiate the end of the session. The initiating terminal shall use the following procedure:

1. For each logical channel carrying video, it shall stop sending video at the end of a complete picture and then close the logical channel.

2. It shall close all outgoing logical channels carrying data and audio.

3. It shall transmit the H.245 message EndSessionCommand, and then discontinue all H.245 message transmissions. This message shall contain an indication to the far end regarding the mode the terminal will enter after the end of the session (disconnect line, analogue telephony, or other mode).

4) On subsequent receipt of EndSessionCommand from the remote end, it shall proceed to phase G, except that if the initiating terminal indicated an intention to disconnect the line after the end of session, the terminal shall not wait for receipt of EndSessionCommand from the remote end, but shall proceed directly to phase G.

A terminal receiving EndSessionCommand without first having transmitted it shall:

a. if the initiating terminal's EndSessionCommand message indicated "disconnect line", optionally follow step 3 above, then proceed to phase G.

b. otherwise, follow step 3 above, then proceed to phase G. If possible, the responding terminal should proceed to the new mode indicated in the initiating terminal's EndSessionCommand message.

Phase G – Supplementary services and call clearing

If the terminal arrived at phase G by involuntary disconnection, it shall disconnect or revert to analogue telephony, depending on predetermined configuration.

A terminal wishing to terminate a call shall first initiate session end procedure described in phase F.

In phase G, the terminal should proceed as it indicated in the EndSessionCommand message. If it indicated a change to another digital communication mode, it shall begin the new mode at the equivalent of phase D. Otherwise, it shall initiate the cleardown procedures defined in ITU-T Rec. V.34, except that it shall not physically disconnect the GSTN connection if it indicated an intention to revert to analogue telephony mode.

These procedures ensure that:
• the distant terminal does not erroneously invoke a fault procedure;
• the human user gets the right indications via tones and announcements from the network exchange;

• relevant messages can be displayed for the human user by the terminal.

2.2 Introduction to H.245

H.245 provides a number of different services, some of which are expected to be applicable to all terminals that use it, and some are more specific to particular ones. In H.245, these services are named as Signaling Entities (SE). Each SE defines a series of procedures to handle with corresponding messages. Procedures are defined to allow the exchange of audiovisual and data capabilities; to request the transmission of a particular audiovisual and data mode; to manage the logical channels to transport audiovisual and data information; to establish which terminal is the master terminal and which is the slave terminal for the purposes of managing logical channels; to carry various control and indication signals; to control the bit rate of individual logical channels and the whole multiplex; and to measure the round trip delay, from one terminal to the other and back. Figure 4 illustrates the object-oriented implementation of H.245 module.

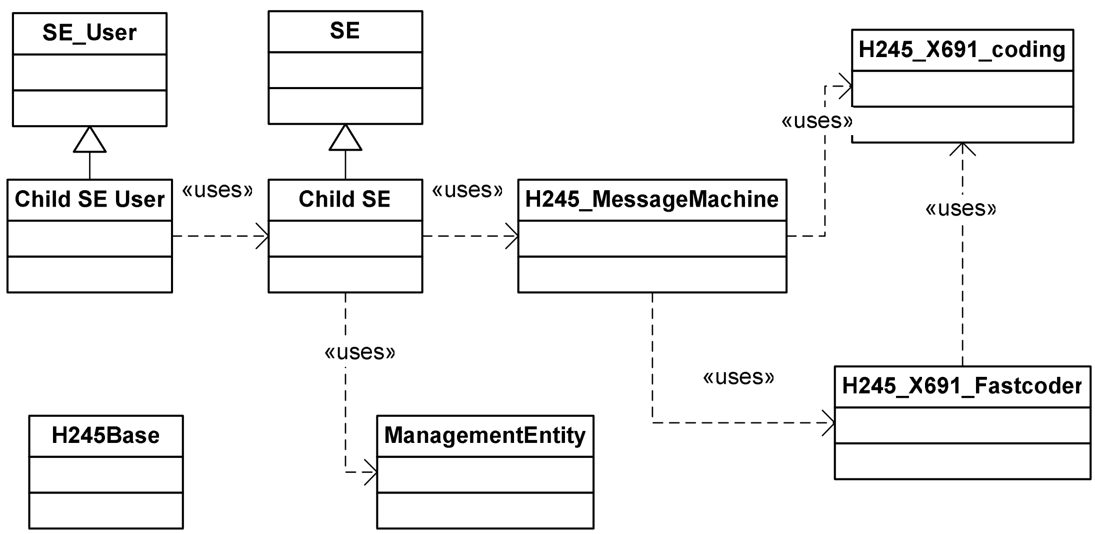


Figure 4 : object oriented implementation of H.245 module

2.2.1 Signaling entities

Signaling entities (SEs) is used to interact with peer (remote) SEs, i.e. SE-SE communication, to keep in step with one another for exchanging request, response, indication and command messages. For each single type of SE, there will be an associated outgoing SE and an incoming SE. To be more specific, the outgoing SE in fact communicates with a peer incoming SE in the remote terminal by using request/response message model. The outgoing (sending) SE sends request messages while incoming SE reacts with response messages. For an example, terminal CapabilityExchangeSignalingEntity (CESE) should be divided into an outgoing CESE and incoming CESE. Local outgoing CESE communicates with remote incoming CESE, meanwhile, local incoming CESE communicates with remote outgoing CESE as illustrated in Figure 5. Figure 5 also shows that the incoming and outgoing CESE with a single terminal are independent to each other, this is because the establishment of channels in each direction is independent.

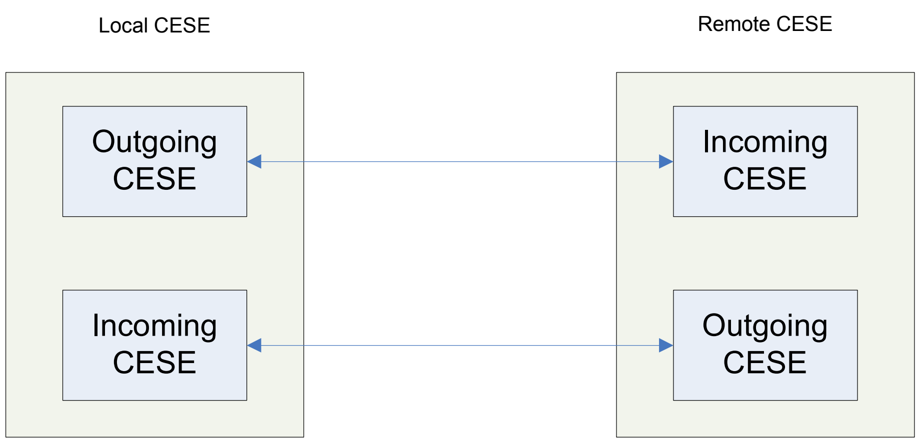


Figure 5: Local CESE interacts with remote CESE

A request message requires that some action to be taken by the peer SE and that the peer SE sends back a response message within a certain time. The response message takes the form of either an acknowledgment message indicating that the SE has failed or refused to take the requested action. There is always a timer associated with the request/response interaction, which is started by the outgoing SE. Be noted that the incoming B-LCSE is the only incoming SE containing a timer. The sending SE may get no response from peer SE before the time expires, it will retry sending the same request N times before giving up. When a procedure can be attempted more that once in case of encountering procedure failure, the SE should also include a counter that is incremented each time a procedure is initiated and compared with a maximum value before the procedure is reattempted. If the maximum attempt number is reached, SE should abort this procedure. Table 1 gives the instance and persistency of each SE for each call session.

Table 1: The instance and persistency of each SE for each call session

Procedure	**Associated SEs**	**Instances**	**Persistent**
Master slave determination	MSDSE	One	No
Capability exchange	Outgoing CESE Incoming CESE	One	No
Unidirectional logical channel signaling	Outgoing LCSE Incoming LCSE	One per unidirectional logical channel	Yes
Bidirectional logical channel signaling	Outgoing B-LCSE Incoming B-LCSE	One per bidirectional logical channel	Yes
Close logical channel	Outgoing CLCSE Incoming CLCSE	One per logical channel	Yes
Multiplex entry exchange	Outgoing MTSE Incoming MTSE	One per multiplex entry	No
Request multiplex entry	Outgoing RMESE Incoming RMESE	One per multiplex entry	No
Mode request	Outgoing MRSE Incoming MRSE	One	No
Round trip delay	RTDSE	One	No
Maintenance loop	Outgoing MLSE Incoming MLSE	One per bidirectional logical channel	Yes

A. Master-slave determination signaling entity

Conflicts may arise when two terminals involved in a call initiate similar events simultaneously and only one such event is possible or desired. For example, conflicts may arise when resources are available for only one occurrence of the event. To resolve the conflict, one terminal shall act as the master and the other terminal shall act as a slave terminal. Rules specify how the master and slave terminal shall respond at times of conflict.

The master-slave determination procedure allows terminals in a call to determine which terminal is the master and which is the slave. The terminal status may be re-determined at any time during a call; however, a terminal may only initiate the master slave determination process if no procedure that depends upon its result is locally active.

B. Capability exchange signaling entity

The capability exchange procedures are intended to ensure that the only multimedia signals to be transmitted are those that can be received and treated appropriately by the receiving terminal. This requires that the capabilities of each terminal to receive and decode be known to the other terminal. It is not necessary that a terminal understands or stores all incoming capabilities; those that are not understood, or can not be used shall be ignored, and no fault shall be considered to have occurred. When a capability is received which contains extensions not understood by the terminal, the capability shall be accepted as if it did not contain the extensions.

The total capability of a terminal to receive and decode various signals is made known to the other terminal by transmission of its capability set.

Receiving capabilities describe the terminal's ability to receive and process incoming information streams. Transmitters shall limit the content of their transmitted information to which the receiver has indicated it is capable of receiving. The absence of a receive capability indicates that the terminal cannot receive (is a transmitter only).

Transmit capabilities describe the terminal's ability to transmit information streams. Transmit capabilities serve to offer receivers a choice of possible modes of operation, so that the receiver may request the mode which it prefers to receive. The absence of a transmit capability indicates that the terminal is not offering a choice of preferred modes to the receiver (but it may still transmit anything within the capability of the receiver).

These capability sets provide for more than one stream of a given medium type to be sent simultaneously. For example, a terminal may declare its ability to receive (or send) two independent H.263 video streams and two independent G.731.1 audio streams at the same time. Capability messages have been defined to allow a terminal to indicate that it does not have fixed capabilities, but that they depend on which other modes are being used simultaneously. For example, it is possible to indicate that higher resolution video can be decoded when a simpler audio algorithm is used; or that either two low resolution video sequences can be decoded or a single high resolution one. It is also possible to indicate trade-offs between the capability to transmit and the capability to receive.

Non-standard capabilities and control messages may be issued using the NonStandardParameter structure. Note that while the meaning of non-standard messages is defined by individual organizations, equipment built by any manufacturer may signal any non-standard message, if the meaning is known.

C. Logical channel signaling entity

An acknowledged protocol is defined for the opening and closing of logical channels which carry the audiovisual and data information. The aim of these procedures is to ensure that a terminal is capable of receiving and decoding the data that will be transmitted on a logical channel at the time the logical channel is opened rather than at the time the first data is transmitted on it; and to ensure that the receiving terminal is ready to receive and decode the data that will be transmitted on the logical channel before that transmission starts. The OpenLogicalChannel message includes a description of the data to be transported, for example, H.262 MP@ML at 6 Mbit/s. Logical channels should only be opened when there is sufficient capability to receive data on all open logical channels simultaneously.

A part of this protocol is concerned with the opening of bi-directional channels. To avoid conflicts, which may arise when two terminals initiate similar events simultaneously, one terminal is defined as the master terminal, and the other as the slave terminal. A protocol is defined to establish which terminal is the master and which is the slave. However, systems that use this

Recommendation may specify the procedure specified in this Recommendation or another means of determining which terminal is the master and which is the slave.

D. Close logical channel request signaling entity

A logical channel is opened and closed from the transmitter side. A mechanism is defined which allows a receiving terminal to request the closure of an incoming logical channel. The transmit terminal may accept or reject the logical channel closure request. A terminal may, for example, use these procedures to request the closure of an incoming logical channel which, for whatever reason, cannot be decoded. These procedures may also be used to request the closure of a bi-directional logical channel by the terminal that did not open the channel.

E. Multiplex table entry modification signaling entity

The H.223 multiplex table associates each octet within an H.223 MUX message with a particular logical channel number. The H.223 multiplex table may have up to 15 entries. A mechanism is provided that allows the transmit terminal to specify and inform the receiver of new H.223 multiplex table entries. A receiving terminal may also request the retransmission of a multiplex table entry.

F. Audiovisual and data mode request signaling entity

When the capability exchange protocol has been completed, both terminals will be aware of each other's capability to transmit and receive as specified in the capability descriptors that have been exchanged. It is not mandatory for a terminal to declare all its capabilities; it need only declares those that it wishes to be used.

A terminal may indicate its capabilities to transmit. A terminal that receives transmission capabilities from the remote terminal may request a particular mode to be transmitted to it. A terminal indicates that it does not want its transmission mode to be controlled by the remote terminal by sending no transmission capabilities.

G. Round trip delay determination signaling entity

It may be useful in some applications to have knowledge of the round trip delay between a transmit terminal and a receiving terminal. A mechanism is provided to measure this round trip delay. This mechanism may also be useful as a means to detect whether the remote terminal is still functioning.

H. Maintenance loops signaling entity

Procedures are specified to establish maintenance loops. It is possible to specify the loop of a single logical channel either as a digital loop or decoded loop, and the loop of the whole multiplex.

I. Commands and indications

Commands and indications are provided for various purposes: video/audio active/inactive signals to inform the user; fast update request for source switching in multipoint applications are some examples. Neither commands nor indications elicit response messages from the remote terminal. Commands force an action at the remote terminal whilst indications merely provide information and do not force any action.

A command is defined to allow the bit rate of logical channels and the whole multiplex to be controlled from the remote terminal. This has a number of purposes: interworking with terminals using multiplexes in which only a finite number of bit rates are available; multi-point applications where the rates from different sources should be matched; and flow control in congested networks.

2.2.2 ASN.1

In H.245, the message syntax is defined using a notation called Abstract Syntax Notation One (ASN.1) [12]. ASN.1 defines a number of simple data types and specifies a notation for referencing these types and for specifying values of these types.

The ASN.1 notations can be applied whenever it is necessary to define the abstract syntax of information without constraining in any way how the information is encoded for transmission. It is particularly, but not exclusively, applicable to application layer protocols. The following is an example of message definition by ASN.1 and the example message is OpenLogicalChannel message.

2.2.3 X.691

ASN.1 deals only with the syntax and semantics of message specifications. The binary encoding of data structures is covered in other Recommendations, such as X.691 [13] (packed encoding rules or PER). BER allows data to be deciphered by systems that have general knowledge of ASN.1 but do not know the details of the specification used to form the data. In other words, the data types are encoded along with the data values. PER is much more efficient since only data values are encoded and the coding is designed with very little redundancy. This method can be used when both the transmitter and the receiver expect data to adhere to a known structure.

H.245 is implemented using the packed encoding rules. Since both sides of a call know that messages will conform to the H.245 specification it is not necessary to encode that specification into the messages. For decoding simplicity, the aligned variant of PER is used. This forces fields that require eight or more bits to be aligned on octet boundaries and to consume an integral number of octets. Alignment is done by padding the data with zeros before large fields.

Since the performance of X.691 message encoding greatly affect the performance of call setup and command message exchange, a special algorithm, which is discussed in chapter 3, has been proposed to enhance the performance of such operation. Therefore, one more setup is added to this procedure to check whether the enhanced procedure is applicable or not, if yes, then use this procedure; if not, then the process can proceed to normal X.691 message encoding procedure. Figure 6 is the program flow deployed in our implementation.

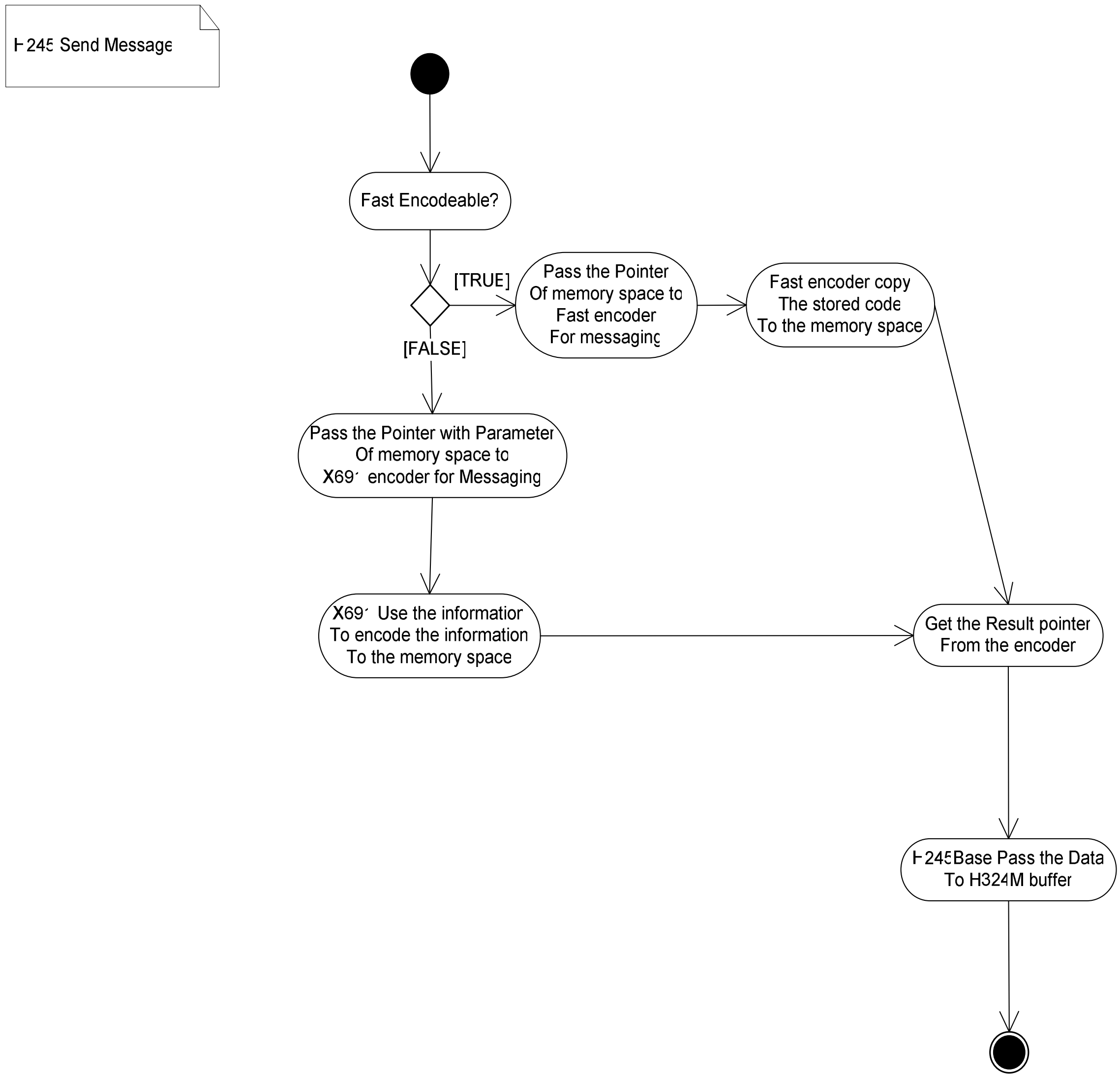


Figure 6: H.245 ASN.1 message processing flow with added enhanced approach

2.2.4 SE user

SE user is the supervisory process that communicates with outgoing and incoming SE via defined primitives in H.245. In other words, this process can be treated as the user who issues commands to SE and SE executes these commands and responses with executed result(s). Traditionally, we refer to the user of the MSDSE as the MSDSE user, and the user of both the outgoing and incoming CESEs as the CESE user and so on so forth.

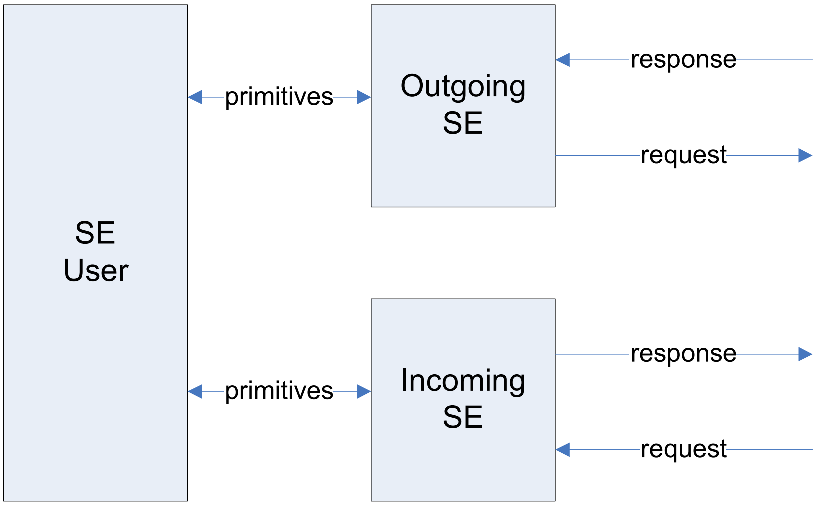


Figure 7: SE User and SE communicates using primitives

H.245 does not specify the process of SE user explicitly but just SEs. However, SE user also plays an important role throughout the whole recommendation. Virtually, SE users are the top level interfaces between application layer and H.245 module. In this 3G-IP video call gateway environment, if layered approach analysis is used, the SE user acts as the translator in between of 3G-324M processes and H.245 SE processes. Once 3G-324M processes want to issue instructions to remote terminal, these instructions are firstly passed to SE user, the SE user then translates these instructions to corresponding primitives to associated SE.

2.2.5 Communication between SE users and SE

As mentioned above, the interaction with the user of a particular function is specified in terms of primitives transferred at the interface between the SE and the SE user. According to the H.245 recommendation, primitives are for the purpose of defining protocol procedures and are not intended to specify or constrain implementation. There may be a number of parameters associated with each primitive as the aid of information exchange.

