Dynamic VoIP codec selection on smartphones

Antonio Estepa, Rafael Estepa, Juan Vozmediano, Pablo Carrillo

Dept. of Telematics, University of Seville (Spain)

C/Camino de los Descubrimientos s/n, 41092, Seville (Spain)

Tel: +34-954-487-384 E-mail: {aestepa,rafa,jvt}@trajano.us.es, pablo.carrillo.alvarez@calvium.com

Received: March 27, 2014 Accepted: June 8, 2014 Published: June 30, 2014

DOI: 10.5296/npa.v6i2.5370 URL: http://dx.doi.org/10.5296/npa.v6i2.5370

Abstract

Voice over IP (VoIP) applications can choose a plethora of different speech codecs, which differ in bandwidth, listening speech quality, and resilience to quality degradation under packet loss. However, VoIP Codecs also exhibit differences in facets such as computational complexity or traffic generated that impact on the energy consumption of smartphones due to the use of processor.

This work deals with the study of energy consumption differences among VoIP codecs. We compare the execution time required to encode/decode reference conversations. Our results show that computational complexity has a significant impact on battery consumption (a factor of up to 10 was found between different codecs). Based on our results, we provide a ranking of energy efficiency. We also propose a simple algorithm for codec dynamic selection considering the factors of quality, energy and bandwidth. Our algorithm reacts to network conditions choosing the codec that provides less battery consumption constrained to user-defined targets for minimum quality and maximum codec bitrate.

Keywords: VoIP, codec, energy efficiency, QoS.

1. Introduction

Mobile devices such as smartphones or tablets have been widely adopted in recent years in both business and residential markets. The evolution of technology with increasing processing power at low cost, and the growing presence of public Wi-Fi hotspots is spurring the use of current mobile devices as multi-purpose machines running productivity, social and multimedia applications. In particular, the scope of this paper is in VoIP applications such as Facetime, Skype, hangouts, etc. that offer voice and/or video calls using the free Internet.

Codecs are at the heart of VoIP applications. There are numerous standardized codecs, each operating at a particular bit-rate, and providing a specific listening quality or loss resilience [1, 2]. These factors have traditionally marked the trade-off for codec selection either statically by developer/user configuration, or dynamically under changes of network conditions [3, 4].

However, the battery life still persists as a key concern in users of portable devices. The actual battery drain on a device depends on the hardware (i.e. smartphone particular circuitry) but also on the software running upon it (e.g. VoIP app). Codecs differ on computational complexity and the rate and size of the packets generated, which in turn impacts on the use of processor and Wi-Fi card. Consequently, for a specific device, different VoIP codecs should exhibit a differentiated demand of battery. We think that this fact should be taken into account, together with bandwidth and perceived quality, for codec selection in battery-operated devices.

The study of energy consumption in portable devices running VoIP software is not new. Some studies are focused on energy-saving at the network interface card (NIC) component, either by optimizing the MAC sub-layer operational values for specific network conditions [5], using alternative NICs when possible [6] and most notably, by using the Power Saving Mode (PSM) of Wi-Fi cards with cross-layer approaches [7]. However, their proposals either prevent the use of standard VoIP software, or change the default NIC operational behavior. Besides, in a previous work [2] we showed that the energy expenditure at the Wi-Fi card due to VoIP traffic is for most codecs significantly smaller than energy expenditure attributable to the process of encoding/decoding voice frames at the processor. In addition, energy-wise, the traffic generated by different VoIP codecs does not exhibit significant differences since all codecs send small packets every 10ms-30ms, which means that almost 98% of the time, 802.11 g/n/ac cards are in idle/sleep state which dominates the energy consumption. Therefore, we believe that the use of the processor is the main factor that differentiates the energy efficiency of each different codec. However, our previous work was done in a laptop with i386 processor and those results should be extended to smartphones with arm architectures.

The goal of this paper is to present a codec selection algorithm that takes into consideration bandwidth, QoS and energy efficiency. In particular we have two objectives:

- To present an empirical study of the processor demand of VoIP codecs in a real smartphone (iphone 4S)

- To propose an algorithm for codec dynamic selection that minimizes the energy consumption attributable to the VoIP application whilst maintaining a minimum target QoS and bandwidth requirement.

The remainder of this paper is as follows. Section 2 provides an overview of the main characteristics of VoIP codecs and their relation with energy consumption. Section 3 provides an experimental study of different codec that sorts them according to their demand of processor. Section 4 presents a simple codec selection algorithm that utilizes the former ranking to dynamically select the codec that minimizes the energy consumption on a device constrained to obtain a minimum user-selected QoS and codec bit-rate limit. Section 5 explains how we have checked our algorithm in a real test-bed. Section 6 provides the experimental results and comment on then. Finally, Section 7 concludes the paper.

2. VoIP Codecs and energy consumption

Figure 1 represents the main components involved in the media pipelining of one side of a VoIP application. The device's sound card generates 8 KHz digital samples (PCM 16 bits) of the original sound. Sets of these samples are processed by the codec to generate code-words termed frames at periodic intervals. The frame size and inter-frame period will depend on the particular compression algorithm applied by each codec. Frames generated by a codec are packetized to be delivered to the right codec in the destination. This process includes the use of the real-time protocol (RTP), UDP and IP protocols. Overhead savings can be obtained by packing multiple frames into a single IP datagram. However, most VoIP applications apply when feasible the default inter-packet value of 20 ms as suggested by the RTP profile [8]. IP packets are transmitted and received by the NIC of the device which, for our study, will be a Wi-Fi card.

