**Dissertação: Estado d’Arte**

Conteúdo

[Sumário: Quadro Comparativo – Concorrência 3](#_Toc468239417)

[Tecnologias a considerar: 5](#_Toc468239418)

[Testes/Tarefas de Probing: 7](#_Toc468239419)

[Apps concorrentes: 7](#_Toc468239420)

[Tarefas da Probe Intrusiva: 8](#_Toc468239421)

[Common Output Parameters: 8](#_Toc468239422)

[Tarefas Principais: 8](#_Toc468239423)

[Tarefas Secundárias: 10](#_Toc468239424)

[Info Parameters: 12](#_Toc468239425)

[Apps Concorrentes: 12](#_Toc468239426)

[ARQOS Radio Parameters 15](#_Toc468239427)

[Android Network Info Parameters and where to find them: 18](#_Toc468239428)

[Android Device Info Parameters and where to find them: 19](#_Toc468239429)

[Android 20](#_Toc468239430)

[Gravar e Reproduzir Audio de chamadas 20](#_Toc468239431)

[Permissões 20](#_Toc468239432)

[O que varia com as API’s 21](#_Toc468239433)

[API 4.4 (Kitkat) 21](#_Toc468239434)

[API 5.0 (Lolipop) 21](#_Toc468239435)

[API 6.0 (Marshmallow) 21](#_Toc468239436)

[API 7.0 (Nougat) 22](#_Toc468239437)

[VOLTE: 23](#_Toc468239438)

[Jio4GVoice 23](#_Toc468239439)

[Vantagens do VoLTE 23](#_Toc468239440)

[Notas do VoLTE 23](#_Toc468239441)

[Telemóveis compatíveis com VoLTE: 23](#_Toc468239442)

[Best smartphones that support VoLTE 26](#_Toc468239443)

[VoWifi 27](#_Toc468239444)

[VoWifi – Vantagens para o utilizador final 27](#_Toc468239445)

[VoWifi – Vantagens para o operador 27](#_Toc468239446)

[VoWifi – Quais são as possíveis desvantagens? 27](#_Toc468239447)

[Chamadas Wifi e a Qualidade de Serviço 27](#_Toc468239448)

[Chamadas Wifi e as chamadas de emergência 28](#_Toc468239449)

[Terminais móveis que suportam: 28](#_Toc468239450)

[Audio 29](#_Toc468239451)

[Fatores consideráveis para a qualidade de serviço VoIP 29](#_Toc468239452)

[Métodos de avaliação de desempenho 30](#_Toc468239453)

[Teste: Desempenho VoIP: Qualidade de Voz em Redes IP 30](#_Toc468239454)

[48K / Audio HD 31](#_Toc468239455)

[IVR: 32](#_Toc468239456)

[Comparar qualidade de video 33](#_Toc468239457)

[Why you cannot simply compare bitrate/resolution/etc: 33](#_Toc468239458)

[There are different types of video quality metrics! 33](#_Toc468239459)

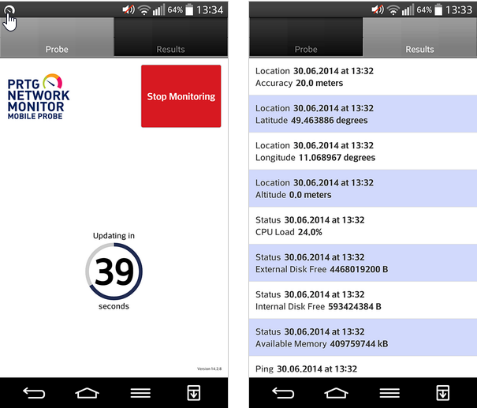
[Other Metrics/Measures 33](#_Toc468239460)

[Summary 34](#_Toc468239461)

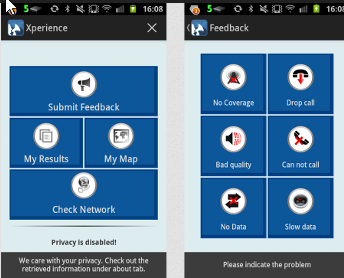
[Video Encoding Recommendations 34](#_Toc468239462)

[CamcorderProfile 35](#_Toc468239463)

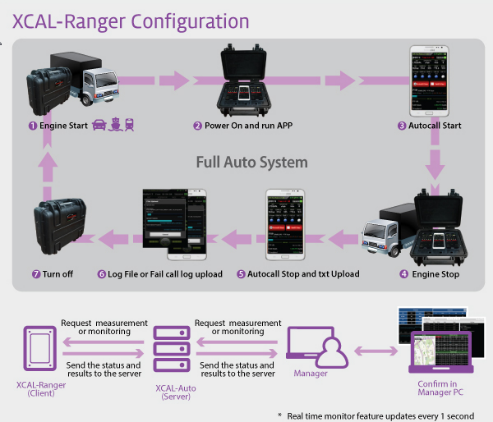
# Sumário: Quadro Comparativo – Concorrência

**PRTG:**

* App na playstore(lite version)
* Métricas de wi-fi (speed, força do sinal, …)
* Testes: ping, load time of pages, HTTP
* Device status (CPU, battery, disk space)
* Location and tracking of device

**Xperience (Streambow):**

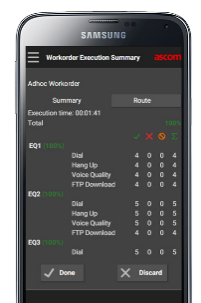
* App na playstore(lite version)
* Métricas de wi-fi (speed, força do sinal, …)
* Testes: ping, load time of pages, HTTP
* Device status (CPU, battery, disk space)
* Location and tracking of device

**Xcal-Ranger (Accuver):**

* Instalação em veículos
* Recuperação de erros e Error report
* Suporta Voice, VoLTE, FTP, HTTP, Youtube
* Schedule para programação de testes
* Log files, upgrades automáticos

**Nemo Handy:**

* Nokia Only
* Testes (HTTP, FTP, SMS, ping)
* Tecnologias: GSM, CDMA, HSDPA, EVDO, LTE, HSPA



**TEMS (Ascom):**

* 14 devices Only
* Dial, Hang Up, Voice Quality, FTP Download/Up, HTTP, Ping, Idle

**Amanzitel – Customer IQ (Dinglicom):**

* Android e IoS
* full layer 3 information

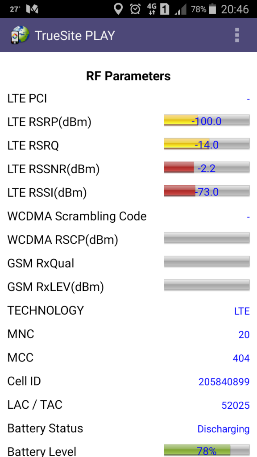
**XGMA SP (focus infocom):**

* Only Android
* Tec: GSM, GPRS, EDGE, UMTS, HSPA, HSUPA, LTE, Het Net
* Testes:

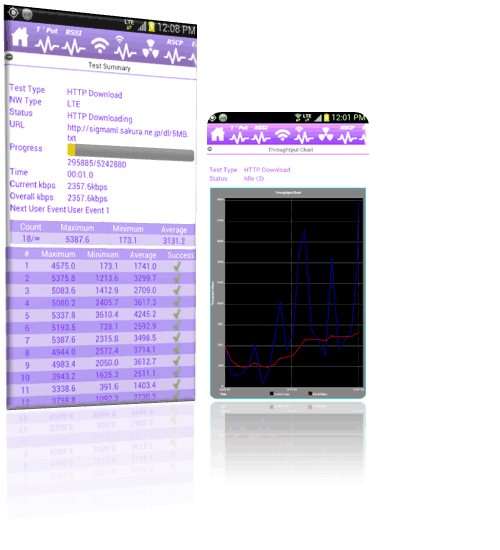
|  |  |
| --- | --- |
| **Voz** | **Dados** |
| MOC, MTC, Multi-Rab | Web browsing, HTTP D/Upload |
| PESQ, POLQA, VOIP(Skype) | FTP “”, UDP, Email, SMS, MMS |
| Voicemail, VoLTE | Video stream, music, mobile TV |

**Keynote (SIGOS):**

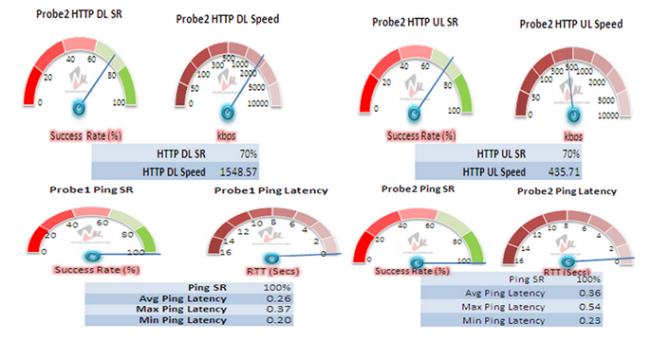
* App no GooglePlay (só testa internet speed)
* Qualidade de voz 3G, SMS, CSFB, VoLTE, HTTP, FTP, Streaming, QoVideo/Voz, speedtest
* Android e IoS

**The TrueSite Solution (VIAVI):**

* Controlador central de 6 devices
* Deteta falhas/ Ajuda a recuperar de falhas
* Testes discretos, real time
* Android 4.1+
* Configuração de sequências de testes
* Testa: FTP, HTTP, CSFB-circuit, LTE, 3G, E-MAIL, Force band, TCP, UDP bandwith, MMS, SMS, PING, Traceroute, video-voice/VoLTE

**Sigma -ML (Meritch):**

* Troubleshoot mobile wireless
* Testes (nada a acrescentar aos testes acima)
* Exportação CSV c/ localização
* Only Android
* Configuração de sequências de testes
* All layers info

****

**FalconProbe (Falcon smart):**

* Tec: GSM, EDGE, UMTS, HSPA, CDMA, WIFI, WIMAX, LTE
* Voice, Video Conference, Streaming, email, messages, FTP
* Dispositivos numa box, podem ser colocados em pontos críticos da rede
* Android 2.2+

**QualityProc android Swissqual (Rohde-schwarz):**

* Caixinha para devices / Controlo remoto
* 24/7 problem identification
* Voice (POLQA, PESQ, SQuad8)
* VoLTE tb c/ data, Video streaming/quality, apps(fb,dropbox)
* Envia resultados de todos os devices para backend
* Descodifica L3 text messages, RTP packets
* Tec: GSM, GPRS, EDGE, WCDMA, HSUPA, HSDPA, LTE, 3GPP, L2, L3, TCP/IP, IMS, SIP  
    
  <http://www.swissqual.com/en/products/optimization2/qualipoc-android/>

**SQlive (Telchemy):**

* MOS for VOIP
* Packet loss, jitter, delay, throughput
* Tec: 4G, LTE, Video/Voice, IPTV, Stream, Video Conference

# Tecnologias a considerar:

* GSM/EDGE/UMTS/HSPA/CDMA/Wifi/ WiMAX/LTE, GPRS, WCDMA, HSUPA, HSDPA DC, LTE-FDD & TD-LTE, CDMA2000®, EVDO Rev.0/A.

Limiar de QoS:

**Acesso à Rede**: Indicação de que a rede móvel está presente

**Acesso ao Serviço**: Quando o utilizador pretende utilizar um serviço, o operador móvel deve disponibilizar (ex. estabelecer uma chamada de voz);

**Integridade do Serviço:** Qualidade do Serviço durante a sua utilização (*v.g.* Qualidade Áudio durante uma chamada de voz);

**Manutenção do Serviço**: Forma como termina a utilização do serviço (de acordo ou contra a vontade do utilizador).

**Disponibilidade da Rede (cobertura):**

A disponibilidade da rede é a probabilidade dos serviços móveis estarem disponíveis para um utilizador (cobertura radioeléctrica das redes). 

