Dial over Data solution

MAX WELTZ

KTH Information and
Communication Technology

Master of Science Thesis
Stockholm, Sweden 2008

COS/CCS 2008-02

Dial over Data solution

Max Weltz

February 21, 2008

KTH / ICT / CoS
Academic Supervisor and Examiner: Gerald Q. Maguire Jr.
Industrial supervisor: Jörgen Steijer, Opticall AB

Abstract

The increased use of computer networks has lead to the adoption of Internet-based solutions for reducing telephony costs. This has proved to be a boon to callers who can reach the other party directly via the Internet. Unfortunately numerous business persons still need to call to and from mobile phones which are currently a domain where the customers are generally tightly bound to their operators.

To provide a simple solution to this problem for companies, Opticall AB has designed an integrated system called the Dial over Data solution, coupling a mobile interface with a low-rate communication channel, which allows calls to be originated remotely at the best price, exploiting the customer company's existing network. This scheme allows the customer company to easily control telecommunications costs, to monitor their employees' efficiency, and more generally speaking to claim a central role in the communications of their employees.

The proposed solution allows distant callers (usually employees of the customer company) to benefit from the company's internal network, which is usually more cost effective and offering connectivity to more networks than a cell phone. The Dial over Data solution enables communication between any phone accessible from the customer company's telephony network (such as SIP clients, landline phones, and mobile phones) at a lower cost.

This thesis project analyzes existing technologies and compares them to the pre-existing prototype to ascertain the validity of the method and of the components used. This project also explains the improvements brought to the features offered by the DoD solution: the initial prototype has been developed into a stable and functional product, and has been tested internally. Prompted by a need for scalability and additional features, the replacement of Asterisk for the handling of SIP calls by other SIP servers has also been considered and tested.

Sammanfattning

Den nuvarande ökningen av datanätverk har lett till adoptionen av Internetbaserade lösningar för att förminskar kostnader inom telefoni. Tyvärr behöver åtskilliga affärsmän fortfarande ringa till och ifrån mobiler som återstår som ett område där kunderna är fastkedjade till deras operatörer. För att tillföra en enkel lösning till detta problem för kundföretag har Opticall AB planlagt ett integrerat system som kallas Dial over Data som kopplar ihop ett mobilt gränssnitt, med en billig kommunikationsmedel, som tillåter telefonsamtal påbörjas avlägset på det billigaste priset tack vare företagets nätverk. Det ger möjligheten till kundföretaget att vara centralt för sina personals kommunikationer. Det medger ett enkelt sätt att kontrollera kostnader samt övervaka personalens effektivitet.

Den Dial over Data lösningen är en lösning som tillåter avlägsen besökarna med kundföretagets inre nätverk kommer att dra nytta av eftersom det är mer kostnadseffektiv och flexibel än en blott mobiltelefon. Denna möjliggör kommunikation mellan SIP-klienter, fast telefoni och mobiltelefoni för en lägre kostnad till företaget utan att framkalla besvär för sina anställda. Konnektiviteten till företagets inre nätverk samt en låg besvärlighetsnivå är garanterade respektive genom konfigurationsförmågan av produkten och ett praktiskt gränssnitt som är redo för korporativkontaktlistor och visar alla informationen som är relevanta till förbrukarnas erfarenhet.

Det här avhandlingsprojektet analyserar existerande teknologier och sätter dem i relation till den sedan tidigare framtagna prototypen för att utröna validiteten hos metoden och beståndsdelar. Projektet förklarar även de förbättringar som gjorts till de egenskaper som erbjuds av DOD-lösningen: prototypen har utvecklats till en stabil och funktionell produkt och har testats internt. Driven av behovet för skalabilitet och ytterligare egenskaper har ersättandet av Asterisk för hantering av SIP samtal av andra SIP servers övervägts och testats.

Résumé

L'omniprésence des réseaux informatiques aujourd'hui a pour effet de pousser de plus en plus à l'adoption de solutions en ligne pour réduire les coûts liés à la téléphonie. Malheureusement de nombreux hommes d'affaires doivent toujours appeler vers et depuis des téléphones portables. Malheureusement, la téléphone mobile demeure un domaine où les clients sont étroitement dépendants des opérateurs. Afin de fournir une solution à ce problème, Opticall AB a conçu un système intégré appelé le Dial over Data Server, utilisant une interface mobile qui permet à des appels d'être lancés à distance au meilleur prix en utilisant les capacités de communication de l'entreprise cliente. Un tel dispositif place l'entreprise cliente au centre des communications de ses employés en permanence. Ceci permet un meilleur contrôle des coûts et une surveillance centralisée de l'activité des employés.

Le procédé Dial over Data est une solution permettant aux appelants (généralement employés de l'entreprise cliente) situés à distance de bénéficier du réseau interne de l'entreprise cliente, généralement moins cher et plus flexible qu'un téléphone mobile traditionnel. Ceci permet d'établir des connexions à des clients SIP et à des téléphones fixes ou mobiles à un moindre coût pour la compagnie, avec peu ou pas d'inconvénients pour les employés. La connectitvité au réseau interne de l'entreprise et l'absence d'inconvénients sont garantis respectivement par la configurabilité du produit et de son interface pour conserver les habitudes et les données utilisateurs telles que les carnets d'adresse.

Dans ce rapport de stage nous avons analysé les différentes technologies existantes et les avons mises en perspective par rapport au prototype déjà existant afin d'en valider les composants et le principe. Le prototype a été développé jusqu'à atteindre l'état de produit stable et fonctionnel, testé en interne. Les fonctionnalités du produit ont également été accrues pour offrir plus qu'une simple réduction des coûts. Le remplacement d'Asterisk pour la prise en charge des appels SIP a notamment été envisagé et testé, notamment dans l'espoir de fournir plus de fonctionnalités et de meilleures performances..

Acknowledgments

I would like to thank here Mattias Hansson, for all the ideas and leads he gave me during my stay at Opticall AB; Jörgen Steijer, for his camaraderie and his technical guidance in the use and configuration of the telephony equipments involved in this project; Gerald Q. Maguire Jr., for his advice, suggestions, and numerous comments concerning this thesis project; Marceau Coupechoux, for his encouragement; Terry Pratchett, for his amusing books, whenever I was somewhere between work and home; Renae, for her support when I eventually reached home; and finally, my parents, who provided me with everything I missed from France.

Contents

List of Figures

List of Tables

Glossary

AGI Application Gateway Interface

AJAX Asynchronous JavaScript and XML

AMD Answering Machine Detection

AoR Address of Record

B2BUA Back-to-Back User Agent

BS Base Station

BSN Block Sequence Number

CDR Call Detail Record

DECT Digital Enhanced Cordless Telecommunications

DoD Dial over Data

DTMF Dual Tone Multi Frequency

EIS Enterprise Information System

FMC Fixed-Mobile Convergence

FXO Foreign eXchange Office

FXS Foreign eXchange Subscriber

GAN Generic Access Network

GPRS General Packet Radio Service

GSM Global System for Mobile communications

HLR Home Location Register

HTML HyperText Markup Language

HTTP HyperText Transfer Protocol

HTTPS HTTP over SSL

IAX InterAsterisk eXchange

IRQ Interrupt ReQuest

ISP Internet Service Provider

J2EE Java 2 Enterprise Edition

JAAS Java Authentication and Authorization Service

JNDI	Java Naming and Directory Interface
JSP	Java Server Pages
JSR	Java Specification Request
JVM	Java Virtual Machine
LCR	Least Cost Routing
LDAP	Lightweight Directory Access Protocol
MIDP	Mobile Information Device Profile
MS	Mobile Station
MSC/VLR	Mobile Switching Center/Visitors' Location Register
NT	Network Terminator
OSP	Open Settlement Protocol
PABX	Private Automatic Branch eXchange, synonym with PBX
PBX	Private Branch eXchange
PCM	Pulse Code Modulation
PLMN	Public Land Mobile Network
PRI	Primary Rate Interface
PSTN	Public Switched Telephony Network
RLS	Resource List Server
RTP	Real-time Transport Protocol
SDP	Session Description Protocol
SER	SIP Express Router
SIM	Subscriber Identity Module
SIP	Session Initiation Protocol
SMS	Short Message Service
SMSC	Short Message Service Center
SSL	Secure Socket Layer
STUN	Simple Traversal of UDP Through NAT
TE	Terminal Equipment
TFI	Temporary Flow Identity
TLLI	Temporary Logical Link Identifier
TURN	Traversal Using Relay NAT
UA	User Agent
UDP	User Datagram Protocol
UMA	Unlicensed Mobile Access

URI Universal Resource Identifier

USSD Unstructured Supplementary Service Data

VoIP Voice over IP

VPN Virtual Private Network

WLAN Wireless Local Area Network

XCAP XML Configuration Access Protocol

XML eXtensible Markup Language

Chapter 1

Introduction

Recent years have seen the booming growth of companies such as Skype [80]. These companies have specialized in cheap telephone services for the public, using the increasing availability of broadband computer networks in both private and public areas. Today 1.1 billion people have access to Internet, and it is expected that this figure will grow to 1.6 billion in 2011, accord to JupiterResearch [42]. Of these, over 80 million users [76] are using Skype today. Other competitors in this segment are the Internet Service Providers (ISP) that offer triple-play packages to their subscribers. In Europe, the ISP share of the Voice over IP (VoIP) calls is over 50 %. In the US, 20 % of the companies were using Voice over IP solutions in 2007 [41]. These solutions offer a lot of flexibility to customers for the routing of their online and landline calls, but they rarely offer attractive solutions for mobile calls. Only a few mobile phones are Skype-enabled and the VoIP solutions sold by ISPs are available only within the customer's home network operator.

In the mobile phone realm, customers are often tightly bound to the services offered by their operators: these often specify minimum subscription periods, locked phones, and various other business strategies to lock-in and control the customer. As traditional wide area wireless networks are very expensive to deploy these operators are reluctant to share revenues or customers with other players. Therefore users really have few alternatives when it comes to mobile communications. Fortunately, most mobile phones can now be connected to the Internet thanks to the General Packet Radio Service (GPRS) technology – and the third generation technologies where available, making it possible to develop innovative solutions for the benefit of the users, such as Fring [25].

With that in mind, Opticall AB[65] designed an integrated system, called "Dial over Data" (DoD), to provide customer companies with greater flexibility and control– and lower prices – for their mobile communications, while keeping in mind convergence between traditional telephony and VoIP telephony. Coupling a mobile interface with an Asterisk PBX and a GSM gateway (or any other equipment and/or connection providing low rates) located in the customer company's network, this solution exploits the particularities of the current offers from mobile phone operators in Sweden. For the reader who is not familiar with the Swedish mobile telecommunications market, it might be useful to note that most operators have subscriptions with free SMS messages and/or phone calls within the operator's own network; additionally, some operators offer bundled pricing of large numbers of flat rate minutes at a very low cost[1].

The scheme proposed by Opticall AB offers extra benefits to customer companies. Currently companies control the handling of all the calls placed from fixed line phones within their premises; thus they can control authorization, billing, monitor their employees, and so on. With the DoD solution, companies can play a **central role** in the communications of their employees, whether

[1] For example, SEK 0.1/minute, or approximately €0.01/minute, might be a typical price compared to 1 or more SEK per minute (i.e., more than €0.1/minute).

the employee is at the company's premises or in the field. Therefore, in addition to cost reduction, companies could fully control costs (for example, by not allowing calls to certain numbers via specific operators, from certain employees, at certain times, etc.) and they can monitor[2] in real-time the efficiency of their employees (for example, know without waiting the monthly bill for each employee, that an employee uses the company's subscription too much for private calls, or know how many customers the employee actually called during the last week).

In the following, we give an overview of the technologies relevant to the DoD solution, focusing on the server part. We then describe in the third chapter the initial state of the DoD project at the beginning of this thesis project, showing the initial achievements and describing the expected developments of the product. The fourth chapter introduces the various alternatives that could be pursued in this project; specifically the technological alternatives for the development of the DoD solution, possibly leading to a revision in some of the past choices in the light of this new examination. Other solutions providing similar or complementary benefits to the DoD solution are described as well, only for comparison purposes. The fifth chapter presents an hypothetical case study of a company migrating to the DoD solution. This chapter focuses on dimensioning considerations and cost savings. The sixth chapter presents an examination of the improvements realized during this thesis project. The seventh chapter is devoted to one major change brought to the DoD solution during this project: the addition of OpenSER to the DoD Server. The eighth chapter includes an overview of the tests that helped quantify the accomplishments realized via the DoD solution and the analysis of the tests' results. The final chapter presents our conclusions and suggest future work on the DoD project.

[2] The companies are referred to the data privacy laws enforced in their respective countries as well as the contracts binding them to their employees to verify under which conditions some information can be monitored.

Chapter 2

Background studies

In this part, we explain the technologies that are used in the DoD solution; especially showing why they are relevant. The DoD solution has as a short term goal allowing cheaper phone calls for entreprises when at least one end of the calls is mobile. In this thesis we focus on a business clientele, thus the DoD solution is likely to have to deal with SIP phones, whether they are soft phones or hard phones. The final product should be based on technologies familiar to companies on the server side and to the general public on the client side, this explains some of the choices made in the design and orientation of the DoD solution: the server side is based on Asterisk, the JBoss Application Server, and Java Beans, whereas the client side offers a web based interface and a Java mobile client, and should interface with Skype in the long term.

2.1 Preliminary definitions

In addition to the terms presented in the glossary provided on page x, we introduce here a few definitions of terms that we are using throughout this thesis report.

Company's network Often in this document, references are made to the company's network. This term covers both the company's telephony and computer networks, including all media (wireless local area networks (WLANs), landlines, cordless phones, mobile phones, etc.) and services (VoIP using SIP, Skype, etc.) that are offered and operated by or for the company.

Mobility In this document, the terms mobility or mobile refer, except where otherwise noted, to a service (in the broadest sense) that can be accessed at any time and anywhere using one type or another of available connection. This definition is close to that used by R. Kalakota and M. Robinson in [52] for "*mobile but online*".

Presence Presence is the ability of a user to notify a service, and *a fortiori* the other users of that service, of his or her current status (busy, available, etc.) or context (at home, at the office, with a customer, etc.), which modifies the behavior of the service concerning his or her calls (which can be forwarded, broadcasted, rejected, accepted quietly, etc.), as well as the behavior of the users benefiting from the presence information.

Customer *versus* user In the rest of this document, the term *customer* refers to the company that requests and pays for a service to be made available for its employees who constitute the *end-users* (that we simply call *users*) of this service.

Components of the DoD solution The entire DoD solution is implemented physically using a DoD system, that we refer to as comprising the following elements:

- the DoD Server, a machine (or machines) where the Asterisk PBX, SER, and the DoD JBoss server are running,
- databases storing the configurations of the DoD Server (and later, Asterisk's Call Details Records (CDR))
- the afore-mentioned DoD JBoss server providing features to the client interfaces and handling the communication with the Asterisk PBX,
- client interfaces providing users with the features of the DoD server (placing calls, accessing contact lists, etc.):
  - a Web interface (in its normal, and later, lightweight version) accessed over HTTP,
  - a mobile client developped in Java 2 Mobile Edition (J2ME) using Mobile Information Device Profile (MIDP) 2.0,
- a GSM gateway attached to the Asterisk PBX (or any other means to terminate calls).

2.2 GSM gateway

2.2.1 Principle

To connect to the mobile networks of various operators, the DoD solution can use one or more (GSM) gateways. Other solutions can be considered to connect to mobile networks but mobile gateways are likely to provide the easiest and cheapest way to do so. In our case we use a Blue Tower [86]. Note that here we have focused on connections to GSM networks, but there could be other types and brands of mobile gateways to connect to other types of mobile networks. Each GSM board in this gateway supports two modules of four SIM cards. A limitation is that only one SIM card in each module is active at one time (i.e., each GSM phone module can only be connected to a single GSM network at one time). In total this particular GSM gateway can support four such boards (for a total of 8 GSM phone modules). The firmware of the gateway enables the user to dispatch outgoing and incoming calls according to date, time of day, prefix of the callee or caller, etc. Through interfaces to the PSTN, the gateway can also forward calls to the PSTN or accept calls from the PSTN[1]. Figure 2.1 illustrates the use of a GSM gateway. This equipment allows a smart – and presumably cheaper – treatment of the calls to and from mobile phones inside an enterprise network rather than just sending them to or accepting them from the PSTN or the company's PBX. Note that the decision to based the DoD solution on a GSM gateway rather than other means was made in an earlier project and has been used as a generic working hypothesis for this project. However, we mention in chapter 7 other means that can be used in replacement of a GSM gateway. For instance, the means in use as of the end of this thesis was a SIP trunk.

2.2.2 Internals

For the curious reader, this section describes the hardware aspects of the Blue Tower GSM gateway. Additional details can be found in the guide [2] starting on page 12.

[1] More generally speaking, the gateway comports two ISDN PRI interfaces, one of which can be used as a fallback solution for calls that were not processed by any of the channels of the GSM gateway. The first ISDN PRI interface is used for connection with the Asterisk server

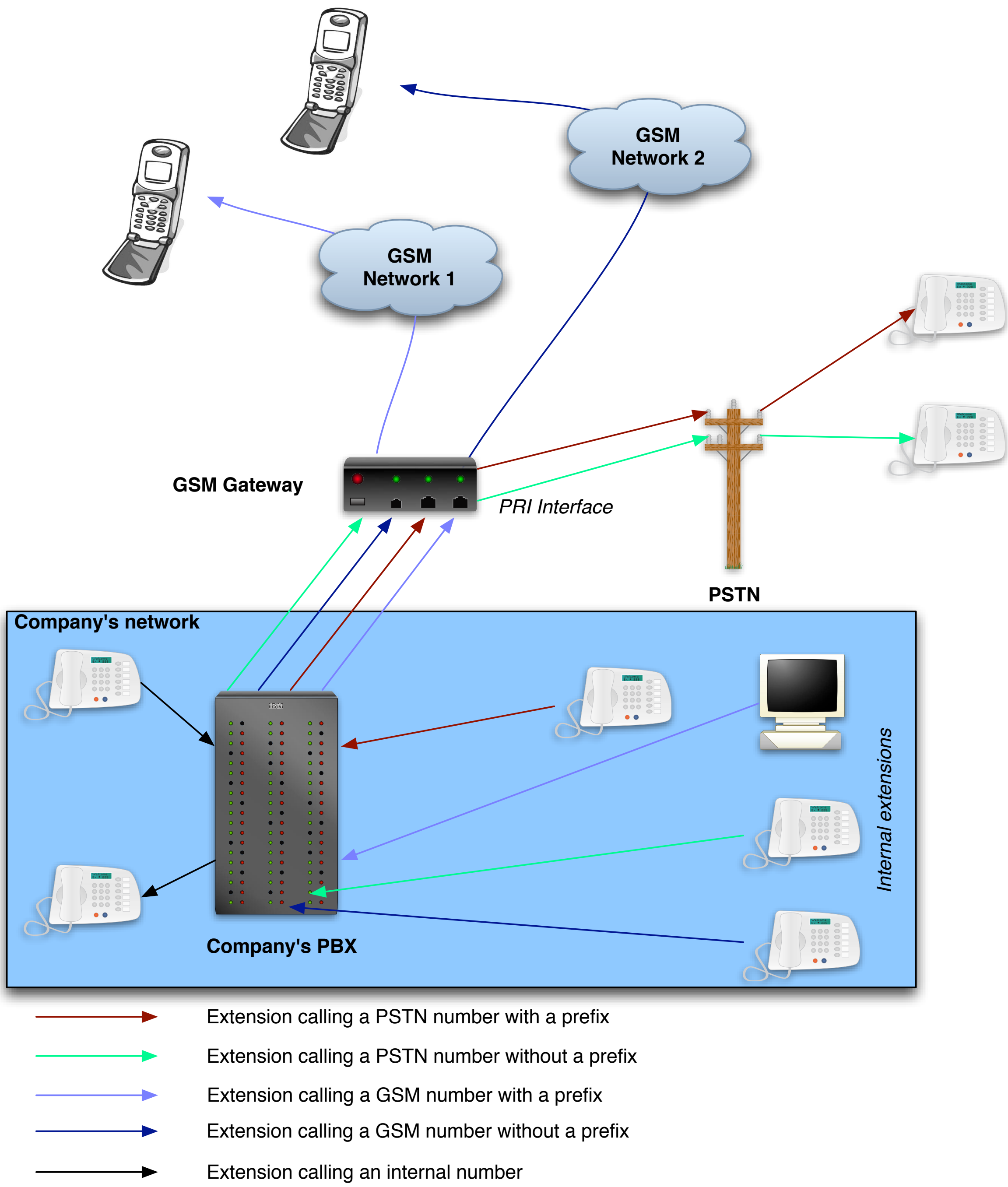


Figure 2.1: Principle and use of a GSM gateway, adapted from the figure on page 7 of [2]

Hardware

All the boards mentioned below were built on a 4-layered printed circuit board of 160x100mm. In the case of the boards mentioned as hot-pluggable, pins 1 and 32 are used for the hot-swapping power feed. The boards are organized around a Pulse Code Modulcation (PCM) bus, which separates control from data in the internal communications of the gateway; this makes it easier to process the voice streams entering or leaving the gateway. All cards are organized around the PCM bus. Logically, the gateway looks like a pool of cell phones.

Network-related boards

GSM gateways from 2N Telekomunicace accept four kinds of boards for network communications: GSM, 3G, VoIP, and ISDN PRI boards. The GSM boards come in seven flavors provided with two GSM modules (each capable of acting like a cellular phone - in terms of being able to originate or terminate a cellular call) from various constructors (such as Sony-Ericsson, Siemens, and Wavecom), different numbers of SIM card holders (from two to sixteen), and DTMF receivers. Those boards can be hot-plugged, which is handy for swapping SIM cards. The board accepts two external antennas , one for each module. Both GSM and 3G modules are connected using the PCM bus of the GSM gateway.

A VoIP board contains processors used for signaling or voice conversion and a main processor used for the board control. The board offers a 10/100BaseT Ethernet port for the reception of RTP packets (cf. section 2.3). An ISDN PRI board offers two ISDN PRI interfaces. These ISDN PRI interfaces can be setup as Master or Slave (the timing is assured by the PCM bus circuits) and as Terminal Equipment (TE) or Network Terminator (NT, cf. figure 2.2 for typical settings in an installation).

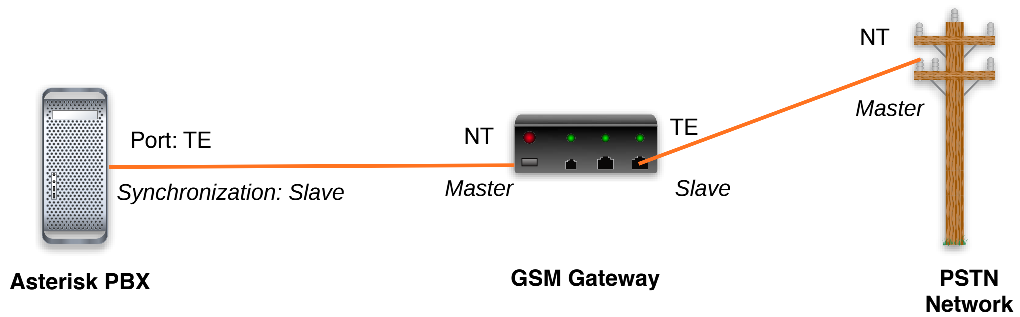


Figure 2.2: Connection of telephony equipments [2]

System boards

Beside the GSM, 3G, VoIP, and ISDN PRI boards that a gateway can include, there are two other types of board: the CPU board and the AUX board. The CPU board provides the GSM gateway with an Ethernet port and a serial port. The role of the CPU board is to run and control the GSM gateway. The AUX board provides an extra serial port and a connector for a microtelephone or headphone. The AUX board is used to record voice messages and place test calls.

2.2.3 Number portability

One argument that may reduce the interest in operating a mobile gateway is the recent possibility to keep one's mobile phone number when changing operator (so-called "number portability"). Because of this convenient possibility, it is now difficult to know to what operators a number really belongs. Thus it becomes more difficult to place calls using a SIM card belonging to the same network as that of the called party. However, under the Swedish Telecommunications Act, it is compulsory for operators to declare all ported numbers to the Swedish Number Portability Administrative Centre AB (SNPAC) [16]. In turn, the SNPAC provides access to this database to their customers. Several levels of services are available, from the simple Web forms to direct connection over a Virtual Private Network (VPN), from simple look-up of the number's operator to full routing informations. Note that number portability is not an issue if all the SIM cards in the gateway have flat-rate minutes which have the same cost to all operators. If this is not the case, then it may be necessary to connect to the SNPAC to determine which operator a given number is associated with and then use the appropriate SIM card in the GSM gateway.

2.3 Session Initiation Protocol (SIP)

A key design choice for the DoD solution was to primarily use widely used and tested open-source components. The Session Initiation Protocol (SIP) is one of them. The use of SIP in the DoD solution is at several levels. Currently SIP is used to communicate with SIP clients (soft phones for instance), cf. section 3.3. In the future it will also be used to provide presence-related services.

2.3.1 Description of SIP

SIP is described in the RFC 3261 [71]. According to this same RFC, the purpose of SIP is to be "*an application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants. These sessions include Internet telephone calls, multimedia distribution, and multimedia conferences.*" Establishing a SIP session between two devices does not itself include an agreement concerning the codecs used, this is the role of the Session Description Protocol (SDP). SIP enables SDP offers and responses to be exchanged by the potential parties thus enabling agreement concerning the media used by the different parties, via the exchange of these SDP messages [76].

Among the functionalities that have made SIP one of the most widely used VoIP protocol, along with H.323, is its support for presence and Addresses of Record (AoR). An AoR is a unique identifier which can represent several associated Universal Resource Identifiers (URIs) (for example, one or more cell phones, SIP soft clients, etc.). Both these functions will be of use in the DoD solution. Note that SIP is not used for the actual data transfer between the devices that have established a SIP session, instead other protocols have been designed for that purpose, such as the Real-time Transport Protocol (RTP).

2.3.2 SIP network

A SIP network consists of several elements, illustrated in figure 2.3 and described in detail in [76]:

- a SIP User Agent (UA) has two parts, the SIP UA client and the SIP UA server. The client originates the calls and the server receives them. A SIP UA is a device with SIP capabilities like a SIP phone or a computer with a SIP client.
- a SIP registrar associates a SIP address to an IP address. This is done using *REGISTER* messages (cf. section 2.3.3).
- a SIP proxy passes SIP messages between one UA and another, using the registries of the SIP Registrar. The SIP Proxy Server handles the request itself.
- a SIP redirect server, contrary to the SIP Proxy, replies to the originating UA with a message indicating how to pursue with its request.
- a SIP gateway allows SIP calls to reach other networks such as the PSTN or H.323 networks.