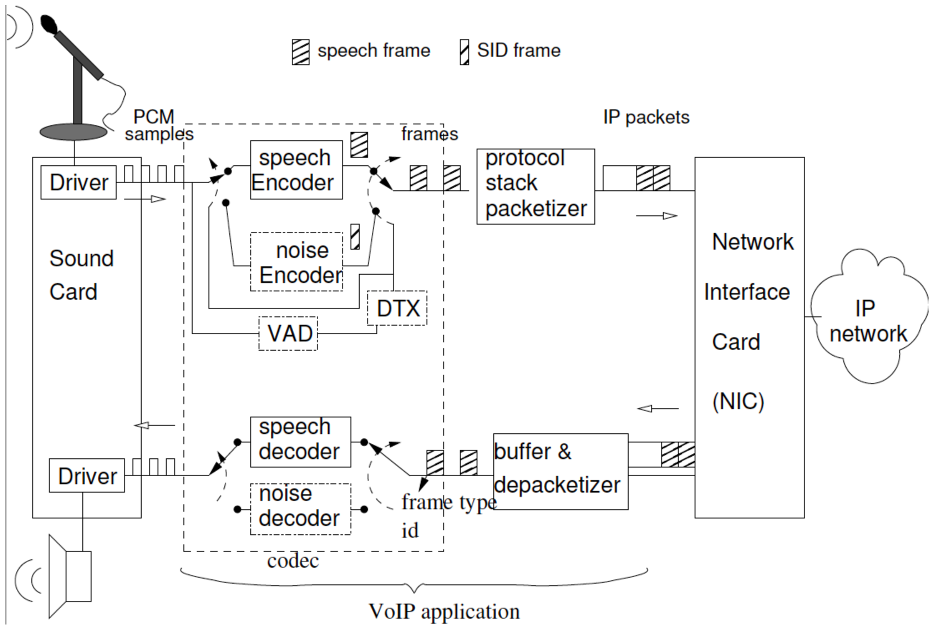

Figure 1. Elements involved in the media pipeline

In the reverse path, the frames encapsulated in received IP packets are placed in a buffer managed by the VoIP application to compensate the network jitter. Every inter-frame period, a new frame is read from the buffer and decoded to obtain again a set of 16-bit PCM samples that reconstruct the original signal with some distortion. Some codecs define their own Packet Loss Concealment and redundancy schemes mechanisms to mitigate the sound effect of missing frames. Therefore, codecs mainly differ in the sound quality, bitrate, perception of packet loss, and computational requirements.

In order to reduce their bandwidth requirement, some codecs can also generate variable bit-rate (VBR) traffic using the following:

- *Silence Suppression*. This feature prevents the generation of frames during voice inactive periods, obtaining bandwidth savings over 50%. It relies on a Voice Activity Detector (VAD) algorithm to tell whether voice is active or not. A Discontinuous Transmission Algorithm (DTX) determines for each unvoiced frame, the need to send a background noise update from the encoder to the receiver. This background noise update is encoded differently from speech frames in smaller-sized frames which are decoded by the Comfort Noise Generator (CNG) at the receiver codec [9] thanks to a frame-type field included in the frames. G.729b, G.723.1 and AMR are examples of codecs that feature Silence Suppression

- *Multi-rate*. Some codecs offer multiple encoding algorithms which results in multiple bit-rate by generating frames of different sizes or/and changing the inter-frame period. Each mode of operation in these multi-rate codecs exhibits also a different sound quality, and is indicated in the frames. G.723.1, AMR and iLBC are examples of multi-rate codecs

Table 1 summarizes the main characteristics of the ITU-T, 3GPP and IETF narrowband 1 codecs that are widely adopted in today's VoIP applications. For each codec it provides its sampling and bit-rate, the size of speech and SID frames, the encoding algorithm family, its capacity for DTX and the intrinsic listening speech quality resulting from the ITU-T P.862 PESQ measurement method. Finally, we show the URL of the reference source code from the corresponding standards

codec	bit-rate (Kbps)	frame size (ms)	speech frame size (bit)	SID frame size	DTX	QoS (MOS)	C source
G.711	64	20	640	-	No	4.39	http://www.itu.int/rec/T- REC- G.711/es
G.723.1	6.3	30	192	32	Yes	3.69	http://www.itu.int/rec/T- REC-G.723.1/es
	5.3		160			3.49	
G.729	8	10	80	-	No	3.75	http://www.itu.int/rec/T-REC-G.729/es
G.729A				-	No	3.67	
G.729B				10	Yes	3.51	
G.729AB				10	Yes	3.55	
AMR	12.2	20	244	39	Yes	3.97	http://3gpp.org/ftp/Specs/html/26073.htm
	10.2		204			3.93	

	7.95		159			3.69	
	7.4		148			3.71	
	6.7		134			3.64	
	5.9		118			3.55	
	5.15		103			3.44	
	4.75		95			3.39	
iLBC	15.2	20	303	-	No	3.86	http://www.ietf.org/rfc/rfc3951.txt
	13.3	30	399			3.82	

Table 1. Characteristics of the codecs under study.

From the media pipeline described previously we can conclude that the main hardware components used by the VoIP app are:

- *Processor*: used by the codec process to run the algorithms for encoding, decoding and voice activity detection (if implemented).
- *Wi-Fi card*. Used by the VoIP application to send and receive speech frames encapsulated in IP packets. As default, every 20ms a new packet is generated.
- *Other*: components such as display, sound card or memory are clearly used by VoIP applications. However, we will assume that the particular codec makes no difference in the use of these components.

