CELLULAR LOCATION UPDATE USING AN INTELLIGENT NETWORK PLATFORM.

by

NC DAWOOD GHAFOOR
NC ARSALAN HAMEED
PC KAMRAN HAYAT KHAN
PC TAUSEEF MEHMOOD
(BESE-8)

Submitted for Bachelors in Engineering (Computer Software)
Military College Of Signals

Project Advisor
Dr. Saeed Murtaza

ABSTRACT

The principle behind Intelligent Networks (IN) is the separation of call and bearer control from service control. This enables the rapid introduction of new services, features and the ability to offer integrated service packages thereby reducing the reliance on switch manufacturers for the provision of new services.

Global System for Mobile communications (GSM) is the accepted standard for mobile communications not only in Europe, but world wide. GSM is also one of the first networks with a standardised modularised approach to its architecture.

This thesis presents an architecture to integrate GSM and IN networks enabling the provision of GSM mobility services from an IN platform. The approach is to move *mobility* provision and management functions within a GSM network to an IN platform, so providing *mobility* as an IN service rather than a GSM specific service.

This proposal will enable the rapid creation of mobility based value added services. Furthermore the proposed IN - GSM integration scenario can be seen as an evolutionary step towards third generation mobile system UMTS.

The approach taken is to transform existing GSM mobility procedures such that they can fit the IN Service Independent Building blocks (SIB) architecture, thereby coexisting with IN SIBs on a Service Control Point. The GSM switching and radio access network is retained to enable the maximum reuse of the existing system.

The proposed project presents results from simulation studies carried out to compare the performance of the proposed architecture against the GSM network. Signalling protocol based simulations models were developed on VC# and MATLAB for both the proposed architecture. The GSM simulation model was analyzed based on mathematical principles and different KPIs (Key Performance Indicators) were monitored and evaluated. Finally the results were justified by mathematical implication and reasoning.

TABLE OF CONTENTS

INTRODUCTION

In this document a new architectural approach based on ***Intelligent Networks*** (IN) is presented for the core network of a GSM mobile network. This approach provides a means for the evolution of the second generation mobile system GSM to ***Universal Mobile Telecommunications System*** (UMTS), the third generation mobile communication system. The advantage of this approach over the existing GSM core network architecture is its use of IN's modular and distributed concepts. The new architecture will offer a platform for:

- The rapid and easy introduction of new services in the GSM network.
- An UMTS control platform evolving from GSM.
- ˜Backward compatibility of the GSM radio access network in an UMTS environment.
- ˜The integration of GSM mobility management services with fixed network IN services.

Simulation models were created to study the effect of moving Cellular Location Update to an IN platform on the control network and on the quality of service as seen by the user.

GSM designed in the 1980s has a modular approach to its design and encompasses features such as distributed processing and functional separation of call control from the switch. GSM is a digital feature rich network. These factors have combined to give rise to a popular standard for mobile communications which, although initially designed for Europe, has become a world standard.

GSM provides excellent voice services and supplementary features are included in its standards, but it lacks the environment for the creation of new services as offered by IN, as it predates IN standards. The ability to create and offer new services rapidly is a primary discriminator for operators and service providers. To enable rapid and easy service creation in GSM, an Intelligent Network environment needs to be introduced to the GSM network. The solution offered by the industry is the ***Customised Application for Mobile network Enhanced Logic*** (CAMEL) which offers GSM calls access to IN services in the users home network. CAMEL provides a bridge between two separate networks, GSM and IN. In CAMEL there is no true integration of the two networks and call processing is suspended in GSM while IN services are executed. This is adequate for the type of services defined in CS1, but as the complexity of services grow a truly integrated solution is required: an integrated solution that will be of mutual benefit to both networks.

UMTS is in the process of being specified by European Telecommunication Standards Institute (ETSI). UMTS is a system capable of supporting a variety of mobile access networks sharing a common core network. Compared to today's networks, a stronger integration of mobile and fixed networks is expected of the UMTS network; it also supports access bandwidths of up to 2Mbits/s. The scale of GSM's success and the investment made in GSM has necessitated a new line of thought: it is now widely accepted that the air interface for UMTS will be (and needs to be) revolutionary while the UMTS core network will take an evolutionary path from the GSM core network. Furthermore GSM is expected to provide traditional voice services into the foreseeable future, while UMTS at its inception will be used mostly for provision of multimedia and high bit rate data services. Any future third generation network must provide backward compatibility to GSM. It is envisaged that both GSM and UMTS networks will coexist, preferably sharing a common core control network and hence any new core network must offer backward compatibility with the GSM radio access network.

The UMTS core network is based on the IN concept. Mobility and service provisioning in UMTS is offered from the IN platform as added intelligence. Therefore, the GSM core network and IN networks must be integrated to form the basis for the evolution to the UMTS architecture. The requirements for such an architecture as outlined in are that it must:

- encompass the existing GSM and IN architectures;
- reduce the time and effort needed to enhance these standards, by focusing available expertise / resources onto a common framework, thus avoiding

replication and minimising cost.

- protect investment to date.
- provide a modular structure so that operators / manufactures can pick and choose
  what they implement.
- provide a ubiquitous, standard platform, but with interfaces, such that operators and manufacturers may compete by offering proprietary and differentiating features.
- provide a “single track” approach for evolving towards UMTS.

The key drivers for the integration of GSM and IN from the GSM perspective are, therefore, the need for GSM evolution to UMTS while maintaining backward compatibility with the GSM access network and the need for a service provision environment in GSM.

IN is an ‘overlay’ network, which is independent of the
access network and this is exploited in UMTS to bring about the closer integration of mobile and fixed networks and manage multiple access networks from a single control platform. It is essential to note the difference between the IN concept and the IN implementation today. The IN conceptual model is expected to remain constant and provide an uniform approach to IN implementations, which will evolve with the technology of the day to meet increasing service and performance requirements. IN is restricted to some extent today by the lack of technology to support it, rather than being limited by the concept itself.

The key to IN lies in the ***Service Independent Building Block*** (SIBs) these are reusable network-wide units from which services are composed. New
services are implemented using the SIBs from the existing library of SIBs and new SIBS are only introduced as and when required. This technique results in a fast and efficient method of service implementation.

In this thesis, GSM call and mobility functionality are defined in terms of SIBs, which will enable the close integration of these functionalities with supplementary service SIBs. The result is the ability to use GSM functionality in any other fixed or wireless networks based on IN for mobility management, security and call

control. Hence service such as *Universal Personal Telecommunications* (UPT) can be provided using GSM SIBs.
The main hindrance to GSM-IN integration is signalling. GSM uses the ***Mobile Application Part*** (MAP) , specifically designed for signalling in the GSM core network. Intelligent networks use the ***Intelligent Network Application Part*** (INAP) developed with services and applications in mind . Both protocols are based on the ITU *Common Channel Signalling System No7* (SS7) and use the Transaction Capabilities offered by SS7. Although they are both similar in principle, the packaging used is different. Due to the limitations of processing speed and network capabilities at the time GSM standards were defined, GSM procedures were 'hard-wired' for optimisation by offering it as one package; IN procedures are modular in definition. The distinction between the two protocols is made at the beginning of a signalling transaction when protocol identifiers are used to distinguish between INAP and MAP. From then onwards, protocol identifiers are not used and only message numbers are used. Both MAP and INAP messages share the same range of message identity numbers, so that simply integrating both protocols would result in contention.
A standardised approach is required for integrating the two protocols. The UMTS protocol will contain elements of both the INAP and MAP. The current trend is that IN is being expanded to include mobility and GSM to include IN functionality, resulting in two sets of super-standards. In this thesis a 'pre-UMTS' protocol focusing on mobility management is presented, by retaining the modularity and reusability of IN and by including GSM's tried and tested mobility functionality. This brings together the best of both worlds and prevents the wheel from being reinvented.
There is no benefit in either of the protocols inventing functionality found in the other.
If the functionality required is not sufficiently supported by the existing protocols, then the existing protocols could be extend to support the requirements. The benefit is that both networks will mutually benefit from the extensions. The architecture presented in this thesis enables the evolution of the GSM core

network to support UMTS and retain the capability to continue to support the GSM radio access network from the evolved core network.

MAP was designed and developed with the specific aim of achieving speed and optimisation for the GSM network. To date no publication has investigated the performance issues related to moving GSM mobility management and call control to a more generic architecture and platform. This thesis presents results on the effect of the control network and quality of service (as perceived by the user) as a result of providing GSM functionality from an IN platform. Furthermore, the use of an IN architecture can result in the distribution of GSM functionality which may not be optimum for a mobile network. Therefore the simulation models investigate the effect the various physical IN architectures will have on the signalling network and on the GSM procedures.

There are two obvious paths for integration and evolution, GSM into IN or *vice versa*.

This thesis argues that Location Update, in principle, is independent of the access network and by no means restricted to the cellular networks. Intelligent Networks on the other hand are access network independent and offer the same range of services across the multitude of access networks, therefore absorption of GSM into IN is supported here.

Chapter 2 serves as a brief introduction to the GSM network covering elements of both the radio access network and the core network. Signalling in the various parts of the GSM network is described with emphasis on the core network signalling protocol, the *mobile application part*. The signalling procedures for GSM Location Update are discussed.

Chapter 3 introduces the concepts behind Intelligent Networks and its role in supplementary service provision today. The IN conceptual model which describes the various levels from service description to service execution across the physical entities is given. A detailed description of the signalling protocols used within the IN framework is provided, as signalling is central to the research presented in this document.

The integrated GSM / IN architecture proposed is described in Chapter

4. The working of GSM mobility management and call control from an IN platform and the resulting signalling procedures are also described and the modifications necessary to the existing GSM network are identified.

The simulation models developed by the author to investigate the performance of the architecture presented in Chapter 4 is described in Chapter 5. This chapter details the modelling of the various parts of the network, the assumptions made for the simulation study and modelling user mobility. The verification and validation of the models are discussed.

In chapter 6, the various physical architectures that could result from GSM - IN integrated case are presented. Simulation models for these architectures are discussed and the results of the simulation are studied to determine the most suited physical architecture.

In chapter 7 the approach taken in this thesis is validated and verified. A comparison with other approaches tackling the integration of GSM and IN is made. Finally, Chapter 8 concludes with the merits of this approach highlighted and its limitations identified. Areas for further work are also given.

2. THE GSM NETWORK

2.1 AN INTRODUCTION

Global System for Mobile communications (GSM) was born from the need by several European countries to introduce a common mobile communication network and overcome the limitations of the existing analogue system. The analogue system was limited in several ways, including its inability to cope with the unprecedented growth in the demand for mobile communications, the use of open channels allowing for easy 'eavesdropping' and 'cloning', the inflexibility in the introduction of value added services and the lack of a common network across Europe, among others.

In 1982 the Conférence Européenne des Postes et Télécommunications (CEPT) formed the "Groupe Spécial Mobile" (GSM) (later to be called Global System for Mobile communications) to define the standards for a new mobile communications system. Although GSM was introduced as an European specific standard, it has been adopted by several countries world wide. The system was required to allow roaming in participating countries, offer services and facilities found in other public networks and use an internationally standardised signalling system for interconnection of mobile switching centres and location registers.

In the late 1980s it was realised, the specification and implementation of GSM could not be achieved in a single instance. A limited GSM roll-out (phase 1) was effected in 1991, offering basic voice telephony only. The specifications for phase 2, an 'enhancement' to phase 1, includes new supplementary services and the introduction of half rate speech channels. GSM as a standard has been in a constant state of evolution since its inception and will continue to do so into the foreseeable future.

GSM as a network is not defined by a set of rigid and stagnant standards. It is a network not only willing to evolve, but by the very nature of its specifications it needs to evolve. These qualities embodied within GSM make the results described in this thesis feasible and a practical reality.

"A platform [GSM] which is full of hooks, mechanisms and not at least potential to continue to build on and provide mobile communications in all its possible forms and varieties. Even before Phase 2 standard has been completed, GSM has grown far beyond its original geographical "limitations" and the Global System for Mobile communication really starts to deserve its name. With Phase 2, and in particular Phase 2+, GSM will also expand far beyond its originally intended functional boundaries and open up for new applications, new access methods, new technologies and thus altogether for new categories of market, needs and users. It looks promising." Jonas Twingler, GSM co-ordinator of ETSI.