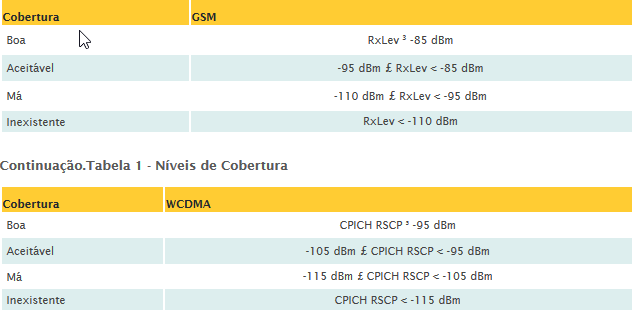


Figura 1: Limiar de QoS

# Testes/Tarefas de Probing:

## Apps concorrentes:

* FTP download/upload,
* HTTP download
* Ping test
* E-mail send/receive
* Circuit switch fall back (CSFB) — key for evaluating LTE and 3G service
* Force band, Force RAT/PCI
* iPERF — measures maximum TCP and UDP bandwidth
* MMS — tests if an MMS can be sent to itself within a certain time
* MT voice — tests for incoming calls
* PDPACT — performs a PDP activation
* PING — test simple network connectivity, round-trip time, and PING connectivity
* POLQA voice quality test (refer to the configuration guide or contact
* RFTOGGLE — enables airline mode for the configured period
* SMS — tests if an SMS can be sent to itself within a certain time
* TECHCHK — checks if the device is registered to a specific technology such as LTE or WCDMA
* Trace RT/Trace RT change —determines the route taken to a destination by sending Internet control message protocol (ICMP) echo packets with varying time-to-live (TTL) values to the destination.
* Video — displays video on any available network including WiFi
* Voice — includes a universal voice test that includes a circuit switch and VoLTE modes
* LAN Scan – see PC-name of users using the wi-fi network

|  |  |  |
| --- | --- | --- |
| Testes | Corre em  Background | Pede permissões |
| *FTP Download/Upload* | x | INTERNET and WRITE\_EXTERNAL\_STORAGE); also ACCESS\_NETWORK\_STATE para ver conetividade |
| *HTTP Download/Upload* | x | INTERNET |
| *Ping* | x | ACCESS\_NETWORK\_STATE |
| *E-mail send/receive* | x | INTERNET and also ACCESS\_NETWORK\_STATE |
| *CSFB* | x | “” |
| *Force-band* | Não | Needs root e é para telemóveis (Nexus 5,6x…) |
| *TCP/UDP bandwith* | x | Sim |
| *Calls* | VER | CALL\_PHONE |
| *MMS e SMS* | Intent | Sim, mas dá para mandar em background |
| *Traceroute* | x | ACCESS\_NETWORK\_STATE |
| *PDPACT* |  |  |
| *RFTOGGLE* | x | Sim |
| TECHCHK | x | “” |
| VoLTE | x |  |
| *Sequence loop* | Depende dos testes | As dos testes feitos… |

## Tarefas da Probe Intrusiva:

No PDF – SupportedTasks.

### Common Output Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Common Output Parameters | Notes | Format Type | Added in API |
| *Macro ID* | In tests programmed by the management system, this ID identifies the macro to which the task belongs inside the test.  In test scripts this parameter is always zero. | Int |  |
| *Task Number* | Number of the task inside the macro. The combination of Macro ID with the Number  of the task is unique inside a test. | Int |  |
| *Task start date* | Não consegui ver formato, por não ter acesso à pagina dado pelo link no pdf | Date (String) |  |
| *Task end date* | Não consegui ver formato, por não ter acesso à pagina dado pelo link no pdf | Date (String) |  |
| *ICCID* | ICCID of the used SIM card.  Class TelephonyManager – método getSimSerialNumber ()  No caso de "Multiple SIM Card Support" for [Android 5.1](https://developer.android.com/intl/pt-br/about/versions/android-5.1.html) (API 22). Instead of using getSimSerialNumber(), use [SubscriptionManager](https://developer.android.com/intl/pt-br/reference/android/telephony/SubscriptionManager.html) class. | String | 1  22 |
| *GPS Information* | informações de localização são dados pela class Location  Não consegui ver mais detalhes do formato, link pdf… mas | Ver link pdf | 1 |
| *Task success* | Ver pdf … ended\_with\_sucess() / get\_error() | String |  |

### Tarefas Principais:

|  |  |  |
| --- | --- | --- |
| Main Tasks | Notes | Output Parameters |
| *Make Voice call* | Ok, permissão CALL\_PHONE no manifest | Fixed value(int: value-1)  Destination number(string) Call setup time(int - miliseconds)  Audio recording file(string) |
| *Start data call* | É preciso root no device | -- |
| *Answer voice call* | Ok, Ajuda: <http://stackoverflow.com/questions/26924618/how-can-incoming-calls-be-answered-programmatically-in-android-5-0-lollipop> | Caller number(string - MSISDN)  Fixed value(int: value-1)  Audio recording file(string)  Time spent waiting for RING since start(int - miliseconds) |
| *Hang up voice call* | Ok | Call duration(int - miliseconds)  VoIP parameters…  Fixed value(int: value-1) |
| *Suspend active cal* | Not in standart devices, mas a Class Call tem metodo para isso |  |
| *Retrieve call on-hold* | Not in standart devices, mas a Class Call tem metodo para isso |  |
| *Answer waitting call* | Igual a Answer voice call ? |  |
| *Switch active/suspended call* | Not in standart devices |  |
| *Setup three party conference* | Not programatically, needs to be system app | --- |
| *Reject call* | Ok | Caller number(string - MSISDN) |
| *Call Forwarding* | Não dá | --- |
| *Send SMS* | Ok, SmsManager added in API level 4 permissão SEND\_SMS, added in API level 1 | Time spent delivering to the MMSC – protected by Android  Destination number(string)  Text message(string) |
| *FTP Download* | Ok, Ajuda:  import org.apache.commons.net.ftp.FTP;  import org.apache.commons.net.ftp.FTPClient; | Local name of downloaded file  Downloaded file URL(string-url)  --  Lista de métodos aqui (apache): <https://commons.apache.org/proper/commons-net/apidocs/org/apache/commons/net/ftp/FTPClient.html> |
| *FTP Upload* | Ok | “” |
| *HTTP Download* | Ok, 1.Assync Task, 2.Download from Service or 3.Use Android class – DownloadManager(handle threads, streams, etc with GingerBread) | Local name of downloaded file  (string “empty” or “filename”)  Data size (float - bytes)  Download time (float - seconds)  Accessed URL(string - url) |
| *HTTP Upload* | Ok | “” |
| *Receive SMS* | Ok, documentado na api 19 aqui: <https://developer.android.com/reference/android/provider/Telephony.Sms.Intents.html#SMS_RECEIVED_ACTION> | Sms message(string)  Sender number(string-MSISDN)  SMSC number – is not permited  Delivery time – not directly  SMS enconding (0-GSM 7-bit  8-bit // 2- USC2(16-bit)  See Constants of class SmsMessgae added in Api 4) |
| *Ping* | Ok, some solutions:  <http://stackoverflow.com/questions/3905358/how-to-ping-external-ip-from-java-android> | Minimum ICMP latency  Average ICMP latency (float)  Maximum ICMP latency  Destination IP address or name  IP obtained from DNS lookup  Operation log filename. (string-IP)  Source ip adress |
| *Record áudio* | No acess to in-call audio, algumas aplicaçoes fazem-no… | Audio file name(string) |
| *Play audio* | Não dá para rep. em chamada | -- |
| *Send MMS* | Sim, dá | File name(“empty” or <filename>)  Destination number |
| *Receive MMS* | Not directly supported and is mostly undocumented  <https://groups.google.com/forum/#!topic/android-developers/0MFewXwSrw4> | Matched expected text(string)  Sender number(string)  File name(“empty” or <filename>) |
| *Send USSD* | Sim, Ajuda: <http://umeshisran4android.blogspot.pt/2015/11/how-to-readparse-ussd-messages.html> | Response to USSD(string) -> parse is not direct  USSD request(string) |
| *Setup radio parameters* | Force band, não dá | --- |
| *Send UDP/TCP raw IP* | Sim, <https://guides.codepath.com/android/Sending-and-Receiving-Data-with-Sockets> | Number of transmited packets(int) |
| *Receive UDP/TCP raw IP* | Sim, com TrafficStats(Android class) só é possível saber numero de bytes/pacotes transmitidos/recebidos desde que o dispositivo deu boot. [Added in API level 8] | ver que parametros de qos se consegue obter…(delay, jitter, packet loss)  Number of received packets  Source test module ID |
| *Send email* | Ok | Depende da implementação parâmetros da probe: <http://i.imgur.com/Y76ThHo.png> |
| *Receive email* | Ok | “” |
| *Traceroute* | Ok Ajuda:  <https://github.com/olivierg13/TraceroutePing/tree/master/src/com/og/tracerouteping> | Number of hops (int)  Destination IP address(string) |
| *Send DTMF* | *Sim, apartir da api 23 Call Class* | Sequence of DTMF tones |
| *Wait* | *Sim, System.sleep* | Caller number(string) |
| *Setup Internet Connection* | *Sim* | Ip address  Mask  Gateway  Primary and secondary dns |
| *Turn On WiFi* | Ok, através do WiFiManager | Sem parametros |
| *Turn Off Wifi* | Ok | “” |
| *Associate WiFi* | Ok, Ajuda: <http://stackoverflow.com/questions/8818290/how-do-i-connect-to-a-specific-wi-fi-network-in-android-programmatically> | Parametros em: WifiInfo API 1:  AP BSSID(string – MAC address)  Channel frequency(int - Mhz)  Connection bitrate(int - Mbps)  RSSI(int - dBm)  Ip address  Mask  primary/secundary DNS |
| *Deassociate WiFi* | Ok, cuidado ao utilizar TrafficStats dados são contados desde o boot do dispositivo. | Number of packets received(int)  Number of bytes received(bytes)  Number of packets transmitted  Number of bytes transmitted  RSSI(int - dBm) |
| *Scan WiFi networks* | Sim, com BroadcastReceiver | Detected networks |
| *Execute Script* | *Restrições das tarefas a executar* | <http://i.imgur.com/z2Pz4NM.png>  <http://i.imgur.com/IYyTjNQ.png> |

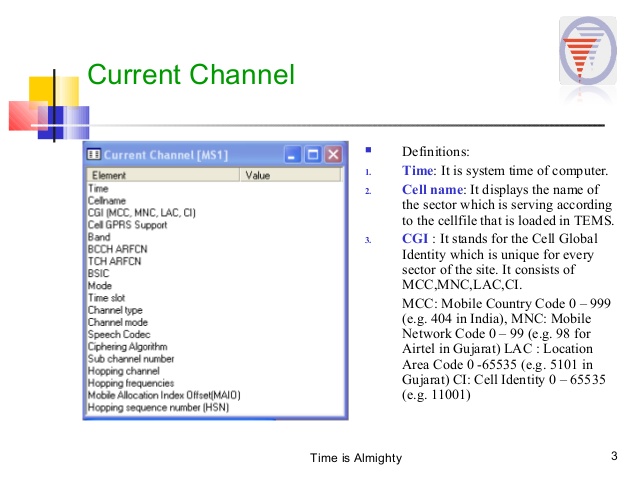
Tarefas Secundárias:

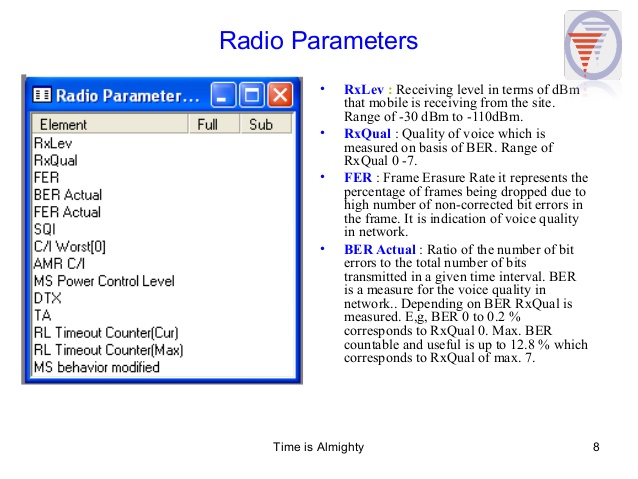
|  |  |  |
| --- | --- | --- |
| Non Prioritary Tasks | Restrições | Parametros |
| *Make WAP call* | ??? |  |
| *Start Custom call with APN number and authentication parameter* | Now, to add or edit APNs you can't just put that code in any app. You need android.permission.WRITE\_APN\_SETTINGS which is a permission only available to system apps. The app won't be portable, but it appears this can be done using a rooted phone or a custom rom.  Mas.. testar esta solução: <http://stackoverflow.com/questions/7867079/android-apn-enforcement> |  |
| *Measure Amplitude* | ------ |  |
| *Generate frequency* | Not during the call |  |
| *Measure distorcion* | …, Not acess to n-call audio data |  |
| *Measure noise* | ------ |  |
| *Measure voice delay* | “” |  |
| *Measure error rate* | Não dá |  |
| *Measure WAP access time* |  |  |
| *Measure attach/detach time (GPRS)* |  |  |
| *PESQ/POLQA analisys - Source* | Não dá para reproduzir áudio na chamada |  |
| *PESQ/POLQA analisys – Destination* | No acess to in-call audio, algumas aplicaçoes fazem-no… |  |
| *Audio detection* | Not acess to n-call audio data |  |
| *Portal login* | *Depende das especificações, mas sim* |  |
| *Portal logout* | “” |  |
| *VoIP Transfer Call* | Not in standart devices |  |
| *Connect to a telnet server* | Ssh?, Ver JuiceSSH… |  |
| *Start PPPoE call* | Não dá |  |
| *Send messages on IM service(ex: Hangouts)* | Penso que sim, ver + à frente |  |
| *Receive messages on IM service(ex: Hangouts)* | Penso que sim, ver + à frente |  |
| *Iperf TCP client/server* | Vi app para… |  |
| *Iperf UDP client/server* | Vi app para… |  |
| *Iperf TCP bidirectional client* | Vi app para… |  |
| *Iperf UDP bidirectional client* | Vi app para… |  |
| *Torrents, p2p file sharing* |  |  |
| *Verify TCP availability using TCP Syn Scan* | Ver… em principio dará (sockets) |  |
| *Acess video streaming based on Microsoft implementation* | Sim, ver… |  |
| *Acess video streaming (progressive streaming)* | ver |  |
| *Login on Meo Go video streaming* | Sim, need more detail to deploy |  |

