



RHODES UNIVERSITY
Where leaders learn

Development Of A BingBee Phone Network

Submitted in partial fulfilment
of the requirements of the degree
Bachelor of Science (Honours) in
Joint Computer Science and Information Systems
at Rhodes University

By Taurai Saurombe

5 November 2007

Supervisor: Professor Peter Wentworth

Co Supervisor: Hannah Slay

Abstract

This paper discusses the development and implementation of a BingBee VoIP Telephone system. The aim of this project is to create a fully functional VoIP system which can be used as an edutainment resource for young children. The system has multiple functions varying from the provision of simple telephone calls to requests for book readouts and also offers some educational games which will be played over the handsets through an Interactive Voice response menu from the system. A prototype proof-of-concept system with three handsets has been deployed at the Raglan Road Multipurpose Centre Pre School, a small school catering for previously disadvantaged children in Grahamstown in South Africa.

Acknowledgements

I would like to thank my supervisor Professor Peter Wentworth for his guidance and patience with me during the year and for all the help he gave me when I got stuck. I would also like to thank M.Tsietsi for his assistance on the Asterisk related problems I had that he helped me to solve and also I would like to extend my thanks to the staff and young children at Raglan Road School for using this system. I would also like to extend my thanks to Mr Barry Irwin for his help in setting up my Fedora 7 virtual server. Also special thanks to the Computer Science Department at Rhodes University for equipment supply and also thanks to the departments' sponsors shown below:



Table of Contents

CHAPTER 1: Project Background	7
1.1. Definition of Project.....	7
1.2. Project Objectives	7
1.3. Project Deliverables	7
1.4. Project Plan	7
1.5 A Guide to the Remainder of the Report.	9
1.6. Chapter Summary	9
CHAPTER 2: Literature Review.....	10
2.1. Introduction.....	10
2.2. VoIP	10
2.3. Asterisk	12
2.4. Storytelling and Edutainment.....	14
2.5. Telephones and Edutainment	14
2.6. Conclusion	15
2.7. Chapter Summary	16
CHAPTER 3: Design and Implementation	17
3.1. Installing Asterisk	17
3.2. The Asterisk Dialplan	17
3.2.1. Contexts	18
3.2.2. Extensions	18
3.2.3. Priorities	20
3.2.4. Applications	21
3.3. Difficulties Encountered	23
3.3.1. NAT and Asterisk	23
3.3.2. Asterisk Configurations	25
3.3.3. Server Firewall.....	27
3.4. Chapter Summary	29
CHAPTER 4: Results and Evaluation after each release	30
4.1. System Release 1	30
4.1.1. User Stories for System Release 1	30
4.1.2. Developed Functions.....	30

4.1.3. Difficulties Encountered	34
4.1.3. System Evaluation and User Feedback.....	35
4.2. System Release 2	36
4.2.1. User Stories for System Release 2	36
4.2.2 Developed Functions.....	36
4.2.2.1. Call Function.....	36
4.2.2.2. Book Readout Function	37
4.2.2.3. Say Number Function	38
4.2.2.4. Calculator Function.....	39
4.2.3. Difficulties Encountered	39
4.2.3. System Evaluation and User Feedback.....	39
4.3. Chapter Summary	39
CHAPTER 5: Further Work and Conclusions	41
5.1. Alternative uses.....	41
5.2. Chapter Summary	42
References:.....	43
Appendix.....	46
How to Test the System Running on Chameleon	46
How to Configure the Telephone Handset.....	46
Directory of the Currently Available Services.....	47
Complete Project User Stories	48
Asterisk Configuration Files	49
Asterisk Configuration Files	49
extensions.conf (The Asterisk Dialplan).....	49
sip.conf.....	54
Project Poster	56

Table of Figures

Figure 1: A screenshot of the Asterisk CLI when "core show applications" is run.....	22
Figure 2: SIP session with DNS servers [16].....	27
Figure 3: Wav file properties to be used for the Book Readout Function	29
Figure 4: Telephone Settings Web Interface.....	32
Figure 5: Telephone Address Book.	32
Figure 6: Proposed Single Site Network Layout	36
Figure 7: Telephone internal settings to test the server running on Chameleon.....	46

Table of Listings

Listing 1: Example context to illustrate priorities.....	21
Listing 2: Example context	22
Listing 3: The current Asterisk Dialplan for the BingBee Phone Network	53
Listing 4: The current sip.conf file for the BingBee Phone Network	55

CHAPTER 1: Project Background

1.1. Definition of Project

There is currently an increasing interest in the field of telephony to offer value added services through telephones. The mobile telephone industry is now offering ever more increasing services and with this theme in mind, we think there is an exciting emerging opportunity to use desktop telephones instead of cell telephones to provide edutainment resources for young children. This idea to offer educational services over desktop telephones is also encouraged by the increased sophistication of the newer IP telephones being introduced these days. This project aims to develop a VoIP system to offer traditional telephony services and additional custom services like book readouts and edutainment games to young children.

The recent availability of open source PBXs, specifically Asterisk, now gives us the ability to create small special-purpose telephone networks with programmable services that allow us to add custom functionality to our VoIP networks.

1.2. Project Objectives

In this project we want to test the idea that a VoIP network can be developed to offer custom services for edutainment purposes specifically targeting young children. The way we want to proceed to test this idea is to develop services for desktop telephones like book readouts, arithmetic games, counting games and in addition also develop the VoIP network to handle the traditional call setup function of a telephone network.

1.3. Project Deliverables

The final deliverable for this project is an Asterisk server that is running a complete VoIP system that offers various edutainment services.

1.4. Project Plan

Because of the XP programming methodology adopted for this project, the BingBee telephone system was deployed early in two small releases, the first one at Raglan Road Pre School, and then the second one in the Computer Science department as a test bed before going on to do the complete deployment at Raglan Road Pre School. Prototyping was used to test the system after each major function had been developed to ascertain if the system was functioning as expected.

The following table gives a guide of the activities that were carried out to add services to the server

and their rough time to completion during the project, although, in the spirit of XP, some of these dates were changed from the originally planned dates because of difficulties encountered and the fact that a connection had not yet been established between the server room and the classrooms where the users are.

Completion Date	Activity
30 March <i>System Release 1</i>	<ul style="list-style-type: none"> • Completed the development of a VoIP application with a four digit dialing code system that ran without a server and deployed it at Raglan Road Pre School • Added a Speed Dial function through the telephone address books
October <i>System Release 2</i>	<ul style="list-style-type: none"> • Added an Asterisk server. • Added to the Dialplan so that predefined numbers can be associated with reading a book to the user with the capability of navigating backwards and forwards between pages, or repeating a page. • Added voice-prompt and response facilities • Introduced simple arithmetic games and counting games which run over the telephone handset and require keypad input on the telephone. • Added upload facilities to the application so as to enable the future content providers to be able to create new content, upload it and have the content available on the system automatically without the input of the programmer. • Made a small release within the Computer Science Department and on my desk to test the system before committing to a final release at Raglan Road Pre School in the future

1.5 A Guide to the Remainder of the Report.

The rest of this report will contain a Literature review on this project, a discussion of how the VoIP system was developed, how services and functions can be added to Asterisk, how the Asterisk Dialplan is structured and can be developed, what difficulties were encountered. At the end of this report there is an appendix that contains the complete User Stories used to develop this system, a directory of the currently available services on the system, some of the code added to Asterisk to make the system functional and the poster for this project.

1.6. Chapter Summary

This chapter provided a brief introduction into the nature of the BingBee Phone Network, its aims and its purposes. This chapter also outlined the various stages that were gone through during the development of this system and the specific deliverables that were produced at each stage and at the end of the chapter a brief guide to what the rest of this report contains was provided.

CHAPTER 2: Literature Review

2.1. Introduction

The BingBee umbrella project is an edutainment project designed to increase literacy and numeracy skills in young children. The aim of this investigative sub-project is to develop a VoIP technology system to add to this BingBee. This will be done by developing services in Asterisk. The following review summarises some literature on VoIP systems, Asterisk which is the core system in this project, storytelling, edutainment and the usage of telephones for edutainment.

2.2. VoIP

M.Chetty, E.Blake and E.McPhie [2006] define VoIP as referring “to a range of protocols designed to send voice over packet switched networks, traditionally the domain of internet traffic.” In VoIP voice is sampled at a certain frequency which can be set to any desired value on the devices in use. The sampled voice is then digitised and then finally packaged into packets before being sent over the IP network. VoIP uses different protocols for the call setup and the actual conversation between two communicating telephones. Signalling protocols like SIP, H.323 and IAX are used for the call setup and then the Real Time Transmission Protocol (RTP) is used to carry the voice between the telephones. RTP has been specifically optimised to for the transmission of real time data. About the idea of creating additional services in VoIP systems which this project tested, J.Hitchcock [2006] in his paper entitled *Decorating Asterisk: Experiments in Service Creation for a Multi-Protocol Telephony Environment Using Open Source Tools* wrote that “Voice over IP (VoIP) is no longer a single service application, but an array of marketable services of increasing depth, which are moving into the non-desktop market.”

M.Chetty et al [2006] also offer a couple of advantages that VoIP offers over the Traditional Public Switched Telephone Network (PSTN). These advantages include the fact that VoIP makes more efficient use of bandwidth by only transmitting when something useful is being sent. This is done by silence compression which cuts out the bits when there is silence in a conversation thereby minimising the packets sent across a network. Also VoIP unlike the traditional PSTN does not require a dedicated link between two end points since it uses packet switching instead of circuit switching for communication, and this allows the network to be used for multiple conversations concurrently. The other advantage of VoIP that M.Chetty et al [2006] also mention is that VoIP obviates the need to separate data and voice streams as will be necessary in traditional telephony

since both types of packets can be carried on the same line at the same time. One of the major disadvantages of VoIP however is the difficulty to guarantee a certain level of quality of service. M.Chetty et al [2006] mention that this is because voice quality in VoIP networks varies with the audio or video codec that is being used and also such factors such as latency, packet loss and jitter may also degrade the overall voice or video quality of a conversation. To try and increase the voice quality in VoIP networks M.Chetty et al mention that in networks where the same network is also used for data transport it is necessary to give the VoIP packets a higher priority over the data packets in order to combat the deterioration in quality of service as the network gets congested. This issue on sound quality relates strongly to this project since the main function of the system, the Book Readout Function, is sound dependant and will be affected strongly with the sound quality that the VoIP network in use can deliver.

W.Yu, S.Chellappan and D.Xuan [2005] wrote a paper on VoIP entitled *P2P/Grid-based overlay architecture to support VoIP services in large-scale IP networks* which suggests using a P2P grid for a VoIP network instead of the traditional client/server model that is currently widely used. W.Yu et al [2005] give a detailed suggestion of how to implement a P2P network layout for a VoIP network but however do not give any comparisons between the P2P network layout and the client/server layout. W.Yu et al [2005] do however mention that experimental results demonstrate that the P2P + Hierarchy model for conferencing applications can achieve better performance than all other VoIP models in terms of minimizing the network bandwidth overhead. This certainly suggests an angle to the deployment of the system being developed in this project should the deployment become large scale in the future.

In *A Generic API for Interoperation between Heterogeneous Overlays for Peer to Peer SIP* by M.Tsietsi, G.Wells and A.Terzoli [2007], the authors describe an approach for inter operability between heterogeneous peer to peer overlays for serverless SIP. This paper also mentions how SIP has risen in popularity in recent years and how peer to peer networks are essential for supporting distributed services. SIP or “Session Initiation Protocol is an application-layer control (signalling) protocol for creating, modifying, and terminating sessions with one or more participants. It can be used to create two-party, multiparty, or multicast sessions that include Internet telephone calls, multimedia distribution, and multimedia conferences. (cit. RFC 3261). SIP is designed to be independent of the underlying transport layer; it can run on TCP, UDP, or SCTP. VoIP- Voice over Internet Protocol also can be described as telephony over a computer network”, [Wikipedia 2007]. The VoIP network that was developed in this project also uses SIP as its signalling protocol.