Since we are interested in finding differences in the energy consumption of a device attributable to codecs, our interest will be focused in Processor and Wi-Fi interface card. However, in a previous work [2] we found that there are no significant differences in the energy spent at the Wi-Fi card due to VoIP traffic in spite of the codec used. Although it is true that each codec generates a differentiated traffic pattern, all codecs send small packets every 10ms-30ms, which means that almost 98% of the time, 802.11 g/n/ac cards are idle/sleep state which is the dominant factor in the energy consumption.

Thus, the use of the processor will dominate the differences in the energy consumption at hardware components attributable to VoIP codecs.

3. Related Works

As stated in the Introduction, most literature is focused either on improving QoS or improving energy efficiency by means of spending less energy at the radio interface (i.e. Wi-Fi card). The challenge of preserving quality in VoIP over Wi-Fi has been extensively addressed in literature. Research efforts to support quality in VoWi-Fi have been focused on two main approaches: (a) dimensioning works aimed at finding the maximum number of simultaneous VoIP flows that IEEE 802.11 networks can accommodate while satisfying QoS constraints (e.g. network delay, packet loss ratio) [10]–[13]; and (b) link-layer proposals aimed at meeting QoS constraints in the IEEE 802.11 network by finding optimum values for MAC layer variables such as contention window size, maximum retry limits, etc.

[7],[14],[15]. Dynamic adaptation of VoIP application variables such as codec or other factors [16], to preserve QoS has been also addressed in literature. A recent survey of existing proposals can be found in [17].

Since VoIP applications may be running on battery-powered terminals, an emergent research topic concerned with energy efficiency has been developed over recent years. A comprehensive survey can be found in [18] where the authors review existing MAC-layer based strategies to save energy in IEEE 802.11 by minimizing the time spent in active states (i.e.TX,RX).. Nevertheless, as stated in [2], VoIP traffic is just about 1-2% of the bandwidth available in current Wi-Fi cards, in spite of the codec used. Therefore, energy expenditure at the Wi-Fi card will be dominated by idle/sleep state. Consequently, different codecs should not offer significant differences in the energy expenditure attributable to the NIC of a device.

To the best of our knowledge, no successful attempts have been made to dynamically adjust VoIP application variables to manage the optimization of both QoS and energy efficiency. In [20] the authors emulate VoIP traffic between two laptops and fetch the battery level changes with emulated traffic of G.711, G.729 and G.723 codecs. Based on this, they make a ranking of energy consumption of these codec and suggest to use the codec with higher intrinsic QoS when the battery level is full and switch to more energy-friendly codecs when the battery is drained to a certain threshold. In [19] the authors study the interdependencies between QoS, bandwidth and energy attributable to codecs. However, it seems preliminary work with unclear methodology where energy and MOS of G.711, G.729 and G.723 codecs are analyzed to discuss different cases when each one would be the best choice. Besides, the far end of the conversation, silence suppression or codecs such AMR or iLBC are ignored.

4. Experimental demand of processor in a smartphone

We have designed an experiment to measure the use of the processor made by each codec in Table 1 while encoding / decoding a set of reference conversations. The experiment is similar to the one conducted in [2] but executed in a smartphone iphone4S (processor Dual-core 1 GHz Cortex-A9), which requires a dedicated new app.

We changed each codec source code from Table 1 to fetch the processor time used during the encoding/decoding process using the *getrusage* standard C function. A set of real telephony conversations (two channel audio PCM files) have encoded and decoded as in the reality (i.e. encode one way and decode the reverse way of the conversation). Thanks to the use of *getrusage* we can find out the processor time devoted to encoding and decoding of each conversation.

Then, assuming a simple lineal model, the processor consumes $P \times t$ where P is a power constant that depends on the particular processor and t is the execution time of the task. Remember that we do not intend to find out the absolute value of energy consumption but to establish a comparison between different codecs.

4.1 Numerical Results.

4.1.1 Processor Time

Table 2 shows the processor time utilized in encoding / decoding our 2-hours reference conversations (average results for 5 iterations) per second of conversation. The last column shows the time required to encode one way and decode the reverse way of the conversations.

	codec	Encoding speech: processor time per 1 sec. of speech (conf.interval 95%)		Decoding speech processor time per 1 sec. of speech (conf. interval. 95%)		Total processor time per 1 sec. of speech
1	ILBC20	0,149301743	(± 0,00107)	0,061013747	(±0,03009)	0,21031549
2	ILBC30	0,174936419	(±0,001023)	0,063908692	(±0,02602)	0,23884511
3	G729ab	0,420311609	(±0,061631)	0,127081461	(±0,02646)	0,54739307
4	G729a	0,550021348	(±0,001070)	0,121189463	(± 0,0051)	0,67121081
5	AMR7.4VAD	0,620452875	(±0,063804)	0,10930029	(± 0,15619)	0,729753166
6	AMR4.75VAD	0,632234821	(±0,064395)	0,110665	(± 0,16836)	0,742899821
7	AMR12.2VAD	0,681453596	(±0,054655)	0,112052262	(± 0,15182)	0,793505859
8	G729b	0,654882493	(±0,053413)	0,172082041	(± 0,06861)	0,826964534
9	G723_53VAD	0,822204927	(±0,047399)	0,090501021	(± 0,00744)	0,912705948
10	AMR7.4	0,830066951	(±0,000957)	0,126136726	(± 0,00325)	0,956203677
11	AMR4.75	0,85029538	(±0,000959)	0,129093318	(± 0,00946)	0,979388698
12	AMR12.2	0,898256708	(±0,000855)	0,128837097	(± 0,00517)	1,027093806
13	G729	0,938950473	(±0,001111)	0,192109315	(± 0,01279)	1,131059789
14	G723_63VAD	1,073761337	(±0,043416)	0,090385851	(± 0,14366)	1,164147188
15	G723_53	1,11295565	(±0,001344)	0,101190911	(± 0,00744)	1,21414656
16	G723_63	1,485209578	(±0,002240)	0,10106657	(± 0,0068)	1,586276148