# Info Parameters:

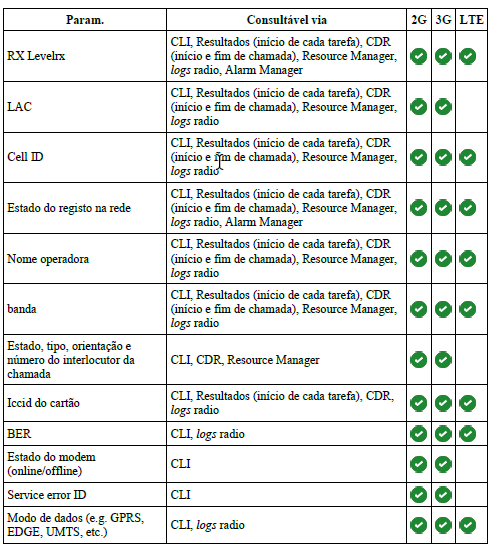
## Apps Concorrentes:

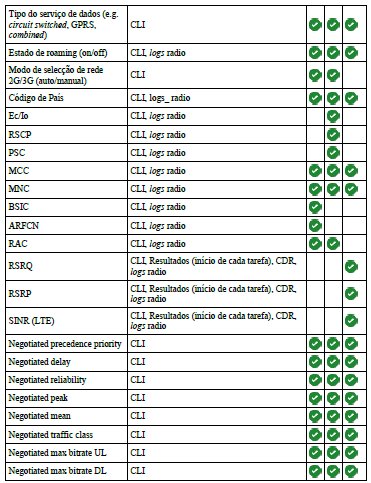
* LTE:
  + PCI
  + RSRP(dBm)
  + RSRQ
  + RSSNR (dBm)
  + RSSI (dBm)
* WCDMA:
  + Scrambling Code
  + RSCP (dBm)
* GSM:
  + RxQual
  + RxLEV (dBm)
* TECHNOLOGY
* MNC
* MCC
* Cell ID
* LAC / TAC
* Device:
  + Tarefas
    - Memória
    - Memória livre
  + Battery Staus
  + Battery Level
    - Nivel
    - Condição
    - Tecnologia
    - Voltagem
    - Temperatura
  + Latitude
  + Longitude
    - Precisao
  + Orientação/Direção
* Satellites Visible
* Satellites Tracked
* Speed (m/s)
* Height
* Connection:
  + Default Gateway IP
  + DNS Server IP
  + External IP
  + External IPV6
  + HTTP Proxy
* WI-FI info:
  + If is enabled
  + Data State (Connected or not)
  + HandShake State (Completed/not completed)
  + SSID (ex: eduroam)
  + BSSID (ip…)
  + Vendor (ex: CISCO Systems, Inc)
  + Ip Address
  + SubnetMask
  + IPV6 Address
  + MAC
  + Speed
  + Signal Strengh (dBm)
  + Frequency ??
  + Channel
  + Acess point hidden
  + DNS secondary
  + DHCP Server
  + DHCP lease (ex: 14400 s)
* Cell information:
  + NetworkType/ Tecnhology
  + Data State
  + Data Activity
  + Roaming
  + IP Address
  + IPV6 Adress
  + Operator name
  + MCC/MNC
  + Signal Strenght (dBm)
  + Phone Type (GSM) ??
  + SIM State (ex: Ready)
  + SIM Operator Name
  + SIM MCC/MNC (ex: 268/06 (pt))

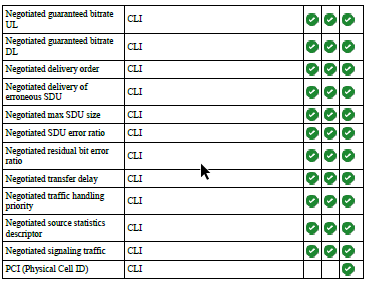


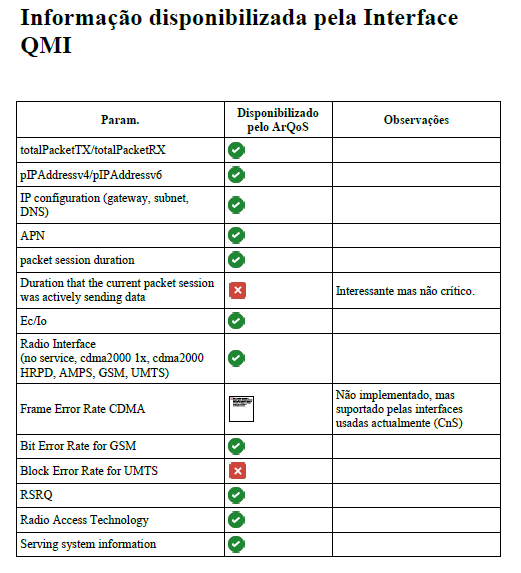
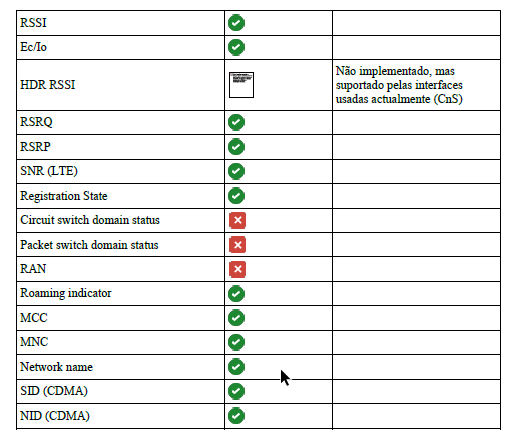


## ARQOS Radio Parameters







## Android Network Info Parameters and where to find them:

|  |  |  |
| --- | --- | --- |
| Parameters | Notes / How to retrieve | Added in API |
| Rx Level (Receive Signal Strength) | Yes, see Android SDK, the class: CellSignalStrength | 17 |
| Tx Level | No, It's impossible |  |
| LAC | <https://developer.android.com/reference/android/telephony/CellIdentityGsm.html> getLac() | 17 |
| Cell ID | “” getCid() | 17 |
| Nome operadora | TelephonyManager -> [getNetworkOperatorName()](http://developer.android.com/reference/android/telephony/TelephonyManager.html#getNetworkOperatorName%28%29) | 1 |
| Banda | TelephonyManager -> getDataNetworkType() ou getNetworkType() | 24 , 1 |
| |  | | --- | | **Estado, tipo, orientação e número do interlocutor da chamada - ????** | |  |  |
| ICCID | Not programatically |  |
| BER | SignalStrenght -> [getGsmBitErrorRate](https://developer.android.com/reference/android/telephony/SignalStrength.html#getGsmBitErrorRate%28%29)( | 7 |
| Estado de Roaming | ConnectivityManager manager = (ConnectivityManager)getSystemService(Context.CONNECTIVITY\_SERVICE);  boolean isConnected = isNetworkConnected(manager, ConnectivityManager.TYPE\_MOBILE); | 1 |
| Codigo do Pais | TelephonyManager tm = (TelephonyManager)getSystemService(TELEPHONY\_SERVICE); String Country\_number = tm.getSimCountryIso() |  |
| Ec | Yes, see <http://developer.android.com/reference/android/telephony/CellSignalStrengthCdma.html> | 17 |
| RSCP | In ex: Wcdm ->getDbm() <https://developer.android.com/reference/android/telephony/CellSignalStrengthWcdma.html#getDbm()> | 18 |
| MCC | The [TelephonyManager](http://developer.android.com/reference/android/telephony/TelephonyManager.html) has a method to return the MCC+MNC as a String ([getNetworkOperator()](http://developer.android.com/reference/android/telephony/TelephonyManager.html#getNetworkOperator%28%29)) which will do you what you want. You can get access it via: Easy getted by doing this: <http://stackoverflow.com/questions/890366/how-do-you-get-the-phones-mcc-and-mnc-in-android> | 1 |
| MNC | Igual dos 3ºs primeiros digitos para a frente | 1 |
| BSIC | [TelephonyManager](http://developer.android.com/reference/android/telephony/TelephonyManager.html) -> getCellLocation | 1 |
| ARFCN | Not directly see: <https://groups.google.com/forum/?fromgroups=#!topic/android-platform/tVyNMnXtcEI> |  |
| PSC | CellIdentity<Tec> Android class -> [getPsc](https://developer.android.com/reference/android/telephony/CellIdentityWcdma.html#getPsc%28%29)() | Depende  Da  Tec. |
| RSRQ | Pelo SignalStrenght.toString() - Ver resposta stack:  <http://stackoverflow.com/questions/5545026/how-to-get-lte-signal-strength-in-android> |  |
| RSRP | Pelo SignalStrenght.toString() - Ver resposta stack:  <http://stackoverflow.com/questions/5545026/how-to-get-lte-signal-strength-in-android> |  |
| SINR | Não encontrei nada que ajude a obter, diretamente pela api android |  |
| SNR | O mesmo que SINR |  |
| RAC | Penso que não |  |
| RSSI | <https://developer.android.com/reference/android/telephony/SignalStrength.html#getEvdoDbm()> | 7 |
| PCI | https://developer.android.com/reference/android/telephony/CellIdentityLte.html#getPci() | 17 |
| Channel Number | No, it's impossible |  |
| Layer-3 Messaging | No for all models.  Only available for some specify model with Qualcomm Chipset with ROOT permission.  Ver+ mesmo assim.. |  |
| HTTP/FTP Measurements | Yes. But you need to do the FTP/HTTP yourself. |  |

Ps: Good Repository to take a deeper look after(Network Parameter): <https://github.com/yareally/SignalInfo>

## Android Device Info Parameters and where to find them:

|  |  |  |
| --- | --- | --- |
| Device Parameters | Notes / How to retrieve | Added in API |
| Memória | MemoryInfo mi = new MemoryInfo();  ActivityManager activityManager = (ActivityManager) getSystemService(ACTIVITY\_SERVICE);  activityManager.getMemoryInfo(mi);  Mi.totalMem | 16+ |
| Memória livre | MemoryInfo mi = new MemoryInfo();  ActivityManager activityManager = (ActivityManager) getSystemService(ACTIVITY\_SERVICE);  activityManager.getMemoryInfo(mi);  Mi.availMem | 1 |
| Battery Status | BatteryManager class: ver Summary Values | 1 |
| Battery Level | <https://developer.android.com/reference/android/os/BatteryManager.html#EXTRA_SCALE> | 5 |
| Battery Condition | BatteryManager class: ver Summary Values | 1 |
| Battery Voltagem | <https://developer.android.com/reference/android/os/BatteryManager.html#EXTRA_VOLTAGE> | 5 |
| Battery Health | BatteryManager class: ver Summary Values | 1 |
| Battery Temperature | <https://developer.android.com/reference/android/os/BatteryManager.html#EXTRA_TEMPERATURE> | 5 |
| CPU Temperature | Class Sensor -> int TYPE\_AMBIENT\_TEMPERATURE | 1 |
| Orientação/Direção | getResources().getConfiguration().orientation | 1 |
| Lat/Lon Coordinates | Get from LocationManager class | 14 |
| Satellites used - GPS | <http://stackoverflow.com/questions/6580603/getting-the-number-of-satellites-from-location-object> |  |

# Android

## Gravar e Reproduzir Audio de chamadas

You cannot record phone calls very well in Android, because the in-call audio is not available to SDK applications ACTION\_ANSWER is <http://developer.android.com/reference/android/content/Intent.html#ACTION_ANSWER>

a broadcast Intent

Uma opção é gravar o som do microfone, por exemplo com o alta-voz ligado.