GSM is one of the first 'intelligent' networks with distributed processing, clear separation between the switch and bearer control and to use *Common Channel Signalling System No.7*. This provides GSM the hooks, mechanism and the potential to evolve and grow. This potential combined with the similarities between Intelligent Networks and GSM network architectures will be exploited will in this thesis to present an evolutionary path to a 3rd generation of mobile communication network.

Although GSM has been thoroughly covered in , a brief overview of GSM is given in this chapter, with the aim of highlighting the clear separation between GSM's radio access and core networks. The clear separation of core and access networks are vital to the evolution of any network to ensure that one is not restricted by the other and changes to one does not necessarily result in the replacement of the complete network. The evolutionary path presented in this thesis relies on this separation.

The following sections will highlight the separation between the core and access networks and will show that the key to call, service and mobility management lies in the core network and not the radio access network. For our purposes, a detailed explanation on the workings of the radio access network is not necessary and therefore will not be presented.

2.2 GSM SUB SYSTEMS

GSM architecture is composed of two main parts;

- ˜The radio access network or the *Base Station Subsystem*.
- ˜The switching, call handling and mobility management network, referred to as the

The primary object of this Project will involve the core network and this chapter will reflect this, but to present a complete picture elements of the radio access network are introduced.

Human interaction with the GSM network is via the *Mobile Station*, which consists of the ***Mobile Terminal*** (MT) and the *Subscriber Identity Module*. The mobile terminal provides the user interface and the radio connectivity to the network. The *subscriber identity module* is a smart card with information pertaining to the customer, security parameters and the ability for the network to identify the user. The separation of the mobile station into two entities allows the GSM network to cater for both terminal mobility and personal mobility. Unlike the analogue systems where a user is tied to the terminal, in GSM two users A and B can share terminal X and maintain individual billing as well. Hence *subscriber identity module* roaming (i.e. personal mobility) is catered for in GSM.

The ***Base Transceiver Station*** (BTS) or '***cell***' provides the means for two way radio communications with the mobile terminal. Any signal processing specific to the radio interface is handled by the cell. A user must keep the network informed of his whereabouts i.e. update his location, so that the network can direct calls to the users current location. The mobile terminal monitors the signal strength from the surrounding cell sites and reports the identity of the cell site with the strongest signal to the network. The cell identity is associated with the user and when the user needs to be contacted, the cell site is paged. Cell sizes vary in size from a few meters to a few kilometres in radius. In areas where the cell sizes are small, a user on the move will generate a large volume of signalling traffic in location update procedures as a result of the frequent cell boundary crossings. In order to minimise the volume of signaling traffic, cells are grouped into ***Location Areas*** (LA) and the user reports the location area currently being roamed rather than the cell identity. Now the user need only update the network when a *location area* boundary is crossed.

Several cells are managed by the ***Base Station Controller*** (BSC), which is responsible for the allocation, release and management of radio channels. The BSC is a small switch linking the several cells under its control to the ***Mobile Service switching Centre*** (MSC). The radio access network includes the mobile terminal, BTS and the BSC as illustrated in Figure 2-1.

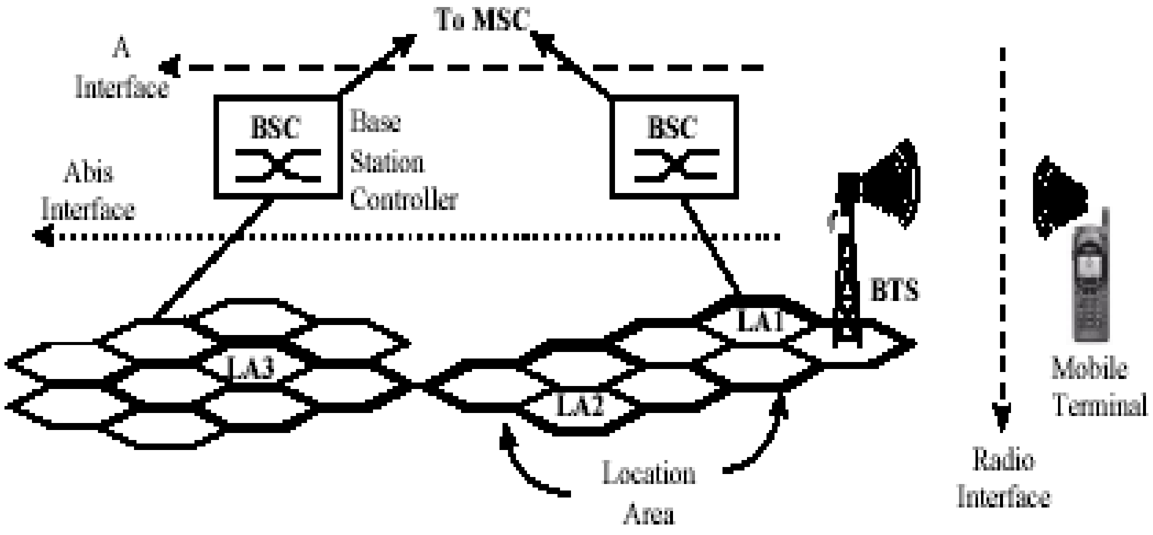


Figure 2-1 : GSM Base Station Subsystems.

The MSC is primarily a large switching centre providing connectivity between mobile stations within it's area of coverage and the outside world. A MSC's coverage is a geographical area, determined by the network operator. The mobility management functions catered for by the MSC include setting up of mobile-originating and mobileterminating calls, inter-BSC, intra-MSC and inter-MSC handovers, and location updating.

Once a mobile station comes into the coverage of a MSC, it becomes the responsibility of the ***Visitor Location Register*** (VLR) attached to the MSC. VLRs are temporary databases containing data necessary to setup calls to and from the mobile station by the MSC. Information contained within the VLR includes the *location area* being roamed, the mobile stations roaming number, the *International Mobile Subscriber Identity* and *Mobile Station ISDN Number*. The VLR keeps the ***Home Location Register*** (HLR) updated on the location of the user. The VLR's functionality is in two parts: a database for temporary storage of user data as described above and (the second part) mobility management and call handling control functionality, which includes procedures such as registration of users, call setups, authentication, location updating, among others. Although the standards draw a clear distinction between the MSC and the VLR, in practice they are implemented as one entity.

All mobile networks need to maintain a record of a user's present whereabouts in a permanent centralised location, the HLR. The HLR contains user subscription information in addition to the present location of the user. The user's subscription parameters include roaming limitations, supplementary services subscribed to, charging information, etc. A user's HLR can be identified from the user's phone number as a GSM network has several HLRs. The HLR also houses the *Authentication Centre* and the *Equipment Identity Register*. In addition, the HLR offers mobility management and database management functionality.

Upon interrogation, the HLR provides routing information to the user's present or last known location in terms of address of the roaming MSC, local mobile terminal identity and the VLR address.

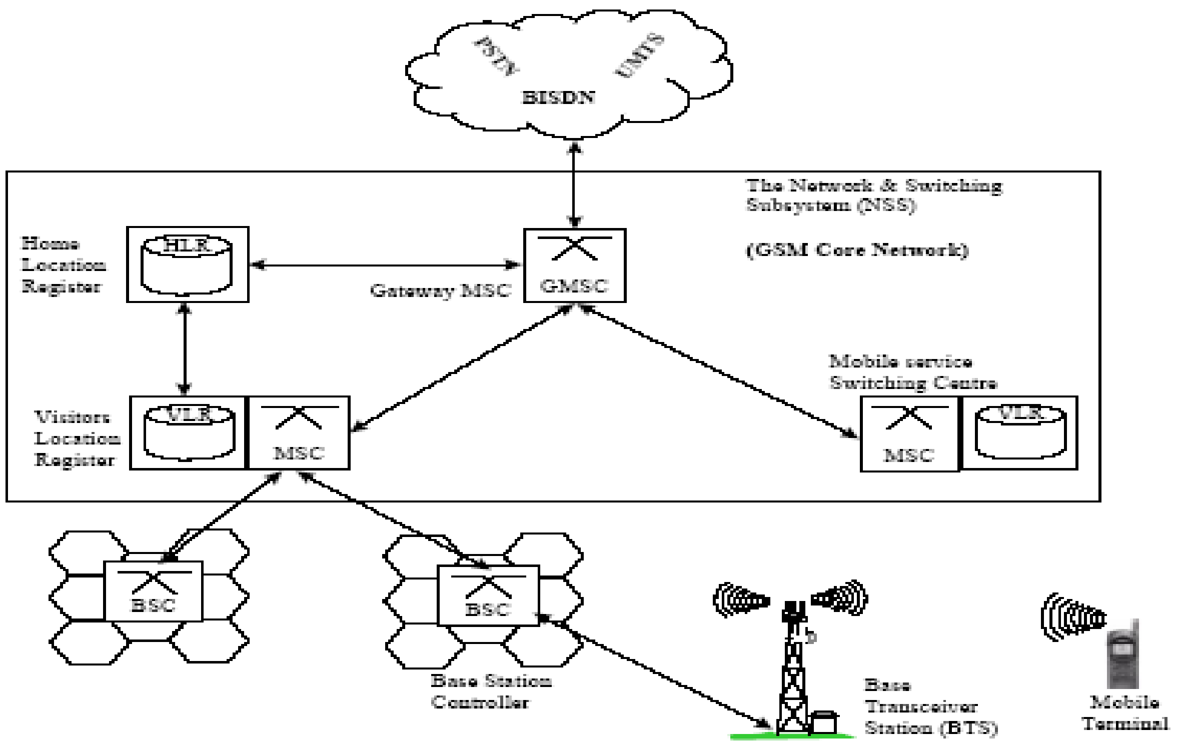


Figure 2.2-Complete GSM Architecture

For a call destined for the GSM network where the originating network cannot directly interrogate the users HLR, the call is routed to a ***Gateway MSC*** (GMSC) by the originating network. The GMSC interrogates the HLR and routes the call to the subsequent MSC. The MSC, HLR, VLR and GMSC make up the core network as illustrated in Figure 2-2.

2.3 SIGNALLING IN GSM

For a network to function successfully, it must have the ability communicate within the network and with entities outside its boundaries. The GSM network is no exception, but uses a larger variety of signalling protocols and different transport mechanisms compared to other networks. The transportation mechanisms used by GSM signalling protocols are;

- ***Link Access Protocol for Data mobile channel*** (LAPDm) is used between the mobile terminal and cell (i.e. the radio interface); this is a

GSM specific signaling standard. LAPDm makes uses of the dedicated ***Standalone Dedicated Control Channel*** (SDCCH) as its carrier over the radio interface. The data rate over the SDCCH channel is very slow ~(1Kbps) and it is only used for signalling outside a call.Once a call has been setup and a voice/data channel is available, signallingmessages are transmitted over the voice channel by ‘stealing’ a burst, i.e. the voice channel is used as a carrier for signalling data. This is referred to as the ***Fast Associated Control CHannel*** (FACCH). The faster data rate of FACCH is useful when time critical procedures such as handovers need to be conducted.

- The Abis interface between the cell sites and the BSC makes use of a derivative of NISDN signalling, ***Link Access Protocol for the ISND ‘D’ Channel*** (LAPD), this is a 64 Kbits/s signalling.
- For the ‘A’ interface and for interfaces between the various core network entities (such as VLR, HLR, MSC, GMSC), 64Kbits/s SS7 channels are used.

The signalling protocols that use the various transport mechanisms are:

1. *Radio Interface Layer 3 - Mobility Management* (RIL3-MM) - between the mobile terminal and the MSC/VLR for user location and security management.
2. *Radio Interface Layer 3 - Call Control* (RIL3-CC) - between the mobile terminal and the MSC/HLR for call control management.
3. *BSS MAnagement Part* (BSSMAP) - between the MSC and the BSC for messages specific to a connection.
4. *Mobile Application Part* (MAP) - between the various core network entities for non-call related signalling.
5. *ISDN User Part* (ISUP) or *Telephone User Part* (TUP) - between core network switches for call related signalling.