To assist in the specification, SE states are defined. These states are conceptual and reflect general conditions of the SE in the sequences of primitives exchanged between the SE and the user, and the exchange of messages between the SE and its peer. For each SE the allowed sequence of primitives between the user and the protocol entity is defined using a *state* *transition diagram*. The allowed sequence constrains the actions of the user, and defines the possible responses from the protocol entity. If a primitive parameter described as being null, is equivalent to the parameter not being present. Figure 7 illustrates that primitives are used internally for SE User and SE while SE uses response and request to interact with remote SE

2.3 Introduction to H.223

H.223 is the ITU-T recommendation for data multiplexing protocol. It is the real sending/receiving interface for all the video/audio/data applications during the call session. The recommendation defines the structures and routines for data transmission multiplexing. Therefore, it is also the major focus of investigation about how to improve the multimedia data transmission efficiency of 3G mobile networks.

The H.223 recommendation specifies a packet-wise multiplexing protocol designed for the exchange of one or more information streams between higher-layer entities such as data and control signals and audio and video codecs. Thus, although the actual underlying physical bearer is a circuit-switching connection, the data formats in the description of the recommendation are in the form of packets.

The H.223 module consists of two distinct layers: the Adaptation Layer (AL) and the Multiplex Layer (MUX). The AL is mainly responsible for error protection (eg. CRC calculation) and optional retransmission for lost or corrupt packets. It is actually an interface for upper-layer applications, and it still deals with different sources separately. The MUX layer performs the actual multiplexing. In this layer, data traffic from different sources would be multiplexed into one packet according to the multiplexing patterns defined in the multiplex table.

In layered approach, as shown in Figure 8, each of these layers provides service for the above layers and utilize the service of the layers below, the input packet of a layer is called Service Data Unit (SDU); the output packet is called Protocol Data Unit (PDU).

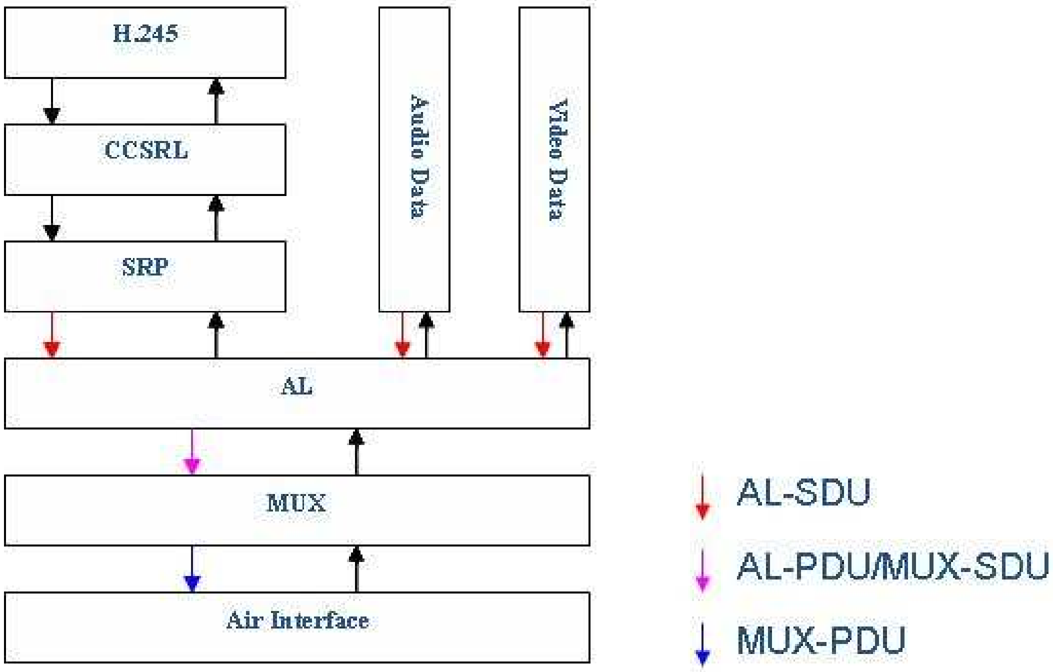


Figure 8: Layered view of 3G-324M protocol stack

Therefore, the unit of information exchanged between the AL and the higher-layer AL users is an AL-SDU. AL-SDUs contain an integer number of octets. The AL adapts the AL-SDUs to the MUX layer below by adding extra bytes for purposes such as error detection, sequence numbering and retransmission. The information unit exchanged between peer AL entities is called an AL-PDU. An AL-PDU is conveyed as one MUX-SDU.

According to different numbers of extra bytes with different functions, the AL can be further divided into the following three more specific types:

- AL1 is designed primarily for the transfer of data or control information. It does not provide any error control, and no extra bytes are added.

- AL2 is designed primarily for the transfer of digital audio. 1 octet for an 8-bit CRC and an optional 1 octet for sequence numbering are added.

- AL3 is designed primarily for the transfer of digital video. 2 octets for a 16-bit CRC and 1 or 2 optional control octets are added.

After the AL-SDUs are transformed into AL-PDUs by the AL, the AL-PDUs are sent to the MUX layer as MUX-SDUs. The MUX layer is responsible for transferring information received from the AL to the far end using the services of an underlying physical layer. MUX-SDUs are transferred by the MUX layer to the far end in one or more variable-length packets called MUX-PDUs.

With different AL layer types, there are also two different types of logical channels. One is the segmentable logical channels. For this type of logical channel, the MUX-SDU can be segmented into different parts, and sent to the remote peer via a number of MUX-PDUs. The other is non-segmentable logical channels. For this kind of logical channel, the MUX-SDU can not be segmented but must be sent to remote peer as a whole via one MUX-PDU.

A MUX-PDU can be generally divided into two parts: the header and the information field. Octets from multiple MUX-SDUs (of multiple logical channels) may be present in a single MUX-PDU information field. The header octet contains a 4-bit Multiplex Code (MC) field which specifies, by reference to a multiplex table entry, the logical channel to which each octet in the information field belongs. Multiplex table entry 0 is permanently assigned to the control channel. Other multiplex table entries are formed by the terminals and are signaled to the far end via the control channel prior to their use. A more detailed discussion about the multiplex table entries will be given in the following sections.

2.3.1 The Adaptation Layer

A. Adaptation Layer Type 1 (AL1)

As the AL designed for the transfer of data or control information, AL1 does not provide any error detection or correction capability. The AL1-SDU is identical to AL1-PDU as it is illustrated in Figure 9. Therefore, any necessary error control operations, such as retransmission procedures, should be provided by the higher layers. e.g., 3G-324M has defined the CCSRL and

SRP layers to provide error detection, sequence numbering and retransmission functions for reliable.

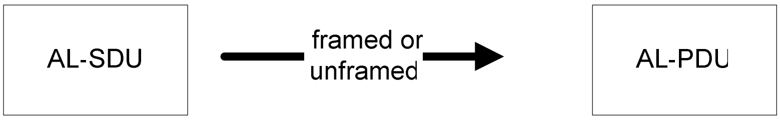


Figure 9: From AL1-SDU to AL2-PDU

AL1 provides two different transfer modes: the framed transfer mode and the unframed transfer mode. In the framed transfer mode, AL1 is used to transfer frames generated by a higher-layer protocol. In this case, frames are first mapped to AL-SDUs and these are then passed by AL1 in MUX-SDUs to the MUX layer. In the unframed mode, AL1 is used to carry an unframed octet sequence. Any internal framing present in the octet sequence is not visible to AL1 which passes the octets received from the higher layer to the MUX layer without paying any attention to framing. Therefore, the logical channel for AL1 unframed mode must be segmentable.

B. Adaptation Layer Type 2 (AL2)

AL2 is designed primarily for the transfer of digital audio. It provides an 8-bit CRC for error detection; AL2 also supports optional sequence numbering which may be used to detect missing and corrupted AL-PDUs. AL2 transfers variable-length AL-SDUs of integral number of octets.

After adding the 8-bit CRC and the optional sequence number (SN), the format of the AL-PDU for AL2 is shown in the Figure 10.

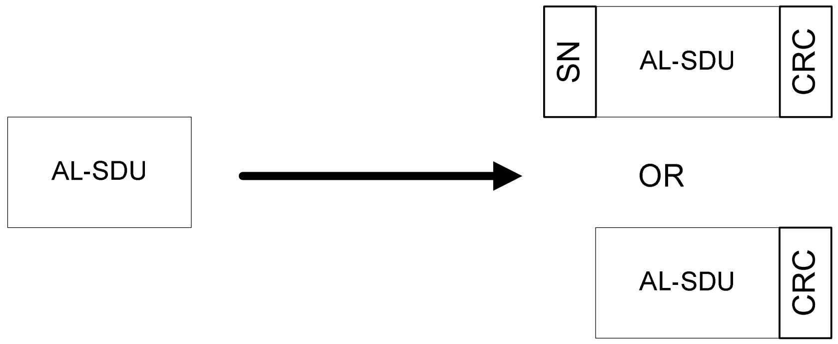


Figure 10: From AL2-SDU to AL2-PDU

The 8-bit CRC provides an error detection capability across the entire AL-PDU. It shall contain the remainder of the division (modulo 2) by the generator polynomial $p(x) = x^8 + x^2 + x + 1$ of the product x^8 multiplied by the content of the AL-PDU, excluding the CRC field, and including the SN field, if it is used. The polynomial representing the content of the AL-PDU is generated using bit number 1 of the first octet as the coefficient of the highest-order term.

The optional 8-bit SN provides a capability for sequencing AL-PDUs. It can be used by the AL2 receiving entity to detect missing and corrupted AL-PDUs. If SN is used, the first AL-PDU transmitted by the AL2 sending entity shall have the SN field set to 0. For each subsequent transmitted AL-PDU which belongs to that logical channel, the value of the SN field shall be incremented by 1 modulo 256.

C. Adaptation Layer Type 3 (AL3)

AL3 is designed primarily for the transfer of digital video. It includes a 16-bit CRC for error-detection, and also supports optional sequence numbering which may be used to detect missing and corrupted AL-PDUs. What's more, designed primarily for video, it provides an optional retransmission procedure.

After adding the 16-bit CRC and the optional 1 to 2 control octets, the format of the AL-PDU for AL3 is shown as the Figure 11

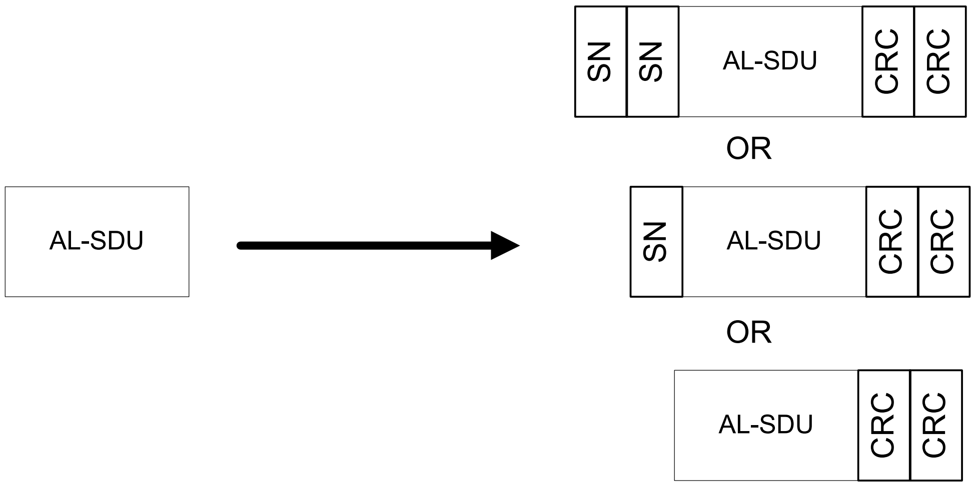


Figure 11: From AL3-SDU to AL3-PDU

The 16-bit CRC provides an error detection capability across the entire AL-PDU, including the control field, if used. It uses the generator polynomial $g(x) = x^{16} + x^{12} + x^5 + 1$. And the CRC field is defined as the complement of the sum (modulo 2) of

a) the remainder of $x^k (x^{15} + x^{14} + x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1)$ divided (modulo 2) by the generator polynomial $g(x) = x^{16} + x^{12} + x^5 + 1$, where k is the number of bits in the AL-PDU, not including the bits in the CRC field; and

b) the remainder of the division (modulo 2) by the generator polynomial $g(x) = x^{16} + x^{12} + x^5 + 1$, of the product of x^{16} multiplied by the content of the AL-PDU, excluding the bits in the CRC field. The polynomial representing the content of the AL-PDU is generated using bit number 1 of the first octet as the coefficient of the highest-order term.

In contrast to the CRC procedure used for the 8-bit CRC in AL2, the CRC procedure used here includes pre- and post-conditioning.

The optional control field consists of a Payload Type (PT) field, which indicates the function of the AL-PDU payload, and a Sequence Number (SN) field, as illustrated in Figure 12.

<table>
<tr><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th></tr>
<tr><td colspan="7">SN</td><td>PT</td></tr>
<tr><td colspan="8">SN</td></tr>
</table>

PT Payload Type
SN Sequence Number

Figure 12: Format of AL3-PDU control octets

The AL-PDU for AL3 can have 0, 1 or 2 octet control field. The actual number of octets in the control field is determined by the terminal and shall be signaled to the remote terminal in the H.245 OpenLogicalChannel message.

The 1-bit PT field indicates the payload type of the AL-PDU. When the PT field is set to "1", the AL-PDU payload field shall contain an AL-SDU. Such an AL-PDU is referred to as an I-PDU. When the PT field is set to "0", the AL-PDU payload field shall contain a supervisory message used in the retransmission procedure. Such an AL-PDU is referred to as an S-PDU.

The SN field is 7 or 15 bits, depending on the length of the control field. In I-PDUs, the SN field shall contain a send sequence number N(S). In S-PDUs, the SN field shall contain a receive sequence number N(R) of an I-PDU. Using the SN field, the AL3 receiver may detect that an AL-PDU is missing or has been corrupted by the MUX layer. The AL3 receiver should discard any corrupted AL-PDUs that it detects.

D. Primitives between AL and AL users

The information exchanged between AL and the AL user includes the following primitives:

1. AL-DATA.request: This primitive is issued by an AL user to AL to request the transfer of an AL-SDU to a corresponding AL user.

2. AL-DATA.indication: This primitive is issued to an AL user by AL to indicate the arrival of an AL-SDU.

3. AL-Abort.abort: This primitive is issued to AL by an AL user to signal that a partially delivered AL-SDU is to be aborted.

4. AL-DRTX.indication: This primitive is optionally used in AL3. This primitive is issued to an AL3 user by AL3 to indicate that a declined retransmission condition has occurred in the local transmitter. As the AL3 is designed to provide optional retransmission capabilities for the video packets, the receiver can send the Selective Reject (SREJ) message to request the sender to retransmit one of the sent AL-PDUs. However, in some situations (the sent AL-PDU has already been deleted from the sending buffer), the sender might be unable to perform the retransmission operation. Thus, the sender has to send the Declined Retransmission (DRTX) message to the receiver; meanwhile the AL-DRTX.indication is performed to indicate the AL3 user so that some operations can be done by the AL3 user to make up for the declined retransmission.

2.3.2 The Multiplex Layer

After the operations of the AL layer, the MUX layer provides the capabilities to transfer MUX-SDUs from the sending AL to the receiving AL using the services of a physical layer below. Data contents from different MUX-SDUs from different logical channels are multiplexed into one or several MUX-PDUs according to the multiplex patterns defined in the multiplex tables.

The MUX layer may interface with one or more AL layers. The information exchanged between the MUX layer and each individual AL includes the following primitives:

1. MUX-DATA.request: This primitive is issued to the MUX layer by an AL sending entity to request the transfer of a MUX-SDU to the corresponding receiving entity.

2. MUX-DATA.indication: This primitive is issued by the MUX layer to an AL receiving entity to indicate the arrival of a MUX-SDU from the corresponding sending entity.

3. MUX-Abort.request: This primitive is issued to the MUX layer by an AL sending entity to signal that a partially delivered MUX-SDU is to be discarded. This primitive may be used by all AL types.

4. MUX-Abort.indication: This primitive is issued by the MUX layer to an AL receiving entity to signal that a partially delivered MUX-SDU is to be discarded.

A. Mobile Level 0

At level 0, the MUX-PDUs are preceded and followed by the HDLC flags 0x7e ("01111110"). The flag preceding the MUX-PDU is defined as the opening flag. The flag following the MUX-PDU is defined as the closing flag. The closing flag may also serve as the opening flag of the next MUX-PDU. Flags can also be transmitted repetitively between MUX-PDUs. Meanwhile, for level 0, the MUX layer is also responsible to keep transparency of the HDLC flags in MUX-PDUs. Thus, the MUX layer has to examine the data contents of the PDU. At the sending entity, it has to insert a "0" bit after all sequences of five contiguous "1" bits to ensure that a flag is not simulated within the MUX-PDU. At the receiving entity, it has to discard any "0" bit which directly follows five contiguous "1" bits.

The format of the MUX-PDU is shown in the Figure. 13 below.

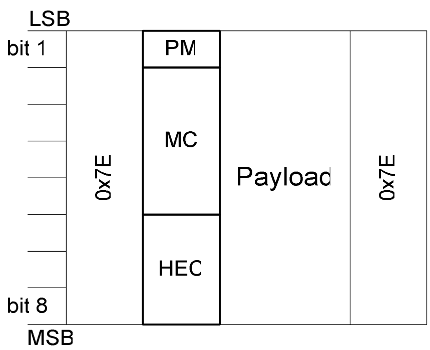


Figure 13: Mobile level 0 MUX-PDU format

The MUX-PDU has a header part of 1 octet. The header part consists of three fields: the HEC (Header Error Control) field (3 bits, bit 8~6), the MC field (4 bits, bit 2~5), and the PM (Packet Marker) field (bit 1).

The 4-bit MC is an index referencing one of the maximum 16 MultiplexEntryDescriptors in the multiplex table. The referenced MultiplexEntryDescriptor defines to which logical channel each octet of the MUX-PDU information field belongs.

The 3-bit HEC provides error detection capability for the MC using a 3-bit CRC. It contains the remainder of the division (modulo 2) by the generator polynomial $P(x) = x^3 + x + 1$ of the product x^3 multiplied by the content of the MC field.

The 1-bit PM is used to mark the boundary of the MUX-SDUs from segmentable logical channels. As we have mentioned, the MUX-SDUs from non-segmentable logical channels must be put into the MUX-PDU as a whole. Thus they don't need a mechanism to mark boundaries. However, the MUX-SDUs from segmentable logical channels can be segmented into several parts and put into the several MUX-PDU. Therefore, a mechanism is needed to point out the boundaries. For MUX-PDUs of H.223 level 0, PM is used to do the marking. Only one segmentable MUX-SDU is permitted to terminate within a MUX-PDU. As soon as the end of any MUX-SDU from a segmentable logical channel is reached, the MUX-PDU shall be terminated with a closing flag and the PM field in the next MUX-PDU shall be set to "1".

B. Mobile Level 1

The MUX layer for level 1 and 2 provides the same primitive interfaces as level 0. However, due to different requirement of error detection/correction, there are some differences about the header parts, the flags, etc.

The major difference between MUX layers of level 0 and level 1 is the flags used. This level has two different flag modes. In the basic mode, the MUX layer uses the 16-bit PN-flag 0xe14d. In the double-flag mode, two consecutive PN-flags are used together to delimit the MUX-PDUs. As the same as level 0, the double-flag preceding the MUX-PDU is defined as the opening flag.

The double-flag following the MUX-PDU is defined as the closing flag. The closing flag may also serve as the opening flag of the next MUX-PDU. Repetitive double-flags are also allowed at this level.

Different from level 0, the MUX layer of level 1 doesn't assure the transparency of flags in the data contents. It doesn't provide inserting/discarding '0' operations. The octet aligned structure of the MUX-PDUs can be used to reduce the emulation of flags. Emulation may further be reduced by using the HEC check of the multiplex header.

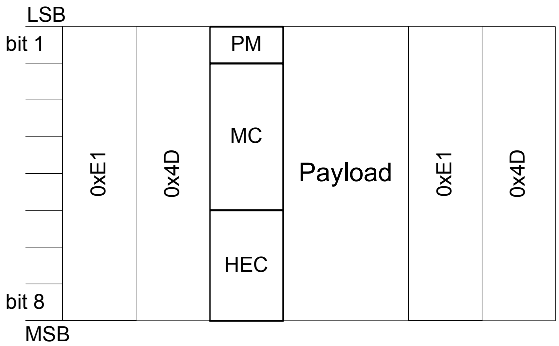


Figure 14: Mobile level 1 MUX-PDU format

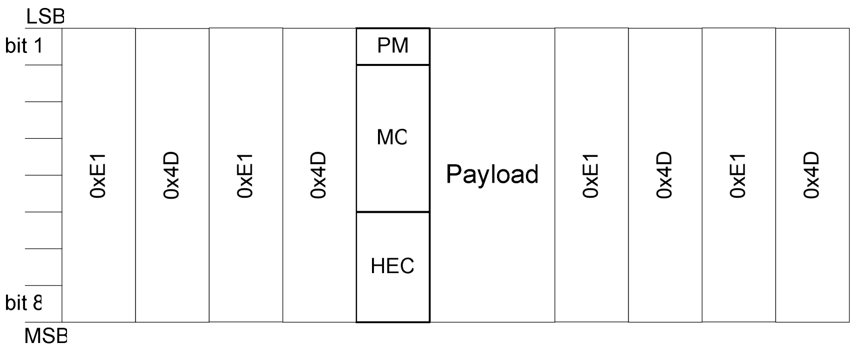


Figure 15: Double flag mobile level 1 MUX-PDU format

And the flag detection is also different in this level. It's done by correlation of the incoming bit stream with the flag. The output of the correlation is compared with a Correlation Threshold (CT). Whenever the output is equal or greater than the threshold, the receiver should decide that a flag has been detected.

C. Mobile Level 2

The flags used for this level is based on the level 1 flags, but there are also some differences. For level 2, there are two different kinds of flags: the PN-flag (0xe14d) and its complement. As there are two different flags, the complement flag can be used to signal additional information, such as indicating the PM (marking of segmentable MUX-SDU boundaries) information.

Similar to level 1, the MUX layer of level 2 is also using a correlation flag detection method. In determining the correlation sum, the correlator should interpret the zeros of the MUX-PDU flag to be "–1". The output of the correlator should then be compared with both a Correlation Threshold (CT) and its negative (–CT). The receiver should decide that a flag has been detected when the output of the correlator is either equal to or greater than CT (PN-flag), or if the output is less than or equal to –CT (PN-flag complement).

As the flags have already provided the PM information, the PM field is not needed in the MUX-PDU header. Meanwhile, some other fields have been added to improve the error detection/correction capabilities. The MUX-PDU format of level 2 is shown as the Figure 16 and 17 below.

The header field for level 2 contains 3 octets. It consists of three fields: the MC field (4 bits, same as level 0 and 1), the MPL (8 bits, Multiplex Payload Length) field, and the Parity bits field (12 bits).