VOICE\_CALL

Added in [API level 4](https://developer.android.com/guide/topics/manifest/uses-sdk-element.html#ApiLevels)

int VOICE\_CALL

Voice call uplink + downlink audio source

Capturing from VOICE\_CALL source requires the [CAPTURE\_AUDIO\_OUTPUT](https://developer.android.com/reference/android/Manifest.permission.html#CAPTURE_AUDIO_OUTPUT) permission. This permission is reserved for use by system components and is not available to third-party applications.

---------------------------------------------------------------

CAPTURE\_AUDIO\_OUTPUT

Added in [API level 19](https://developer.android.com/guide/topics/manifest/uses-sdk-element.html#ApiLevels)

[String](https://developer.android.com/reference/java/lang/String.html) CAPTURE\_AUDIO\_OUTPUT

Allows an application to capture audio output.

Not for use by third-party applications.

Constant Value: "android.permission.CAPTURE\_AUDIO\_OUTPUT"

<https://developer.android.com/reference/android/media/MediaRecorder.AudioSource.html#VOICE_CALL>

Mas no stack tem pessoas que conseguem o.O

<http://stackoverflow.com/questions/18672031/recording-calls-in-android-why-this-not-works>

Iphone com som do micro também:

<http://stackoverflow.com/questions/1809347/how-can-i-record-a-conversation-phone-call-on-ios>

## Permissões

<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.WRITE\_SETTINGS" />

<uses-permission android:name="android.permission.ACESS\_WIFI\_STATE" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />

<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />

<uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE"/>

<uses-permission android:name="android.permission.WAKE\_LOCK" />

**…BATTERY\_STATS, BROADCAST\_SMS, CALL\_PHONE, CHANGE\_WIFI\_MULTICAST\_STATE**

## O que varia com as API’s

### API 4.4 (Kitkat)

* **SMS Provider -** The [Telephony](https://developer.android.com/reference/android/provider/Telephony.html) content provider (the "SMS Provider") allows apps to read and write SMS and MMS messages on the device. It includes tables for SMS and MMS messages received, drafted, sent, pending, and more.
* Adaptative Playback - APIs, enabling seamless change in resolution during playback onto a [Surface](https://developer.android.com/reference/android/view/Surface.html)
* Read shared files only with permission [READ\_EXTERNAL\_STORAGE](https://developer.android.com/reference/android/Manifest.permission.html#READ_EXTERNAL_STORAGE)
* Android now includes a complete framework that allows users to print any document using a printer connected over Wi-Fi, Bluetooth, or other services.´
* Changes in WebViews, Alarm sync, Reusable bitmaps

More in: <https://developer.android.com/about/versions/android-4.4.html>

### API 5.0 (Lolipop)

* Lots of Desing and Material design changes including high performance graphics
* New metada in notifications
* Document-centric apps
* Perform concurrent operations with **Bluetooth Low Energy** (BLE)
* **NFC** APIs now allow apps to register an NFC application ID (AID) dynamically.
* Screen capturing and sharing - Android 5.0 lets you add screen capturing and screen sharing capabilities to your app.

More in: <https://developer.android.com/about/versions/android-5.0-changes.html>

### API 6.0 (Marshmallow)

* **AudioManager Changes** - Setting the volume directly or muting specific streams via the [AudioManager](https://developer.android.com/reference/android/media/AudioManager.html) class is no longer supported. The [setStreamSolo()](https://developer.android.com/reference/android/media/AudioManager.html#setStreamSolo%28int,%20boolean%29) method is deprecated, and you should call the [requestAudioFocus()](https://developer.android.com/reference/android/media/AudioManager.html#requestAudioFocus%28android.media.AudioManager.OnAudioFocusChangeListener,%20int,%20int%29) method instead. Similarly, the [setStreamMute()](https://developer.android.com/reference/android/media/AudioManager.html#setStreamMute%28int,%20boolean%29) method is deprecated; instead, call the [adjustStreamVolume()](https://developer.android.com/reference/android/media/AudioManager.html#adjustStreamVolume%28int,%20int,%20int%29) method and pass in the direction value [ADJUST\_MUTE](https://developer.android.com/reference/android/media/AudioManager.html#ADJUST_MUTE) or [ADJUST\_UNMUTE](https://developer.android.com/reference/android/media/AudioManager.html#ADJUST_UNMUTE).
* Optimize apps for power saving battery
* **Request permissions at Runtime**
* Previously, if an app forced the device to connect to a specific Wi-Fi network by using [enableNetwork()](https://developer.android.com/reference/android/net/wifi/WifiManager.html#enableNetwork%28int,%20boolean%29) with the disableAllOthers=true setting, the device disconnected from other networks such as cellular data.
* **Apache HTTP Client Removal**
* Android is moving away from OpenSSL to the [BoringSSL](https://boringssl.googlesource.com/boringssl/) library.

Outras -> <https://developer.android.com/about/versions/marshmallow/android-6.0-changes.html#behavior-text-selection>

### API 7.0 (Nougat)

* Battery and Memory - Android 7.0 includes system behavior changes aimed at improving the battery life (Doze)
* File system permission changes -Passing file:// URIs outside the package domain may leave the receiver with an unaccessible path. Therefore, attempts to pass a file:// URI trigger a FileUriExposedException. The recommended way to share the content of a private file is using the [FileProvider](https://developer.android.com/reference/android/support/v4/content/FileProvider.html).
* The [DownloadManager](https://developer.android.com/reference/android/app/DownloadManager.html) can no longer share privately stored files by filename.
* Screen Zoom

More in: <https://developer.android.com/about/versions/nougat/android-7.0-changes.html>

## Root…

## System App…

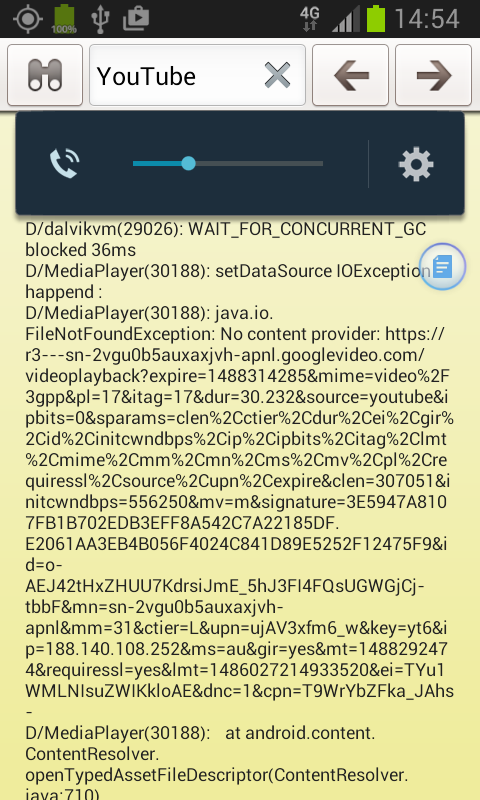
## Log File:

### Informação de rede:

* **Signal Strength** (tested being on 4G):
  + W/SignalStrength( 553): getDbm 1
  + D/SignalStrength( 553): getLteDbm=-81
  + W/SignalStrenght( 553): getDbm=-81
  + W/ ”” : getAsuLevel 1
  + W/ ”” : getLteDbm=-81
  + D/ ”” : Lte Asu level: 59
  + W/ ”” : getAsuLevel=59
* **Making a Call:**
  + **D/InCallTouchUi: updateState: current state = OFFHOOK**
  + D/InCallTouchUi: - updateState: showing in-call controls...
  + D/InCallTouchUi: - updateState: HIDING incoming call widget...
  + D/NotificationMgr: updateInCallNotification(): is a Voice call
* **Making Call (Other test** Speaker onn, Mic muted, etc…**):**
  + D/InCallTouchUi: updateState: current state = OFFHOOK
  + D/InCallTouchUi: - updateState: showing in-call controls...
  + D/CallNotifier: handleMessage : 1
  + D/CallNotifier: onPhoneStateChanged: state = OFFHOOK
  + D/PhoneApp: updateProximitySensorMode: lock already released.
  + D/CallNotifier: onPhoneStateChanged: OFF HOOK
  + **D/PhoneUtils: setAudioMode()...OFFHOOK**
  + D/CallManager: setAudioMode useInCallMode = false, Baseband = csfb
  + **D/PhoneUtils: setAudioMode() no change: MODE\_IN\_CALL**
  + D/Ringer: stopRing()...
  + D/Ringer: - stopRing: null mRingHandler!
  + W/Vibrator: Vibrator.cancel()
  + **D/CallNotifier: - posting UPDATE\_IN\_CALL\_NOTIFICATION request...**
  + **D/PhoneUtils: - hangup(Call): regular hangup()...**
  + **D/PhoneUtils: ==> hungup = true**
  + **D/PhoneApp: pokeUserActivity()...**
  + D/PhoneUtils: Not supported: BargeIn
  + D/PhoneUtils: ConnectionHandler: updating mute state for each connection
  + **D/PhoneUtils: setMuteInternal: using setMicrophoneMute(true)...**
  + D/AudioHardwareMSM8660: setMicMute\_nosync calling voice mute with the mMicMute 1
  + D/NotificationMgr: - Updating status bar icon: resId = 2130838622
  + D/NotificationMgr: updateInCallNotification(): is a Voice call
  + D/PhoneNumberUtils: ril.ecclist0 112/255,112
  + I/ProviderInstaller: Installed default security provider GmsCore\_OpenSSL
  + I/ActivityThread: Pub com.android.settings.reverb.params: com.android.settings.autohaptic.ReverbContentProvider
  + D/ApplicationPolicy: isStatusBarNotificationAllowed: packageName = com.android.phone
  + D/NotificationMgr: updateSpeakerNotification(true)...
  + D/InCallScreen: Handler : 101
  + D/InCallScreen: Handler: handling message { what=101 when=-359ms obj=android.os.AsyncResult@4280a228 } while not in foreground
  + D/InCallScreen: onPhoneStateChanged: Activity not in foreground! Bailing out...
  + D/CallNotifier: onDisconnect: cause = LOCAL, incoming = false
  + D/CallNotifier: autoretrySetting = 0
  + D/CallNotifier: stopRing()... (onDisconnect)
  + D/Ringer: stopRing()...
  + D/overlay: FROM\_STATE = OV\_BYPASS\_2\_LAYER TO\_STATE = OV\_CLOSED
  + 02-28 17:59:01.055 809-809/? D/Ringer: - stopRing: null mRingHandler!
  + W/Vibrator: Vibrator.cancel()
  + **D/CallNotifier: - need to play CALL\_ENDED tone!**
  + D/CallNotifier: mIsGsmRedialCall = false
  + D/CallNotifier: wake up screen
  + D/VoIPInterfaceManager: getState()...
  + D/VoIPInterfaceManager: Not exist call session
  + W/AlarmManager: FACTORY\_ON= 0
  + D/CallNotifier: resetAudioStateAfterDisconnect()...
  + E/Bluetooth HS/HF: audioOff(): mPendingScoForA2dp: false, mPendingScoForWbs: false, mConnectedSco: null, mA2dpState: 11, mA2dpSuspended: false, mVoiceRecognitionStarted: false
  + D/PhoneUtils: turnOnSpeaker(flag=false, store=true)...
  + D/PhoneUtils: restoreNoiseSuppression, restoring to: true
  + D/PhoneUtils: isNoiseSuppressionOn: dualmic\_enabled=false
  + D/PhoneUtils: turnOnNoiseSuppression: true
  + W/AlarmManager: FACTORY\_ON= 0
  + D/PhoneUtils: isWBMode, return true
  + D/PhoneUtils: isSupportedNoiseReductionInWBMode, return false
  + D/PhoneUtils: canEnableNoiseSuppression, return false
  + D/PhoneUtils: turnOnNoiseSuppression: Can't turn on in current audio state
* **Call now answered**
  + D/CallNotifier: handleMessage : 3
  + D/CallNotifier: DISCONNECT
  + D/CallNotifier: onDisconnect()... CallManager state: IDLE
  + D/NotificationMgr: updateStatusBar: state = 0x0
  + I/DBG\_DM: [v5\_1211\_3\_1][Line:1126][onCallStateChanged] >>>>>>>>>>> onCallStateChanged
  + D/STATUSBAR-StatusBarManagerService: manageDisableList what=0x0 pkg=com.android.phone
  + **D/CallNotifier: onDisconnect: cause = LOCAL, incoming = false**
  + D/CallNotifier: autoretrySetting = 0
  + **D/CallNotifier: stopRing()... (onDisconnect)**
  + **D/Ringer: stopRing()...**
  + D/Ringer: - stopRing: null mRingHandler!
* **Call Dropped** (other side)
  + D/CallNotifier: - onDisconnect(): logNumber set to: xxxxxxx
  + D/CallNotifier: InCallTonePlayer:
  + D/EcidContact: ECID AOSP VERSION: 2.3.6
  + D/InCallScreen: Handler : 102
  + **D/InCallScreen: onDisconnect: cause = ERROR\_UNSPECIFIED**
  + **D/InCallScreen: - onDisconnect: delayed bailout...**
  + D/EcidContact: No CityID found
  + I/CallLogAsync: remind\_me\_later\_set0
  + **D/InCallScreen: - onDisconnect: switching to 'Call ended' state...**
  + D/CallNotifier: InCallTonePlayer.run(toneId = 4)...
  + D/InCallScreen: - inCallScreenMode = CALL\_ENDED
  + D/InCallScreen: - updateScreen: updating the in-call UI...
  + D/InCallTouchUi: updateState: current state = IDLE
  + D/InCallTouchUi: - updateState: showing in-call controls...
  + D/EndCallButtonsView: updateState...
  + D/CallNotifier: InCallTonePlayer: stream :0
  + D/PhoneNumberUtils: ril.ecclist0 112/255,112
  + I/CallLogAsync: dormant\_set0
  + D/EndCallButtonsView: presentation: 1 / isCheckInvalidNumber: false
  + E/ActivityThread: Failed to find provider info for com.cequint.ecid
  + D/lights: LCD : 32 +
  + D/lights: LCD : 32 -
  + E/AudioPolicyManagerBase: stream type [10], return media stratege
  + E/AudioHardwareMSM8660: outputDev=1, inputDev=0
  + I/AudioPolicyManagerBase: checkAndSetVolume() dev 1 with 0
  + D/InCallTouchUi: - updateState: HIDING incoming call widget...
  + D/InCallTouchUi: hideIncomingCallWidget()...
  + D/CallCard: updateForegroundCall()...
  + D/CallCard: displayMainCallStatus(call DISCONNECTED)...
  + D/CallCard: updateIncomingPopupMsg
  + D/InCallScreen: isBluetoothAudioConnected: ==> isAudioOn = false
  + D/InCallScreen: isBluetoothAudioConnectedOrPending: ==> FALSE
  + D/InCallScreen: - onDisconnect: cleaning up after FG call disconnect...
  + E/Bluetooth HS/HF: handlePreciseCallStateChange() mPhone=IDLE, mFCall=DISCONNECTED, mBCall=IDLE, mRCall=IDLE, mCallsetup=0, mCall=1, mCallheld=0
  + D/CallNotifier: handleMessage : 1
  + D/CallNotifier: onPhoneStateChanged: state = IDLE
  + D/PhoneApp: updateProximitySensorMode: releasing...
  + D/SensorManager: unregisterListener:: Trklfufi 9 cwv5budiwrd5kfimfi5HwofiVbubgfiKfimrcf$\*8@,\*f,/a:(
  + D/SensorService: SensorDevice::activate (enable 1, disable 0) = 0
  + D/SensorService: activating sensor disable handle=4
  + D/SensorService: activating sensor disable, Done!
  + D/SensorService: SensorDevice::activating sensor handle=4 ns=200000000
  + D/PhoneApp: updateProximitySensorMode: lock already released.
  + D/PhoneUtils: Not supported: BargeIn
  + D/PhoneUtils: ConnectionHandler: updating mute state for each connection
  + D/PhoneUtils: setMuteInternal: using setMicrophoneMute(false)...
  + D/AudioHardwareMSM8660: setMicMute\_nosync calling voice mute with the mMicMute 0
  + D/lights: LCD : 53 +
  + D/InCallScreen: Handler : 101
  + D/PowerManagerService: releaseWakeLockLocked : PROXIMITY\_SCREEN\_OFF\_WAKE\_LOCK : mProximityWakeLockCount : 0 tag=PhoneApp uid=1001 pid=780
  + D/PowerManagerService: disableProximityLockLocked
  + D/VoIPInterfaceManager: getState()...
  + D/VoIPInterfaceManager: Not exist call session
  + D/VoIPInterfaceManager: getState()...
  + D/VoIPInterfaceManager: Not exist call session
  + D/InCallScreen: requestUpdateScreen()...
  + D/PhoneApp: updateWakeState: callscreen true, dialer false, speaker false...
  + D/PhoneApp: updateWakeState: keepScreenOn = true (isRinging false, isDialing false, showingDisc true)
  + E/AudioHardwareMSM8660: write(): dec\_id = 1 cur\_rx = 60
  + E/AudioHardwareMSM8660: cur\_rx for pcm playback = 60
  + E/AudioHardwareMSM8660: value of device and enable is 60 1 ALSA dev id:22
  + D/InCallScreen: Handler : 122
  + D/InCallScreen: - inCallScreenMode = CALL\_ENDED
  + D/InCallScreen: - updateScreen: updating the in-call UI...
  + D/InCallTouchUi: updateState: current state = IDLE
  + D/InCallTouchUi: - updateState: showing in-call controls...
  + D/EndCallButtonsView: updateState...
  + D/PhoneNumberUtils: ril.ecclist0 112/255,112
  + D/EndCallButtonsView: presentation: 1 / isCheckInvalidNumber: false
  + D/InCallTouchUi: - updateState: HIDING incoming call widget...
  + D/InCallTouchUi: hideIncomingCallWidget()...
  + D/CallCard: updateForegroundCall()...
  + D/CallCard: displayMainCallStatus(call DISCONNECTED)...
  + D/CallCard: updateIncomingPopupMsg
  + D/InCallScreen: isBluetoothAudioConnected: ==> isAudioOn = false
  + D/InCallScreen: isBluetoothAudioConnectedOrPending: ==> FALSE
  + D/lights: LCD : 110 +
  + D/lights: LCD : 110 -
  + I/power: \*\*\* release\_dvfs\_lock : lockType : 1
  + I/PowerManagerService: Light Animator Finished currentValue=110
  + D/PowerManagerService: releaseDVFSLockLocked : all DVFS\_MIN\_LIMIT are released
  + E/AudioPolicyManagerBase: stream type [10], return media stratege
  + D/CallNotifier: InCallTonePlayer:calling resetAudioStateAfterDisconnect + 4
  + D/CallNotifier: resetAudioStateAfterDisconnect()...
  + E/Bluetooth HS/HF: audioOff(): mPendingScoForA2dp: false, mPendingScoForWbs: false, mConnectedSco: null, mA2dpState: 11, mA2dpSuspended: false, mVoiceRecognitionStarted: false
  + D/PhoneUtils: turnOnSpeaker(flag=false, store=true)...
  + D/PhoneUtils: restoreNoiseSuppression, restoring to: true
  + D/PhoneUtils: isNoiseSuppressionOn: dualmic\_enabled=true
  + D/PhoneUtils: isWBMode, return false
  + E/AudioPolicyManagerBase: stream type [10], return media stratege
  + D/NotificationMgr: updateSpeakerNotification(false)...
  + D/PhoneApp: updateProximitySensorMode: lock already released.
  + D/PhoneUtils: turnOnNoiseSuppression: false
  + E/AudioHardwareMSM8660: outputDev=1, inputDev=0
  + I/AudioPolicyManagerBase: checkAndSetVolume() dev 1 with 0
  + I/AudioPolicyManagerBase: checkAndSetVolume() dev 1 with 0
  + I/AudioHardwareMSM8660: DualMic feature Disabled
  + E/AudioHardwareMSM8660: outputDev=1, inputDev=0
  + D/AudioHardwareMSM8660: new\_rx = 22
  + D/AudioHardwareMSM8660: new\_tx = 23
  + D/AudioHardwareMSM8660: updateDeviceInfo() called for default case
  + D/AudioHardwareMSM8660: updateDeviceInfo: E
  + D/AudioHardwareMSM8660: case VOICE\_CALLVOIP CALL 4
  + D/ACDB-LOADER: ACDB -> send\_voice\_cal, acdb\_rx = 60, acdb\_tx = 61
  + D/ACDB-LOADER: ACDB -> send\_voice\_rx\_topology
  + D/ACDB-LOADER: ACDB -> ACDB\_CMD\_GET\_DEVICE\_INFO
  + D/ACDB-LOADER: ACDB -> send\_voice\_tx\_topology
  + D/ACDB-LOADER: ACDB -> ACDB\_CMD\_GET\_DEVICE\_INFO
  + D/ACDB-LOADER: ACDB -> ACDB\_CMD\_GET\_AFE\_DATA
  + D/ACDB-LOADER: ACDB -> AUDIO\_SET\_SIDETONE\_CAL
  + D/ACDB-LOADER: ACDB samplerate = 256
  + D/ACDB-LOADER: ACDB samplerate = 256
  + D/ACDB-LOADER: ACDB -> ACDB\_CMD\_GET\_VOCPROC\_COMMON\_TABLE
  + D/ACDB-LOADER: ACDB samplerate = 256
  + D/ACDB-LOADER: ACDB samplerate = 256
  + D/ACDB-LOADER: ACDB -> ACDB\_CMD\_GET\_VOCPROC\_COMMON\_TABLE
  + D/ACDB-LOADER: ACDB samplerate = 256
  + D/ACDB-LOADER: ACDB samplerate = 256
  + D/ACDB-LOADER: ACDB -> ACDB\_CMD\_GET\_VOCPROC\_COMMON\_TABLE
  + D/ACDB-LOADER: ACDB samplerate = 256
  + D/VoIPInterfaceManager: getState()...
  + D/VoIPInterfaceManager: Not exist call session
  + D/VoIPInterfaceManager: getState()...
  + D/VoIPInterfaceManager: Not exist call session
* **+1Test Dropped Call:**
  + **D/EcidContact: No CityID found**
  + **E/ActivityThread: Failed to find provider info for com.cequint.ecid**
  + **I/CallLogAsync: remind\_me\_later\_set0**
  + **D/CallCard: updateForegroundCall()...**
  + **D/CallCard: displayMainCallStatus(call DISCONNECTED)...**

### Informação aplicacional:

* **Apps abertas** (ActivityManager):
  + Google Maps: *W/ActivityManager( 553): Activity stop timeout for ActivityRecord({742f20ba0 com.google.android.apps.maps/com.google.android.maps.MapsActivity})*
  + YouTube: “”
* **YouTube** (Video parameters):
  + **….**

****

### Informação do device:

* **Dialer** (Input Values):
  + **I/InputReader: Touch event's action is 0x1 (deviceType=0) [pCnt=1, s=]**

….

