Context-aware VoIP

Javier Lara Peinado
Víctor Ariño Pérez

October 30, 2011

Contents

List of Figures

List of Tables

Authors

- Victor Ariño Pérez (arino@kth.se)
- Javier Lara Peinado (javierlp@kth.se)

Victor Ariño Pérez was responsible for performing the matlab simulations and the study of CODEC switching based on network resources. He wrote the network-related sections of this paper.

Javier Lara Peinado was responsible for setting up an energy-aware VoIP system and conducting the battery level measurements. He wrote the sections dealing with power saving.

This work was organized as a two-person project, building upon previously published work.

Abstract

This article explores how CODECs are important to VoIP, and how the communications can be improved by dynamically switching CODEC as a function of the network feedback (thanks to RTCP) and as a function of the battery usage of the terminal (laptop, cellphone) running a SIP client. We try to examine and simulate several scenarios to show the benefits of dynamic CODEC switching based upon this two paramters: network delay and battery.

1 Introduction to VoIP

Wikipedia.org [10] describes it as:

> Voice over Internet Protocol (Voice over IP, VoIP) is a family of technologies, methodologies, communication protocols, and transmission techniques for the delivery of voice communications and multimedia sessions over Internet Protocol (IP) networks, such as the Internet.

Unlike the traditional Public Switched Telephone Network (PSTN) in VoIP architecture all data is processed at the endpoints.

This does not mean that servers connected to the network are not necessary, their main roles are now to provide a mapping between phone numbers and IP addresses (or other identifiers). Interested in earning money with this technology, this operator will charge fees based upon a tariff scheme, in order to make a profit on that service provided - hence records of service usage will be collected and used to generate a bill for the customer. For this reason one or more servers will be involved in setting up and tearing down sessions that are used for providing the service(s).

In the traditional PSTN voice was sampled at 8kHz and transmitted at 64 kbps [8] using an 8bit-encoding of the information. As a result of this fixed choice of CODEC each telephone call generated the same amount of information per unit time. In contrast, in VoIP the voice encoding can be different for every call providing higher or lower data rates [17] depending on the user preferences (language, network status, available battery, ...), and thus providing a higher or lower voice quality.

In this paper we will focus on the fact that we can use an adaptative CODEC selection scheme in order to achieve the maximum voice quality and while minimizing/optimizing the network delay and/or the terminal battery. Since the only relevant entities to this scheme are the endpoints, they must be able to switch the CODEC without causing a major inconvenience for the users. We will analise this behavior in the following sections.

2 Voice Encoding: CODECs

An audio CODEC is an algorithm for encoding and decoding audio samples. This algorithm may also include compression and decompression of digital audio data [5]. The name is due to a blend of "coder-decoder". So, depending on the algorithm and the amount of processing time, sample size, etc. a better or worse voice quality can be achieved. In this paper we will consider the following CODECs: G.711, G.722, G.723.1, G.726, G.729, AMR, iLBC and GSM. We will compare them in terms of the network delay and battery power consumption in the following sections.

The main characteristics of CODECs under study are shown in table 1. Based on these chacaracteristics we provide an analysis of the influence of these CODECs on the battery use and on the network traffic congestion.

Table 1 – Characteristics of the VoIP CODECs under study based in [13]

CODEC	**Sampling Rate (kHz)**	**Bitrate (kbps)**	**Encoding algorithm**	**MOS**
G.711	8	64	PCM	4.39
G.722	16	48	SB-ADPCM	3.16
G.723.1	8	6.3	MP-MLQ	3.69
G.726	8	16/24/32/40	ADPCM	3.85
G.729	8	8	CS-ACELP	3.75
AMR (1)	8	12.2	MR-ACELP	3.97
AMR (2)	8	7.4	MR-ACELP	3.71
AMR (3)	8	4.75	MR-ACELP	3.39
iLBC	8	15.2/13.3	iLBC	3.86
GSM 06.10	8	13	RPE-LTP	3.5

3 CODEC selection based on the network conditions

The main criteria for CODEC selection based on the network conditions basically focus on the network performance (in this paper we will focus on delay as this is one of the major factors in the perceived voice quality). VoIP is is a real-time service, so a bounded delay is critical for mantaining an interactive conversation. Xiaokun Yi mentioned in his thesis [27] that there are four main factors that constrain a real-time application:

- **Link usability** involves two issues, one is the presence of the link, and the other is the permission to use this link
- **Available end-to-end bandwidth**: is usually restricted by the network interface, competing traffic and link errors.
- **Packet delay** causes voice quality degradation. ITU G.114 recommends [14] that delay on voice communications should be lower than *150ms* for one-way communication.
- **Packet loss** may happen due to bottlenecks or lack of connectivity. Some CODECs implement a method to hide the loss of packets by masking or re-replaying the last correctly received package.

Thus, following the model suggested by Xiaokun Yi [27] we can draw the voice delay as a function of the network traffic load as shown in figure 1 (This figure was inspired by a figure in [27] page 11).