Table 2. Processor time required to encode /decode one sec. of speech. (G.711 requires no processing time)

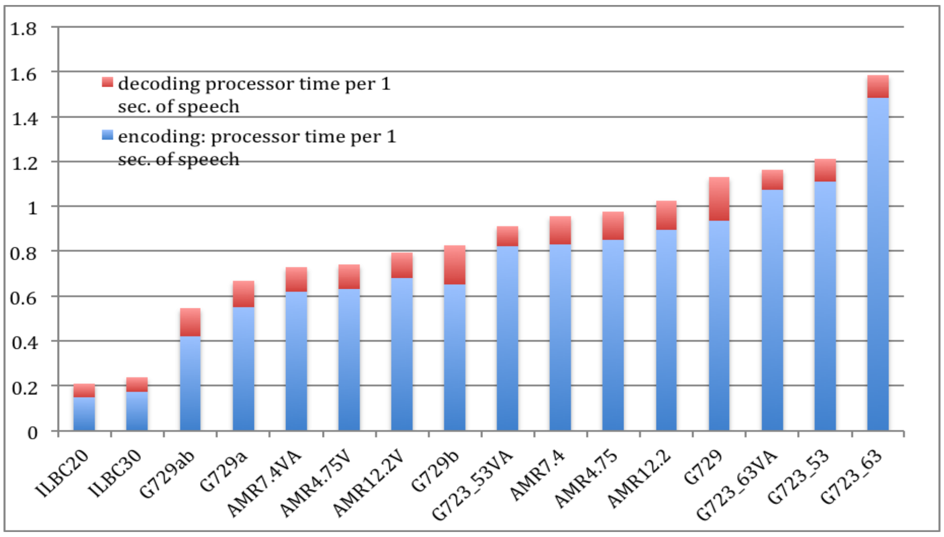

Figure 2. Processor time required to encode /decode

Figure 2 illustrates data from Table 2. From the previous results we highlight the following:

a) Significant differences are found between codecs. For example, G.723 (mode 6.3) requires almost 10 times more processing time than iLBC20. Those codecs that require more than 1 second of processor per 1 second of speech are clearly not usable in this device in its standard implementation without optimization.

b) Codec performing VAD exhibit higher variance in their results, which can be traced back to the fact that each conversation has its own voice activity pattern.

c) The process of decoding requires significantly less processing effort than the process of encoding.

Experimentally we have checked other arm architectures, obtaining a ranking with minor variations respect to our results. However, *intel*® architectures provide a ranking with significant variations to ours (seen [2]).

4.1.2 Battery life attributable to the processor.

A specific app was also developed to fetch the battery discharge process while encoding frames. We simply set the airplane mode with no other process running and start a infinite loop encoding frames. We log every time that battery decreases 5% starting from 100%. We obtained a battery discharge of 0,42% every minute of use of processor for the encoding/decoding task. We also repeated this process without encoding frames to find out the baseline battery consumption. Subtracting this baseline rate[1], we obtain 0,19% per every minute of use of processor. Table 3 uses the data in Table 2 to show the longest conversation possible according to each codec. Only feasible codecs are listed. Note that the smartphone was unused before this experiment and hence, the battery was new.

codec	AMR4.75	AMR7.4	G723_53VAD	G729b	AMR12.2 VAD	AMR4.75 VAD	AMR7.4 VAD	G729a	G729ab	ILBC30	ILBC20
(min)	243,0087	248,9009	260,763	287,7995	299,9347	320,3662	326,1376	354,583	434,7881	996,4616	1.131,63

Table 3. Battery life for each codec due to the use of the processor.

5. Codec Selection Algorithm

In this section we provide a simple codec selection algorithm that dynamically selects the most energy-friendly codec constrained to a minimum target QoS and a maximum bandwidth.

Our algorithm considers the following factors:

- **QoS**: it can be assessed periodically during conversation. We use the E-Model[21] for QoS estimation. The E-Model provides a score ranging from 0 to 100 termed the R factor, where R values greater than 70 are commonly considered acceptable. In its simplest form, the E-Model can be expressed as the addition of the following additive factors:

$$R = R_0 - I_d - I_{e\text{-}eff} \qquad (1)$$

[1] A total of 0,42%of discharge rate is measured without subtracting the baseline rate. Then, the processor would fully discharge the 1432mAh (5.3Wh) battery after 526min encoding frames.