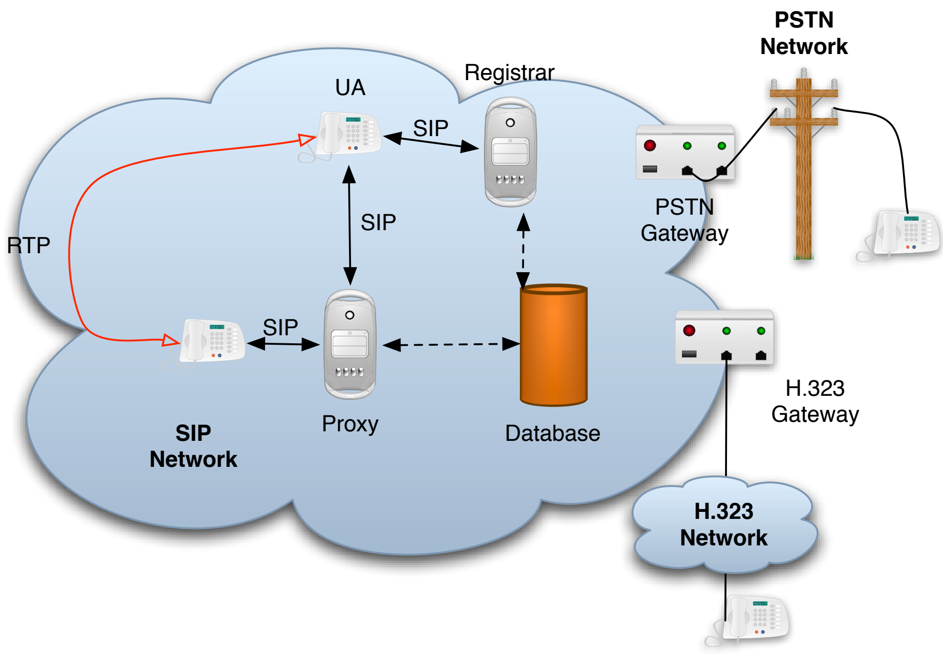


Figure 2.3: A SIP network [48]

2.3.3 SIP messages

In this subsection we illustrate the SIP messages used when establishing a communication session between two UAs, starting from address resolution to the actual establishment of the session. In our case, the first UA is the SIP client of the user of the DoD solution, the second UA is the callee, and the Proxy Server is the Asterisk server. Figure 2.4 illustrates the request address resolution in a SIP exchange using a proxy. Figure 2.5 represents the full course of a SIP session from its establishment to its termination. Note that a proxy could also be part of the exchanges depicted in figure 2.5, it would simpy pass the messages from one UA to the other. There are additional SIP messages besides those depicted here; these messages can be used to forward calls, modify the session, notify UAs of user presence, etc.

2.3.4 Role in the DoD solution

In the DoD solution, a SIP proxy is located in the Asterisk PBX (see section 2.4) and calls were initially limited to SIP clients internal to the company's own network domain. Note that some GSM gateways offer VoIP functionalities allowing several such gateways to be connected over the Internet. This is not strictly necessary as a suitable Asterisk configuration could also be used to route the necessary messages via SIP to another SIP-enabled server.

2.4 Asterisk

2.4.1 Software PBX

Asterisk is an open-source PBX program. It is the major component of the telephony part of the DoD Server. Asterisk is a free open-source PBX software solution. It enjoys a large community of users, partly due to its open-source and free nature and because of its demonstrated functionality. Asterisk can also be run on a wide variety of platforms: various flavors of Unix: Linux, Mac OS X, and various flavors of the BSD UNIX operating system. This made it an interesting choice for the implementation of the DoD solution.

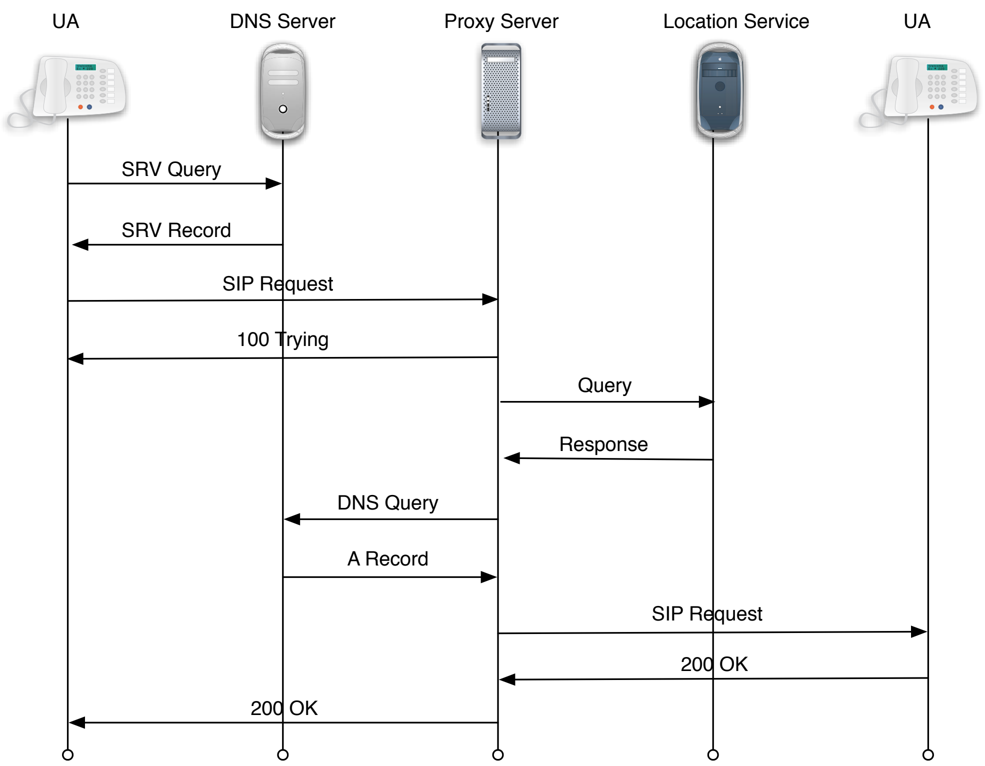


Figure 2.4: A SIP session request address resolution [76]

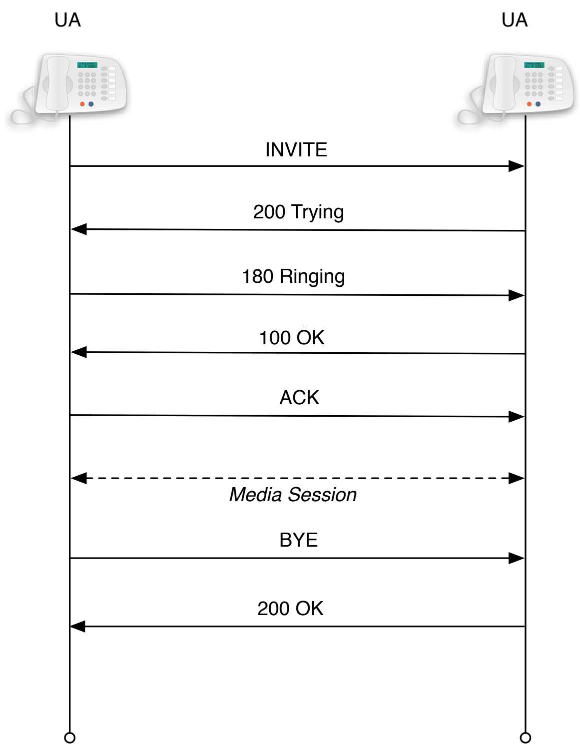


Figure 2.5: A SIP session establishment [76]

Another important reason for selecting Asterisk was that it benefits from the large open-source developers' community. Additionally, Asterisk now supports use of external databases such as PostrgreSQL and has both an Application Programming Interface (API) and an Application Gateway Interface (AGI) allowing developers to use Asterisk from their own applications, or the other way round [57]. In our case, Asterisk's API is extremely useful for placing calls from a Java-based server. In terms of connectivity, Asterisk supports both H.323 and SIP, connections to multiple gateways, and the use of ISDN PRI interfaces. It is also possible to forward calls to another Asterisk server using SIP or the InterAsterisk eXchange (IAX) protocol. This possibility allows larger companies to share resources distributed over several sites (cf. section 3.5.2).

2.4.2 About dialplans

Asterisk offers a very flexible configuration system enabling it to adapt to a wide variety of use cases that can be encountered. The configuration can be in terms of specific hardware configuration, SIP configuration, extensions configuration, and so on. The extensions are configured into what is called a **dialplan** in Asterisk's parlance. Dialplans are the core of Asterisk' logics that the system manager has to deal with every day as dialplans determine how each call made inside the company. Dialplans also determine where to place a call in a queue listening to a message loop, while waiting for the call to be served, for example when the caller calls the company's customer support number. An example of a dialplan is shown below:

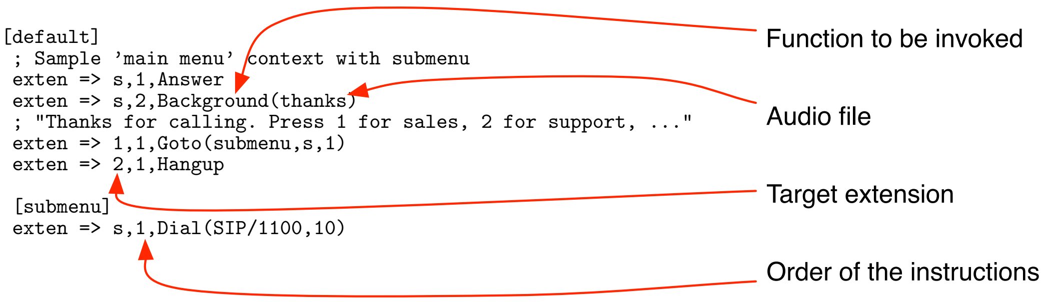


Figure 2.6: Example of dialplan as found in Asterisk's `extensions.conf` [19]

The above humorous example illustrates the basics of dialplans in Asterisk. The text within brackets are contexts, here "`default`" and "`submenu`". Contexts are used to separate different parts of the dialplan to ease understanding the dialplan's logical structure and avoid potential troubles by clearly isolating extensions. Under each context several entries introduced by the keyword "`exten`" appear. These entries refer to a dialed extension on a phone and that reached Asterisk. The first string after "`exten =>`" designates the extension called (such as "1", "2", etc.), a pattern to match it, or a special extension (such as "`s`" which is the default extension). Following the extension, there is a digit indicating in which order the different entries referring to an extension are to be processed. The final part of each entry is the most important: it defines the behavior of the Asterisk server with regard to the current call. We can see that Asterisk has various commands such as dialing a channel (for example, an extension on an ISDN PRI interface or a specific SIP client), going to another part of the dialplan (via the "`Goto`"), waiting for DTMF[2] input from the user, or playing sound files, etc.

[2] Dual Tone Multi Frequency (DTMF) refers to the pairs of tones emitted when a key is pressed on a phone. Decoding this DTMF signal to extract the information about which key has been pressed is generally used when accessing automated lines to navigate through menus.

2.5 Java Beans and the JBoss Application Server

2.5.1 Java Beans

Java Beans are a subcategory of Java classes following specific rules. According to their official definition, they are "*reusable software components that can be manipulated visually in a builder tool*" [105]. In the case of the DoD solution, they are specially suited for passing objects from one instance of a class to an instance of another class. They are often used for sessions and transactions purposes, such as database accesses [60].The modularity of the components developed for the DoD solution makes it possible to adapt the solution to the specific needs of a customer. Using enterprise-oriented Java solutions brings to the DoD solution features such as the JNDI component (see section 2.5.2) to access corporate LDAP servers, which are often used as a directory of employees or customers by larger companies.

Figure 2.7 represents a generic J2EE application (such as the DoD JBoss server) and its architecture. We can see the system is neatly divided into largely independent components, which allows reusability and facilitates adaptability. This is useful in large companies where several systems might need to access the same resources or offer the same interfaces, but in our case it will provide the possibility to develop different modules for different customers. For instance, the module (Business Tier) connecting the DoD JBoss server to the Asterisk server could be replaced to connect to another type of PBX. It is also possible to retain the Business Tier and modify only the Web tier to present users with the usual corporate interface (Client Tier). The last tier of the system is the Enterprise Information System (EIS) tier. The EIS represents the company-wide system supporting user activities [103]. In the case of the DoD JBoss server, that would concern their database of users - for authentication purposes, and potentially a contact list derived from the company's customer records to present users with an easy-to- use contact list (for example on the user's mobile phone).

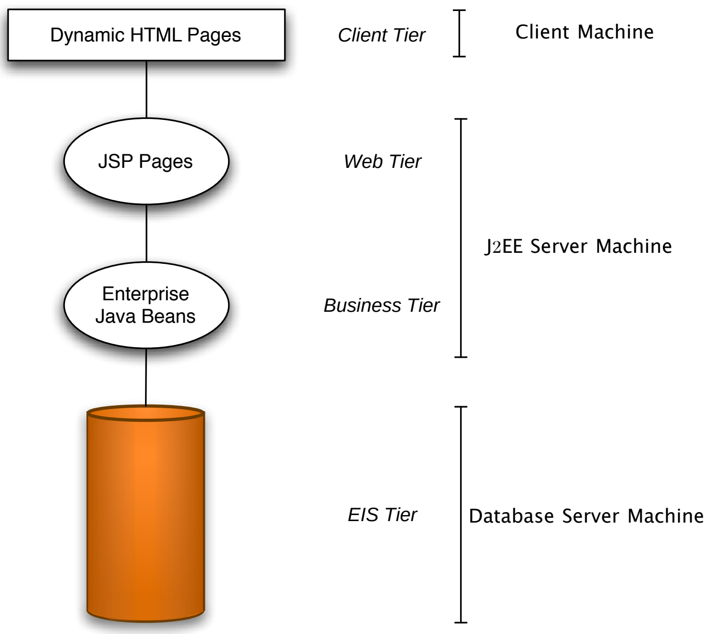


Figure 2.7: Schema of a multi-tiered J2EE application [11]

2.5.2 The JBoss Application Server

The JBoss Application Server [46] is an all-in- one enterprise server with a free open-source branch. This server utilizes an Apache Tomcat [89] web server and a J2EE application server with support for Java Naming and Directory Interface (JNDI). This setup allows for an easy deployment of Java Server Pages (JSP) in websites including Java Enterprise components such as Java Beans. This server integrates features, such as simplified management of login protected components of the web interface.

2.5.3 Role in the DoD solution

Java Enterprise technologies are used to bring to the user interface contents of various databases (configuration and user database, but also contact lists) and forward information sent by the user interface to the Asterisk server after suitable processing, including determination of the type of the caller's and callee's extension (SIP client, internal or external extension), application of the configuration's parameters (waiting time, conference number, etc.), *et caetera*.

Chapter 3

The DoD solution

In its initial state at the beginning of this thesis project, the DoD solution was functional in a prototype form: no advanced services were offered to the user or were offered only in a draft form; also some bugs were known. The system consisted of a JBoss server for the web interface, an Asterisk server to place the calls, and a GSM gateway to forward them to the callee if the extension was not a SIP client internal to the company. The initial state was primarily a proof of concept that demonstrated that the solution was viable and should be further developed to reach the commercial stage, as expected from the conclusions of Ning Zhou's report [109] which preceded this thesis project.

In this chapter we describe the functions a solution such as the DoD solution should provide, the initial state of the DoD solution at the beginning of this thesis project and its problems, and sketch the objectives of this particular thesis project.

3.1 Target customers and applications

The DoD solution was at first meant to be part of a company's processes to reduce their phone bills. Interestingly 64 % of the amount of the phone bills of companies are calls from landline phones to mobiles [31]. To reduce the costs associated with calls to mobile phones, **one** solution is for the DoD solution to integrate a GSM gateway into the company's network[1]. The value of this gateway builds upon a simple idea: instead of placing a call from the mobile phone of a salesperson out in the field to a customer directly, the salesperson requests a connection to be established between her handset and the customer using the most efficient means. The DoD Server determines the lowest cost means to establish the call. In the case of a mobile to mobile call, the GSM gateway is able to connect an outgoing call to the caller's network to an outgoing GSM call in the the same mobile network as the callee. This avoids the interconnection charges mobile operators charge for cross-operator calls. In the case of a mobile to PSTN or mobile to SIP phone call, the Asterisk server routes the incoming call to the PSTN or to the callee's SIP client.[2]

It appeared quickly that the DoD solution should not only offer cost reductions, but should also enable companies to take full advantage of their new place at the center of the communications of their employees. As mentioned earlier, this allows the company to filter certain calls depending on various criteria and to monitor in real-time the activity of its employees. This central role also enables the company to decide upon the best way to place calls, instead of leaving the decision to an external operator; leading to cost savings even for non-mobile calls. The company can

[1] This is the idea we worked with throughout this thesis. Companies and other workers are welcome to contribute and experiment with other solutions to suit other needs or provide different services and variable cost reductions.

[2] Indeed in the DoD Server, there are two consecutive components providing the logics for routing calls: the Asterisk dialplan followed by the GSM gateway Least Cost Routing (LCR) records.

also decide freely of who can use the DoD solution, including customers and temporary workers, such as consultants. On the whole, the DoD solution offers greater flexibility for companies to handle their phone communications.

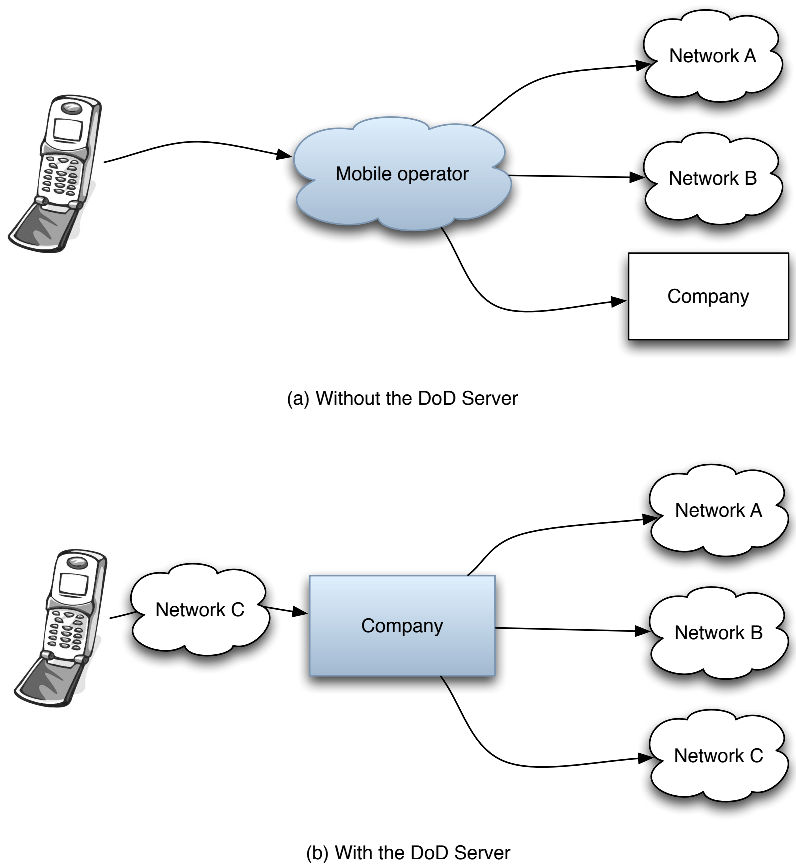


Figure 3.1: Central role of companies with the DoD solution

3.2 Assumptions

The DoD project makes few assumptions concerning the device used by the caller. This device can be a mobile phone with an installed client, a desktop phone sitting next to a computer that is used to access the Web interface and place the call, or a cell phone with HTML capability. There is indeed no restriction on the phone number used by the caller to place calls. This means the caller can even use a fixed phone at the premises of a customer or in a hotel. The goal is that the call should be routed as efficiently as if it were made from a phone in the caller's office inside the company's buildings. The overall assumption is that the Asterisk PBX's dialplan and the GSM Gateway's LCR records should suffice to guarantee the lowest rate[3].

3.3 Operation of the DoD solution

The DoD solution was first implemented by Ning Zhou at Opticall AB [109] as another thesis project at KTH. She established the basic hardware configuration to use, established communication between the DoD JBoss server and the Asterisk PBX, defined and tested the

[3] Note that the assumption is that the company has obtained advantageous subscriptions to access to the networks which they need to connect to.

primary functions, and made the first tests to compare latencies of calls made with and without the DoD solution from a cellular phone, to: a SIP phone, another cellular phone, or a landline phone.

Figure 3.2 shows the basic operation of the DoD solution. In this figure, the call is initiated through a mobile interface (steps 1, 2, and 3). The application can be invoked on mobile phones via a Web interface or through a Java application on a mobile phone, hence the name for the product: "Dial over Data". The DoD Server consults its database (steps 4 and 5), then initiates a call (step 6) from the Asterisk PBX to the originator of the call (in this case the caller's mobile phone, which explains why the outgoing call is made from the GSM gateway as shown in step 7). Once the originator of the call has answered this call, then the DoD Server is notified and places a call (steps 8 and 9) to the callee (here again, the call is made to a mobile phone using the GSM gateway, potentially using another operator's network). Finally the server bridges both calls (step 10). The query to the database insures that only authorized users can utilize the DoD Server (steps 4 and 5). This authorization step can also be used to restrict access depending on the callers, calls, or callees, or to collect information for internal billing or statistical purposes.

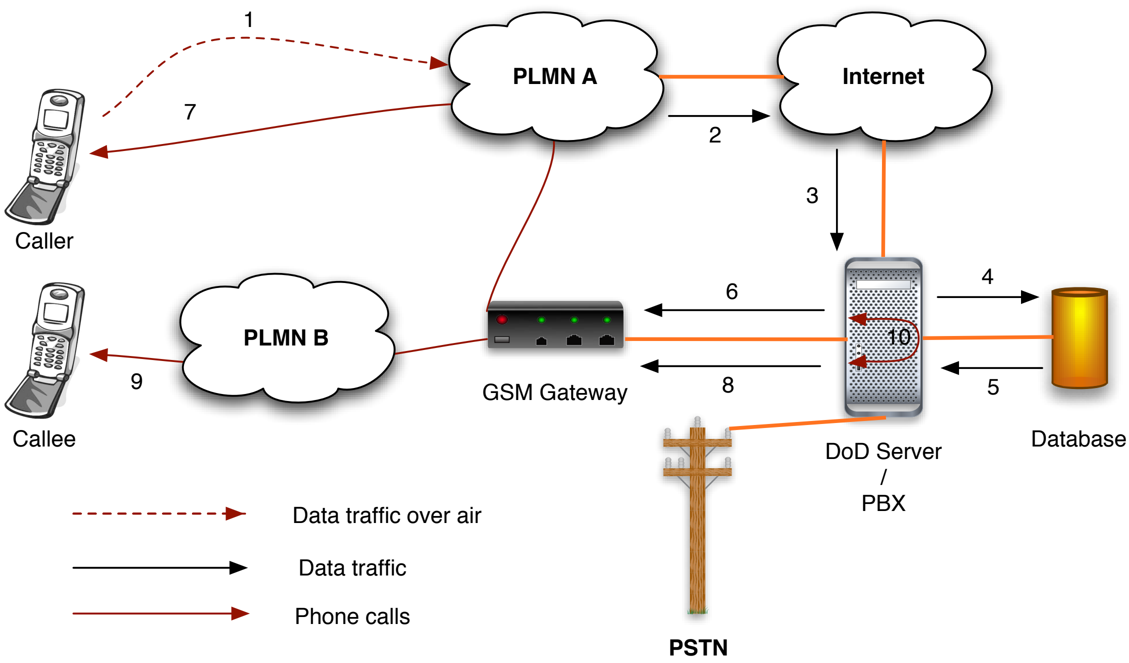
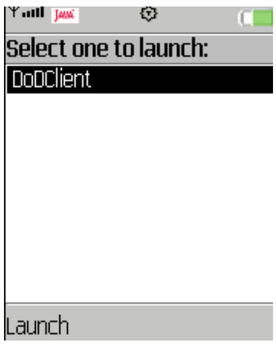


Figure 3.2: Basic operation of the DoD solution

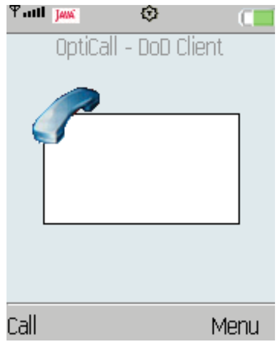
To summarize, the DoD solution provides a way for callers geographically outside the company's network to access it and make use of the internal mechanism used to determine the cheapest way to place a call. The information sent by the interface is processed by the DoD JBoss server and is then forwarded towards an Asterisk PBX. The Asterisk PBX then decides which of its associated resources (SIP registrar, PSTN, GSM gateway) will be used to place a call at the lowest cost, in our case either by the logic provided by the Asterisk dialplan, or by the logic provided by the GSM gateway.

3.4 User interfaces

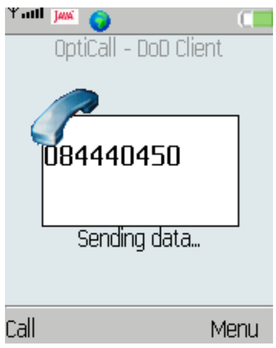
As mentioned in the previous section, the user can access the features of the DoD solution using multiple interfaces: a Web interface over HTML, designed for fully-featured Web browsers such as Mozilla Firefox or Internet Explorer; a mobile client, developed using Java 2 Mobile Edition (J2ME), designed for mobile phones offering support for the Mobile Information Device Profile 2.0 (MIDP 2.0). The user is refered to figures 3.3 and 3.4 for an overview of, respectively, the operation of the mobile client and the Web interface. Later during this thesis project, we added a lightweight version of the Web interface (see section 6.2.1), designed for mobile Web browsers such as those found in mobile phones. To avoid focusing on the actual implementation and coding work behind the interfaces, we will not describe them in more details.



