ETSI TR 103 114 V1.1.1 (2013-10)

Speech and multimedia Transmission Quality (STQ); QoS Parameters and measurement methodology for Smartphones

Reference
DTR/STQ-00194m

Keywords
3G, GSM, multi service testing, QoS, smartphone

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Contents

Intellectual Property Rights 4
Foreword 4
Introduction 4
1 Scope 5
2 References 5
2.1 Normative references 5
2.2 Informative references 5
3 Definitions and abbreviations 5
3.1 Definitions 5
3.2 Abbreviations 5
4 Smartphones as a QoS Test Environment 6
4.1 Boundary conditions of smartphones 6
4.2 Popular Operating Systems used on Smartphones 7
4.2.1 Android® 7
4.2.2 iOS™ 7
4.2.3 Windows Phone 8℠ 7
4.2.4 Blackberry™ OS 8
4.2.5 Symbian® 8
5 Limitations of Smartphones 8
5.1 Device Limitations 8
5.2 Operating System Limitations 8
5.3 Approximations due to working on application layer 9
5.4 Attended, unattended and automated Operations 9
5.5 Further Performance Considerations 9
6 Basic Settings for QoS Assessments 10
6.1 General 10
6.2 Trigger points 10
6.3 Timeouts 10
7 How to determine QoS parameters in smartphone QoS testing 11
7.1 Web Browsing HTTP 11
7.1.1 HTTP Service Non-Accessibility [%] 11
7.1.1.1 Trigger Points 11
7.1.1.2 OS specific mappings 11
7.1.1.3 Android OS® 11
8 Test Methodology 11
9 Multi-Service Testing 12
9.1 Basic scenarios for stand-alone devices 12
9.1.1 Basic principle 12
9.1.2 Characteristics and limitations 14
9.1.3 Example: Data as a background service ("CS in PS") 15
9.1.4 Example: Voice as a background service ("PS in CS") 15
9.1.5 Multi-service QoS parameter constellations 16
9.2 Extended scenarios for device pairs 16
9.3 Download and Upload Throughput Tests in parallel 17
History 18

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://ipr.etsi.org).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Report (TR) has been produced by ETSI Technical Committee Speech and multimedia Transmission Quality (STQ).

Introduction

Smartphones can be used as a platform for the execution of Quality of Service measurements. However, compared to host based systems, some restrictions apply.

These restrictions of smartphones are not only dealing with limited processing power and memory, but also with the availability of information to execute QoS measurements. The access to the operating system, the provided functionality to gain network and service related information might make a difference when trying to implement QoS testing within smartphone environments.

In detail, differences and necessary changes compared to host based systems are pointed out. However, the well-defined QoS environment already existing for host based systems should be reused wherever possible. Smartphone specific procedures and settings should complement or override existing definitions if necessary.

A specialty of smartphones is the parallel usage of different services. The so called multi-service testing describes a complex scenario where at least two services are used in parallel. This situation has to be considered when discussing QoS measures and results.

1 Scope

The present document is intended to discuss the specialties of Quality of Service testing procedures executed on smartphones and to give according guidance of how to use smartphones for successful QoS testing.

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

2.1 Normative references

The following referenced documents are necessary for the application of the present document.

Not applicable.

2.2 Informative references

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI TS 102 250-2: "Speech and multimedia Transmission Quality (STQ); QoS aspects for popular services in mobile networks; Part 2: Definition of Quality of Service parameters and their computation".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

App: application that can be added by the user to his smartphone

NOTE: Applications have at least one component which is running on user level and which is visible to the user.

host based system: computing system similar to a personal computer acting as controlling entity for the communications device itself (e.g. a USB data stick or another kind of mobile device)

smartphone: mobile device based on an operating system which can be programmatically controlled via a programming interface, in combination with the possibility to run applications on user level

NOTE: The form factor is smaller than that for tablets.

tablet: mobile device which has a display with touch functionality and a display size of about 20 cm or more

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

API	Application Programming Interface
App	Application
ARM	Advanced RISC Machines
CS	Circuit Switched
DL	Downlink
FACH	Forward Access Channel
GUI	Graphical User Interface
HTTP	HyperText Transport Protocol
IP	Internet Protocol
LAC	Location Area Code
MOS	Mean Opinion Score
OS	Operating System
POSIX	Portable Operating System Interface
PS	Packet Switched
QoS	Quality of Service
RAB	Radio Access Bearer
RIM™	Research In Motion, earlier name of the company BlackBerry™
SD	Secure Digital
SMS	Short Message Service
TCP	Transport Control Protocol
UE	User Equipment
UL	Uplink
URL	Uniform Resource Locator

4 Smartphones as a QoS Test Environment

This clause emphasizes the special characteristics of smartphones with respect to QoS tests and presents some of the most popular operating systems.