This project is about adding a service to a VoIP network. In *VoiceXML dialog system of the multimodal IP-Telephony—The application for voice ordering service*, M. Stai [2005] says voice

quality and the integration between voice and data in recent years has increased considerably and that has led to value added telephony techniques which have to some degree alleviated the dependence on the telephone to provide a universal platform for multimodal telephony applications. M. Stai [2005] also goes to mention how IP telephony was started as just a way to cut costs involved in calling expenses but in recent years has grown influential in the Internet Telephony industry. M. Stai [2005] goes further to state that various kinds of new applications have been developed and further new applications for internet telephony keep being introduced everyday with examples like e-commerce applications, messaging applications and e-learning applications to name a few. To add to this list, this project developed an edutainment application for a VoIP network.

This project is going to be deployed on a high bandwidth LAN so the efficiencies of the protocols used in VoIP networks are thus not particularly an issue: the range and flexibility of providing services is what is more important to us. The high bandwidth availability in our LAN also suggests that with suitably equipped telephones we could choose high-quality codecs, capable of playing high quality audio with no adverse effects on system and network performance. The limited range of the first deployment of the project which was just on a LAN also reduces the priority of the other issues that are encountered in VoIP networks like NAT and security problems in this project.

2.3. Asterisk

Asterisk is a Public Branch Exchange (PBX) software which consists of a PBX, a SIP Proxy, a built in Interactive Voice Response (IVR) menu and a server that supports the four major VoIP protocols that are currently in use around the world namely SIP, MGCP, H.323 and IAX. For large VoIP networks, Asterisk runs with Sip Express Router (SER) which is a SIP proxy that can support large numbers of clients on behalf of Asterisk. Asterisk is a complete PBX in software which is both standards based and Open Source. J.Hitchcock [2006] also mentions the ILanga PBX Proxy VoIP service which integrates different technologies seamlessly and was developed at Rhodes University to be a complete cost effective computer based PBX [J.Hitchcock 2006].

PBX or Private Branch eXchange is a telephone exchange that serves a particular business or office, as opposed to one that a common carrier or telephone company operates for many businesses or for the general public [Wikipedia 2007].

J.Hitchcock [2006] writes that Asterisk allows for easy service creation in VoIP systems because it allows various telephony protocols to communicate in a transparent manner and also Asterisk acts as a middleware between services and telephony technologies which makes Asterisk the ideal platform for developing services for a converged VoIP network. Asterisk at the moment has the

potential to be expanded to do much more than the current subset of telephony applications that is being used for. Asterisk currently shows the same potential that the PC showed in the early eighties and went on to dominate the computer market[M .Spencer , 2004] so Asterisk has the potential to be used in a wide variety of services. From this paper and other literature surrounding Asterisk cited in this review, Asterisk seems like it is a very good candidate to be the service provider for this project and all the services that will need to be developed for this system. Successful development of all the edutainment services that were planned fro this project has proved the suitability of the use of Asterisk as the PBX for the development of edutainment services for a VoIP network. All the required services were developed through the Asterisk Dialplan.

There are four APIs in Asterisk namely, The Application API, the Codec Translation API, the File Format API and the Channel API [J.Hitchcock. 2006]. The Channel API is used for the interfacing of the Asterisk PBX with the various telephony technologies that Asterisk operates with while the File format API is for handling and interfacing with the various file formats that are used in Asterisk. In this project, we are using the File format API to handle the various wav files that are needed for some of the edutainment services. These various file formats include the sound files that might be played in various telephony menus and the power of Asterisk is that any new file formats can be easily added to the File Format API by creating and adding the relevant module. The Codec Translation API works in a similar way to the File Format API but is responsible for encoding and decoding the audio streams that pass between conversations though the Asterisk PBX. [J.Hitchcock. 2006]. This API architecture is what has been increasing the popularity of Asterisk since it means that the modules within Asterisk are separated form each other and can be modified separately with no need to tweak the whole system.

Asterisk can be extended in different ways. This can be done though the Dialplan which is a file which contains scripting commands of how Asterisk should handle calls. Another way to extend Asterisk is through external interfaces which allow other applications to connect to Asterisk and these external interfaces are the Manager API and the Command Line Interface. One of the most powerful ways to extend asterisk and create custom services is through the Asterisk Gateway Interface (AGI). AGIs can be called from within the Dialplan to instruct Asterisk to execute an external executable which can be written any of a number of languages that Asterisk supports which include Java, C, C++, Perl and Python to name some of them. It should also be noted that support for a new language which Asterisk does not currently support can also be provided through the AGI.

2.4. Storytelling and Edutainment

In *Storytelling based Edutainment Applications* by S Göbel, O Schneider, R Wichert, A Hoffmann, J Dechau, U Lohde, I Iurgel and A Feix [2003], the authors say, “Content, which is presented in an exciting and diversified way, is indispensable to bind users to a system”. S Göbel *et al* [2003] also go on to mention that one of the challenges of edutainment today is how to apply novel storytelling technologies without overstraining ordinary users. This point is very relevant to this project so ensuring that technical issues, system complexity and system handling are not be too difficult is very important since this system is intended for a very young age group whose technical expertise is very limited. On the topic of edutainment in the field of Mobile Edutainment, S Göbel *et al* [2003] go on to mention that when a storytelling component is introduced, it “helps to include a story as a medium to create curiosity and motivation and to convey the teaching material”. From this we can adapt the concept of storytelling in Mobile Edutainment to apply it to this project since this concept is inline with the project’s goal of providing a system which provides edutainment across a VoIP system. Adding a story telling component to the VoIP system as was done in the paper by S Göbel *et al* [2003] to Mobile Edutainment will help to turn plain telephony in this project into more than just a simple voice service.

In a paper entitled *Technology And Grease: A Formula For Edutainment And Efficiency*, D Landt, S Knazze, P Sud [2001] quote the visionary R.W. Oliver [1998] as having once said that “the age we are moving into will be characterized by development of intellectual capital—increasingly through “edutainment,” or entertaining education. This is a cooperative process”. This paper highlights what this project is trying to do in stating that in this postinformation age, edutainment is necessary to motivate and engage students to work towards their futures. The paper by D Landt *et al* [2001] is written around how community colleges in the United States are planning and using edutainment in their classes to provide effective learning techniques and this ties in with the aim of this project to teach young children valuable skills like basic counting and reading skills when they read along with the system’s book readout functionality.

2.5. Telephones and Edutainment

Although at the moment there is a lack of literature on the use of desktop telephone handsets for edutainment, an article that appeared in an online magazine called eContent Magazine in May 2005 written by P.A Salz[2005] entitled “*Random House Gets VOICEL: Invests in Edutainment*”, mentions that although most traditional publishing houses have been out of the mobile service industry, this is about to change because of the improvement in mobile telephone technology and also because of the large profits made from mobile content delivery by other industries such as the

music and news industries. The publication houses are going to enter this mobile market by providing edutainment across mobile telephones. This article by P.A Salz [2005] goes on to mention that VOCEL, a company in the United States that was bought out by a publication house called Random House Inc, and is in the business of being “a provider of branded applications to mobile telephones.”, has already developed content offerings which “enable students to practice drills in math, reading, and grammar via mobile telephones.” Although the BingBee VoIP project is being run on a landline telephone network with desktop telephone handsets as end points, the same type of content that is being offered by this company in the United States can also be offered on the BingBee VoIP Telephone Network. This article also goes on to mention that VOCEL's technology uses interactive messages to provide these edutainment contents to the users. We too aim to provide interaction, but using an IVR menu instead of interactive messages. IVR or Interactive Voice Response is a telephone technology that allows a computer to detect voice and touch tones using a normal telephone call. The IVR system can respond with pre-recorded or dynamically generated audio to further direct callers on how to proceed. IVR systems can be used to control almost any function where the interface can be broken down into a series of simple menu choices, [Wikipedia 2007].

2.6. Conclusion

VoIP has become common, and telephone handset prices have fallen rapidly, and we see rapid addition of new services and applications to its capabilities. Asterisk has established a niche for itself in the VoIP industry and has shown incredible flexibility in how it can be used as the base application in the creation of new services to VoIP networks. Because of this, the VoIP network is a good option to provide edutainment functions to the target end users of this system as content can be simplified enough for young children. Also the literature on edutainment in telephones is mostly available for edutainment over mobile telephones but because of the related cost, offering this edutainment functionality on a telephone network in Africa becomes practical if land line telephones are used in a closed classroom or school situation, since they are cheaper, comparatively easier to physically secure than mobile telephones, and can better withstand the damage the telephones are likely to suffer from the young children that this project is aimed at providing and edutainment service for. The services that have been developed in this project also run satisfactorily on desktop IP telephones so this shows the practicability of the idea of running an edutainment service on desktop telephones.

2.7. Chapter Summary

This chapter has been about a literature review on the various topics that affected this project. The chapter discussed the current view on the technology that is being used in this project from the point of view of the authors that wrote the various papers in these various fields and how Asterisk is a suitable PBX to develop edutainment VoIP networks. This chapter also discussed the literature on the various findings that were made on the topics of VoIP, Asterisk, storytelling in edutainment, edutainment and the use of telephones for edutainment purposes and how these topics relate to this project.

CHAPTER 3: Design and Implementation

Asterisk is the main component of this project so implementation of the functions that the system has to be able to carry out was done by adding scripts to the Asterisk Dialplan. This chapter explains where to obtain Asterisk, how to install it and the basic activities that should be carried out to get an Asterisk server running. This chapter also explains the main components of the Asterisk Dialplan in very low-level detail and the likely problems that are likely to be encountered in setting up an Asterisk server.

3.1. *Installing Asterisk*

Installing Asterisk on an open source operating system requires that the operating system have a GCC compiler of Version 3.x or later with all its dependencies installed. The latest version of Asterisk can be obtained for download from this web site: <http://www.asterisk.org/>.

After downloading the compressed file to the computer, it must then be extracted to a folder. To install Asterisk, start a command line terminal with root rights and change the directory in the terminal to the one where the uncompressed Asterisk installation files are. After this, the next step is to run the command “*make*” in the terminal. This command will built Asterisk from the source files.

If the “*make*” command executes successfully and runs to completion without any errors, the next command that will install the built files is the “*make install*” command which will install Asterisk onto the system. It is also recommended on the first time installation of Asterisk to run the “*make samples*” command next which will add default configuration files required for the operation of Asterisk into the Asterisk folders. All these instructions and other information on the installation of Asterisk can also be found in the “*README*” file which can be found in the folder that the Asterisk source files were extracted to.

Installing Asterisk on a full installation of Fedora 6 or 7 completes with no problems.

3.2. *The Asterisk Dialplan*

The Dialplan in Asterisk is “the master plan of control or execution flow for all of its operations. It controls how incoming and outgoing calls are handled and routed. This is where you configure the behaviour of all connections through your PBX.”[M. Spencer, M. Allison, C. Rhodes. 2003]. The

Dialplan is defined in the `extensions.conf` file which can be found in the Asterisk folder, usually in this folder `“/etc/asterisk”` although this might differ depending on how Asterisk was installed. The Asterisk Dialplan is made of 4 main parts which are *contexts*, *extensions*, *priorities*, and *applications*. These parts all work together to form a fully functional Dialplan.

3.2.1. Contexts

Contexts are used to define scope and are used to keep different parts of the Dialplan from interacting and separate in order to increase a systems security and also to limit certain users to only a limited set of applications that they are allowed to access on the system. This makes sure that when a telephone dials into a specific context, then only the actions defined in that context will be executed. This is handy in situations where dialling a specific default extension number in deferent areas of for example a company, is supposed to direct the caller to a specific service which might differ with departments. An example of this would be at a university where dialling “22” has been defined in the Dialplan to direct a call to that particular department’s secretary, so dialling “22” in the Physics department would dial the Physics department’s secretary while dialling “22” from the Computer Science department would dial the Computer Science department’s secretary. Contexts are also used to create auto-attendants to direct callers to choose extension numbers from a list of available numbers. A context is defined by putting its name in [] brackets which effectively defines where the context starts for example *[firstContext]*.

A context ends where the next context is defined. If however there are default actions which must be uniformly executed irrespective of the context from which a call originates, then these actions must be defined in the *[globals]* context. The globals context defines actions that will be executed across all contexts in a Dialplan.