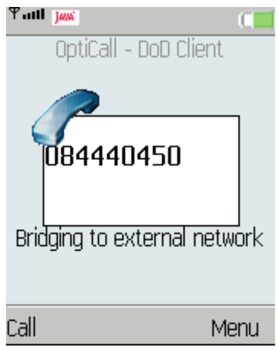
(a) Step 1: Executing the mobile client



(b) Step 2: Main screen of the mobile client



(c) Step 3: Typing in the callee's phone number and sending it to the DoD Server



(d) Step 4: Waiting for the first call once the call information has been sent to the DoD Server

Figure 3.3: Operation of the mobile client

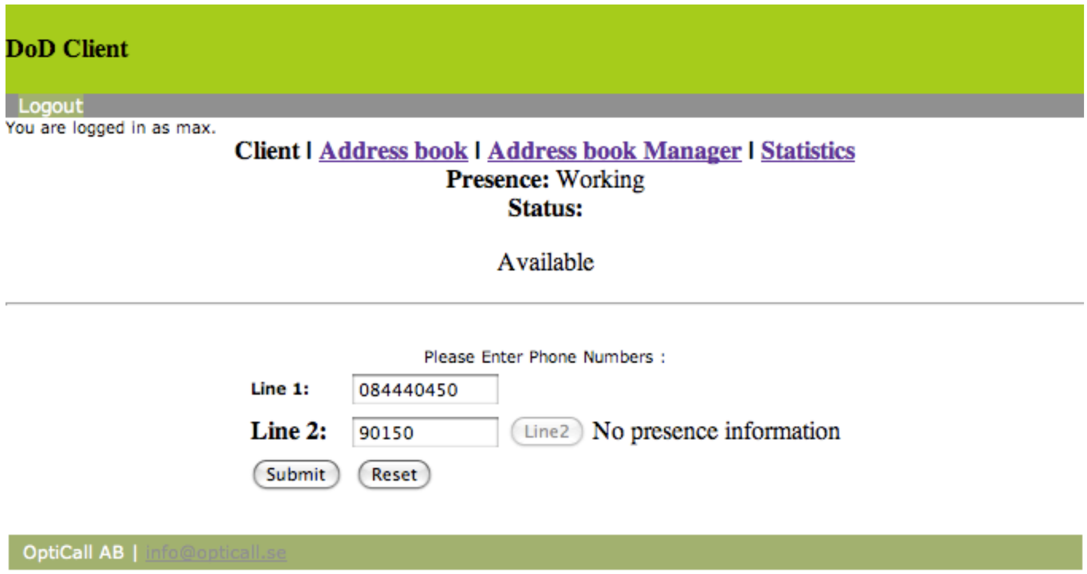


Figure 3.4: Web Interface

3.5 Issues

3.5.1 Latency

As mentioned in section 3.3, latency was initially thought to be an issue. Indeed the GSM gateway might need to activate two new SIM cards previously inactive to establish a call at the lowest rate. The connection of a GSM or 3G module to the network consists of multiple messages [56] as depicted in the figure 3.5. As we can see, the connection of two SIM cards that were previously inactive to the network (according to [2], it takes 10 to 25 seconds to connect a SIM card), followed by twice the time to set up a call, plus the time it takes for the information to propagate from the user interface through the data network, and the processing time of the DoD system - could be a rather long time that could be a deterrent for the use of this product. The amount of time required would also increase in the case of conference calls. However, not all of these contributions to the time need to occur sequentially, hence they are not necessarily strictly additive delays. Furthermore, some of these delays (such as the time to setup a call) could not be avoided even when placing the call in a traditional way.

It is important to notice that the SIM card-activation delay is usually avoided since typical installations of a GSM gateway use only one SIM card per module, to avoid wasting resources (mobile phone subscriptions in this case). However, there will always be the delay to place a call (approximately 4 seconds [1]) even with all SIM cards activated prior to placing a call request. We study the delays encountered by a call through the DoD Server later in section 8.4.1.

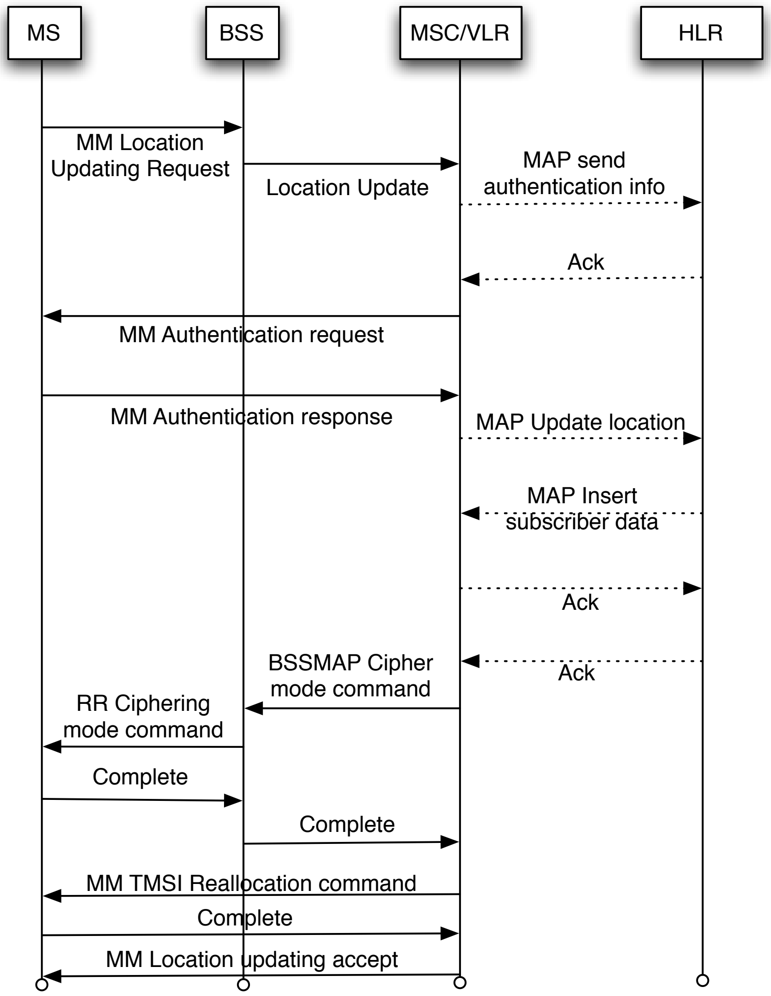


Figure 3.5: Connection to the GSM network

3.5.2 Scalability

The DoD solution aims at a wide range of customers, it should be possible to adapt it to various constraints concerning the existing installations of the company, but also it should scale accordingly to the necessities of the smallest and the biggest companies. The expected load on the DoD solution is very dependent on the customer's demand. For an example of customer's needs, the reader is referred to chapter 5. Scalability of the DoD Server has several limiting factors that are detailed in the next sections.

Theoretical background

When dimensioning this telephony system, the main concern is to have the smallest number of rejected calls. The probability of a call being rejected is given in telecommunication systems by Erlang's B-law. Before enunciating this law, we state some conditions for our system , along with some limitations:

- Call arrivals are assumed to be a Poisson process, which means the interarrival times follow an exponential law (cf. equation 3.1), where λ is the mean number of arrivals per second.
- In our case the interarrival time is limited by the gateway, as there must be a delay of 2 seconds between two calls using the same GSM module according to [2]. Strictly speaking,

this is only valid for the gateways mentioned in that document, but it is likely that the vendor's other products would have similar limitations.

- Call durations are assumed to be exponentially distributed, as shown in equation 3.2, where $1/\mu$ is the mean call length and R_t the remaining time for the call.
- The system's servers are the GSM modules of the GSM gateway.
- The GSM gateway is queue-less. That is if all GSM modules are busy, the call is dropped. In a practical situation, dropped calls could be routed to the PSTN network using the ISDN PRI connection of the GSM gateway.

$$P(\Delta t > u) = e^{-\lambda u} \tag{3.1}$$

$$P(R_t > u) = e^{-\mu u} \tag{3.2}$$

So in the case of a system with a mean interarrival time for calls λ, λ being longer than 2 seconds, a mean holding time $1/\mu$, and m servers, the probability of rejecting a call is equal to:

$$P_b = \frac{\frac{\frac{\lambda}{\mu}^m}{m!}}{\sum_{k=0}^{m} \frac{\frac{\lambda}{\mu}^k}{k!}}$$

For example, assuming an average of 3 calls from a SIP phone to a mobile phone per hour, each lasting in average 5 minutes, the system would need 4 GSM modules in our case, that is to say 2 boards, to reach a grade of service (the probability for a call to be rejected) of 0.001[4] [69]. In such a case, the utilization rate of the system would be of only 6.25 %.

Asterisk PBX

Studies have already been made concerning the scaling of the Asterisk PBX and one can find on the Internet numerous users commenting on their configuration and the number of calls it can handle [100]. Those comments range from the description of a home installation to that of company installations with over 100 clients. According to one of these sources, a machine powered by an Intel Pentium II 400 MHz MMX with 128 MB RAM is enough for one VoIP phone, four computer SIP clients, and a telephone [101]. Two Asterisk servers powered by Intel Pentiums IV 2.6 GHz and 2 GB RAM are enough to handle 30 concurrent SIP to Zap[5] calls (similar to the conferences used in the DoD Server) with 3,000 phone calls per day [102].

Other reports can be used to help the dimensioning of an Asterisk system. Here is a list of some of the important criterions to consider to dimension properly a system [99]:

- the number and types of internal phones connected: SIP, H.323, analog, etc.
- the number and type of external lines: analog, PRI, VoIP, etc.
- the expected number of concurrent calls
- the codecs used for voice treatment
- the features the server should provide: text-to- speech using Festival [63], voice-mail, echo cancellation, etc.

[4] This is a very high grade of service, most commercial systems offer (or try to) a grade of service comprised between 1 and 5 % [38].

[5] In Asterisk, Zap is a module providing access to devices using the Zaptel drivers. Such devices allow connection to traditional analog telephone equipments.

- the expected scalability and reliability of the system: how many dropped calls, what voice quality loss are considered acceptable?
- the quality of the underlying IP network
- the number of servers planned for deployment

GSM network

As noted in [2], one should be cautious when deploying numerous GSM modules in a single location as it might overload the GSM network of one or even several operators, which could prevent the gateways from being fully usable, as employees or by-passers try to place calls using their cell phones. We provide here a brief and theoretical explanation on this phenomenon.

First, some basics concerning cellular networks need to be explained. Figure 3.6 represents a typical cellular network. Operators have a limited range of frequencies at their disposal so they usually divide their allocated frequencies into narrower subranges. Those subranges are then allocated one to each cell, while avoiding using the same subrange in two adjacent cells (in order to avoid interferences). The spatial pattern of reusing the frequencies over the cells is called the reuse pattern. Figure 3.7 illustrates an example of reuse pattern of size 3.

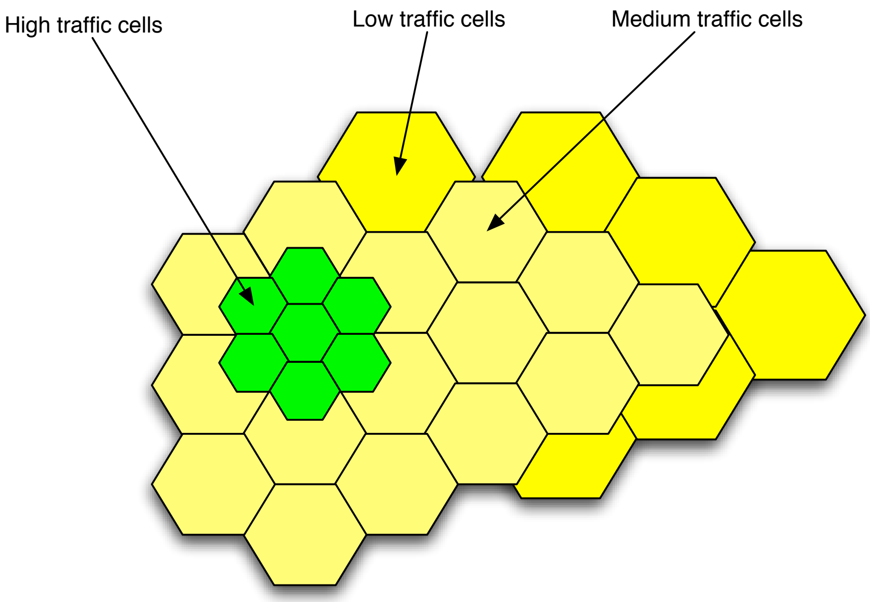


Figure 3.6: Example of cellular network [54]

The frequency span available in a GSM cell is limited, therefore the capacity in terms of call channels for a cell is limited. The smallest cells in terms of radius are the ones that are located in high density areas, such as commercial centers, they are typically a few hundred meters wide. This limitation in turn affects the maximum number of GSM modules from a GSM gateway that can be deployed in a single DoD Server installation, located in a single cell. The channel capacity of a cell is given by the following formula [54]:

$$N_{percell} = \frac{W}{w_u K} \tag{3.3}$$

where W is the total bandwidth allocated to an operator, w_u the spectral occupation for a duplex band and K the size of the reuse pattern. In the case of a GSM network, with w_u equal to 50 kHz and a K equal to 9, this gives a capacity of 2.2 channels per cell per megaHertz of

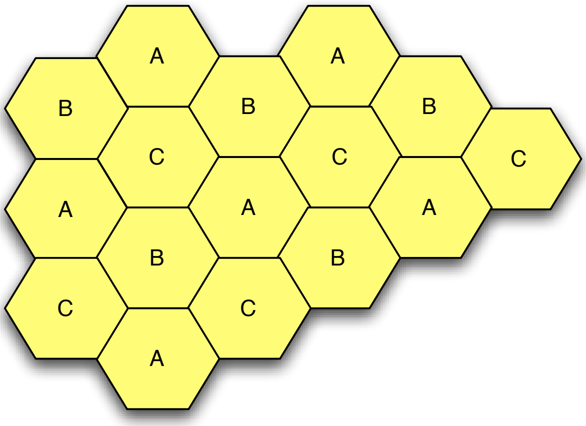


Figure 3.7: Example of reuse pattern

bandwidth [54]. In Europe, GSM can be found under two forms: GSM- 900 and GSM- 1800. GSM- 900 cells use two frequency bands: from 890 MHz to 915 MHz (used for data transmission) and from 935 MHz to 960 MHz (used for data reception) [104]. GSM- 1800 cells also use two frequency bands: from 1710 MHz to 1785 MHz (used for data transmission) and from 1805 MHz to 1880 MHz (used for data reception) [104]. In Sweden, in 2001 [6], frequencies for the GSM network were divided as follow:

Table 3.1: Allocated bandwidth to Swedish GSM operators in 2001 [6]

Operator	Telia	Tele2	Vodafone
Spectrum (in MHz)	2 x 25.2	2 x 18.6	2 x 18.6

The above mentioned result gives an average capacity of 110.88 channels for Telia and 81.84 for its competitors. In our example of a size 3 network, it would lead to respectively 36.96 and 27.28 channels per cell, assuming a uniform repartition of the frequency span between the cells of one operator[6]. This may seem like it provides a fair safety margin, but most offices are located inside crowded areas and it is likely that a significant proportion of those channels are used by normal users, including employees of the company. A wise decision if a large DoD Server is planned for deployment would be to add GSM boards and gateways progressively and check the influence on the quality of service for both normal GSM users and the GSM gateways. Spreading the SIM cards over different operators can also be a solution but might result in more interoperator connection fees. Another solution is, if possible, to install two smaller DoD Servers in distant locations (i.e. in two different cells) rather than a larger one in a single location.

Operators might also limit the number of calls per minute for each SIM card or utilize another parameter to ensure fairness of treatment between users (or to purposely be unfair to users whom they view as abusing their subscription). Therefore further restrictions other than those mentioned in the previous paragraph could apply.

[6]Such a repartition is very unlikely. Resources in terms of frequencies are very scarce and operators usually limit to the minimum the number of channels per cell. One should also remember that usually, one of the channels of each cell is used for signaling.

Table 3.2: Capacity of the 2N Telekomunicace gateways [2]

Gateway	GSM Modules	Minutes per month	Number of constant calls
Blue Tower	8	Over 125,000	2 calls
Blue Star	16	Over 250,000	5 calls
Star Gate	32	Over 500,000	11 calls

GSM gateway

We consider in this report Blue Towers, 2N's gateway. However, other gateways by the same company or other vendors could be used if a greater capacity was needed. Ultimately the GSM network limits the number of SIM cards that can be used in such a gateway at the same time. This limits in turn the total number of calls which can be effectively made via a GSM gateway. This finally caps the number of GSM modules and gateways which are really useful at a given site. Note also that in our case, quite often what is considered a single call from the DoD JBoss server's point of view is actually two calls from the GSM gateway's point of view, as one call connects the caller's mobile and another the callee's mobile. Therefore the overall number of simultaneous calls should be divided by two.

The gateway user's guide [2] indicates the capacity of their various gateways. We have listed these values in table 3.2. Note that the number of constant calls is not an estimation of the number of simultaneous calls possible or expected. It is only a comparison of the number of minutes one could expect from a PRI in a month with the number of minutes one will get from the GSM gateway in a month[7].

JBoss Application Server

The performances and scalability of the JBoss Application Server are linked to various conditions: the parameters of the Java Virtual Machine (JVM) and the parameters of the JBoss Application Server itself, performances of the database it is accessing, hardware configuration (processor speed, amount of RAM), etc. Scaling up the JBoss Application Server can thus be done in multiple ways, including ultimately by clustering of several servers [45]. A study [15] showed that the JBoss Application Server, even without being the highest-performance solution, is still perfectly suitable for this application.

3.5.3 Data traffic

Another point worth consideration is the data connection necessary to initiate the calls. The cost of data traffic via mobile phones can be expensive. Hence it is important to make sure that the data sent over the air to and from the mobile phone is minimal (but not to the detriment of functionality) when the interface is accessed from a mobile phone. One technique to reduce the traffic is caching the pages for a longer time on the mobile phone and giving them a longer validity period on the server. Such a technique could be very advantageous as the DoD interface is not changed frequently when actually used inside a company. Only the contact list might be subject to frequent changes depending on the activity of the user. The contact list might change daily for a sales representative but monthly for a software developer. The frequency of changes also depends upon whether the company shares a single public directory or not. As this page should be kept up-to-date, it is better to use a short caching period, a lightweight style, and split contacts over several pages if there are many of them (depending on their initials, on keywords, etc.).

[7]From these values, one can notice that a GSM module is used at only 35 % per day in average.

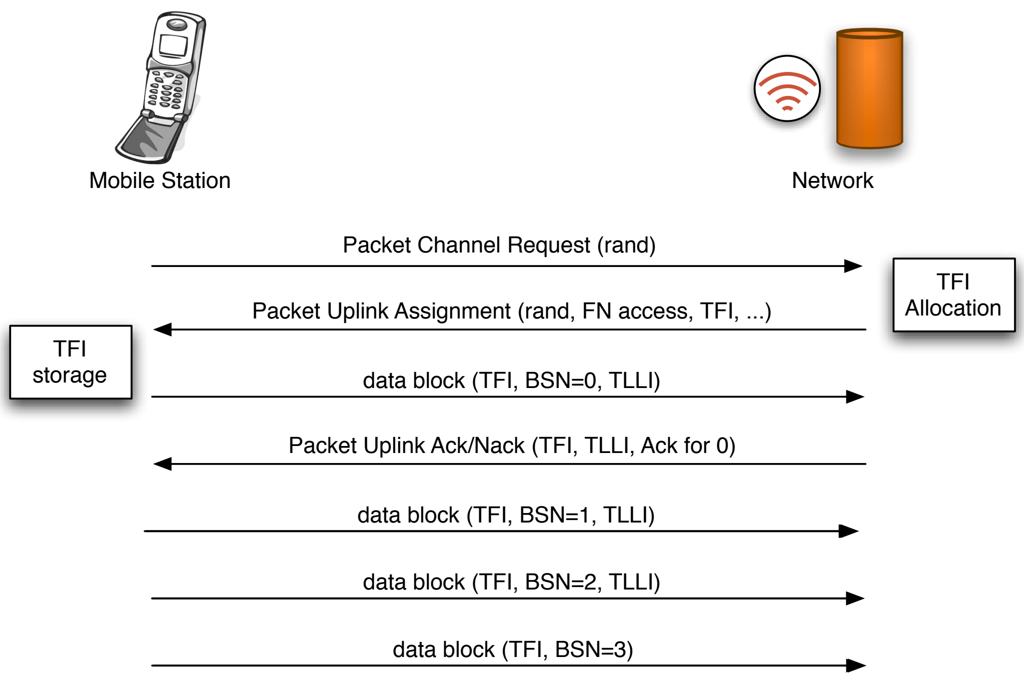


Figure 3.8: Establishing a packet connection to the GPRS network [28]

Table 3.3: Various prices for mobile data traffic in Sweden in November 2007

Operator	Subscription	Access type	Price
Tele2 [85]	Comviq Kontant	GPRS/3G	SEK 15 /MB
Tre [94]	Privat	3G	SEK 10 /MB
Telia [82]	Mobil	GPRS, Edge, 3G	SEK 20 /MB

Figure 3.8 represents the connection of a mobile phone to the GPRS data network. A typical size for a simple DoD session is approximately of 30 KB in download and 5 KB in upload for the non-optimized HTML version. We noted in table 3.3 various prices for data traffic for several mobile operators in Sweden[8]. In these three cases, the user is not charged for maintaining a session open. In the DoD JBoss server case, even using the heavy-weight HTML pages, 30 calls can be placed per month for 10 to SEK 25[9]. It is expected that the savings generated on the actual costs by the DoD solution (see chapter 5) largely compensate these extra costs of 10 to 25 Swedish crowns.

The advantage of the mobile client over the mobile Web interface is that the mobile client needs to be downloaded only once. Therefore, there is no more need to transfer large amount of data between the mobile phone and the server to simply place a call. However, this does not mean the communications between the mobile client and the DoD Server should not be optimized as well. See section 6.3 for more information on the progress made in this area during this thesis project.

In 2006, only 13 % of the mobile phone subscriptions used an UMTS/3G service [55]. The current trend is for an increase in the data traffic exchanged over UMTS connections [55] which makes us optimistic about the fact that soon enough, data traffic will not be a problem which

[8]Different prices might apply for the operator's portals, but are not detailed here.
[9]SEK 10 are approximately €1.10, SEK 25 €2.70

will allow the DoD Server to bring as many features as requested to a mobile client (such as real-time presence of other users).

3.5.4 Detecting non-human users

Other problems that are to be considered lie in the process itself. After the GSM gateway calls the original caller, it initiates the other call, the one to the callee, but should only do so as soon as the original caller picks up the phone. There might be a problem if the caller's answering machine answers the call instead of the user[10]. In such a case, the callee would think he was called by an answering machine which might be unsuitable or unpleasant. Such a problem could occur if another call takes place before the GSM gateway places the first call (that is to say before step 7 on figure 3.2), or if the original caller becomes unreachable, because he has left the GSM coverage zone for instance. Some additional issues also occur during conferences: if one of the callee has her answering machine picking up the phone instead of her, it is that service that joins the conference, which temporarily prevents the other people from communicating until the recorded message has ended (note also that the conference might now be recorded by the answering machine - which might not be desired).

3.5.5 Fraud

Another rather serious problem can occur if the Asterisk dialplan is not properly conceived. In the case of a conference, if the initiator of the conference leaves first, the guests of the conference could stay on the line and could continue talking to each other. While it might be sometimes desirable to let two guests finish their discussion using the company's resources, for the settlement of an important contract or because one of those guests is an employee and should benefit from the DoD solution. The real concern which this failure to terminate some of the calls raises is that if the Asterisk dialplan is poorly conceived, it could result in more serious issues such as the possibility for guests to stay in that conference for a very long time, costing the company money and resources. If the dialplan is not designed properly (or is based on faulty assumptions), a callee, that would stay in a conference and be the last one in that conference, could be returned back to a part of the dialplan where he or she could start placing calls using the company's own telephony network or access employees' voicemail. This is a worst-case scenario, but it points out that Asterisk dialplans have to be properly designed and require appropriate attention when installing the DoD solution. We hope that mentioning these issues here will prompt DoD administrators to design their dialplans carefully, so as to prevent such fraud from happening.

3.5.6 Convergence

The key in a convergence solution is to regroup as many aspects of communications as possible:

1. a single interface, consistent across devices, platforms, and networks and - this interface should also follow the *de facto* (and of course *de jure*) standards,
2. a single identifier or address to give to your contacts,
3. a single voicemail box,
4. a single system to act as a gateway to all other communication channels,[11]
5. a consistent experience for the mobile workforce: unique contact list between different applications, etc.
6. a single solution, suitable for a large variety of companies.

[10] This can happen if the caller has forgotten to turn on his or her cell phone, if he or she is currently outside of a zone covered by mobile phone operators, etc.

[11] This should not exclude the possibility to have a redundant system to prevent failures.

In its initial state, the DoD solution did not offer a high degree of convergence. It offered a centralized means to place calls (item 1 and item 4), but the other aspects were not addressed. Item 2 remains currently one of the most distant goals ; but items 3, 5, and 6 are potentially attainable objectives (for the last two points, the reader is referred respectively to subsection 6.2.1 and section 6.4).

One of the criteria for an attractive convergence solution is the number of different media that are connected to each other by that solution. In our case the limiting factors for the connectivity that should be improved to allow a wider integration of the DoD solution are:

- the DoD mobile interface
  - pre-processes the caller's and callee's identifiers, thus limiting about what media and calls are supported.
  - manages the display of the caller's contacts. It would be interesting to extend this part of the system to make available external contact sources (LDAP, webmail contact list, etc.).
- the resources available to the Asterisk PBX to place the calls. An improvement of this could require significant changes in the Asterisk dialplans. These changes are not necessarily the concern of the programmer, but rather the responsibility of the administrator of the company's Asterisk PBX. However, efforts should be made to make the necessary changes as convenient as possible.
- the DoD JBoss server core program implements only methods to communicate with an Asterisk PBX. However, the modularity of the source code makes it possible to write interfaces for other PBX's. Such extensions are **outside** the scope of this thesis project.

3.6 Goals for this thesis project

The goals for this thesis project are to bring the DoD solution (both the server and the interfaces) to a point where it is a potentially commercial product before testing the qualities of the DoD solution for the issues listed previously (especially delays and scalability, but also reliability). Bringing the DoD solution to a point of commercialization includes for instance the improvement of the initial features, the addition of new ones in order to offer more than only cost reduction, and the studying of the possibility for the DoD solution to interface with various telephony systems through the DoD Server, other than a GSM gateway. The quality of the product should then be evaluated by a series of tests.

Chapter 4

Alternatives

In this chapter, we will describe in a first time the various technological alternatives that could be interesting for the DoD solution to follow. In a second time, we will present a few competitor products and technologies. As parts of the Dial over Data solution are parented and as most those competing products are also patented, this presentation of competing solutions will only serve as elements of comparison for the reader.

4.1 Technological alternatives

4.1.1 Alternatives to the GSM gateway

It is important to emphasize here that a GSM gateway is **not** a mandatory component of the DoD solution. A GSM gateway is in fact not necessary to place calls to mobile phones, but it is likely to be the best way to provide low cost for a small volume of mobile calls. Loosely speaking, a GSM gateway can be replaced by a cost-efficient fixed line or a SIP trunk providing low tariffs; it all depends on the needs of the customer. The GSM gateway is expected to improve cost savings, but its presence is **not** a necessity to benefit from the DoD solution. Even without a GSM gateway, the use of the DoD solution gives back to the company a central role in the communications of its employees. It is also important to note that this report mention exclusively 2N Telekomunicace's gateways as the DoD solution was developed using these gateways. It is however **not** a requirement to use a GSM gateway of that brand, should a company decide to use a GSM gateway with the DoD solution.

Using a GSM gateway offers many possibilities, choices of different interfaces (proprietary software, web interface), and different configurations (number of boards, type of boards: GSM, 3G, VoIP, or PRI). The reader is referred to section 3.5.2 for examples of the impact of the number of boards and/or gateway. Smaller companies could replace a GSM gateway by a pool of mobile phones connected to the Asterisk PBX over Bluetooth [12] using an extension known as `chan_mobile` that would be in charge of the calls towards the callers and the callees involved in DoD calls. This could be a viable solution when no features (such as least cost routing) are expected, except for those functions provided by Asterisk. However, no efforts have been pursued in this direction during this work. We simply mention here the `chan_mobile` project's description ad refer the interested reader to the project's website [17] for more information. This website states that: "chan-mobile is an Asterisk channel driver that allows you to use Bluetooth devices as [Foreign eXchange Office (FXO)] or [Foreign eXchange Subscriber (FXS)] channels." The advantage of this solution would be that:

1. it avoids the cost of a GSM gateway.
2. it is possible to use a handset which has a bundled subscription - hence the operator is fronting all or part of the cost of the handset(s).