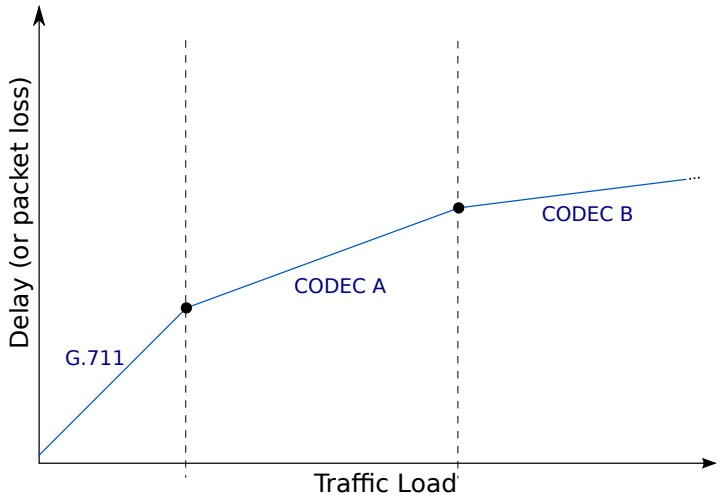

Figure 1 – VoIP communication delay vs network traffic load

In figure 1 switching is represented by dots showing that at a certain threshold (indicated by the dashed lines), switching CODEC can change the rate of the increase in the delay with traffic load. Additionally, the changes in CODEC can cause a loss or increase in perceived voice as a function of the CODEC. This change in perceived voice quality is mostly due to use of CODEC that generates a lower data rate (as seen in table 1), these lower data rate CODECs generally have a lower MOS (Mean Opinion Score) as shown in the table 1.

For instance, if we switch from G.711 (64kbps) to G.723.1 (6.3kbps) the traffic generated by our VoIP conversation is reduced by a factor of nearly 10, but at the cost of reducing the perceived quality from a MOS score of 4.39 to 3.69. Thus a factor of 10 reduction in data rate lead to a huge decrease in perceived voice quality. The ideal order to switch would be to make small changes as shown in figure 2.

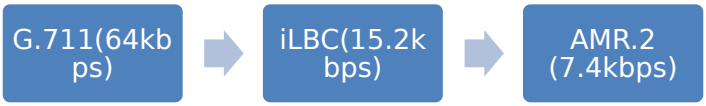

Figure 2 – Possible CODEC switch order

Based upon network feedback, applications could adaptively switch between different CODECs or adjust the parameters of the current CODEC [27].

This behavior can be observed in the state graph (figure 3) where green arrows represent network performance increase (i.e., a decrease in delay) and red arrows the decrease of network performance (i.e., a increase in delay).

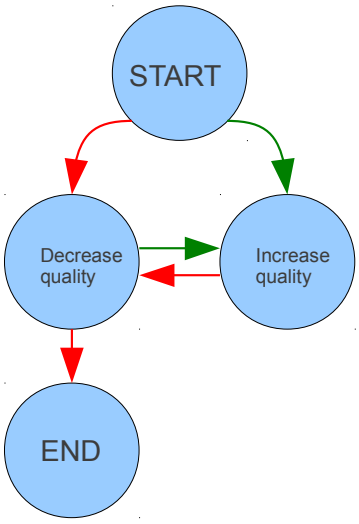

Figure 3 – CODEC switching state machine

So, if the network performance increases a CODEC is chosen which provides high quality while when the network performance decreases a CODECis selected that decreases the data rate - resulting in lower perceived quality.

Another possibility may be to adapt the sample rating, as Xiokun Yi showed that this could improve the overall performance. However, for simplicity in this paper we will mantain the same sampling rate and not change the CODECs parameters, just change between the CODECs listed in the table 1.

Ekiga [4], the VoIP client that we used for the testing of CODEC switching scheme, uses the G.711 CODEC by default for voice encoding/decoding as this CODEC offers a better MOS score at a higher bitrate cost. Next we will present some performance tests in order to compare this default CODEC's behavior to variable network load, then we will compare the diferent CODECs' behaviours and suggest switching schemes.

The reader can appreciate in the figure 4 that the delay increases exponentially with the inverse of the available data rate. This means that for huge traffic loads (low throughput) the communication delay may be over the blue line, which indicates the 150ms maximum delay recommended by G.114 [14]. The upper line of this graphic shows the behavior of G.711 (Ekiga's default CODEC). Although this CODEC provides the highest quality (as shown by its MOS score in Table 1) it also leads to the largest data frames - hence the highest data rate of the CODECs that we consider.

The difficulty consists in defining appropriate thresholds and back-off margins for CODEC-switching based upon difference in the network load. Suppose that we are using a network that is completely dedicated to VoIP traffic, then if all the users switch their CODECs to a lower bit-rate producing CODEC, the average network load will decrease drastically, hence the VoIP clients would all try to upgrade again to a better MOS CODEC (which would produce data at a greater rate) and the network load will raise quickly again - fall again and so on. For this reason we must introduce hysteresis such that a CODEC is used for longer periods of time, and that not all the users change CODEC at the same time. In a stationary state all of the users could converge to use the same CODEC, unless some users were more privileged and allocated larger amounts of bandwidth - which causes the other users to use an even lower data rate CODEC.

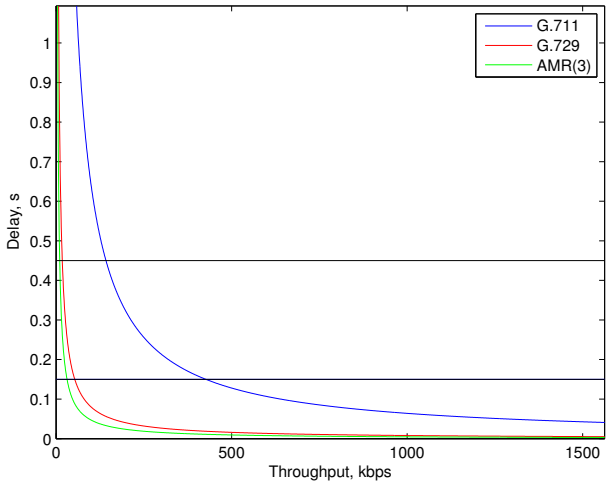

Figure 4 – Delay versus data rate for several different CODECs

To understand the difference in demands between a high-quality CODEC and a bandwidth friendly one, figure 4 shows the relation between maximum throughput and the delay for the AMR(3), G.729 and G.711 CODECs (from left curve to right respectively). The two horitzonal black lines indicate the 150ms and 450ms delay bounds.