3.2.2. Extensions

To try and avoid ambiguity, “exten” or “extens” shall be used to refer to the Dialplan component and “extension number” or “extension numbers” shall be used to refer to numbers dialled on the telephones. . Extens here refer to "matching rules" in the Dialplan. They need not necessarily coincide with a service’s number in the system or physical telephone extension numbers connected to the PBX.

The next layer in the Dialplan is the “extensions” layer .The logic elements that are provided for extens in the Dialplan have enough programming power to provide for sequencing, tests, selection,

repetition and iteration. The elements of the extens that provide for these programming concepts are explained below.

One or more extens can be defined inside a single context. Extens are used to define the way in which a call flows and how Asterisk will handle that call after it has been triggered by an incoming call or by certain digits that have been typed on the connected telephone.

The extens can be used to define, for example, the length for which a call can ring unanswered before it is redirected to voice mail and it is also using an exten that the voicemail address for an extension number is defined. Extens in Asterisk are defined in the following format

exten => name, priority, application ()

- ***exten =>*** is a keyword, indicating that this line of the Dialplan adds an extra matching rule about extens. In its most simple case, an exten rule could be triggered by matching a single digit entered on the telephone. The consequence of triggering the rule is to execute the application, which could dispatch control of the flow of the call to a new context, or to a specific service, such as an IVR service.
- ***name*** refers to a particular pattern for a specific exten. This can be either a number or an alpha name or an alphanumeric name. “Dialplan exten names can be simple numbers like "412" or "0". They can be alphanumeric names like "john" or "A93*". Although a typical telephone can't dial an extension number called "john" (some can though), often your Dialplan logic will involve jumping from one exten to a different exten, and for those jumps you may define exten names with any name you like, as you don't wish them to be dialled directly.” [Asterisk Wiki]. The following is an extract which shows what characters can be used for pattern matching [Asterisk Wiki],

Table 1: Characters that can be used for pattern matching in the Asterisk Dialplan

X	matches any digit from 0-9 This feature was used in this project in <i>_1XXX</i> and <i>_3XX</i> . <i>_1XXX</i> was used to put into the Dialplan the logic that any number that starts with 1 and has four digits will be used to dial another telephone. <i>_3XX</i> was used for the Book Readout Function to put into the Dialplan the logic that any number that starts with 3 and is made up of three digits will be directed to the Book Readout Function.
Z	matches any digit from 1-9
N	matches any digit from 2-9
[1237-9]	matches any digit or letter in the brackets, (in this example, 1, 2, 3, 7, 8, 9)
.	wildcard, matches one or more characters
!	wildcard, matches zero or more characters immediately

- **priority** A single extension number can have multiple commands that are to be executed in steps when a call is placed to it. **priority** specifies the order in which multiple commands will be executed and **priority** is also useful when specific flow control is required. In terms of conventional programming languages, the priority allows one to control the sequence in which multiple matching rules are executed.
- **Application ()** this is the specific application or command that Asterisk will execute on the call when this priority of the matched exten is triggered. It is almost like a method call in conventional programming languages

3.2.3. Priorities

As mentioned above priorities specify the order in which multiple matching exten rules will be executed by Asterisk. This can be used to provide loops or conditional jumps in the logic that handles the call. A lower priority number indicates a higher priority as Asterisk will execute the lower priority numbers first in a given exten. An example of this is illustrated in Listing 1. An exten commands Asterisk to ring John’s office for an incoming call to his telephone number. This exten has priority “1”. If the call is unanswered after twenty seconds the next exten redirects the call to John’s home number. This will have priority “2”. If the call is still unanswered after a further twenty seconds, the next exten will redirect the call to the voicemail box for the 1012 telephone

number. This exten has priority of “3”. This means that when an incoming call for the Johns’ extension number 1012 is unanswered, Asterisk will first ring John’s office telephone on 1012. If that telephone is not answered within twenty seconds Asterisk will then redirect the call and ring John’s home telephone number, 2340. If the extension number 2340 is also not answered, Asterisk will then redirect the call to the voicemail box assigned for the office extension number, 1012.

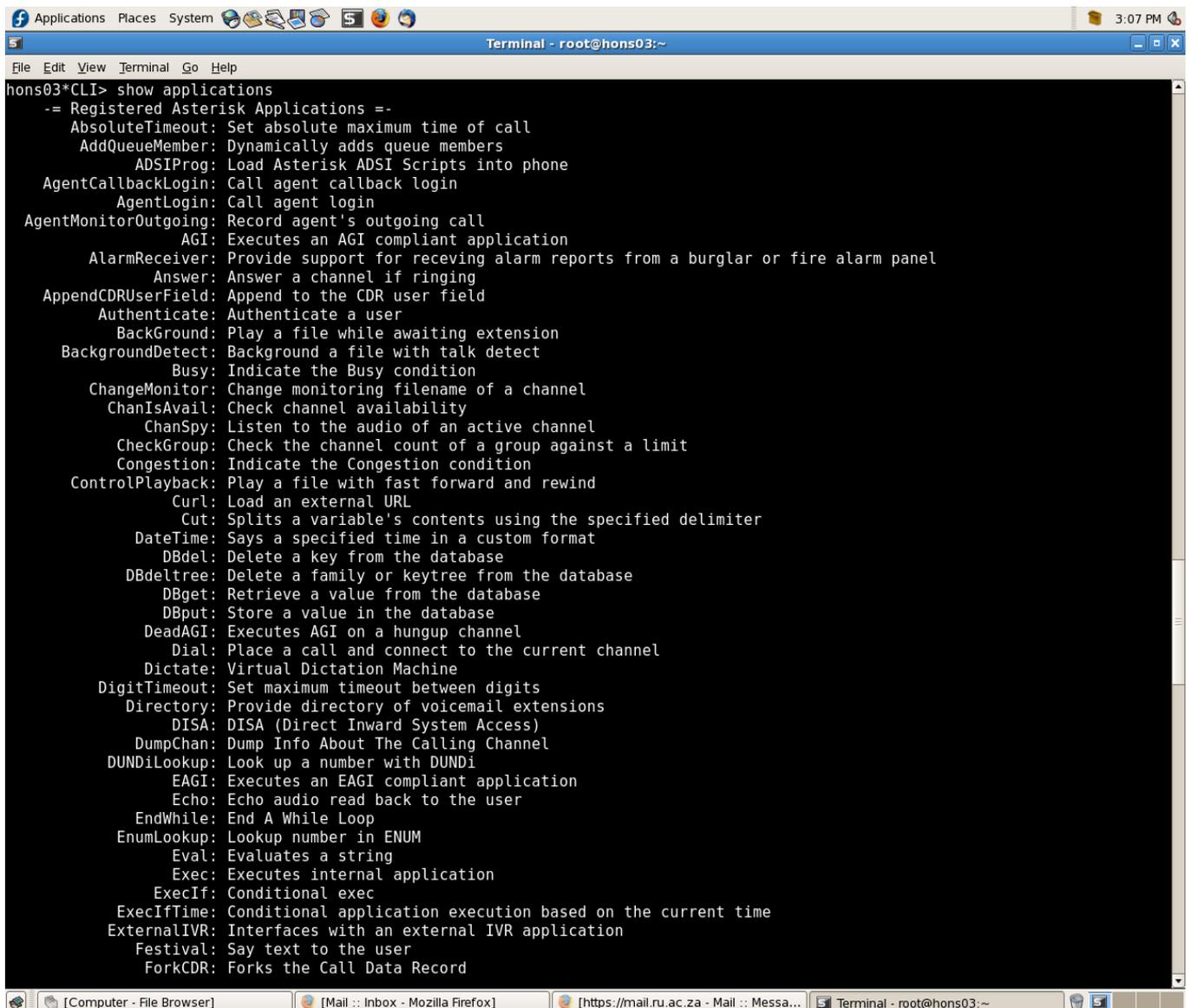
```
[ calls]
exten => 1012, 1, Answer ()
exten => 1012, 2, Dial (SIP/${EXTEN}, 20)
exten => 1012, 2, Dial (SIP/2340, 20)
exten => 1012, 3, Voicemail (1012)
```

Listing 1: Example context to illustrate priorities

3.2.4. Applications

As mentioned above applications are the commands that Asterisk will execute on the call when an exten rule is triggered. Executing the “*core show applications*” command on the Asterisk CLI produces a list of the built-in applications/commands that are available in Asterisk. CLI or command line interface is a method of interacting with an operating system or software using a command line interpreter. The CLI allows the user to interact with Asterisk. The user is capable of running commands or inquiring the status of the running Asterisk application using the CLI. The CLI is also very useful to debug faulty networks because Asterisk prints out error messages onto the CLI. The CLI also allows a user to control an Asterisk server remotely.

Figure 1 below shows a screenshot of some of the available applications in Asterisk and a brief explanation of what each application does.

A screenshot of a Linux terminal window titled "Terminal - root@hons03:~". The terminal shows the command "hons03*CLI> show applications" and its output, which is a list of Asterisk applications and their descriptions. The applications listed include AbsoluteTimeout, AddQueueMember, ADSTProg, AgentCallbackLogin, AgentLogin, AgentMonitorOutgoing, AlarmReceiver, Answer, AppendCDRUserField, Authenticate, Background, BackgroundDetect, Busy, ChangeMonitor, ChanIsAvail, ChanSpy, CheckGroup, Congestion, ControlPlayback, Curl, Cut, DateTime, DBdel, DBdeltree, DBget, DBput, DeadAGI, Dial, Dictate, DigitTimeout, Directory, DISA, DumpChan, DUNDiLookup, EAGI, Echo, EndWhile, EnumLookup, Eval, Exec, ExecIf, ExecIfTime, ExternalIVR, Festival, and ForkCDR. The terminal window is part of a desktop environment with a taskbar at the bottom showing icons for a file browser, Mozilla Firefox, and a mail client.

```
hons03*CLI> show applications
-- Registered Asterisk Applications --
  AbsoluteTimeout: Set absolute maximum time of call
  AddQueueMember: Dynamically adds queue members
  ADSTProg: Load Asterisk ADSI Scripts into phone
  AgentCallbackLogin: Call agent callback login
  AgentLogin: Call agent login
  AgentMonitorOutgoing: Record agent's outgoing call
  AlarmReceiver: Provide support for receiving alarm reports from a burglar or fire alarm panel
  Answer: Answer a channel if ringing
  AppendCDRUserField: Append to the CDR user field
  Authenticate: Authenticate a user
  Background: Play a file while awaiting extension
  BackgroundDetect: Background a file with talk detect
  Busy: Indicate the Busy condition
  ChangeMonitor: Change monitoring filename of a channel
  ChanIsAvail: Check channel availability
  ChanSpy: Listen to the audio of an active channel
  CheckGroup: Check the channel count of a group against a limit
  Congestion: Indicate the Congestion condition
  ControlPlayback: Play a file with fast forward and rewind
  Curl: Load an external URL
  Cut: Splits a variable's contents using the specified delimiter
  DateTime: Says a specified time in a custom format
  DBdel: Delete a key from the database
  DBdeltree: Delete a family or keytree from the database
  DBget: Retrieve a value from the database
  DBput: Store a value in the database
  DeadAGI: Executes AGI on a hungup channel
  Dial: Place a call and connect to the current channel
  Dictate: Virtual Dictation Machine
  DigitTimeout: Set maximum timeout between digits
  Directory: Provide directory of voicemail extensions
  DISA: DISA (Direct Inward System Access)
  DumpChan: Dump Info About The Calling Channel
  DUNDiLookup: Look up a number with DUNDi
  EAGI: Executes an EAGI compliant application
  Echo: Echo audio read back to the user
  EndWhile: End A While Loop
  EnumLookup: Lookup number in ENUM
  Eval: Evaluates a string
  Exec: Executes internal application
  ExecIf: Conditional exec
  ExecIfTime: Conditional application execution based on the current time
  ExternalIVR: Interfaces with an external IVR application
  Festival: Say text to the user
  ForkCDR: Forks the Call Data Record
```

Figure 1: A screenshot of the Asterisk CLI when "core show applications" is run

A good practice is to invoke the *Answer* () application first in a context. The *Answer* () application answers the call and then sets up the line which facilitates the operation of other additional applications that might be employed in the call.