4.1.2 Alternatives to SIP

Other VoIP protocols are available, such as H.323, and proprietary protocols such as Skype. A choice made by my predecessors [109] was to focus on SIP, because it is the most widely used VoIP protocol in industry products. Nevertheless many products still implement H.323 and it may be beneficial to handle this protocol in the DoD Server in the future. The DoD solution should allow a large variety of users to connect at the lowest price to any other user. So, there are three primary reasons to add additional communications standards: (1) to enable users to place a call from whatever device and service they can access, (2) to enable callers to place calls towards anyone, and (3) to exploit a wide variety of options so as to chose the lowest-cost way to route the call.

4.1.3 Alternatives to Asterisk

The DoD solution is extremely reliant on communicating the information entered by the user via the user interface to an Asterisk server. Other PBX software could have been chosen for this project such as Brekeke [13] or Callweaver [14] (formerly OpenPBX). However, Asterisk is the most widely used open-source PBX so it is a logical choice to begin implementing solutions using it. Thanks to the modularity of the code of the DoD Server, it would not be difficult to adapt the project to Callweaver's or Brekeke's manager interface if a real need for this were to occur.

Even though Asterisk is used in the DoD Server as a SIP server, Asterisk only behaves as proxy and does not offer all the functionalities one might expect. More precisely, Asterisk merely acts as a back-to-back user agent in the middle of the conversation [109]. The DoD JBoss server places all calls to the Asterisk server, so it is necessary that the Asterisk server is part of the SIP network. Nevertheless, it is possible to add additional components, such as a SIP Express Router (SER) [40] (cf. chapter 7) to provide better performances and scalability (cf. section 3.5.2).

4.1.4 Alternatives to a Java-based solution

The server

JBoss Application Server is only one solution among others; alternatives include JOnAS [49] or Geronimo [27]. A complete list of all JavaEE-compatible servers is maintained by Sun Microsystems [59] for several versions of J2EE. While JBoss is not certified to be compatible with the latest JavaEE 5, it is certified for the earlier J2EE 1.4 and benefits from a large community of users and contributors. The JBoss Application Server does not offer the highest performance [15], at the moment the decision was made to use it, JBoss was the most promising solution available on the open-source market.

The Asterisk API

The interface to Asterisk's manager has been coded in Java during the previous stages of this project. Even keeping a Java-based solution, there are other solutions that could have been used, namely several alternative APIs. They were reviewed during an early thesis project [109] so we only discuss here the most significant one.

Asterisk-Java combines both an API and an Application Gateway Interface (AGI) (cf. below in this section 4.1.4) for Asterisk in Java [44]. It is also the most complete API for Asterisk today in Java to our knowledge. It works at the socket-level and is independent of the Asterisk version used. Prior to this thesis, the choice was made to develop a limited subset of an API quite similar to Asterisk-Java. As the functions the DoD Server offers kept increasing, more message types needed to be supported (sent to and received from the Asterisk server). Therefore it was deemed to be a wise choice to stop using the old home-made API and instead use a more complete one: Asterisk-Java. More over, the former solution relied on certain assumptions concerning the messages to and from the Asterisk server. Those messages were bound to change (and had indeed changed) between different versions. This was a further reason to adapt our project to use Asterisk-Java.

Asterisk Application Gateway Interface

While the Asterisk API allows external applications to issue commands and receive data from external programs, the Asterisk AGI allows the Asterisk server to use external programs, most often scripts. As all calls are originally placed by the DoD JBoss server, it is impossible to use only the AGI to process the calls. However, the AGI can complement the DoD JBoss server to adapt to special conditions. For example, the AGI can add easily another level of processing of the call-related information between the Web interface and the beginning of the actual call. Ideally all the processing should be done inside the DoD JBoss server, potentially by adding separate modules, but not all modifications might justify of an implementation in the DoD JBoss server itself and the smaller customizations will be done on a case by case basis. The AGI could not fully replace the JBoss server as AGI applications need to be called from inside Asterisk which would prevent them from offering a user interface at the worst, at the best it would split the current JBoss server in two parts: a simple web interface to trigger an action into Asterisk and a complex AGI script to process this action and derive the actual call from it.

Thread-level integration

For the sake of completeness, we need to mention a third type of solution: the integration between Java and Asterisk at the thread-level, such as done by JAsterisk [43]. The project seemed to have been stalled since 2005 and has limited or no support. Furthermore it would require a larger modification of the potentially preexisting Asterisk installations at the customer's premises.

4.2 Other solutions

4.2.1 Competing products

The method described in section 3.2 for a mobile user to place a call to a caller is only one way to address the problem of remotely-initiated calls. This specific method is patented in Sweden [33] and awaiting a patent with a broader application zone [32][1], but other methods are perfectly conceivable. Other systems have indeed been developed concerning the convergence of telecommunication channels, including the mobile channel. The company Telepo [87], current leader of the Fixed-Mobile Convergence (FMC) market according to [30], FirstHand Technologies [84], and others are working on such solutions, using a wide range of different technical solutions to achieve the same goals.

Telepo and OnRelay's Mobile Branch eXchange: different methods

The method used by Telepo AB is depicted in figure 4.1. It consists of the following: the caller first sends a data transmission containing the necessary data to contact the callee. Then the caller has to call a specific number from her mobile phone (or distant land line), then the distant PBX places a call to the callee, to the caller, and eventually bridges both calls. That solution requires two actions by the user while only one is really necessary. Besides, this method does not always result in the lowest possible costs. Indeed, the call between the caller and the distant PBX has a fixed end (the distant PBX), and the caller might want to originate a call using any phone[2], in case her cell-phone battery is dead for instance. In such a case, the advantages derived from any previous arrangement to have the lowest cost possible between the distant PBX and the caller's mobile phone are wiped out for these calls.

[1] This explains why in the rest of this section we will only compare the DoD solution to other solutions and not try to revise the core operation of the DoD solution.

[2] Such as a public phone, a phone located inside a customer or partner company's premises, etc.

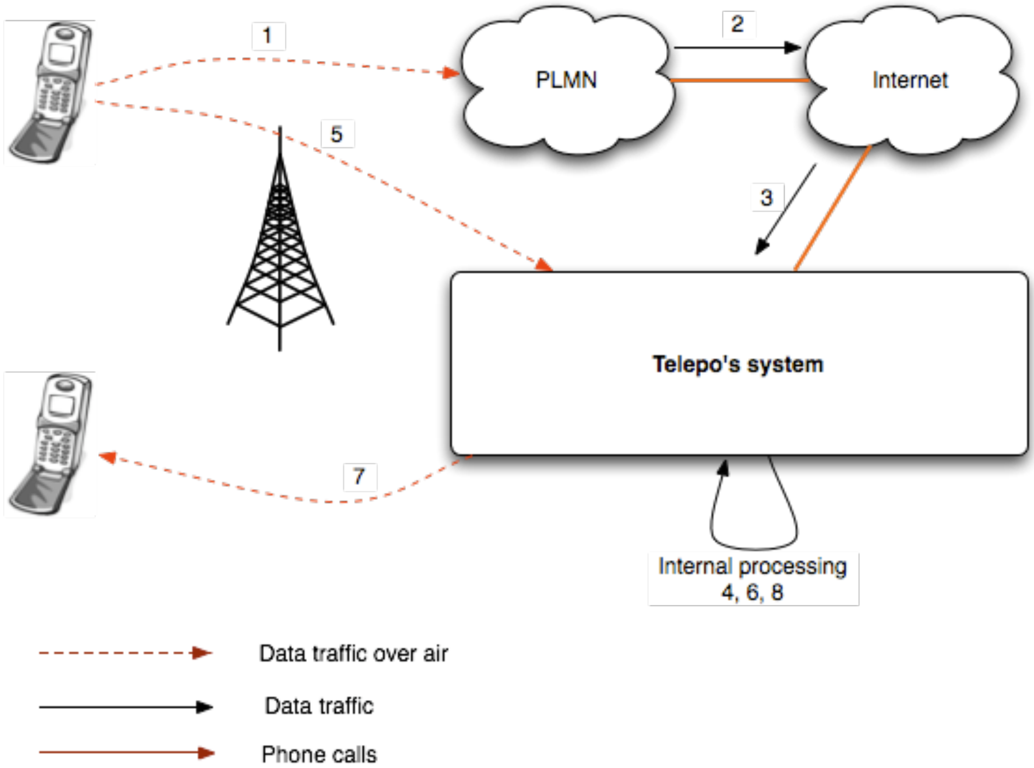


Figure 4.1: Principle of Telepo's solution

The method patented by OnRelay [68], called Mobile Branch eXchange (MBX), is quite similar to the method described in the previous paragraph. The mobile phone of an user sends a data request to the MBX containing the phone number of the second party and, at the same time, over a different pathway, call the MBX which will then connect this call to the outgoing call to the second party. OnRelay's method claims to bring the PBX features into the mobile phone, in the form of a thick client. This enables the mobile user to forward calls or manage conferences, among others, which should obsolete the use of desk phones in companies using OnRelay's system, according to OnRelay. Yet again, the necessity to place a call to a specific number might not always provide with the lowest cost.

SICAP Prepaid Roaming: an operator-oriented product

The Swiss company SICAP proposes a competing solution called Prepaid Roaming [75]. On the one hand, it achieves the same purposes as the DoD solution. That is to say being available on any cell phone, having an anti-fraud system, and avoiding any need to replace the company's fixed and mobile phones; among others advantages. On the other hand, it seems to be oriented towards mobile operators: prepaid users are the main target and the product is designed to interface with GSM networks at a lower level than the DoD solution. The objective of the DoD solution is to offer a solution to companies, while, the services SICAP provides are mainly of interest to operators.

Calling cards: two-step dialing

Another solution known by frequent travelers is the calling card. It is generally a prepaid card that offers cheap rates from any phone. Usually this service is primarily sold for callers wishing to make international calls. The principle is to dial the phone number of the service provider, followed by a PIN code or some other sort of identifier, and then dial the number the user wishes to be connected to. Again, this is rather cumbersome and does not allow the user, for instance, to use stored numbers from her mobile phone without appending a prefix to all her existing directory. This results in a long and/or tedious transition period to switch to the calling-card system, especially if the changes have to be made by each users.

Actually some mobile phones offer an integrated management for their calling cards, but this is not currently a feature common to all cellular phones. For these phones, the user simply needs to register the necessary informations (phone number, identifier) related to the calling card, then the phone automatically and seamlessly dials all future calls using this calling card. For phones lacking these calling card features, some programs [61], [47] are available as a substitute[3].

4.2.2 Complementary products

Seamless handoff

Several companies such as FirstHand Technologies [84] and OptiMobile AB [66] offer seamless handoff solutions for dual-mode phones. These companies are often wireless access point providers and they have devised a mechanism to pass on an active conversation from a WiFi network using a VoIP protocol to a GSM network. This applies for instance when the user answers a call in his or her office, but leaves the premises during the call, for instance to go and visit that customer (cf. figure 4.2). This solution would be of interest to complement the DoD solution. While both have the same goal - lowering telephony costs of companies - these solutions are not interchangeable. Indeed the seamless handoff mechanism does not reduce costs for calls originated outside. In fact, using only the seamless handoff mechanism, a caller located outside the company's network does not get access to the logic that determine the lowest-cost way to place a call – as this logic is located inside the company's network.

Unlicensed Mobile Access

Another convergence solution, quite similar to the seamless handoff, is provided by the Unlicensed Mobile Access (UMA), also known as Generic Access Network (GAN) [95], backed among others by Kineto Wireless [107]. In the case of the UMA technology, the handoff between the WiFi network and the GSM network is done without changing protocol. Indeed, UMA enables the GSM signaling to be carried over so-called unlicensed media (such as WiFi or Bluetooth). There are currently only a few UMA-enabled phones [106], but a recent decision of TeliaSonera to provide home access points (called femtocells) using the UMA technology to improve 3G coverage could change the current trend [96]. Note that the UMA technology requires the involvement of the operator to enable the GSM traffic to be handled continuously, whereas the seamless handoff mentioned previously does not require such an agreement and can therefore be deployed more easily.

4.2.3 Design alternatives

Several options are available for the user to request a call from the central server:

- an interface over IP: this solution is used in our DoD solution.
- a Short Message Service (SMS): this solution could be used together with the DoD interface on a mobile phone, sending the commands via one or more SMS messages to the DoD Server instead of sending it over IP. The problem with this solution is that it requires two different implementations (one SMS-based for the mobile phone client and one IP-based for the Web client). Another problem with SMS-based solutions is that SMS messages are delivered with a "best effort" level of service [81], which means the request could be delayed for some

[3] Only for phones utilizing the Symbian [83] platform. Unfortunately, J2ME was not designed to access most advanced features of mobile phones (such as the audio channel used during the calls, as well as the calling/called phone numbers), for security reasons.

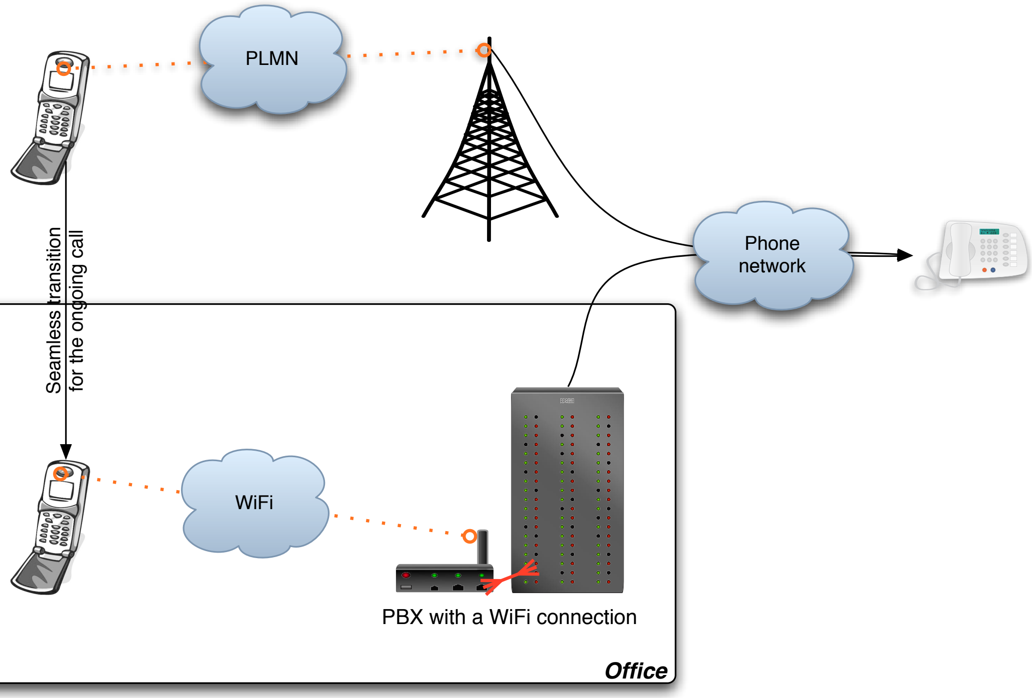


Figure 4.2: Principle of the seamless handoff

time or simply never reach the intended recipient. The validity period of an SMS can be between a few minutes or a few weeks, it can also be an absolute date [62]. This means the call from the JBoss server might be placed not only five minutes later, but the next day and without the user's knowledge or control.[4]

- a phone call followed by digit input: this is rather cumbersome and requires either that the user navigates through complex menus and/or to know by heart the numbers of all his or her contacts. As mentioned in subsubsection 4.2.1, there is no global solution to automate this input.

- Unstructured Supplementary Service Data (USSD): this solution is used in the SICAP Prepaid Roaming solution [75]. USSD could be described as real-time SMS: the service provides the ability to send information over the GSM network without going through a store-and-forward service such as SMS. The real-time aspect makes USSD more suitable for our purpose than an SMS that can be lost or variably delayed. Further more USSD are always routed through the Home Location Register, which means there is no change for the service even when abroad. Additionally, the service does not involve the Short Message Service Center (SMSC). SICAP's solution is to send data back to the home network concerning the call the user wants to place. This translates concretely into the user typing on her phone's keypad a string of digits such as ∗`101`∗`0701234567`# [24]. There are no specifics about the actual interface on SICAP's online description of their product [75], but it is likely to offer a visual interface more convenient than the mere keypad. This solution operates at a lower level than the one selected to operate the DoD solution, but it probably offers the most viable alternative of those presented in this list.

[4]According to the Java Specification Request 120 [51] concerning the Wireless Messaging API, it is not possible to change that validity period from within a MIDP application. However, this is possible on Symbian phones.

Chapter 5

Case study

5.1 Introduction

This chapter describes a virtual case study of a company (Acme, Inc.)[1] migrating to the DoD solution. We first describe the initial situation of the company, then discuss the dimensioning of a suitable DoD solution, were it to be installed in this company. Finally we calculate the expected savings realized by this company if switching to the DoD solution. Note that the currency unit used in this chapter is the Swedish crown (SEK 1 ≈ €1.1 ≈ $1.5[2]). Note also that this study does not discuss the case of SMS messages and calls placed while roaming in a foreign operator's network, as this is not a situation addressed by the DoD solution. Existing data traffic over GPRS or UMTS is also not considered. Only the traffic generated by the use of the DoD solution are accounted for later in this study.

5.2 Initial situation

Acme, Inc. has 30 employees that use a mobile phone paid for by the company. The total duration of their 2,480 calls over a month is equal to 6,500 minutes. These 6,500 minutes are divided as follow: 3,750 minutes represent the 1,515 calls made to a mobile phone and 2,750 minutes represents the 965 calls to a fixed phone or a phone located abroad. A month is here considered to be equal to 255 hours, corresponding to the hours worked by the employees. This means there is approximately a call every 6 minutes and a total traffic of 0.42 Erlangs shared among all these mobile phones. All these values are summarized in table 5.1.

Each one of the 30 employees has a subscription to a mobile phone operator, costing a monthly SEK 280, for a total of SEK 8,400. The extra expenses due to phone calls not included in the monthly subscription amount to SEK 7,000. These extra costs are divided as follow: SEK 3,800 for calls to mobile phones and SEK 3,200 to landlines and international numbers. On the whole, Acme, Inc. is currently spending SEK 15,400 on mobile telephony costs every month. This means Acme, Inc. is spending approximately SEK 515 per employee having a company mobile phone per month, or only SEK 450 per employee (SEK 13,500 in total) excluding international calls. These two last figures are the figures that interest us.

5.3 Dimensioning

In this section we determine the appropriate equipment (in our case, the number of GSM gateways and GSM boards) to deploy in the case of Acme, Inc. We do not consider dimensioning issues related to the second leg of calls to fixed phones and international phones. In this particular example, the routing of calls to external landline phones, in Sweden or abroad, is left to the

[1] Figures are based on an actual company.
[2] Approximate values taken December, 20th, 2007.

company. As a result, Acme, Inc. would have to seek a provider offering the lowest tariff rates for landline-to-landline calls and for the provider offering the lowest rates for international calls.

Considering that all mobile calls are now addressed by the DoD solution, and that only calls to Swedish mobile phones are placed using the GSM gateway[3], Acme, Inc.'s future GSM gateway will need to handle 9,900 minutes per month representing 3,390 calls (3,030 calls for mobile-to-mobile calls and 360 calls for mobile-to-landline calls). We calculate that this telephony system must support an arrival rate of 13.29 calls per hour and a service rate of 20.55 calls per hour. With a quality grade of 1 %, that is to say one phone call is dropped every hundred phone calls, we can calculate that Acme, Inc. should have four lines [69]. In a GSM gateway, four lines are obtained by the use of two boards, each giving two modules simultaneously usable. Referring to table 3.2, we can see that four modules could support up to 62,500 minutes a month. Compared to the 9,900 minutes actually needed, this is an utilization rate of 15.8 % of the GSM gateway. We summarize the case in table 5.1. The findings of this section are summarized separately in table 5.2.

Table 5.1: Case study

Monthly traffic	Type	Number of calls	Total duration
Without DoD	Mobile to mobile	1,515	3,750 minutes
	Mobile to others	965	2,750 minutes
With DoD	Mobile to mobile	3,390	9,900 minutes
	Landline to landline	360	2,400 minutes
	Landline to international number	*605*	*350 minutes*

Table 5.2: Dimensioning of the DoD system

Parameter	Value
Arrival rate	13.29 calls per hour
Service rate	20.55 calls per hour
Grade of service	1 %
Number of lines needed	4
Minutes needed	9,900
Minutes provided	62,500
Rate of utilization	15.8 %

5.4 Savings

As we mentioned in the previous section, the DoD solution does not directly address cost savings for calls to landlines and international calls, even though it potentially enables cost savings on these calls. We give an estimation of the cost savings for calls to landlines, without implying that this is the best that could be achieved. It is up to each company to find the best offers from telecommunication operators for its particular needs. Similarly, cost savings on international calls are not discussed at all since it is largely dependent on the countries called. Comparing all telephony offers (including triple-play offers) is not the goal of this thesis.

The intended GSM installation at Acme, Inc. is composed of four lines and there are four mobile operators that are currently frequently called by Acme, Inc. We call them A, B, C, and

[3] Calls to landlines and international numbers are sent to the regular ISDN line of Acme, Inc.

D. Currently, the mobile subscriptions paid for by Acme, Inc. are subscriptions to operator A, which is the operator whose phones are the most frequently called. A simple solution would be to have one SIM card of each operator in the GSM gateway, thus offering the four necessary lines. However, it is likely this solution is not optimal since there is a large difference between, for instance, the traffic exchanged with operator A and the traffic exchanged with operator D (the ratio is of 13 to 1, see table 5.3). Several other solutions are then possible: increasing slightly the number of lines to have at least one SIM card for each operator, but still approximately match the respective proportions of the traffic to each operator; using two SIM cards from operator A, one for operator B, one for operator C, and none for operator D; or subtle variations between the previous solutions.

Table 5.3: Repartition of the traffic per operator

Operator	A	B	C	D
Traffic (arbitrary unit)	13	12	6	1

We describe here an intermediate solution. We consider a best-case scenario using two SIM cards for operator A and one SIM card for each other operator. By best-case scenario, we mean that for each call, the SIM card providing the lowest rate is always available. Among other conditions, this translates into "an employee can call a phone from operator A if and only if two employees are not calling at the same time". The details of the various costs and costs savings are presented in table 5.4 and in figure 5.1.

It seems necessary to comment on this table. When the DoD solution is applied, there is only the connection fee to pay[4]. This explains why costs are calculated per call in the second part of the table (with the DoD solution), and not per minute as in the first part (without the DoD solution). The new number of calls placed to mobile phones belonging to operator A is now $645 * 2 + 500 + 300 + 70 + 360 = 2,520$. This reflects the fact that all calls have a leg in the network of operator A ($645 + 500 + 300 + 70 + 360$[5] $= 1875$) on top of the calls that were already addressed to operator A without the DoD solution (645). When considering data traffic, we consider that the employees are using the DoD MIDP client, thus generating 2 KB of data traffic per call. There is no longer a need to buy post-paid subscriptions for the employees, each user only needs a prepaid card with an attractive price for data traffic. In this case, we assumed a prepaid offer from operator A to keep the analysis simple. Note also that we are using similar prepaid cards in the GSM gateway to benefit from a connection-fee only tariff to mobile phones in operator A's network, while we use subscriptions from other operators to obtain suitable commercial conditions. This set of choices reflects the Swedish market at the current time.

5.5 Conclusion

With the DoD solution, Acme, Inc. can save over SEK 10,000 a month and reduce their mobile telephony costs per employee from SEK 450 to less than SEK 110. This represents a reduction of 75 %. The savings described above could increase if calls placed to internal phones and to international numbers were included in the calculations. The savings could also be increased by negotiating better telephony offers. On the other hand, some costs are to be deduced from these savings: the costs related to the GSM Gateway and the DoD solution. These costs were not included as the pricing of the DoD solution has not been fixed yet. This study also assumed a best-case scenarios where five SIM cards were enough to provide the best price available for every single call. This might not occur in a real case, depending on the calling patterns of the employees.

[4] A small fee paid for every single call, no matter the duration of the call.
[5] Calls to a landline phone

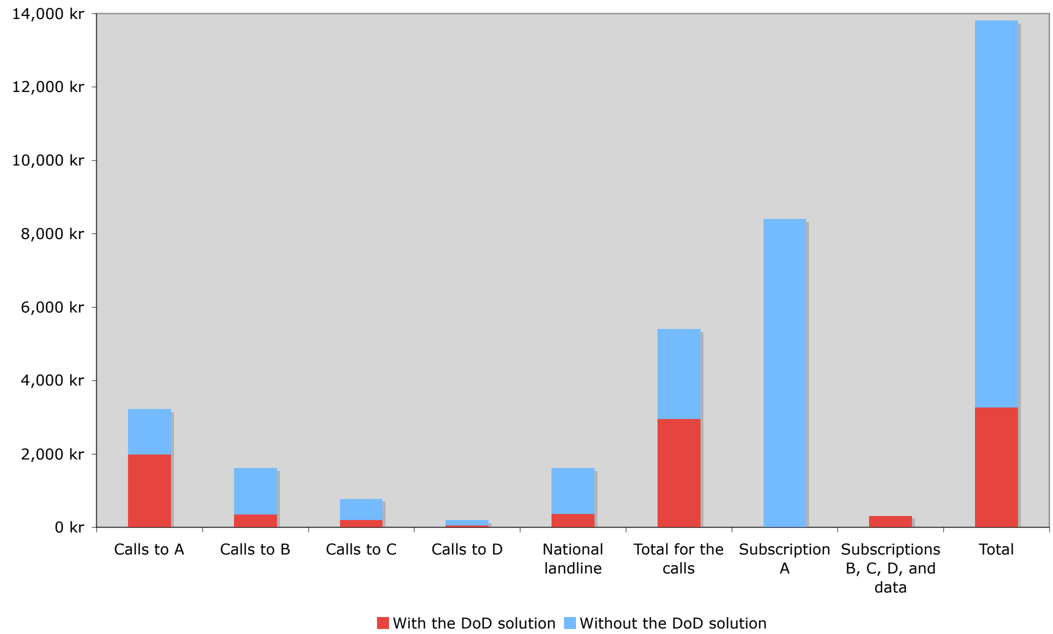


Figure 5.1: Telephony costs at Acme, Inc.

Table 5.4: Telephony costs at Acme, Inc.