The MPL describes the length of the information field in octets. As an 8-bit field, its value ranges from 0 to 254. The 255 value is reserved for future usage. Therefore, the MPL field has constrained the maximum MUX-PDU length to be 254.

The Parity bits field provides error detection and correction capabilities for the header part. It is using the Extended Golay (24, 12, 8) code, which comes from the original Golay (23, 12, 7) code. The description of (23, 12, 7) means that the total length of a Golay codeword is 23 bits, in which 12 bits are information, and the left 11 bits are check bits. 7 is the minimum Hamming distance between any two Golay codewords, i.e. each Golay codeword has 7 or more bits different from every other. Therefore, the Golay code can detect and correct a maximum of (7-1)/2 = 3 bit errors in any pattern.

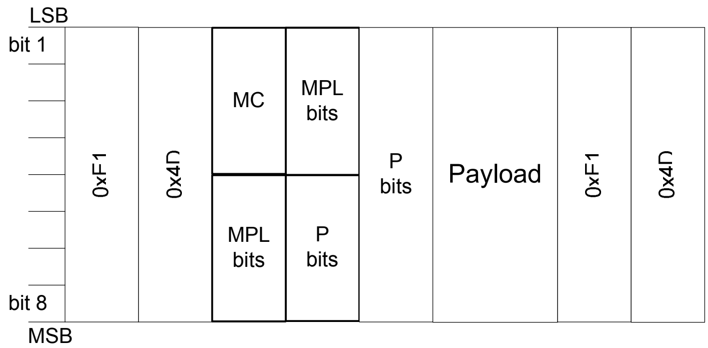


Figure 16: Mobile level 3 MUX-PDU format

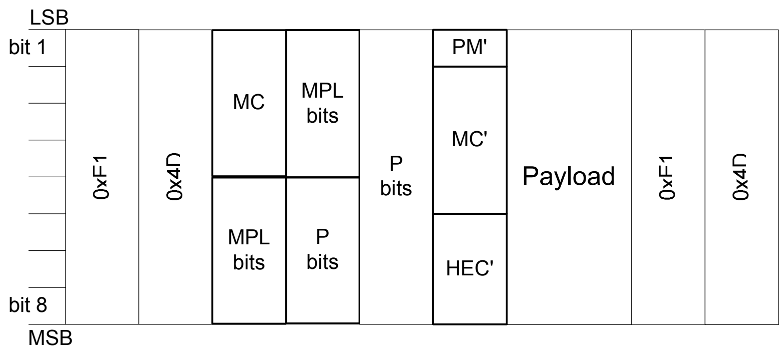


Figure 17: Optional four octet header of mobile level 3 MUX-PDU

In order to improve the error detection/correction capabilities of Golay code, an overall parity bit is added to produce the Extended Golay code. The extended code can still detect and correct all the 3-bit errors. What's more, it can detect all the 4-bit errors, but not corrected.

Note that, there is a tradeoff about the Golay codes' correction capabilities. If the Extended Golay code is used without any correction attempts, it can detect a maximum of 6-bit errors in any pattern and all the odd number of bit errors. If the code is used with correction attempts, it can detect and correct a maximum of 3-bit errors in any pattern, detect all the 4-bit errors and detect all the odd number of bit errors. The H.223 recommendation has not specified whether the correction capabilities shall be used. Therefore, the implementers have to make their own decisions according to the practical situations.

The parity bits P can be derived from the equation below.

$$
\begin{bmatrix} P1 \\ P2 \\ P3 \\ P4 \\ P5 \\ P6 \\ P7 \\ P8 \\ P9 \\ P10 \\ P11 \\ P12 \end{bmatrix}
=
\begin{bmatrix}
1&0&1&0&1&1&1&0&0&0&1&1 \\
1&1&1&1&1&0&0&1&0&0&1&0 \\
1&1&0&1&0&0&1&0&1&0&1&1 \\
1&1&0&0&0&1&1&1&0&1&1&0 \\
1&1&0&0&1&1&0&1&1&0&0&1 \\
0&1&1&0&0&1&1&0&1&1&0&1 \\
0&0&1&1&0&0&1&1&0&1&1&1 \\
1&0&1&1&0&1&1&1&1&0&0&0 \\
0&1&0&1&1&0&1&1&1&1&0&0 \\
0&0&1&0&1&1&0&1&1&1&1&0 \\
1&0&1&1&1&0&0&0&1&1&0&1 \\
0&1&0&1&1&1&0&0&0&1&1&1
\end{bmatrix}^{T}
\bullet
\begin{bmatrix} MC1 \\ MC2 \\ MC3 \\ MC4 \\ MPL1 \\ MPL2 \\ MPL3 \\ MPL4 \\ MPL5 \\ MPL6 \\ MPL7 \\ MPL8 \end{bmatrix}
$$

D. Multiplex Table

Multiplex table is the core element in the H.223 module. It defines the multiplexing patterns of a terminal. Each terminal defines its own multiplex table. And at the initialization phase of a call session, the terminal should exchange its multiplex table entries with the remote peer through corresponding H.245 control messages.

A multiplex table has maximum 16 table entries. Each table entry is defined with a MultiplexEntryDescriptor. The MultiplexEntryDescriptor takes the form of an elementList. There are two different types of elements in the list. The first type is the simple element which specifies a slot of data from a specific information source. This type of element has two data fields: the first data field specifies the Logical Channel Number (LCN) of the information source; the second data field specifies the length of the data slot.

{LCN1, RC 21}, {LCN2, RC UCF} (RC: repeat count; UCF: until closing flag)

This is a typical example of a MultiplexEntryDescriptor which is made up of two simple elements. The first element {LCN1, RC 21} specifies that 21 octets from the MUX-SDU of LCN1 will be filled into the MUX-PDU. The second element {LCN2, RC UCF} means that after the 21 octets from LCN1, octets from MUX-SDU of LCN2 will be filled into the MUX-PDU until the closing flag of the MUX-PDU packet.

The second type of MultiplexEntryDescriptor is more complicated one which is called nested MultiplexEntryDescriptor. It is used to specify a more complex multiplex pattern for the bit-stream in physical channel. The nested one also has two data fields: the second data field also specifies the slot length with a RC or UCF; however, the first data field takes the form of a subElementList. This means that the MultiplexEntryDescriptor is a kind of nested list. Inside the elements of the list, there can be sub element lists.

{LCN1,RC21},{LCN4,RC25},{{LCN2,RC1},{LCN3,RC1},RC UCF}

For the example above, the data field {{LCN2, RC1}, {LCN3, RC1}, RC UCF} is a nested structure. Its first data field is {LCN2, RC 1}, {LCN3, RC 1}, a 2-element subElementList, and its second data field is RC UCF. In real operation, the pattern {LCN2,RC1},{LCN3,RC1} will repeated until the closing flag is encountered.

According to different element list sizes, nesting depth and subElementList sizes, the MultiplexEntryDescriptors can be divided into basic and enhanced ones. At the initialization phase of the call session, the terminals shall signal their capabilities to interpret either the basic or enhanced MultiplexEntryDescriptor via H.245 control messages.

The basic MultiplexEntryDescriptors must satisfy the following constraints: the maximum elementList size is 2; the maximum nesting depth is 1; the maximum subElementList size is 2; the first element of the elementList must not use a non-segmentable logical channel more than once; and the second element of the elementList must use segmentable logical channels.

The terminals signaling capabilities to interpret enhanced MultiplexEntryDescriptor must also give a specific description about the capabilities with H.245 h223MultiplexTableCapability indication messages. And they should be able to handle all the basic MultiplexEntryDescriptors.

2.4 Chapter Summary

3GPP adopted H.324M with some mandatory requirements to form 3G-324M protocol stack for low-bit rate multimedia communication in 3G circuit switched network. In this protocol stack, H.245 recommendation is used for call control and signaling and H.223 recommendation is used for multiplexing. These protocols work together closely to provide reliable service over error prone wireless network.

Chapter 3. AN EFFICIENT H.245 MESSAGE PROCESSING FOR MULTIPLE CALL HANDLING IN 3G GATEWAY

3.1 Introduction

The penetration rate of third generation mobile service (3G) has increased explosively worldwide in recent years and as per the reports at 3GSM Congress in Barcelona (Feb 2007), the total number of WCDMA and 3G-CDMA subscribers have hit 104M and 325M respectively [1,2] by Dec 2006. Also in this report [1], it is clearly noted that the WCDMA subscriber sector is particularly growing very fast, i.e., 102% year to year and 16.6% quarter to quarter. This rapid spread of 3G is because of its multidimensional features like mobile TV, virtual personal assistance, realtime road traffic information and mobile bating etc along with the basic requirement of conversational video calling. Starting from the simple voice call to all these high end mobile services lead to high traffic and eventually the 3G gateway faces the challenge of handling incredibly large number of calls at the same time so as to ensure a good QoS to all the subscribers. Otherwise the gateway faces not only the loss of QoS, but also a severe hazard of performance bottleneck.

The 3rd Generation Partnership Project (3GPP) has adopted the H.324M with some modifications in codec and error handling requirements to create 3G-324M standard for 3G wireless networks. Adequate attention has been focused on 3G-324M in recent years. Some of those researches specifically concerned about implementation issues of this standard. Sanghyun Park et al [14] have implemented H.324M, however their simulation result involve serious performance degradation produced by the header because of corrupted flags. A.Basso and H.Kalva [15] have studied this limitation of 3G-324M for supporting streaming and messaging services and proposed a set of requirements that imply some extensions and clarifications of the standard to better support them. Some researchers focused their works in the ASN.1 messaging encoding. Rein Vesilo [16] proposed the design of ASN.l coders and compilers using recursive descent techniques from its theoretical basis in compiler theory with software engineering considerations. Since BER encoding/decoding have low efficiency, redundant codes. Thus, Qian Lv et al [17] have proposed a tree like method called "database definition" to store the necessary ASN.1 syntax information to increase the efficiency of ASN.1 message handling. Meanwhile, Most of the works are about video quality improvement. Lee Yen-Chi et al [18] have proposed a cross-layer decoder design that efficiently recovers more video data in the presence of transmission errors for 3G-324M video telephony over WCDMA networks with supporting simulation results. Miki T. et al [19] described the error-resilient audio-visual coding, MPEG-4/GSM-AMR, and the terminal standard, H.324 mobile extensions for 3G mobile multimedia terminals. However, the performance improvement of a 3G gateway which is designed to handle a large number of calls is seldom studied. Furthering this contrast, we have the genuine anxiety for investigating performance issues while we were engages in developing our won 3G gateway.

This paper is based on the results and realizations based on our on hand experimental results collected from our gateway. Here, we propose a precompiled-store-reuse algorithm to shorten 3G video call setup and message exchange time thus to enhance overall performance of a 3G gateway. The proposed approach requires an extra amount of high speed memory to be used for storing precompiled H.245 message binary streams. According to our experiments only a few H.245 message are frequently used for a normal 3G video calls.

The rest of this paper is organized as follows. In section 3.2, the redundancy of normal H.245 message handling is given and possible solution is discussed. We propose a table lookup algorithm approach to enhance the performance of video call handling of a gateway in Section 3.3. In Section 3.4, the experimental result are presented and discussed. Finally, we conclude this paper in Section 3.5.

3.2 Background

According to ITU-T standards, H.245 is a control channel protocol capable of conveying information for multimedia communication. In voice and video telephony as well as in VoIP, this sub-protocol (of 3G-324M) is basically responsible for call initialization, setup and for continuing the conversation. This sub-protocol manages this information exchange through a set of predefined messages. Some members of this message set are vital for the call initialization and setup, while some are responsible for continuing the conversation. So any attempt to improve the performance of the H.245 procedure is nothing but to handle this message exchange process efficiently. As this sub-protocol is responsible for the call setup and conversation continuation, through its message exchanges, its efficiency improvement will eventually lead to shorter call setup time and better conversation quality. Also in high traffic condition, for handling large number of call concurrently and to provide good conversational quality (after all the QoS), H.245 is the only vital protocol which should be taken care of properly.

According to ITU-T X.691 recommendation, initially, H.245 messages are in the form of ASN.1 and then they are converted into binary streams, when the call is initiated. After the requested terminal receives the bit stream, it reconstructs those messages back in to meaningful ASN.1 text and then it sends proper responses to react to the requester terminal.

In H.245 module, messages are defined in tree-like structure as is depicted in Figure 18. This defines a general message type called, MultimediaSystemControlMessage (MSCM). MSCM further comprises four different types of special messages, namely request, response, command and indication. A request message corresponds to a specific action and requires an immediate response through a response message. A command message requires an action but no explicit response. An indication message contains information (such as h223SkewIndication etc) that does not require action or response.

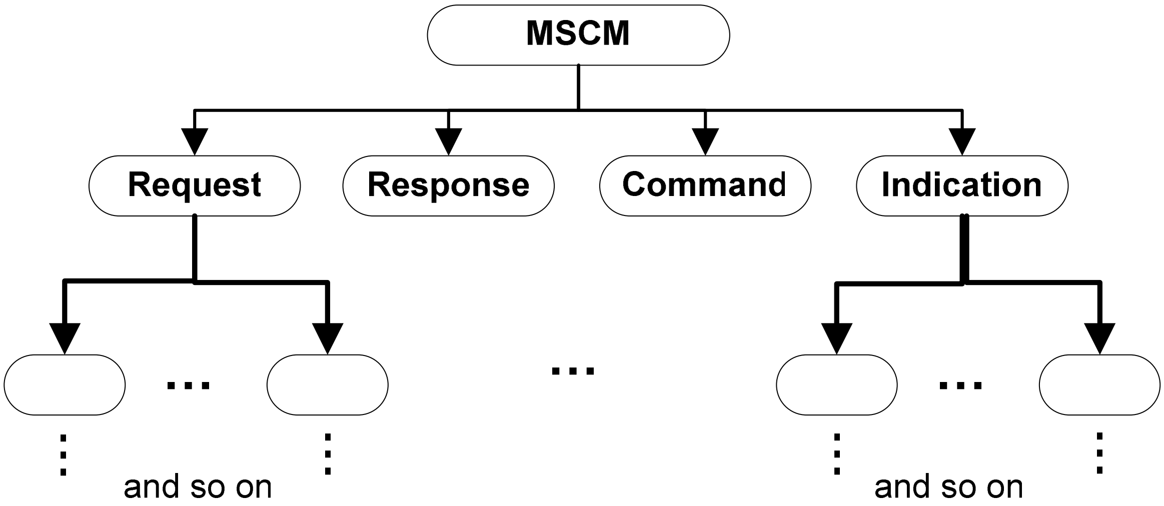


Figure 18: Hierarchical representation of the H.245 messages

3.2.1 Characteristics of H.245 messages in real 3G system

During this research and implementation work, we have verified that the most common H.245 messages for a 3G video call are:

1. TerminalCapabilitySet
2. TerminalCapabilitySetAck
3. MasterSlaveDetermination
4. MasterSlaveDeterminationAck
5. VendorIndentification
6. MultiplexEntrySend
7. multiplexEntrySendAck
8. OpenLogicalChannel
9. OpenLogicalChannelAck
10. MiscellaneousCommand:VideoFastUpdatePicture
11. RoundTripDelayRequest
12. EndSessionCommand:disconnect

Here only the message 6 and 8 are dynamic because their content depends on the messages sent from the remote terminal. Normally, message 1 to message 9 are only exchanged during call setup phase and message 10 and 11 are used to maintain the call, and finally conversation is terminated using EndSessionCommand:disconnect command.

To present a clear idea about the details of a message and its sub-messages, let's take TerminalCapabilitySet as an example. The message, TerminalCapabilitySet is used by sending terminal to inform receiving terminal about its multiplexer capabilities and also about its supported media (audio and video) codecs etc. A sample of the hierarchical message structure of TermnalCapabilitySet is shown in Figure 19. Here, we may observe that most contents can be kept unchanged since they describe the capability of a multimedia mobile terminal (as they are fixed for that specific terminal). However, there is a field called sequenceNumber which is used to label instances of TerminalCapabilitySet so that the corresponding response can be identified. If there appears multiple instances of TerminalCapabilitySet with the same content, then only the sequenceNumber filed can easily be changed dynamically.

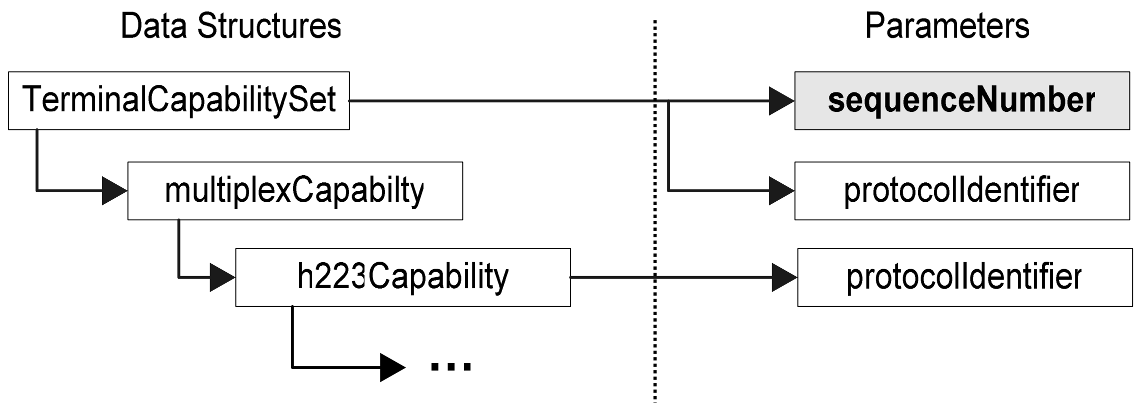


Figure 19: Message structure of TerminalCapabilitySet

3.2.2 Suggestion of a novel procedure

Traditionally H.245 messages are encoded in a chunk-by-chunk style. Each chunk corresponds to a sub-message (as is described in the previous paragraph) under the main message. For each chunk of input, encoding process must be called every time and the chunks are being encoded serially. So in this serial and individual encoding process for each and every message leads to high cost of system time.

Also we may note some very important points here:

1. Each main message and its sub-messages are the same for all the individual calls.
2. Through experience, we observe that some sub-messages under a main message are also repeated more than once.
3. Message like TerminalCapabilitySet request is no doubt same for all calls. But still, the response is also very few in number as in reality, terminal capabilities are fixed and there are a few (may be a few tens) number of different mobile terminals produced by different brands. So for most calls (may be thousands number of calls at a time in the gateway), these requests just same and responses are just little (10:1000, i.e., number of mobile terminal brands: number of total concurrent calls) varied.

Hence, it seems that the earlier mentioned serial coding process is just unnecessary killing of system time. In this contrast, if a low-cost process serves the same objective of producing the same encoded output, that is definitely worth to adopt. In this scenario, with the motivation of improving the call handling efficiency of the 3G gateway, we intuitively may suggest an innovative method on an experimental basis. In this method, we consider the possibility of reusing of a replicated encoded message set which is already compiled earlier for a previous call, unless the message data is changed! And if there is any change in some sub-messages, that part can easily be handled by dynamically compiling that part and updating the returned value in the whole encoded message string. We describe the method in detail as follows.

So in the message encoding programming level, it is not necessary to encode the whole string for each call, rather the string can be just updated with usually a minor (practically) changes in few bits. Here the precompiled message streams may be saved in a *lookup-table* and that can be accessed for each call instead of recompilation each time. Eventually it saves an appreciable amount of encoding compilation time.

Table 2: Traditional approach for h.245 message encoding

Input	Process	Output
M1	Encoding	A
M2	Encoding	B
M2	Encoding	B
M3	Encoding	C
M1	Encoding	A
M1	Encoding	A

Table 3: Suggested table lookup approach for efficient h.245 message encoding

Input	Process	Output
M1	Encoding +Reuse	A
M2	Encoding +Reuse	B
M2	Reuse	B
M3	Encoding +Reuse	C
M1	Reuse	A
M1	Reuse	A

A visual description of the above suggested reusable pre-compiled mess data set and its lookup table implementation method is compared to the traditional message encoding method is depicted in the following tables. Here we suppose that M1, M2 and M3 are H.245 messages and A, B and C are their encoded binary streams respectively. Table.1 shows the traditional approach for h.245 message encoding where each message is encoded individually. Table.2 illustrates assumed improved approach for H.245 message encoding in which reuse of previous encoded bit streams is deployed along with a minor update by dynamic compilation process. Comparing Tables 2 and 3, we get exactly the same output but at least three extra encoding processes can be skipped. So this leads to saving of at least 50% compilation time and thus giving rise to shorter call setup time (and also provides free system resources and time for better quality conversation in the post setup phase) in this example.

3.3 Table Lookup Message Encoding

Algorithm: table lookup message processing
Input: H.245 ASN.1 messages *m*
Output: binary stream *B(m)*

1. initialize system
2. pre-compile a message set *M(m)* and store into a table *T(m)*
3. for each i^th message H.245 ASN.1 message m_i
4. if $m_i \in M(m)$ then
5. retrieve bit stream B(m_i) from table $T(m_i)$
6. if replacement needed then
7. replace changed filed(s) and return
8. *else* return the bit stream
9. else dynamically compile it and store it into table *T(m)*
10. end for
11. repeat step 3 to step 10

Fig.3 Table lookup message processing algorithm

Basing on the earlier discussions and suggestions, here we present the technical details of the *loopup-table* message processing approach. An algorithmic representation is shown in above and its process flow is illustrated diagrammatically in Figure 20. Basically this algorithm consists of two major functions. In the first process, the system is initialized and then a set of messages, M(*m*) are (pre)compiled followed by storing the return data (bitstreams) into a table. Here we defined M(*m*) as a set of frequently used messages. Each time, when system comes across a new message that is not in the set, then the message is added automatically to the database in the table for future use. In the second part of the algorithm, simply message matching is performed. The input of this algorithm is a single H.245 message, m_i (m_i is the i^th message to be encoded), and the output is it's corresponding bit stream B(m_i). Looking from the programming point of view, during the system initialization, a H.245 message set is compiled into bit streams and then are stored in a table *T(m)*. For each H.245 message m_i (to be encoded), it is looked for in the precompiled message database (i.e., table *T(m)*), if a match is found, then it is retrieved and returned. After that, the returned bit stream B(m_i) is further checked whether a replacement is needed, if no, B(m_i) is returned directly, if yes, then the changed field(s) is dynamically compiled and replaced and returned. However, if the message m_i is not found, then it is just compiled and returned dynamically and is also stored in the database for further use in the future.