The listing below is a context example of what might be in the Dialplan.

```
[calls]
exten => _1XXX, 1, Answer ()
exten => _1XXX, 2, Dial (SIP/${EXTEN}, 20)
exten => _1XXX, 3, Hangup()
```

Listing 2: Example context

The above context example also illustrates the exten pattern matching feature in the Dialplan.

`_1XXX` means that this particular pattern will be triggered by matching any 4-digit inbound number beginning with the digit '1'. In this example there are three rules that will be executed sequentially as dictated by the priorities, first the call is answered, then Asterisk will dial the SIP telephone with the dialled extension number for twenty seconds, and then if the call is not answered, Asterisk will then hang up the line.

3.3. Difficulties Encountered

The main problem with the telephone network was securing the reliability of the telephone logons to Asterisk. After searching for this problem on the Asterisk wiki, there was a suggestion to set the default IP addresses of each telephone in the *sip.conf* file of Asterisk for each of the telephones that will be in the network. This suggestion did not however solve the problem since there were still instances when some of the telephones failed to log on and also some of the call sessions had no audio.

Research into the known causes for this audio problem in Asterisk online gave suggestions that a NAT server operating on the same network as the telephones has been known to cause the problems being encountered. Although the proposed single site network layout for this project is on a LAN, we were contemplating running the BingBee Phone Network as a remote service to the Pre School and other possible future locations from a server based on the university. Since this will require traffic to be sent across the NAT server between the university network and the outside network, research into the workings of NAT and how NAT might affect SIP and RTP protocols was done to ensure that this was not the cause to the problem at that time and also so as to get ideas of how to tackle the problem when the system will be deployed as a remote service.

3.3.1. NAT and Asterisk

NAT stands for Network Address Translation. NAT is also known as Network Masquerading, Native Address Translation or IP Masquerading .NAT was originally developed to combat the shortage of IP addresses since more IP addresses can be created using NAT for organisations in their private network than they can actually register on the internet. This means that one address on the internet can be used to represent multiple computers inside a private network (usually a LAN). NAT changed the once simple nature of the internet from a simple peer to peer network to a network where separate closed networks with multiple hosts can be created and still be able to connect to the internet [Wikipedia 2007].

What happens in NAT is that source and destination IP addresses are rewritten as they pass through a NAT server or device such as a NAT router. The NAT server replaces the source address with its own address before sending the packet onto the internet and makes an entry in its translation table of the address of the device making the requested connection to the internet. When the NAT server receives a response from the internet it then searches its translation table for the address of the device that made the original connection requests and then rewrites the destination IP address to the address of that device before sending the packet on the LAN.

The reasons why NAT is a problem to VoIP networks is that conventional VoIP protocols are designed to route the audio protocols and the signalling protocols separately. The signalling protocol passes signalling information between the two endpoints such as the ringing of the telephones, information on whether or not the call is answered, and the codec capabilities of the endpoints. Part of this exchange involves randomly selected RTP ports numbers for establishing the separate audio channel. “The NAT router may be able to handle the signalling traffic, but it has no way of knowing that the audio traffic is related to the signalling and should hence be passed to the same device the signalling traffic is passed to. As a result, the audio traffic is not translated properly between the address spaces.”[Asterisk Wiki 2007] .There is a number of competing workarounds that all involve putting extra intelligence into the NAT gateway, and performing some extra endpoint-to-gateway signalling.

Asterisk can also be configured to either ignore or allow for NAT using the “*nat=*” setting in the sip.conf file. There is also support in Asterisk to specify values like the address of the NAT server and also values of for how long a connection to the outside network can be kept alive when idle. One of the settings found in the sip.conf file, the “*canreinvite=yes*” setting, does however cause problems with NAT. If this setting is set to “yes”, this allows the phones to bypass the Asterisk server and route audio streams directly between themselves once the call has been established which causes problems if the Asterisk server in the middle was handling the NAT issues. The full nature of this setting will be discussed under the asterisk Configurations subheading.

It should however be noted that IAX was designed to bundle signalling and the audio stream into a single end-to-end connection, so IAX is inherently more NAT friendly than the other VoIP protocols. IAX is the Inter-Asterisk eXchange protocol native to Asterisk PBXs and is supported by a number of other softswitches and PBXs. IAX is used to enable VoIP connections between servers as well as client-server communication. IAX now most commonly refers to IAX2, the second version of the IAX protocol [Wikipedia 2007]. IAX is very useful when a VoIP network is to be run over a NAT enabled network since it was specifically designed for this purpose.

In order to make sure that NAT was not the cause behind the telephone logon problem and the problem of audio being unreliable and inconsistently available; the following experiment was carried out. First, the telephones' network settings were changed from DHCP to static and then the telephone IP addresses were hardwired. This was also done on the computer that was running Asterisk. A small network comprising a switch as the central device was then constructed which had three telephones and the Asterisk server as the only network connections. After this small network had been constructed and run, the same log on problems were encountered so this was a clear indication that the problem was not being caused by NAT at that time.

3.3.2. Asterisk Configurations

The *sip.conf* file contains a lot of configurations that can be changed to tweak the behaviour of Asterisk. One of these configurations that the Asterisk user manual mentions is the “*host=*” setting. Researching on the host keyword on the Asterisk wiki and in the commented out code that comes preinstalled when installing Asterisk turned up the following,

-“host” is the domain or host name for the SIP server, [Asterisk Wiki 2007]

This setting is used to configure how user accounts log onto the Asterisk server. The primary mechanism for logging on is the user code and password, but Asterisk can also perform some additional security checks to restrict which devices are permitted to supply the credentials. If the configuration is set to “*host = dynamic*”, then an endpoint host has to register, or “log on” to the account to use the server services, using the user code and user password credentials to register the device, this setting however gives the user some flexibility since they can log on from any IP address or host name.

If the configuration “*host = IP Address*” is used, then a particular user account is only allowed to log in from one specific IP Address only. Trying to log onto the Asterisk server from the wrong IP Address even though the right user credentials are supplied will not succeed.

Less restrictive than the “*host = IP Address*” is the “*host=hostname*” setting, which uses DNS to get a device's IP address from the given hostname. This allows authentication from mobile or nomadic devices whose IP number may change depending on where they are. The configuration which sets “*host = dynamic*” is however the one which is most recommended since it gives the system the most flexibility than the other settings.

To see if the telephone logon problem with Asterisk was related to the lack of audio in the telephone calls, research was done into the various Asterisk configurations that can be made to resolve the lack of audio problem. The audio problem at this point was very specific in that it was always the caller telephone which did not get audio from the telephone it had just called. As mentioned under the NAT and Asterisk subheading earlier, information on Asterisk turned up the fact that the “*canreinvite=*” setting was responsible for determining how Asterisk will route audio between two end points. One of the two options of how audio can be routed is that the RTP packets that carry the voice streams between two telephones can be routed through Asterisk. This means that a voice stream will be sent from the sending telephone to Asterisk and then Asterisk will then forward this voice stream to the other telephone that is taking part in the conversation, this option is exercised in Asterisk by this setting “*canreinvite = no*”.

The other option on how audio can be routed in a call is to have the telephones reinvite each other directly once a call session has been established to cut the Asterisk server out of the call stream. This is enabled by doing the following setting “*canreinvite=yes*”. This means that once the call has been established by the SIP signals between Asterisk and the telephones, the telephones will request the IP address of the telephone they are communicating with from Asterisk and once this has happened the RTP packets that carry the voice streams will be routed directly between the telephones without having to first go through Asterisk like when the setting is set to “*canreinvite = no*”. The “*canreinvite=no*” setting is recommended in networks which are not standardised, that is when telephones from different vendors are operating in the same network or the telephones have different configurations. The reason for this is that the telephones might be using different audio compression codecs and if this is so they will not be able to communicate directly with each other. Forcing the audio streams to go through Asterisk will however resolve this potential problem since when each telephone registers with Asterisk it specifies the audio compression codec it is currently using. If telephones using different codecs are communicating, Asterisk will then do the conversion between the different codecs transparently and enable the two telephones to communicate with each other which would otherwise be impossible if the “*canreinvite=yes*” setting was being used

After experimenting with this “*canreinvite=*” setting for a few minutes, the results obtained showed that when this setting is set to “*canreinvite=no*”, which forces that RTP streams to go through Asterisk, there was no audio at all in the telephone calls although the telephones were able to ring each other. This showed that the SIP packets were being sent and received but the RTP packets were being lost, dropped or blocked at some point in the network. Since there were times when audio was present in the call sessions when the “*canreinvite=*” setting was set to

“*canreinvite=yes*”, this led to the realisation that the problem affecting the RTP audio streams was being caused by the Asterisk server.

Possible causes to this problem that were suggested by internet forums indicated that a wrong configuration in Asterisk might be the cause of this problem so at this point some of the possible solutions to this problem were to go through the Asterisk configuration files that had been changed after installing Asterisk to check if any necessary configurations had been overlooked or done incorrectly. Another proposed solution was uninstalling and then reinstalling Asterisk again in case Asterisk had not installed correctly the previous time when it had been compiled from source code and installed.

3.3.3. Server Firewall

Professor Peter Wentworth who is supervising this project however suggested that a better approach to this problem would be to isolate the telephone network again to reduce network traffic and then use a packet sniffer to analyse the packets that are being sent between the telephones and the Asterisk server to see if this might give an indicator as to the root cause behind these problems in the VoIP telephone network. The packet sniffer that was then used for this exercise was Wireshark.

A typical VoIP session using SIP should produce the following SIP packets on the network; INVITE, TRYING, RINGING, OK, ACK, as illustrated in the diagram below.

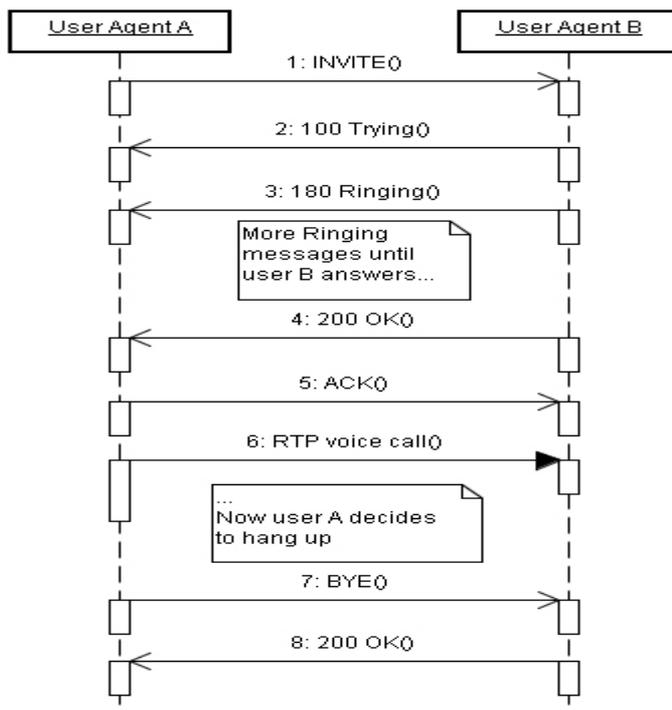


Figure 2: SIP session with DNS servers [16]

While using Wireshark, all the expected SIP packets were present in the call sessions and were being sent as theoretically specified by the SIP protocol description for SIP telephone calls.

After monitoring the network using Wireshark when the “*canreinvite=*” setting is set to “*canreinvite=no*”, which forces the RTP streams to go through Asterisk, it was apparent that the calling telephone was sending the RTP packets to Asterisk but Asterisk was not forwarding them to the telephone being called. This trend suggested that this was the root of the problems that were being experienced in the telephone network and that solving this would make a significant difference to the current progress in the project. After going through the lecture slides from the RTMM course which describes the flow of events between the caller, callee and Asterisk, it was apparent that all the required packets for a successful call session were being generated by the telephones but the server was not behaving correctly. At this stage the packet sniffer was configured to filter just SIP and RTP packets for display so to enable further analysis this filter was removed to capture all the traffic in the small VoIP network.