	Type	Cost per unit	Number	Total
Without DoD	A	SEK 0.79	1,550 minutes	SEK 1,224.50
	B	SEK 1.17	1,400 minutes	SEK 1,608.75
	C	SEK 1.16	670 minutes	SEK 777.20
	D	SEK 1.45	130 minutes	SEK 188.50
	National landline	SEK 0.67	2,400 minutes	SEK 1,608.00
	Total for the calls	—	*6,150 minutes*	*SEK 5,406.95*
	Subscription A	SEK 280	30 subscriptions	SEK 8,400
	Total	—	—	**SEK 13,806.95**
With DoD	A	SEK 0.79	2,520 calls	SEK 1990.80
	B	SEK 0.69	500 calls	SEK 345.00
	C	SEK 0.69	300 calls	SEK 207.00
	D	SEK 0.69	70 calls	SEK 48.30
	National landline	SEK 0.15 [55]	2,400 minutes	SEK 360.00
	Total for the calls	*12,300 minutes*	*3,750 calls*	*SEK 2,951.10*
	Prepaid card A	Already paid for above	2 prepaid cards	SEK 0
	Subscription B	SEK 69	1 subscription	SEK 69
	Subscription C	SEK 49	1 subscription	SEK 49
	Subscription D	SEK 69	1 subscription	SEK 69
	Data	SEK 15	8.64 MB	SEK 129.60
	Total	—	—	**SEK 3,267.70**
Savings		**SEK 10,539.25**		

Chapter 6

Improvements

6.1 Basics

What early customers will first be looking for in the DoD Server product are its functionalities. Therefore, the conference capabilities have been extended, among other services offered. Other functions that are expected from such a solution include presence notifications and call forwarding. This services are made possible through SIP. Presence notification will be detailed later in chapter 7. The DoD solution offers functions adapted for professional use, for users, managers, and administrators of the DoD solution (cf. figure 6.1).

The administrator
- has access to full settings through a dedicated Web interface. Timeout periods, dialplan contexts, and the number of conferences simultaneously allowed can be adjusted to suit the needs of the company. Users and mobile operators can also be managed via this interface.
- The manager
  - has access to detailed statistics for each user as both raw values and graphs, to allow monitoring of user activity. The manager can also access aggregated values per day, week, or month for each user.
  - The user
    - can place calls and manage their contacts. He or she can also access a small subset of the statistics concerning the user him- or herself.

Figure 6.1: Access level model for the DoD interface

The statistics are gathered using the call detail record (CDR) informations collected by Asterisk, therefore potentially considerable amounts of valuable data about the company's communications are available. This same information could be used for the verification of the proper functioning of the DoD solution, for example the following could be extracted: numbers called by each employee, duration of each leg of the call, eventual errors during the call, etc. These statistics are currently displayed both as graphs and as pure numerical values. Contact with companies who adopt this solution will help defining the most vital informations to be extracted from the CDR, currently, the provided values are:

- for the manager and the administrator:
  - last call's duration, total duration of the calls over the previous day, week, and month for each employee as well as the average length of this employee's calls (as text)
  - for each employee: average length of the calls, total duration of the calls for the day, the week, the month (as graphs)
- for the user:
  - as text: the duration of the last call, and the cumulated duration of the calls placed during the previous day, week, and month

Different type of calls can be placed by the users. Here we will not discuss the media that can be used (SIP, PSTN, etc.), but rather focus on the functions themselves.

Single calls The status for each call and line can be seen during the conversation. This function is available only in the Web interface.

Conferences involving several callees can be created and lines hung-up separately by the caller. Status can also be shown for lines and calls during conference calls.

It is here important to note that the implementation of Asterisk has some side-effects on our possibilities. For instance, when an extension (such as a mobile phone or a SIP phone) is ringing, there is no way of knowing from inside the Asterisk PBX who the caller is, which means there is no certain way to know if the ringing is the consequence of a DoD operation. Therefore the status of a ringing line is uncertain and hanging-up a ringing line is prohibited inside the user interface to avoid losing non-DoD calls.

Presence has been roughly integrated in the DoD solution. Through the Web interface, each user can set a presence state as well as a "Do not disturb" call, flag that prevents this user from being called using the DoD solution when this flag is activated. The whole presence integration is based on the DoD database (specifically the users' table) and dedicated servlets for the back-end, and Asynchronous JavaScript and XML (AJAX) for the Web front-end. Presence is also to a lesser extent integrated in the mobile interface and the lightweight Web interface.

6.2 Interfaces

In this section we describe the Web and mobile interface in their current state at the end of this thesis project. This section focuses only on their features rather than on how to use these two interfaces. A deployment schema summarizing the location of the various components of the DoD solution and their various interactions is located at the end of this chapter, in figure 6.3.

6.2.1 Web interface

At the end of this thesis project, two Web interfaces exist: a full Web interface for browsers running on computers and a lightweight interface for browsers running on mobile phones. The first interface is described throughout this thesis, so here we only describe the lightweight interface (see also section 6.3). The lightweight interface was added to allow users to access the DoD interface on borrowed mobile and rental phones when on the move. However, the browsers found on mobile phones are usually quite limited and it is advisable to download and install the DoD mobile client if the DoD system is to be used regularly from a specific mobile phone. Not all features of the full Web interface were retained for the lightweight interface: statistics are less exhaustive, administration is not possible, the contact list is read-only, etc. but the look-and-feel of the fully featured Web interface is preserved.

6.2.2 Mobile interface

Because of its nature, the mobile interface cannot offer as many possibilities as the Web interface. First of all, there is no possibility to remotely manage the system using the mobile interface; there is also no access to the full statistics (only to individual elements of the statistics). As a result, the mobile interface is privilege agnostic: administrators, managers, and users use the same interface with the same possibilities. A generic user can place calls to IP or H.323 phones, mobile or landline phones[1], send SMS messages, and consult the history of his or her calls. It is possible for the user to configure (to some extent) the application (which server to use, username and password associated with the user, font size of the outgoing call history). The look-and-feel of the menus depends to a large extent on the mobile phone, but the functions and their associated shortcuts[2] are consistent across handsets. It is possible for the developer to change the design of the application to match the specific needs or graphic requirements of a customer.

6.3 Data traffic

Efforts have been made during this thesis project to reduce the data traffic sent over expensive media, such as the GPRS link. Therefore a lightweight Web interface has been designed for mobile browsers. Another improvement is due to the release of Opera Mini [64]. This browser enables a reduction of the size of Web pages by up to ten times. However this has a cost: the data needs to be processed by Opera's own servers, which makes it impossible to use HyperText Transfer Protocol over Secure Socket Layer (HTTPS) connections. Opera Mini is an attractive alternative to provide a DoD Client for mobile phones for which there was no time, possibility, or desire to run exhaustive tests of a MIDP client, for instance MIDP 1.0 mobile phones. Additionally improvements have been made to the Java MIDP client by using return codes instead of plain-text-strings transmissions.

We can compare the typical data traffic for the normal version of the Web interface to that of the lightweight Web interface, with both a normal browser and Opera Mini[3]. Table 6.1 shows that the lightweight version has reduced data traffic by a factor of two. We can also see that Opera Mini does not bring much of an improvement on the lightweight version, presumably because the light weight version has already been drastically streamlined.

Table 6.1: Compared data traffic

Interface	Downloaded data	Uploaded data	Total	Gain compared to normal
Normal	30 KB	5 KB	35 KB	–
with Opera Mini	16.6 KB	12.5 KB	29.1 KB	17 %
Lightweight	11.7 KB	5 KB	16.7 KB	52 %
with Opera Mini	9.3 KB	6.8 KB	16.1 KB	54 %

6.4 Adaptability

An important improvement was to provide better adaptability to preexisting customer configurations. The DoD Solution should not preclude customers who already have some telephony equipments. Therefore, all the settings necessary to place a call[4] have been

[1] Using the address book of the mobile phone.

[2] That is to say, the bottom arrow is always associated to the call history, etc.

[3] Opera Mini generic version for MIDP 2.0, with default settings, running on the Sony Ericsson WTK2 emulator emulating a Sony Ericsson K750. The emulator was running on the development machine 8.1.1

[4] This information includes Asterisk extensions, contexts, and channels.

made configurable from the administrative interface (rather than statically configured into the product). As Asterisk itself comes in many versions, a choice has been made for this thesis project to use an API called Asterisk-Java for all solutions. This API provides support for all Asterisk versions seamlessly. This API enables access to all of the possibilities offered by Asterisk, therefore any new function or request by a customer need not require a deep change in the DoD Server (unlike the situation before this thesis project began).

Adaptation can be realized at three places in the DoD project: in the JBoss server code itself, in the configuration database, and in routing logic (in both Asterisk and the GSM gateway). The GSM gateway logic is installation dependent and concerns mostly least cost routing tables. Table 6.2 shows various improvements and the level at which they were implemented.

Table 6.2: Improvements and where they were implemented

Improvement Level	JBoss Server	Configuration	Asterisk
Call control	X		X
External SIP server support		X	X
H.323 support	X	X	X
Statistics	X		X
Non-human user detection			X

6.4.1 Target platforms

The DoD solution aims at being very adaptable and offers support for a wide variety of clients. In our case, this means providing several Web interfaces for various browsers and devices and also providing software clients that exploit the potentials of each environment. For example, Symbian phones offer more extensive control of the phone than Java MIDlets. For example, Symbian applications can interact with ongoing calls, whereas Java MIDlets can only be executed during a call if the phone has sufficient resources - in most cases these MIDlets are simply paused until the necessary resource become available. Portability of the DoD solution to different mobile phones has improved with the introduction of the Opera Mini web browser [64] and the development of a lightweight Web interface for the DoD interface, cf. section 6.3.

6.5 Detecting non-human users

The detection of non-human users on the line[5] can be done by the use of DTMF input: for instance, requesting the user to type a single digit when picking up the phone, a request that an answering machine could not understand. This can be considered similar to the use of captchas found on Internet, as the goal is to detect whether there is a human or a computer on the line. In our case, there is no need to be tricky, thus the request can always be for the same digit, for all users. This solution might seem to put an unnecessary burden on the user, but live tests have shown that users get used to it after only a few phone calls[6]. Actually, if a Symbian client was developed, this DTMF process could be automated inside the client, making DoD calls more transparent to the user.

Another solution is the use of sound detection to decide if the user on the other end is human or not. Asterisk provides a command called Answering Machine Detection (AMD) [97] that can discriminate between cases based on the sound pattern of the first few seconds of a call. Here are the criterion used by their detector:

[5] Reminder: this needs only to be done for the initiator of a call and for participants of a conference.

[6] After explaining the actual process of a phone call with the DoD solution, users faced problems only by pressing the digit too late. There needs to be a time-limit for each call to be validated so as to continue using a channel for no reason.

- Duration of the initial silence before the greeting: if this value is exceeded then a machine is considered to be detected.[7]
- Duration of the greeting: if this value is exceeded, then a machine is considered to be detected.
- Duration of the silence after the greeting: if this value is exceeded then a human is considered to be detected.
- Maximum number of words is the maximum number of words in the greeting: if this value is exceeded then a machine is considered to be detected.[8]

This solution is of a statistical and empirical nature, as no two users, answering machines, or automated messages necessarily follow the same pattern. The best solution is to use AMD and in case of failure (that is to say the system believes it is interacting with a machine or was unable to take a decision), fall back to DTMF input (cf. figure 6.2). In our case, the initiator of a call expects a call and will not try to greet the DoD call, so it can be expected that a user talking at the beginning is an answering machine or another sort of automat. However, in the case of a conference, a callee is likely to start talking right after picking up the phone, to introduce him- or herself. The DTMF input request message is personalized so as to inform conference participant of what is happening and who requested their presence in the conference.

Note that the callee does not have to go through user detection in the case of a single call. If a voicemail system is reached, the caller can simply leave a message and hang up: there is no other user bothered by the audio message. The DTMF-fall-back behavior can also be seen as useful for callers in the case their phone is set up to screen calls through their answering machines: when the recorded message starts playing and requests for a DTMF input, the user can still utilize his or her phone and press the correct digit(s).

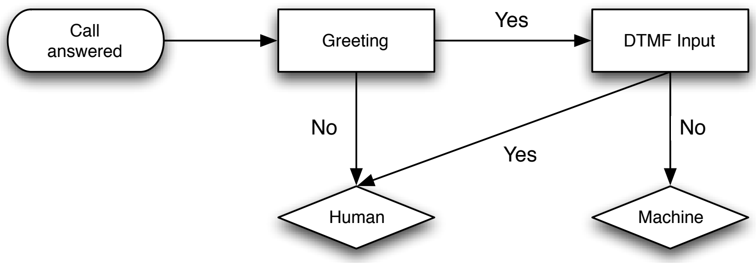


Figure 6.2: Proposed flow of operation for the non-human user detection

The system currently enforces the AMD solution mentioned above in section 6.5. It has been tested for over a month. Further tests are needed to be able to determine the most efficient settings (duration of the greetings for a human compared to a machine, time necessary to input a single digit or possibly a PIN code on the keyboard, etc.). Both solutions (AMD and DTMF input alone) currently seem to be viable. The AMD solution takes more time (up to five seconds to validate human presence), but causes less hassle to the users; whereas DTMF input is faster, but requires a manual action. Also the AMD solution is not suitable for a callee. For these users, a voice message followed by DTMF input would be preferred since the callees might not be familiar with the DoD process.

[7] In our case we modified this behavior so that if the initial silence duration is exceeded then a human is considered to be detected.

[8] Silence threshold and "definition" of a word (length, silence between two words, etc.) can be freely set by the administrator.

6.6 Fraud prevention

6.6.1 Callee fraud

Proper termination of operations must be guaranteed from both the DoD JBoss server and the Asterisk dialplan to avoid zombie calls from clogging the resources of the system. Proper termination of calls in the Asterisk dialplan is also a fraud-related issue; as if a callee was to stay connected to the Asterisk PBX after the caller has left the conversation, he or she could potentially exploit a loose dialplan to place calls using the company resources (cf. subsection 3.5.5). This problem can be solved for conferences by giving a special status to the initial caller. When this privileged (or "marked") user leaves the conference, this conference is automatically closed by Asterisk.

6.6.2 Unauthorized or fortuitous accesses and calls

The DoD JBoss server has a secure login interface (using Java Authentication and Authorization Service (JAAS) [58]) and time-limited Web sessions using HTTPS to prevent unauthorized persons from using the DoD solution. Further control can be enforced by limiting the duration, the origin, or the destination(s) of the call, or the time of the day during which certain calls are allowed. Hazardous user behavior such as going forward and backward inside the browser's history (and thus resending some HTML forms containing call requests) has been taken into consideration so as to avoid accidental calls.

Note that using HTTPS has a cost. For instance, there is a significant data overhead and in the case of the Opera Mini web browser and its traffic reduction mechanisms can no long be used. This might be of no importance when accessing the normal Web interface via a broadband connection (as the downloaded data traffic currently amounts to 110 KB and the uploaded data traffic to 14 KB for a typical session), but for the lightweight interface the traffic is greatly reduced hence the effects of overhead are proportionally increased, hence this effect has to be considered carefully as the downloaded data traffic will increase when using HTTPS by up to 23.6 KB compared to the initial 11.7 KB (8.9 KB and 5 KB in the case of the upload). If HTTPS connections are desired and the extra cost is deemed to be too great, then the solution is to use the Java MIDP client for the DoD solution, as it requires a download traffic of only 985 B and an upload data traffic of 1.1 KB per call.

6.7 Convergence

We mentioned in section 3.5.6, the different elements that the DoD solution should unify. Here is the state of the DoD solution regarding those elements at the end of this thesis project:

1. The DoD JBoss server offers a consistent Web interface on both desktop browsers and mobile phone browsers, cf. point 1 of section 3.5.6. Many different types of devices can interface with the DoD solution: there is a Java MIDP 2.0 client for recent mobile phones, a Symbian client is under development with an aim of better integration for more transparent use for the mobiles offering the Symbian OS, and MIDP 1.0 phones can access the lightweight interface through Opera Mini[9].
   As it is not possible to provide exactly the same look-and-feel in both the Web interface and via the Java MIDP 2.0 client, the choice has been made to compensate by introducing some of the cell phones' functions into the mobile client. The DoD solution encourages users to utilize a single client for all their mobile phone needs. More precisely, it was deemed cumbersome to switch between using the client to place DoD calls and the normal phone interface to place direct calls or send text messages. Therefore efforts have been made to integrate all these functions directly into the MIDP client.

[9] The legacy stand-alone Java MIDP 1.0 client has been dropped during this thesis project due to the lack of interest for such an obsolete platform.

2. Thanks to call-forwarding, one's colleagues only need enter a SIP URI to contact of this user, no matter where he or she is, cf. point 2 of section 3.5.6[10].

3. As chapter 7 will show, the DoD Server can communicate with other SIP servers and other PBXs inside the company, but also potentially with outside SIP servers and PBXs, cf. point 4 of section 3.5.6.

4. The point 6 of section 3.5.6 is not yet a reality; even though the modularity of the DoD solution is an important step in the direction of a solution to suit all companies' needs. Yet, this would be achieved through different modules to interface with various external equipments, whether they are data equipments such as databases or telephony equipments such as proprietary PBXs. As no decision has been made concerning the external equipments that will be supported, no work has been done concerning voice mails yet (see point 3 of section 3.5.6).

[10]This can also be achieved using the FollowMe functionality of Asterisk or a judicious AGI script making use of the internal directory. In such cases, the solution works even if the company's telephony system does not use SIP.

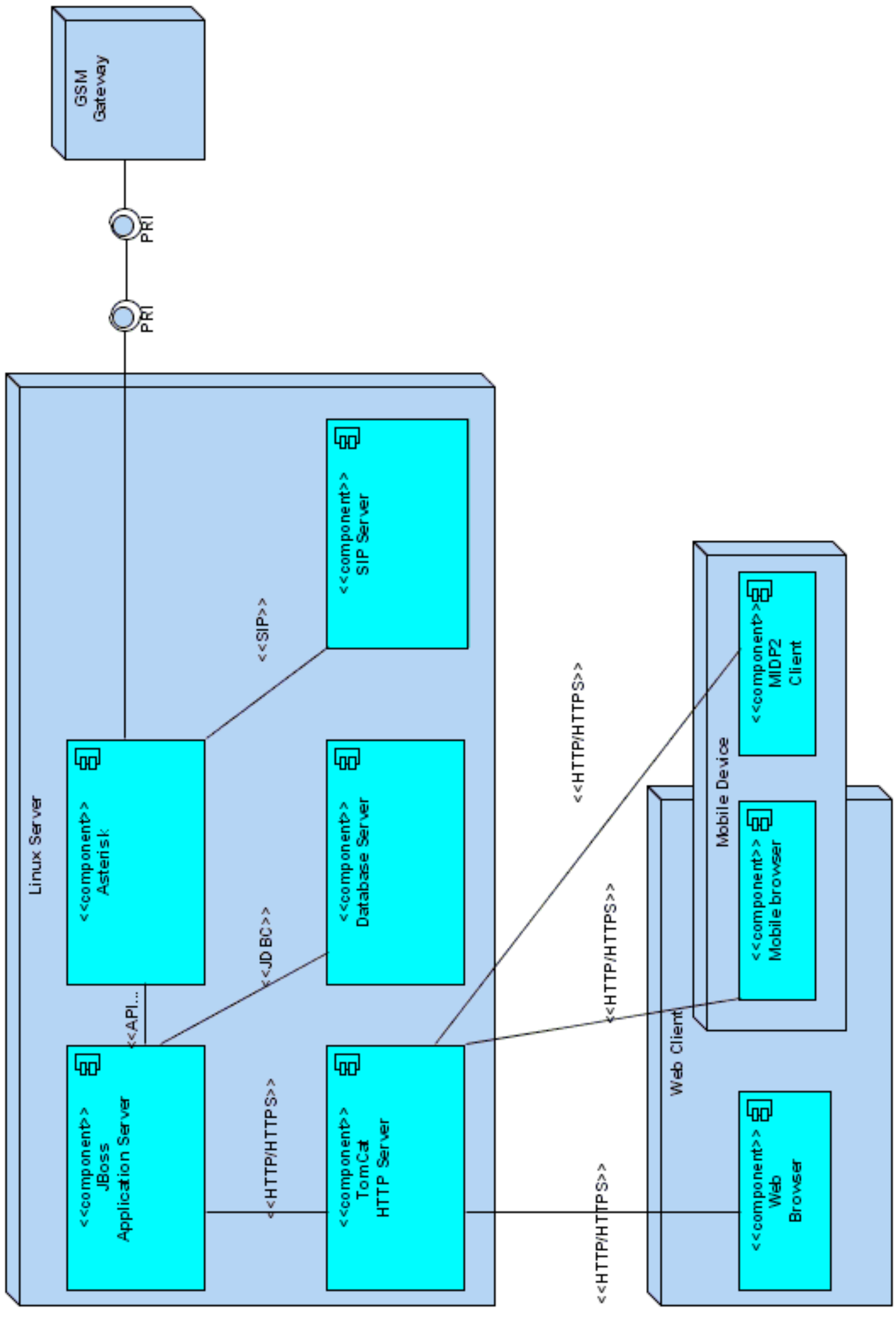


Figure 6.3: Deployment schema for an installation of the DoD solution

Chapter 7

Addition of the SIP Express Router to the DoD Server

As mentioned earlier, SER is a SIP router often used in conjunction with Asterisk as Asterisk lacks the functions of a real SIP proxy. The idea of adding a SER server to an Asterisk PBX is quite widespread. It is addressed in documents [98] and [20], and also used in the Carrier Class solution [18]. We have decided to use SER both as registrar and proxy for our SIP clients. In this section, we use SER as a short-hand that encompasses both SER and OpenSER. The later is a spin-off of the original SER project.

7.1 Principle

The virtual organization of our network is shown in figure 7.1. This figure is a variant of figure 3.2. In the case depicted in figure 7.1, the DoD JBoss server and the Asterisk server are no longer on the same machine, and the SER server is on a third machine. However, it is possible for the three of them to be run on the same machine. In fact, this configuration has been tested successfully during this thesis project. While this will put a heavy burden on a single machine and defeat, to some extent, the original intent of adding the SER server which was to offer better scalability of the DoD solution, the current design choice for the DoD solution only adds a single machine to the company's network.

Numerous accesses to databases are done for various purposes. Asterisk currently accesses a database for the billing, SER accesses a database for the authentication of the SIP clients, and the DoD JBoss server accesses a database for the authentication of DoD users, managers, and administrators. Figure 7.1 also shows that the DoD JBoss server communicates only with the Asterisk PBX; which in turn provides the link between the DoD JBoss server and the SER server. Therefore it is a necessity for the Asterisk PBX to communicate with (i.e. place and receive calls to and from) the SER server. This is done by registering the Asterisk PBX as a SIP client to the SER server and adding the Asterisk PBX to SER's list of trusted hosts.

Figure 7.2 illustrates the new operation of the DoD solution when a user of a mobile phone wishes to call a SIP client. The main conclusion that is to be drawn from this figure is that the Asterisk PBX is on the path of the call between the mobile phone and the SIP client. Unfortunately, this is inevitable as SER needs access to the PSTN and the PLMN. However, Asterisk is now freed of all the pure SIP calls (such as internal calls), which leads to better performances for these DoD calls.

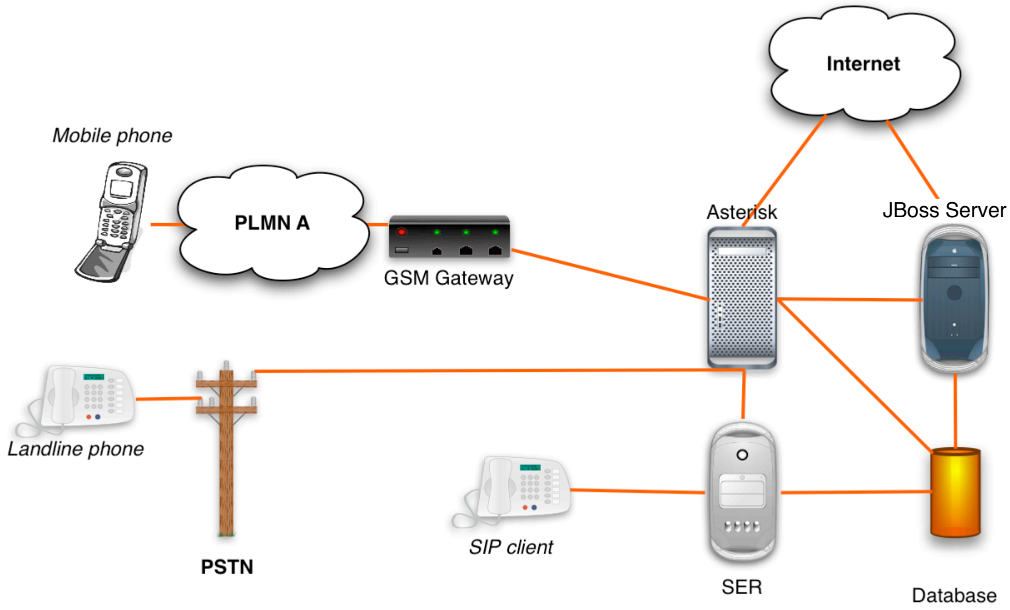


Figure 7.1: Connections inside the DoD network with a SER server

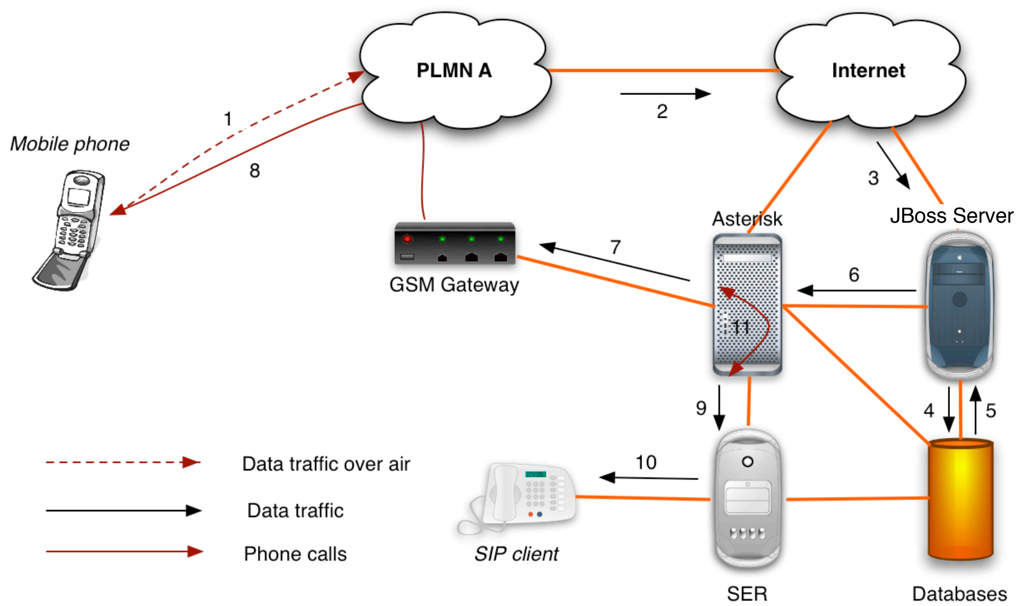


Figure 7.2: Operation of the DoD solution with a SER server

7.2 Additional functionality

The addition of a SER server is justified, among others, by the necessity of presence support in the DoD solution. Note that initially the only SIP clients to be accessible to the SER server, and therefore to the DoD solution, are **local** SIP clients. The SER server, just like Asterisk, could also access external SIP domains. In this case, connections to external SIP servers should be done directly via the SER server, rather than via the Asterisk PBX. This facilitates inter-domain SIP calls without penalizing DoD calls. Furthermore, SER, with the help of MediaProxy or RTProxy, is able to circumvent most NAT issues, likely to happen when connecting with external SIP clients (cf. subsection 7.2.2).