* + I/InputDispatcher: Delivering touch to current input target: action: 0x1
  + **I/InputReader: Touch event's action is 0x0 (deviceType=0) [pCnt=1, s=0.1200 ]** ->(Outra tecla)
* **Home:**  D/Launcher.HomeFragment: performOnHomePressed
  + **…**
  + D/Launcher.HomeFragment: onResume
  + D/MenuAppsGridFragment: onResume
* **Bluetooth** (Transferência de ficheiro)**:**
  + **D/BluetoothEventLoop: DISCOVERY STARTED**
  + D/BluetoothEventLoop: Device property changed
  + **D/BluetoothBondState: setBondState 10 reason: 0**
  + D/BluetoothBondState: setProfilePriorities\*\*\*\*
  + E/BluetoothBondState: Proxy is null:android.bluetooth.BluetoothA2dp@42c90138:android.bluetooth.BluetoothHeadset@426fde18:null
  + **D/BluetoothBondState: bond state 12 -> 10 (0)**
  + W/BluetoothBondState: setBondState() called to unbond device, but reason code is invalid. Overriding reason code with BOND\_RESULT\_REMOVED
  + D/BluetoothEventLoop: Device property changed
  + D/BluetoothNotiBroadcastReceiver: onReceive
  + D/BluetoothEventManager: DeviceFoundHandler created new CachedBluetoothDevice: cachedDevice
  + D/BluetoothDevicePreference: profile ::OPP connectionStatus::0
  + D/BluetoothDevicePreference: profile ::PAN connectionStatus::0
  + D/BluetoothDevicePreference: Is my device connected::false
* **Bluetooth** (Turn Off):
  + **D/STATUSBAR-BluetoothQuickSettingButton: Bluetooth deactivate()**
  + D/BluetoothService: bt disable
  + D/BluetoothAdapterStateMachine: BluetoothOn process message: 2
  + **E/BluetoothAdapterStateMachine: Bluetooth state 12 -> 13**
  + **E/BluetoothAdapterStateMachine: broadcastState : 13**
* **Battery Service** (Periodic)
  + D/BatteryService: update start
  + **D/BatteryService: level:13 scale:100 status:2 health:2** present:true voltage: 3783 temperature: 321 technology: Li-ion AC powered:false USB powered:true icon:17303267 invalid charger:0 online:1 charge type:0 current avg:0
  + D/ThermistorObserver: mBatteryBRForAMOLED :: onReceive(). **newTemperature =321**
  + D/STATUSBAR-BatteryController: onReceive() - ACTION\_BATTERY\_CHANGED
  + D/STATUSBAR-BatteryController: onReceive() - level:13
  + D/STATUSBAR-BatteryController: onReceive() - plugged:2
  + **D/STATUSBAR-BatteryController: onReceive() - BATTERY\_STATUS\_CHARGING:**
  + D/STATUSBAR-PhoneStatusBar: ACTION\_BATTERY\_CHANGED
  + D/STATUSBAR-PhoneStatusBar: NORMAL\_BATTERY

# VOLTE:

## Jio4GVoice

Android 4.0 ou superior

Permite chamadas c/ VoLTE em todos os dispositivos compatíveis tendo o Jio Sim Card

## Vantagens do VoLTE

As vantagens de utilizarmos a rede 4G para a utilização do VoLTE são grandes: velocidade maior e mais transmissão de dados. Para o usuário final os benefícios serão estes:

**Voz em HD**: A qualidade da chamada no VoLTE é maior e em alta definição. O método de compreensão utilizado atualmente entrega aproximadamente 8kbps, enquanto a compreensão de áudio realizada pelo VoLTE pode ser de até 13kbps. A nova faixa de dados operando em 700MHz também contribui para chamadas mais claras e limpas.

**Recursos extras**: As operadoras poderão oferecer um pacote de serviços extras, que incluem chamadas de vídeo, transferências de arquivos e anexo de imagens, além de traduções simultâneas.

**Conexão mais rápida**: O tempo de chamada do VoLTE é consideravelmente menor em comparação com as ligações atuais.

**Consumo menor de bateria**: Os serviços VoLTE consomem menos energia em comparação com serviços OTT como as chamadas do Skype.

**Integração com Wi-Fi**: É possível utilizar o VoLTE através de qualquer Wi-Fi, o que conseqüentemente irá economizar o plano de dados ativo no dispositivo.

## Notas do VoLTE

Precisamos salientar que esta é uma tecnologia nova e, ainda precisa de desenvolvimento para atingir um patamar ideal de qualidade.

Implementações de VolTE ainda não são inter-compatíveis. (Problema)

## Telemóveis compatíveis com VoLTE:

4G VoLTE devices list in India – Updated (02-10-2016)

<http://newsmarkets.in/4g-volte-devices-in-india-updated/>

|  |  |  |
| --- | --- | --- |
| 1 | Alcatel | Pop 3 |
| 2 | Alcatel | Pop 3 |
| 3 | Apple | iPhone 5 |
| 4 | Apple | iPhone 5S |
| 5 | Apple | iPhone 6 |
| 6 | Apple | iPhone 6S |
| 7 | Apple | iPhone 7 |
| 8 | Apple | iPhone 7S |
| 9 | Asus | Laser 5.0 (ZE500KL) |
| 10 | Asus | Zenfone Pegasus 3 |
| 11 | BlackBerry | Venice |
| 12 | Blackberry | Priv |
| 13 | Coolpad | Mega 2.5D |
| 14 | Coolpad | Note 5 |
| 15 | Gionee | S6 |
| 16 | Gionee | F103 |
| 17 | HTC | One M9+ |
| 18 | HTC | One A9 |
| 19 | HTC | 10 |
| 20 | HTC | Desire 10 Lifestyle |
| 21 | HTC | Desire 10 Pro |
| 22 | Huawei | Honor 5A |
| 23 | Infocus | M370 I |
| 24 | Infocus | M808 I |
| 25 | Infocus | M535 |
| 26 | Intex | Aqua 4G |
| 27 | Intex | Aqua Ace Mini |
| 28 | Intex | Aqua ace 2 |
| 29 | Intex | Aqua Craze |
| 30 | Intex | Cloud Sting |
| 31 | Intex | Cloud String HD |
| 32 | Intex | Cloud string v2 |
| 33 | Intex | Aqua Music |
| 34 | Intex | Aqua S7 |
| 35 | Intex | Aqua Race 2 |
| 36 | Karbonn | Aura |
| 37 | Lava | A71 |
| 38 | Lenovo | Vibe Shot |
| 39 | Lenovo | Vibe P1 |
| 40 | Lenovo | A 6000 Plus |
| 41 | Lenovo | Z2 |
| 42 | Lenovo | Z2 Plus |
| 43 | Lenovo | A6600 |
| 44 | Lenovo | A6600 Plus |
| 45 | Lenovo | A7700 |
| 46 | LG | Nexus 5X |
| 47 | LG | Spirit 4G |
| 48 | LG | G3 4G LTE 32GB |
| 49 | LG | G4 |
| 50 | LG | G4 Stylus 4G |
| 51 | LG | M1DS |
| 52 | LG | K7 |
| 53 | LG | K10 |
| 54 | LYF | LYF water 1 |
| 55 | LYF | LYF water 2 |
| 56 | LYF | LYF water 3 |
| 57 | LYF | LYF Earth 1 |
| 58 | Lyf | Flame 1 |
| 59 | Lyf | Flame 3 |
| 60 | Lyf | Flame 4 |
| 61 | Lyf | Water 1 |
| 62 | Lyf | Flame 2 |
| 63 | Lyf | Water 5 |
| 64 | Lyf | Wind 5 |
| 65 | Lyf | Flame 6 |
| 66 | Lyf | Water 8 |
| 67 | Lyf | Water 4 |
| 68 | Lyf | Wind 6 |
| 69 | Lyf | Flame 7 |
| 70 | Lyf | Wind 7 |
| 71 | Lyf | Flame 8 |
| 72 | Lyf | Wind 3 |
| 73 | Micromax | Canvas Amaze |
| 74 | Micromax | Juice 4G (Q461) |
| 75 | Micromax | Canvas Mega 4G |
| 76 | Micromax | Canvas Knight |
| 77 | Micromax | Canvas Silver 5 |
| 78 | Microsoft | All devices with and above 8.1 Supports VoLTE |
| 79 | Motorola | Moto E (2nd Gen) |
| 80 | Motorola | Moto G (3rd Gen) |
| 81 | Motorola | Moto G Turbo |
| 82 | Motorola | Moto X Play |
| 83 | Motorola | Moto G4 Plus |
| 84 | Motorola | E3 Power |
| 85 | Motorola | G4 Play |
| 86 | Oppo | Neo 7 |
| 87 | Oppo | S9 |
| 88 | Oppo | F1 |
| 89 | Panasonic | Eluga Arc |
| 90 | Panasonic | Eluga A2 |
| 91 | Panasonic | Eluga Note |
| 92 | Panasonic | P 77 |
| 93 | Samsung | Galaxy Core Prime 4G |
| 94 | Samsung | Galaxy A5 |
| 95 | Samsung | Galaxy A7 |
| 96 | Samsung | Galaxy A8 |
| 97 | Samsung | Galaxy S6 |
| 98 | Samsung | Galaxy S6 Egde |
| 99 | Samsung | Galaxy S6 Edge Plus |
| 100 | Samsung | Galaxy Note 4 |
| 101 | Samsung | Galaxy Note 5 |
| 102 | Samsung | Galaxy Note Edge |
| 103 | Samsung | Galaxy J5 |
| 104 | Samsung | Galaxy J7 |
| 105 | Samsung | Galaxy On5 |
| 106 | Samsung | Galaxy On7 |
| 107 | Samsung | Note 5 Duos |
| 108 | Samsung | J2 |
| 109 | Samsung | S7 |
| 110 | Samsung | S7 Edge |
| 111 | Samsung | C5 |
| 112 | Samsung | C7 |
| 113 | Samsung | Galaxy On5 Pro |
| 114 | Samsung | Galaxy On7 Pro |
| 115 | Samsung | Galaxy On 8 |
| 116 | Sony | Xperia Z5 Dual |
| 117 | Sony | Xperia Z5 Premium  Dual |
| 118 | TCL | 950 |
| 119 | Videocon | Krypton3 V50JG |
| 120 | Vivo | X33 |
| 121 | Xiaomi | Redmi 2 Prime |
| 122 | Xiaomi | Note 3 |
| 123 | Xiaomi | Redmi 2 (2014818) |
| 124 | Xiaomi | Mi 5 |
| 125 | Xioami | Mi MAX |
| 126 | Yu | Yunique |
| 127 | Yu | Yuphoria |
| 128 | Yu | Yureka Plus |
| 129 | Yu | Note (YU6000) |
| 130 | Yu | Yureka 3 (YU5200) |
| 131 | Yu | Yutopia (YU5050) |
| 132 | ZTE | Blade S6 |
| 133 | ZTE | Blade S6 Plus |

(pelos comentários dos users no site, alguns telemóveis não suportam, lista para ser atualizada novamente brevemente)

## Best smartphones that support VoLTE

[Samsung Galaxy S7 and S7 Edge](http://www.digit.in/mobile-phones/samsung-galaxy-s7-edge-price-6677.html)

[iPhone 6s](http://www.digit.in/mobile-phones/apple-iphone-6s-price-5927.html)

[LG G5](http://www.digit.in/mobile-phones/lg-g5-price-6688.html)

[OnePlus 3](http://www.digit.in/mobile-phones/oneplus-3-price-37443.html)

[Huawei P9](http://www.digit.in/mobile-phones/huawei-p9-price-6857.html)

[Nexus 6P](http://www.digit.in/mobile-phones/huawei-nexus-6p-price-5990.html)

[Xiaomi Mi 5 32 GB](http://www.digit.in/mobile-phones/xiaomi-mi-5-32-gb-price-6708.html)

[Samsung Galaxy Note 5](http://www.digit.in/mobile-phones/samsung-galaxy-note-5-price-5921.html)

<http://www.digit.in/top-products/best-smartphones-that-support-volte-179.html>

# VoWifi

The new VoWiFi capability differs from existing VoIP technology like Skype or Facetime in that these ‘Over The Top’ services cannot maintain the call connection when moving from WiFi to cellular coverage, since they require the user to download a separate application, use a separate dialer interface, and make the call using a different phone number.

Very important, however, is the fact that this new VoWiFi capability is only available to consumer phones once the service provider has invested in a VoWiFi infrastructure. Without this investment, VoWiFi services are not available to the consumer. Therefore, in this two-part blog, we will touch on some important VoWiFi services themes that have emerged as catalysts to the SP VoWiFi capital spending investment, as well as what we see as the three different deployment plans being adopted by service providers to support VoWiFi service.

## VoWifi – Vantagens para o utilizador final

* Única e uniforme voice dialer no smartphone
* Permite aos serviços de voz comunicar sobre Wifi em casa, no escritório ou em qualquer hotspot.
* Wi-fi tem melhor cobertura indoor.
* **Transferências de chamada entre Wi-fi e LTE usando VoLTE nos dispositivos compatíveis.**
* Serviços de Roaming são minimizados, se o operador usar o mesmo modelo de negócio que a T-Mobile. Chamadas de qualquer parte do mundo são cobradas como uma chamada móvel em qualquer parte que haja Wi-fi.