Removing the filter showed some ICMP packets with “Host unreachable” messages that were being sent back to the telephones every time they tried sending an RTP packet. This led to the conclusion that the problem was firewall related since the host was definitely in the network and up. The security settings of the Fedora firewall seemed to be blocking off the RTP packets. RTP uses a dynamic port range of 16384-32767 which makes it particularly difficult to pass through a firewall. After setting the Fedora firewall to off, all the previous problems that had been on the VoIP network were resolved and the network was now reliable for the telephone logons and all the call sessions had audio.

Setting the Fedora firewall to off is a temporary workaround: one would need to turn the firewall feature back on and refine the settings in a production environment. However, in the context of our "closed world" school scenario in which the network is not connected to the internet and unauthorised users have no access to the server, and given that our key objective is to validate the idea that telephones could be used by young children for edutainment, pursuing details of firewall settings was considered beyond the scope of the current work.

The next issue was finding the right audio codec to use to convert the audio files of the books to. After reading the Asterisk wiki on the sound file topic, the wav file type is the best choice. The wav files have to be however converted to a specific configuration for them to be played with Asterisk without having to install any additional modules. The wav files have to be of a frequency of 8 kHz, 16 bit sampling rate and be mono channel. Changing these settings might increase the sound quality

but will result in Asterisk being unable to play the sound files with the default codecs. Installing additional codecs is possible but complicated so to ensure the wav files play correctly they have to be converted to the properties shown in Figure 3 below, or poor sound quality or very low volume output will be experienced.

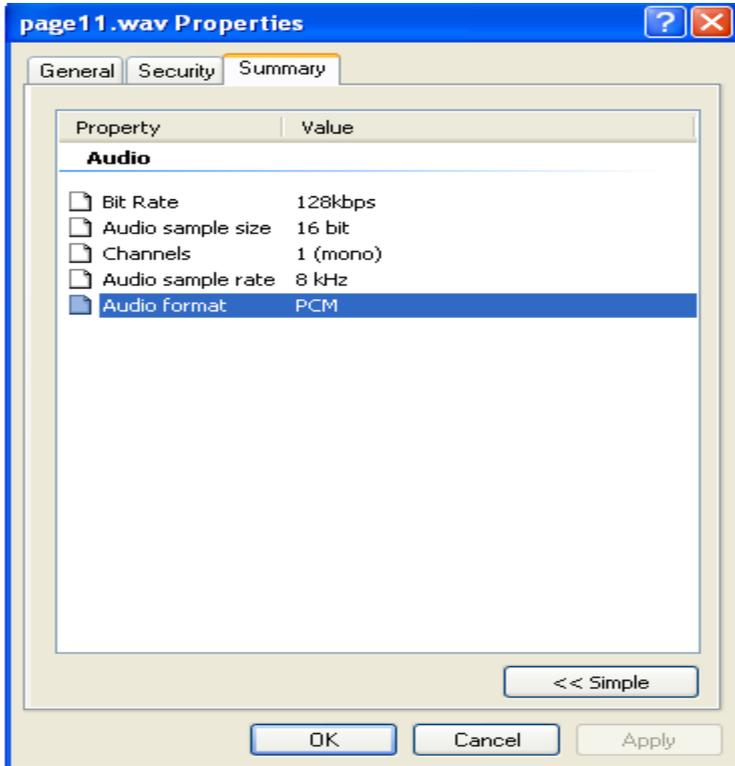


Figure 3: Wav file properties to be used for the Book Readout Function

The ControlPlayback method will be used to playback the audio files for the book readouts since out of all the Asterisk audio playback methods it offers the best level of user control on the file playback. The ControlPlayback method offers fast forwarding and rewinding a preset number of milliseconds, pausing playback, stopping playback and skipping to the next file.

3.4. Chapter Summary

This chapter was reported the specific activities that were carried out in the implementation of this project. It laid out the installation, development and configuration of Asterisk which is the backbone of this system. This chapter also gave a brief overview of some of the more serious difficulties that were encountered at this stage in the development of the VoIP system, specifically the problems related to Asterisk that were encountered and the various proposed solutions to these problems, the main one being the interaction of RTP packets with the operating system firewall.

CHAPTER 4: Results and Evaluation after each release

Having completed the installation and configurations highlighted in the previous chapter, this chapter follows on to describe the development activities that followed in the development of the VoIP network on Asterisk to test the idea that edutainment services can be offered over a VoIP network and lists first, the User Stories that were developed and then the functions that were developed on the system based on these User Stories for each release.

4.1. System Release 1

4.1.1. User Stories for System Release 1

Since the BingBee Telephone Network is being developed using the Extreme Programming methodology, each system release's functionality is determined by the user stories that are agreed on at the start of each stage. The user stories for System Release one are listed below

1. Users can dial another telephone if they know it's hard wired IP address
2. Users can dial another telephone in a different place by the Speed Dial facility using a double digit number and then pressing the "*Speed*" key to dial any of the numbers in the telephone address book.
3. Users can clearly hear each other over the telephones(satisfactory voice quality and clarity)

4.1.2. Developed Functions

The first release of the BingBee telephone network was done on the 2nd of April 2007. This system release involved the wiring of Ethernet cables around the classroom at Raglan Road Pre School and installation of telephone handsets daisy chained together by the laid out Ethernet cables .

The aim of this system release was to deploy a VoIP network with just the traditional telephony functionality. This was provided by the inbuilt functionality that comes with the IP telephones. The telephone features that were used for this system release were the two Ethernet ports that are on each telephone which allow one telephone to be daisy-chained to the next. The other feature that these IP telephones have is a feature which allows the user to dial another telephone using the other telephone's IP address. So the network for System release 1 was constructed by daisy chaining the telephones together with Ethernet cables. To get the telephones to be able to dial each other, first, the telephones must have IP addresses assigned to them, by DHCP if it is available. If the

telephones are going to be on a network without a DHCP server which will be the case in this project, then the IP address for each telephone will have to be hard wired on each telephone.

The telephones' IP addresses can be changed in two ways. If the telephones are set to DHCP, then changing the telephone's internal settings can be done using the telephone's web interface. This can be done by using the telephone's IP address as an address on a web browser which will bring up a log on page to log into the telephone's internal settings. The other option to change the Ethernet telephone's internal settings is by directly logging onto the telephones themselves and changing the settings there. Logging onto the telephones directly can be done by pressing the keys "**1234**" and the pressing the "**#**" key. At this stage the telephone will prompt the user for a password which by default is set to "**1234**". After this the "**Speaker**" key must then be pressed and this will then log the user into the telephones internal settings. Navigation between the different telephone internal settings can then be done using the "**VOL+**" and "**VOL-**" keys. To change a setting the user must press the "**SET**" key and then the "**OK**" key to accept any changes. After that the user must press the "**EXIT**" key and then the "**OK**" key a last time on which the telephone will reboot and boot up with the new settings affected.

For the BingBee Telephone Network, the IP addresses for the telephones which were deployed at Raglan Road Pre School were changed using the second way, by using the telephone's web interface which brings a screen like the one shown below in Figure 4 which facilitates the set up of the telephones internal settings.

network settings					
iptype	static	ppp id		ppp pin	
local ip	192.168.0.11	subnet mask		router ip	
dns		dns2		mac	00-09-45-63-83-b1
protocol settings					
protocol	sip	use service	<input type="checkbox"/>	service id	
service type	common	service addr		nat ttl	30
nat traversal	disable	nat addr		pin	6759
phone number	7673	account	7673	control port	6802
register port	5060	signal port	5060	rtp port	6802
register ttl	60	rtp tos	0	dtmf	rfc 2833
local type	phonenumber	call type	advanced		
phone settings					
use dialplan	disable	dial number		ddcode	10
idcode	86	iddprefix	00	dddprefix	0
inertline	disable	local prefix	0	nonlocal prefix	0
answer	30	ring type	dtmf7	use digitmap	<input type="checkbox"/>
forward number		fwd poweroff	<input type="checkbox"/>	fwd noanswer	<input type="checkbox"/>
fwd always	<input type="checkbox"/>	fwd busy	<input type="checkbox"/>	call waiting	<input type="checkbox"/>
audio settings					
audio type	g711a	audio frames	2	g.723.1 high rate	<input checked="" type="checkbox"/>
vad	<input checked="" type="checkbox"/>	agc	<input type="checkbox"/>	aec	<input checked="" type="checkbox"/>
handset in	11	handset out	20	speaker out	29
other settings					
password	1234	super password	12345678	debug	output
sntp ip	210.59.157.10	use daylight	<input type="checkbox"/>	upgrade addr	
timezone	(GMT+02:00)Harare				
Save/Reboot					

Figure 4: Telephone Settings Web Interface

One of the major functions which are in-built in the telephones which allowed for the deployment of the telephone network without a server to be easier is the address book. Since this telephone network is intended for young children, dialling using the telephone IP addresses is just too complicated for children of this age. The address book allows for the implementation of the speed dial facility where the user can call another telephone using only a two digit speed dial number. For this to work, the IP numbers of the telephones have to be added to the address book of the telephones in the relative numbers that will be used to dial them as illustrated in Figure 5 below.

Phone Book					
No.	Name	Phone Number	No.	Name	Phone Number
001			002		
003			004		
005			006		
007			008		
009			010		
011	Phone 11	192.168.0.11	012	Phone 12	192.168.0.12
013	Phone 13	192.168.0.13	014		
015			016		
017			018		
019			020		
021			022		
023			024		
025			026		
027			028		
029			030		
031			032		

Figure 5: Telephone Address Book.

After these entries have been added to the telephone address book, the user can then dial the other telephone by pressing the unique number they require to contact and then pressing the “*Speed*” button on the telephone to dial the other telephone.

Since this two-digit speed-dial mechanism is extensible for other services which will be provided by Asterisk, we envisaged that after the Asterisk server was integrated into the network, two-digit speed-dialling would still be the norm, whether for direct calls or for other services like book readouts and counting. To make this speed dial extension, when the Asterisk server was incorporated to the network, the various telephone numbers and extension numbers for the various services that were also added to the telephone Address books in the manner shown above in Figure 5. It should however be noted that the telephone address books are a static feature so all the additional material that will be added after deployment by the various content providers will not be available through the speed dial feature unless the administrator later on logs on again into the telephones’ internal system to add the new speed dial numbers for the new material to the telephone internal address books.

Since the first release was going to be purely as a telephone system without any additional services, experimentation with the Windows version of Asterisk was done in parallel with developing the BingBee Telephone network to run without a server since security of the server at the premises was still an issue and also factoring in the cost of the server at that stage when an Asterisk server was not really necessary led to the decision of deploying Release One of the BingBee Telephone network without a server.

Experimentation with the Windows version of Asterisk was later dropped after finding out that support for this Windows version was not as readily available on the internet as was for the open source version. Also, no one in the Computer Science Department here at Rhodes University had ever tried it to this angle was left as a possible later further task for this project. Also the only professional support I could find was from a company called WillVoice which charges for its services while professional help for the open source version of Asterisk was readily available within the department.

Because of this, in order to minimise the cost of the BingBee telephone system and in light also that there are a lot of unconventional services which might not be supported by the company making the Windows version of Asterisk that were developed in the BingBee Phone Network, the open source version of Asterisk which runs under open source operating systems was finalised as the application of choice to develop the final BingBee Phone Network system since it is fully flexible in terms of

that any functionality can be added to it and also it is freely available and there are a number of people in the Rhodes University Computer Science department who have extensive experience with this version of Asterisk so support for it will be readily available in case of unforeseen problems.

4.1.3. Difficulties Encountered

A number of problems were encountered in the set up of the telephone network for release one, the main one being that for a time only one of the telephones out of the total of three could make calls to the other telephones. To determine the real problem causing this, an experiment was devised to explore the various causes that could have been leading to this problem.

The experiment in question involved setting up a small network comprising of a hub, the three telephones in the network and a computer all connected to the hub. The purpose of this experiment was to find out the difference in network traffic being generated by the telephone that was operating as desired and the two others which were not operating correctly. To do this Wireshark was used to inspect the packets in the network when dialling the different telephones through the hub which all these devices were connected to.