7.2.1 Presence

In SER, the presence of the peers is registered using the XML Configuration Access Protocol (XCAP). Contact lists are obtained by a peer using the Resource List Server (RLS). Therefore presence subscriptions requires the collaboration of XCAP and RLS. Both XCAP and RLS are described below, more informations is available in SER's handbook [39]. All file examples below come from this handbook. Attempts have been made to set up an XCAP server inside the DoD Server, but it was not deemed to be a priority. However, we present here the general mechanism of the presence server due to its importance for future work.

XML Configuration Access Protocol

XML Configuration Access Protocol (XCAP) is a protocol allowing to query data stored as XML files. These queries can read, modify, or delete parts of the data over an HTTP connection. The original intent of XCAP was to store contact lists, presence information, and authorization policies. For more details, the reader is referred to the presentation of Jonathan Rosenberg covering XCAP from the XML basics to the XCAP query syntax [72].

It is likely that some of the XCAP exchanges will take place over a wireless interface (over WiFi, GSM, or DECT connections for instance), hence the use of XPath queries in XCAP (we provide a quick reminder on XPath queries below). That way only the relevant part of the data is queried and eventually sent back over the air interface. A few examples of presence files are shown in figures 7.3 and 7.4.

The XPath query language

Taking the example of the file in document 7.3, we can show several XPath queries and their results. For instance `rls-services/service/@uri` would return `sip:smith-list@iptel.org` and `sip:cz@iptel.org`. An XPath query can be read similarly to a Unix path, except for the strings beginning with `@` which represent an attribute and not an element. Other example, `rls-services/service[@uri="sip:cz@iptel.org"]/list/entry[@uri= "sip:abc@iptel.org"]/display-name` would return `A B`. This shows how to evaluate conditions on the attributes of the elements in a XPath query. We will finish our brief overview on the possibilities of the XPath query language in showing how to test the value of an element and how to browse the XML tree upwards. `//entry[display-name="A B"]/../@name` would return `czech iptel`.

A concrete example in our case would be to retrieve all the URIs watched by the client with the URI "sip:cz@iptel.org". This can be done with the following XPath query: `//service[@uri="sip:cz@iptel.org"]//entry/@uri`. This would also work in the case there are multiple `list` elements associated to the URI "sip:cz@iptel.org".

```
<?xml version="1.0" encoding="UTF-8"?>
<rls-services>
	<service uri="sip:smith-list@iptel.org">
		<resource-list>http://localhost/xcap-root/resource-lists/users/smith/
resource-list.xml/~~/resource-lists/list[@name=\%22default\%22]
		</resource-list>
		<packages>
			<package>presence</package>
		</packages>
	</service>
	<service uri="sip:cz@iptel.org">
		<list name="czech iptel">
			<entry uri="sip:abc@iptel.org">
				<display-name>A B</display-name>
			</entry>
			<entry uri="sip:cde@iptel.org">
				<display-name>C D</display-name>
			</entry>
			<entry uri="sip:efg@iptel.org">
				<display-name>Ef Ge</display-name>
			</entry>
		</list>
		<packages>
			<package>presence</package>
			<package>email</package>
		</packages>
	</service>
</rls-services>
```

Figure 7.3: Example of RLS services document [39]

```
<?xml version="1.0" ?>
<resource-lists>
	<list name="default">
		<list name="work">
			<entry uri="sip:someone@iptel.org">
				<display-name>Someone</display-name>
			</entry>
			<entry uri="sip:smith@iptel.org">
				<display-name>Jonathan Smith</display-name>
			</entry>
		</list>
		<entry uri="sip:vasek@iptel.org">
			<display-name>Vasek</display-name>
		</entry>
		<entry uri="sip:vaclav.kubart@iptel.org">
			<display-name>Vaclav Kubart</display-name>
		</entry>
	</list>
</resource-lists>
```

Figure 7.4: Example of resource lists document [39]

Privacy

Privacy in XCAP can be enforced by rule files in the XML format containing a black list of users that will not receive any presence information and/or a white list of users that are entitled to access such information. Discrimination can be done both at the AoR level and at the domain level. Document 7.5 provides an example of an XML document defining authorizations.

```
<?xml version="1.0" ?>
<ruleset xmlns="urn:ietf:params:xml:ns:common-policy"
                xmlns:pr="urn:ietf:params:xml:ns:pres-rules">
        <rule id="blacklist">
                <conditions>
                        <identity>
                                <id>sip:nemo@somewhere.net</id>
                        </identity>
                </conditions>
                <actions>
                        <pr:sub-handling>block</pr:sub-handling>
                </actions>
                <transformations/>
        </rule>

        <rule id="whitelist">
                <conditions>
                        <identity>
                                <domain domain="iptel.org"/>
                        </identity>
                </conditions>
                <actions>
                        <pr:sub-handling>allow</pr:sub-handling>
                </actions>
                <transformations/>
        </rule>
</ruleset>
```

Figure 7.5: Example of presence authorization document [39]

Resource List Server

The Resource List Server (RLS) is also based on the use of XML files for storing informations [73]. The RLS serves the generic purpose of storing lists, possibly nested, of any kind of elements - hence they are suitable for the storage of contact lists. A RLS file contains `service` nodes with `uri` attributes containing:

- `list` elements containing `entry` elements for each contact. These entry elements can store the URI as an attribute and the name of the contact as a value.
- `resource-list` elements containing references to external contact lists in a similar format.

Document 7.3 provides an example of RLS services document. Document 7.4 shows an example of resource list document. The `package` elements define the events for which subscriptions are allowed for this particular service. If no package is specified then subscriptions for all events are allowed [73]. In the figure 7.3, two packages are used: `presence` and `email`. Other examples of packages are `refer`, `key press stimulus`, `message-summary` etc. [77].

Comparison with other solutions

Asterisk Support of presence in Asterisk is more rudimentary than that provided by SER with XCAP. It requires the editing of the dialplan for each SIP extension that wishes to advertise itself. This is really cumbersome when a customer has numerous extensions or when the extensions' names or number change frequently.

Internal solution We integrated presence support in the DoD databases as mentioned in section 6.1. It enables users to change their presence information (or check on their colleagues) via the mobile client or the Web interface. However, they cannot access the presence information entered in their colleagues' SIP (soft) phones. Indeed, such a solution leads to a duality in the presence information, as there are two completely independent repositories for presence informations. Therefore, it seems preferable to consider this as a draft implementation of what presence support should be in a future DoD solution. The current integration provides a good framework for the DoD front-end, but the back-end should eventually integrate data from a future XCAP server. A database could still be used for data that the XCAP server cannot handle, but all the informations about presence should be kept consistent. The substitution of XCAP-provided data for the data currently provided by a database look-up is expected to be easy due to the modular design of the DoD JBoss application.

7.2.2 Network Address Translation

SER already has built-in NAT transversal for SIP signaling [26], but the audio (and potentially video) streams still need to be addressed. SER allows better handling of NAT-transversal via MediaProxy [8]. NAT issues had already been commented on by Ning Zhou in [109]; where she pointed out that if both SIP clients and proxies have not been configured properly, the communication might only have one-way audio.

MediaProxy acts as a RTP proxy and solves NAT issues by using the Traversal Using Relay NAT protocol (TURN) [74]. Other protocols are available for NAT transversal, such as Simple Traversal of UDP Through NAT (STUN). The difference between STUN and TURN is that TURN needs to use a relay located at a public IP address, whereas STUN uses addresses that might not always be usable by every single peer. Therefore TURN provides a one-size-fits-all solution – at the cost of a public server needing to be in the media path.

MediaProxy, like SER, provides load-balancing and geographical distributions capabilities. These possibilities have not been used during this thesis project, but they support the general goal of designing the DoD Server to be as scalable and adaptable as needed.

7.3 Conclusion

7.3.1 General proof of connectivity

Beyond the integration of SER into the DoD Server is the general possibility to connect the DoD Server, through the Asterisk PBX, to other communication servers, such as H.323 gatekeepers, whose role is similar to that of a SIP proxy, or other proprietary PBXs. For instance, during laboratory experiments, we connected the corporate innovaphone IP800 PBX [36] to the DoD Server by registering the Asterisk PBX as a gateway to the innovaphone PBX; thus benifiting from the PRI interfaces of the IP800 and its integrated H.323 gatekeeper (see figure 7.6).

Later in this thesis project, we integrated to the DoD solution with a SIP trunk and, even later, with a VoiceBlue Lite gateway from 2N [4], offering VoIP and GSM connectivity in a single box. This gateway replaced the Blue Tower gateway mentioned previously, but could also be used in addition to a Blue Tower gateway. The advantage of the Voice Blue Lite gateway was that it connects to Asterisk as a regular SIP client via an ethernet connection to the network, therefore removing the need to have a PRI card in the DoD Server. Indeed such cards are rather expensive[1] and can be tricky to configure: there could be problems with the handling of the Interrupt ReQuests (IRQs), the obscure text-based configuration requires a deep knowledge of telephony networks, etc.

As a result of these attempts, it appeared that even if the integration of SER into the DoD Server might not be a primary concern for smaller companies, the possibility to integrate the DoD Server with the existing PBX of a company (or the other way round) and with various call-termination means (e.g. GSM gateway via SIP or via an ISDN PRI connection, external SIP trunk, etc.) are definitely an asset for the DoD solution.

7.3.2 Future work

This thesis project integrated SER with the DoD Solution. This integration was not straightforward, because of the changes necessary in the JBoss server, in Asterisk's configuration, and also in the flow of the calls. We provided in this section an overview on SER's presence solution with the hope it will prove handy for the future DoD developers. We also would like to repeat here our advice on how to best integrate presence into the DoD solution: first integrate presence for XCAP presence into SER and the SIP clients; then integrate the XML resources into the JBoss interfaces.

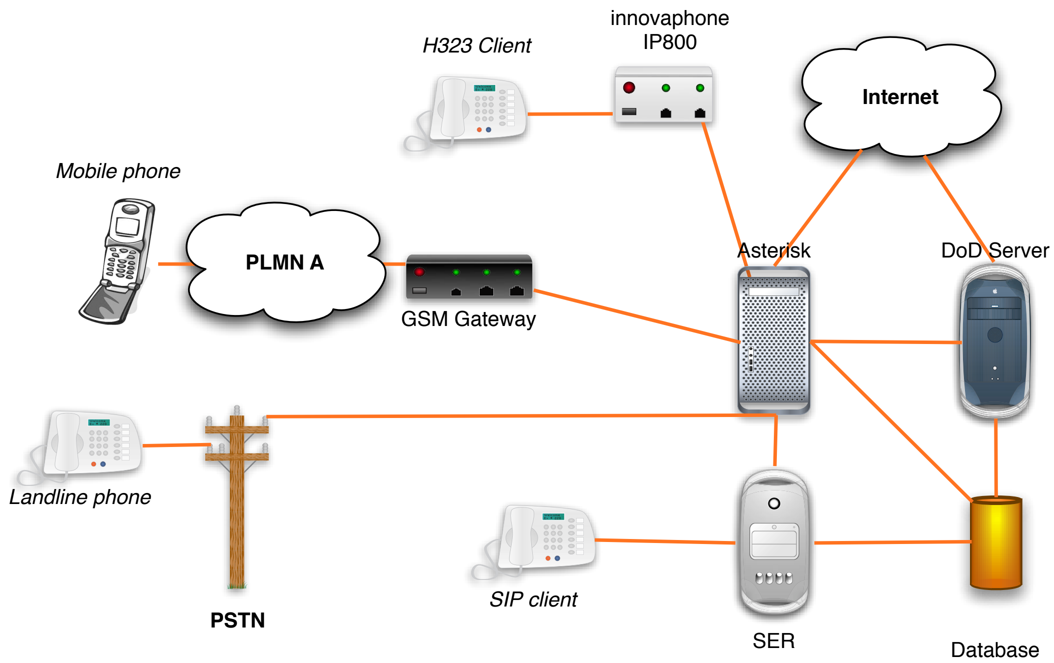


Figure 7.6: Connections inside the DoD network with an innovaphone IP800

[1] US$ 470 for a TE110P such as used during this thesis project [23], but US$ 500 is a good price estimation for a new, low-end Digium PRI card.

Chapter 8

Tests

Before releasing the DoD solution into a commercial environment, testing needs to be done. For a communication platform, many criteria are significant. Some are quantitative: delay to establish a call, maximum simultaneous calls, etc., while others are qualitative: audio quality of a call, ease of use of the solution, *impression* given to the user of the delay to establish a call, etc. We ran tests in our laboratory to measure the former and we also deployed the DoD solution for internal use to get feedback on all qualitative factors.

Additionally, other aspects are important in a commercial VoIP solution, such as security; however, these were not tested or explicitly evaluated. Many of these issues are specific to each site where the DoD solution would be deployed. The DoD solution has been designed with security in mind (e.g. use of HTTPS, JAAS for user authentication, etc.), but security flaws in third-party software or improper configuration could lead to security problems. Hence as part of this thesis project, a guide of best practices was developed to assure a reasonable level of security for the DoD solution.

8.1 Test environment

8.1.1 Test machines

The development and the testing of the DoD solution utilized the following equipment:

- development platforms: two Intel Pentium 4-powered desktop computers (CPU: 2.80 GHz, RAM: 1 GB, Ethernet 100 Mbps) running Microsoft Windows XP Professional Service Pack 2, one of them installed with Eclipse 3.1.2 with JBoss-IDE 1.6.0.GA and JBoss 4.0.1sp1.
- test platforms:
    1. a low-end laptop (CPU: 750MHz Mobile Pentium III, 128 MB RAM, Wireless 802.11b 11 Mbps) running Microsoft Windows 2000
    2. a Intel Celeron-powered desktop computer (CPU: 2.53 GHz, RAM: 512 MB, Ethernet 100 Mbps) running Microsoft Windows Server 2003
    3. a Intel Celeron-powered desktop computer (CPU: 2.53 GHz, RAM: 256 MB, Ethernet 100 Mbps) running Microsoft Windows XP Professional Service Pack 2
- Asterisk platform and stable JBoss Application Server : a Intel Celeron-powered desktop computer (CPU: 2.53 GHz, RAM: 1 GB, Ethernet 100 Mbps) running Fedora Core 7, JBoss Application Server 4.0.1sp1, OpenSER 1.2.2, and Asterisk SVN-branch-1.2-r82334 with an ISDN PRI interface (Digium Wildcard TE110P T1/E1) with the Zaptel drivers version 1.2.

- GSM gateway: a 2N Telekomunicace [86] Blue Tower [3] with one GSM board, one ISDN PRI card and an auxiliary card (offering a serial port and a RJ11 plug) running firmware version 2.20.49.
- mobile phone clients:
  - Motorola E1000
  - Sony Ericsson k700i
  - Sony Ericsson P1
  - Sony Ericsson Z1010
- SIP hardware clients:
  - a Hitachi WirelessIP 5000E [34].
  - an innovaphone IP110 [35].
- H.323 hardware client: an innovaphone IP110.
- SIP software clients:
  - X-lite 3.0 [22], 3CX VOIP Phone 1.17 [5], and SJphone 1.65 [79] running on the development platforms.
- DoD Interface clients:
  - Internet Explorer 7.0 and Mozilla Firefox 2.0.0.7 on the development platform.
  - Safari 3.0.4 on an Intel Core 2 Duo-powered laptop computer (CPU: 2.2 GHz, RAM: 2 GB) running Mac OS X 10.5.1.

Figure 8.1 represents the network used during most tests and summarizes the software equipments of the various machines.

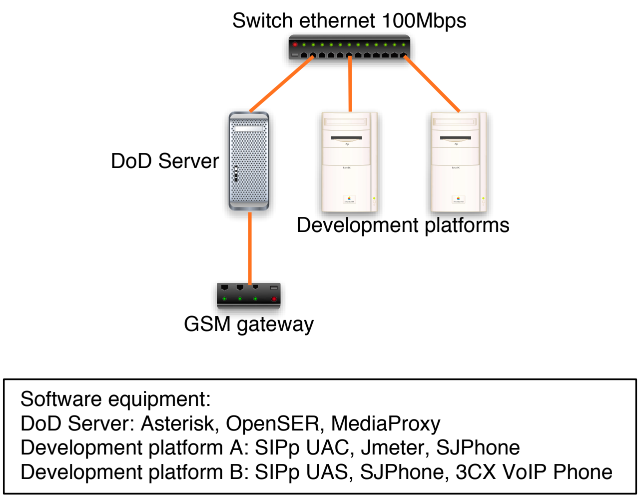


Figure 8.1: Configuration of the test network and test machines

8.1.2 Test tools

Different test tools were necessary to test the different components of our project:

- Apache JMeter[10] version 2.3 has been deployed on the development platform to test the JBoss Server (by monitoring HTTP and HTTPS requests, among others).
- SIPp [78] version 2.0.1 has also been deployed on the development platform to perform load testing of Asterisk, (Open)SER, and MediaProxy with both SIP and RTP traffic.
- Wireshark [108] and tcpdump [88] to capture and analyze network traffic.
- home-made shell scripts to analyze the logs of the various softwares tested.

8.1.3 Common aspects to the tests

Comments made in this section concern some of the factors influencing the performance of a VoIP solution. Unless otherwise specified, all assumptions stated in this section apply to all the tests run in our laboratory.

CODECs

One of the most demanding tasks which the Asterisk platform must execute is the transcoding (i.e., converting information encoded with one CODEC to the format for another CODEC). Voip-Info.org's web page concerning Asterisk dimensioning [99] gives a number of examples of the load which a simple Asterisk PBX without SER or the DoD JBoss server can support. One of these is that a 1 GHz laptop computer could handle 250 concurrent calls with re-invited RTP without any CODEC transcoding (resulting in an approximate load of the CPU of 9 %) **but** could only support 50 simultaneous calls with audio transcoding between two flavors of G.711: u-Law (or Micro-Law) and A-Law[1] ,(resulting in a load of 99 %) even though this example represents a transcoding that is not particularily resource-consuming.

The scenario depicted above in figure 8.2 involves calls placed between two SIP clients running on desk computers. In a normal situation, each SIP client usually has multiple CODECs installed[2]. During the SDP messages exchanges, the SIP clients agree on which CODEC to use for the audio transmission. Therefore there will be no need for Asterisk to transcode the streams. It would be the same if the two legs of the call were going through the GSM gateway. The GSM gateways from 2N Telekomunicace can transcode between the GSM CODEC and A-Law and u-Law for instance, and other PBXs such as innovaphone's products can handle this process as well. As a result, it is very unlikely that Asterisk will have any transcoding to do since the SIP clients and the GSM gateway have a non-empty set of common CODECs.

Echo-cancellation

Another important contributor to the load of the Asterisk host is echo cancellation. For example, a study [70] showed that one machine could handle 150 calls with a 75 % load on the CPU *without* echo cancellation, whereas approximately 60 calls *with* echo cancellation resulted in the same CPU load. During our tests, the software echo cancellation was enabled.

[1] These are sometimes denoted as G.711u and G.711a or PCMU and PCMA.

[2] For instance, the SJPhone version used is offering support for 5 CODECs out of the box: GSM, iLBC 20 ms, iLBC 30 ms, G.711 A-Law, and G.711 u-Law. Even a dedicated wireless handphone, such as Hitachi's WirelessIP 5000, supports several CODECs (G.729a, G.711 A-Law, and G.711 u-Law).

Audio quality An important criterion for VoIP solutions is the audio quality of a call. According to Intel [37], "even a 1 % loss can significantly degrade the user experience with the ITU-T G.711 voice coder (vocoder), which is considered the standard for toll quality. Other coders degrade even more severely because they compress the data more rigorously." Some other studies [67] [21] place that limit at 2 %. However, between 1 % and 2 %, it is usually reckoned that the user is aware of the quality degradation. UDP packet loss will be our primary criterion to check for audio quality in these tests.

8.2 Laboratory tests

8.2.1 JBoss Application Server with calls

Design

Using Apache Jmeter, we created a series of scenarios mimicking a typical use of the Web interface by a user (cf. figure 8.2). In this example, the call will be placed between two SIP soft clients set to automatically answer calls. Two types of tests can be derived from this:

- A load test: running as many instances of this scenario as possible in parallel. This is achieved by executing multiple threads from Jmeter, each performing similar actions.
- A stability test: running a reasonable number of instances of this scenario in parallel, numerous times in a row. This is achieved by executing several threads from Jmeter, each performing similar actions and placing these simulated user actions in a loop.

Obviously, these tests also test the Asterisk server and for SER – as they actually involve calls placed between SIP clients. The DoD solution is also composed of a mobile Java client and a lightweight Web interface, but we will not test them, as the servlets and web pages used by these two interfaces are a subset or a reduced version of the ones used to carry out the Jmeter test. Significant values to be measured in these tests are:

- Percentage of successful logins, calls, updates, hang-ups, and logouts.
- Delays between a request and the reply: for instance, the delay between placing a call and the caller's phone ringing. This is the relevant delay for the caller. Once the caller picks up his or her phone, he or she hears a ringing tone, even though the other line is probably not yet connected. The additional delay induced by the DoD solution (waiting for the GSM gateway to place the second call, for instance) is then masked by this. As the extra delay is now perceived by the caller as the callee picking up the phone after a slightly longer time period than the callee actually does, we must make sure this extra delay is not too long and therefore not perceived as a reluctancy of the callee to answer the phone or as meaning that the callee is currently away from his or her phone (see section 8.4.1).

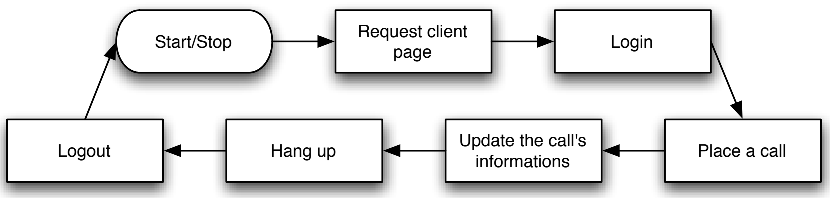


Figure 8.2: Typical use of the Web interface

We consider a user, typing at a reasonable speed and fairly familiar with the DoD solution. This user places from his or her office a dozen calls per day, looking up the appropriate phone numbers in an external database. A typical DoD session for this user looks as follows:

1. A user requests the client page and is automatically redirected to the login page, as JBoss's implementation of the JAAS requires.

2. After the login request is sent and approved, the user is automatically redirected to the client page. We model the time for the user to type his or her login information and validate it with a uniform random timer set to go off between 20 to 40 seconds after the reception of the login page.

3. Next, a call is placed. We model the delay to type or retrieve the appropriate phone numbers with a uniform random timer set to go off between 20 to 50 seconds after the reception of the client page.

4. Next, the user requests an update concerning the status of the call[3]. We model the delay between the call request and the update request with a uniform random timer set to go off after 5 to 15 seconds.

5. After a certain delay, the call is terminated. We chose to model this delay with a lognormal (cf. equation 8.1) random timer (as suggested in [29]) with a σ of 1 and a μ of 4. The lognormal-distributed random variable generator is based on the contributions at JScience [50]. The time unit for this delay is of 20 seconds.

6. Finally, the user logs out. We model the delay between the user hanging up and the user logging out with a uniform random timer set to go off between 5 to 15 seconds.

$$f(x;\mu,\sigma) = \frac{e^{-\frac{(\ln x-\mu)^2}{2\sigma^2}}}{x\sigma\sqrt{2\pi}} \tag{8.1}$$

We chose to model the delay between two sessions after the usual model of delay between two phone calls: that is to say a Poisson's law[4]. We used a Poisson-distributed random variable generator (cf. algorithm 1) based on Donald Knuth's algorithm [53]. In our case, λ has a value of 20. The time unit for this delay is 10 seconds.

Algorithm 1 Poisson-distributed random variable generator

$L \leftarrow e^{-\lambda}, k \leftarrow 0, p \leftarrow 1.$

while $p \geqslant L$ **do**
  $k \leftarrow k + 1.$
  Generate uniform random number u.
  $p \leftarrow p * u.$
end while
return $k - 1.$

Many comments can be made about the above scenario. Hanging up via the DoD interface is not a requirement, we only test it here for the sake of completion; all calls can be hung up directly from the phone of the user, whether it is a soft phone, a desk phone, or a mobile phone. Logging out from the DoD interface after each call is not a requirement either. However, it is a good practice for the users to log out if they are not planning to place another call for a long time, especially if using the Web interface in a public place. This test has been designed to test

[3] This models a user who is rather anxious that the call was not actually placed.

[4] A Poisson model is valid in the case of large systems with numerous callers but it was used here nonetheless. This model also does not apply in the case where the user's main task is to place phone calls, since in such a case there should be only a small constant delay between two consecutive calls.

the capacities of the J2EE application and of the JBoss Application Server, hence the use of the Web interface even for simple actions.

Results

While the tests described in the following sections will consider traffic-related issues, this first test will simply aim at proving that the system is functioning properly. For this purpose, we will set up JMeter to simulate two users enacting the above scenario and placing 100 calls each. We will analyze the differences between the expected scenario and the events that actually occured. Jmeter was run on one of the development platforms (cf. section 8.1.1) and the SIP clients were executed on both development platforms (two copies of SJPhone and one copy of 3XC VoIP Phone). OpenSER with MediaProxy was used to handle the SIP calls and the associated RTP streams.

Figure 8.3 presents the CPU load chronologically at moments when the system was executing precisely two active ongoing calls. The values were taken at random times and ordered chronologically (clockwise, starting at the top of the graph). The first conclusion from this graph is that the CPU load does not significantly increase over time. Obviously, it still depends on the number of ongoing calls. Other measures showed that the average CPU load when no calls (and no connected user to the JBoss server) were active was of 0.3 %; when one call was active (and one or two connected users), the load was of 3.82 %; finally when two calls (and therefore two connected users) were active, the CPU load was of 6.30 %. The load generated by one user being connected to the JBoss server is roughly of 2 % (even if two users were connected).