2.4 LOCATION UPDATING PROCEDURE

For a mobile network to offer connectivity to a mobile user, the location of the user must be known. The procedure of a mobile terminal informing the network of its whereabouts is referred to as location updating. The request by a mobile terminal for location updating upon entry into every new cell (which may be as small as 100 meters in radius), will place undue stain on the network in terms of excessive signalling traffic. To optimise network performance and reduce signalling load, cells are grouped into *location areas*. With *location areas*, the mobile terminal is only required to update its location when it enters a new *location area*. In addition, the network requires the mobile terminal to carry out periodic location updating. The time between periodic location updates is set by the network operators and can vary from 6 minutes to a little more than a day. There are other situations (such as a VLR failure) where location updating procedure is initiated; these situations are rare compared to others and are not described here.
A user's location is stored in three different locations in the GSM network; the *subscriber identity module*, the VLR attached to the roaming MSC and the HLR. For the purposes of routing a mobile terminating call, the HLR only stores the destination of the MSC being roamed, it being the VLR that stores the *location area* the mobile terminal is currently in. This leads to two variations in the location updating procedure

- ˜**Intra-MSC location update** : The mobile terminal moves into a new *location area* within the same MSC. In this case only the VLR needs to be informed and the HLR need not be informed as the MSC roamed is unchanged.

- ˜**Inter-MSC location update** : The mobile terminal comes into the coverage of a new *location area* controlled by a different MSC to the one being roamed. In this case the VLR associated with the new MSC needs to be informed. The new VLR will then have to update the HLR with the new MSC's address and the old VLR will have to delete the user from its records.

2.5 INTRA-MSC LOCATION UPDATE PROCEDURE

The procedure for a intra-MSC location update is given using pseudo code to describe it:

MT

{
Detects that coverage is provided by a new location area
}

Initiate a location update procedure

{
Transmit a LOCATION UPDATING REQUEST message over the standalone dedicated control channel. The message contains the *International Mobile Subscriber Identity* (IMSI) or the *Temporary Mobile Subscriber Identity* (TMSI) and the new location area identity} MSC
}
Forward LOCATION UPDATING REQUEST to the VLR.

{
Before the users records are updated with the new location, the user needs to be authenticated
}
Initiate authentication procedure

```
{
The VLR will make an authentication request to the mobile terminal
}
MT
 {
Respond to authentication request
}
```

Send AUTHENTICATION RESPONSE message

VLR **IF (Authentication outcome = success)**

Begin

The IMSI is the unique number associated with a mobile user. A GSM node is able to derive the user’s country of origin, the ***Public Land Mobile Network*** (PLMN) within the country and the HLR associated with the user from the IMSI. The SS7 address of the HLR can be derived using translation tables and
hence the routing information. The IMSI comprises of the ***Mobile Country Code*** (MCC), ***Mobile Network Code*** (MNC) (eg. Vodafone, Cellnet) and the ***Mobile Subscriber Identification Number*** (MSIN). Usually the most significant digits of the MSIN will resolve the identity of the HLR.

When a user registers on a VLR, the VLR allocates a TMSI to the user. This avoids the IMSI being used over an insecure radio channel. The TMSI is half the size of IMSI, hence it improves radio channel usage. A TMSI is associated with a LA within each MSC/VLR. When the user moves on to a new LA, a new TMSI is allocated. The TMSI contains adequate information for a GSM node to identify the issuing VLR.

The authentication functions in GSM serves to prevent unauthorised access to the network. Each user is assigned a secret key called *Ki*, which is stored in the

SIM and the user has no access to it. The *Ki* assigned to a user is also stored in the HLR. Running *Ki* and RAND (a random number varying from 0
to 2128 - 1) through an algorithm known as A3, produces a ***Signed RESult*** (SRES). So when a request for authentication is made only the RAND is transmitted to the mobile terminal. Both the mobile terminal and the network posses the same A3 algorithm, hence they should produce identical SRESs. The SRES is transmitted back to the network and compared with the SRES produced by the network, if they are identical the user is authenticated. The possibility of identifying the Ki from the SRES and RAND depends on the complexity of the A3 algorithm.

Request ciphering of radio channel.

Issue a new TMSI

{
in FORWARD NEW TMSI message
}

End

MT : **Acknowledge receipt of TMSI.**

VLR : **Update local record**.

Forward LOCATION UPDATE ACCEPTED message to mobile terminal

{
Location update procedure is complete
}

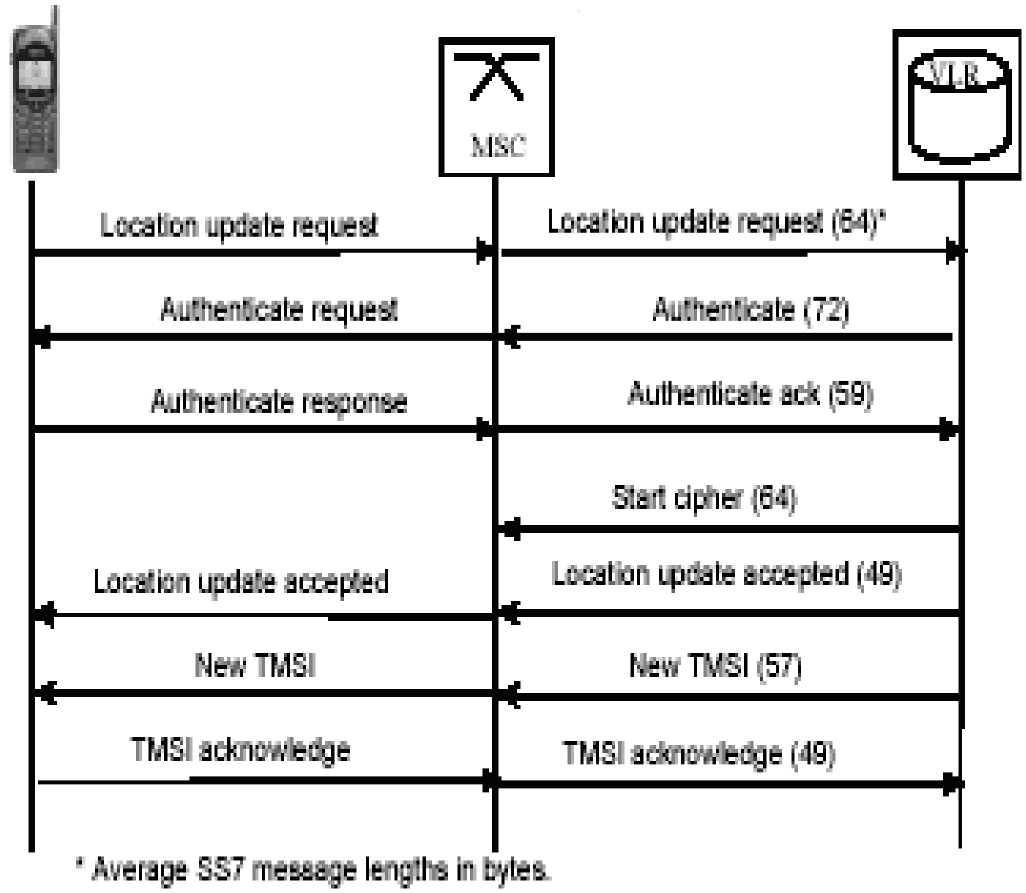


Figure 2-3: Location Update Procedure

3. INTELLIGENT NETWORKS

3.1 'INTELLIGENT' NETWORKS?

The most intelligent switches in the history of telecommunication were perhaps the first human operated telephone exchanges. The problem was, they were a little too intelligent and hence Strowger invented the first mechanical automatic switch. From the first mechanical switches, telecommunication switches, exchanges and networks have grown in size to become one of the most complex large systems in the world; in terms of complexity and in 'intelligence'.

Is the aim in telecommunications to develop a network which would have a similar level of intelligence as pre-Strowger days without the human element? The trend in telecommunications technology would suggest that. The Strowger exchange was certainly a giant step forward for automation, but an even greater step backwards for intelligence. Intelligence in the network has grown since the Strowger days, with the stored program control exchanges of the 1970s which offered supplementary services on PABXs. The 1980s saw the introduction of ISDN and the provision of supplementary services in the public networks, with the service code embedded in switches. It would appear that 'intelligence' has been available in networks for sometime, so why did we have to wait until the 1990s for Intelligent Networks?

Intelligent Networks (IN) is a misleading term; it is not as it may suggest the first introduction of intelligence to the network. IN is a concept that was introduced to solve some of the problems associated with large telecommunication systems. The problem was that all telecommunication networks needed to evolve and keep up with current technology to offer an acceptable level of service. But the telecommunication networks of the 1980s faced the following difficulties:

- Switches were at the heart of telecommunication systems, with intelligence hard coded in the switch software. Hence the network was expensive to improve or evolve and invariably the network was heavily dependent on the switch manufacturers. As a result, the introduction of new services and features were excessively time consuming.

- . The unprecedented growth in telecommunication networks in recent years has seen the introduction of a large numbers of switches into the network. If the trend of intelligence in switches continued the previous problem would be made worse.

˜The interval between the emergence of new technology is ever decreasing.Therefore the switches need to be updated more often. In practice, switches were updated infrequently and as a result new technology was introduced in leaps rather than in a gradual and continuous fashioñ̰.Network evolution was often restricted by the lack of backward compatibility and incompatibility between different manufactures.The solution came in the form of IN, where the network was compartmentalised to allow the separate evolution of the individual components. Intelligence was distributed away from the switches such that several switches could share ‘intelligence resources’. Finally bearer and service control was separated.The Intelligent Network is not a self contained network, but is a concept that could be applied to any communications network. With IÑ̰;Introduction times for new services are drastically reduced.

- ˜The reliance on the switch manufacturers for the provision of new service has been reduced, enabling the provision of a broader range of services by multiple vendors in a competitive environment.
- ˜Through distribution and compartmentalisation, the network can be brought up to date, without the need for expensive modernisation of the whole network, but by updating only the necessary elements. The IN concept is such that it maintains backward compatibility.

Intelligent networks has enabled the introduction of services such as ‘call waiting’, ‘call forwarding’, ‘freephone’ services among others.

The principle behind rapid service creation in intelligent networks is the construction of services from ***Service Independent Building Blocks*** (SIBs). SIBs are reuseable components that implement a network function. In an IN there is a library of SIBs: a new service is constructed from existing SIBs and new SIBs are only added to the library when the functionality required by the service does not exist. SIBs are resident in the ***Service Control Point*** (SCP) (the term ‘*control point*’ is also used in this thesis to refer to the SCP) , the nerve centre of the intelligent network platform.

For a service to be executed, the request for it needs to be detected and processed. The request for services are detected at the switch and the appropriate service logic is invoked at the *service control point*. The functionality within the control point is supported by the ***Service Data Point*** (SDP) , providing data essential to the provision of IN services and the ***Specialised Resource Function*** (SRF), for interfacing with the user (for such things as voice prompts) or data collection functionality.

The ultimate goal of intelligent networks is to provide services to a wide variety of access networks through a common platform, so giving them all the same range of services. The re-use of services by several access networks increases network efficiency and will mean that fixed networks and mobile networks will offer the same range of services.

Since the inception of GSM, no modifications have been made to the *mobile application part* of the signalling. This is because any future mobility protocols will be addressed as a part of intelligent networks. This thesis addresses the issues of migration of the GSM network to an intelligent network and thereby implementing GSM mobility procedures from an IN platform. This will be in line with UMTS, where mobility will be offered as added intelligence from an intelligent network platform. This chapter will serve as an introduction to elements and signalling in Intelligent Networks.