However after consulting Mosiuoa Tsietsi who is a Masters student working on a VOIP system and has some experience with the VOIP handsets being used for this project, it was discovered that the checkbox on the telephone which allows for the configuration of the telephones to use a server or not, was not fully functional and was the root of the problem being faced. To operate the telephones without a server, this checkbox must be unchecked but it turned out that this was not enough, the server addresses in the fields below this checkbox for all the server settings that should have been deactivated by un-checking the “***Server Service***” checkbox also had to be cleared, of which initially this had been assumed as unnecessary after unselecting the “***Server Service***” checkbox. After clearing the fields in question on all the telephones the network started functioning correctly. Another problem that led to all this difficulty is that the telephone user manual which comes with the telephones does not include this information at all.

Another solution to the dialling problem that had been put into consideration was to download a newer version of the telephones’ firmware and update the telephones’ firmware. Research into this solution in the department however revealed that doing this has been known to make the telephones unstable in the past causing them to crash unpredictably so this option was discarded without a trail and was also made unnecessary when the real cause of the telephone network problem was discovered.

4.1.3. System Evaluation and User Feedback

After a week of the first release of the BingBee network, users reported that the telephone network was unreliable especially when trying to connect a call between the telephones that are on either side of the central telephone which also acts as a hub for the network. Connecting these two outer telephones is reported as being opportunistic and erratic at best and also it has been reported that making such a connection takes numerous tries of dialling to go through.

In order to establish as to whether this fault is as a result of the middle telephone and whether its replacement will solve the problem or not, further experimentation was done on the system to determine whether the problem could be solved with just the current hardware using a different set of three telephones or whether additional hardware like a switch was going to have to be purchased. This experiment was really important in the respect that it had to be determined when the BingBee Phone System comprised of only three telephones before it was expanded to a bigger network whether the telephones will be able to cope with the additional strain that will be added with the increased number of telephones. The same system was also going to be tested with all the telephones connected to a central switch so as to establish if there was a difference in reliability to the telephone network when there was switch and when the telephones were just daisy chained together. This issue was not resolved but since this daisy-chain network was going to be replaced by a server network I decided to test again after the main release is done to see if the problem recurs.

4.2. System Release 2

The proposed logical network layout for a single site for this system release is shown in Figure 6 below:

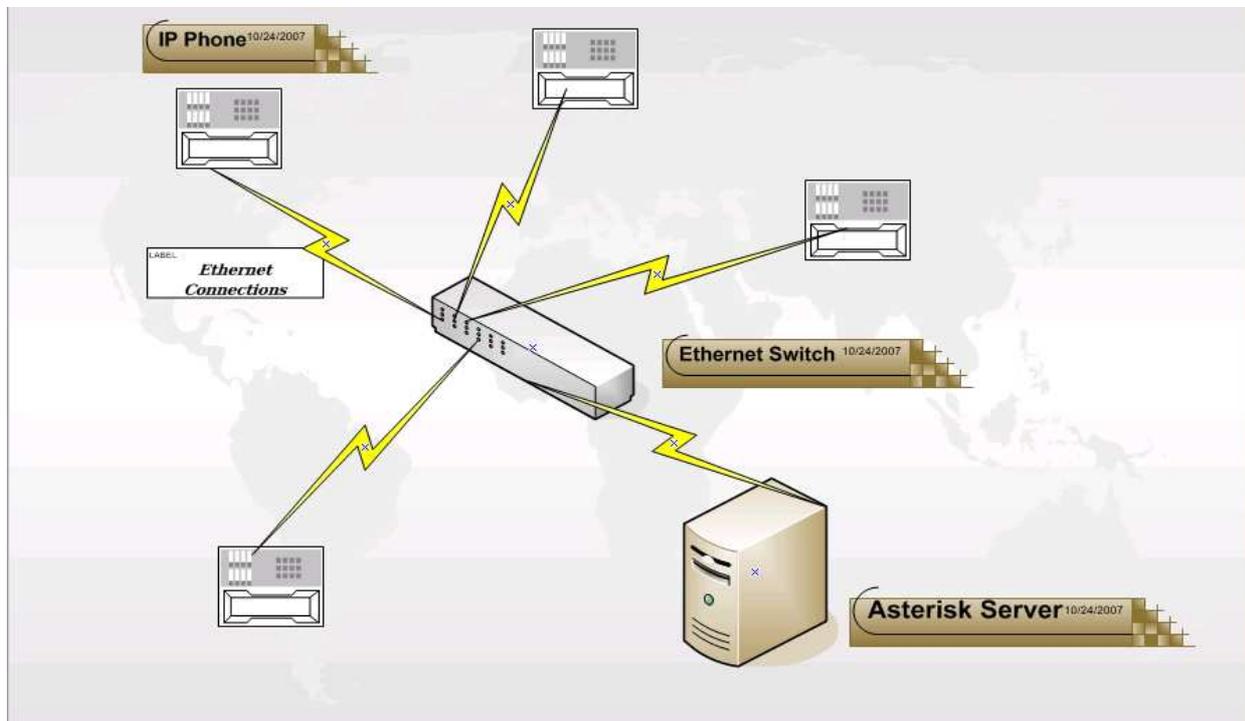


Figure 6: Proposed Single Site Network Layout

4.2.1. User Stories for System Release 2

1. Users can dial a number on the front of a book, and the telephone will play back the book to the user page by page (system is capable of navigation between the pages if so desired).
2. Users dial an extension number. After that every time the user presses a key the telephone will tell the user which number they have dialled in either English, Xhosa, or both the languages depending on which extension number or Speed Dial the user pressed.
3. Users can dial a pre assigned number or Speed Dial for either a Division, Multiplication, Addition or Subtraction calculation. Telephone will prompt the user for the two numbers to calculate, separated by pressing the asterisk "*" key. The user will then be told the answer to the mathematical problem by the telephone.
4. Users dial an extension number. After the user is connected the system will start counting from the user from one going upwards.

4.2.2 Developed Functions

4.2.2.1. Call Function

The first issue that had to be addressed for System Release 2 was the migration of the VoIP network from the daisy chained network which was done in System Release 1 to a server centralised network using Asterisk as the VoIP PBX.

4.2.2.2. Book Readout Function

The main function developed for the system for System Release 2 was the Book Readout Function. This function allows for the upload of books in audio format that can be read back to the user by dialling a specific number for each different book.

To make this function as dynamic as possible, additional effort was made to develop the ordinarily static Asterisk Dialplan to cater for additional content that will be made in the future. Ordinarily the Dialplan requires that each page in each book be added to the Dialplan manually for it to be available on the system and so each book would be required to have a separately coded extension number in the Dialplan to handle readout requests for the book.

But since the Dialplan logic elements have enough programming power to dynamically test for the existence of files in the file store, the Dialplan was extended to be dynamic and allow for the addition of new content to the system without the need for a programmer to tweak the Dialplan every time there is need to add new content for the Book Readout Function. By associating each book with a folder which is named a number between 300 and 399 which must not already be in use for another book, we've managed to use wildcard matching and tests for files to be able to dynamically add new books by simply dropping their audio tracks into a folder numbered with the extension number that will designate the book. In this way our effective call routing is now dynamic, driven by the existence or otherwise of data on the file store.

Each audio file which represents a page in the folder must be named in the following format “*page\$*”, where the \$ sign represents the page number. The page numbering must start from “*page0*” and then go upwards. The Book Readout Function is configured in such a way that it will automatically playback upon request any book added in this manner and added to the folder on the Asterisk server in a folder with this address “*/var/lib/asterisk/books*”. The system is also configured to automatically pick up any varying number of pages that each book might contain. As an example, to add a new book for this functionality, the audio files for the content can be uploaded to the server in a folder which is named “322”, and this number must not already be in use for another book. Each audio file which represents a page in the folder must also be named in format explained above. The system is also configured to automatically pick up any varying

number of pages that any book might contain and to give a polite error message should a user dial an extension number in the range 300 to 399 to which content has not yet been assigned to. The Book Readout Function also provides to the user the options of, fast forwarding, rewinding, repeating, skipping or pausing page playback. This is accomplished by using the “**ControlPlayback ()**” function in Asterisk which provides this functionality. It should also be noted that the Book Readout Function can be used for more than just books; we are now able to add to the system audio tracks that teach counting or any sing-along songs, e.g. songs to teach the alphabet or tell stories.

This function is implemented in the “**book**” context in the Dialplan.

4.2.2.3. Say Number Function

The next functionality that was developed for System Release2 was a number readout function to help the young children learn how to count from zero to nine by pressing the appropriate key on the telephone handset. This function is provided in three different ways on three different extension numbers.

If the user dials “22”, the user will be taken to an extension number which provides for the playback of the number that the user has just pressed in English. This means that for example, if a user presses the number three key, then the telephone handset will say to the user “**three**” in the English language. This function is implemented in the “**numbers**” context in the Dialplan.

The next way the Say Number Function is implemented is to readout the number of the key the user has just pressed in Xhosa. If the user dials “23”, the user will be taken to an extension number which provides for the playback of the number that the user has just pressed in Xhosa. This means that for example, if the user presses the number three key, then the telephone handset will say to the user “**ntathu**” in the Xhosa language. This function is implemented in the “**xhosa**” context in the Dialplan.

The third way that the Say Number Function is implemented is so as to help the young children learn to count in both English and Xhosa at the same time and also to help them learn what each number is in both languages. If the user dials “24”, the user will be taken to an extension number which provides for the playback of the number that the user has just pressed in English and then Xhosa. This means that for example, if a user presses the number eight key, the telephone handset will say to the user “**eight**” in the English language and then say “**bhozo**” again to the user in Xhosa language. This function is implemented in the “**both**” context in the Dialplan.

4.2.2.4. Calculator Function

The next function that was developed for the system for release two is the calculator function. This function allows for the carrying out of division, multiplication, addition and subtraction calculation functions using the telephone handsets as calculators. A user can dial “26” to do division calculations, “27” to do multiplication calculations, “28” to do addition calculations or “29” to do subtraction calculations. All the calculations must be done with a maximum of two numbers at a time and also the two numbers must be separated by the “*” key. The telephone will prompt the user to enter two numbers separated by an asterisk and then tell the user the answer to the mathematical problem after a 3 second timeout.

This function is implemented in the “*division*”, “*multiplication*”, “*addition*” and “*subtraction*” contexts in the Dialplan.

4.2.3. Difficulties Encountered

One of the problems I encountered was the difficulty I had in sourcing Xhosa readers to translate some of the material we had in English into Xhosa. Getting somebody to translate a whole book from English to Xhosa was very difficult but I did manage to get a friend to do the translation for the Say Number Function. The other problems I encountered at this stage are fully explained in Chapter 3 like the problem I had with the server firewall.

4.2.3. System Evaluation and User Feedback

After testing the system myself for sometime I later on moved the service onto a Fedora 7 virtual machine running on one of the Computer Science department’s servers, Chameleon. The system proved to be very stable as no problems up to this stage have yet popped up but we will be waiting on user feedback when the system is finally released on the Pre School to the final users.

4.3. Chapter Summary

This chapter listed the various functions that were added to the system for each system release and also stated which contexts the functions were developed in. The chapter also listed some user feedback that was received after each release and also stated some of the main difficulties that were

encountered during the development of the system for each of these two releases.

CHAPTER 5: Further Work and Conclusions

In conclusion, what this project has done is to show that we are already in a position to treat handsets as viable "educational toys" in the classroom. Although the current trend in the industry right now is to offer edutainment through cell phones, this project has however proved that this edutainment service can also be successfully offered over traditional desktop telephones. The current development and improvement in desktop IP telephone handsets also points to these telephone handsets becoming cheaper and more sophisticated in the near future and there are some predictions that the telephone will be the future terminal of choice for accessing rich web content. This project has also successfully shown that Asterisk is an appropriate and practical PBX to undertake such a project.

A possible extension to this project will be to add functionality to the VoIP network so as to exploit the small screen on the VoIP telephones in the activities that the users will be engaged in. There currently are some VoIP telephones that have bigger colour screens than the ones being currently used in this project. Using these telephones will enable the addition of further functions and edutainment games with visual components that can be played on the telephones.