As we mentioned before, a CODEC switching scheme based on static thresholds can overreact to slight changes in the network conditions, becoming a new problem instead of a solution. To illustrate this, the graphic in figure 5 shows the results of a Matlab simulation of a marginally unstable network ($\lambda = \mu = 5$) with a varying number of users (shown on the left axis) and the VoIP load they generate on the right. During the first half of the time they all use the G.711 CODEC but they all change at the same moment to iLBC. It is clear that for the first CODEC the tendency of the network load is to increase more and more with the number of users, but, with the change to the second CODEC the the load increases more slowly with the number of users.

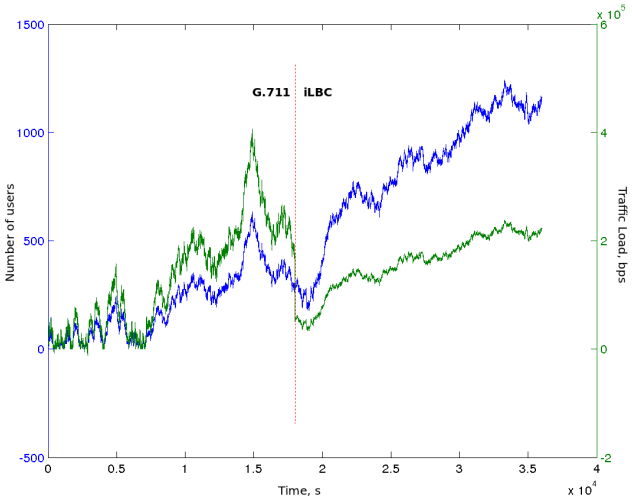

Figure 5 – Number of users (blue) and traffic load (green)

As mentioned previously, if all the users change CODEC at the same time, they may consequently cause a decrease of the network load proportional to the number of users within the network (as can be seen in figure 5). Thus, if the number of users within the network is large enough this reduction may be perceived as a chance to upgrade to a better CODEC again which will cause congestion. As

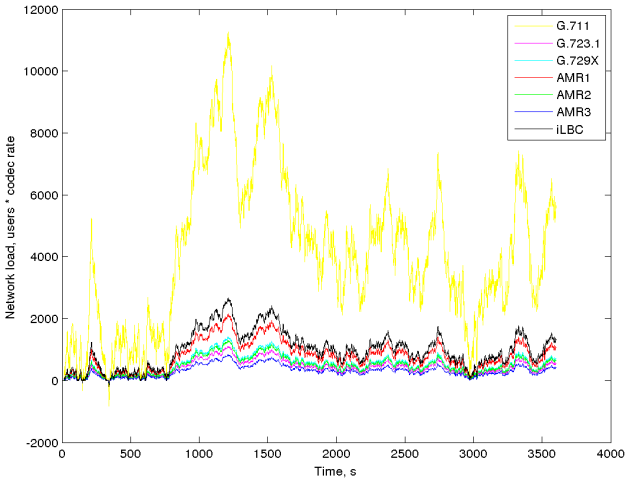

Figure 6 – CODEC Rates for $\lambda = \mu = 5$ Poisson processes

noted earlier this will require adding some hysteresis to prevent oscillation and to avoid pushing the network into a congested state.

The following CODECs have been considered: G.711, G.723.1, G.729X, AMR1, AMR2, AMR3 and iLBC. In the figure 6 can be seen the network load due to the use of each CODEC, in a Poisson user distributed network with $\lambda = \mu = 5$.

3.1 Network Feedback

The key question is "How and when to switch?". This can also be re-phrased as the question: How to tell the other enduser(s) that we are receiving poor voice quality and we would appreciate if they would transmit with a lower quality CODEC? RTCP [23] is used to give feedback (statistics) about the RTP data, its primary function is to provide feedback about de QoS (Quality of Service) to the endpoints. If a VoIP client monitors this information and sees that a peer is getting bad QoS, then it is maybe a good idea to switch CODEC. Please note that, as mentioned above, changing the CODEC is not the only alternative, in some scenarios changing the sampling rate or changing some of the encoding parameters might be a better means to achieve a quality improvement [27].

Taking into account that the network conditions is a *plus-value* feature for a SIP client, Xiaokun Yi pointed out in his thesis the means to do this shoud be un-intrusive (i.e., the number of feedback packets should be much smaller than the number of media packets) and light-weight (the computational effort should not degrade de quality of the service).

While we can use the RTCP to provide information about the state of the network (as it affects the related RTP flow), the question remains: When to decide to switch? Several cases can be considered. The simplest case is to define a single threshold, when this threshold is exceeded we switch to the CODEC that offers a sufficient gain in improvement, but without causing a large enough effect that all others will also change CODEC (as described above). On the other hand, Maja Sulovic, et al. [24] proposed a mechanism where this has been taken into account and the threshold was set using a more dynamic approach.

3.2 VoIP Client Implementation

Our first idea was to implement all this into minisip [12], an open-source SIP client created by KTH students and external contributors. We thought that would be a relatively light task, as our aim was only to make slight changes to a program we supposed stable.

However, once we got our hands on the code, we realised that Xiaokun Yi had already implemented his CODEC switching based on network feedback. Unfrotunately, we had some problems running properly minisip, specifically making it connect to public SIP servers. The documentation was not helpful and we could not contact the author to advise us on troubleshooting these issues.