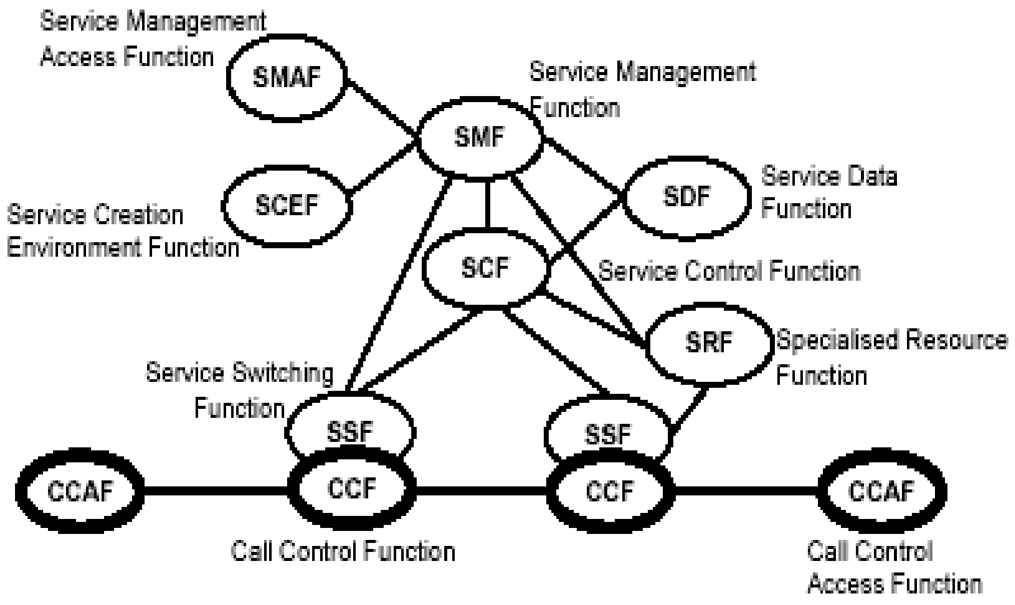

Service Management Access Function
SMAF
Service Management Function
SMF
Service Creation Environment Function
SCEF
Service Data Function
SDF
SCF
Service Control Function
SRF
Specialised Resource Function
Service Switching Function
SSF
SSF
CCAF
CCF
CCF
CCAF
Call Control Function
Call Control Access Function

Figure 3.1

3.2 IN Conceptual Model

The principle behind Intelligent Networks (IN) is the separation of call and bearer control from service control. This enables the rapid introduction of new services, features and the ability to offer integrated service packages thereby reducing the reliance on switch manufacturers for the provision of new services. Global System for Mobile communications (GSM) is the accepted standard for mobile communications not only in Europe, but world wide. GSM is also one of the first networks with a standardized modularized approach to its architecture.

Service independent building blocks (SIB) are at the heart of the IN concept. Not only are SIBs service independent, but also they are reusable in different services without the need for modification. These reusable entities allow intelligent networks to construct, test and offer new services rapidly. Most services can be constructed from the library of existing SIBs, but if a functionality is unavailable, then it is only that functionality that needs to be added to the library as a SIB or a combination of SIBs.

The *global functional plane* views the IN structure as a single entity. It describes a service or service feature and their inter-working with the basic call process using *global service logic*. The *global service logic* can be seen as the glue that holds the chain of SIBs within a service, hence it is the only service dependent entity in the *global functional plane*. Furthermore, it provides all the data needed by the various SIBs, defines the logical connectivity between SIBs, and the logical commencement and termination of services.

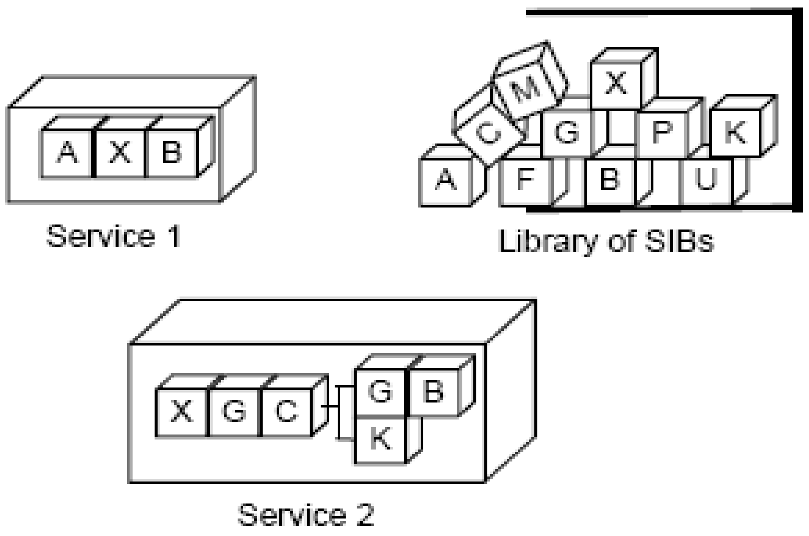


Figure 3.2 IN Concept

3.3 MOTIVATION FOR INTEGRATION

The primary objective of GSM is the provision of mobility and, originally the provision of supplementary services was not a key objective. As GSM is a digital network, facilities were made for the provision of a limited number of supplementary services internally. GSM failed to provide a dedicated environment for the development and provision of supplementary services. As the importance of supplementary services to network operators became apparent, GSM operator's developed a solution for service creation and provision in GSM networks, CAMEL. CAMEL is based on IN principles, which enables call processing to be suspended while a request for information on the completion of the call is made to the home network. The problem with CAMEL is the lack of interaction or interworking between
INAP used for service provision and MAP for mobility management. The local processing of services is also not possible with CAMEL. In the short term CAMEL is a solution to service provision in the home and roamed networks. However it is not a long term solution that would truly integrate GSM and IN. Before the integrated architecture is presented, the need for such an architecture must be justified. The justification is found by looking to the future. GSM is a second generation mobile network standard that has out-performed all expectations to become a world standard and ***Universal Mobile Telecommunications System*** (UMTS) is the next generation mobile network system that is being standardized by ETSI. Intelligent networks (Which separate service control from bearer control) will form the platform that will be used for control of mobility in UMTS. The nature of UMTS is such that from a single control platform it will offer coverage to several radio access technologies while sharing a common backbone network with the fixed network. For GSM to be one of the access networks covered by UMTS, GSM will need to migrate to UMTS.
For the migration to take place, GSM mobility management needs to be transferred to an IN platform away from the existing GSM switches, VLRs and

HLRs. The contribution of this Project is a pre-UMTS architecture in which GSM's mobility management functionality is transferred to an IN platform.

Once again the need for a pre-UMTS architecture needs to be justified against a direct step from GSM to UMTS. The step from second generation mobile telephony (GSM) to third generation mobile telephony (UMTS) will be an evolution rather than a revolution. There will be several aspects of UMTS that will be revolutionary, such as the new radio access networks offering high bandwidth radio interfaces, but these new radio access networks will coexist with existing technology. Furthermore, GSM operators view the GSM / IN integrated pre-UMTS architecture as the launching pad for UMTS. This would enable the reuse of existing infrastructure and allow for a fast track approach to UMTS leading to the gradual introduction of the UMTS network sooner rather than introducing a completely new system later.

In the GSM-IN pre-UMTS architecture described here, mobility management functionality is separated from the GSM access network and is offered as a service from the IN platform. This results in both mobility services and supplementary services existing on a single platform. This single IN platform will allow new supplementary services based on mobility or otherwise to be developed rapidly and with ease for the GSM network.Furthermore, this proposal offers a single unified platform for service creation and evolution. The structure also eliminates today's problems of having to develop services excluding GSM mobility functionality, coping with the incompatibilities between the service domain and the mobility management domain and reverting to the home network for service provision. Having eliminated the dependence of mobility management on the radio access network, it now becomes possible to offer mobility based supplementary services such *Universal Personal Telecommunications* (UPT) services using mobility procedures based on GSM. Standardization bodies are currently working on introducing mobility procedures to intelligent networks. This Project proposes the use of the tried and tested GSM mobility procedures as the basis for mobility in IN, instead of redefining mobility procedures for IN and reinventing the wheel.

However IN mobility procedures will need to address different access networks (such as DECT) and although GSM mobility procedures may not be suitable for all these new access technologies, they can be used as the basis on which other mobility services are built. The principal behind IN is the reuse of existing functionality as much as possible and the addition of new functionality only as and when it is needed. So IN only needs to add to the GSM mobility functionality when it cannot cater for mobility management in another access network by reusing GSM mobility functionality. If GSM and IN are not merged now to form a unified standard there is the danger of two sets of standards being produced, each duplicating the other's functionality. This is a situation that must surely be avoided.

3.4 Project Scope and Limitations (w.r.t IN)

The aim of this Project is not to reinvent GSM or dispose of the existing GSM system, but to reuse as much of the existing system as possible with the minimum of change. To this effect the following aims have been identified for the integrated architecture (GSM-IN) :

- Separate the radio access elements and the mobility management elements of the GSM network.

- Retain the radio access network with minimal change.

- Transform the mobility management and call handling network architecture to an IN architecture, as mobility will be offered from an IN platform.

4. GSM-IN INTEGRATION

4.1 GSM / IN INTEGRATED ARCHITECTURE

The integrated network (GSM-IN) proposed will be GSM in the radio access network with IN dominance over the control (core) network. The point of interaction between the radio access network and control network (IN) will continue to be the MSC. The MSC will include elements of both IN and GSM. The MSC will have to be equipped with functionality to recognise and manage IN service requests, i.e. with ***Service Switching Functionality*** (SSF). The MSC is the logical point for mounting the *service switching functionality*, as all mobility management messages and service messages from and to the mobile terminal are aimed at the MSC for forwarding or processing in the GSM network. The switching point between the radio access network and the outside world will remain the MSC, as there is no reason to change this. The physical node with both MSC and *service switching functionality* will be referred to as the ***Mobile Service Switching Point*** (MSSP).

The mobility functionality from the MSC, VLR and HLR will be mounted on the ***Mobile Service Control Point*****6** (MSCP). Service and mobility data stored on the HLR in GSM network will be moved to the ***Mobile Service Data Point*** (MSDP) and integrated with IN service data and user data. Hence a service data point with mobility data will be referred to as *mobile service data point*. Once this is achieved, the HLR will no longer be required.

The VLR in GSM serves two purposes: it provides mobility control functionality and serves as a temporary database where user information is stored when the user is roaming the MSC it serves. Caching user information on the VLR reduces the number of occasions on which it needs to be retrieved from the HLR, thereby reducing the total signalling traffic on the network. Once the information on the user is downloaded by the VLR, no further request for information from the HLR is necessary. With the mobility control functions moved to the MSCP, the VLR now serves only as a temporary database. In GSM-IN the temporary database attached to a MSSP will contain both IN and GSM user data and hence is referred to as MSDPtemp. It is possible to offer mobility services without the temporary database.

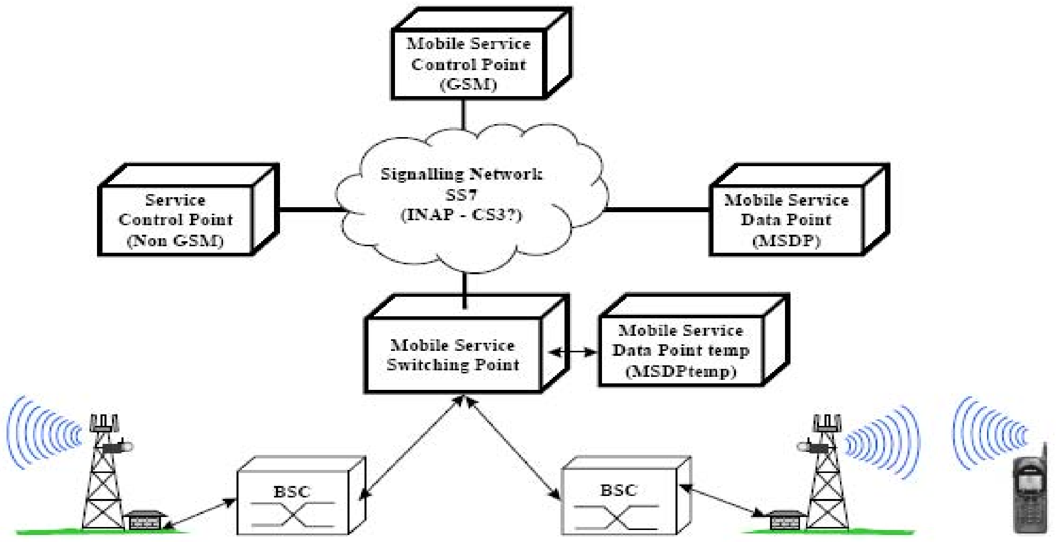

Mobile Service Control Point (GSM)
Signalling Network SS7 (INAP - CS3?)
Service Control Point (Non GSM)
Mobile Service Data Point (MSDP)
Mobile Service Switching Point
Mobile Service Data Point temp (MSDPtemp)
BSC
BSC

Figure 4.1 GSM-IN Integrated Architecture

4.2 SIGNALLING IN THE INTEGRATED ARCHITECTURE

Having proposed an integrated architecture it is necessary to consider a signaling system to make the architecture functional. An initial approach might have been to use existing MAP protocols from IN platforms. This would have been the simplest solution available, and from the routing point of view there are no conflicts at the SCCP between MAP and INAP. MAP is identified by subsystem numbers from 5 to 10, while INAP subsystem numbers are defined by the operator. Since routing of messages to the application is flexible, it will be possible to route both INAP and MAP messages to the same user part. Hence there is no conflict, and messages can be routed correctly.