Since there are already examples of edutainment services being offered on mobile telephones, enabling the BingBee VoIP content to be offered on a mobile telephone network can also be a possible future extension to this project.

Future work can also be done on this project to integrate a soft telephone into the BingBee kiosk network so that BingBee kiosk users can also use the services and functions that the telephone network offers.

Further work on this project can also be done to try and implement the system on a windows platform which is what the BingBee kiosks run on. Doing this would enable the installation of the Asterisk server on one of the BingBee kiosks there negating the need for a separate server and thus reducing costs for the deployment of the BingBee Phone Network.

5.1.Alternative uses

One of the alternative uses that this project could be used for is as a menu reader for a take away restaurant or pizza place. The system will be available for unsure customers who would dial in to the system to hear the menu of what is on offer and navigate between the different pages of what is

on offer at the restaurant. The customer would then press a predefined number which will navigate them out of the menu and to a an assistant to make their order once they have made up their mind about which menu item they want, thereby freeing up the assistants time from explaining the menu to customers to have more time to take final orders.

5.2. Chapter Summary

This chapter gave a conclusion to this report and stated the expected direction that this work is predicting edutainment and telephones to take in the near future. This chapter also talked about further work that can follow the work that was started in this project. Also discussed in this chapter was one possible alternative use that a system like this might be employed to do

References:

[1] *Asterisk For Windows* [Online],

Available: <http://www.en.voipforo.com/asterisk/asterisk-for-windows.php>,

Accessed 23/02/07

[2] *BingBee website* [Online]

Available: <http://bingbee.coe.ru.ac.za/>,

Accessed 01/04/07

[3] D. Comer, Prentice Hall “*Computer Networks and Internets with Internet Applications (fourth edition)*”, New Jersey, [2004]

[4] D.G. Kelly, C. Jennings, L. Dang “*Practical VoIP Using VOCAL* “
O'Reilly [2002]

[5] F. Giunti, “*Edutainment, Technotainment and Culture*” [Online] [2004]

Available: <http://sumscorp.com/articles/pdf/2004%20Edutainment,%20Technotainment%20and%20Culture.pdf>

Accessed: 10/06/07

[6] Figure 2 reference [Online]

Available: <http://blogs.zdnet.com/images/sequence.jpg>

Accessed: 05/10/07

[7] “*How Network Address Translation Works*” [Online]

Available: <http://computer.howstuffworks.com/nat.htm>

Accessed: 25/06/07

[8] J. Hitchcock “*Decorating Asterisk: Experiments in Service Creation for a Multi-Protocol Telephony Environment Using Open Source Tools,*” [Online] [2006],

Available: <http://research.ict.ru.ac.za/list.html>, Rhodes University,

Accessed: 25/06/07

[9] J.McKenzie “*Beyond Edutainment And Technotainment*” [Online] [2003]

Available: www.agp.org

Accessed: 12/06/07

[10] J. Van Meggelen, J. S. and Leif Madsen “*ASTERISK: the future of telephony*”, O’Reilly, USA, [2005].

[11] K.Salah “*On the deployment of VoIP in Ethernet networks: methodology and case study*”

[Online] [2005]

Available: [http://0-www.sciencedirect.com.echea.ru.ac.za/science?_ob=ArticleURL&_udi=B6TYP-](http://0-www.sciencedirect.com.echea.ru.ac.za/science?_ob=ArticleURL&_udi=B6TYP-4GHSDVG-)

[4GHSDVG-1&_user=736737&_coverDate=05%2F15%2F2006&_alid=591733980&_rdoc=52&_fmt=summary&_orig=search&_cdi=5624&_sort=d&_docanchor=&_view=c&_ct=91&_acct=C000040938&_version=1&_urlVersion=0&_userid=736737&_md5=a5a31207e76f834b0129f5daeabf7cde](http://0-www.sciencedirect.com.echea.ru.ac.za/science?_ob=ArticleURL&_udi=B6TYP-4GHSDVG-1&_user=736737&_coverDate=05%2F15%2F2006&_alid=591733980&_rdoc=52&_fmt=summary&_orig=search&_cdi=5624&_sort=d&_docanchor=&_view=c&_ct=91&_acct=C000040938&_version=1&_urlVersion=0&_userid=736737&_md5=a5a31207e76f834b0129f5daeabf7cde)

Accessed: 12/06/07

[12] M.Chetty (a), E.Blake (b) and E.McPhie(c)

“*VoIP deregulation in South Africa: Implications for underserved areas*” [Online] [2006]

^aCollege of Computing, Georgia Institute of Technology, Atlanta, Georgia, USA

^bDepartment of Computer Science, University of Cape Town, South Africa

^cPolicy Associate, Bridges.org, South Africa

Available: http://pubs.cs.uct.ac.za/archive/00000381/01/policy_paper_mchettyf.pdf

Accessed: 10/06/07

[13] M. Addis “*New technologies and cultural consumption – edutainment is born!*” [Online]

[2005]

Available: www.emeraldinsight.com/researchregister

Accessed: 12/06/07

[14] M. Spencer, M. Allison, C. Rhodes “*The Asterisk Handbook (Version 2)*” [2003]

[15] M. Ito “*Engineering Play: Children’s software and the cultural politics of edutainment*”

[Online] [2006] University of Southern California, USA

Available: <http://www.itofisher.com/mito/EngPlay.pdf>

Accessed: 10/06/07

[16] M. Tsai “VoiceXML dialog system of the multimodal IP-Telephony—The application for voice ordering service” [Online] [2005]

Institute of Information Management, National Chiao Tung University [Online]

Available: www.sciencedirect.com

Accessed 15/06/07

[17] M.Tsietsi, G.Wells and A.Terzoli “A Generic API for Interoperation Between Heterogeneous Overlays for Peer to Peer SIP” [Online] [2007]

Department of Computer Science, Rhodes University

Grahamstown, South Africa

Available: <http://www.cs.ru.ac.za/courses/Honours/RTMM/software/index.php>

Accessed 10/06/07

[18] P.A Salz “Random House Gets VOICEL: Invests in Edutainment”, [Online] [2005]

Available: <http://www.econtentmag.com/>

Accessed: 12/10/07

[19] voip-info.org [Online]

Available: <http://www.voip-info.org/>

Accessed: 05/07/07

[20] W.Yu, S.Chellappan and D.Xuan

“P2P/Grid-based overlay architecture to support VoIP services in large-scale IP networks”

[Online] [2005]

Accessed: 12/06/07

Available: <http://www.cse.ohio-state.edu/~chellapp/papers/05-fgcs-ycx.pdf>

[21] Wikipedia the free Encyclopaedia [Online]

Available: http://en.wikipedia.org/wiki/Main_Page

[22] Z. Okan “Edutainment: is learning at risk?” [Online] [2003]

Accessed: 11/06/07

Available: <http://www.ingentaconnect.com/content/bpl/bjet;jsessionid=491etqih1pep.alexandra?>

Appendix

How to Test the System Running on Chameleon

To test the services running on the BingBee Phone System running on *chameleon.ict.ru.ac.za* using one of the preconfigured user accounts, please configure your hand set as explained below.

How to Configure the Telephone Handset

To configure the IP telephone settings, first press the “**LOCIP**” key to get the telephone’s IP Address. After that type the IP Address you get into your web browser. You should be directed to a log in page with the title “welcome to phone settings” where the default password is “1234”. After logging into the settings page, please change the phone settings to read exactly like the ones shown in the screenshot below in Figure 7,

network settings					
iptype	static	ppp id		ppp pin	
local ip	146.231.124.34	subnet mask	255.255.248.0	router ip	146.231.120.1
dns	146.231.129.102	dns2	146.231.129.98	mac	00-09-45-63-83-e6
protocol settings					
protocol	sip	use service	<input checked="" type="checkbox"/>	service id	146.231.121.239
service type	common	service addr	146.231.121.239	nat ttl	30
nat traversal	disable	nat addr		pin	1234
phone number	1600	account	1600	control port	6802
register port	5060	signal port	5060	rtp port	6802
register ttl	120	rtp tos	0	dtmf	rfc 2833
local type	account	call type	advanced		
phone settings					
use dialplan	disable	dial number		ddcode	10
iddcode	86	iddprefix	00	dddprefix	0
innerline	disable	local prefix	0	nonlocal prefix	0
answer	30	ring type	dtmf7	use digitmap	<input type="checkbox"/>
forward number		fwd poweroff	<input type="checkbox"/>	fwd noanswer	<input type="checkbox"/>
fwd always	<input type="checkbox"/>	fwd busy	<input type="checkbox"/>	call waiting	<input type="checkbox"/>
audio settings					
audio type	g711u	audio frames	2	g.723.1 high rate	<input checked="" type="checkbox"/>
vad	<input type="checkbox"/>	agc	<input type="checkbox"/>	aec	<input checked="" type="checkbox"/>
handset in	11	handset out	20	speaker out	17
other settings					
password	1234	super password	12345678	debug	output
snmp ip	210.59.157.10	use daylight	<input type="checkbox"/>	upgrade addr	
timezone	(GMT+02:00)Harare				
<input type="button" value="Save/Reboot"/>					
Address Book					
Update Firmware, Digitmap and Ring					

Figure 7: Telephone internal settings to test the server running on Chameleon

After this press the “**Save/Reboot**” button and the phone should log onto the BingBee Phone Network server.

Directory of the Currently Available Services

<i>Extension Number</i>	<i>Short Description Of Service</i>
300	This extension number uses the Book Readout Function and currently reads to the user a book entitled “ <i>The Pig And The Box</i> ”.
315	This extension number uses the Book Readout Function and currently reads to the user a book entitled “ <i>The Crow Who Could Fly</i> ”.
22	This extension number is designed to teach the user the number they have just pressed in English. E.g. an adult dials 22 for a child, every additional time the child presses a number, the phone will tell the child the number he/she has just pressed in English.
23	This extension number is designed to teach the user the number they have just pressed in Xhosa. E.g. an adult dials 23 for a child, every additional time the child presses a number, the phone will tell the child the number he/she has just pressed in Xhosa.
24	This extension number is designed to teach the user the number they have just pressed in both English and Xhosa. E.g. an adult dials 24 for a child, every additional time the child presses a number, the phone will tell the child the number he/she has just pressed in English and then Xhosa.
25	This extension number is designed to teach the user to count from one going upwards. After the user dials 25, the telephone will start counting to the user from zero going upwards in English.
26	This extension number is designed to use the calculator function. After the user dials 26, they will be taken to a service which prompts the user to enter two numbers to do division calculations after which they will then be told the answer to the division calculation.
27	This extension number is designed to use the calculator function. After the user dials 26, they will be taken to a service which prompts the user to enter two numbers to do multiplication calculations after which they will then be told the answer to the multiplication calculation.
28	This extension number is designed to use the calculator function. After the user dials 26, they will be taken to a service which prompts the user to enter two numbers to do addition calculations after which they will then be told the answer to the addition calculation.
29	This extension number is designed to use the calculator function. After the user dials 26, they will be taken to a service which prompts the user to enter two numbers to do subtraction calculations after which they will then be told the answer to the subtraction calculation.

Complete Project User Stories

1. Users can dial another telephone in a different place by the Speed Dial facility a double digit number and then pressing “*Speed*” to dial any of the numbers in the telephone address book.
2. Users can clearly hear each other over the telephones(satisfactory voice quality and clarity)
3. Users can dial a number on the front of a book, and the telephone will playback the book to the user page by page (system is capable of navigation between the pages if so desired)
4. Users dial an extension number. After that every time the user presses a key the telephone will tell the user which number they have dialled in either English, Xhosa, or both the languages depending on which extension number or Speed Dial the user pressed
5. Users can dial a pre assigned number or Speed Dial for either a Division, Multiplication, Addition or Subtraction calculation. Telephone will prompt the user for the two numbers to calculate, separated by pressing the asterisk “*” key. User will then be told the answer to the mathematical problem by the telephone
6. Users dial an extension number. After the user is connected the system will start counting form the user from one going upwards.