Therefore, we decided on implementing the CODEC switching feature on another open source VoIP client, and Ekiga [4] was our application of choice. Due to limited time we only implemented the energy-aware side of the CODEC switching mechanism, as it will be explained on the following section.

4 Dynamic CODEC switching based on energy consumption

4.1 Introduction

Even when the network resources available are a key factor limiting the quality of a VoIP call, as stressed on the previous section, there remains another crucial parameter which determines whether or not the call can be completed successfully: energy consumption versus available energy. After all, VoIP exhibits high energy consumption compared to other popular applications such as web browsing [21], due to the use of the processor and the network interface during the entire session. The increasing popularity in both business and residential markets of portable devices such as smartphones, netbooks, tablets and laptops [9] only makes this problem more relevant. The constant evolution of such platforms in terms of processing power, memory capacity, and screen resolution, allows them to be used as multi-purpose machines for all kinds of applications, but also increases their power requirements, making battery life a main concern of their users.

Determining the individual contributions to energy expenditure of each piece of software that is executed on top of those devices can become quite a complex task, as most of them make use of many components such as the CPU, network interface or screen. The particular weight of each component in the overall energy budget is not only device-dependent, but also software-dependent [18].

As a side effect to their popularity, there has been some research dealing with the power requirements of WiFi capable portable devices when running VoIP applications leading to proposals for a wide variety of methods to reduce their energy expenditure. Most of the energy-saving techniques address the network interface as it accounts for a very significant portion of the power consumption [20]. The method operate by either by optimizing the MAC-sublayer operational values for specific network conditions, switching between different network interfaces whenever possible [11, 21], or taking advantage of the power saving mode (PSM) feature of WiFi interface with cross-layer approaches [15, 25]. Such approaches, though effective, suffer from a lack of generality, as they rely an the awareness of the VoIP application traffic pattern, thus preventing the use of standard VoIP software or default network interface driver implementations.

In this section of the project, as we did in the previous section concerning the network resources issue, we will address the energy problem from the point of view of which CODEC to choose. In fact, both problems are closely related, because the impact of the choice of CODEC on the computational requirements and traffic pattern also affects the device's power consumption, which lead us to study the effectiveness of CODEC switching on reducing the energy demands of VoIP applications. The use of CODEC switching as an energy-saving technique is not a novel idea. In [19] the battery level and the QoS thresholds are taken into account when switching CODECs, although in that work the battery measurements were made during the execution of a VoIP traffic emulator, without using actual VoIP CODECs, therefore ignoring the effect of the coding and decoding processes. Such a setup is in our opinion too detached from the real life issue of a graphical user client for VoIP running on a portable device.

Detailed measurements of the battery consumption of standard CODECs can be found in [13] where such expenditure is broken into its CPU and NIC components. However, that work stops at the measurement setup stage and does not analyze the possible benefits of a CODEC switching algorithm.

The aim of this section is to provide an overview of the different energy requirements of the most popular standard CODECs and to demonstrate the effectiveness of a CODEC switching scheme in prolonging the battery life of a handheld WiFi capable device during a VoIP call. Even though such a technique trades QoS for a longer battery life, maintaining an adequate balance between both parameters can actually improve the user experience [19], as the the user will enjoy a greater device autonomy, a key feature of these devices. To prove this point we will implement a simple energy-aware VoIP system based on Ekiga [4] an open source graphical user client, running on top of two Sony Vaio VPC-EB laptops with Ubuntu 10.04 as the operating system. As mentioned above, energy consumption is significantly device-dependent, so our measurements can only be translated qualitatively to other platforms. Throughout this section we will only consider audio CODECs, using the same ones that we

already studied. However, our results can be easily extended to video CODECs.

4.2 Overview of VoIP energy requirements

In order to evaluate the power saving effect of CODEC switching we must first determine on which component of a device's energy budget CODECs have the most significant impact. Based on an overview of the media processing pipeline in VoIP given in the previous section, we present a simplified sketch of how a VoIP application consumes energy in figure 7.

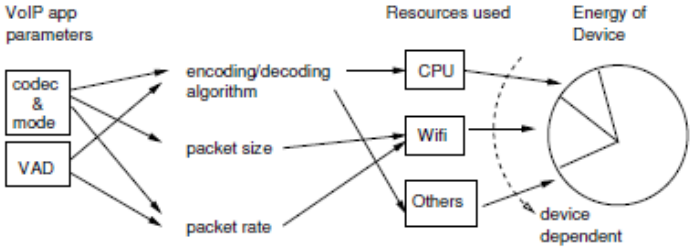

Figure 7 – Sketch of the VoIP energy budget [1]

Similar to other network-intensive software, VoIP makes use of the following hardware components:

- **Processor (CPU)**: The coding and decoding of the audio information, as well as the computing power required for running the VoIP user client itself put a significant load on the device's CPU, as the encoding should be significantly faster than the inter-frame delay. Complementary algorithms such as Voice Activity Detection(VAD) and echo reduction also take their toll, although in the case of VAD the extra processor load can be offset by the reduction in network traffic [26].
- **Memory**: Although some memory is needed by the encoding and decoding algorithms, according to [26] CODECs do not require access to large amounts of memory. The algorithms typically use relatively small buffers which can be stored in cache, avoiding the extra energy expenditure of accessing secondary memory or hard drives.
- **Network interface**: This is the component where the choice of CODEC has the greatest impact, as it effectively determines the network traffic requirements, as each CODEC has its own packet size and inter-packet periods. As the wireless interface accounts for a significant percentage of the global energy consumption of a WiFi capable device [26], reducing the network traffic is an obvious method of saving power.
- **Others**: The audio and video components, such as the screen, loudspeakers, and microphone also have a significant impact on energy consumption, but are not significantly CODEC-dependent.