The problem arises at the transaction capabilities layer. If mobility services (i.e. MAP) are to be combined with supplementary services (i.e. INAP) then it will be necessary to mix messages from both protocols during a dialogue. The problem in SS7 is that the application context (i.e. MAP or INAP) needs to be agreed on prior to the start of any communications . Defining a new MAP / INAP combined application context and using that for signalling in the integrated architecture would lead to conflicts between INAP and MAP operations; these conflicts arise as a result of both MAP and INAP using the same local values for operation, hence the local values are not unique. For example, in GSM local operation value 26 is used for the 'page' message while in IN 'EventNotificationCharging' message has the same value. This is not the only example. Hence MAP operations cannot co-exist with INAP operations without
modifications. A solution to this problem is to renumber the MAP operation local values (also suggested by , where the suggestion was made to add a prefix to all MAP local values). Another suggestion in was the use of object identifier values for each operation, making them unique, but this will add an enormous price in message length and processing overheads. The problem with these suggestion is, effectively, there will still be two protocols; also by retaining MAP, it will be necessary to retain the HLR and the VLR for carrying out some of the MAP operations.
Having considered these options and other possibilities such as
(a) different encoding schemes
(b) the use of flags and tagging, the result is that a considerable effort needs to be expended in making INAP and MAP co-exist.
What is suggested here is to take the process a step further and absorb MAP into INAP, eliminating MAP completely. The advantage of the absorption is that INAP does not yet cater for mobility but in the future, it will. If MAP is absorbed into INAP now, then INAP does not have to reinvent the wheel on mobility and the possibility of conflict between MAP procedures and INAP mobility procedures is also eliminated.

A problem posed by the definition of MAP is the need for a node involved in a MAP transaction to contain the complete set of MAP *application service elements*. By moving MAP to INAP not only is MAP broken up into smaller and more efficient components, but MAP operations are defined as INAP operations with unique operation codes eliminating conflicts with INAP. Furthermore if MAP procedures are defined in INAP generically then they can be reused to support mobility in other access networks as well.

4.3 MOBILE SERVICE CONTROL POINT

The term ***Mobile Service Control Point*** (MSCP), will be used to identify a *service control point* that has the functionality to cater for mobility services. Hence within *mobile service control points* must reside ***Mobile SIBs*** (MSIBs), which are dedicated mobility functions. This is the only difference between *mobile service control points* and *service control points*.
In IN CS-1, *service control points* cannot interact to share, negotiate or handover control between *service control points*; CS-3 will provide this functionality, which will be of benefit for the efficient provision of GSM services from an IN platform. CS-3 also introduces multiple points of control which again will be of benefit.

4.4 MOBILE SERVICE INDEPENDENT BUILDING BLOCKS

By modularising the GSM mobility procedures (location updating, handovers, call set up, etc.) described in Chapter 2, it is possible to identify commonality within the various procedures. Examples of common sub-procedures are authentication, issuing a new TMSI, paging, database enquiry and database updating among others. These sub-procedures are self contained and identical irrespective of the mobility procedure using it. Each of these sub procedures will need to be converted to a SIB, i.e. Authentication SIB, Paging SIB and TMSI SIB. For database enquiry and database updating a modified version of the IN CS-1 *service data management* (SDM) SIB can be used and this will be referred to as *mobile service data management* (MSDM) SIB.

The complete list of SIBs necessary to offer GSM mobility functionality is listed below.

- ˜Authentication SIB - Calculates RAND and forwards it to the mobile terminal and checks the calculated value against the returned value.
- ˜Cipher SIB - Instructs the radio access network to cipher a channel to mobile terminal.
- ˜TMSI SIB - Issues a new TMSI to the mobile terminal.
- ˜MSDM SIB - Used for creating, updating and deleting records on databases.
- ˜Paging SIB - Instructs the radio access network to page a mobile terminal in a specified area.
- ˜Location_Update SIB - If the location update procedure was successful then the user is informed accordingly or else the SIB is used to process any errors that may have occurred during the procedure and forward the appropriate error message to the user (Eg. Roaming not allowed).

4.5 MOBILE SERVICE DATA POINT

A **Mobile Service Data Point** (MSDP) is a *service data function,* where users service data and mobility data are stored. A users 'character set' will contain users mobility data (such as the location of MSC being roamed and roaming limitations) and supplementary service data (such as supplementary services subscribed). The advantage of a combined 'character set' is the availability of complete user information locally. The location of the *mobile service data point* will be identified from the users telephone number just as the address of the HLR is derived from the users number in GSM.

4.6 MOBILE SERVICE DATA POINT TEMPORARY

The temporary databases functionality is identical to the *mobile service data point*. The temporary database is used by a user only when roaming an associated MSC. The *service control point* will use the MSC - *mobile service data point temporary* association to find location of the *mobile service data point temporary.*

4.7 LOCATION UPDATING PROCEDURE

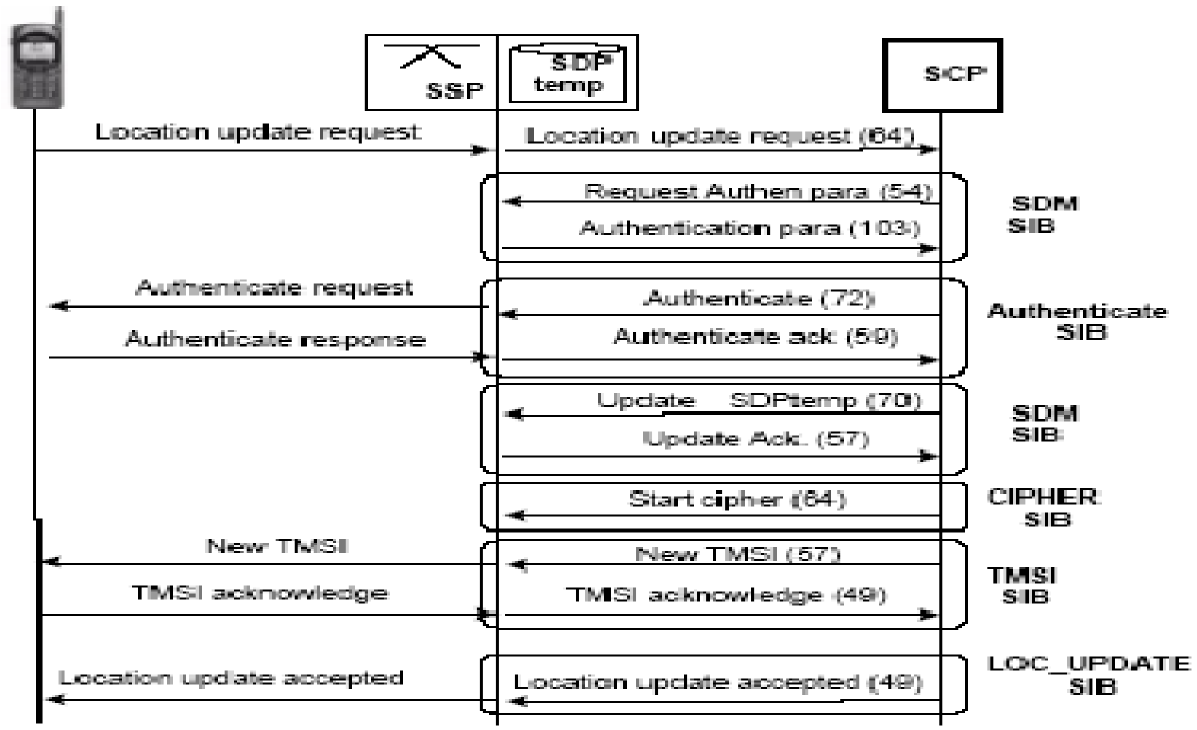


Figure 4.2 Location Update Procedure

4.8 GSM - IN: INTEGRATED APPROACH EXAMPLE

An example of a mobile originating call is used to illustrate the
workings of the GSM-IN architecture. The mobile terminal informs the network of the need for service and a signalling channel is allocated to the mobile terminal over the radio interface. The mobile terminal passes the type of service required to the MSC (i.e. the *mobile service switching point*) as a call set-up request message. The *mobile service switching point* treats this message as a trigger for the ‘call_setup_outgoing’ service and composes a INAP message with a request for the service which is then sent to the *mobile service control point*. The ‘call_setup_outgoing’ service will require information on the user and this is obtained from the *mobile service data point* or *mobile service data point temp* using the *mobile service database management* SIB . The *mobile service control point* will send an authenticate request to the user via the *mobile service switching point*, which now acts as a translation function for messages between the control point and the mobile terminal . Once authentication is successfully completed , a channel is ciphered and a new TMSI is issued by the *mobile service control point* . The number of the called party and the type of service is now sent by the mobile terminal to the *mobile service control point .* The control point checks the requested service type against the users subscription parameters and instructs the *mobile service switching point* to setup up a path to the called party with the necessary bearer capabilities and to allocate a radio channel to the mobile terminal. The call is connected through when answered.

5. Simulation and Analysis

5.1 Need for Simulation

Practical Implementation of the Project on Real network is proposed but for analysis phase Simulation analysis is consider due to its

- Low Cost
- Low Implications
- High Control
- Convenient Modification
- Easily Comprehendible Scenarios
- Possibility of testing more than one scenarios

Due to above mentioned reasons the project is simulated rather than implementing on a real time network.

5.2 Simulation Environment

Simulation is implemented in two separate parts

- VC#
- MATLAB

VC# part comprise of the software modules for example

- SCP
- SSP
- Links
- Messages
- Queues

MATLAB is used to make an analyzer primarily concerned with reading of data and presenting it in graphical format for easy analysis later on. MATLAB provides with auto-adjustable axes in which axes adjust automatically as the data defines its new range

5.3 Simulation Flow

Simulation starts by the initiation of a Location Update request by the main application.Each Location Update request sets up a whole session comprising of signaling exchange between SSP,SCP and theur respective queues over the defined link.

Each module of SSP and SCP have the capability to send the messages directly over the link but none of them can receive the message directly, rather the design feature incorporates the use of queses independently associated with SSP and SCP . Messages from link are added to respective queues which behave in FIFO manner and thus give link a chance to get idle.
Messages pass over the link randomly generating random amount of signaling at any time instant though the number of bytes that each type of message carries stay the same as per network standard. Any violation of that would be against realistic scenarios.

Each message generated has an inherent Time To Live (TTL). If TTL for a message expires while with in a queue the message is discarded and packet loss increase is incremented. Further more Link introduces a realistic delay time to each message making the simulation more realistic in nature.Only an ideal link will transfer massages from one point to another over it without any delay. This delay period further reduces the TTL of each message.

Main Appllication maintains a complete count of total message passed uptil any arbitrary time ‘x’ , total packets lost uptil that time and total signaling done till that particular time. For analysis purposes rate of change in signaling is monitored rather than specific total signaling.

5.4 Key Performance Indicators

Key Performance Indicators or KPIs are used to set parameters for network performance. On the basis of these KPIs we could categorize the network as being functional or otherwise.

Following KPIs were introduced

- Rate Of Change of Signaling over the link
- Message Loss Ratio (MLR)

5.4.1 Rate Of Change Of Signaling Over Link

As mentioned the main application maintains a count of total amount of signaling at each time instant. For analysis purposes this signaling is rather useless as long as we do not know its time derivative.

Rate of change is monitored because sudden rise or fall in the amount of signaling depicts the PREDICTABILITY of network.

If rate of signaling stays absolutely linear the predictability of network becomes 100% which is not practical under real time scenarios. But as per our observations for most of the time rate of change do not take swift hypes. If such a case would have been it shows that sometimes network is congested while in the next moment it is idle. It also reflects that resource utilization is minimal.