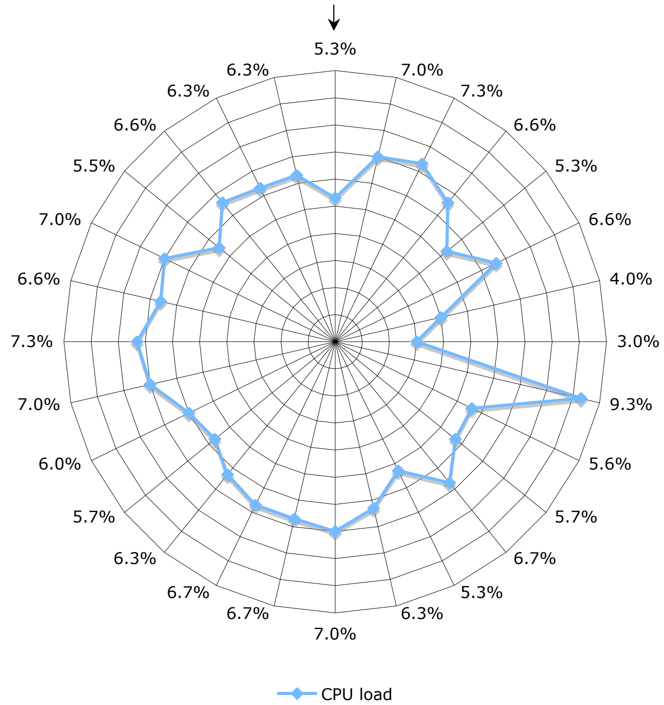


Figure 8.3: Evolution of the CPU load over the time, with 2 ongoing calls, each ring representing 1 % of CPU load

Comparing with OpenSER and MediaProxy alone

Running two simultaneous calls with SIPp using OpenSER and MediaProxy only resulted in an average CPU load of 4.5 %. The JBoss Application Server adds only around 2 % of additional load, which seems reasonable.

Table 8.1 various informations related to the JMeter test client interacting with the JBoss server. We can note that the longest delays occur when requesting an interaction with Asterisk, that is to say when the status of the lines is updated or when one of the line is hanged up via the Web interface (note that calling does not require an *inter*action). We can also see that during this test, no error were encountered.

Table 8.1: Results for each operation

Action	Average delay	90 % delay	Error rate
Connecting	642 ms	1375 ms	0 %
Authentication	61 ms	47 ms	0 %
Calling	472 ms	625 ms	0 %
Updating	1,122 ms	1,125 ms	0 %
Hanging up	1,693 ms	2,156 ms	0 %
Logging out	39 ms	16 ms	0 %
Total	675 ms	2,110 ms	0 %

No test for audio quality was run, as this particular test's goal was not to focus on the telephony part of the DoD System. However, out of curiosity, an operator manually tested[5] the sound quality of the calls several times during the test and found it to be perfectly acceptable.

8.2.2 JBoss Application Server only

The DoD JBoss server can be used by a client with only a few requests per session. A typical use of the DoD JBoss server has been monitored on the development machine (cf. section 8.1.1) using ManageEngine by AdventNet Inc.[7] and the results show that the memory usage of the JVM increases by approximately 3 MB[6] at the beginning of a new call placed from the interface, but returns to normal during the call, provided that no update of the call status is requested by the user.

Load tests were applied to the JBoss Application Server. During these tests, the system was otherwise idlling (that is to say, Asterisk, OpenSER, etc. were running but not receiving any calls). We found that the JBoss Application Server could handle 200 connections to the main page (named "Connecting" in table 8.1) in 35 seconds (5.6 hit per second), resulting in an average CPU load of 5 % (with peaks at approximately 20 % at the highest).

8.2.3 SER tests

Design

SER was tested with the help of SIPp to generate SIP signaling traffic along with pre-recorded audio traffic. This test replays exchanges recorded in the pcap format using Wireshark, tcpdump, or any of the numerous tools available for network packet capture. The default test scenarios

[5] using the PC's microphone and speakers

[6] This figure includes the interface pages loaded and the various modules (configuration, user list, connection with Asterisk, etc.) required to place a call.

for SIPp play one-way audio, hence these will be used in our tests, unless otherwise specified. Since there is no precise call pattern, the following tests are more a proof that the system can (or cannot) handle various conditions rather than a test of the limits of the DoD Server's performances.

Initially we present some basic tests concerning only the SIP stack. The tests were the basic SIPp tests (UAS and UAC)[7] on two separate machines on the same network. A third machine was running OpenSER. Each call is composed of six messages in total (cf. figure 8.4). The pause corresponds to the period when RTP streams are played, if the test includes any RTP traffic. Two batches of tests have been run: one generating 100 calls per second, another generating 500 calls per second. We decided to focus on the CPU load and on the number of failed calls (if any occurred). The host running OpenSER was a Intel Celeron-powered desktop computer (CPU: 2.53 GHz, RAM: 1 GB) running Fedora Core 7, and OpenSER 1.2.2.

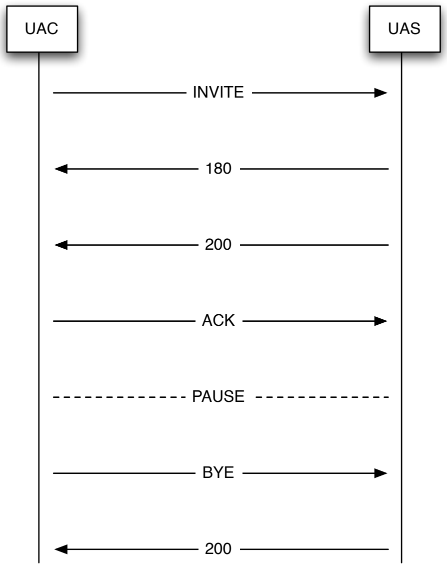


Figure 8.4: SIPp UAC/UAS test

We ran tests with two different audio streams. The first one is 9 second long, the other one was 400 second long. They are both encoded using G.711 A-law and are composed of, respectively, 236 RTP packets (with a frame size of 30 ms) and 20,024 RTP packets (with a frame size of 20 ms). We will refer to the first one as audio stream A, the second one as audio stream B. Audio stream A is the audio stream provided in standard distributions of SIPp.

Results

SIP messages

We can see in figure 8.5 that there is no problem loading this SER server with 100 calls per second[8]. However, with a load of 500 calls per second, we noticed that problems arise. To define more precisely the limiting number of calls per second that our SER server can handle, we ran a progressive test, increasing the number of calls per second by 50 calls per second, every 10 seconds, starting at 50 calls per second. The results are shown in figure 8.6. The SIP messages leading to dropped calls in this case is the first "200" message sent by the UAS (see figure 8.4), which is compulsory to establish a dialog [71]. Some of the "100" (Trying) and "180" (Ringing)

[7] These tests do not involve authentication of the SIP clients.

[8] The test was executed during 500 seconds, which explains the abrupt drop in the graph at 500,000 calls.

messages sent by the UAS prior to the establishment of the dialog are not compulsory and their loss does not result in a dropped call, as long as the "200" (OK) message is present. However, the presence of the "100" and/or of the "180" messages is not enough to establish a dialog.

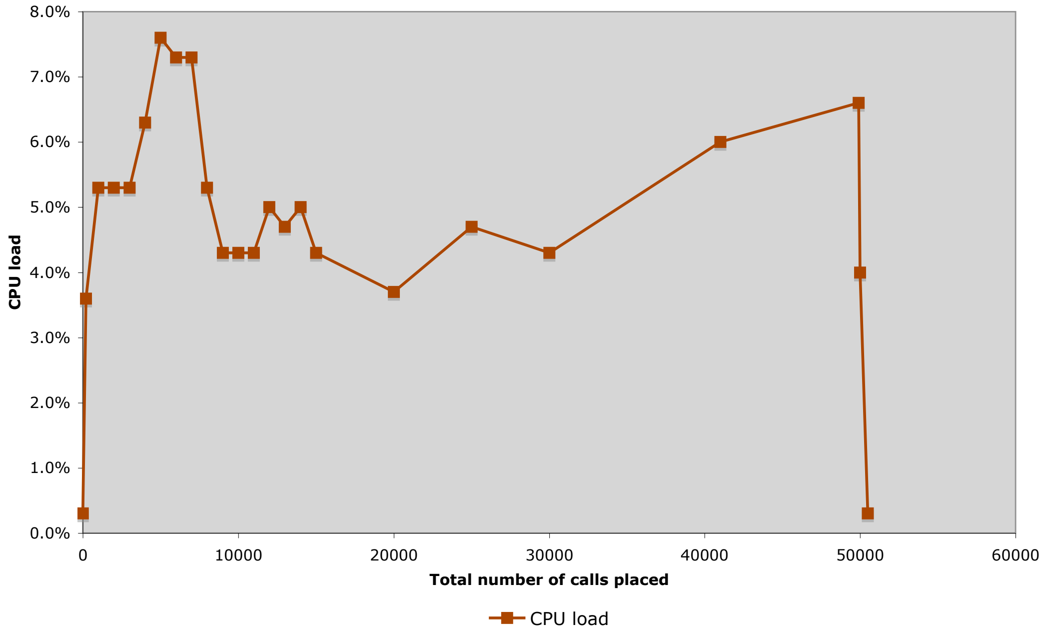


Figure 8.5: CPU load at 100 calls per second

To try to corroborate the results of others that prompted us to use SER as our SIP proxy, we ran the same tests on the same machine using Asterisk as the SIP proxy. We can see that Asterisk starts mishandling calls at around 300 calls per second, while SER only starts having problems at 450 calls per second. Concerning the CPU load[9] generated by SER, we noticed that, when the testing starts, the load always peaks before stabilizing at a value proportional to the number of ongoing calls. However, there seemed to be no direct link between CPU load and the number of failed calls.

One-way audio calls with OpenSER with MediaProxy

In this test, the DoD Server is running MediaProxy 1.9.0.1 and OpenSER is using this proxy for audio streams. We still use the SIPp UAS-UAC test, but this time the UAC is playing a B call to the UAS[10]. The results of this test are displayed in figure 8.7. In this test, we initiated 260 calls at the rate of one per second between two hosts, MediaProxy being in charge of the RTP streams. These 260 calls correspond to only 130 real calls, as one end of the call sends almost no RTP packets. We pushed the envelope further on this test and let it run until it reached stability (411 active calls) and both OpenSER and MediaProxy kept up with the pace. However,

[9] In the rest of this report, the CPU load is the CPU load relative to all the applications running as part of the DoD solution, this excludes the load generated by the system on which the DoD server is actually running.

[10] MediaProxy needs both the caller (UAC) and the callee (UAS) to connect to it via the port associated with the call to actually forward RTP packets between the UAS and the UAC. Therefore the UAS sends a short RTP stream to the UAC. That stream is an out-of-band DTMF signal made of 10 RTP packets so we can neglect it in our considerations.

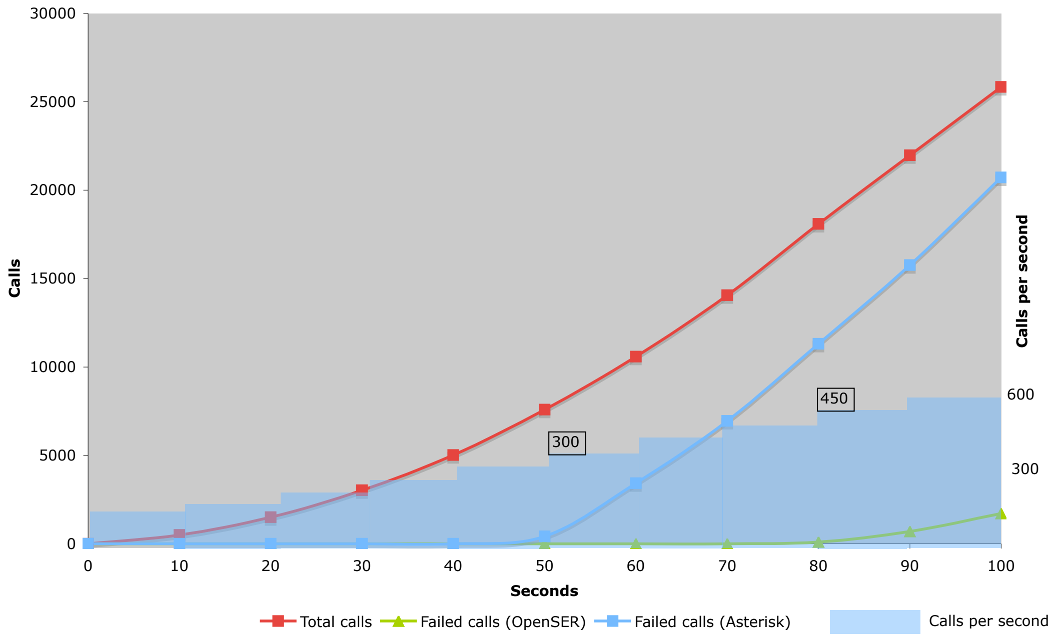


Figure 8.6: Relation between the number of failed calls and the calling rate

OpenSER was not able to handle 260 calls in the case the calls where placed at a higher rate than one call per second.

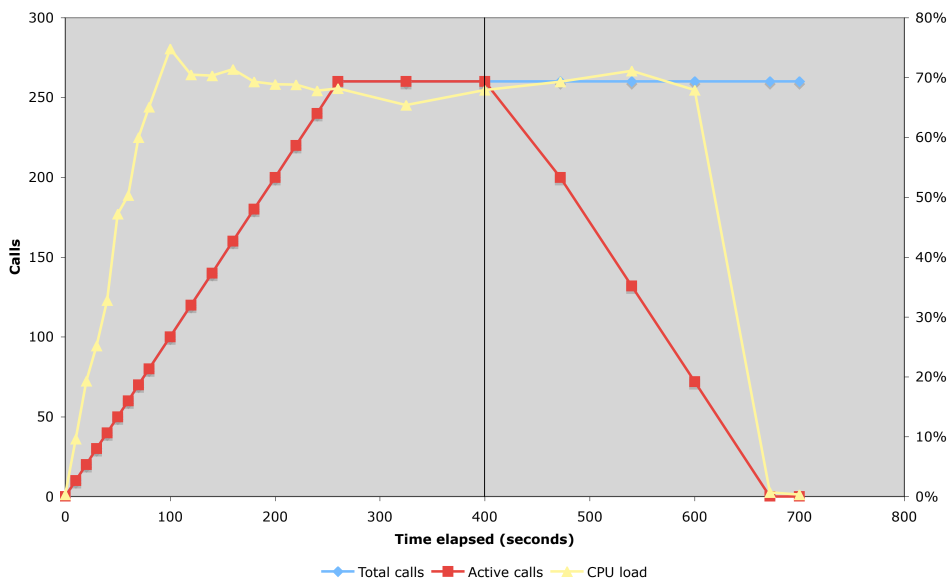


Figure 8.7: SIPp A test - 1 new call per second for 260 seconds

We also checked for UDP packet loss during the test described in the previous paragraph. With 260 simultaneous calls, the last 50 calls had a packet loss of 35 % to 48 %. This high packet loss accounts for the stable CPU load noticed after approximately a hundred calls. In any case, this clearly demonstrates that even though 260 calls can be executed in parallel on our test machine, these calls do not permit communication between two users.

Comparing with OpenSER without MediaProxy

The values above should be compared with those presented above that showed that OpenSER on the same machine without MediaProxy could handle up to 450 simultaneous calls without errors. The management of RTP streams by MediaProxy resulted in a loss of 86 % in capacity.

Table 8.2: SIPp B test - RTP packet loss

Simultaneous calls	Average packet loss	Below a packet loss of 1 %	Below 2 %	Failed calls
1	0.01 %	100 %	100 %	0
10	0.01 %	100 %	100 %	0
20	0.01 %	100 %	100 %	0
30	0.03 %	100 %	100 %	1
50	0.04 %	100 %	100 %	4
75	—	100 %	100 %	43

Comparing with Asterisk

The calls reported as failed in table 8.2 failed because MediaProxy did not get any packet from the UAS and therefore did not establish the session internally. This might be caused by the fact that the UAS was only sending a very small amount of packets. However, running the same test with Asterisk, we reached 50 simultaneous calls without losing any calls. The advantage of Asterisk in this case is that it processes both the SIP messages and the RTP streams inside a single application. Therefore the RTP socket is opened and both clients are registered to it without the need for Asterisk to receive any RTP packet. Note that in tables 8.2 and 8.3 percentages are given per placed calls, that is to say failed calls do not account for packet loss.

When running 50 SIP B calls at once using SIPp with Asterisk alone, the RTP packet loss was of 0.002 %. Furthermore, no call was dropped during the test. This tends to prove that Asterisk offers a better RTP solution for calls that do not have NAT-related issues. Therefore the OpenSER-MediaProxy association should be used only with external calls when NAT is a real issue. Note that here, the limiting application is MediaProxy and not SER/OpenSER since it is the RTP streams that generate the most load.

Two-way audio calls with OpenSER with MediaProxy

We ran a second series of tests where this time, both UAS and UAC were sending an audio stream B to one another. This series of test is closer to real calls and therefore is the most significant, but yet, the previous test highlighted a shortcoming of the OpenSER-MediaProxy solution: if one SIP client is for some reason sending very few UDP packets, the connection might not get established at all. It appeared that the limit of 23 calls could be exceeded and also that the load resulting from 23 calls with a two-way RTP stream was not double of that of 23 calls with a one-way RTP stream. In this case, two -way calls used only 20 % more of CPU resources compared to one-way calls. Further testing show that the threshold at which the calls drop below the 1 % quality limit is of 58 calls. It is also worth noting that the packet loss equally affects each call.

Table 8.3: Bursts of traffic on an idling system

Simultaneous calls	CPU load after 30 seconds	Calls below a packet loss of 1 %	Below 2 %
23	61.0 %	100 %	100 %
30	59.7 %	100 %	100 %
40	73.3 %	100 %	100 %
50	71.0 %	100 %	100 %
55	72.2 %	100 %	100 %
60	73.8 %	0 %	100 %

Comparing with Asterisk

When running 58 simultaneous calls using Asterisk, the CPU load after 30 seconds was only of 7.6 % and no calls were dropped. Furthermore all calls were far below the threshold of 1 % dropped packets (0.01 % in average). This proves again the superiority of Asterisk in terms of capacity.

The last tests we will conduct in this section concern sudden bursts of traffic on a loaded system. We tested sudden bursts of traffic on a previously idling system, which corresponded in terms of real use to the moment when offices open in the morning. Sudden bursts of traffic on a loaded system correspond to the case when an emergency arises and therefore should be tested for as well. In our test, we will consider a system handling 10 simultaneous B calls (running for one minute, resulting in a CPU load of approximately 20 % after one minute) when the burst occurs. The table presents values for all calls and not only the calls that belong to the burst.

Table 8.4: Bursts of traffic on a loaded system

Calls in the burst	CPU load after 30 seconds	Calls below a packet loss of 1 %	Below 2 %
10	37.7 %	100 %	100 %
20	56.3 %	100 %	100 %
30	73.1 %	100 %	100 %
40	75.2 %	100 %	100 %
50	72.8 %	0 %	0 %

We can see that the values for CPU load reported in table 8.4 are consistent with the ones reported in table 8.3, when adding the 10 background calls. Not so surprisingly, the largest burst we could emit before falling above the threshold of a 2 % packet loss was 47 calls. Above we noticed problems starting at 58 calls and 58 calls, which after subtracting the 10 background calls lead to a maximum burst size of 47 calls to avoid problems. Again, the packet loss was always equally spread between the calls, including the 10 background calls.

Conclusion on OpenSER and MediaProxy

The tests analyzed above showed that OpenSER used with MediaProxy, even if they provide with advanced features such as NAT transversal, does so at a pretty high cost in terms of performance (for 58 simultaneous normal (B) calls, OpenSER with MediaProxy was using over 70 % of CPU while Asterisk was barely using 8 %). To meet the goal of deploying a minimal equipment inside the premises of our customers, it seems better not to use MediaProxy unless there are real NAT-related issues, and in such case, MediaProxy should be run on a separate machine.

8.2.4 Combined tests

Design

Here we use the SIPp tests as a tool to simulate a background use of the DoD server while running other tests, such as the Jmeter series of tests. Seeing that the use of MediaProxy does not lead to an optimal system in common cases, we use Asterisk alone in this part. These series of tests are the most significant for our project as they test DoD's telephony component along with the software component.

We ran a JMeter test similar to that depicted in section 8.2.1, only with four virtual users instead of two and they placed 45 calls each (the test lasted approximately 45 hours). During this test, there were 10 continuous SIP calls (generated by SIPp with their RTP streams), handled by Asterisk. We chose to handle the RTP streams as well to represent the worst case scenario of calls being SIP to ZAP, thus requiring Asterisk to stay in the middle of the conversation. All this is assumed to simulate a solid traffic for a small company: with JMeter, we simulate outgoing traffic; with SIPp we simulate internal traffic.

The setup for this test was slightly different from that of the previous JMeter tests as it appeared that Windows XP does not behave reliably when a large number of sockets are handled over time – in addition to other resources being used. Therefore, on both development platforms, we ran 3 soft SIP clients. Machine 1 (as described in section 8.1.1) ran 2 soft SIP clients, machine 2 ran the SIPp UAC, and machine 3 executed the SIPp UAS. Again, all machines were located on the same LAN.

Results

In figure 8.8, we display the CPU load of the DoD server versus time when 3 DoD calls were ongoing. We can see that the CPU load is relatively stable at around 2 %, including the 10 SIPp calls. Peaks were noticed during these tests when the JBoss sessions (one per user connected) were actively used (when a user requests an update for instance). Such peaks were no higher than 8 % even when all users requested an update simultaneously. It is encouraging that on a rather old platform, the DoD server behaved properly while running contiguously for 45 hours. With peaks at 8 %, we can expect to support loads ten times larger with the same configuration we used in our tests – and obviously more calls could be supported if we were using a recent multiple-core server as DoD server.

Several comments can be made about the delays reported in table 8.5. First, one can see that, except for the "hanging up" action, delays are pretty stable compared to that observed during the first tests (see section 8.2.1). The longer delay observed concerning the "hanging up" action is best explained by examining the actual process behind it. When an action is to be done on an existing line, the JBoss Application server needs to know the identifier of this line, as used by Asterisk. Therefore, Asterisk has to poll the statuses of all the lines before the JBoss Application Server can undertake the action. Indeed, when the call is placed, Asterisk does not return the line identifier for the originating line, let alone the callee's line, to the JBoss Application Server. The only way to keep track of the lines is via a polling. This polling could be done systemically

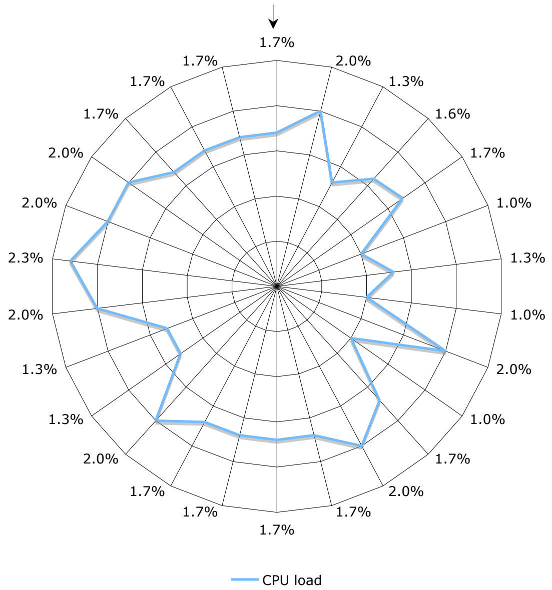


Figure 8.8: Evolution of the CPU load over the time, with 3 ongoing calls, each ring representing 0.5 % of CPU load

but it seems more suitable to do it only when requested by the user. Indeed, if the user does not wish to check on the call's status or hang its line via the DoD interface, there will be no need for the JBoss Application Server to know the identifiers of the lines involved in this call. As a result, with the additional ten calls running, Asterisk has additional lines to poll, thus making the whole action longer. A solution to this problem could be use a cluster of two (or more) Asterisk servers, assigning one or more exclusively to DoD calls, and the other one(s) to internal calls. This way, the lines created by the internal traffic would not slow down the processing of the actions within the DoD solution.[11]

Table 8.5: Results for each operation

Action	Average delay	90 % delay	Error rate
Connecting	1,043 ms	1,200 ms	0 %
Authentication	83 ms	63 ms	0 %
Calling	696 ms	703 ms	0 %
Updating	1,151 ms	1,125 ms	0 %
Hanging up	2,133 ms	4,125 ms	0 %
Logging out	12 ms	16 ms	0 %
Total	928 ms	2,171 ms	0 %

8.2.5 Conclusions

The results given in this chapter are applicable for a single-site installation, running an all-in-one DoD server. Running single-site or multi-site clusters of DoD Server, especially with separate machines for Asterisk, SER, and JBoss would lead to other results; but such installations are not the intended customers for the first release of the DoD solution.

[11] Note that the DoD Asterisk server(s) could communicate with the non-DoD Asterisk server(s) using the IAX protocol.

From the results, it seems advisable to use Asterisk with OpenSER[12] for the DoD Server in the case NAT transversal is not an issue. However, if presence is not requested, Asterisk alone should suffice. However, if there are serious NAT transversal issues, MediaProxy becomes a necessity – along with OpenSER – but in such a case, the installation should utilize at least two separate servers - with MediaProxy running on a separate server. In terms of performance, even though the use of OpenSER with MediaProxy and Asterisk altogether on a single machine resulted in a lesser scalability, the reader is reminded that this setup – on a fairly old machine, was still able to handle approximately the equivalent of an E1, which should be more than sufficient for numerous customer companies (cf. chapter 5)

8.3 External tests

8.3.1 SIP stack

The limited scalability of Asterisk regarding SIP calls is addressed by the substitution of a SER server to the Asterisk server for SIP registrations and pure SIP calls (cf. chapter 7). According to a study [20], a SER dual-node cluster can handle the SIP messages of up to 20,000 simultaneous calls and there is the possibility of adding more nodes or more clusters. This should be more than sufficient for most of the intended customers. In the Asterisk user community, it is deemed reasonable to expect capacities of a few hundred SIP users on a single Asterisk installation, without SER [99][13]. In the rest of this section we will report the results of studies lead by TransNexus® on the SIP performance of Asterisk, SER, and OpenSER and compare their results to ours.

Asterisk

In this first test [90], Asterisk is used as a back-to-back user agent (B2BUA) between a SIP client and five possible hosts, four of these hosts leading to a failure in the SIP messages exchanges (either by not replying or rejecting them for instance). SIPp [78] is used as a SIP client and the only valid SIP host. The host running Asterisk is a Dell Precision 490 server with two Intel Xeon 5140 dual core CPUs (2.33 GHz, 4 GB RAM). This machine is significantly more powerful than the ones used for our tests, therefore the numerical values cannot be directly compared between these tests and ours. The complete description of the benchmark ran by TransNexus is described in [92], including the configuration files. Other noticeable differences from our tests are the use of a separate machine to store the call detail records (CDR) along with the addition of an Open Settlement Protocol server (OSP) for inter-domain billing purposes. The TransNexus team also decided to limit the number of retransmissions of SIP messages down to one. This is justified for their tests since they are purposefully trying to contact unreachable hosts. However, this does not properly model a real-life situation where the server should be more insistent when facing a failure.

Figure 8.9 corroborates the comment made above about the relative significance of CODEC translation in the CPU load on the DoD Server. Another important point made by this study is the cost calculation. The host used for the tests costed US$ 3,000. Using no CODEC translation, it could run 1,500 simultaneous calls; with CODEC translation, it could support only 400 simultaneous calls. This leads respectively to a cost of US$ 2 per port and US$ 7.5 per port[14].