where R_0 is a fixed value of 93,2, the factor I_d accounts for the negative effect of delay in conversation interactivity, and $I_{e,eff}$ represents impairment associated with codec compression, packets loss rate and packet loss behavior. The $I_{e\text{-}eff}$ factor can be further expressed as:

$$I_{e-eff} = I_e + (95 - I_e) \times \frac{P_{pl}}{P_{pl}/BurstR + B_{pl}} \qquad (2)$$

where P_{pl} is the packet loss ratio, *BurstR* is related to packet losses burst length, and B_{pl} is a codec dependent parameter related to the capacity of supporting not randomly distributed packet losses. In [22] can be found the I_e and B_{pl} parameters for the main ITU-T codecs (G.711, G723, G729,…). For codecs not defined by ITU-T such as iLBC, we will consider a non linear regression model derived for each codec by the least squares method and curve fitting [23]. Then, $I_{e\text{-eff}}$ has the following form

$$I_{e-eff} = a \times \log(1 + b \times P_{pl}) + c \qquad (3)$$

where values for parameters *a*, *b* and *c* can be found in [23]. The delay impairment (I_d) [24] can be expressed as:

$$I_d = (0.024 - d) + 0{,}11(d - 177.3) \times H(d - 177.3)$$

Where *d* represents the overall delay (one-way delay, including network and terminals) and H is the heavy side function.

- **Bitrate**: each codec at Table 1 operates at a certain bit-rate depending on the mode of operation and the use of VAD. A smartphone user might want to use only those codecs that operates at a bit-rate lower than a certain threshold.
- **Energy Ranking**: each codec will hold the energy index (first column) of Table 2. At any moment, from those codecs eligible (i.e. that meet bitrate and R contrains), the best codec would be the one with the lowest raking number. We are aware that different smartphones could exhibit a different ranking, but since the ranking is related to codec complexity we don't believe that there would be a big difference energy-wise.

5.1 Proposed Algorithm

The algorithm utilizes a bi-dimensional table (*TableCodecs*) with the following columns: {#rakingEnergyindex, codec(mode,vad), bit-rate, R} indexed by the codec ranking such as in Table 4 and one file per every combination of codec, mode and VAD possible. Periodically, network conditions should be checked by the VoIP client and the QoS (or equivalently, R factor) should be evaluated for that period of time.

The algorithm can be described as:

```
Input: maximum_bitrate ($BW_{max}$) , Minimum_R_value ($R_{min}$), TableCodecs

%loop to delete codecs that do not meet bitrate constraint
for each entry i in TableCodecs do
    if (TableCodec[i]_bitrate > BWmax)
       delete entry;
     endif
endfor
%get current packet loss and delay
evaluate_network_conditions(PL,delay);
%update R for each codec
for each entry i in TableCodecs do
    codec[i]_R = estimateR(codec[i],PL,delay);
endfor
%select the codec that meets Rmin and has the lowest ranking index
for each entry i in TableCodecs do
  if(codec[i]_R > Rmin)
            output = i;
 endif
endfor
Output: codec [i]
```

The algorithm is executed periodically every time network conditions are updated (every 7.6 seconds in our results).

6. Test bed and Implementation Issues

6.1 Scenario set-up

We have tested our algorithm in a test bed. In our test scenario two VoIP Clients (*pjsua* command-line [25]) are inter-connected through a router that emulates network conditions running the *Network Emulator for Windows Toolkit* (NEWT). The VoIP clients send a 30 min conversation playback from recorded telephony conversations (LDC data bank, CallHome recordset) [26], each station playing its respective side of the conversation.

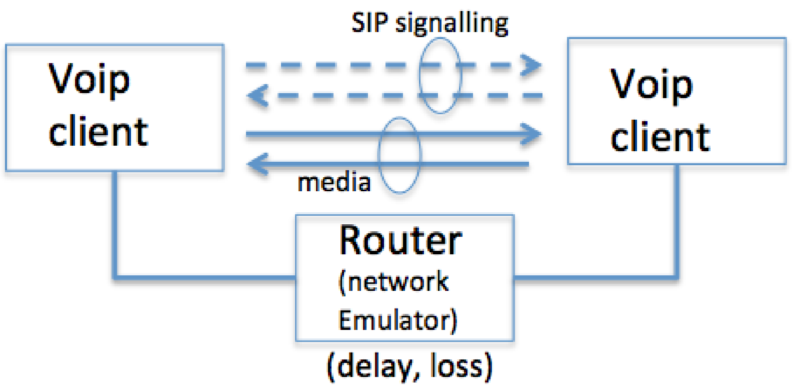

Figure 5. Scenario for the test-bed

6.2 VoIP client implementation

Our VoIP clients contain the modifications necessary to accommodate our algorithm from Section 4. As shown in Figure 4, we have created a module that communicates with the VoIP client. Thus, changes in the original *pjsua* VoIP client are restricted to providing QoS-related information to our module, reading the algorithm output (codec) and sending a SIP RE-INVITE message to its peer so that the new codec can take effect. VoIP clients can be configured to locally apply the new codec selection or to use the codec received from the other side of the call.