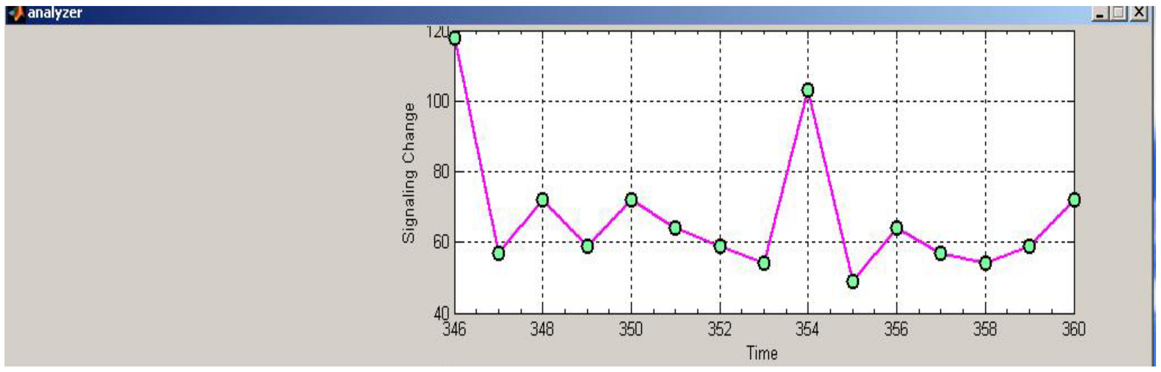


Figure 5.1 Rate of Change of Signaling

5.4.2 Message Loss Ratio (MLR)

MLR is the ratio of the total messages lost in queues to the total number of messages passed over the link uptill time 'x'.

This feature depicts the user service facility that whether user are getting the IN services or not.

In our case MLR at any time 'x' is less than MLR at any time 'x-1' for any x>0

This implies that if network continues to be functional for infinite amount of time (t→∞) , MLR tends toward zero (MLR→0). This shows that with this design the network is towards being ideal though it could never because MLR is greater than zero if even one message drops out of the queue.

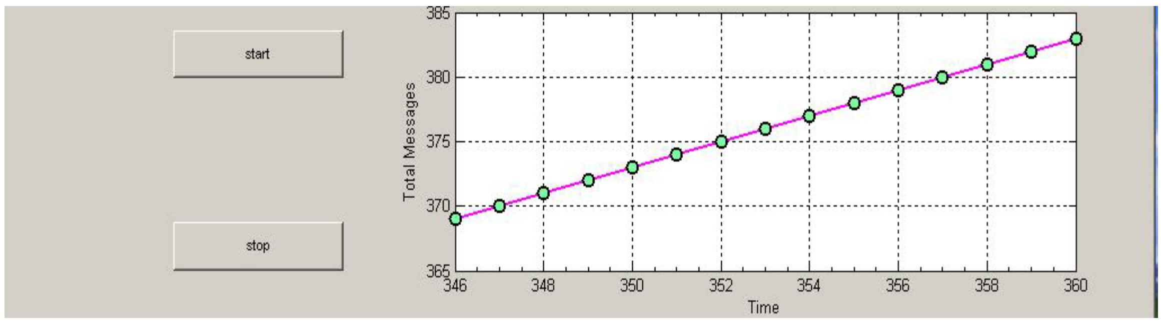


Figure 5.2 Total Message vs time

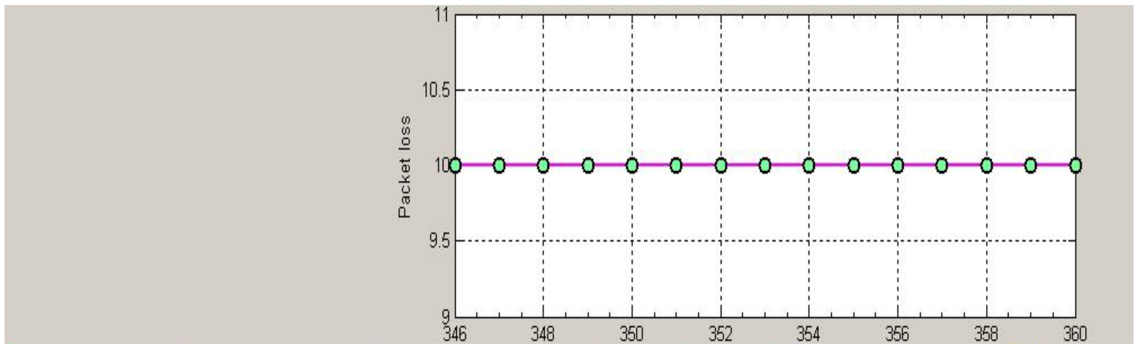


Figure 5.3 Packet Loss vs time

5.4.3 Pattern of signaling change

Even though messages tend to increase with every time ‘x’, the rate of change of signaling does not. This is because queues are introduced in SSP and SCP. This allow message to be resident in the queue leaving the link empty for transfer of further messages.

6. Project Implications

6.1 Location update-Based VAS (Value Added Services)

With this feature operator can excise better and tighter control over the user mobility if required and can also introduce Location Area Based services Like

- Subscription to service by user with in only specific location areas : it means that now operator can restrict the user to any user wanted or may be government wanted Location areas and disallow the provision of network services in those particular disallowed Location Areas thus clearly giving a upper hand to the Operator and a much better control on Operators own service

- Different call charges within Different Location areas : This could be used as a further control provision service for the operator on its own service that is the operator can apply different call charges to calls originating from different areas for may be Business Strategy purposes which would count as a provision for the Operator

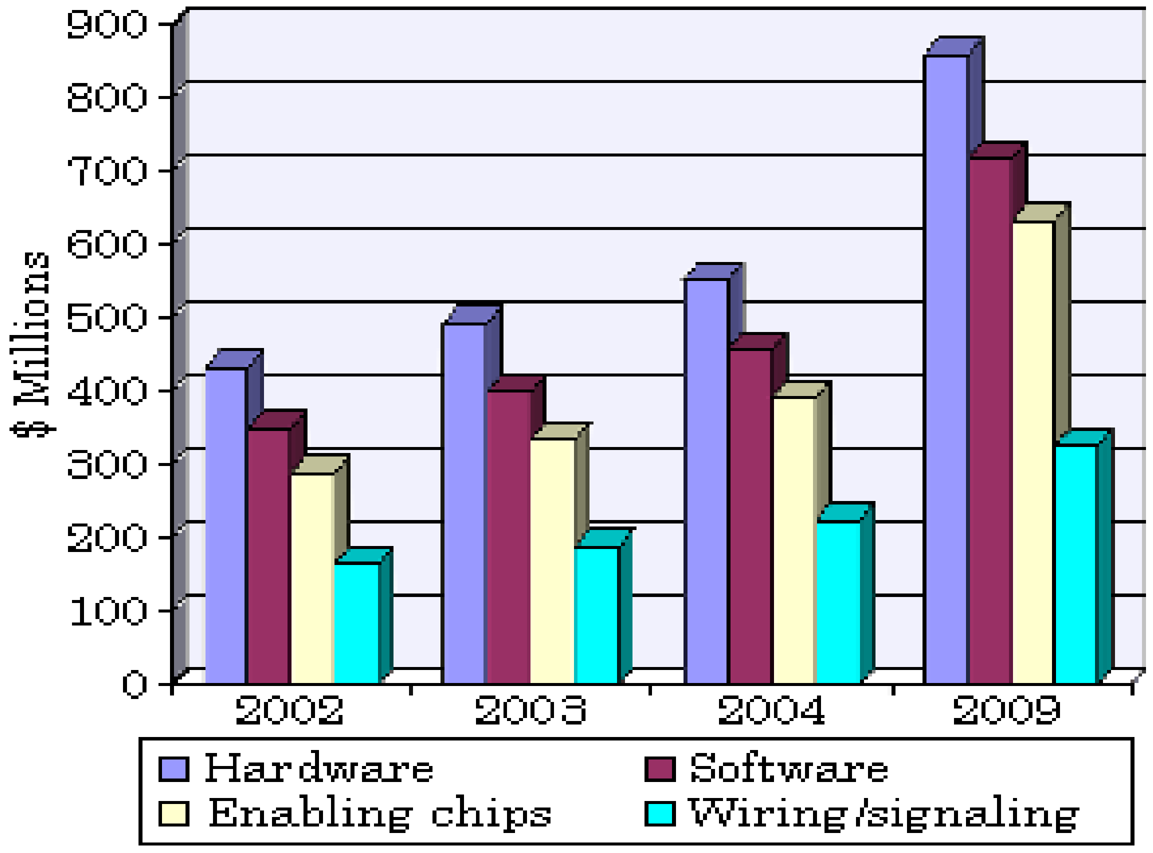
- Priority in call setup based on Location Areas : Operator would be provided with another control provision service that according to business strategy the areas that generate the bulk of the revenue for the operator could be provided with a call set up priority which means that during busy times of the day the calls originating from those Location areas would be preferred over the rest of non-prior areas.

6.2 Enhanced Network Predictability

By monitoring the rate of change of signaling over the link we can easily conclude that with this design the network predictability has enhanced based on t he analysis done in chapter 5.

6.3 Enhanced Service Availability

Since MLR decreases as time goes on the availability of the service thus becomes better and also make the network more reliable in the long span of network operation Based on Analysis in chapter 5.



Project Code

Analyser (Matlab)

```
function varargout = analyzer(varargin)
% ANALYZER M-file for analyzer.fig
%      ANALYZER, by itself, creates a new ANALYZER or raises the existing
%      singleton*.
%
%      H = ANALYZER returns the handle to a new ANALYZER or the handle to
%      the existing singleton*.
%
%      ANALYZER('CALLBACK',hObject,eventData,handles,...) calls the local
%           function named CALLBACK in ANALYZER.M with the given input
arguments.

%      ANALYZER('Property','Value',...) creates a new ANALYZER or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before analyzer_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to analyzer_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help analyzer

% Last Modified by GUIDE v2.5 17-Mar-2006 07:48:54
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @analyzer_OpeningFcn, ...
                   'gui_OutputFcn',  @analyzer_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%
```

```
% --- Executes just before analyzer is made visible.
function analyzer_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
axes(handles.axes1)
xlabel('Time')
ylabel('Signaling Change')
axes(handles.axes2)
xlabel('Time')
ylabel('Total Messages')
axes(handles.axes3)
xlabel('Time')
ylabel('Packet loss')

% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to analyzer (see VARARGIN)

% Choose default command line output for analyzer
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes analyzer wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = analyzer_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure

% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
% --- Executes on button press in start.
function start_Callback(hObject, eventdata, handles)
% hObject    handle to start (see GCBO)
global flag
flag=1
%t=timer()
%set(t,'ExecutionMode','fixedDelay','Period',10)
%t.TimerFcn = 'read(''dawood.dat'')'
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
while(flag==1)
[A B C D]=textread('project_data.txt','%u %u %u %u')
axes(handles.axes1)
plot(A,B,'-mo',...
                'LineWidth',2,...
                'MarkerEdgeColor','k',...
                'MarkerFaceColor',[.49 1 .63],...
                'MarkerSize',8)
xlabel('Time')
ylabel('Signaling Change')
            %stem(A,B)
%plot(A,B)
set(handles.axes1,'XMinorTick','on')
grid on
```

```
%%%%%%%%%%%%%%%%%%%%%%%OVERHEAD%%%%%%%%%%
%%%%%%%%%%
axes(handles.axes2)
plot(A,C,'-mo',...
                'LineWidth',2,...
                'MarkerEdgeColor','k',...
                'MarkerFaceColor',[.49 1 .63],...
                'MarkerSize',8)
  xlabel('Time')
  ylabel('Total Messages')
%stem(A,C)
%plot(A,C)
set(handles.axes2,'XMinorTick','on')
grid on
%%%%%%%%%%%%%%%%%%PACKETLOSS%%%%%%%%%%%%%%
%%%%%%
axes(handles.axes3)
plot(A,D,'-mo',...
                'LineWidth',2,...
                'MarkerEdgeColor','k',...
                'MarkerFaceColor',[.49 1 .63],...
                'MarkerSize',8)
 xlabel('Time')
 ylabel('Packet loss')
%stem(B,C)
%plot(B,C)
set(handles.axes3,'XMinorTick','on')
grid on
pause(3)
end
%
```

```
%read('dawood.dat')
%disp(B)
%start(t)
%read('dawood.dat')
%[A B C]=textread('dawood.dat','%u %u %u')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%SIGNALLING%%%%%%%%%%%
%%
% --- Executes on button press in stop.
function stop_Callback(hObject, eventdata, handles)
% hObject    handle to stop (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global flag
flag=0
```

Request Form(VC#)

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using System.IO;

namespace project
{
	/// <summary>
	/// Summary description for Form1.
	/// </summary>
	public class Form1 : System.Windows.Forms.Form
```