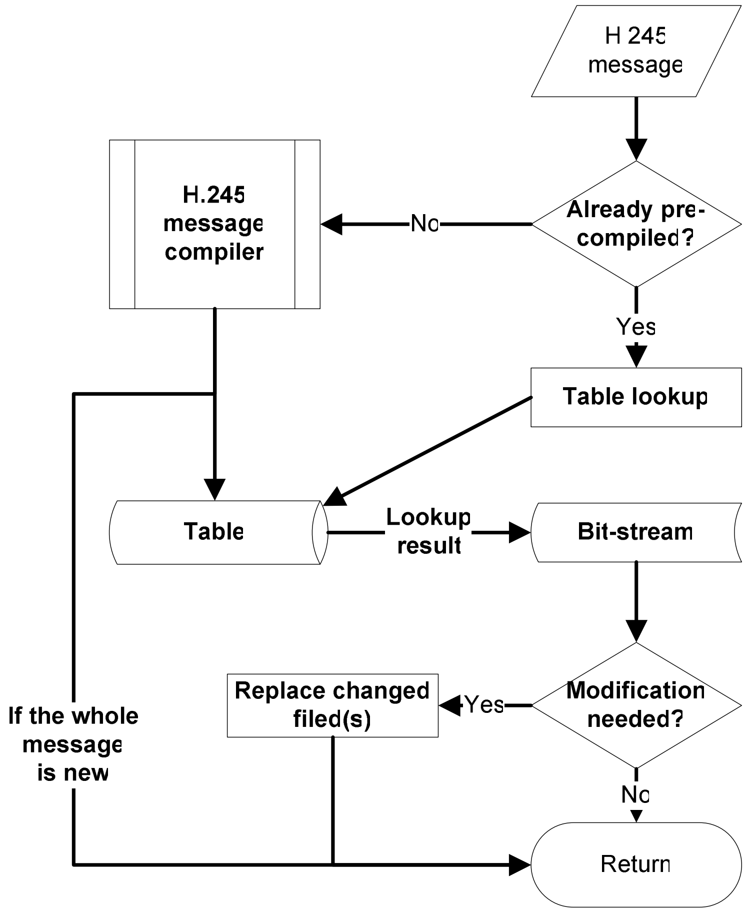


Figure 20: Table lookup process flow

3.3.1 Table Management

One of the key issues of *lookup-table* approach is to find the efficient way to manage the table data storage and retrieval system. We propose that the table should be managed in an *index-based* fashion. Each encoded message string is saved as an array in that table with an index number. So in the retrieval process, each entity is returned in terms of its reference number (integer). This approach is used because, it much less time consuming way to locate and return each table entry as compared to the traditional algorithms involving special key generation. Thus, an appreciable amount of time can be saved for data retrieval in the table and eventually leading to a big gain in the overall system performance. The Figure 21 describes this procedure pictorially.

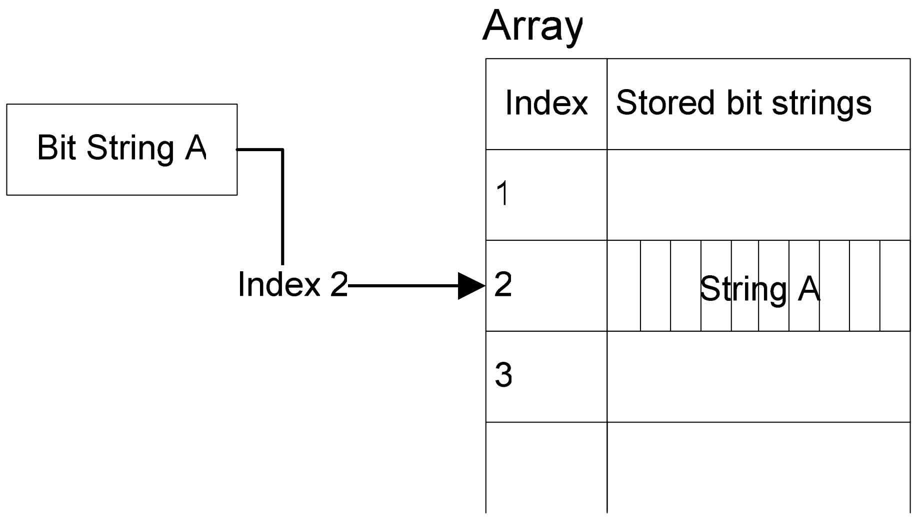


Figure 21: Array Implementation for the lookup-table

3.4 Test Results and Discussion

In this section, we present real life implementation of the proposed method and have demonstrated its efficiency though our experimental results. During this experiment, in order to track the call handling efficiency, we added two test modules namely MessageFeeder and DataCollector to our indigenous 3G gateway. MessageFeeder is responsible for selecting a large number (say 10000 at a time and then iteratively increasing in arithmetic progress) of H.245 message sets (that means a large number of incoming calls at the same time) to be input to the 3G gateway. And the DataCollector is employed to record the time at the start of encoding the messages and the time corresponding to bit stream is returned (that is the end of the encoding). It is practically obvious that these two factors can be compared so as to let us study the performance and efficiency of our 3G gate way in the view of handling the huge traffic of incoming calls. In this testing process, sets of H.245 message samples are system generated. In our experiment, the number of messages we chose are ranging from 10,000 to 100,000. In fact the longer the message length the better is the *performance enhancement* as the longer time for compilation is saved. But in practical, some messages are short length while some are long. So to have a balance in the testing process, we have taken consideration of both these types of message structures and have chosen samples randomly so that the collected data in the respective tables are nearly the average values.

The proposed method is compared to a method that uses the traditional way, i.e., if there is a message, then encode it and output it. The traditional procedure is time consuming as it follows this 'message-encode-output' procedure for each individual new incoming message. Since we aim at shortening call setup time, the time saving of handling the call setup and continuing the conversation in terms of H.245 message processing can of course claim the efficiency of our procedure. For the same set of messages, we calculated the time spent in message encoding for normal approach (t_1) and that of proposed approach (t_2) respectively. The *performance enhancement* (in terms of reducing call setup time) is calculated by using formula:

$$\frac{t_1 - t_2}{t_1} \times 100\%$$

If the result is positive, then it is obvious that we have achieved the *performance enhancement*.

We also successfully passed our 3G-gateway with embedding the currently proposed algorithm into it, for the compatibility test (whether the algorithm is compatible with the gateway protocol). We made phone calls to different brands/makes of 3G handsets and also to the Dilithium 3G Network Analyzer too. All attempts of 3G video calls were setup and conversations were hold successfully. It proves that the proposed algorithm is fully compatible with the existing 3G protocol as well as with all commercially available 3G handsets/equipments too.

The experimental result of performance enhancement in the *lookup-table* method of message processing is illustrated in Figure 22. From this graph, two important observations can be noted. Firstly, the performance remains almost unchanged when number of calls is increasing.

Secondly, the performance is linearly improving with more fractions of messages found in the lookup table.

The first observation just proves the very purpose of the currently introduced message processing approach, i.e., in this approach, the call handling performance of the 3G gateway remains invariant in spite of the number of calls increase consistently to a huge amount (i.e., from 10,000 to 100,000). So this procedure enables the gateway to successfully face the challenge of facing any amount of traffic load without degrading its performance.

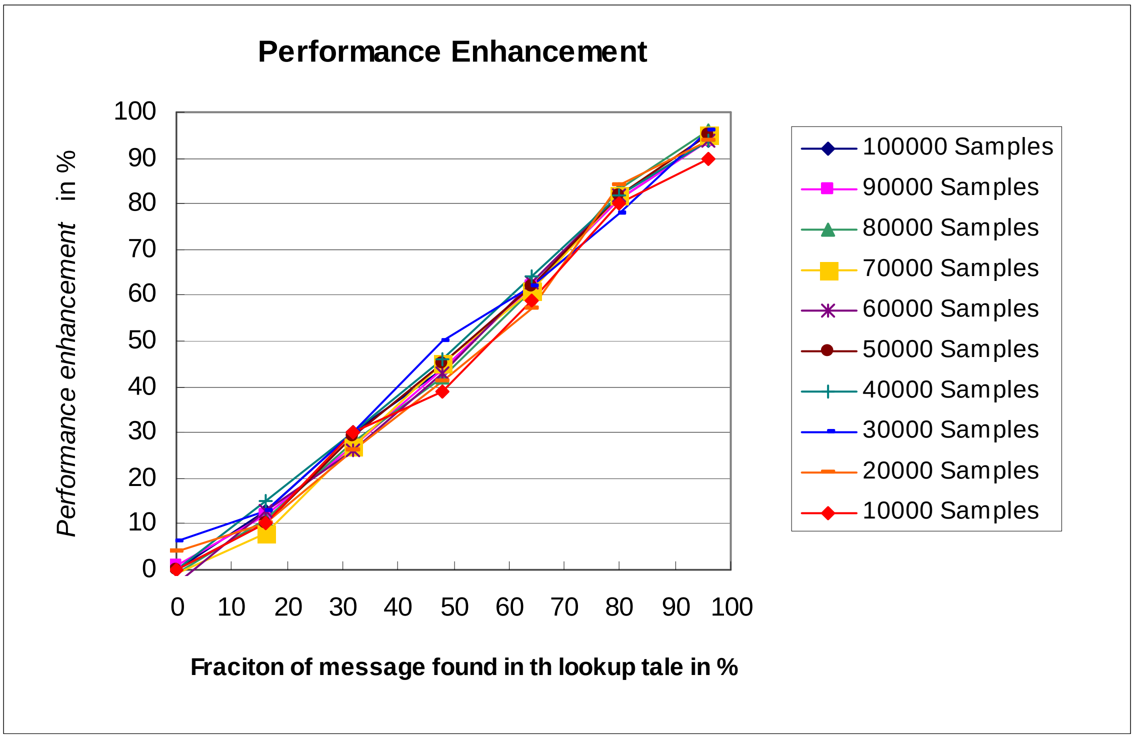


Figure 22: experimental result of table lookup message processing

Let us consider the second observation in the light of the discussion in section 3.2. There are twelve common messages for a normal 3G video call where ten out of them are responsible for call setup and are almost same in case of all calls. So these messages can be pre-compiled and can be stored in the lookup-table and can be directly retrievable for all calls. Thus the percentage retrievable message processing is 10/12 ≈ 83.3 %, and so also the prediction for the *performance improvement*. And amazingly, this prediction can be visually proved in the Figure 22. In this Figure, *Performance Enhancement* means the ratio of elapsed time for call setup before and after applying the binary reuse method in a call.

3.5 Chapter Summery

In this work, we have proposed a novel approach to enhance the efficiency of 3G gateway in order to handle huge number of multimedia calls at a same time. The details of this algorithm have been discussed to show that our assumption is correct and most of pre-compiled h.245 message bit stream can be reused so as to reduce time consumption in runtime encoding. Finally, throughout our experimental testing in a real 3G video call gateway, we have proved that our algorithm can give a guaranteed performance enhancement and is fully compatible with existing 3G protocols.

Chapter 4. AN EFFICIENT DATA DISPATCHING OPTIMIZATION FOR H.223

4.1 Introduction

In this chapter, a novel approach of an efficient direct message dispatching is proposed for H.223 multiplexing/demultiplexing procedure to speed up the message multiplexing and demultiplexing. As mentioned in chapter two, H.324 functional elements collect and generate four types of multimedia stream, here introduce them again briefly.

- The control information between different terminals or between terminals and the Multipoint Control Units (MCUs);
- Information from data applications such as T.120 data for real-time audio-graphics conferencing;
- Audio stream encoded by audio codecs;
- Video stream encoded by video codecs.

H.245 only deals with control stream according to H.245 recommendation, whereas all kinds of multimedia will eventually be passed into H.223 for multiplexing throughout the whole call session. This can be concluded that the efficiency of H.223 packet multiplexing will amazingly affect the performance of whole system, i.e. 3G-IP gateway.

4.2 Multiplex Entry

Multiplex table is the core element in the H.223 module. It defines the multiplexing patterns of a terminal. Each terminal defines its own multiplex table. And at the initialization phase of a call session, the terminal should exchange its multiplex table entries with the remote peer through corresponding H.245 control messages.

A multiplex table has maximum 16 table entries. Each table entry is defined with a MultiplexEntryDescriptor. The MultiplexEntryDescriptor takes the form of an elementList. There are two different types of elements in the list. The first type is the simple element which specifies a slot of data from a specific information source. This type of element has two data fields: the first data field specifies the Logical Channel Number (LCN) of the information source; the second data field specifies the length of the data slot.

{LCN1, RC 21}, {LCN2, RC UCF} (RC: repeat count; UCF: until closing flag)

This is a typical example of a MultiplexEntryDescriptor which is made up of two simple elements. The first element {LCN1, RC 21} specifies that 21 octets from the MUX-SDU of LCN1 will be filled into the MUX-PDU. The second element {LCN2, RC UCF} means that after the 21 octets from LCN1, octets from MUX-SDU of LCN2 will be filled into the MUX-PDU until the closing flag of the MUX-PDU packet.

The second type of MultiplexEntryDescriptor is more complicated one which is called nested MultiplexEntryDescriptor. It is used to specify a more complex multiplex pattern for the bit-stream in physical channel. The nested one also has two data fields: the second data field also specifies the slot length with a RC or UCF; however, the first data field takes the form of a subElementList. This means that the MultiplexEntryDescriptor is a kind of nested list. Inside the elements of the list, there can be sub element lists.

{LCN1,RC21},{LCN4,RC25},{{LCN2,RC1},{LCN3,RC1},RC UCF}

For the example above, the data field {{LCN2, RC1}, {LCN3, RC1}, RC UCF} is a nested structure. Its first data field is {LCN2, RC 1}, {LCN3, RC 1}, a 2-element subElementList, and its second data field is RC UCF. In real operation, the pattern {LCN2,RC1},{LCN3,RC1} will repeated until the closing flag is encountered.

The maximum 16 MultiplexEntryDescriptors in the multiplex table have defined all the possible multiplexing patterns of a mobile terminal. Each time a MUX-PDU (packet) is sent out or received, it shall be multiplexed or demultiplexed according to chosen MultiplexEntryDescriptor. Therefore, the performance of handling a MultiplexEntryDescriptor can greatly affect the overall performance of the data transmission.

However, H.223 does not allow unlimited elementList size, subelementList size, or nesting depth in the MultiplexEntryDescriptor. H.223 multiplexer can operate in two modes. In the basic mode of operation, the multiplexer must support an elementList size of up to 2, a subelementList size of up to 2, and a nesting depth of up to 1. On the other hand, the multiplexer may be able to operate in an enhanced mode, where it can support an elementList size of up to 255, a subelementList size of up to 255, and nesting depth of up to 15. Although H.223 multiplexer provides two types of operation mode, basic mode is always preferred since enhanced mode is not practical in real world applications.

Table 4: Examples of MultiplexEntryDescriptors
(LCN: logicalChannelNumber, RC: repeat Count, UCF: untilClosingFlag)

Row	MultiplexEntryDescriptor	Element ListSize	Nesting Depth	Subelement ListSize	Example
1	{LCN1,RC UCF}	1	0	0	All audio
2	{LCN3,RC UCF}	1	0	0	All video
3	{LCN1,RC21},{LCN3,RC UCF}	2	0	0	Audio, All video
4	{{LCN2,RC1},{LCN3,RC3} ,RC UCF}	1	1	2	1:3 data video
5	{LCN1,RC4},{{LCN2,RC1} ,{LCN3,RC2},RC UCF}	2	1	2	Audio, 1:2 data video
6	{LCN1,RC21},{{LCN2,RC2 },{LCN3,RC6},{LCN0,RC1 }RC UCF}	2	1	3	Audio, 2:6:1 data video control
7	{LCN1,RC21},{LCN4,RC25 },{{LCN2,RC1},{LCN3,RC 1}RC UCF}	3	1	2	Audio I, Audio II, 1:1 data video
8	{{LCN1,RC25},{{LCN2,RC 1},{LCN3,RC1},RC5},RC UCF}	1	2	2	2-level nesting

For basic mode, if there are only simple elements in the elementList, say, the single element represents the MultiplexEntryDescriptor, the structure can be handled easily by following the elements serially. However, when the descriptor becomes complex, i.e. in nested form, each time multiplexing is performed, searching for the corresponding entry in the nested MultiplexEntryDescriptor is required, therefore, the processing overhead increases greatly compared with former situation. This can be easily concluded that the problem resides in the nested structure of the elementList. Each element in the list could be extended to a subElementList, which might also contain both simple elements and subElementLists. Especially for the enhanced MultiplexEntryDescriptors, there can be a very big nesting depth, which can lead to repetitive callings of recursive functions to handle.

4.3 Direct Data Dispatching

Fortunately, through the deep analysis of MultiplexEntryDescriptors, a novel approach of an efficient direct data dispatching is proposed. The main idea of this approach is to tear the whole MultiplexEntryDescriptor down to a serial string thus to increase the efficiency of multiplexing. One key point is that RC UCF will appear only once in the MultiplexEntryDescriptors. That is, exactly one part could be repeated until the packet is ended. As a result, the whole MultiplexEntryDescriptors could be divided into two parts: the first part is called the RC part, which is made up of elements having finite repeating count; the second part is called the UCF part, which would be repeated until the end of the packet is reached. So the whole serialization process could be divided into two steps. At the first step, we find the point where the UCF part starts and divide the whole descriptor into the RC part and the UCF part. Then at the second step, we serialize the two parts into two separate sets of multiplex string to build another set of multiplex table.

Algorithm:
Input: a MultiplexEntryDescriptor
Output : a multiplex string

```
1:  input a MultiplexEntryDescriptor
2:  search for UCF
3:      if not found, go to RC (step 7):
4:      if found //deal with UCF part
5:           recursive call to break down the element list of UCF part
6:           obtain the multiplexing string from recursive call
7:  RC: //deal with RC part
8:  search for count number, n
9:  set i = 0, while i < n
10:     recursive call to break down the element list of RC part
11:     obtain the multiplexing string from recursive call
12:     increment i
13: exit
```

Table 5: corresponding multiplexing string for Table 4

Row	MultiplexEntryDescriptor	Fixed String	Repeating String
1	{LCN1,RC UCF}		1
2	{LCN3,RC UCF}		1
3	{LCN1,RC21},{LCN3, RC UCF}	111111111111111111111	3
4	{{LCN2,RC1},{LCN3,RC3},RC UCF}}		1333
5	{LCN1,RC4},{{LCN2,RC1}, ,{LCN3,RC2},RC UCF}	1111	233
6	{LCN1,RC21},{{LCN2,RC2}, {LCN3,RC6},{LCN0,RC1}RC UCF}	11111111111111111111	223333330
7	{LCN1,RC21},{LCN4,RC25}, {{LCN2,RC1},{LCN3,RC1},RC UCF}	1111111111111111111144 44444444444444444444	23
8	{{LCN1,RC25}, {{LCN2,RC1},{LCN3,RC1},RC5}, RC UCF}		11111111111111111111111 112323232323

After initializing multiplex table, each MultiplexEntryDescriptor is divided into “Fixed string” and “Repeating String” representing RC and UCF respectively. Table 5 shows the corresponding multiplexing string after running conversion algorithm for each MultiplexEntryDescriptor in the Table 4. Let us take MultiplexEntryDescriptor {{LCN1, RC30},{LCN2,RC20} , RC2} , {{LCN2 , RC12} , {LCN3 , RC13} RC UCF} as an example, which is an nested multiplex descriptor with nesting depth of 2. Figure 23 illustrates the procedure of converting a UCF part of a descriptor into a multiplexing string while Figure 24 shows the RC part. When this MultiplexEntryDescriptor is input to the algorithm, the algorithm searches for UCF first. In this example, UCF is found and thus the whole MultiplexEntryDescriptor is spitted into two parts, RC part ({{LCN1, RC30},{LCN2,RC20} , RC2}) and UCF ({{LCN2 , RC12} , {LCN3 , RC13} RC UCF}) part. After that, recursive call is applied to these two parts to convert the corresponding elementList into multiplex strings.