4.3 Experimental setup

As we mentioned in the introduction, we used two Sony Vaio VPC-EB laptops with Ubuntu 10.04 as test platforms, by running the open source VoIP client Ekiga on top of them. In order to connect those two clients we created two SIP accounts on iptel.org IP telephony service [3] which provides a free SIP server. The topology of for these tests is shown in Figure 8.

[1] Picture taken from [13], with explicit permission from the author.

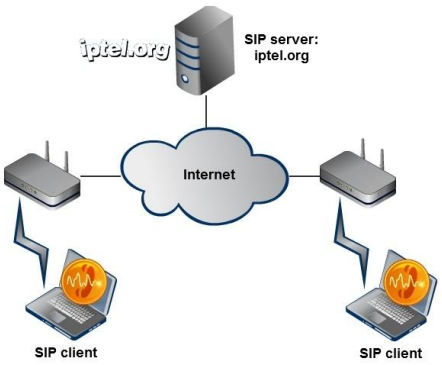

Figure 8 – Experimental setup topology

The stable release of Ekiga does not support dynamic CODEC switching during a VoIP call, so we had to build a customized battery-aware version of that program, modifying the opal library along the lines of [27] and making use of the Ubuntu version of the ACPI library [1]. This library provides routines for reading ACPI(Advanced Configuration and Power Interface, a standard for device configuration and power management) values in an easy way.

One of our major concerns during all the tests was to keep all factors affecting battery consumption other than the actual VoIP calls as constant as possible. To that end we disabled all the power management features, including screensaver, on both computers during the measurements and did not modify the screen brightness nor the loudspeaker's volume level during the tests. Additionally, to further standarize the tests we fed the audio inputs of both VoIP clients directly by playing an audio file directly into it, avoiding the microphone input and all the variability that it entails. As input file we chose a version of Macbeth's first act whose reproduction is allowed for educational purposes [6], playing it on both endpoints for 15 minutes, the duration of each test.

4.4 Battery Measurements

4.4.1 Introduction

To build an effective energy saving algorithm which makes use CODEC switching we first must determine the effect that the CODECs supported by our VoIP program have on battery consumption. However, as discussed on a previous section, this contribution is not easy to isolate from all the other processes and components consuming power on a laptop. Many quite complex models have been proposed to solve this [13]. However, in this work we will take a simpler approach, as we only need qualitative results, because the quantitative data obtained will be device and software-dependent.

As we stressed on the breakdown of a VoIP program's energy requirements, CODECs only have a significant impact on the load of the CPU and the network interface. Therefore, our objective will be to substract all the other factors from the laptop's energy consumption during the measurement.

4.4.2 CPU energy consumption

First we need to determine the "baseline" power consumption of our test system, so we can compare the energy expenditure of a VoIP call with that of the normal operation of the laptop. To determine this baseline we collected several 15 minutes samples of the battery level of the computer, always starting with the laptop fully charged. During this measurement we will keep the screen active, but disabled the networking interface while executing Ekiga and the console program(mplayer) used to play the test audio file, so the energy consumption of Ekiga's graphical interface and the audio reproduction could be eliminated. The results of these measurements are shown in Figure 9.

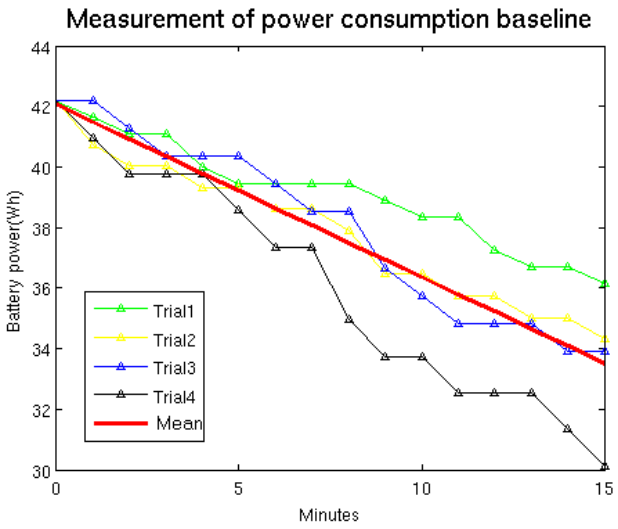

Figure 9 – Power consumption baseline

As it can be observed, there is a high variability between the different samples and none of them show a linear drop. However, for simplicity's sake we will use the average value of all of them as the baseline power consumption rate, treating it as a constant, instead of utilizing a more complex model.

To study the CPU energy expenditures of each CODEC we will make a 15 minute VoIP call to a local echo server running on the same laptop with Ekiga, thus avoiding the power expenses of network traffic, so the only significant contribution of the different CODECs is on CPU load. The result of running this test can be seen in the Figure 10.