The conclusions concerning the scalability of Asterisk for SIP calls reached by their study are the following:

[12] To provide better support of presence informations

[13] Again these results depend on the hardware used. In our case, we reached a rate of 450 simultaneous SIP calls per second handled by OpenSER on a reasonably old server.

[14] In their report [90], the team of TransNexus included the price of the CODEC used. Other CODECs could have been used without any cost for the same test, so we did not incorporate this cost in the results reported in this thesis.

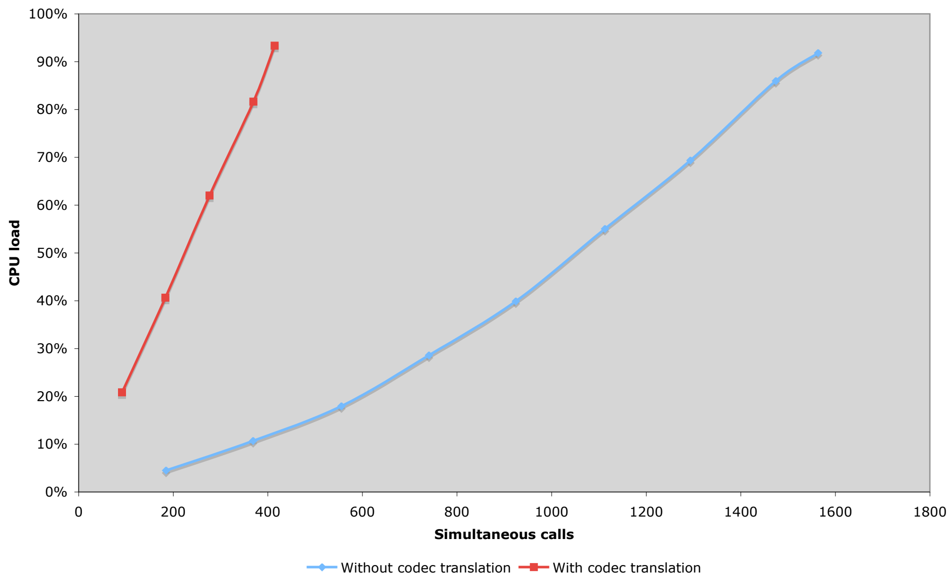


Figure 8.9: Asterisk performance as B2BUA as reported in [92]

- 1 GHz of CPU resource is enough for 160 simultaneous calls without transcoding.
- 1 GHz of CPU resource is enough for 40 simultaneous calls with transcoding.

During our tests, we could support 160 simultaneous calls without transcoding with a load of 40 %. Considering that we were using a CPU running at 2.53 GHz, this results in 158 calls per 1 GHz of CPU used, which is consistent with the results of TransNexus' study.

OpenSER and SER

In [91], TransNexus compares the performance for pure SIP calls of three systems: SER 2.0, OpenSER 1.1, and OpenSER 1.2. The complete modus operandi for these tests is to be found in [93]. Note that these tests did not include RTP streams in contrast to the ones concerning Asterisk and to some of the tests during this thesis project.

The conclusions concerning the scalability of OpenSER 1.2 and SER 2.0 for SIP calls reached by this study are the following: 1 GHz of CPU resource is enough for 60 calls per second without any audio support with a load of 60 %. During our tests we coud support 450 calls per second on a machine using an Intel Celeron 2.53 GHz, which leads to 138 calls per second per 1 GHz of CPU resource loaded at 60 %. This is over twice as many calls. Such a difference can be explained by the fact that OpenSER was not compiled with the same options in both cases and also by the fact that in our case, all calls succeeded on the first try, which was not the case in the test ran by TransNexus®. The study concludes that there is little difference between the performances of OpenSER 1.2 and SER 2.0, except maybe in their post-dial delay. However, OpenSER 1.2 is said to perform 13.4 % more CPU-efficiently than OpenSER 1.1. The reader is referred to the detailed results of their test [93] for graphs and exact results.

8.4 Field tests

This section is devoted to the analysis of the results from field tests. That is to say, these results come from tests with actual users and, thus, focus on qualitative results rather than quantitative ones.

8.4.1 Latency

We present in table 8.6 results concerning the delay elapsed between a called being placed and the caller's line actually ringing as well as the total delay before the callee's line is ringing. The SIP phone was registered with OpenSER and OpenSER was associated with a MediaProxy. The H.323 phone was registered to an innovaphone PBX IP800 connected to Asterisk over SIP. The mobile and landline phones were called using a GSM gateway (*or a SIP trunk*) connected to Asterisk. This is the setup detailed on figure 7.6. The delays shown in this table do **not** include the delay for the non-human detection. These delays were measured without any other call running on the server. 0-second delays signify here that the event seems to happen instantaneously for the user, the purpose of this test was not to achieve millisecond resolution measurements. We also show in table 8.6 delays for direct calls in comparison.

Table 8.6: Latency using a GSM gateway (*or a SIP trunk*)

Caller's line	Callee's line	Interface	Delay to caller's line	Total delay
Mobile phone	Landline phone	Web interface	9 seconds (*5*)	15 seconds (*10*)
		Lightweight[a]	9 seconds (*6*)	15 seconds (*12*)
		Java client	14 seconds (*12*)	19 seconds (*16*)
		Direct call	–	5 seconds
Mobile phone	SIP phone	Web interface	8 seconds	8 seconds
		Lightweight	8 seconds	8 seconds
		Java client	13 seconds	13 seconds
Mobile phone	H.323 phone	Web interface	10 seconds	10 seconds
		Lightweight	9 seconds	9 seconds
		Java client	15 seconds	15 seconds
SIP phone	Mobile phone	Web interface	0 second	11 seconds
		Lightweight	1 second	9 seconds
		Direct call	–	11 seconds
SIP phone	SIP phone	Web interface	0 second	0 second
		Lightweight	2 seconds	2 seconds
SIP phone	H.323 phone	Web interface	0 second	0 second
		Lightweight	1 second	1 second
H.323 phone	Mobile phone	Web interface	0 second	10 seconds
		Lightweight	2 seconds	10 seconds
H.323 phone	SIP phone	Web interface	0 second	0 second
		Lightweight	2 seconds	2 seconds
H.323 phone	H.323 phone	Web interface	0 second	0 second
		Lightweight	1 second	1 second

[a] Lightweight Web interface, over GPRS link

Comparing with a loaded server

We measured the latency for a call between a mobile phone and a H.323 phone using the Web interface while artificially loading Asterisk with SIP calls with one-way RTP streams using SIPp. With up to 100 simultaneous calls this did not increase the delay observed by the user. This proves the transversal time for Asterisk is stable even with a relatively high load.

Here is a short summary of the results displayed in table 8.6. The lightweight Web interface designed for a mobile use is at the most 1 or 2 seconds slower than its normal counterpart. The

Java implementation is slower than the lightweight Web interface even though both are using the same GPRS network. One reason for this difference is that the MIDP client needs to establish a connection with the DoD Server **before** placing a call whereas the browser used to navigate on the lightweight interface has already opened that connection. Calls to mobile phones are the longest to be established. It takes approximately 10 seconds. Calls to a landline phone using the GSM gateway take only 6 seconds on average. The difference between the two is explained by the longer call termination in the mobile network, compared to the PSTN network. However, the delay to connect to a phone is much shorter (up to 5 seconds) in the case the call was made using a SIP trunk (cf. values in italic).

> **Comparing with other solutions**
>
> The delays mentioned above can seem quite long, but, as mentioned in section 8.2.1, only the delay necessary to place the first half of the call is perceived by the caller, the delay to connect the second leg of the call is hidden from the caller by a ringing signal. Furthermore, we observed when trying to place calls with other mobile solutions, such as Fring [25], that the approximate time to connect the caller to the callee was of approximately ten seconds.

8.4.2 Audio quality

Besides the quantitative tests ran concerning the audio quality (e.g. measuring packet loss ratio), we ran some qualitative tests. It turned out that the call termination means was also playing a great role, just like it does when it comes to delays (see section 8.4.1). When connecting two mobile phones, the use of a GSM gateway resulted in a quality inferior to that of a mobile to mobile call whereas the use of an externally provided SIP trunk resulted in a very good quality, comparable to that obtained in a call between two fixed lines. When connecting internal phones to external phones (wether they were mobile or fixed phones), we found that the quality was good.

8.5 Conclusions

All the tests described here shed a positive light on the DoD solution. Its actual components appeared to be stable and support sufficient load for even medium-sized deployments. Such guarantees had not previously been achieved for the DoD solution, which is encouraging for its future development. It also appeared that NAT transversal solutions can be implemented in the DoD solution at the cost of an extra machine to support the additional processing. Keeping in mind the desire of designing a DoD solution that could run on a single machine, it seems that adding SER to the DoD solution would reduce greatly its telephony performances but benefit the users by providing more of the features generally expected from SIP.

Concerning user-oriented aspects, such as audio quality and delays, we reached the conclusion that the call termination channel played a very important role. Namely, the GSM gateway is the means offering the worst performances for these two criterions of all the means tested during this thesis. This, their poor scalability potential, and the transience of the cost reduction they bring[15] made us revise the role of GSM gateways in the DoD solution.

[15] Since operators are likely to change their tariffs to adapt to this new form of concurrence.

Chapter 9

Conclusion

After having described at great length all the work done in this project, whether as research or development, we will now present our conclusions, beginning by a summary of the achievements of this thesis project and finishing with a list of both short- and long-term development possibilities for this project in the future.

9.1 Achievements

During this thesis project, we improved the DoD solution in various domains:

- stability has been improved and tested, which proves the DoD solution is ready for a commercial deployment;
- features have been added that expand the benefits of the DoD solution beyond the simple cost reduction factor, thus expanding the potential sales;
- the DoD solution connectivity has been improved in terms of call termination as well, which offers greater flexibility to companies to choose the most advantageous (in the broadest meaning of the term) way for them to terminate their calls.

A user's guide was also written during this thesis project. It is aimed at customers and details the use of the DoD solution for both common users and system administrators. Details are given on how to configure the DoD solution to interface with the existing computing and telephony equipment of the company. More generally speaking, significant effort has been made to facilitate future work to improve the DoD solution. The documentation of the project and its organization have been significantly revised. UML diagrams and JavaDoc have been created, a source-code versioning system has been set up; all in the intent of enabling new comers to work more easily on the DoD solution. We did not mention these works earlier as their details are of little interest for a thesis project; however, we think they are valuable enough to be mentioned here.

We explored the possibility to use SER and MediaProxy to handle SIP calls instead of Asterisk, in order to improve NAT handling and offer presence support, among other advantages. However, these programs seemed to be quite resource-consuming. Therefore, our conclusion was that they might only be suitable for small companies or for the large companies that could afford multi-node installations. Additionally, support for presence has been implemented directly into the JBoss Application Server. Note that this does not preclude presence information to be shared with SER or other solutions in the future. Indeed, thanks to the strong modularity of our project and the flexibility of the JBoss Application Server, it is possible to connect to various databases (such as presence information databases) without requiring major modifications in the source code.

We extensively tested the DoD solution. We tested OpenSER and MediaProxy – which lead to the conclusions stated above, but also tested Asterisk and the JBoss Application Server. These tests – supported by the comparison with the tests of others – guarantee the stability of the DoD solution for quite large installations even though our testing was limited to relatively old test equipments. The fact that both Asterisk and the JBoss Application Server have the possibility to run on clusters of multiple nodes is another guarantee of scalability. The main concern for the scalability of the DoD solution is the means used to terminate calls. SIP trunks provided by external SIP providers offer a solution that scales well, whereas GSM gateways have limited scalability due to the cellular nature of the GSM network and the limited number of channels which operators are likely to make available at a given physical site. It is up to the end-user companies to decide what is their preferred means of call termination, keeping in mind factors including reliability, costs, audio quality, and scalability. The DoD solution has actually been designed to enable companies to connect to a wide range of communication networks.

9.2 Future work

As the reader should realize, the possibilities for future development of the DoD solution are very wide and the principle step remaining before finalizing the solution is to define precisely where to stop these developments. This thesis project did not aim at integrating as many features as possible into the DoD solution. Even though integrating support for LDAP or other databases and external proprietary equipment would be of benefit for the product, it was of little interest for a thesis project. We chose instead to prove that it was possible to add communication with external communications systems primarily as a proof of concept. This interfacing is not complete and currently consists only of passing along calls to another interface via Asterisk, but in the long term it would be interesting to focus on interconnecting to a number of different PBXs and offer more support and functionalities for them, using the PBX's API when available. This would mean establishing a partnership or a contract with the manufacturers to have privileged access to the internals of their devices. There is also some work left if all of the features offered by Asterisk are to be integrated into the DoD Server directly (such as call recording).

The project has currently no loose ends, but many features are ready to be implemented that could add to the functionality of the solution. Full and interactive support for presence and conferences, as well as complete integration of all of the mobile phones' features into the MIDP client (such as direct calls and SMS) are some of these functions which should be added in the future. To continue the convergence of the DoD solution, developing a client for the Symbian platform and the up-coming Android platform [9] would be beneficiary as well. The current focus inside Opticall AB is on platforms, such as Symbian, that offer deeper integration with mobile phones, meaning complete transparency and no hassle for the users.

Bibliography

[1] GSM 02.08. GSM system performance. Technical report, ETSI PT12, August 1995.

[2] 2N Telekomunikace. User Guide for Stargate, Bluestar and Bluetower. Version 1.3, 1 September 2006.

[3] 2N Telekomunikace. Blue tower. http://www.2n.cz/products/gsm_gateways/isdn_pri_gsm_gateways/bluetower_gsm_gateway.html, 20 September 2007.

[4] 2N Telekomunikace. VoiceBlue Lite. http://www.2n.cz/products/gsm_gateways/voip_gsm_gateway/voiceblue_voip_gsm_gateway.html, 10 January 2008.

[5] 3CX. VOIP Phone manual. http://www.3cx.com/VOIP/voip-phone-manual.html, 19 October 2007.

[6] Stelacon AB. Utredning avseende GSM-frekvenser. http://pts.se/Archive/Documents/SE/Utredning%20avseende%20GSM-frekvenser.pdf, 2002.

[7] AdventNet Inc. ManageEngine. http://manageengine.adventnet.com/, 24 September 2007.

[8] AG Projects. MediaProxy. http://www.ag-projects.com/MediaProxy.html, 18 October 2007.

[9] Open Handset Alliance. Home page. http://www.openhandsetalliance.com/index.html, 21 November 2007.

[10] Apache Jakarta Project. JMeter. http://jakarta.apache.org/jmeter/, 01 November 2007.

[11] Eric Armstrong, Jennifer Ball, Stephanie Bodoff, Debbie Bode Carson, Ian Evans, Dale Green, Kim Haase, and Eric Jendrock. *The J2EE 1.4 Tutorial, For Sun Java System Application Server Platform Edition 8.1 2005 Q1*. Sun Microsystems, Inc., December 2004.

[12] Bluetooth. Learn. http://www.bluetooth.com/Bluetooth/Learn/, 20 November 2007.

[13] Brekeke. Home page. http://www.brekeke.com/, 20 September 2007.

[14] Callweaver. Home page. http://www.callweaver.org/, 20 September 2007.

[15] Emmanuel Cecchet, Julie Marguerite, and Willy Zwaenepoel. Performance and scalability of EJB applications. Master's thesis, Rice University, Houston, Texas, 2002.

[16] Swedish Number Portability Administrative Center. Home page. http://www.snpac.se/, 21 September 2007.

[17] Chan-mobile. Home page. http://www.chan-mobile.org/, 20 November 2007.

[18] Carrier Class. Home page. http://www.carrierclass.net/, 15 October 2007.

[19] Collective. VOIP Wiki - a reference guide to all things VOIP. http://www.voip-info.org/, 13 September 2007.

[20] SKYY CONSULTING. ITSP Scalable Architecture Using Asterisk. http://www.skyyconsulting.com/itsp_voip_asterisk.php, 15 October 2007.

[21] Les Cottrell, Warren Matthews, and Connie Logg. Tutorial on Internet Monitoring & PingER at SLAC. http://www.slac.stanford.edu/comp/net/wan-mon/tutorial.html#loss, 1 June 2007.

[22] CounterPath. X-Lite. http://www.counterpath.com/xlitedownload.html, 18 October 2007.

[23] Digiumcards. TE110P Digium Asterisk. http://digiumcards.com/te110p.html, 10 January 2008.

[24] ETSI. TS 03.90 - Unstructured Supplementary Service Data (USSD). 3GPP, 25 June 1999.

[25] Fring. Home page. http://www.fring.com/, 07 January 2008.

[26] Adrian Georgescu. Best practices for SIP NAT traversal. Technical report, AG-Projects, 17 May 2007.

[27] Apache Geronimo. Home page. http://geronimo.apache.org/, 21 September 2007.

[28] Philippe Godlewski, Xavier Lagrange, and Philippe Martins. L'Accès Paquet dans GPRS. In *Support de cours RMOB*. École Nationale Supérieure de Télécommunications (ENST), March 2003.

[29] Bernard Goldfarb and Catherine Pardoux. *Méthodes d'ajustements graphiques : Diagramme Probabilité – Probabilité*, MODULAD, 33:1–5, 2005.

[30] GSM Awards. GSM association awards 2007 - winners 2007. http://www.gsmawards.com/history/2007_winners/telepo.html, 3 September 2007.

[31] Mattias Hansson. GSM Gateway 2007. Opticall AB Seminar, Kista Science Tower, 4 September 2007.

[32] Mattias Hansson and Jörgen Steijer. Method, call setup device and computer product for controlling and setting up calls with reduced costs. WO/2006/083208, August 2006.

[33] Mattias Hansson and Jörgen Steijer. Method, call setup device and computer product for controlling and setting up calls with reduced costs. EP1847104, 24 October 2007.

[34] Hitachi. WirelessIP 5000 home page. http://www.wirelessip5000.com/eng/, 20 September 2007.

[35] innovaphone. IP110. http://www.innovaphone.com/index.php?id=125&L=4%23WT, 20 September 2007.

[36] innovaphone. IP800. http://www.innovaphone.com/index.php?id=120&L=0, 20 November 2007.

[37] Intel. Overcoming barriers to high-quality voice over ip deployments. Technical report, March 2003.

[38] Intel. *Traffic Engineering Model for LAN Video Conferencing*, 22 November 2005.

[39] iptel.org. SER presence handbook. http://ftp.iptel.org/pub/ser/presence/presence-handbook/, 15 October 2007.

[40] iptel.org. SIP Express Router, Home page. http://www.iptel.org/ser/, 21 September 2007.

[41] IT Facts. 20% of US businesses use VOIP. http://www.itfacts.biz/20-of-us-businesses-use-voip/8340, 28 April 2007.

[42] IT Facts. Home page. http://www.itfacts.biz/, 3 September 2007.

[43] JAsterisk. Home page. http://sourceforge.net/projects/jasterisk/, 21 September 2007.

[44] Asterisk Java. Home page. http://asterisk-java.org/, 21 September 2007.

[45] JBoss. High accessibility and clustering topics. http://wiki.jboss.org/wiki/Wiki.jsp?page=JBossHA, 21 September 2007.

[46] JBoss. jboss.org: community driven. http://labs.jboss.com/, 3 September 2007.

[47] Jetexy. EasyCall. http://www.jetexy.com/eng/index.html, 02 November 2007.

[48] Alan B. Johnston. *SIP: understanding the Session Initiation Protocol*. Artech House Publishers, January 2001.

[49] JOnAS. Home page. http://wiki.jonas.objectweb.org/xwiki/bin/view/Main/WebHome, 21 September 2007.

[50] JScience. Home page. http://jscience.org/, 22 November 2007.

[51] JSR 120 Expert Group. Wireless Messaging API (WMA) for Java 2 Micro Edition. Technical report, Java Community Process (JCP), 25 April 2003.

[52] Ravi Kalakota and Marcia Robinson. *M-Business, The Race to Mobility*. McGraw-Hill, 2001.

[53] Donald Knuth. *Seminumerical Algorithms*, volume 2 of *The Art of Computer Programming*. Addison-Wesley, 1969.

[54] Xavier Lagrange. Planification cellulaire régulière. In *Support de cours RMOB*. École Nationale Supérieure de Télécommunications (ENST), November 1999.

[55] Jonas Larsson and Fredrik Öst. The Swedish telecommunications market 2006. Technical report, Post & telestyrelsen, 7 June 2007.

[56] Philippe Martins. Procédures dans le réseau coeur GSM/GPRS. In *Support de cours RMOB*. École Nationale Supérieure de Télécommunications (ENST), 2006.

[57] Jim Van Meggelen, Jared Smith, and Leif Madsen. *Asterisk, The Future of Telephony*. O'Reilly, August 2005.

[58] Sun Microsystems. Java Authentication and Authorization Service (JAAS) Reference Guide for the Java SE Development Kit 6. http://java.sun.com/javase/6/docs/technotes/guides/security/jaas/JAASRefGuide.html, 21 November 2007.

[59] Sun Microsystems. Java EE Compatibility. http://java.sun.com/javaee/overview/compatibility.jsp, 21 September 2007.

[60] Sun Microsystems. Enterprise JavaBeans Technology. http://java.sun.com/products/ejb/, 21 February 2008.

[61] Killer Mobile. CallingCard. http://www.killermobile.com/callingcard.html, 02 November 2007.

[62] Johan Montelius. Messaging – SMS, WAP Push and MMS. In Course material for KTH, editor, *Developing Mobile Applications - 2G1722*, February 2007.

[63] The University of Edinburgh. Festival. http://www.cstr.ed.ac.uk/projects/festival/, 24 September 2007.

[64] Opera Software ASA. Home page. http://www.operamini.com/, 21 November 2007.

[65] Opticall AB. Home page. http://www.opticall.se, 3 September 2007.

[66] OptiMobile AB. Home page. http://www.optimobile.se/, 25 September 2007.

[67] Per-Oddvar Osland and Khanh Dinh. Perceived VoIP quality under varying traffic conditions. Telenor, August 2004.

[68] Ivar Plathe and Sven Evensen. Mobile branch exchange. WO02078368, 2 January 2004.

[69] Sanzheng Qiao and Liyuan Qiao. A robust and efficient algorithm for evaluating Erlang B formula. Master's thesis, Mc Master University, Ontario, Canada, 1998.

[70] Quasar, Parabel Ltd. Board testing: CPU load in voice mode. http://quasar.sourceforge.net/bench.htm, 23 November 2007.

[71] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC3261, June 2002.

[72] Jonathan Rosenberg. XCAP Tutorial. In *IETF 59*, March 2004.

[73] Jonathan Rosenberg. Extensible markup language (XML) formats for representing resource lists. RFC4826, May 2007.

[74] Jonathan Rosenberg, Rohan Mahy, and Christian Huitema. Traversal Using Relay NAT (TURN). Internet-Draft, September 2005.

[75] SICAP. Prepaid roaming. http://www.sicap.com/home/index.php?rubrikid=2&subsiteid=24, 20 September 2007.

[76] Henry Sinnreich and Alan B. Johnston. *Internet Communications Using SIP - Delivering VoIP and Multimedia Services with Session Initiation Protocol*. Wiley, 2006.

[77] SIPKnowledge. SIP/IMS Research - SIP Related IETF RFCs and Drafts. http://www.sipknowledge.com/SIP_RFC.htm, 20 November 2007.

[78] SIPp. Home page. http://sipp.sourceforge.net/, 24 September 2007.

[79] SJ Labs. Home page. http://www.sjlabs.com/, 18 October 2007.

[80] Skype. Home page. http://www.skype.com, 3 September 2007.

[81] SMS Forum. Home page. http://smsforum.net/, 31 July 2007.

[82] Telia Sonera. Home page. http://www.telia.se/, 20 September 2007.

[83] Symbian. Home page. http://www.symbian.com/, 02 November 2007.

[84] FirstHand Technologies. Mobile Unified Communications Solution, Enterprise Mobility Leader, Enterprise Fixed Mobile Convergence. http://www.firsthandtech.com/, 3 September 2007.

[85] Tele2. Home page. http://www.tele2.se/, 20 September 2007.

[86] 2N Telekomunicace. Home page. http://www.2n.cz/, 13 August 2007.

[87] Telepo. Home page. http://www.telepo.com/, 3 September 2007.

[88] Luis MG *et al.* Home page. http://www.tcpdump.org/, 7 November 2007.

[89] Apache Tomcat. Home page. http://tomcat.apache.org/, 3 September 2007.

[90] TransNexus. Asterisk performance. http://www.transnexus.com/White%20Papers/asterisk_V1-4-11_performance.htm, 29 November 2007.

[91] TransNexus. OpenSER and SER performance test and comparison. http://www.transnexus.com/White%20Papers/OpenSER-SER_Comparison.htm, 29 November 2007.

[92] TransNexus. Performance Benchmark Test for Asterisk B2BUA. Technical report, 13 November 2007.

[93] TransNexus. Performance Benchmark Test for OpenSER and SIP Express Router. Technical report, 4 June 2007.

[94] Tre. Home page. http://www.tre.se/, 20 September 2007.

[95] UMA Techonology. Home page. http://www.umatechnology.org/, 30 January 2008.

[96] Unstrung. TeliaSonera preps femto trial. http://www.unstrung.com/document.asp?doc_id=143665&f_src=unstrung_gnews, 21 January 2008.

[97] Voip-Info.org. Answering machine detect. http://www.voip-info.org/wiki/index.php?page=Asterisk+cmd+AMD, 21 November 2007.

[98] Voip-Info.org. Asterisk at large. http://www.voip-info.org/wiki-Asterisk+at+large, 15 October 2007.

[99] Voip-Info.org. Asterisk dimensioning. http://www.voip-info.org/wiki/index.php?page=Asterisk+dimensioning, 24 September 2007.

[100] Voip-Info.org. Asterisk hardware recommendations. http://www.voip-info.org/wiki/view/Asterisk+hardware+recommendations, 24 September 2007.

[101] Voip-Info.org. Asterisk setup home. http://www.voip-info.org/wiki/view/Asterisk+setup+home, 24 September 2007.

[102] Voip-Info.org. Asterisk setup medium office 100. http://www.voip-info.org/wiki/view/Asterisk+setup+medium+office+100, 24 September 2007.

[103] Wikipedia. Enterprise information system. http://en.wikipedia.org/wiki/Enterprise_Information_System, 20 September 2007.

[104] Wikipedia. GSM frequency bands. http://en.wikipedia.org/wiki/GSM_frequency_bands, 24 September 2007.

[105] Wikipedia. JavaBeans. http://en.wikipedia.org/wiki/Java_Beans, 3 September 2007.

[106] Wikipedia. Generic Access Network, devices. http://en.wikipedia.org/wiki/Unlicensed_Mobile_Access#Devices, 30 January 2008.

[107] Kineto Wireless. Home page. http://www.kinetowireless.com/index.html, 30 January 2008.

[108] Wireshark. Home page. http://www.wireshark.org/, 19 December 2007.

[109] Ning Zhou. Dial over Data via GSM Gateway. Master's thesis, Department of Microelectronics and Information Technology (IMIT), Royal Institute of Technology (KTH), Sweden, June 2005.

COS/CCS 2008-02