## VoWifi – Vantagens para o operador

* Cobertura indoor melhor comparativamente às estações celulares. Isto resolve o “radio tight” na construção da estrutura moderna
* Reduzido ruído
* Apps como Skype e Viber menos relevantes e importante para os clientes visto terem a cobertura indoor com VoWifi, maior uso da tecnologia.
* Não têm que investir em Wi-fi footprint podem basear-se nas redes Wi-Fi já existentes.
* Partilha infraestrutura semelhante com o IMS-based VolTE

## VoWifi – Quais são as possíveis desvantagens?

### Chamadas Wifi e a Qualidade de Serviço

Abdicar do controlo por parte do operador sobre QoS para voz, porque o tráfego vai sobre redes que eles não administram. Contudo com o aumento da capacidade da rede WiFi com o 801.11ac dando velocidades de Gigabit e aumentando a capacidade de prioritizar o tráfego multimédia com WMM, a Qualidade de Serviço na maioria dos casos não seria um desafio/problema. As chamadas Wi-Fi são também uma alternativa às redes chamadas celulares sobre LTE que têm em geral baixa qualidade.

### Chamadas Wifi e as chamadas de emergência

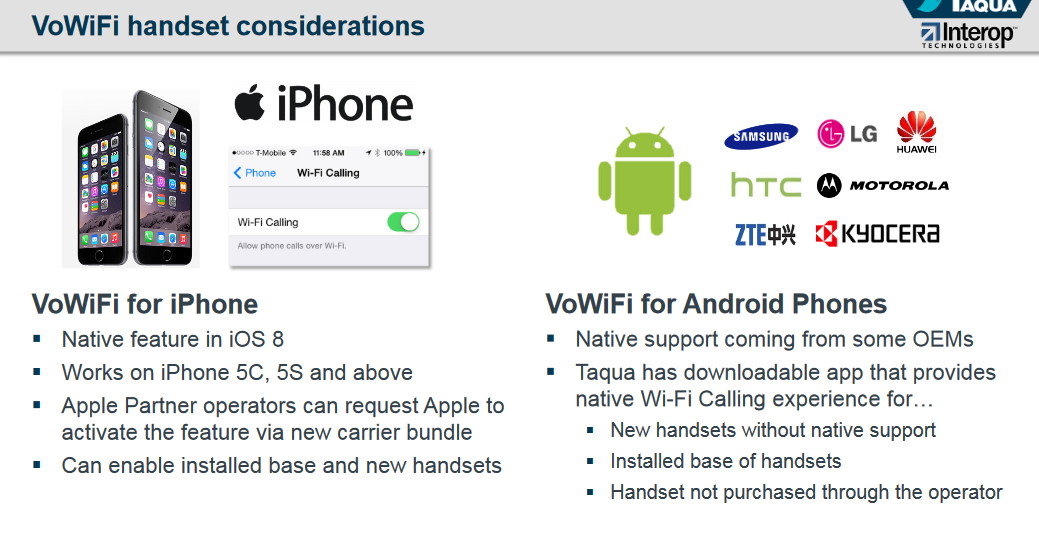
Como em qualquer serviço de voz, os operadores de emergência têm dificuldade em detetar a localização do utilizador. Depois do registo da chamada Wi-Fi o utilizador tem de registar uma localização por defeito, normalmente o endereço de sua casa. As capacidades do Aptilos para localizar os APs com a localização dos mesmos vai melhorar a informação sobre o local onde se encontra o utilizador na rede Wi-Fi. Grandes empresas como a Apple têm também dados da informação da localização de pontos de acesso Wi-Fi. Tudo depende de como as empresas/agências verão isto, mas o problema da localização não está relacionado com a solução Wi-Fi, o mesmo problema encontra-se presente nos serviços de voz OTT.

**Resumo:** Apps como o Facetime ou o Skype não conseguem manter a chamada quando o utilizador passa de Wifi para uma rede celular, é aqui que o VoWIFI ganha em comparação, visto que estas aplicações fazem o download de um dialer, application e um número de telefone diferente.  
 É importante referir também que só é possível passar a usar VoWIFI quando houver o investimento na infraestrutura para suportar esta tecnologia.  
 Vantagens e Desvantagens…

https://www.aptilo.com/wi-fi-calling/what-is-wi-fi-calling-vowifi

<http://www.delloro.com/chris-depuy/voiceoverwifi-sending-a-strong-signal-to-consumers-and-service-providers-part-1>

## Terminais móveis que suportam:

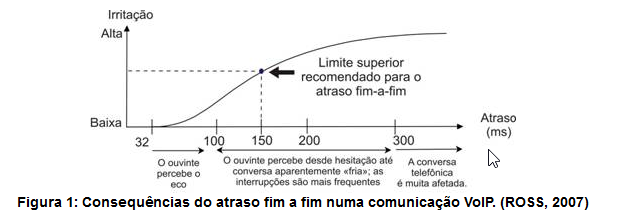


# Audio

## Fatores consideráveis para a qualidade de serviço VoIP

* Latência(tempo de transmissão do 1º bit até ser recebido)
* Tempo necessário de geração de um número suficiente dos pacotes de voz para preencher o payload (carga útil) do pacote IP.
* Perda de pacotes
* Largura de banda disponível
* Outro fator é a supressão de silêncio ou Voice Activity Detection (VAD),
* Jitter

Um dos problemas críticos de uma comunicação VoIP é o atraso fim a fim ou latência, que consiste na diferença de tempo em que é transmitido o primeiro *bit* do pacote de voz, do tempo que esse mesmo *bit* é recebido, a diferença não pode ultrapassar 150 milissegundos (ms), de acordo com a recomendação G.114 da ITU-T. Caso existam atrasos acima desse patamar, haverá sobreposição das falas nas chamadas, ou seja, a perda de sincronização.



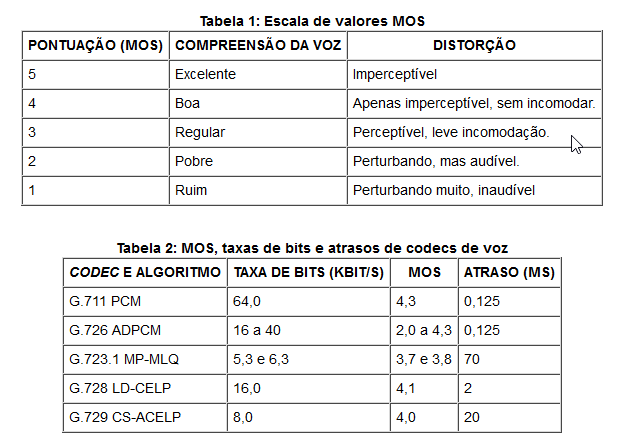
A largura de banda disponível para o tráfego de voz numa rede IP é de extrema importância, como trafegam também dados, informações de tempo real necessitam de prioridade. Como exemplo, uma ligação VoIP consome em média 8kbit/s de codificação, utilizando o *codec* G.729, há mais 16kbit/s dos cabeçalhos IP, RTP e UDP, resultando em 24kbit/s em média por ligação. Se não houver prioridade dos pacotes de voz pela rede, outros dados consumirão a largura de banda.

Outro fator é a supressão de silêncio ou *Voice Activity Detection* (VAD), como a comunicação entre origem e destino é *halfduplex*, ocorrem pausas quando um termina e o outro começa. Essas pausas podem ser suprimidas e recriadas na outra extremidade, para manter o tom natural da conversa, evitando a transmissão de pacotes de silêncio pela rede. Numa ligação telefônica, constatou-se que 22% é a comunicação propriamente dita, enquanto que 56% são pausas e outros 22% são padrões repetitivos (JÚNIOR et al, 2010). A supressão de silêncio é feita pelos algoritmos de compressão de voz, utilizados pelos *codecs*.

Um último fator a ser considerado, porém não sendo o menos importante, é o *jitter*. *Jitter* é o termo que define as variações de tempo da chegada do pacote de voz ao destino, ou seja, a variação do atraso fim a fim. Numa comunicação telefônica, os fluxos de pacotes de voz devem chegar ao destino numa harmonia constante, e de preferência no mesmo ritmo com que foram gerados pela origem. Se o *jitter* for muito grande, mesmo o atraso se mantendo dentro dos limites aceitáveis, a qualidade da comunicação vai decrescer até se tornar impossível. O *jitter* não pode ultrapassar 20 ms.

## Métodos de avaliação de desempenho

Existem alguns métodos de avaliação de desempenho da qualidade de voz trafegada por uma rede IP: MOS, PSQM, PAMS e PESQ.



## Teste: Desempenho VoIP: Qualidade de Voz em Redes IP

Estudos realizados anteriormente definem dois métodos básicos para medir o tráfego com QoS em uma rede: **Intrusivo** e **Não-Intrusivo**.

O **Intrusivo** se refere à injeção controlada de pacotes na rede e subseqüentemente da coleta destes pacotes. Já o **Não-Intrusivo** está relacionado com a medição do comportamento da rede através da observação da taxa de chegada de pacotes em um sistema final.

Na técnica pesquisada por nós os testes foram realizados considerando o tráfego do tipo intrusivo, através de uma ferramenta composta de dois agentes: gerador e coletor.

**Caracterizando o Tráfego de Teste:**

Para este teste foi adotado como codec o G.711 sem supressão de silêncio. Essa escolha foi feita por se tratar do pior caso em termos de taxa de bit gerada e, além disso, ele é muito usado para reduzir retardos de codificação/decodificação. Neste exemplo cada fluxo de voz é caracterizado por um tráfego do tipo CBR (Constant Bit Rate), em uma taxa de 80 kbits e cujo tamanho dos pacotes é igual a 200 bytes. o tráfego de voz sintetizado para esse teste corresponde a NCS x 80 kbit/s.

Note que aqui foi considerado um número de chamadas simultâneas constante durante todo o teste por motivo de simplificação do gerador de testes e da análise. Em um ambiente real, o número de chamadas e a duração das chamadas variam com o tempo.

**Medindo a Qualidade**

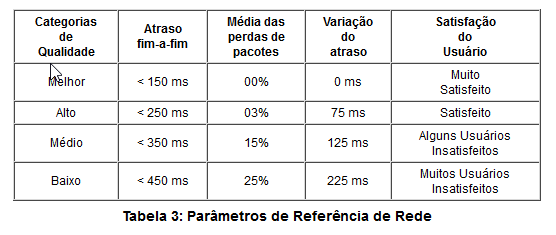
Para aferir a qualidade de voz obtida, é necessário definir os níveis de qualidade de voz e os correspondentes valores limites dos parâmetros de desempenho de rede. Neste exemplo, as seguintes métricas são consideradas para medir a qualidade de voz:

**Vazão**: taxa de transmissão em bits por segundo (bit/s);

**Atraso por pacote**: tempo que um pacote leva da origem até o destino;

**Variação no atraso**: é a variação do atraso por pacote entre dois pacotes subjacentes. Por meio dessa métrica, consegue-se analisar quão constante é o roteamento de pacotes processado por salto (*hop)*;

**Taxa de perda de pacotes**: quantidade de pacotes que foram descartados por fluxo numa transmissão de dados (Voz).



<http://www.teleco.com.br/tutoriais/tutorialmondesvoip/pagina_5.asp>

Ver : <http://www.madeira.eng.br/wiki/index.php?page=Qualidade+de+Voz+e+MOS>

## 48K / Audio HD

HD Voice is essentially wideband audio technology, something that long has been used for conference calling and VoIP apps. Instead of limiting a call frequency to between 300 Hz and 3.4 kHz, a wideband audio call transmits at a range of 50 Hz to 7 kHz, or higher. That’s much more in line with the human voice, which transmits audio between 75 Hz and 14 kHz.

Most of the latest smartphones now support HD Voice, including the iPhone 5, Samsung S III (and IV), HTC One, Nokia Lumia 920 and Sony Xperia Z. All of the phones have speakers and microphones that support wideband audio.

Now, all we need are carriers to actually enable HD Voice. They will do so by using a higher-quality speech codec called AMR-WB (Adaptive Multi-Rate Wideband). That codec doubles the frequency of the call, which means you’ll hear less muffled voices and be able to better distinguish between “s” and “f” sounds. VoLTE already supports AMR-WB, so it’s not very difficult for carriers like AT&T to enable HD Voice once calls have moved over to its LTE network.