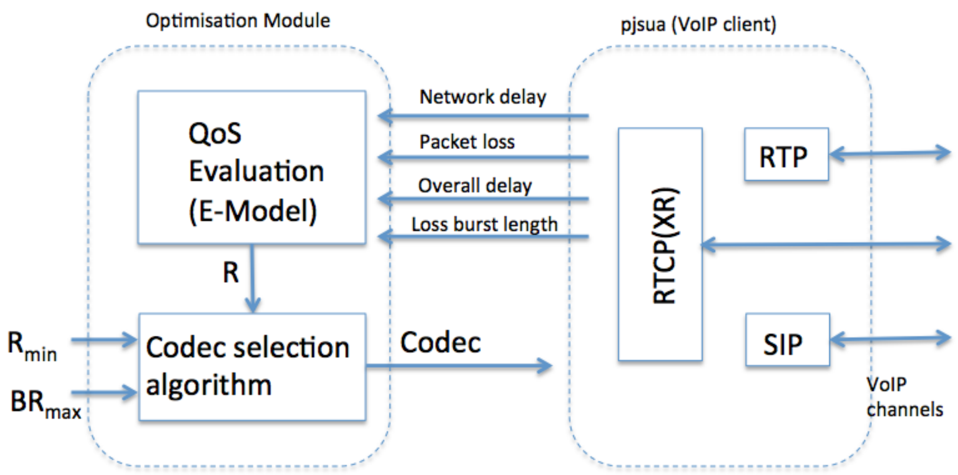

Figure 4. Scheme of the VoIP client implemented

QoS-related information (network delay, packet loss, overall delay and loss burst length) is periodically evaluated by *pjsua* and transported in various RTCP reports such as Extended (XR), Sender (SR) and Receiver (RR). By taking advantage of this functionality already implemented, we can simply use these counters and complement them with some additional locally-obtained application-level information to get the input variables for our algorithm. In contrast to the information included in the RTCP reports, these counters are reset between successive reports which allows for an evaluation of the quality of service for one time interval. The actual period configured in our tests to evaluate QoS variables is 7.64 seconds as a compromise between representative of the actual network conditions and reaction speed.

After obtaining all the input variables from the pjsua client, the optimization module evaluates the new value of *R* for each codec in *TableCodecs*. Note that packet loss variations will change the value of R in a different way for each codec, changing the set of eligible codecs every time that the algorithm is executed, which happens periodically (e.g. every 5 seconds). Notice that each codec will be affected by packet loss in a different way. Figure 4 illustrates this fact, showing that some codecs are more sensitive to packet loss than others.

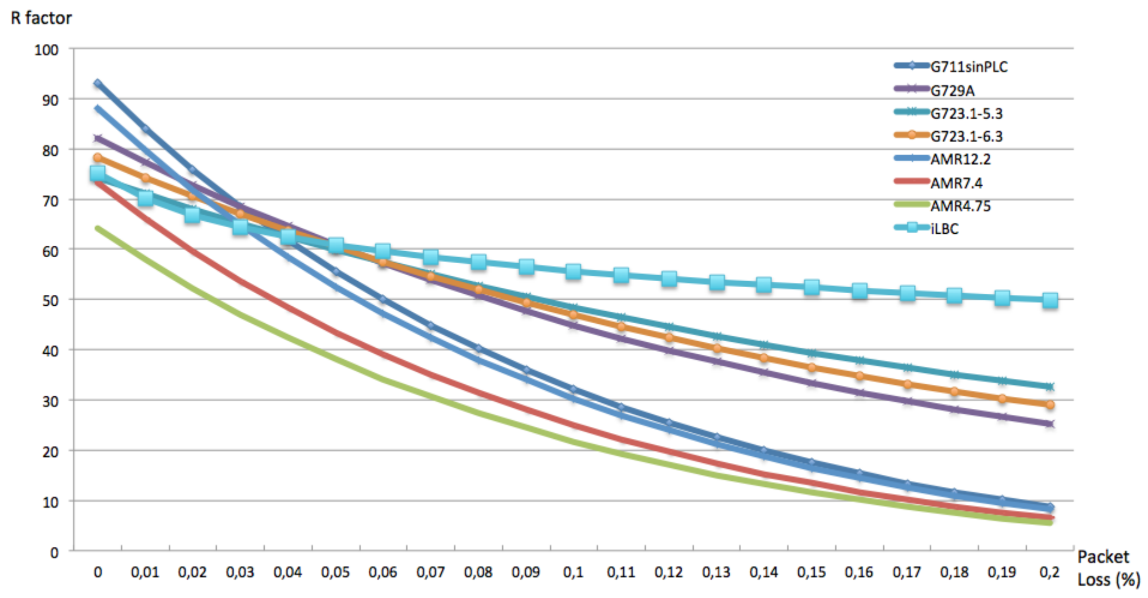

Figure 6. Variation of R with packet loss for each codec on Table 4

7. Numerical Results

Table 4 represents the list of codecs used in our experiment. VAD codecs have not been included since we filter by the codec bit-rate instead of the mean bandwidth.

EnergyIndex	codec	Bit-rate (bps)
0	G711	64
2	ILBC30	13,3
4	G729a	8
10	AMR7.4	7,4
11	AMR4.75	4,75
12	AMR12.2	12,2
15	G723_53	5,3
16	G723_63	6,3

Table 4. TableCodecs used in our results (without R column)

Tables 5 and 6 show the results of our codec selection algorithm for different constraints of bit-rate and a minimum R value of 65. The results assume a constant *burstR* value of 2, and delays and loss rate are as set on the network emulator (4ms of network delay in Table 5, and 100ms of network delay in Table 6).