Asterisk Configuration Files

extensions.conf (The Asterisk Dialplan)

```
; The "General" category is for certain variables.
;
[general]

static=yes
writeprotect=no
autofallthrough=no
clearglobalvars=no

priorityjumping=no

userscontext=default
;
; You can include other config files, use the #include command
; (without the ';'). Note that this is different from the "include" command
; that includes contexts within other contexts. The #include command works
; in all asterisk configuration files.
#include "filename.conf"
#include iax.conf

; The "Globals" category contains global variables that can be referenced
; in the DialPlan with the GLOBAL DialPlan function:
; ${GLOBAL(VARIABLE)}
; ${${GLOBAL(VARIABLE)}} or ${text${GLOBAL(VARIABLE)}} or any hybrid
; Unix/Linux environmental variables can be reached with the ENV DialPlan
; function: ${ENV(VARIABLE)}
;
[globals]
CONSOLE=Console/dsp ; Console interface for demo
;CONSOLE=Zap/1
;CONSOLE=Telephone/telephone0
IAXINFO=guest ; IAXtel username/password
SIPINFO=guest
MYPAGE=0
bookNum=0
;IAXINFO=myuser:mypass
```

[calls]

```
exten => 22,1,GoTo(numbers,s,1)
exten => 23,1,GoTo(xhosa,s,1)
exten => 24,1,GoTo(both,s,1)
exten => 25,1,GoTo(loopcount,s,1)
exten => 26,1,GoTo(division,s,1)
exten => 27,1,GoTo(multiplication,s,1)
exten => 28,1,GoTo(addition,s,1)
exten => 29,1,GoTo(subtraction,s,1)
```

;extension which dispatches a call to the books context, saves the extension number as the
;book number first in a global variable.

```
exten => _3XX,1,Set(bookNum=${EXTEN})
exten => _3XX,2,GoTo(book,s,1)
```

;extension to connect telephones to one another

```
exten => _1XXX,1,Ringing()
exten => _1XXX,2,Dial(SIP/${EXTEN},20)
exten => _1XXX,5,Hangup()
```

[book]

;extension which plays back the requested book to the user. Uses the global variable bookNum as
; the requested book to playback

```
exten => s,1,Answer
exten => s,2,Set(MPAGE=0)
exten => s,3,ControlPlayback(/var/lib/asterisk/Books/${bookNum}/page${MPAGE},4000,*,#,1,0,2,,j)
exten => s,4,NoOp()
exten => s,5,ControlPlayback(/var/lib/asterisk/Books/${bookNum}/page${MPAGE},4000,*,#,1,0,2,,j)
exten => s,6,Set(MPAGE=${MPAGE}+1)
exten => s,7,ExecIf(${MPAGE}>100)Hangup()
exten => s,8,GoTo(book,s,5)
exten => s,104,Playback(/var/lib/asterisk/Books/wrongNumber)
exten => s,105,HangUp()
exten => s,106,Playback(/var/lib/asterisk/Books/theEnd)
exten => s,107,HangUp()
```

[numbers]

;extension to say the number of the key which has just been pressed to the user in english

```
exten => s,1,Answer()
exten => 0,1,SayNumber(0,f)
exten => 1,1,SayNumber(1,f)
exten => 2,1,SayNumber(2,f)
exten => 3,1,SayNumber(3,f)
exten => 4,1,SayNumber(4,f)
exten => 5,1,SayNumber(5,f)
exten => 6,1,SayNumber(6,f)
exten => 7,1,SayNumber(7,f)
exten => 8,1,SayNumber(8,f)
exten => 9,1,SayNumber(9,f)
```

[xhosa]

;extension to say the number of the key which has just been pressed to the user in xhosa

```
exten => s,1,Answer()
exten => 0,1,Background(/var/lib/asterisk/Audio/Xhosa/0zero)
exten => 1,1,Background(/var/lib/asterisk/Audio/Xhosa/1one)
exten => 2,1,Background(/var/lib/asterisk/Audio/Xhosa/2two)
exten => 3,1,Background(/var/lib/asterisk/Audio/Xhosa/3three)
exten => 4,1,Background(/var/lib/asterisk/Audio/Xhosa/4four)
exten => 5,1,Background(/var/lib/asterisk/Audio/Xhosa/5five)
exten => 6,1,Background(/var/lib/asterisk/Audio/Xhosa/6six)
exten => 7,1,Background(/var/lib/asterisk/Audio/Xhosa/7seven)
exten => 8,1,Background(/var/lib/asterisk/Audio/Xhosa/8eight)
exten => 9,1,Background(/var/lib/asterisk/Audio/Xhosa/9nine)
```

[both]

;extension to say the number of the key which has just been pressed to the user in both english

; and xhosa

```
exten => s,1,Answer()
exten => 0,1,SayNumber(0,f)
exten => 0,2,Background(/var/lib/asterisk/Audio/Xhosa/0zero)
exten => 1,1,SayNumber(1,f)
exten => 1,2,Background(/var/lib/asterisk/Audio/Xhosa/1one)
exten => 2,1,SayNumber(2,f)
exten => 2,2,Background(/var/lib/asterisk/Audio/Xhosa/2two)
exten => 3,1,SayNumber(3,f)
exten => 3,2,Background(/var/lib/asterisk/Audio/Xhosa/3three)
exten => 4,1,SayNumber(4,f)
exten => 4,2,Background(/var/lib/asterisk/Audio/Xhosa/4four)
```

```

exten => 5,1,SayNumber(5,f)
exten => 5,2,Background(/var/lib/asterisk/Audio/Xhosa/5five)
exten => 6,1,SayNumber(6,f)
exten => 6,2,Background(/var/lib/asterisk/Audio/Xhosa/6six)
exten => 7,1,SayNumber(7,f)
exten => 7,2,Background(/var/lib/asterisk/Audio/Xhosa/7seven)
exten => 8,1,SayNumber(8,f)
exten => 8,2,Background(/var/lib/asterisk/Audio/Xhosa/8eight)
exten => 9,1,SayNumber(9,f)
exten => 9,2,Background(/var/lib/asterisk/Audio/Xhosa/9nine)

```

[loopcount]

;extension to count up to the user from 1 upwards in english

```

exten => s,1,Answer()
exten => s,2,Set(MPAGE=0)
exten => s,3,SayNumber(${MPAGE},f)
exten => s,4,Set(MPAGE=${MPAGE}+1)
exten => s,5,GoTo(loopcount,s,3)

```

[division]

```

exten => s,1,Background(division)
exten => _X.,1,Set(num1=${EXTEN},num2=${EXTEN})
exten => _X.,2,Cut(num1=num1,*,1)
exten => _X.,3,Cut(num2=num2,*,2)
exten => _X.,4,Set(answer=${MATH(${num1}/${num2},int)})
exten => _X.,5,SayNumber(${answer},f)
exten => _X.,6,Goto(division,s,1)

```

[multiplication]

```

exten => s,1,Background(mult)
exten => _X.,1,Set(num1=${EXTEN},num2=${EXTEN})
exten => _X.,2,Cut(num1=num1,*,1)
exten => _X.,3,Cut(num2=num2,*,2)
exten => _X.,4,Set(answer=${MATH(${num1}*${num2},int)})
exten => _X.,5,SayNumber(${answer},f)
exten => _X.,6,Goto(multiplication,s,1)

```

[addition]

```
exten => s,1,Background(add)
exten => _X.,1,Set(num1=${EXTEN},num2=${EXTEN})
exten => _X.,2,Cut(num1=num1,*,1)
exten => _X.,3,Cut(num2=num2,*,2)
exten => _X.,4,Set(answer=${MATH(${num1}+${num2},int)})
exten => _X.,5,SayNumber(${answer},f)
exten => _X.,6,Goto(addition,s,1)
```

[subtraction]

```
exten => s,1,Background(sub)
exten => _X.,1,Set(num1=${EXTEN},num2=${EXTEN})
exten => _X.,2,Cut(num1=num1,*,1)
exten => _X.,3,Cut(num2=num2,*,2)
exten => _X.,4,Set(answer=${MATH(${num1}-${num2},int)})
exten => _X.,5,SayNumber(${answer},f)
exten => _X.,6,Goto(subtraction,s,1)
```

Listing 3: The current Asterisk Dialplan for the BingBee Phone Network

sip.conf

```
;  
; SIP Configuration for Asterisk  
;  
; Syntax for specifying a SIP device in extensions.conf is  
; SIP/devicename where devicename is defined in a section below.  
;;  
; Useful CLI commands to check peers/users:  
; sip show peers          Show all SIP peers (including friends)  
; sip show users          Show all SIP users (including friends)  
; sip show registry       Show status of hosts we register with  
;  
; sip debug               Show all SIP messages  
;  
; reload chan_sip.so      Reload configuration file  
;           Active SIP peers will not be reconfigured  
  
[general]  
context=calls             ; Default context for incoming calls  
bindport=5060             ; UDP Port to bind to (SIP standard port is 5060)  
bindaddr=146.231.123.67  
srvlookup=yes            ; Enable DNS SRV lookups on outbound calls  
  
disallow=all              ; First disallow all codecs  
allow=alaw  
allow=ulaw  
allow=gsm                  ; Allow codecs in order of preference  
  
useragent=Asterisk PBX    ; Allows you to change the user agent string  
  
usereqtelephone = yes     ; If yes, ";user=telephone" is added to uri that  
contains  
                           ; a valid telephone number  
dtmfmode = rfc2833        ; Set default dtmfmode for sending DTMF. Default:  
rfc2833  
  
registertimeout=20        ; retry registration calls every 20 seconds  
(default)  
registerattempts=10       ; Number of registration attempts before we give up
```

```

;
;nat=no                ; Global NAT settings (Affects all peers and users)
                        ; yes = Always ignore info and assume NAT
                        ; no = Use NAT mode only according to RFC3581
                        ; never = Never attempt NAT mode or RFC3581

support
                        ; route = Assume NAT, don't send rport
                        ; (work around more UNIDEN bugs)

[1500]
Type=friend
Insecure=very
Username=1500
Host=dynamic
Mailbox=1500
Context=calls
Secret=1234
canreinvite=no
defaultip=146.231.124.82

;[1600]
;Type=friend
;Insecure=very
;Username=1600
;Host=dynamic
;Mailbox=1600
;Context=calls
;Secret=1234
;canreinvite=no
;defaultip=146.231.124.34

[1700]
Type=friend
Insecure=very
Username=1700
Host=dynamic
Mailbox=1700
Context=calls
Secret=1234
canreinvite=no
defaultip=146.231.124.96

```

Listing 4: The current sip.conf file for the BingBee Phone Network

Project Poster



RHODES UNIVERSITY
Where leaders learn

BingBee Phone Network

A project by Taurai Saurombe: go3s1570@campus.ru.ac.za
Supervisor- Prof Peter Wentworth: p.wentworth@ru.ac.za
Website: < <http://www.cs.ru.ac.za/research/go3s1570/index.htm> >

SPONSORED BY

Aim

The aim of this project is to create a fully functional edutainment VoIP system which can be used to make sure learning can continue in the absence of a teacher in an effort to bridge the digital divide in young children in line with the BingBee edutainment project. The BingBee Phone system is able provide various services varying from simple telephone calls to requests for book readouts, arithmetic games and some simple counting games which are played over the handset through an Interactive Voice Menu.

The aim is to have a system which is interesting enough to attract the attention of the children and then educate them while they play on the system.



Approach

The necessary VoIP functions were implemented on an Asterisk application running under Linux. The following activities were carried out on the system→

- Enhancement of the Asterisk Dialplan so that predefined numbers can be associated with reading a book to the user.
- Addition of page backward and forward navigation, page repetition and skipping capabilities
- Development of simple counting games that can be played through the Interactive Voice Menu facility.

- Development of simple arithmetic games which can also be played through an Interactive Voice Menu facility.



Conclusion

This project has shown that handsets are a viable "educational toy" for the classroom. Although the current trend in the industry is to offer edutainment through cell phones, this project has proved that this edutainment service can also be successfully offered over traditional desktop telephones using a free PBX like Asterisk.



Bright Ideas®
Projects 39