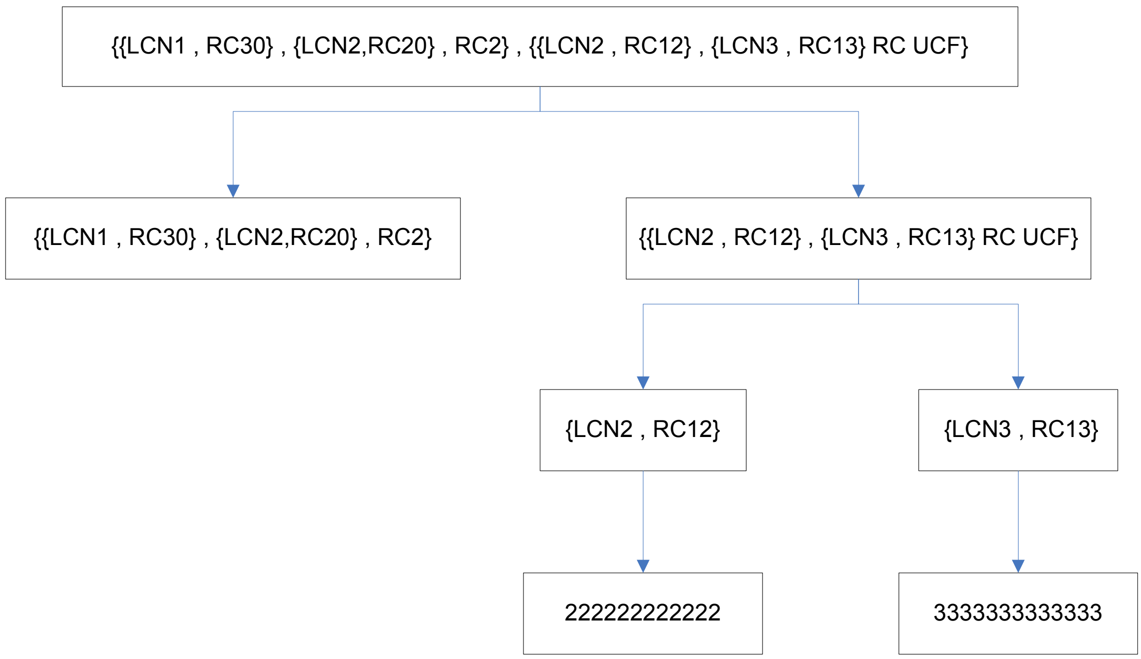


Figure 23: Converting UCF part to multiplex string

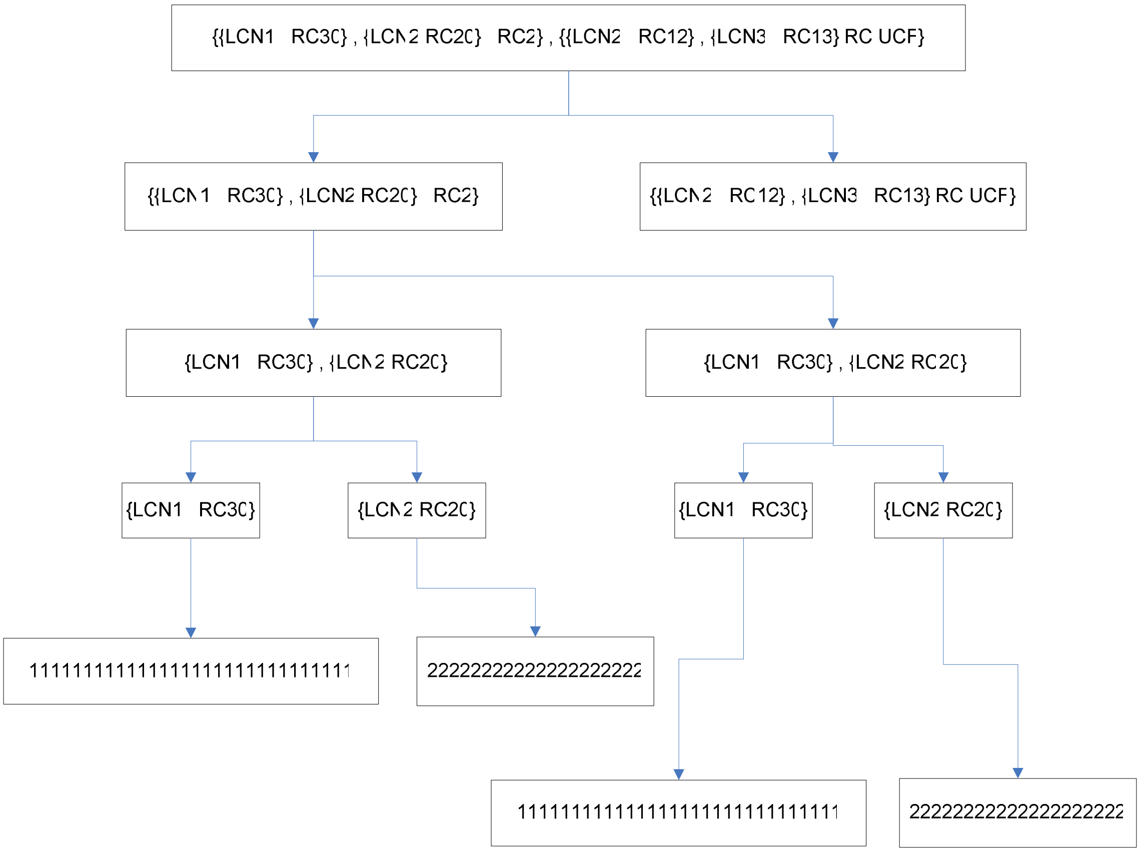


Figure 24: Converting RC part to multiplex string

4.4 Performance Analysis

4.4.1 Processing Overhead

Converting MultiplexEntryDescriptor into multiplex strings will eventually lead to an additional processing overhead, because the original complex table entries have to be broken down combination of multiplex strings. However, it should also be noted that this overhead only has to be taken at the beginning of the communication or when the two peers want to make modifications to the multiplex tables during the communication. So we only have to do the serialization for each entry few times. Thus the processing overhead of multiplexing/demultiplexing during communication should be more critical. With the original complex MultiplexEntryDescriptor structure, when multiplexer multiplex a packet, it has to process the possible nested structures, and recursive function calls could be used. It would be very costly to call recursive functions each time a packet multiplexing is performed. But with the multiplexing string, which is made up of “fixed string” and “repeating string” or either one of these if no RC part of UCF part is present, all we have to do is just to process from the head to the tail of the string. For the list of the RC part, we only have to traverse the whole string and fill in the packet with data from different sources according to the multiplexing string. For the list of the UCF part, we also just traverse to fill in the bytes. But UCF may need to be traversed for several times until the end of the packet is reached. Therefore after conversion of MultiplexEntryDescriptor, the operation of multiplexing will be straightforward and it will consume less processing time. Therefore, if converting overhead is compared with saved multiplexing cost, this overhead is negligible.

4.4.2 Complexity Analysis

In this conversion algorithm, the recursive will be called several times to perform actual multiplex string conversion. From software engineering point of view, if the depth of recursion is large, this kind of processing will be the disaster for the system. Therefore, it is necessary to make sure this algorithm is not a burden to the gateway. In order to have a reasonable analysis of the complexity of this algorithm, some assumptions have been made. First of all, basic mode of multiplexer operation is used since advanced mode is impractical in real world application. Secondly, there are totally 16 (maximum) entries of MultiplexEntryDescriptors are in use, each entry consists of an elementList size of up to 2, a subelementList size of up to 2, and a nesting depth of up to 1. In this setting, there will be an outside loop executing 16 times for 16 MultiplexEntryDescriptors, inside the loop there will be 4 recursions and each recursion has a depth of 1. It is quite reasonable that we consider the recursion with depth of 1 as a normal function execution so the execution complexity can be ignored. Hence, it can be easily concluded that the algorithm is bounded by $O(n)$, when n is the number of multiplex entry with the maximum value of 16.

4.5 Experimental Performance Evaluation

In this section, the multiplexing performances of original and multiplex string are compared and presented here.

In order to evaluate the cost in terms of time, three time parameters have been defined: t_1 is the time used to handle multiplex entry; t_2 is the time used to increment counter; t_3 is the time used to fill in packet according to multiplex entry (dispatching). Meanwhile, let RC30, RC20, RC2, RC12, RC13 and RCUCF in the multiplex entry {{LCN1, RC30},{LCN2,RC20} , RC2} , {{LCN2 , RC12} , {LCN3 , RC13} RC UCF} be RCN1, RCN2, RCN3, RCN4, RCN5, and RCN6 (the total number of iterations until UCF is found) respectively. In order to model the nested multiplex entry, the operation is written in pseudo code below.

```
//handling non UCF part
for(i = 0; i<RCN3;i++)
{
        for(j=0; j<RCN1; j++)
        {
                dispatches to LCN1;
        }
        for(k=0; k<RCN2; k++)
        {
                dispatches to LCN2;
        }
}
//handling UCF part
//RCN6 is the number of iterations
while (flag!=closing flag)
{
        for(j=0; j<RCN4; j++)
        {
                dispatches to LCN4;
        }
        for(k=0; k<RCN5; k++)
        {
                dispatches to LCN5;
        }
}
```

For direct multiplex string model, the only thing the multiplexer needs to do is just to dispatch packet until the end of multiplex string is reached. This results in a very simple pseudo code:

```
while (! End of multiplex string)
{
    Dispatches to corresponding logical channel;
}
```

In both algorithms, as we can see, the time taken for filling packets into corresponding logical channel is proportional to the number of repeat count (RC) and the length when closing flag (UCF) is encountered. Furthermore, as we have discussed in previous section, the conversion overhead for converting nested multiplex entry into direct multiplex string would be negligible, only processing cost of packet processing is compared and analyzed.

There are two modes of multiplex table operation, in basic mode operation of multiplex table, the nesting depth is of 1 and thus the cost of recursive call is bounded. For enhanced mode, the recursive call would be a disaster since the nesting depth could be 15. That is also why enhanced mode is impractical to be adopted in real world application. In this example, only basic mode is analyzed. The total time spent for processing nested structure and multiplexing packets, the time cost can be modeled as:

$$(t_1+t_2+t_3)(RCN1 + RCN2)RCN3 + (2\,t_1+t_2+t_3)(RCN4+RCN5)RCN6 \qquad (4.1)$$

In contrast to this nested multiplexing model, the multiplex string multiplexing model can be modeled as:

$$t_3\,[(RCN1+RCN2)RCN3+(RCN4+RCN5)RCN6]. \qquad (4.2)$$

By taking difference, the saved time when multiplex string algorithm is introduced is:

$$(t_1+t_2)(RCN1+RCN2)RCN3+(2\,t_1+t_2)(RCN4+RCN5)RCN6 \qquad (4.3)$$

Therefore, the performance enhancement by using multiplex string algorithm can be intuitively obtained from (4.1) and (4.2).

$$\frac{(t_1+t_2)(RCN1+RCN2)RCN3+(2\,t_1+t_2)(RCN4+RCN5)RCN6}{(t_1+t_2+t_3)(RCN1+RCN2)RCN3+(2\,t_1+t_2+t_3)(RCN4+RCN5)RCN6}\bullet 100\% \qquad (4.4)$$

The above performance enhancement can exactly be applied to one single entry of nested multiplex descriptor. However, in real operation, the total number of packets flow through a gateway is an incredibly large amount of number. That means, even for single multiplex entry, the number of repeat count (all together in one call session) converges to infinity. In this case, RCN1 to RCN6 all together converge to infinity and thus can be canceled. Therefore, t_1, t_2 and t_3 become major factors affecting performance results. From the definition, t_2 is the time used for counter increment. This value is too small if it is compared with t_1 and t_3 or even other operations, so that t_2 can be ignored without problems. These facts eventually lead to a simplification of (4.4) to:

$$\frac{3t_1}{3t_1 + 2t_3} \bullet 100\% \qquad (4.5)$$

In order to measure the actual performance enhancement, the both algorithm have been implemented for collecting experimental data. A number of experiments have been done to evaluate the parameters t_1 and t_3. The result is illustrated in Tables 6 and 7: Time taken for handling a nested multiplex entry descriptor structure, t_1, for one million times is 47.2ms and the time taken for filling bytes into a packet according to multiplex entry, t_3, for one million times is 353ms. From these measurement results, percentage of performance enhancement can be directly induced by substitution.

$$\frac{3t_1}{3t_1 + 2t_3} \bullet 100\% = \frac{3 \bullet 47.2}{3 \bullet 47.2 + 2 \bullet 353} \bullet 100\% \approx 16.7\%$$

Table 6: Time taken for handling a nested multiplex entry descriptor structure, t1, for one million times

	Start Time	End Time	Time Spent
1	1031 ms	1078 ms	47 ms
2	1031 ms	1078 ms	47 ms
3	1031 ms	1079 ms	48 ms
4	1031 ms	1078 ms	47 ms
5	1031 ms	1078 ms	47 ms
6	1031 ms	1078 ms	47 ms
7	1031 ms	1078 ms	47 ms
8	1031 ms	1078 ms	47 ms
9	1031 ms	1078 ms	47 ms
10	1031 ms	1079 ms	48 ms
		Average	47.2 ms

Table 7: Time taken for filling bytes into a packet according to multiplex entry, t3, for one million times

	Start Time	End Time	Time Spent
1	1078	1431	353 ms
2	1078	1431	353 ms
3	1079	1432	353 ms
4	1078	1431	353 ms
5	1078	1431	353 ms
6	1078	1431	353 ms
7	1078	1431	353 ms
8	1078	1431	353 ms
9	1078	1431	353 ms
10	1079	1432	353 ms
		Average	353ms

4.6 Chapter Summary

Multiplex Table is the key element for H.223 for regulating the rules for multiplexing/demultiplexing packets. This chapter four described an efficient H.223 data dispatching for packet multiplexing/demultiplexing procedure to speed up the message multiplexing and demultiplexing. In this algorithm an innovative conversion of nested multiplex entry into direct multiplex string is suggested. Both the heuristic analysis and the experimental data are given to show a good performance enhancement by using direct multiplex string.

Chapter 5. 3G-IP COMMUNICAION PROTOCOL INTEGRATION

5.1 Introduction

3G-324M is defined to operate on traditional GSTN network. This protocol is also embedded in all 3G handsets and terminals to provide real-time communication. However, as we all know, TCP/IP network is a relative high speed packet switched network, it is impossible to use 3G-324M as the communication protocol as it is used for low-bit rate communication in circuit switched. The born drawbacks make 3G-324M not suitable for multimedia communication in TCP/IP network so that the seeking of another suitable protocol to co-work with 3G-324M over TCP/IP network is a must for the 3G-IP gateway. Fortunately, SIP (Session Initiation Protocol) [20] together with RTP (Teal-time Transport Protocol) can also provide similar functionalities as 3G-324M does, therefore, they are chosen as the main modules for interfacing with TCP/IP network for multimedia communication. To demonstrate the idea of these protocols integration, a novel VIM (Video Instant Messaging) system based 3G-324M and SIP has been developed. This demo system has gained a plenty of positive feedbacks in ITU TELECOM WORLD 2006 and ICT Expo 2007 in Hong Kong, especially from some telecom service providers. From now on, PC users can use a VIM Client to make video calls to any 3G phone user with good video and audio quality.

5.2 VIM System Architecture

Nowadays, most of VoIP softwares are providing voice from PC to landline/mobile phone by introducing a PSTN gateway in the core network. Unfortunately, none of them provides video call from PC to landline/mobile. In contract to this traditional VoIP services PC user can easily make video calls to 3G phones by deploying a 3G-IP gateway in the core network. With reference to Figure 25, which depicts the network topology of VIM system, people can easily tell the differences between the pure voice call from PC to landline/mobile and the audiovisual call from PC to 3G phone. The only difference lies on gateways used in the network.

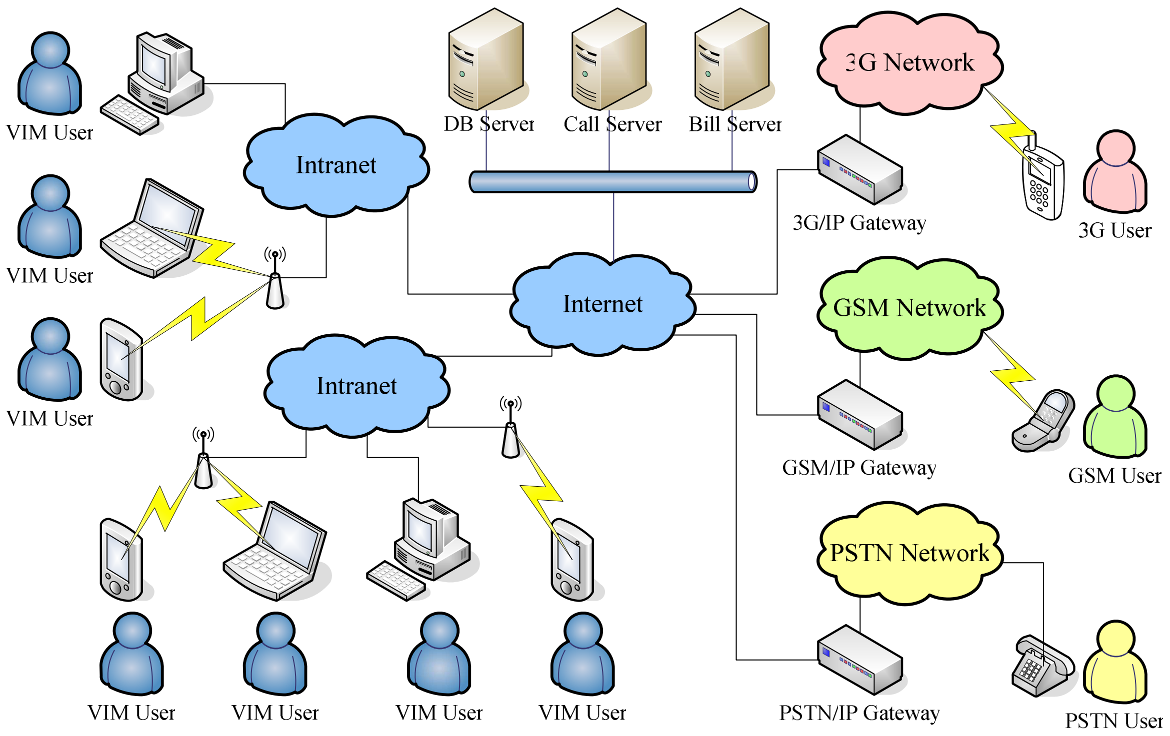


Figure 25: Architecture of VIM

The discussion of this architecture can be divided into five categories and they are discussed as follow:

1. User: VIM users can log in to the server on a PC, Laptop or PDA using their own account and password. One thing need to be noted is that gateways in this architecture should also be treated as users since gateways also need to log in to the server with some special identities.

2. Network Connectivity: The infrastructures such as LAN, WLAN, MAN, WMAN that provide the internet connectivity.

3. Communication protocol: The protocol that the users use to communicate to one another. For example, VIM clients talk to one another using Reduced-SIP (self-defined sub SIP

protocol). Also, VIM client communicates with 3G gateway using Reduced-SIP while the gateway talks to 3G phone users using 3G-324M protocol stack.

4. Gateway: Gateways enable cross network communication. For example, the 3G-IP gateway, the main theme of this thesis, is responsible for bridging multimedia traffic from IP network to 3G network and vice versa. To be more specific, the 3G-IP gateway not only performs bridging but also audio and video data transcoding e.g. from H.264 to MPEG4 as H.264 is not yet supported by 3G handsets

5. Server Cluster: In VIM architecture, only three servers are required. Database Server is used to store users' information. Call Server acts as the proxy between VIM clients and gateways. Bill Server provides billing service. DB and Bill servers interact with call server directly where only call server is exposed to users.

5.3 Session Initiation Protocol (SIP)

The Session Initiation Protocol (SIP), an IETF standard, is a human readable text-based and request-response structured application-layer signaling protocol for creating, modifying, and terminating sessions with one or more participant. These sessions here ranges from Internet telephone calls, multimedia distribution, to multimedia conferences. In November 2000, 3GPP has adopted SIP as signaling protocol and permanent element of the IMS [21] architecture. Till now, SIP is widely used as signaling protocol for Voice over IP (VoIP), along with H.323 and others. In typical use, SIP sessions are simply packet streams of the Real-time Transport Protocol (RTP) packed traffics.

SIP clients traditionally use TCP for carrying control data and UDP for transporting media data to SIP servers and other SIP endpoints. Although SIP is also originally believed to be simpler than H.323 and that is also the reason why SIP is invented. It has become as complex as H.323 in its current state due to the development and amendment of SIP. SIP and H.323 are not limited to voice communication over IP network only, it can also be used in any application where session initiation is required.

There are six types of request message defined in SIP, and they are distinguished by what is called a method:

1. REGISTER: Is used by a client to register an address with a SIP server.
2. INVITE: Indicates that the user or services is being invited to participate in a session. The body of this message would include a description of the session to which the callee is being invited.
3. ACK: Confirms that the client has received a final response to an INVITE request, and it only used with INVITE requests.
4. CANCEL: Is used to cancel a pending request.
5. BYE: Is sent by a User Agent Client to indicate to the server that it wishes to terminate the call.
6. OPTIONS: Is used to query a server about its capabilities.

On the other hand, the response messages (to the request messages) contain Status Codes and Reason Phrases that indicate the current condition of this request. The status code values are divided into six categories:

1. 1xx: Provisional: The request has been received and processing is continuing
2. 2xx: Success: AN ACK, to indicate that the action was successfully received, understood, and accepted.
3. 3xx: Redirection: Further action is required to process this request
4. 4xx: Client Error: The request contains bad syntax and cannot be fulfilled at this server
5. 5xx: Server Error: The server failed to fulfill an apparently valid request
6. 6xx: Global Failure: The request cannot be fulfilled at any server.

Figure 26 gives a good example of how request/response message combination works. User1 (sip:user1@here.com) initiate an invite (INVITE(1)) message to invite User2 (sip:user2@there.com) for a conversation. After proxies redirecting this message to User2, it replies with “200 OK” response message. Once this response message is routed to User1, then other procedures such as terminal capabilities description can proceed.

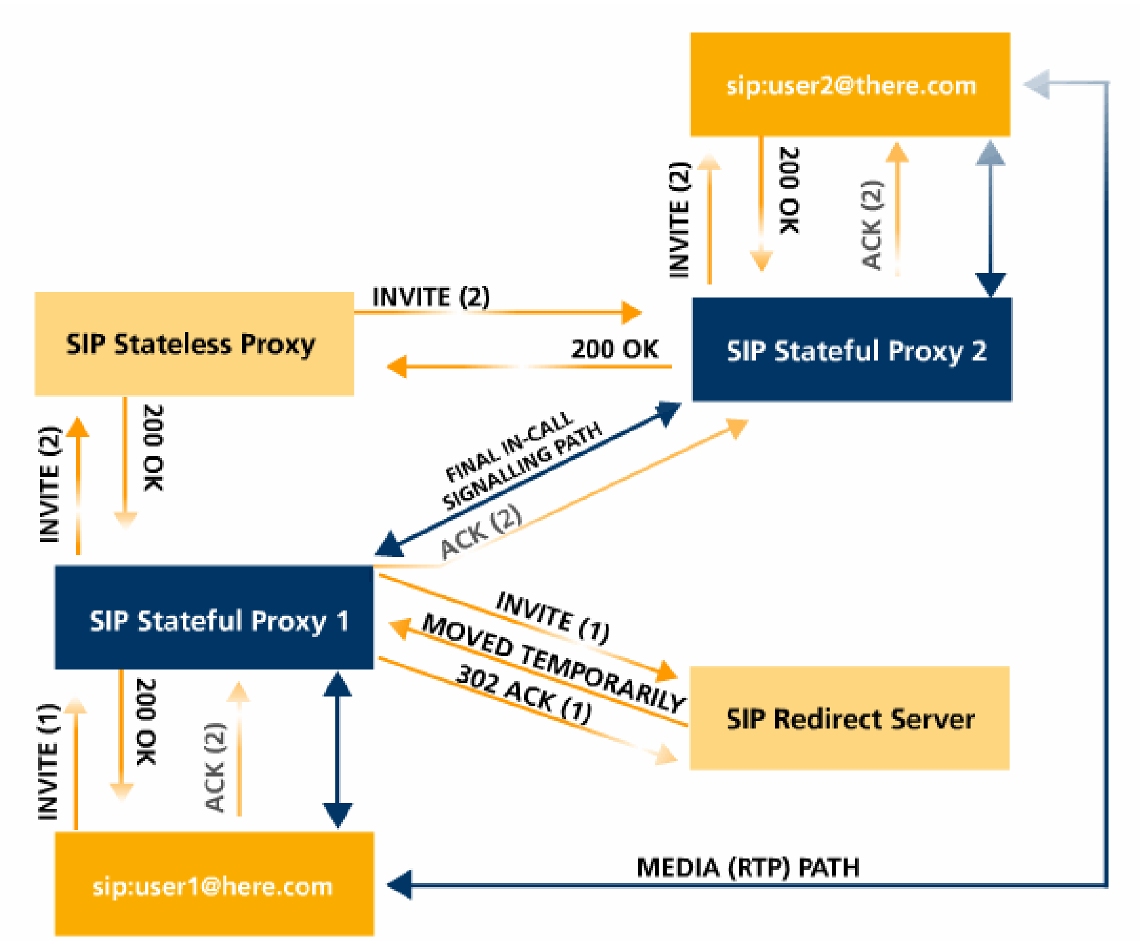


Figure 26: Basic SIP signaling flow

5.4 Reduced Session Initiation Protocol

In the previous step, a powerful session initiation protocol has been briefly introduced. Although this protocol can be used in any application where session initiation is required, it is still too powerful to use in 3G-IP gateway. Therefore, a reduced session initiation protocol (reduced-SIP) has been designed and implemented due to the following considerations:

- Enables fast development in the primary stage and thus reduces the cost including the man power involved

- Reduced-SIP can be extended to a full SIP easily in case of interconnecting to other SIP clients are required.