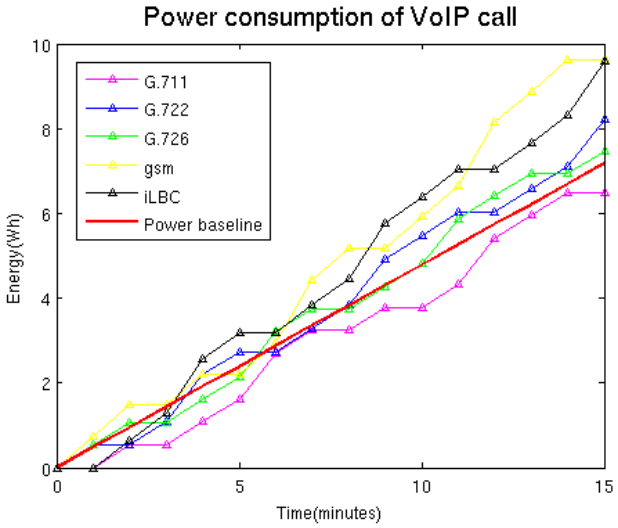

Figure 10 – VoIP calls without networking

Even though there are significant differences between the energy expenditures of each CODEC, all of them fall within the variability found in the baseline power measurements, so we can conclude that CODEC switching is not going to have a noticeable effect on reducing the CPU's energy requirements.

4.4.3 Network interface energy consumption

The WLAN interface of a modern laptop accounts for more than 15% of the energy budget, according to previous studies [19], while for lower-end devices such as a PDA that number can increase up to 65% [22]. Therefore, reducing the load on the network interface isan important step towards saving energy. In order to demonstrate this, we will conduct energy consumption measurements for VoIP

calls in a wireless network, using the setup described in the previous section. The results of these tests are shown in Figure 11.

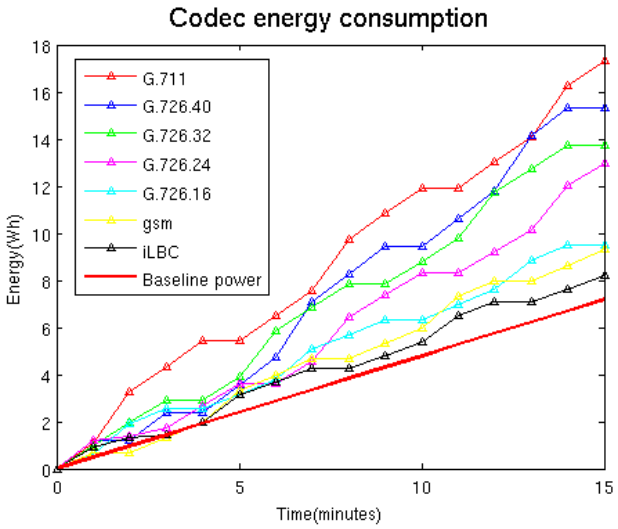

Figure 11 – Power consumption of a VoIP call

As we can see in Figure 11 the choice of CODEC clearly has a noticeable impact on the battery usage, directly related to the bandwidth requirements of each CODEC. G.711 has the highest bitrate of all the CODECs we tested(64kbit/s) and not surprisingly consumes the most energy, while iLBC, with the lowest bitrate(13.33kbit/s in Ekiga's implementation of this CODEC) also has the lowest energy requirements. In fact, Figure 11 shows that ordering the CODECs in terms of bitrate is the same ordering as when we use power consumption as the ordering criteria. This results leads us to propose a CODEC switching scheme to extend the battery life of a WiFi device.

4.5 Energy-aware Ekiga

The current stable version of Ekiga does not support dynamic CODEC switching during a call, so we had to modify it to meet our ends. In order to explain these changes, we first give a brief overview of the program's internal architecture. The program is composed of the following parts:

Ekiga The application's GUI(Graphical User Interface) and the program proper, which relies heavily in the opal library for all the networking functionalities.

PTlib An open source abstraction C++ library which comprises methods to product applications to run in both Windows and Unix systems [2]. It is focused on networking, I/O portability, multi-threading and protocol portability, making it an useful tool for multiplatform network-centric programs.

Opal The Open Phone Abstraction Library is a C++ class library which normalises the numerous telephony protocols into a single integrated call model [7]. It presents the overlying program with an interface to make calls, allowing them to be placed and the media flow to be handled in an identical manner, regardless of the specific telephony protocol or hardware used. To that end, it treats the concept of a *Call* in a pretty abstract way, defining it as the connection between two or more *Addresses* each through a *Connection* created by an *Endpoint*. The call usually has multiple *Media Streams* transferring information (e.g. voice) of a particular *Media Format* from one endpoint to another. The data is copied from a *Source Media Stream* to a *Sink Media Stream* via a *Media Patch* thread. Such structure is illustrated by the following picture.

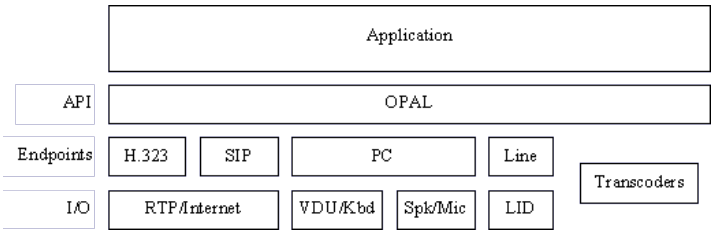

Figure 12 – Opal internal architecture

As the user interface is responsible for invoking the opal functions that start the call and create the required media streams, we modified the thread that initiates and controlls a VoIP call to periodically sample the battery level by invoking an ACPI library function. The information collected is used to decide whether or not there is a need to change the CODEC.

As for the actual energy-saving algorithm used, out of the many possibilities [16] we chose the simplest: a two threshold mechanism which changes the CODEC when the battery power drops below a fixed level. As we proved in the Section 4.4.3, each CODEC has its own power requirements, directly correlated to its bitrate. Therefore, as the battery level drops, a transition should be made from a high bitrate CODEC to a lower bitrate CODEC in order to maintain a balance between battery life and audio quality. In order to keep things simple we only used three CODECs in this CODEC switching scheme: G.711, which provides the best quality at the highest energy cost, iLBC, which sacrifices quality for the lowest power requirements, and the 24 kbit/s version of G.726, which has intermediate characteristics. The mechanism deployed for the switching is illustrated in the state diagram shown in Figure 13.