**Resumo:** Em vez de limitar a frequência de chamadas entre os 300Hz-3.4kHz, usa-se 50Hz-70kHz ou + que é mais perto da frequência da voz humana(75Hz-14kHz); Muitos dos smartphones hoje em dia já suportam este áudio só precisamos de ativar o HD Voice c/ um codec (AMR-WB)(Adaptive Multi-Rate Wideband), o VoLTE já usa esse codec, por isso, não será muito difícil para grandes empresas fazer o mesmo quando as chamadas forem movidas sobre a rede LTE.

## IVR**:**

Short for ***i***nteractive ***v***oice ***r***esponse, a [telephony](http://www.webopedia.com/TERM/T/telephony.html) technology in which someone uses a touch-tone telephone to interact with a [database](http://www.webopedia.com/TERM/D/database.html) to acquire information from or enter data into the database. IVR technology does not require human interaction over the telephone as the user's interaction with the database is predetermined by what the IVR system will allow the user access to. For example, banks and credit card companies use IVR systems so that their customers can receive up-to-date account information instantly and easily without having to speak directly to a person. IVR technology is also used to gather information, as in the case of telephone surveys in which the user is prompted to answer questions by pushing the numbers on a touch-tone telephone.

Interactive Voice Response (IVR) é uma tecnologia madura, estável e útil que torna as interações com os clientes mais eficientes e simples. Nos dias de hoje, a maioria dos indivíduos que telefona para o número de uma empresas de vendas ou suporte espera algum nível de interação com um sistema automatizado. No entanto, os IVR não são criados de forma igual. Alguns são demasiado complexos. Algumas empresas ainda se recusam a utilizar IVR, e outras têm um design tão pobre destes sistemas que servem apenas como um exemplo daquilo que não se deve fazer. A forma como um sistema destes é desenhado pode ter um impacto muito significativo na experiência dos clientes. Será que o seu cliente abandona a chamada com um sentimento de realização ou de frustração? Algumas dicas simples podem-no ajudar a garantir que a sua marca é parte da solução e não do problema.

Pontos a focar:

* Sistema de menu: não diga “Prima um para o departamento de vendas.” Em vez disso diga, “Para o departamento de vendas, prima um.” Isto ajuda os clientes a escutar com mais atenção a opção de que necessitam e a pressionar o botão certo.
* Quanto menos opções melhor, tentar reduzir ao máximo aárvore de decisões
* Objetivo é Facilitar, não dificultar – tornar fácil a opção de pedir ajuda ao operador/agente

Criar um IVR para os clientes pode resultar na melhoria das interações e da experiência dos clientes, tornando a gestão das chamadas mais eficiente. No entanto, também pode ser uma experiência frustrante, que provoca descontentamento e pode até ser prejudicial para a sua reputação.

<http://www.gocontact.pt/blog/como-design-inteligente-ivr-melhorar-experiencia-clientes/>

Ver: <http://materiale-didattico.info/index.php?newsid=174632>

# Comparar qualidade de video

## Why you cannot simply compare bitrate/resolution/etc:

Just comparing video resolution won't tell anything about the quality. In fact, it may be completely misleading. A 1080p movie rip at 700MB size might look worse than a 720p rip at 700MB, because for the former, the bitrate is too low, which introduces all kinds of compression artifacts.

The same goes for comparing bitrate at similar frame sizes, as different encoders can actually deliver better quality at less bitrate, or vice-versa. For example, a 720p 700MB rip produced with XviD will look worse than a 700MB rip produced with x264, because the latter is much more efficient.

You would also have to define how a final "integral score" (the MOS) is composed of the individual quality factors. This heavily depends on several things, including but not limited to:

the type of videos you are comparing (cartoons, movies, news, etc.)

their length

their viewing audience

their original frame size

their original "quality" before they were encoded

We're not even talking about how humans would perceive the videos. Let's assume you have a friend who is watching movies because he or she enjoys crisp details and high motion resolution. They would be much more critical when seeing a low quality rip than a friend who is just watching movies for their content. They probably would not care about the quality so much, as long as the movie is funny or entertaining.

### There are different types of video quality metrics!

Let me give you a list of what I think of is most commonly used for basic evaluation of video quality today. There exist several video quality metrics, which can be classified according to which kind of information is used to determine the quality. In principle and very simply speaking, you distinguish between the following:

**No-reference metrics** – They just have one video as input and output a quality score. In your case you are looking for a no-reference metric, because you often do not even have the original video. Such a metric will take one video and output one quality score. If you do a web search (e.g. on [Google Scholar](http://scholar.google.com/)) for "no-reference video quality", you'll find a lot of resources. They are easier to compute.

**Full-reference metrics** – They have two inputs, one being the original input video and the other being the encoded video. For example, you could take a DVD movie, then create two rips from it, and use a full-reference metric to estimate the quality loss between the original DVD movie (i.e. the MPEG-2 video on the disc) and your rips. This will take a long time to compute, but it's more accurate.

### Other Metrics/Measures

[PEVQ](http://en.wikipedia.org/wiki/PEVQ) is a standardized full-reference metric under ITU-T J.246. It aims at multimedia signals, but not HD video.

[VQuad-HD](http://en.wikipedia.org/wiki/VQuad-HD) is another full-reference metric standardized as ITU-T J.341. Since it's newer, its better suited for HD video.

Both of them are commercial solutions and you'll not find a software to download for them.

There are also some ITU standards on no-reference metrics, such as [ITU-T P.1201](http://www.itu.int/rec/T-REC-P.1201/en) and [ITU-T P.1202](http://www.itu.int/rec/T-REC-P.1202/en), which work with parameters from the bitstream. For those, there's also no standalone software you can use. But you could of course implement it yourself if you have a few days/weeks of time.

### Summary

If you just seek to compare simple objectively measurable criteria like:

Frame size

Bit rate

Frames per second

Video resolution

… a simple call to ffmpeg -i should give you all the details you need at the beginning. Also have a look at the -vstats option. You could then summarize this in a spreadsheet. Note that when you encode videos, x264 for example will log stuff like PSNR straight to a file if you need to, so you can use these values later.

As for how to weigh these criteria, you should probably emphasize the bit rate – but only if you know that the codec is the same. You could generally say that when both videos use x264, the one with higher bitrate is better. Even more generally, you should choose a lower resolution when you have two videos with the same bitrate, since the degradation due to upscaling is not as bad as the degradation due to low bitrate.

Comparing different codecs according to their bit rate is not possible unless you know more about the content and the individual encoding settings. Frame rate is a very subjective thing too and should be counted into your measurements if it is well below 25 Hz.

To summarize, heavily emphasize the bitrate if it's the only thing you have. Don't forget to use your eyes, too :)

<http://superuser.com/questions/338725/compare-two-video-files-to-find-out-which-has-best-quality>

## Video Encoding Recommendations

Table 2, below, lists examples of video encoding profiles and parameters that the Android media framework supports for playback in the H.264 Baseline Profile codec. While table 3 lists examples that the framework supports for playback in the VP8 media codec.

In addition to these encoding parameter recommendations, a device's available video recording profiles can be used as a proxy for media playback capabilities. These profiles can be inspected using the [CamcorderProfile](https://developer.android.com/reference/android/media/CamcorderProfile.html) class, which is available since API level 8.

**Table 2.** Examples of supported video encoding parameters for the H.264 Baseline Profile codec.

|  | **SD (Low quality)** | **SD (High quality)** | **HD 720p (N/A on all devices)** |
| --- | --- | --- | --- |
| **Video resolution** | 176 x 144 px | 480 x 360 px | 1280 x 720 px |
| **Video frame rate** | 12 fps | 30 fps | 30 fps |
| **Video bitrate** | 56 Kbps | 500 Kbps | 2 Mbps |
| **Audio codec** | AAC-LC | AAC-LC | AAC-LC |
| **Audio channels** | 1 (mono) | 2 (stereo) | 2 (stereo) |
| **Audio bitrate** | 24 Kbps | 128 Kbps | 192 Kbps |

**Table 3.** Examples of supported video encoding parameters for the VP8 codec.

|  | **SD (Low quality)** | **SD (High quality)** | **HD 720p (N/A on all devices)** | **HD 1080p (N/A on all devices)** |
| --- | --- | --- | --- | --- |
| **Video resolution** | 320 x 180 px | 640 x 360 px | 1280 x 720 px | 1920 x 1080 px |
| **Video frame rate** | 30 fps | 30 fps | 30 fps | 30 fps |
| **Video bitrate** | 800 Kbps | 2 Mbps | 4 Mbps | 10 Mbps |

<https://developer.android.com/guide/appendix/media-formats.html>

## CamcorderProfile

Retrieves the predefined camcorder profile settings for camcorder applications. These settings are read-only.

The compressed output from a recording session with a given CamcorderProfile contains two tracks: one for audio and one for video.

Each profile specifies the following set of parameters:

The file output format

Video codec format

Video bit rate in bits per second

Video frame rate in frames per second

Video frame width and height,

Audio codec format

Audio bit rate in bits per second,

Audio sample rate

Number of audio channels for recording.

<https://developer.android.com/reference/android/media/CamcorderProfile.html>

# Implementation:

## Tasks:

* Anteder e desligar chamada - simula o pressionar de teclas de auricular
* Gravação de áudio – Tive que simular via software que tinha um auricular ligado para não ser captado o áudio do microfone
* Deteção de quando uma chamada efectuada é atendida – Quando a chamada é iniciada, é iniciada uma thread para detectar quando sistema android muda o modo do áudio para MODE\_IN\_COMMUNICATION.

Notas: Atender uma chamada -> W/System.err: java.lang.SecurityException: Neither user 10119 nor current process has android.permission.MODIFY\_PHONE\_STATE.

W/System.err: at android.os.Parcel.readException(Parcel.java:1620)

W/System.err: at android.os.Parcel.readException(Parcel.java:1573)

“” at com.android.internal.telephony.ITelephony$Stub$Proxy.answerRingingCall(ITelephony.java:1662)

W/System.err: at pt.ptinovacao.arqospocket.service.service.utils.CallUtils.answerCallAIDL(CallUtils.java:48)

W/System.err: at pt.ptinovacao.arqospocket.service.service.utils.CallUtils.answerCall(CallUtils.java:31)

W/System.err: at pt.ptinovacao.arqospocket.service.tasks.AnswerVoiceCall$1.run(AnswerVoiceCall.java:201)

Pequenos textos

1. Introduction: What is QoS

Quality of Service (QoS) in cellular networks is defined as the capability of the cellular service providers to provide a satisfactory service which includes voice quality, signal strength, low call

blocking and dropping probability, highdata rates for multimedia and data applications etc. For networkbased services QoS depends on the following factors [

**Throughput**

The rate at which the packets go through the network. Maximum rate is always preferred.

**Delay**

This is the time which a packet takes to travel from one end to the other. Minimum delay is always preferred.

**Packet Loss Rate**

The rate at which a packet is lost. Thisshould also be as minimum as possible.

**Packet Error Rate**

This is the errors which are present ina packet due to corrupted bits. This should be as minimum as possible

**Reliability**

The availability of a connection. (Links going up/down).

2. What is POLQA

 - POLQA is the next-generation voice quality testing technology for fixed, mobile and IP based networks.

  POLQA has been selected to form the new ITU-T voice quality testing standard, P.863, and will be used with HD Voice, 3G and 4G/LTE as well as legacy PSTN.

**A ver:**

Estudo da Anacom 2011: <http://www.anacom.pt/render.jsp?categoryId=240084>

Anacom Studie(+especifico em valores): <http://www.anacom.pt/streaming/inglesaqos.pdf?categoryId=7713&contentId=18511&field=ATTACHED_FILE>

Ver Disseratação: <https://www.researchgate.net/publication/289308153_Dissertacao_de_Mestrado_Uma_Nova_Abordagem_de_Selecao_de_Celula_e_Handover_para_Redes_LTE_Heterogeneas_Criterios_Adicionais_Baseados_em_Estimativa_de_Capacidade_e_Velocidade_de_Usuario>

<http://www.fit.br/revista/doc/6_92.pdf>

Ver pdf: Quality of Service in LTE

<https://www.sandvine.com/downloads/general/whitepapers/quality-of-service-in-lte-long-form.pdf>

VoWIFI - <http://2pvoyj4eif0oo4rj01598nqo.wpengine.netdna-cdn.com/wp-content/uploads/2016/01/White_Paper_Top_10_VoWiFi_Questions_01.26.16.pdf>

inicio

Ver: <http://sysmech.co.uk/using-active-and-passive-probe-data-to-enhance-network-quality-of-service-qos-and-quality-of-experience-qoe/>