- VIM system does not require all functionalities provided in SIP.

In the VIM system, reduced-SIP is adopted as the communication protocol between 3G-IP gateway and VIM Client. In the reduced-SIP, only four out of six request messages are used and their corresponding functionalities are listed in Table 8. Moreover, the following pictures (Figure 27) illustrate the signaling flow for both call setup and call termination.

Table 8: Request messages used in reduced-sip

Request Code	**Description**	**Response Code**	**Description**
Register	Registration	1xx (100 ~ 199)	Provisional
Invite	Call	2xx (200 ~ 299)	Success
ACK	Confirm	4xx (400 ~ 499)	Client Error
BYE	Hang up	5xx (500 ~ 599)	Server Error
		6xx (600 ~ 699)	3G-IP Gateway Error

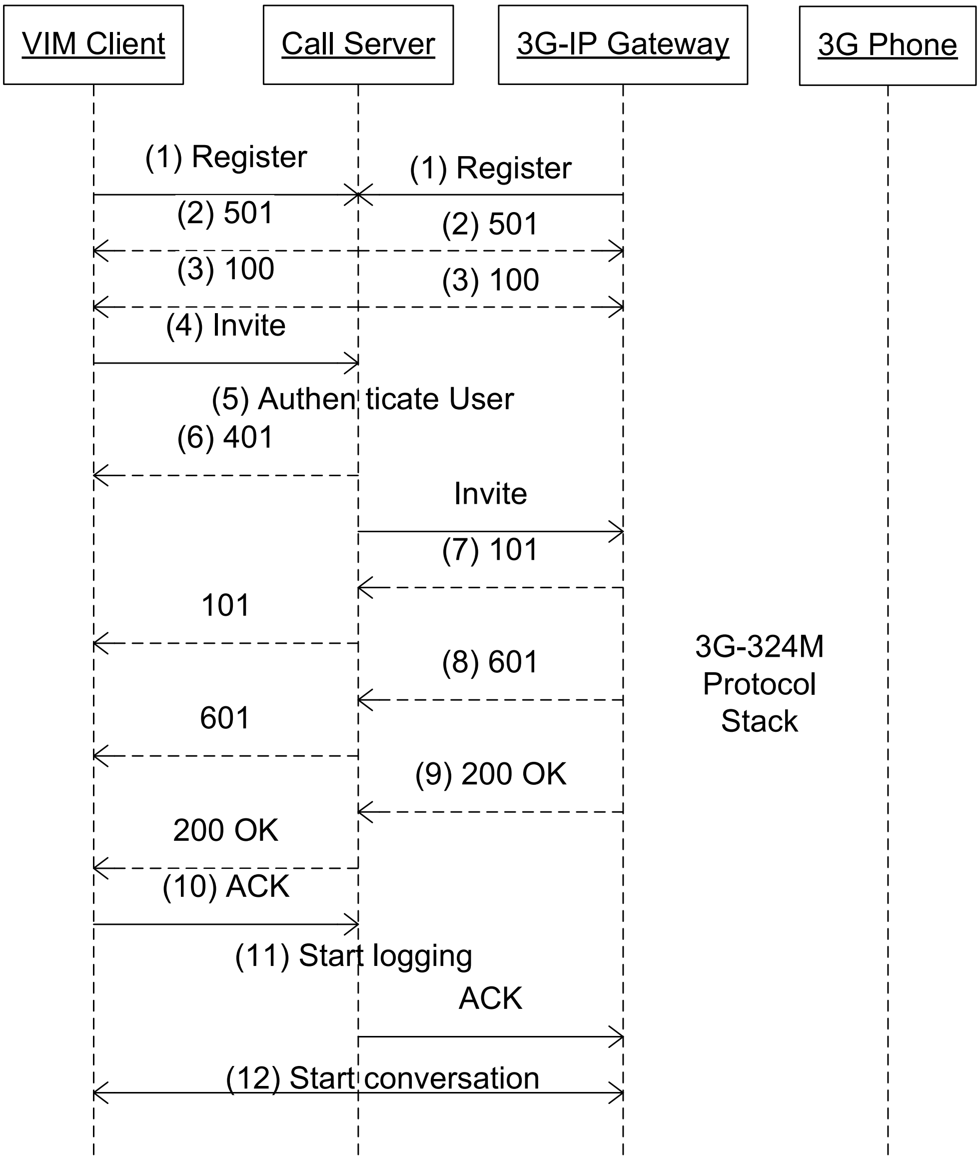


Figure 27: Invite signaling flow between VIM Client and 3G-IP Gateway

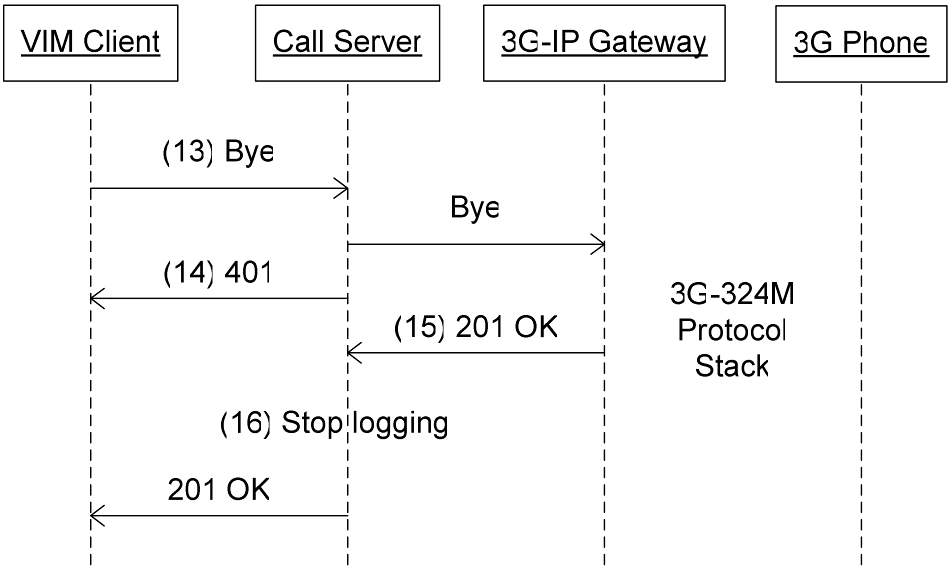


Figure 28: Call termination signaling flow initiated by VIM Client

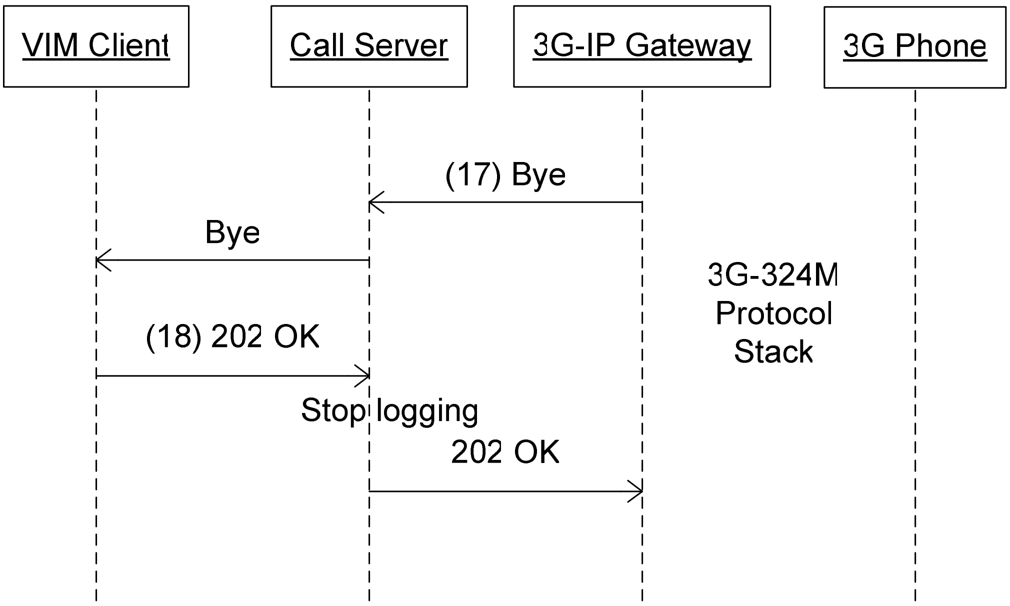


Figure 29: Call termination signaling flow initiated by 3G-IP Gateway

For the typical operation of VIM system, Both VIM Client and 3G-IP Gateway firstly issue a Register request to the Call Server for registration. After that, if the client wants to make a 3G video call from PC to 3G phone, it issues Invite request to the Call Server. Call Server queries DB Server to authenticate the client, it then relay the request to 3G-IP Gateway to make a video call to specified 3G phone using H.245 signaling and control of 3G-324M protocol stack. Once the call session between the Gateway and 3G phone is established, the Gateway return a 200 OK response code to the client via call server. Finally, the client replies the gateway with ACK. In the mean time, after call server received this ACK, it instructs DB server to starting logging call duration for billing. Call session starts immediately if above process is completed without errors, e.g. client error, server error and gateway error.

During the conversation, either client or gateway can terminate a call by issuing BYE request message. If the call is terminated by the client, the gateway will response with 201 OK. In the reverse way, if the BYE request comes from gateway, the client will reply with 202 OK. Meanwhile, once call server detects 201/202 OK, it stops logging call duration and pass this information to bill server for billing. To have a better understanding of how the system operates, Table 9 describes the request and response messages used in call setup and termination for Figure 28 to Figure 29.

In VIM system, the same as SIP, client and server communicate using TCP port 5060 for signaling, and UDP for multimedia traffic. RTP and RTCP are also deployed for multimedia data traffic control. However, as introduced above, 3G-324M protocol stack is used in between the gateway and 3G phone. There must be some transcoding need to be done in order to make multimedia data flowing through gateway to 3G phone.

Table 9: Request and response messages used in call setup and termination

Step	Description
(1)	Client/Gateway register to the server
(2)	Server error, e.g. maximum connection is reached
(3)	Register success
(4)	Client issue Invite to 3G phone
(5)	Call Server queries DB server to authenticate user
(6) (14)	Client error (e.g illegal user)
(7)	Gateway is idle
(8)	Gateway error
(9)	Gateway replies with 200OK
(10)	Client confirms
(11)	Server start logging call start time
(12)	Start conversation
(13) (17)	Terminate a call
(16) (18)	Server start logging call end time

5.5 3G-IP Transcoding

3G video streams are natively transported using circuit switched (CS) network, in such network, a dedicated channel will be setup during call setup period; the following data flowing in the medium is in bit streams and in sequential order, thus QoS is guaranteed. In contrast, most of traffics transmitted through packet switching (PS) network are in packet-wise format and out-of-order-arrival. Therefore, errors will occur while putting CS data into PS data directly and vice versa. In order to accommodate data transmission across these two fundamentally different network, development of a transcoding system is a must. As we have discussed early in this chapter, 3G-324M is the protocol stack used in 3G CS network while IP network uses SIP as the signaling protocol. In fact, these two protocols have some common characteristics that make interchange of data possible by introducing some transcoding techniques. Figure 30 illustrates the protocol mapping of 3G-324M and SIP with RTP/RTCP. We can amazingly find that both SIP and H.245 are responsible for control, signaling as well as terminal capabilities description. Other than signaling, audio/video codecs interface with audio/codec directly. If both parties, say, PC and 3G phone, support the common codecs such as AMR and MEPG4, then no transcoding of multimedia data is required, otherwise, the system needs multimedia transcoding.

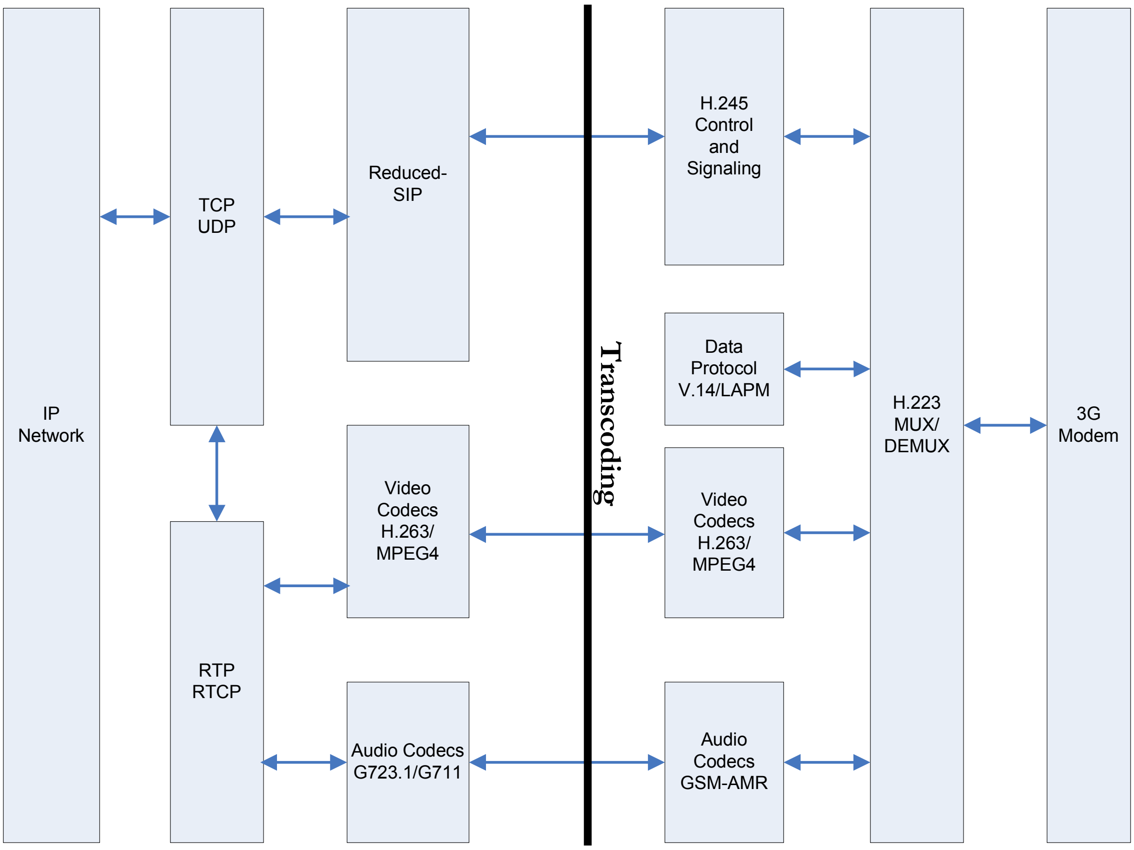


Figure 30: Protocol mapping of 3G-324M and SIP with RTP/RTCP

5.5.1 Overview of Transcoding System

The architecture of the transcoding system is shown in Figure 31. The system consists of four subsystems which are Air Interface, Control Handling, Data Handling, and Transcoding respectively and the functionality of each subsystem will be discussed one by one.

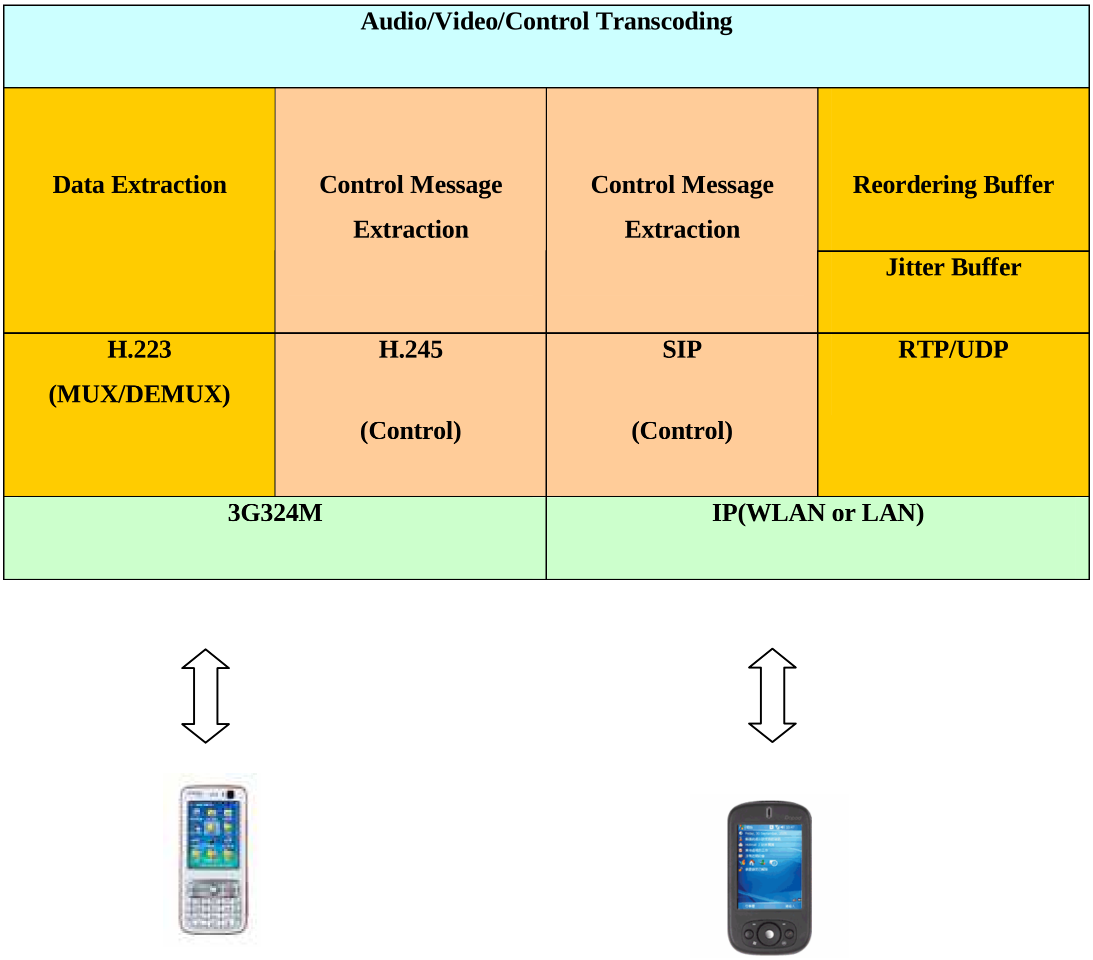
- **Air Interface Subsystem** – As its name indicated, in this subsystem, two physical devices, 3G modem and IEEE 802.11b/g module for sending/receiving data over AIR. However, more physical devices can be added to provide extended services in the similar manner, in this thesis, only 3G and WiFi are main concern (WiMax is the one in the future).

- **Control Handling Subystem –** This subsystem is responsible for control message exchanging between both natively supported (internal) terminals and external terminals. For example, H.245 in 3G324M is able to communicate with any 3G capable cellular phones, but it can also able to, in case of dealing with calls from IP network, extract control data from "H.245 packets" then pass extracted data to transcoding subsystem for further processing (will be discussed in transcoding subsystem). The same concept can exactly applied to IP side.

- **Data Handling Subsystem –** this subsystem is responsible for data exchanging between both internal terminals and external terminals. However, IP side of this subsystem will do more things than 3G side compared with control handling subsystem. To guarantee the QoS to users, reordering buffer is added to reorder in coming packets (in IP networks) and then deliver it to 3G network. In addition, jitter buffer is also added to inject jitter to provide lip synchronization.

- **Transcoding Subsystem –** This subsystem is a key of anyTRAN which provide audio/video transcoding as well as packet reassembly services to control/data handling subsystems. Different mobile terminals may has variant audiovisual capabilities, say, a 3G handset may only support AMR codec audio and H.263 codec for video which are basic requirement for a 3G handset to work properly. On the other hand, a mobile computer may be powerful enough to handle H.264 and other better audio codec. Thus, for these two mentioned mobile devices to communicate properly, an audiovisual transcoding is necessary. Moreover, this subsystem is also responsible for packet reassembly, i.e., it reassemble packets from 3G network to IP network by appending SIP header, or appending NSRP header for packets from IP network to 3G network.

Figure 31: Overview of transcoding subsystem



5.5.2 Data Extraction from H.245/H.223

Generally, for the sending entity, the data contents from different upper-layer applications will firstly be handled by the Adaptation Layer (AL), however, the H.245 control messages have to go through two additional layers, Control Channel Segmentation and Reassembly Layer (CCSRL) and Simple Retransmission (SRP), and then multiplexed into the same physical channel by the MUX layer. For the data transmission over circuit-switching channels, beginning and ending of data packets are generally marked with some special beginning/closing flags. Thus, the emulation of closing flags in the data packets can cause transmission failures as it may lead to extraction of wrong packets. CCSRL layer segments the H.245 control messages into small parts to avoid the emulation of flags so as to improve the reliability of transmission. Besides, it is easier to keep the integrity of small packets through the transmission. In SRP layer, an 8-bit sequence number is added into the packet. At the sender, each time a packet with a sequence number is sent out, a timer is started. At the receiver, each time a packet with the correct sequence number is received, an acknowledgement without sequence number is sent back to the sender. The timer is killed when an acknowledgement is received. If the timer times out, the sender has to retransmit the packet. The sender can not send out another packet until it receives the acknowledgement from the remote receiver. The SRP layer can also be improved into NSRP (Numbered SRP) layer with sequence number added into the acknowledgements. With retransmissions of the lost or modified frames, the layer provides high reliabilities for the transmission of control messages.

For the receiving entity, the data received from the physical channel will firstly be demultiplexed into different logical channels, and then handled by the AL layer. After hat, the data will be forwarded to the corresponding upper-layer applications. Therefore, data can be extracted directly from the layer by removing the header from each packet/segment in the incoming channel and then transform it into SIP packets.

Capturing H.245/H.223 data can be done straight-forwardly from the highest layer by referencing to Figure 32 below. H.245 data can be captured in between of H.245 and CCSRL layer while H.223 data can be captured in between media (audio/video) source layer and AL layer.