	Packet Loss rate (network delay = 4ms)					
(BWmin)	0%	1%	2%	3%	4%	5%
≤ 64kb/s	G711	G711	G711	G711	G729A	-
≤ 15kb/s	iLBC30	iLBC30	iLBC30	iLBC30	G729A	-
≤ 10kb/s	G729A	G729A	G729A	G729A	G729A	-
≤ 7.5kb/s	AMR7.4	AMR7.4	G723.1-5.3	G723.1-5.3	G723.1-6.3	-
≤ 5.5kb/s	AMR4.75	G723.1-5.3	G723.1-5.3	G723.1-5.3	-	-

Table 5. Algorithm output under different constraints of bitrate and packet loss conditions (network delay 4ms)

	Packet Loss rate (network delay = 100ms)					
(BWmin)	0%	1%	2%	3%	4%	5%
≤ 64kb/s	G711	G711	G711	G711	-	-
≤ 15kb/s	iLBC30	iLBC30	iLBC30	iLBC30	-	-
≤ 10kb/s	G729A	G729A	G729A	G729A	-	-
≤ 7.5kb/s	AMR7.4	AMR7.4	G723.1-5.3	G723.1-6.3	-	-
≤ 5.5kb/s	G723.1-5.3	G723.1-5.3	G723.1-5.3	-	-	-

Table 6. Codec selection under different constraints of bitrate and packet loss (network delay 100ms)

As shown in Tables 5 and 6, different codecs are chosen according to network conditions variations and bitrate constraint. G.711 is the more energy-friendly codec but exhibits the highest bandwidth. It is followed by iLBC codec, and G.729A codec. However, under high packet loss, G.729A exhibits a better QoS score than iLBC or G.711. When low bit-rate codecs are necessary, AMR and G.723.1 codecs are selected, for low and medium-high loss rates respectively.

Finally, we would like to quantify the benefits of our prosed algorithm in terms of battery life duration. Taking the data from Table 3 (note that it considers only the processor component) associated to each codec, we compare the battery life duration of the best (our algorithm output) and the works codec possible (the lowest *energyindex*) among those that meet the QoS and bit-rate constraint in the results from Tables 5 and 6. Table 7 shows the results obtained in terms of maximum battery life duration savings (i.e. difference between the best and worst codec possible and feasible in each case). Note that we only consider the power consumption attributable to the use of the processor and that G.711 is assumed to provide a 100% of saving with respect to any other codec.

bit-rate	PL 0% (4ms)	PL0% (100ms)	PL 1% (4ms)	PL 1% (100ms)	PL 2% (4ms)	PL 2% (100ms)	PL 3% (4ms)	PL 3% (100ms)	PL 4% (4ms)	PL 4% (100ms)
≤ 64kb/s	100%	100%	100%	100%	100%	100%	100%	100%	57%	-
≤ 15kb/s	85%	85%	85%	85%	85%	85%	85%	85%	57%	-
≤ 10kb/s	57%	57%	57%	57%	57%	57%	57%	57%	57%	-
≤ 7.5kb/s	54%	54%	54%	54%	15%	23%	23%	0%	0%	-
≤ 5.5kb/s	19%	0%	0%	0%	0%	0%	0%	-	-	-

Table 7. Maximum battery life savings with the algorithm according to data from Table 3

Results from Table 7 shows that can significant battery life savings can be obtained by the use of our algorithm with respect to the worst choice. Obviously, less constrained conditions in bit-rate or packet loss increases the set of potential codecs to be used which increments the range of potential savings.

8. Conclusions and further work

The codec's demand of processor has influence in the battery life. Different codecs show significant differences in this demand which should be taken into consideration by software developers and VoIP users. We have done a study that offers a raking of codecs according to their demand of processor for the iphone4S smartphone. This ranking has been utilized in an algorithm implemented in VoIP clients to dynamically select the most energy-friendly codec in a conversation constrained to a minimum QoS and a maximum bit-rate usage. Results show that all codecs can be useful under certain conditions, although G.711, iLBC and G.729A provide best results in their bit-rate is acceptable to the user.

We are presently working on extending our work to various smartphones with iOS and Android devices and also including the new OPUS codec. This would let us reaffirm that our results can be extended to any arm smartphone processors. We will also intend to include other application-level factors such as packet size (i.e. number of frames encapsulated on each IP packet) in the algorithm.

Acknowledgement

This research is funded by the Spanish Ministry of Science and Innovation (grant TEC2010-20861) and the Andalusian Regional Government (grant TIC-6339.)

References

[1] R. G. Cole and J. H. Rosenbluth, "Voice over ip performance monitoring," SIGCOMM Comput. Commun. Rev., vol. 31, no. 2, pp. 9–24, Apr. 2001. http://dx.doi.org/10.1145/505666.505669

[2] A. J. Estepa, J. M. Vozmediano, J. López, and R. M. Estepa, "Impact of voip codecs on

the energy consumption of portable devices," in Proceedings of the 6th ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks. ACM, 2011, pp. 123–130. http://dx.doi.org/10.1145/2069087.2069104

[3] Sfairopoulou, A.; Bellalta, B.; Macian, C., "How to tune VoIP codec selection in WLANs?," Communications Letters, IEEE , vol.12, no.8, pp.551,553, Aug. 2008. http://dx.doi.org/10.1109/LCOMM.2008.071998

[4] Bonfiglio, D.; Mellia, M.; Meo, M.; Rossi, D., "Detailed Analysis of Skype Traffic," Multimedia, IEEE Transactions on , vol.11, no.1, pp.117,127, Jan. 2009. http://dx.doi.org/10.1109/TMM.2008.2008927.