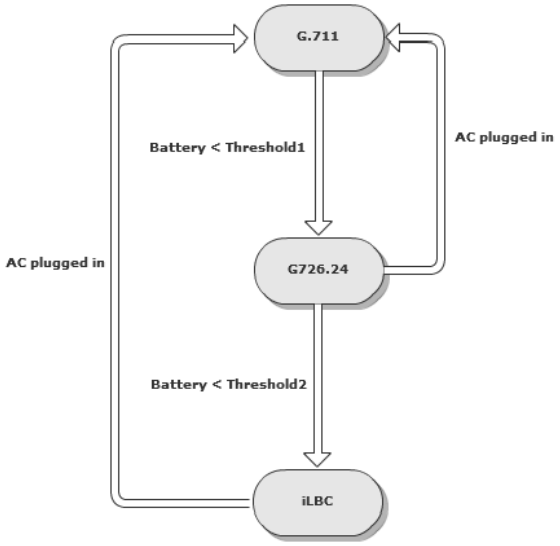

Figure 13 – State transition of CODEC switching

To test this new implementation we started a new VoIP call under the same conditions as the previous ones and studied how the battery's energy level dropped. For the trial we set the static thresholds at $\frac{2}{3}$ and $\frac{1}{3}$ of the full battery capacity, respectively. The result is shown on the in Figure 14, where the horizontal lines represent two thresholds.

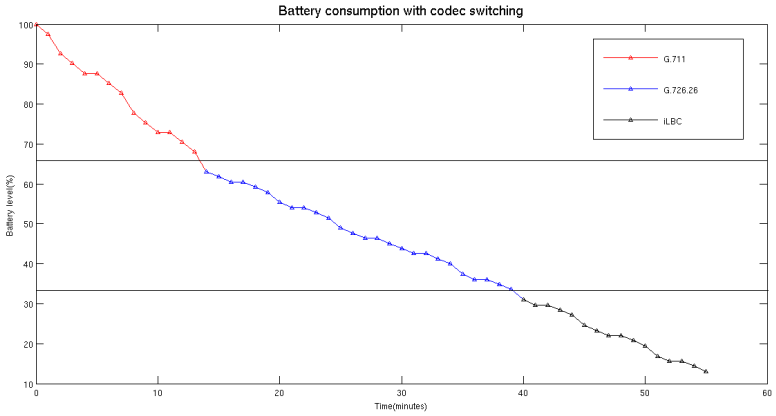

Figure 14 – CODEC switching VoIP call

As can be observed, G.711 expends energy rather quickly, so after a short interval the program changes the CODEC to the 24 kbit/s version of G.726, thus slowing the rate of battery consumption, and when the battery's capacity falls below the second threshold, Ekiga switches to iLBC to make the most of the last third of the battery capacity. To compare the gain in battery life thus obtained, we measured how long a conversation could last with each of the CODECs involved on our CODEC switching algorithm. We started the call with the laptop fully charged and stopped it once the battery reached the 5% level. The results are shown in Table 2. The MOS of each CODEC was extracted from [16].

Table 2 – Maximum conversation length in function on CODEC scheme

CODEC	MOS	Max length of conversation (min)
G.711	4.1	44
G.726-24kbit/s	3.81	62
iLBC-13.3kbit/s	3.68	87
CODEC switching	3.80[2]	72

This simple test shows that CODEC switching can considerably extend the battery life of a VoIP capable device, while keeping the quality of the call within acceptable bounds. The calculation of the MOS of such a call is beyond the scope of this project, but it is reasonable to expect that the value would be close to the weighted mean of the MOS of the different CODECs used.

5 Conclusions and future work

In this project we have shown that CODEC switching is an effective way to adapt a VoIP program to the available resources (in this study these were available network bandwidth and battery level) on the application level, *without* using complicated hardware-specific schemes. We have also demonstrated with measurements on typical laptop devices the considerable performance benefits of using a context-aware VoIP system in terms of talk time and battery consumption.

However, we limited ourselves to analyzing the effect of switching audio CODECs only. We believe that applying the same techniques to video CODECs would have even more significant consequences, due to the high bandwidth and energy requirements of video streaming. Regarding our energy-aware setup, we must stress the simplicity of the switching algorithm that we developed. Using adaptive thresholds or some other more flexible algorithm would definitely be more efficient, so there remains plenty of room for improvement.

Additionally, in this project we have dealt with the energy-aware and bandwidth-aware CODEC switching strategies separately, whereas a true context-aware approach would combine the two approaches. Finding an appropriate balance between all these parameters remains for future work.