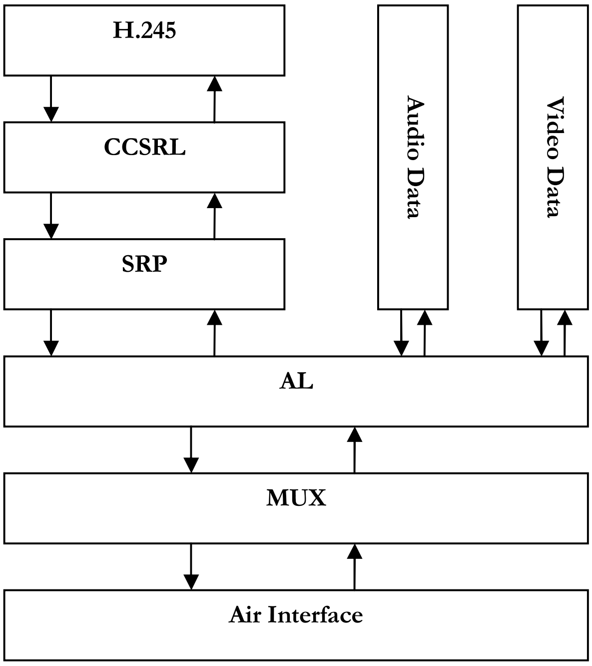


Figure 32: Layered view of 3G-324M

5.5.3 An Example of Message Conversion

Up to now, we should know that SIP and it is also a pure TEXT-based protocol like HTTP running on application layer. Thus, there is no rule regulating how to transform a H245 message to a SIP message. In order to have a clear view of the implement of this transformation, let us take a look at the following example.

Figure 33 shows a typical example of Terminal Capability Message of a 3G mobile phone in text view. In this message, both audio and video capabilities can be identified easily. For audio capabilities, a GSM-AMR codec is used (capabilityIdentifier standard : { 0 0 8 245 1 1 1 }) with maximum bit rate of 122bps (maxBitRate 122). In contrast, the video capabilities are a bit complex, in which a H.263 codec is used with a constant QCIF MPI 2 and maximum bit rate of 460bps and all of additional features are not supported such as PB frame in H.263 stands (eg. pbFrames FALSE). Meanwhile, Figure 34 shows a typical SIP-INVITE message that telling SIP server who is the called party (sip:Bob.Johnson@company.com) and which type of media is/are used (m=audio 20002 RTP/AVP 0; m=video 51372 RTP/AVP 31 32).

Once call request message is received by the transcoding system, it will send an INVITE message to the called client immediately using capabilities specified. If the called party return an acknowledgment to accept specified codec, then no media transcoding is required for this session, however, if the called party return its own preferred codec then media transcoding is required in transcoding system. For example, let *AUsi* and *VIsi* denote the *ith* audio and video codec supported by sender (calling party) and *AUri* and *VIri* be the *ith* audio and video codec supported by receiver (called party) respectively. If set $\{AUs1, AUs2,\ldots AUsn\} \cap \{AUr1, AUr2,..AUrn\}$ is not an empty set, then resulted set will be the selected audio codec(s) for both terminals, and if $\{VIs1, VIs2,\ldots VIsn\} \cap \{VIr1, VIr2,..VIrn\}$ is not an empty set, the resulted set will be selected video codec(s) for both terminals. If, however, resulted set for audio or video codec(s) is an empty set, the transcoding system should be responsible for transcoding between preferred codec(s) of both terminals.

```
request : terminalCapabilitySet : {
   sequenceNumber 1,
   protocolIdentifier { 0 0 8 245 0 10 },
   multiplexCapability h223Capability : {
          ….
          ….
      }
   capabilityTable {
      {
         capabilityTableEntryNumber 1,
         capability receiveAudioCapability : genericAudioCapability : {
            capabilityIdentifier standard : { 0 0 8 245 1 1 1 },
            maxBitRate 122,
            collapsing {
               {
                  parameterIdentifier standard : 0,
                  parameterValue unsignedMin : 6
               }
            }
         }
      },
      {
         capabilityTableEntryNumber 5,
         capability receiveVideoCapability : h263VideoCapability : {
            qcifMPI 2,
            maxBitRate 460,
            unrestrictedVector FALSE,
            arithmeticCoding FALSE,
            advancedPrediction FALSE,
            pbFrames FALSE,
            temporalSpatialTradeOffCapability FALSE,
            errorCompensation FALSE
         }
      }
   },
```

Figure 33: H.245 ASN.1 formatted TerminalCapabilityExchange message

```
INVITE sip:Bob.Johnson@company.com SIP/2.0
Via: SIP/2.0/UDP workstation1000.university.com:5060
From: Laura Brown <sip:Laura.Brown@university.com>
To: Bob Johnson <sip:Bob.Johnson@company.com>
Call-ID: 12345678@workstation1000.university.com
CSeq: 1 INVITE
Contact: Laura Brown <sip:Laura@workstation1000.university.com>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Laura 2891234526 2891234526 IN IP4 workstation1000.university.com
s=Let us talk for a while
c=IN IP4 138.85.27.10
t=0 0
m=audio 20002 RTP/AVP 0
m=video 51372 RTP/AVP 31 32
```

Figure 34: A Typical SIP invite message

5.5.4 Revised Call Signaling

To illustrate the signaling flow after introducing transcoding and signaling mapping between 3G-324M and SIP. Figure 35 is modified version of Figure 27 so that signals between the gateway and 3G phone are also available. Assume no error arises during this operation, both VIM client and the gateway register to the server and the server replies registration is finished successfully. After that, once the client initiate a call by issuing a Invite message, the Call Server relays this message to the gateway and then the gateway extract the target phone number to dial target 3G phone using AT command through 3G modem. Once the target user answered the call, AT command “CONNECT” will be returned to the gateway to indicate the called party has picked up the phone. The gateway and the 3G phone then immediately enter H.245 call setup phase by exchanging both terminal’s capabilities including supported media codecs and open audio and video logical channels for audiovisual communication. As long as the gateway learns that H.245 call setup phase is completed, it sends response message 200 OK to the client via call server. Finally, right after ACK is return from client followed by a 200 OK response, video conversation can proceed. During this video conversation session, transcoding of audio/video data at the 3G-IP Gateway may be required if the VIM Client and 3G Phone do not have a common set of support codecs.

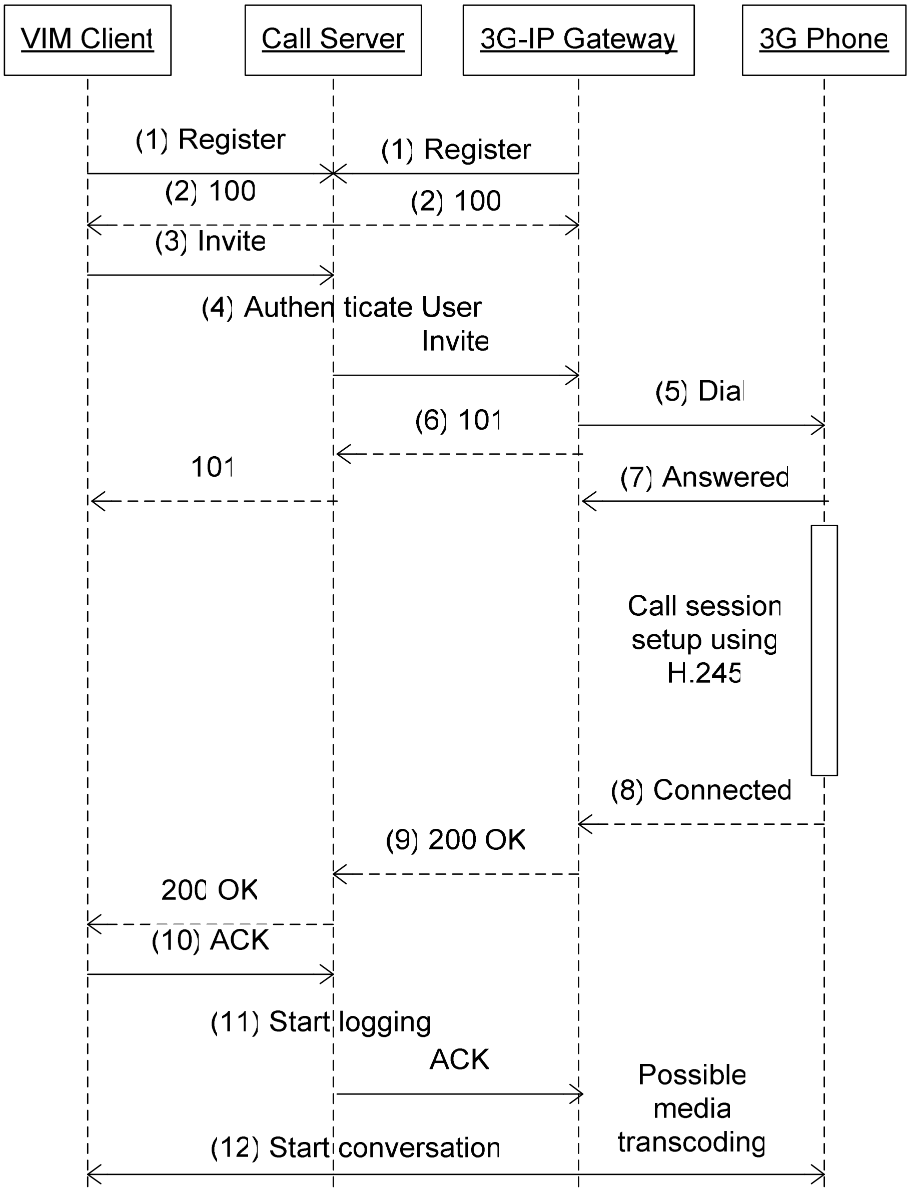


Figure 35: VIM Client, 3G-IP Gateway and 3G Phone signaling

5.6 The Running System

Figure 36 shows the running system of the whole VIM system for evidence. Server and Gateway are running on the same machine (left laptop, Windows XP), while the VIM client is running on the other machine (right laptop, Windows XP). Both laptops are connected to the WiFi AP (Linksys) in the middle. In this picture, both images from VIM client to 3G handset and from 3G handset to VIM Client can be seen clearly. That means the VIM system can provide a good quality of video call from PC to any 3G phones.

Unfortunately, using a 3G modem for the gateway restricts the number of concurrent connection to a single connection only. However, the gateway are designed to handle multiple calls and which is also experimentally verified to handle up to 100,000 concurrent calls at a time

Figure 36: Running VIM demo system

by generating pseudo calls into the gateway.

5.7 Chapter Summary

This chapter introduced a video instant messaging client based on a novel reduced-sip. This reduced-sip enables a fast sip compatible application development. Also, by integrating this sip to a PC to 3G video call gateway, the integration of 3G-324M and SIP communication become possible. As the result, a novel Video Instant Messaging (VIM) system has been developed. This system innovatively uses SIP and 3G-324M to enable from PC/Pocket PC to 3G-324M video call. The architecture of VIM was given and discussed in this chapter. In addition, due to the nature of H.245 call control and SIP signaling, the transcoding of control message has been suggested so as to make the message understandable to both 3G and TCP/IP network.

Chapter 6. NEXT GENERATION MULTIMEDIA COMMUNICATION

6.1. Introduction

These days, iPods, palmtops, tablet PCs and high end mobile phones have become like inexhaustibly essential part in day to day life. It is because, these equipments provide not only the features like phone calling, music playing, photo and vide capturing, but also they do have incredible features like seamless accessibility and connectivity that enables the modern life to be pervasively connected to any equipment around him or to any point in this world. In more technical terms, these advanced communication devices have the provision of auto detection, accessibility and connectivity through various network types such as, infrared, Bluetooth, mobile phone, traditional phone(call divert, handsfree, wireless pbx etc), WiFi (internet, SIP phone and messengers). So, the demand of fancy multimedia applications such as video conferencing, VOD (Video on Demand), IP TV, Mobile trading/bating, on-line games etc have become a general trend in today's networked life style. Eventually these bulky multimedia contents have caused dramatically increasing traffic load over both mobile phone and internet networks. Moreover, there is extremely high demand on the complete convergence and unification of the traditional telephone, mobile phone and the internet networks so as to provide seamless service of any type to subscribers at any place, any time. Some networks and technologies are already inter-interlinked and some are still untagged. This is because, the core part of interlinking all technologies is the gateway that plays the key role to make all these technologies able to communicate among themselves. In this contrast, we of course realize that a highly efficient, reliable, easily maintainable and upgradeable multimedia gateway should be designed and developed that can universally support the integration of all existing technologies and also future technologies too.

During this research, one genuine necessity in this high tech communication world do strikes our mind. "Is not it possible to develop the technology to have any type of communication to any network, at any time and at any place from a single window interface either on the mobile screen or on the PC's ?" This question is pictorially depicted in Figure 37. How nice will it be if from this type of single interface window in both PC and mobile handsets/devices, all the communication and conversation facilities can be availed! In order to support this single window universal communication system, the cross technology and inter network call handling must require a giant, robust and efficient gateway!

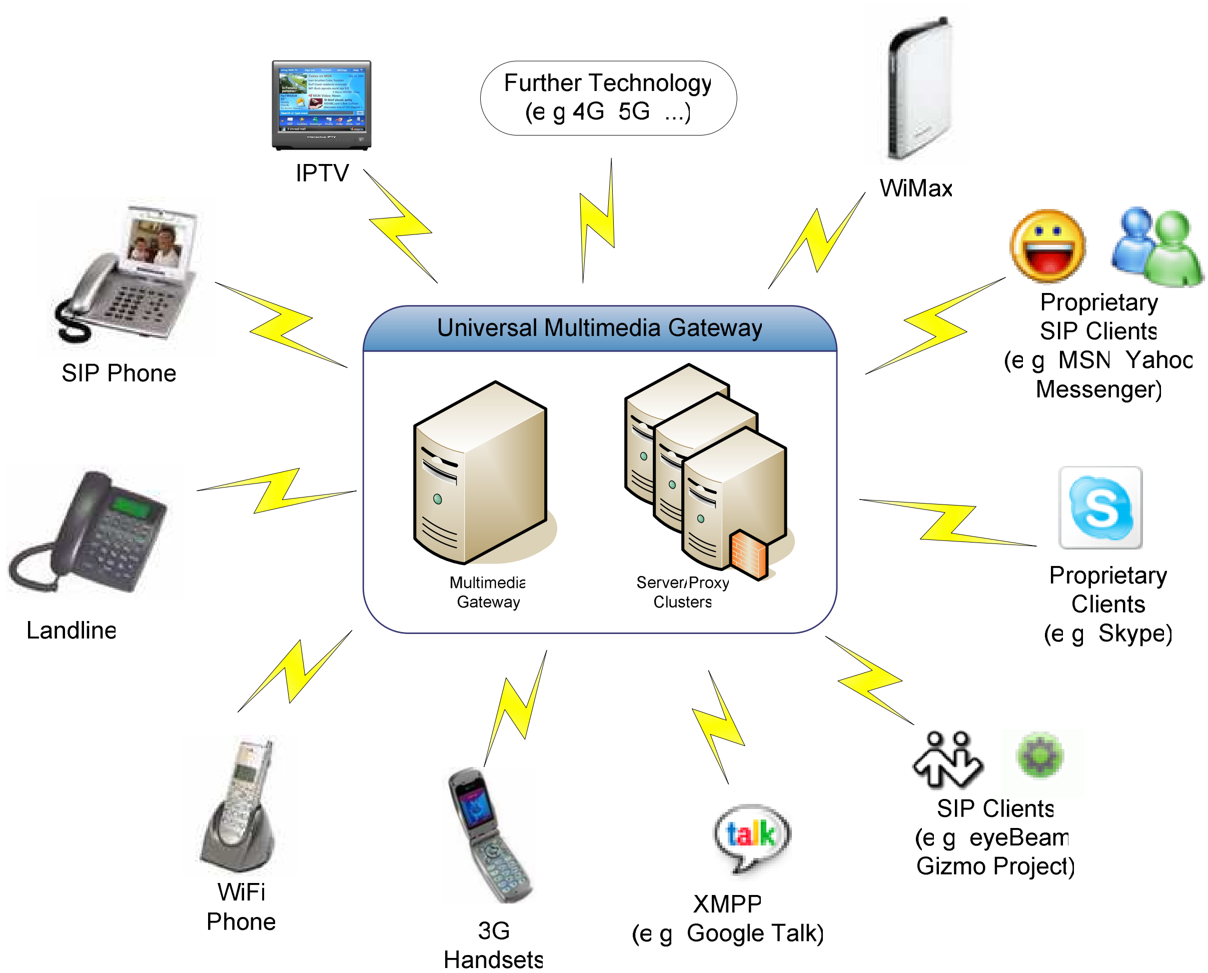


Figure 37: Seamless accessibility through universal gateway

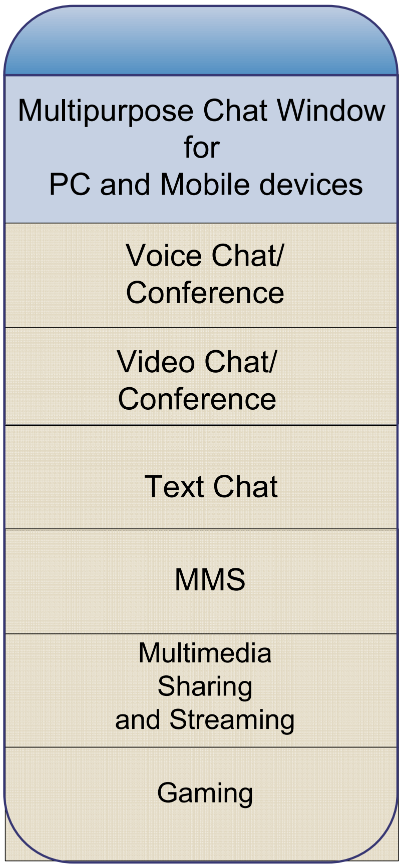


Figure 38: Universal communication service from one single window

Basically, communication networks can be classified in to two categories, such as circuit switching network (CSN) and packet switching network (PSN). The telecommunication networks mainly use the circuit-switching technique due to its service reliability and provision of fixed bandwidth. On the other hand, the internet uses the packet-switching method in order to deal with the burst traffic pattern and working over a range of situationally available bandwidth in the network. At the present time we observe that the mobile networks are in more and more craze of using packet switching technique in order to meet their heavy demand of internet and multimedia applications. Now a days, both these kinds of networks are facing the challenge of explosively increase of multimedia data traffic. As mentioned above, cell phones are now being used to transmit and receive not only voice but also high quality real-time video; desktop PCs are more over used as multimedia terminals for online streaming services or video chatting rather than as a data processing tool. Also there is huge demand for inter network communication. So technology trend moves fast towards the complete convergence of both these basic network types. And packet-switching technologies are made more and more usable in the original-circuit-switching telecommunication networks by proper transcoding and call forwarding methods by the employing suitable gateway. There have already been some practical developments of gateways [22,23,24] that successfully bridge the gaps among the traditional landline phone, 3G mobile and internet networks in order to fulfill many advanced application aspects.

The ITU-T provides the standards H.245, H.223, H.324M and 3GPP's 3G-324M standard are the main frames of an up-to-date multimedia gateway. Some open source projects such as Asterisk [25] and OpenH323 [26] etc are keen in implementing the 3G-324M, especially Asterisk is trying to integrate 3G-324M protocol stack into their soft PBX so as to provide video capability services over PSTN network.

However, there are also some commercially available 3G gateways in the market. Radvision has developed a 3G gateway called Scopia [27]. The Scopia 3G Video Gateways supports video telephony as well as video streaming between 3G-324M based mobile handsets/devices and IP based video media servers. In the meantime, Tandberg and Dilithium have also developed similar Gateways [28,29].

During this research, we too have already developed our indigenous 3G-324M-and-SIP based 3G-IP gateway in the City University of Hong Kong. Through this paper, we claim to have a step forward by achieving some success in developing a universal multimedia conferencing environment and its supporting gateway. Basically this paper is organized as follows. Section II discusses the background of our work. In section III, we present the design and implementation issues of the high efficient 3G-IP gateway and the conferencing environment. The paper concluded in the Section IV.

6.2. Background

6.2.1. Required characteristics of a huge call handling gateway: Robustness and Efficiency

Due the advanced features and services, the penetration rate of third generation mobile service (3G) has increased explosively worldwide in recent years and as per the reports at 3GSM Congress in Barcelona (Feb 2007), the total number of WCDMA and 3G-CDMA subscribers have hit 104M and 325M respectively by Dec 2006. Also in this report, it is clearly noted that the WCDMA subscriber sector is particularly growing very fast, i.e., 102% year to year.

As per our earlier discussions, if various mobile phone, landline and IP phone networks merge together, then the total number of costumers and the call traffic will increase blastingly. In that case, the core gateway should be robust enough to meet the challenge of handling incredibly large number of calls at the same time so as to ensure a good QoS to all the subscribers. Otherwise the gateway faces not only the loss of QoS, but also a severe hazard of performance bottleneck. So the main call handling and signaling protocol, i.e., H.245 should be implemented intelligently in order to help in this situation. Also the protocol should be scalable and modular one so as to be easily maintained and upgraded.

6.2.2. Core Technologies

Earlier days, H.324 provided the ITU standard for voice, video and data transmission over the traditional analog networks. But to cope with the technical advancements, 3GPP has recently recommended 3G-324M that is mainly responsible for the AV (audio and video) conversation. Other standards for video/audio codecs such as, H.263, H.261, G.723.1, AMR (Adaptive Multi Rate) [13] etc. are also used here in.

Another important technology involved is the Session Initiation Protocol (SIP). SIP is a request-response structured application-layer signaling protocol for creating, modifying, and terminating sessions with one or more participants. These sessions mean internet telephone calls, multimedia file sharing/streaming and multimedia conferences etc. So SIP is used whenever any of this session is initiated.

Although most of the commercial firms, developers and research groups use the ITU standards and recommended technologies, still there are lots of barriers among them for their inter-connectivity and inter-accessibility. We present some examples in the following subsection.