[5] P. Serrano, A. Garcia-Saavedra, M. Hollick, and A. Banchs. On the energy efficiency of ieee 802.11 wlans. In Wireless Conference (EW), 2010 European, pages 932 –939, 2010. http://dx.doi.org/10.1109/EW.2010.5483505

[6] 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, MobiSys '07, pages 179–191, New York, NY, USA, 2007. ACM. http://dx.doi.org/10.1145/1247660.1247682

[7] S.-L. Tsao and C.-H. Huang. An energy-efficient transmission mechanism for voip over ieee 802.11 wlan. Wirel. Commun. Mob. Comput., 9:1629–1644, December 2009. http://dx.doi.org/10.1002/wcm.747

[8] H. Schulzrinne and S. Casner. RTP Profile for Audio and Video Conferences with Minimal Control. RFC 3551 (Standard), July 2003. Updated by RFC 5761

[9] A. Benyassine, E. Shlomot, H.-Y. Su, D. Massaloux, C. Lamblin, and J.-P. Petit. Itu-t recommendation g.729 annex b: a silence compression scheme for usewith g.729 optimized for v.70 digital simultaneous voice and data applications. IEEE Communications Magazine, 35(9):64 –73, Sept. 1997

[10] A. Trad, F. Munir, and H. Afifi, "Capacity evaluation of voip in ieee 802.11e wlan environment," in Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE, vol. 2, 2006, pp. 828–832. http://dx.doi.org/10.1109/CCNC.2006.1593155

[11] L. Cai, X. Shen, J. W. Mark, and Y. Xiao, "Voice capacity analysis of wlan with unbalanced traffic," in Quality of Service in Heterogeneous Wired/Wireless Networks, 2005. Second International Conference on, 2005, pp. 8 pp.–9. http://dx.doi.org/10.1109/QSHINE.2005.64

[12] S. Shin and H. Schulzrinne, "Measurement and analysis of the voip capacity in ieee 802.11 wlan," Mobile Computing, IEEE Transactions on, vol. 8, no. 9, pp. 1265–1279, 2009. http://dx.doi.org/10.1109/TMC.2009.49

[13] G. Kuriakose, S. Harsha, A. Kumar, and V. Sharma, "Analytical models for capacity estimation of ieee 802.11 wlans using dcf for internet applications," Wirel. Netw., vol. 15, no. 2, pp. 259–277, Feb. 2009. http://dx.doi.org/10.1007/s11276-007-0051-8

[14] G. Nikolakopoulos, A. Panousopoulou, and A. Tzes, "Experimental controller tuning and qos optimization of a wireless transmission scheme for real-time remote control applications," Control Engineering Practice, vol. 16, no. 3, pp. 333 – 346, 2008. http://dx.doi.org/10.1109/ICIT.2004.1490177

[15] K. Stoeckigt and H. Vu, "Voip capacity analysis in ieee 802.11 wlan," in Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on, 2009, pp. 116–123. http://dx.doi.org/10.1109/LCN.2009.5355192

[16] K. Thambu, X. N. Fernando, and L. Guan, "Transmit Rate Reduction of Wyner-Ziv Video Coding (WZVC) using Multiple Uncorrelated Wireless Receivers", Proceeding of 26th Biennial Symposium on Communications (QBSC 2012), May 27-29, Canada. pp. 174-177. http://dx.doi.org/10.1109/QBSC.2012.6221376

[17] L. S. G. D. Carvalho and E. D. S. Mota, "Survey on application-layer mechanisms for speech quality adaptation in voip," ACM Computing Surveys (CSUR), vol. 45, no. 3, p. 36, 2013. http://dx.doi.org/10.1145/2480741.2480753

[18] T. Shiao-Li and H. Chung-Huei, "A survey of energy efficient mac protocols for ieee 802.11 wlan," Computer Communications, vol. 34, no. 1, pp. 54 – 67, 2011. http://dx.doi.org/10.1016/j.comcom.2010.09.008

[19] Radnosrati, Kamiar, Dmitri Moltchanov, and Yevgeni Koucheryavy. "The Choice of VoIP Codec for Mobile Devices." The Thirteenth International Conference on Networks. (ICN 2014), Nice, France. February 23 - 27, 2014.

[20] Naeem, Muhammad Naeem, Namboodiri, Vinod Namboodiri, Pendse, Ravi Pendse, "Energy implication of various VoIP codecs in portable devices," lcn, pp.196-199, 2010 IEEE 35th Conference on Local Computer Networks, 2010. http://dx.doi.org/10.1109/LCN.2010.5735699

[21] I.-T. G.107, "The e-model, a computational model for use in transmission planning," ITU-T Recommendation G.107, 2009.

[22] A. Kovac, M. Halas, M. Orgon, and M. Voznak, "E-model mos estimate improvement through jitter buffer packet loss modelling," Advances in Electrical and Electronic Engineering, vol. 9, no. 5, pp. 233–242, 2011

[23] H. Assem, M. Adel, B. Jennings, D. Malone, J. Dunne, P. O'Sullivan, "A Genetic Algorithm for Mid-call Audio Codec Switching", IFIP/IEEE IM2013 Workshop: 1st International Workshop on Quality of Experience Centric Management (QCMan), pp 1276-1281.

[24] A. Estepa, R. Estepa, and J. M. Vozmediano, "On the suitability of the e-model to voip networks," in Computers and Communications, 2002. Proceedings. ISCC 2002. Seventh International Symposium on. IEEE, 2002, pp. 511–516. http://dx.doi.org/10.1109/ISCC.2002.1021723

[25] S. PJSIP-Open Source, "Stack and media stack for presence, instant messaging, and

multimedia communication, 2008.”. At http://www.pjsip.org (last accessed June 30, 2014)

[26] C. Alexandra, D. Graff, and G. Zipperlen, “Callhome american english speech,” Linguistic Data Consortium, Philadelphia, 1996