```
{
        private System.Windows.Forms.Button button1;
        public System.Windows.Forms.Timer timer1;
        public System.Windows.Forms.Timer timer2;
        private System.ComponentModel.IContainer components;
        public SSP ssp;
        public SCP scp;
        public System.Windows.Forms.TextBox textBox1;
        private System.Windows.Forms.TextBox textBox2;
        public Link link;
        public int a=0;
        private System.Windows.Forms.TextBox textBox3;
        private System.Windows.Forms.TextBox textBox4;
        private System.Windows.Forms.TextBox textBox5;
        public System.Windows.Forms.Timer timer3;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label label5;
        public int b=0;

        public Form1()
        {
                //
                // Required for Windows Form Designer support
                //
                InitializeComponent();

                //
```

```
			// TODO: Add any constructor code after InitializeComponent
call
			//
		}

		/// <summary>
		/// Clean up any resources being used.
		/// </summary>
		protected override void Dispose( bool disposing )
		{
			if( disposing )
			{
				if (components != null)
				{
					components.Dispose();
				}
			}
			base.Dispose( disposing );
		}

		#region Windows Form Designer generated code
		/// <summary>
		/// Required method for Designer support - do not modify
		/// the contents of this method with the code editor.
		/// </summary>
		private void InitializeComponent()
		{
			this.components                    =                    new
System.ComponentModel.Container();
			this.button1 = new System.Windows.Forms.Button();
			this.timer1 = new
System.Windows.Forms.Timer(this.components);
			this.timer2 = new
System.Windows.Forms.Timer(this.components);
			this.textBox1 = new System.Windows.Forms.TextBox();
			this.textBox2 = new System.Windows.Forms.TextBox();
			this.textBox3 = new System.Windows.Forms.TextBox();
			this.textBox4 = new System.Windows.Forms.TextBox();
			this.textBox5 = new System.Windows.Forms.TextBox();
			this.timer3 = new
System.Windows.Forms.Timer(this.components);
			this.button2 = new System.Windows.Forms.Button();
			this.label1 = new System.Windows.Forms.Label();
			this.label2 = new System.Windows.Forms.Label();
			this.label3 = new System.Windows.Forms.Label();
			this.label4 = new System.Windows.Forms.Label();
			this.label5 = new System.Windows.Forms.Label();
			this.SuspendLayout();
			//
			// button1
			//
			this.button1.Font = new System.Drawing.Font("Microsoft
Sans Serif", 9.75F, System.Drawing.FontStyle.Bold,
System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
			this.button1.Location = new System.Drawing.Point(40, 232);
			this.button1.Name = "button1";
			this.button1.Size = new System.Drawing.Size(168, 24);
			this.button1.TabIndex = 0;
			this.button1.Text = "submit LUREQUEST";
			this.button1.Click += new
System.EventHandler(this.button1_Click);
```

```
			//
			// timer1
			//
			this.timer1.Interval = 1000;
			this.timer1.Tick                    +=                    new
System.EventHandler(this.timer1_Tick);
			//
			// timer2
			//
			this.timer2.Interval = 2000;
			this.timer2.Tick                    +=                    new
System.EventHandler(this.timer2_Tick);
			//
			// textBox1
			//
			this.textBox1.Location   =   new   System.Drawing.Point(144,
136);
			this.textBox1.Name = "textBox1";
			this.textBox1.Size = new System.Drawing.Size(336, 22);
			this.textBox1.TabIndex = 1;
			this.textBox1.Text = "textBox1";
			//
			// textBox2
			//
			this.textBox2.Location   =   new   System.Drawing.Point(144,
184);
			this.textBox2.Name = "textBox2";
			this.textBox2.Size = new System.Drawing.Size(336, 22);
			this.textBox2.TabIndex = 2;
			this.textBox2.Text = "textBox2";
			this.textBox2.TextChanged += new
System.EventHandler(this.textBox2_TextChanged);
			//
			// textBox3
			//
			this.textBox3.Location = new System.Drawing.Point(144,
16);
			this.textBox3.Name = "textBox3";
			this.textBox3.Size = new System.Drawing.Size(336, 22);
			this.textBox3.TabIndex = 3;
			this.textBox3.Text = "textBox3";
			this.textBox3.TextChanged += new
System.EventHandler(this.textBox3_TextChanged);
			//
			// textBox4
			//
			this.textBox4.Location = new System.Drawing.Point(144,
56);
			this.textBox4.Name = "textBox4";
			this.textBox4.Size = new System.Drawing.Size(336, 22);
			this.textBox4.TabIndex = 4;
			this.textBox4.Text = "textBox4";
			//
			// textBox5
			//
			this.textBox5.Location = new System.Drawing.Point(144,
96);
			this.textBox5.Name = "textBox5";
			this.textBox5.Size = new System.Drawing.Size(336, 22);
			this.textBox5.TabIndex = 5;
			this.textBox5.Text = "textBox5";
			//
			// timer3
			//
			this.timer3.Interval = 1000;
			this.timer3.Tick += new System.EventHandler(this.timer3_Tick);
			//
			// button2
			//
			this.button2.Location = new System.Drawing.Point(256, 232);
			this.button2.Name = "button2";
			this.button2.Size = new System.Drawing.Size(168, 23);
			this.button2.TabIndex = 6;
			this.button2.Text = "EXIT";
			this.button2.Click += new System.EventHandler(this.button2_Click);
			//
			// label1
			//
			this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif", 9.75F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
			this.label1.Location = new System.Drawing.Point(0, 16);
			this.label1.Name = "label1";
			this.label1.Size = new System.Drawing.Size(128, 23);
			this.label1.TabIndex = 7;
			this.label1.Text = "SIGNALING ----->";
			//
			// label2
			//
```

```
                        this.label2.Location = new System.Drawing.Point(0, 96);
                        this.label2.Name = "label2";
                        this.label2.Size = new System.Drawing.Size(128, 23);
                        this.label2.TabIndex = 8;
                        this.label2.Text = "Msgs on Link----->";
                        //
                        // label3
                        //
                        this.label3.Location = new System.Drawing.Point(0, 136);
                        this.label3.Name = "label3";
                        this.label3.Size = new System.Drawing.Size(128, 23);
                        this.label3.TabIndex = 9;
                        this.label3.Text = "Msgs on SSP---->";
                        //
                        // label4
                        //
                        this.label4.Font = new System.Drawing.Font("Microsoft Sans
Serif",                9.75F,                System.Drawing.FontStyle.Bold,
System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                        this.label4.Location = new System.Drawing.Point(0, 56);
                        this.label4.Name = "label4";
                        this.label4.Size = new System.Drawing.Size(128, 23);
                        this.label4.TabIndex = 10;
                        this.label4.Text = "PACKET LOSS-->";
                        //
                        // label5
                        //
                        this.label5.Location = new System.Drawing.Point(0, 184);
                        this.label5.Name = "label5";
                        this.label5.Size = new System.Drawing.Size(128, 23);
                        this.label5.TabIndex = 11;
```

```
			this.label5.Text = "Msgs on SCP---->";
			//
			// Form1
			//
			this.AutoScaleBaseSize = new System.Drawing.Size(7, 15);
			this.ClientSize = new System.Drawing.Size(488, 273);
			this.Controls.Add(this.label5);
			this.Controls.Add(this.label4);
			this.Controls.Add(this.label3);
			this.Controls.Add(this.label2);
			this.Controls.Add(this.label1);
			this.Controls.Add(this.button2);
			this.Controls.Add(this.textBox5);
			this.Controls.Add(this.textBox4);
			this.Controls.Add(this.textBox3);
			this.Controls.Add(this.textBox2);
			this.Controls.Add(this.textBox1);
			this.Controls.Add(this.button1);
			this.Font = new System.Drawing.Font("Microsoft Sans Serif",
9.75F,     System.Drawing.FontStyle.Bold,     System.Drawing.GraphicsUnit.Point,
((System.Byte)(0)));
			this.Name = "Form1";
			this.Text = "Form1";
			this.ResumeLayout(false);

		}
		#endregion

		/// <summary>
		/// The main entry point for the application.
		/// </summary>
		[STAThread]
		static void Main()
		{
			Application.Run(new Form1());
		}

		private void button1_Click(object sender, System.EventArgs e)
		{
			ssp=new SSP();
			scp=new SCP();
			link=new Link(scp,ssp);
			ssp.set_link(link);
			scp.set_link(link);
			timer1.Start();
			timer2.Start();
			timer3.Start();
			textBox1.Text="Button click ok";
////////////////////////////////////////////////////////////
////////////////////////////////////////////////
		}

		private void timer1_Tick(object sender, System.EventArgs e)
		{	//SSP ssp=new SSP();
			a++;
			b++;
			//ssp.time();
			scp.time();
			textBox3.Text=link.smooth_running();
			textBox2.Text=scp.smooth_running();
			textBox4.Text=link.smooth_running2();
		}

		private void timer2_Tick(object sender, System.EventArgs e)
		{
			ssp.make_call();
			//timer2.Enabled=false;

		}

		private void timer3_Tick(object sender, System.EventArgs e)
		{
			ssp.time();
			textBox5.Text=link.smooth_running3();
			textBox1.Text=ssp.smooth_running2();
		}

		private void button2_Click(object sender, System.EventArgs e)
		{
			this.Dispose();
			Application.Exit();
		}
			}
}
```

Link(VC#)

```
using System;
using System.IO;
using System.Data;
using System.Text;
```

```
namespace project
{
	/// <summary>
	/// Summary description for Link.
	/// </summary>
	public class Link
	{
		public int total_signaling;
		public int temp1;
		public int temp2;
		public int net_signaling;
		public int total_messages;
		public int total_packet_loss;
		public static int delay=50;
		public SCP scp;
		public SSP ssp;
		public string test;
		public string test2;
		public string test3;
		public int time;
		public int[] array_signaling;
		public int [] array_messages;
		int array_index;


		public Link(SCP scp, SSP ssp)
		{this.time=0;
		this.ssp=ssp;
			this.scp=scp;
			total_messages=0;
			temp1=0;
```

```
            array_signaling = new int[20];
            array_messages=new int[20];

 array_index=0;
}
public string smooth_running()
{
            return test;
}
public string smooth_running2()
{
            return test2;
}
public string smooth_running3()
{
            return test3;
}
public void receive(Message m)
{           total_messages++;
            test3="msg received on link with parameter "+m.parameter;
            m.TTL-=delay;
            total_signaling+=m.size;

            if(m.dest_mod=="SCP")
                        scp.receive(m);
            else if (m.dest_mod=="SSP")
                        ssp.receive(m);
            test="Total signaling on link "+total_signaling.ToString();
            total_packet_loss=scp.packet_loss+ssp.packet_loss;

            test2="total packet loss is"+total_packet_loss.ToString();
```

```
		}

			///////////////////////////////
			///
			if(array_index<15)

			{
				net_signaling=total_signaling-temp1;
			temp1=total_signaling;
			array_signaling[array_index]=net_signaling;
			array_messages[array_index]=total_messages;
			array_index++;
			}
			////////////////////////////////
			else
			{
			write();

			array_index=0;

			}
		}

		public void write()
		{	string						path="D:\\temp\\Final
Project\\PWork\\project_data.txt";
			StreamWriter sw=new StreamWriter(path,false);
			//StreamWriter sw=File.AppendText(path) ;
			for(int i=0;i<15;i++)
			{	time++;
				sw.WriteLine(time   +   "   "   +array_signaling[i]+"
"+array_messages[i]+" "+total_packet_loss);
			}
```

```
                        sw.Close();



                }



        }
}
```

Message(VC#)

```
using System;

namespace project
{
        /// <summary>
        /// Summary description for Message.
        /// </summary>
        public class Message
        {       public string parameter;
                //public int id;
                public int size;
                public int TTL;
                public string type;
                public int delay_in_module;
                public string dest_sib;
                public string dest_mod;
                public    Message(string    parameter,int    size,int    TTL,string
dest_sib,string dest_mod,string type)
                {
                        this.parameter=parameter;
                        //this.id=id;
                        this.size=size;
                        this.TTL=TTL;
                        this.dest_sib=dest_sib;
                        this.dest_mod=dest_mod;
                        this.type=type;

                }
        }
}
```

Queue(VC#)