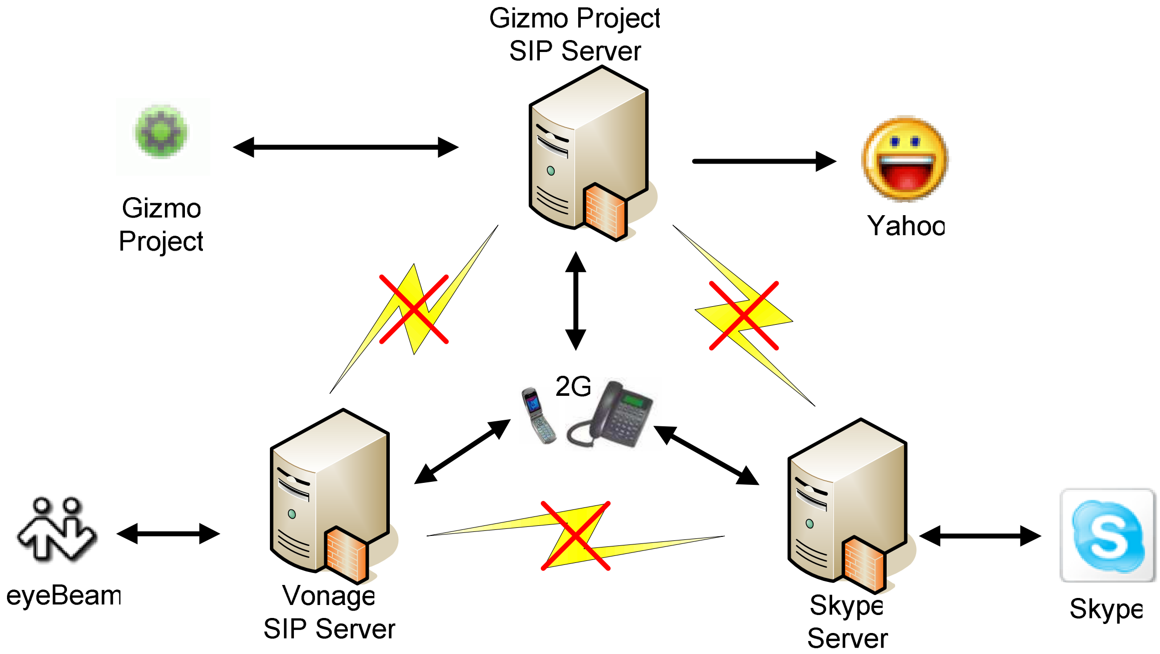


Figure 39: Lacking in inter-network connectivity

6.2.3. Need for network convergence, technologies' interlinking and a universal communication environment

As it is already discussed earlier, the main challenge in interlinking the cross technology networks is the IP-telephony (land line to 3G) gateway development and integrating various technologies after implementing the recommended standards. So far, some progress has already been made. But still the seamless connectivity and accessibility is not yet achieved either in same technology and cross domain or in cross network scenario. In order to make clear, we may discuss some very prominently pointed out examples in both the IP and mobile sectors as follows.

Internet to 3G, 2G and landline calling and vice versa is already plentily available in the market. However, in this case though voice calling is easy, still video calling opportunity is very restricted. Though most of the IP telephony servers and gateways basically work on the SIP technology, still inter server (network) communication is not possible as is shown in the Figure 39. In the instant messaging world, the most popular, Yahoo [30] and MSN [31] have already interlinked for text messaging. However they don't have the internet work voice, video and file sharing services, although they have all these features in their own domain. Eyeball [32] and PhoneGaim [33] etc are instant messaging clients that are both SIP and XMPP compliant so are capable of getting registered in all chat servers that are working on these technologies, also in the mean time they are quite efficient enough to connect to the most popular proprietary chat engines like Yahoo, MSN, Google Talk [34] etc and can provide full voice, video and text message conversation features that is available in these messengers. However, these clients are not capable of getting registered to more than one server so as to give inter messenger communication provision.

Now the mobile/PSTN case may be considered. Landlines are getting called from the IP side so easily these days, while the reverse is not so much popular. Proprietary servers like Yahoo, Skype [35] and SIP servers like Gizmo [36] etc have made the call in/out services fancy these days. But still the 3G video calling is till a dream for the common users although these technologies are already lunched in the market. Though the number of 3G users and IP chat engine users thought the world is extremely high, mobile to IP (and vice versa) direct chatting (text, voice, video) is still so rare.

The Table10 presents a comprehensive comparison of the available services and their interoperability. From this table, it is visual obvious that excepting the IP-IP same domain and mobile-mobile case, multimedia conversation and services are either restricted or not available at all.

Even when the networks will be converged and technologies will be interlinked, there will be a need for a universal communication environment (Figure 38) from which, the user can communicate to any where either from his PC or from his mobile handset/device.

6.3 Some Success in Design and Implementation

As it is already discussed in previous sections, for the universal multimedia communication, a universal gateway to handle incredibly huge number of calls and to support the multiple transcoding is essentially required. Keeping this idea in mind, during this research, we have spent some effort to achieve some success to provide robustness and efficiency to a gateway. We too consider other technical issues like compatibility, scalability and maintainability etc.

6.3.1 Robustness and Efficiency

Table 10: Possibility of Inter-network

Multipurpose Chat Engine for PC and Mobile devices							
Chat Type / Connectivity	Text	Voice	Video	Conf	MMS	Share / stream	Game
IP-IP (Same domain)	✓	✓	✓	✓	✓	✓	✓
IP-IP (Inter domain)	✓	✓			✓		
IP-Mob	✓	✓	✓			✓	
Mob-IP	✓	✓	✓			✓	✓
IP-PSTN		✓					
PSTN-IP		✓					

During this research, we have studied that, in high traffic condition, for handling large number of call concurrently and to provide good conversational quality (after all the QoS), H.245 is the only vital protocol which should be taken care of properly. According to ITU-T standards, H.245 is a control channel protocol capable of conveying information for multimedia communication. In voice and video telephony as well as in VoIP, this sub-protocol (of 3G-324M) is basically responsible for call initialization, setup and for continuing the conversation. This sub-protocol manages information exchange through a set of predefined messages. Some members of this message set are vital for the call initialization and setup, while some are responsible for continuing the conversation. So any attempt to improve the call handling procedure and a good QoS in conversation, the H.245 performance should be improved by implementing it's message exchange process efficiently. As this sub-protocol is responsible for the call setup and conversation continuation, through its message exchanges, its efficiency improvement will eventually lead to make the gateway robust to handle large number of calls simultaneously with shorter call setup time and better conversation quality. We have an innovative way of implementing the H.245 so that the gateway's call handling capacity increases amazingly with increasing number of calls passing through the gateway. In our method, we pre-compile the H.245 message sets in case of the first few calls and reuse them in subsequent calls. That is how we save our call setup time and can make our gateway work smoothly even up to 10 Million calls at same time. It is quite a high no of calls that a gateway may possibly face to handle concurrently in busy hours. The *performance improvement* is the ratio of call setup time before and after our special algorithm is applied to the H.245 standard while implementing it. The test calls in our experiment were machine generated. Also through this experiment, we do claim that not only the call setup time will be reduced, but also the conversation quality will be better as the call continuation message handling also is implemented through the same algorithm. Thus, not only the reliability, but also the QoS is claimed to be improved, which can be tested only when the gateway is deployed in real life call handling. So this gives a strong boost towards daring to design a universal gateway that we speculate in this paper.

6.3.2 A unique multimedia conference environment

Though there are so many soft clients are available in the market to chat multimedia PC-PC, some clients are also available to make 3G video phone calls from PC and vice versa. But to the best of our knowledge, so far there is no single package chatting environment marketly available, in which, PC-PC bidirectional multimedia chatting, PC-2G bidirectional phone calling and PC-3G bidirectional video calling is possible. During this research, we have successfully integrated all these techniques to a single window facility. It is a step forward in the way of attaining the universal communication environment (IP client) that is described in the Figure 38. The soft client (IP) provides means to realize the features and functionality of our gateway.

6.3.3 Other technical factors

In principle, the other major technical issues in designing such a giant gateway must be its scalability, adaptability, compatibility, interoperability and maintain ace-upgrading etc. The gateway protocol structure should be modular and scalable for the ease of maintenance and modifications in its large scale deployment. Also in the gateway, the consideration of various hardware and software technologies should be taken care of so that it can be compatible with large number of infrastructure used world wide and also too many brands of mobile devices, PCs and handsets. Compatibility just gives the basic requirement of the call setup. But for a good quality conversation, the transcending technique and the codecs must be efficient enough. Security is also another factor both technical and management side of the gateway and it should also be looked after. In our gateway development process, we have given enough attention to these factors.

6.4 Chapter Summary

In this paper, we have presented a brief survey on the scenario of the current IP-telephony techniques. Induced by our survey summery, we intuitively suggest for a universal conferencing environment and its supporting gateway. We too have presented our success (some) in designing such a gateway and the soft client (for IP side and is not released to public) that is still unique in the so crowded soft telephony world. We claim it as a major step forward for achieving the unification of various communication networks and technologies. Still the features like multimedia streaming and sharing are yet to be added to our soft client for IP while our communicator client for mobile devices is still under development.

However, in our understanding, to achieve the complete convergence of networks face the hurdle of getting together the service providers too. It involves lots of management and business policy matter and that is beyond the scope this work. Also it involves matters like servers, database, security, authentication and international routing management issues. But we feel these matters will no more be problems when the technical development side of the universal communication environment will be ready!

Chapter 7. SUMMARY AND FUTURE WORK

7.1 Thesis Summary

This thesis presented our innovative work in developing a highly efficient PC-to-3G call gateway in which 3G-324M and SIP are two major protocols used. The thesis firstly introduces the evolution of 3G-324M and then proceeds to detailed introduction to H.245 and H.223.

Since H.245 and H.223 are two core components of 3G-324M so that the major optimization work in this thesis are focusing in enhancing processing speed of these two components. The first proposed algorithm is an efficient H.245 message processing for multiple call handling in 3G Gateway. This algorithm innovatively uses pre-compile method to pre-compile and reuse a set of frequently used messages so as to reduce run time compilation time. This algorithm is experimentally verified to be suitable for high traffic environment. The second proposed algorithm is an efficient H.223 data dispatching for packet multiplexing/demultiplexing procedure to speed up the message multiplexing and demultiplexing. As in H.223, packet multiplexing/demultiplexing based on nested multiplex descriptor processing is very time consuming, in order to tackle with this problem, conversion of nested multiplex entry into direct multiplex string is suggested so as to avoid nested searching of corresponding. The experimental data shows a good performance enhancement by using direct multiplex string.

3G-324M only works in circuit switched network but internet is the packet switched network. In order to make the gateway operating over internet as well, SIP is chosen to be another protocol to be embedded in the gateway. SIP is a fast emerging protocol for session initiation and has been adopted in many communication applications especially VoIP. In addition to its similarity signaling ability to H.245 in 3G-324M, its simple operation enables fast development. With both 3G-324M and SIP in the gateway is not enough for it to operate from internet to 3G network, the transcoding of signal, media stream are still required internally. In this thesis, the system architecture and transcoding subsystem of this gateway were described also. Right following presentation of our gateway system, we have discussed our innovative research about suggestions for next generation multimedia communication model. The model is about the features of clients to be used in the next generation communication network and the abilities of next generation communication gateway.

7.2 Future Work

7.2.1 Separation of H.245 SE state management

From Figure 40 we can see that each call session carries at least one instance of corresponding signaling entities. Some signaling entities are only used during call setup phase while some are persistent for the whole call session. In a gateway, hundreds or thousands call sessions exist at one moment. That creates at least hundreds or thousands instances of different types of signaling entity as shown in Figure 40. This will eventually lead to another big resource burden for the gateway.

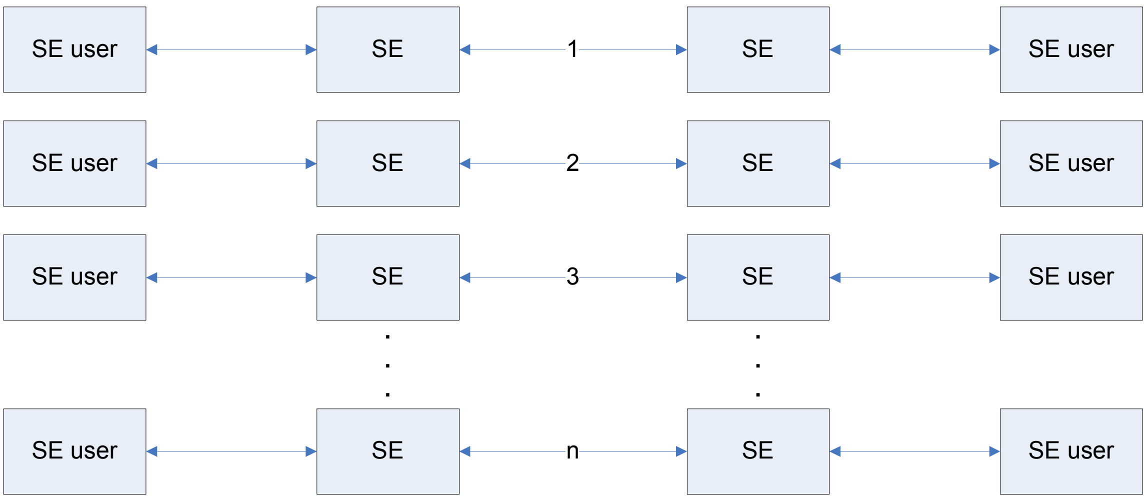


Figure 40: n instance of SE co-existence

However, SE user communicates with SE using a set of defined primitives, SE does not carry other information when active but just its current state! By using this characteristic, if the state can be separated out from SE to manage it separately, then probably that the same type of

SE can share one instance of this SE process so as to save system resource! The idea is shown in Figure 41 and need to be verified experimentally. Moreover, Figure 42 illustrate the possible implementation of enhanced SE so that it state can be separated.

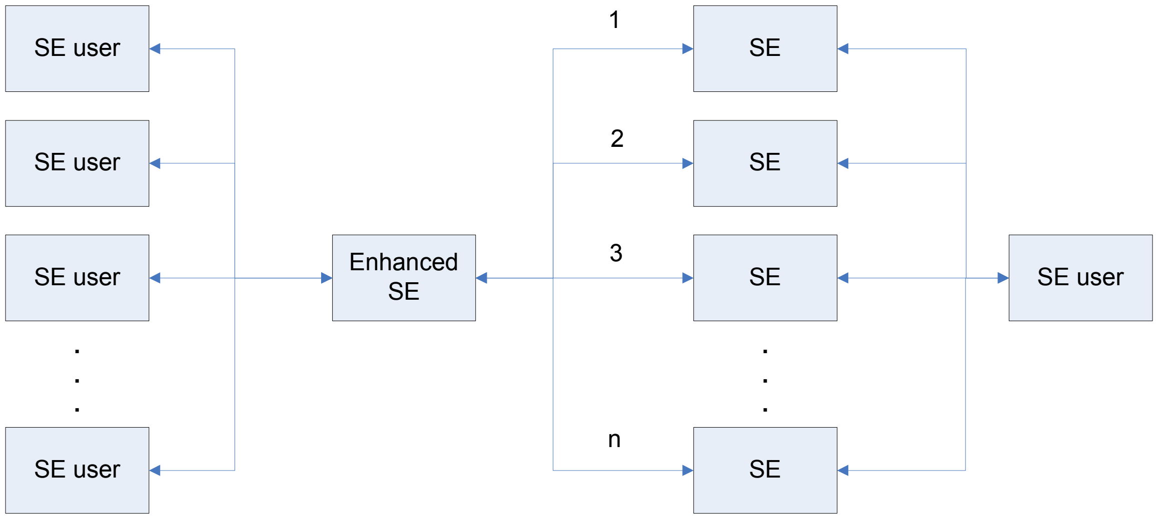


Figure 41: Multiplex SE users share one Enhanced SE

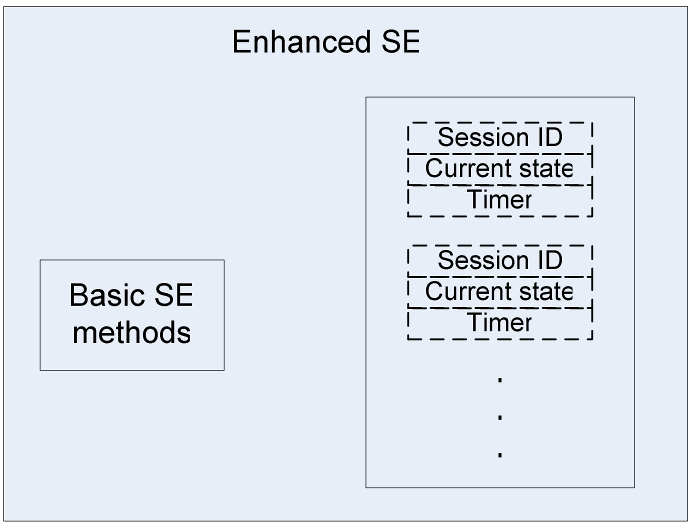


Figure 42: Possible implementation of Enhanced SE

7.2.2 Maximizing throughput

In 3G-324M, if sending terminal has nothing to transmit to receiving terminal, it transmits stuffing bits. Stuffing bits is the starting/closing flag of a packet. Sending stuffing bits means transmitting blank packets, which consist of flags only (no payload), to remote terminal. During experiments while we are developing the PC-to-3G video call gateway, we have found that the stuffing bit output of a 3G-324M communication session is incredibly high compared with normal voice and video data output. Figure 43 is the statistics generated by the Dilithium 3G Network Analyzer. In this Figure, blue represents stuffing bits and it occupies quite a bit area compared with other three types of useful data such video, audio and particularly control. We are still investigating the reason causing this problem, however, we believe that this is a worth doing work that probably will further increase both audio and video quality.

Figure 43: Bandwidth usage captured by Dilithium 3G Network Analyzer

REFERENCES

[1]. http://home.businesswire.com/portal/site/home/?epi_menuItemID=989a6827590d7dda9cdf6023a0908a0c&epi_menuID=c791260db682611740b28e347a808a0c&epi_baseMenuID=384979e8cc48c441ef0130f5c6908a0c&ndmViewId=news_view&newsLang=en&div=-762569457&newsId=20070216005300

[2]. http://www.cdg.org/worldwide/cdma_world_subscriber.asp

[3]. The 3rd Generation Partnership Project (3GPP): http://www.3gpp.org

[4]. ITU-T Rec. H.324, Terminal for low bit rate multimedia communication, March 2002

[5]. ITU-R Rec. PDNR WP8F, Vision, Framework and Overall Objectives of the Future Development of IMT-2000 and Systems beyond IMT-2000, 2002.

[6]. ITU-T Rec. H.245, Control protocol for multimedia communication, July 2003.

[7]. ITU-T Rec. H.223, Multiplexing protocol for low bit rate mobile multimedia communication, July 2001.

[8]. ITU-T Recommendation G.723.1 (1996) - Dual rate speech coder for multimedia communication transmitting at 5.3 and 6.3 Kbit/s.

[9]. ITU-T Recommendation H.263 (1998) - Video coding for low bit rate communication.

[10]. ITU-T Recommendation H.261 (1993) - Video Codec for audiovisual services at px64 Kbit/s.

[11]. ITU-T Recommendation T.120 (1996) - Data protocols for multimedia conferencing - under development.

[12]. ITU-T Recommendation X.680 (1994): Information Technology - Abstract Syntax Notation One (ASN.1) - Specification of basic notation.

[13]. ITU-T Recommendation X.691 (1996): Information Technology - ASN.1 Encoding Rules - Specification of Packed Encoding Rules (PER).

[14]. Sanghyun Park, Jongsung Yoon, Soon-Key Jung, Sung-Jea Ko, Implementation and performance analysis of mobile multimedia terminal (H.324M), TENCON 99 Proceedings of the IEEE Region 10 Conference Volume 2, 15-17 Sept. 1999 Page(s):978 - 981 vol.2

[15]. Basso. A,. Kalva. H, Beyond 3G video mobile conversational services: an overview of 3G-324M based messaging and streaming, Multimedia Software Engineering, 2004. Proceedings. IEEE Sixth International Symposium on 13-15 Dec. 2004 Page(s):474 - 481

[16]. Vesilo, R. Recursive descent techniques for ASN.1 decoding and encoding, TENCON '92. Technology Enabling Tomorrow: Computers, Communications and Automation towards the 21st Century. 1992 IEEE Region 10 International Conference. 11-13 Nov. 1992 Page(s):327 - 331 vol.1

[17]. Lv Qian, Huang Benxiong; Wang Furong, The mechanism of ASN.1 encoding and decoding implementation in network protocols, Information Technology: Coding and Computing, 2003. Proceedings. ITCC 2003. International Conference on 28-30 April 2003 Page(s):622 – 626

[18]. Yen-Chi Lee, Altunbasak, Y., Mersereau, R.M., Multiframe error concealment for PEG-coded video delivery over error-prone networks, Image Processing, IEEE Transactions on Volume 11, Issue 11, Nov. 2002 Page(s):1314 – 1331

[19]. Miki. T, OTAN. S, Kawahara. T, Error resilience features of MPEG-4 audio-visual coding and their application to 3G multimedia terminals, Signal Processing Proceedings, 2000. WCCC-ICSP 2000. 5th International Conference on Volume 1, 21-25 Aug. 2000 Page(s):40 - 43 vol.1

[20]. http://www.ietf.org/rfc/rfc3261.txt

[21]. Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Internet Protocol (IP) multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3 (3GPP TS 24.229 version 7.7.0 Release 7)

[22]. http://www.radvision.com/Products/3GProductsApplications/

[23]. http://www.tandberg.com/products/tandberg_gateways.jsp

[24]. http://www.dilithiumnetworks.com/products/DTG3000.htm

[25]. http://www.asterisk.org/

[26]. http://www.openh223.org

[27]. http://www.abiresearch.com/

[28]. http://home.businesswire.com/portal/site/home/?epi_menuItemID=989a6827590d7dda9cdf6023a0908a0c&epi_menuID=c791260db682611740b28e347a808a0c&epi_baseMenuID=384979e8cc48c441ef0130f5c6908a0c&ndmViewId=news_view&newsLang=en&div=-762569457&newsId=20070216005300

[29]. http://www.3gpp.org/ftp/Specs/html-info/26071.htm

[30]. http://messenger.yahoo.com/

[31]. http://get.live.com/messenger/overview

[32]. http://www.eyeball.com/products/messenger.html

[33]. http://www.phonegaim.com/

[34]. http://www.google.com/talk/

[35]. http://www.skype.com/

[36]. http://www.gizmoproject.com

[37]. Samuel C. Yang, 3G CDMA2000: wireless system engineering, Boston, Artech House, c2004.

[38]. Gonzalo Camarillo, Miguel A. García-Martín, The 3G IP multimedia subsystem (IMS) : merging the Internet and the cellular worlds, 2nd ed. Wiley & Sons, c2006.

[39]. David J. Myers, Mobile video telephony: for 3G wireless nerworks, McGraw-Hill, c2005.

[40]. Alan B. Johnston, SIP: understanding the Session Initiation Protocol, 2nd ed, Artech House, c2004.