[2]Value for the weighted mean

References

[1] "libacpi," http://www.makelinux.net/man/3/L/libacpi. [Online]. Available: http://www.makelinux.net/man/3/L/libacpi

[2] "PTLib," http://www.opalvoip.org/docs/ptlib-v2_8/, May 2004. [Online]. Available: http://www.opalvoip.org/docs/ptlib-v2_8/

[3] "SIP service | iptel.org," http://www.iptel.org/service, 2007. [Online]. Available: http://www.iptel.org/service

[4] "Ekiga ˜ free your speech," http://ekiga.org/, Oct. 2009. [Online]. Available: http://ekiga.org/

[5] "Audio codec - wikipedia, the free encyclopedia," http://en.wikipedia.org/wiki/Audio_codec, Aug. 2011. [Online]. Available: http://en.wikipedia.org/wiki/Audio_codec

[6] "Macbeth-Ohio university," http://www.wiredforbooks.org/shakespeare/, Sep. 2011. [Online]. Available: http://www.wiredforbooks.org/shakespeare/

[7] "OpalVoip - open source voice, video and fax," http://www.opalvoip.org/, Aug. 2011. [Online]. Available: http://www.opalvoip.org/

[8] "Public switched telephone network - wikipedia, the free encyclopedia," http://en.wikipedia.org/wiki/Public_switched_telephone_network, Sep. 2011. [Online]. Available: http://en.wikipedia.org/wiki/Public_switched_telephone_network

[9] "Smartphone user statistics, mobile phone statistics 2011," http://www.smartonline.com/mobile-2/us-smartphone-statistics-q1-2011-overview/, May 2011. [Online]. Available: http://www.smartonline.com/mobile-2/us-smartphone-statistics-q1-2011-overview/

[10] "Voice over internet protocol - wikipedia, the free encyclopedia," http://en.wikipedia.org/wiki/Voice_over_Internet_Protocol, Sep. 2011. [Online]. Available: http://en.wikipedia.org/wiki/Voice_over_Internet_Protocol

[11] Y. Agarwal, R. Chandra, A. Wolman, P. Bahl, K. Chin, and R. Gupta, "Wireless wakeups revisited: energy management for voip over wi-fi smartphones," in *Proceedings of the 5th international conference on Mobile systems, applications and services*, 2007, p. 179–191.

[12] Erik Eliasson, Jon-Olov Vatn, Johan Bilien, Cesc Santasusana, and Max Zomborszki, "Minisip project," http://www.minisip.org/. [Online]. Available: http://www.minisip.org/

[13] Antonio J. Estepa, Juan M. Vozmediano, Jorge López, and Rafael M. Estepa, "Impact of VoIP codecs on the energy consumption of portable devices," Ph.D. dissertation, Universidad de Sevilla, 2011.

[14] ITU, "One-way transmission time (G.114)," May 2003. [Online]. Available: http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-G.114-200305-I!!PDF-E&type=items

[15] John Kotwicki, "An analysis of energy-efficient VoIP communication in wireless networks," Ph.D. dissertation, Case Western Reserve University, Mar. 2004. [Online]. Available: http://vorlon.case.edu/~vxl11/NetBots/kotwicki-thesis.pdf

[16] Y.H. Lu, T. Simunic, and G. De Micheli, "Software controlled power management," in *Hardware/Software Codesign, 1999.(CODES'99) Proceedings of the Seventh International Workshop on*, 1999, p. 157–161.

[17] Gerald Q. Maguire Jr., "Practical voice over IP (VoIP): SIP, and related protocols," http://www.ict.kth.se/courses/IK2554/VoIP-Coursepage-Fall-2011.html, Oct. 2011. [Online]. Available: http://www.ict.kth.se/courses/IK2554/VoIP-Coursepage-Fall-2011.html

[18] A. Mahesri and V. Vardhan, "Power consumption breakdown on a modern laptop," *Power-Aware Computer Systems*, p. 165–180, 2005.

[19] M. Naeem, V. Namboodiri, and R. Pendse, "Energy implication of various voip codecs in portable devices," in *Proceedings - Conference on Local Computer Networks, LCN*, 2010, pp. 196–199. [Online]. Available: www.scopus.com

[20] V. Namboodiri and L. Gao, "Towards energy efficient VoIP over wireless LANs," in *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, 2008, p. 169–178.

[21] T. Pering, Y. Agarwal, R. Gupta, and R. Want, "CoolSpots: reducing the power consumption of wireless mobile devices with multiple radio interfaces," in *Proceedings of the 4th international conference on Mobile systems, applications and services*, 2006, p. 220–232.

[22] V. Raghunathan, T. Pering, R. Want, A. Nguyen, and P. Jensen, "Experience with a low power wireless mobile computing platform," in *Low Power Electronics and Design, 2004. ISLPED'04. Proceedings of the 2004 International Symposium on*, 2004, p. 363–368.

[23] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," *Internet Request for Comments*, vol. RFC 3550 (Standard), Jul. 2003, updated by RFCs 5506, 5761, 6051, 6222. [Online]. Available: http://www.rfc-editor.org/rfc/rfc3550.txt

[24] M. Sulovic, D. Raca, M. Hadzialic, and N. Hadziahmetovic, "Dynamic codec selection algorithm for VoIP," in *ICDT 2011, The Sixth International Conference on Digital Telecommunications*, 2011, p. 74–79.

[25] Shiao-Li Tsao and Chung-Huei Huang, "An energy-efficient transmission mechanism for VoIP over IEEE 802.11 WLAN," *Wireless Communications and Mobile Computing*, vol. 9, pp. 1629–1644, Dec. 2009. [Online]. Available: http://doi.wiley.com/10.1002/wcm.747

[26] R. Venkatesha Prasad, A. Sangwan, HS Jamadagni, MC Chiranth, R. Sah, and V. Gaurav, "Comparison of voice activity detection algorithms for VoIP," in *Computers and Communications, 2002. Proceedings. ISCC 2002. Seventh International Symposium on*, 2002, p. 530–535.

[27] Xiaokun Yi, "Adaptive wireless multimedia services," Master's thesis, KTH, Stockholm, 2006. [Online]. Available: http://web.it.kth.se/~maguire/DEGREE-PROJECT-REPORTS/060511-Xiaokun_Yi-Thesis-with-cover.pdf