```
using System;

namespace project
{


        public class Queue
        {       private static int MAX=10000;
                private int count;
                public Node front;
                public Node end;
                private Node temp;
                private string creator;
                public string test;
                public int append_call=0;
                //private SSP ssp;
                //private SCP scp;
```

```
///
///
public Queue(string whoz)
{
	count = 0;
	front = null;
	end = null;
	temp = null;
	creator=whoz;

}
/// Test to see if the Queue might be empty.
///
public string smooth_running()
{
	return test;
}
public bool empty
{
	get
	{
		return(count==0);
	}
}
///
/// Number of Items in the Queue.
///
public int Count
{
	get
```

```
        {
                return count;
        }
}
///
/// This function will append to the end of the Queue or
/// create The first Node instance.
///
///
public void append(Message m)
{
        append_call++;


        if(count==0)
        {
                front = end = new Node(m, front);
        }
        else
        {
                end.Next = new Node(m, end.Next);
                end = end.Next;
        }
        this.count++;

}
///
/// This function will serve from the front of the Queue.  Notice
/// no deallocation for the Node Class, This is now handled by the
/// Garbage Collector.
///
```

```
        public void discard()
        {
            temp = front;
            if(count == 0)
                throw new Exception("tried to serve from an empty
Queue");
            front = front.Next;
            count--;
            //return temp.message;
        }
        ///
        /// This function will print everything that is currently in the
        /// Queue.
        ///

        public bool isFull()
        {
            if (count==MAX)
            {

                return true;

            }
            else
            {

                return false;

            }
        }
        }

                public void printQueue()
                {
                        temp = front;
                        while(temp != null)
                        {
                                Console.WriteLine("{0}", temp.message.ToString());
                                temp = temp.Next;
                        }
                }

        }
        ///
        /// The Node class holds the object and a pointer to the next
        /// Node class.
        ///
        public class Node
        {
                public Node Next;
                public Message message;


                public Node(Message msg, Node nxt)
                {
                        Next = nxt;
                        message = msg;
                }
        }



}
```

SCP(VC#)

```
using System;

namespace project
{
	/// <summary>
	/// Summary description for SCP.
	/// </summary>
	public class SCP
	{
		//private SCP scp;
		public Queue q=new Queue("SCP");
		public static int delay=50;
		public Link link;
		public int packet_loss;
		public string test;
		public string smooth_running()
		{
			return test;
		}
		public SCP()
		{
			packet_loss=0;
		}
		public void set_link(Link link)
		{
		this.link=link;
		}
		public void time()
		{
			Node n=new Node(null,null);
			n=q.front;
			if(n==null)
				return;
			while(n!= null)
			{
				n.message.TTL-=delay;
				n.message.delay_in_module-=delay;
				n=n.Next;
			}
			//perform discarding activities
			//all msgz in queue r decremented in thier ttl by delay tym dat
			//is 100
			//all msgz are decremented in dere delay in module by 100
			//if ne1z delay in module  is equal to 0 n tym 2 live is greater
dan 0 forwrd n wueu shud b adjusted
			n=q.front;
			if(n==null)
				return;
			if(n.message.TTL>0)
			{forward(n.message);
			q.discard();}// added later on
			else
			{
				q.discard();
				packet_loss++;
				if(packet_loss>5)
				{
						for(int i=1;i<7;i++)
				 {
```

```
                        q.discard();
                    }

                }

            }
            //test="time of SCP called";
            //all msgz in queue r decremented in thier ttl by delay tym dat
            //is 100
            //all msgz are decremented in dere delay in module by 100
            //if ne1z delay in module  is equal to 0 n tym 2 live is greater
dan 0 forwrd n wueu shud b adjusted
        }
        public void receive(Message m)
        {
            m.delay_in_module=100;//and add m to queue

            if(!q.isFull())
            {
                q.append(m);
                //test="q  of  SCP  now  have  "+q.Count.ToString()+"
msgz";
            }
            else
            { int i;
                packet_loss+=7;
                for(i=1;i<6;i++)
                {
                    q.discard();
                }
            }
        }

        public void forward(Message m)
        {
            if (m.dest_sib=="sdm")
                sdm_sib(m);
            else if(m.dest_sib=="auth")
                AUTH_SIB(m);
            else if (m.dest_sib=="tmsi")
                TMSI_SIB(m);
            else if (m.dest_sib=="cipher")
                CIPHER_SIB(m);
            else if (m.dest_sib=="finalize")
                FINALIZE_SIB(m);

            //test="fwd of scp called";


        }

        public void sdm_sib(Message m)
        {
            test="msg in scps sdm with parameter "+m.parameter;
            if (m.parameter=="LUREQUEST")
            {
             Message                                              msg=new
Message("REQAUTHPARAM",54,1000,"sdm","SSP",m.type);
                link.receive(msg);
            }
            else if (m.parameter=="AUTHPARAM")
                    AUTH_SIB(m);
            else if (m.parameter=="AUTHACK")
                    {
                    Message                              msg=new
Message("UPDATESDP",70,1000,"sdm","SSP",m.type);
                link.receive(msg);}

            else if (m.parameter=="UPDATEACK")
                CIPHER_SIB(m);



        }
        public void AUTH_SIB(Message m)
        {       test="msg in scps auth with parameter "+m.parameter;
            //test="SCPz Auth sib called //success";
            if(m.parameter=="AUTHACK")
                sdm_sib(m);

             Message                                      msg=new
Message("AUTHENTICATE",72,1000,"auth","SSP",m.type);
            link.receive(msg);

        }

        public void CIPHER_SIB(Message m)
        {       test="msg in scps cipher with parameter "+m.parameter;

            Message                                       msg=new
Message("STRTCIPHER",64,1000,"cipher","SSP",m.type);
            link.receive(msg);
            TMSI_SIB(m);
```

```
		}
		public void TMSI_SIB(Message m)
		{	test="msg in scps tmsi with parameter "+m.parameter;
			if(m.parameter=="TMSIACK")
				FINALIZE_SIB(m);
			else
			{
					Message					msg=new
Message("NEWTMSI",57,1000,"tmsi","SSP",m.type);
					link.receive(msg);
			}
		}
		public void FINALIZE_SIB(Message m)
		{	test="msg in scps finalize with parameter "+m.parameter;
		Message								msg=new
Message("LUACCPT",49,1000,"finalize","SSP",m.type);
			link.receive(msg);
		}



	}
}
```

SSP(VC#)

```
using System;
using System.Windows.Forms;

namespace project
{
```

```
/// <summary>
/// Summary description for SSP.
/// </summary>
///
public class SSP
{
        public string test;
        public string test2;
        //private SSP ssp;
        public Queue q=new Queue("SSP");
        public static int delay=50;
        public Link link;
        public int packet_loss;
        public int receive_call_count=0;
        //public Form form;
        public SSP()
        {
                //form=new Form1();

                packet_loss=0;




                //
                // TODO: Add constructor logic here
                //
        }
        public void set_link(Link link)
        {
                this.link=link;
        }
```

```
public string smooth_running()
{
return test;
}
public string smooth_running2()
{
        return test2;
}

public void time()
{

        Node n=new Node(null,null);



        n=q.front;
        if(n==null)
                return;
        while(n!= null)
        {
                        n.message.TTL-=delay;
                n.message.delay_in_module-=delay;
                n=n.Next;
        }
        //perform discarding activities
        //all msgz in queue r decremented in thier ttl by delay tym dat
        //is 100
        //all msgz are decremented in dere delay in module by 100
        //if ne1z delay in module  is equal to 0 n tym 2 live is greater
dan 0 forwrd n wueu shud b adjusted
        n=q.front;
```

```
				if(n==null)
					return;



				//if(n.message.delay_in_module<0 && n.message.TTL>0)
				if(n.message.TTL>0)
				{
						forward(n.message);
						q.discard();
				}
				else
				{
					q.discard();
					packet_loss++;
					if(packet_loss>5)
					{
						for(int i=1;i<7;i++)
						{
							q.discard();
						}

					}
				}



			}
			/////////////////////////////////
			public void initiate_call()
			{
				Message                                                      m=new
Message("LUREQUEST",64,100000,"sdm","SCP","intra");
```

```
        link.receive(m);
        //test="initiate_call of SSP called";

}
///////////////////////////////////////////
public void make_call()
{
        initiate_call();

}

/////////////////////////////////////

public void forward(Message m)
{
if (m.dest_sib=="sdm")
        sdm_sib(m);
        else if(m.dest_sib=="auth")
        AUTH_SIB(m);
        else if (m.dest_sib=="tmsi")
        TMSI_SIB(m);
else if (m.dest_sib=="cipher")
        CIPHER_SIB(m);
else if (m.dest_sib=="finalize")
        FINALIZE_SIB(m);
        //test="fwd of ssp success";

}

public void receive(Message m)
{   int loss_time=0;
```

```
                        m.delay_in_module=100;//and add m to queue
                        if(!q.isFull())
                        {       //test2="ssp's packetloss "+packet_loss;
                                q.append(m);
                                //test="q  of  SSP  now  have  "+q.Count.ToString()+"
msgz";
                                //test=m.parameter;
                        }
                        else
                        { //  test2="msg loss in ssp "+packet_loss;
                        //test="cannot add to ssp q";
                        //      if (packet_loss>5)
                        //      {   packet_loss=1;
                        //              for(int i=0;i<7;i++)
                        //                      q.discard();
                        //      }
                        //      else
                        //      {
                                        loss_time++;
                                        packet_loss++;
                        //      }
                        }
                                receive_call_count++;
                        //test2="Receive            of            ssp            called
"+receive_call_count.ToString();
                        //test2=receive_call_count.ToString();
                }
                ////////////////////////////////////////////////////////////
                ///

                public void sdm_sib(Message m)
                {   test2="msg in ssps sdm with parameter "+m.parameter;
                        if (m.parameter=="REQAUTHPARAM")
                        {
                                Message                                         msg=new
Message("AUTHPARAM",103,100000,"sdm","SCP",m.type);
                                link.receive(msg);
                        }


                        else if (m.parameter=="UPDATESDP")
                        {
                                Message                                         msg=new
Message("UPDATEACK",57,100000,"sdm","SCP",m.type);
                                link.receive(msg);
                        }




                        //test="SSP sdm call success";
                }
                public void AUTH_SIB(Message m)
                {       test2="msg in ssps auth with parameter "+m.parameter;
                        if(m.parameter=="AUTHENTICATE")
                        {
                                Message                                         msg=new
Message("AUTHACK",59,100000,"auth","SCP",m.type);
                                link.receive(msg);
                        }
                }
                public void CIPHER_SIB(Message m)
                {

                        //starting to cipher
                }
                public void TMSI_SIB(Message m)
                {test2="msg in ssps tmsi with parameter "+m.parameter;
                        if(m.parameter=="NEWTMSI")

                        {
                                Message                                          msg=new
Message("TMSIACK",49,100000,"tmsi","SCP",m.type);
                                link.receive(msg);
                        }
                }
                public void FINALIZE_SIB(Message m)
                {test2="msg in ssps cipher with parameter "+m.parameter;
                        //    lu finalized
                }
        }
}
```

REFERENCES

[1] S. M. Redl, M. K. Weber, M. W. Oliphant, An introduction to GSM, Artech House, London, 1995.

[2] ETSI GSM recommendations 03.02 - Network architecture.

[3] ETSI GSM recommendations 03.04 - Signalling requirements relating to routing of calls to mobile subscribers.

[4] ETSI GSM recommendations 03.05 - Signalling requirements relating to routing of calls to mobile subscribers.

[5] J. Thörner, Intelligent Networks, Artech House, 1994.

[6] M. Read, The evolution of GSM towards IN, Implementing and Exploiting Intelligent Networks, 20th of March 1996.

[7] J. M. Duran, , J. Visser, International Standards for Intelligent Networks, IEEE Communications Magazine, Feb 1992.

[8] ITU recommendations Q1201 - Principles of Intelligent Network architecture.

[9] L. Isotalo, Applying Intelligent Networks to GSM, ISBN - 0-7803-1820-X/94, IEEE 1994.

[10] N. Lilly, Integrating and evolving the INAP and MAP protocols, The GSM MoU/ETSI workshop on ‘The evolution of GSM towards IN’, Brussels, Belgium, 1995.

[11] B. Jabbari, Intelligent Network Concepts in Mobile Communications, IEEE Communications Magazine, February 1992.