4.1 Boundary conditions of smartphones

This clause describes the different boundary conditions of smartphones to acquire test related information. Different types of smartphones might be used for that purpose:

- Smartphones sold by manufacturers or providers allow to give an impression how normal users experience the quality of the offered services. Since no adaptations besides the addition of the application used for QoS tests are made, the general behaviour of such a device is expected to be nearly unchanged.

- Several devices allow to replace the delivered operating system by customized versions. Especially in the area of Linux based operating systems, so called custom ROMs (Read Only Memory) are popular to extend the user's rights on his own device. By applying these customized operating systems, additional tools like tcpdump can be executed to gather more detailed information, e.g. related to the network traffic of the device. However, the changed operating system as well as additional tools can have an influence on the performance of the services that are under consideration.

- To achieve a deeper knowledge about internal data processing, additional and mainly manufacturer dependent libraries can be added, e.g. to trace layer 3 signalling data. By adding these components, the mobile devices are turning more and more into specialized measurement engines. Especially in relation to the remaining performance for services under test this has to be considered and to be checked in depth. Performance checks and comparisons before and after applying additional tool libraries are strongly recommended to assure that the influence on the Quality of Service experienced by the user is minimized.

For Quality of Service testing, many smartphones bring lots of capabilities with them which support the testing of QoS in nomadic or mobile scenarios:

- Location awareness: Smartphones often can detect their own location by either using satellite navigation systems or network based location services. Furthermore, integrated position and movement sensors help in detecting changes of the current location.
- Service support: Smartphones in general support many different services which can be used for QoS testing as well (e.g. messaging, voice calls, data transmissions, streaming, application specific protocols).
- Mass storage: Most smartphones carry storage capabilities which allow to work autonomously and to upload collected datasets in a later stage, e.g. if a data connection is available again for this purpose.
- Battery power: The available battery power is one important resource constraint once the smartphone is disconnected from its charging entity. The trade-off between performance and battery power shows a strong influence the higher the requested performance of the smartphone is. Therefore, the test design has to be implemented in an intelligent way to gain the highest possible benefit out of the available energy resources.

4.2 Popular Operating Systems used on Smartphones

Since different operating systems on smartphones reached a remarkable market penetration, this clause gives an overview over some of their main characteristics related to QoS testing.

4.2.1 Android®

The Android® operating system is introduced by Google Inc™. It evolved to version 4 which is able to handle applications on mobile handsets and tablet devices in parallel. Android® is used by many different manufacturers for their devices.

Android® implements a Java based virtual machine on top of a Linux kernel. It covers all aspects like power management, memory management and the life-cycles of the applications. The user access to the Linux environment is restricted for security reasons so not all operations on the underlying Linux environment can be performed without modifications.

Alternative vendors of operating systems offer so called Customized ROMs to enhance the pre-defined functionality of the standard firmware and to grant administrator access to the devices. In Android®, devices gaining administrator rights are called to be "rooted". Applications consist of separate activities that may interact with each other. Therefore, applications are compiled of a set of loosely coupled activities.

Different markets allow the user to download applications to his devices. For developers, all applications could be installed locally on available devices.

4.2.2 iOS™

Released by Apple Inc.™, iOS™ (or called iPhone OS in earlier versions) establishes an operating system on top of a POSIX basis. iOS™ was the first operating system which combined touchscreen functionality with the "always on" Internet of mobile devices. iOS™ is only used on devices release by Apple Inc™.

Again, the user's rights are limited by default. Evolving over versions, iOS™ supports multi-tasking and service instances in the latest versions. Currently, no information related to WiFi networks is available on application level, making use of unrooted devices.

In contrast to Android®, all applications have either to be downloaded via the infrastructure of Apple™ by default or could be distributed via privately owned web servers in a dedicated company environment. Local installations are possible, but are restricted to single developer devices.

4.2.3 Windows Phone 8SM

Introduced by Microsoft CorporationSM, Windows Phone is the successor of the Windows Mobile family of operating systems. Phone 8 is currently available in version 8.0 and implements smartphone capabilities on touchscreen devices. Different manufacturers make use of Windows Phone 8SM as an operating system for their devices.

Some restrictions related to automated QoS testing apply whereas the API command set for information retrieval and test automation has been extended compared to Windows Phone 7SM or previous versions of Windows Mobile.

This is also related to the requirement that all applications have to be checked in detail before being made available for download and installation. Automation of QoS test applications has not been fully supported by this model yet.

Applications are released via the Microsoft WindowsSM Store. Direct access to a limited number of development devices per developer license is granted as well.

As a variant of Windows Phone 8SM, the version Windows RT is available to support applications on tablets running on ARM processors. It uses the same OS kernel as Windows Phone 8SM. However, only applications distributed via the Windows Store can be deployed with Windows RT devices.

4.2.4 BlackberryTM OS

The company Research In Motion (RIM)TM published the operating system BlackberryTM OS which is used on many messaging-oriented devices of this manufacturer. The newest released version X provides many useful API commands to retrieve network related information.

Applications based on BlackberryTM OS have to be checked and signed by RIMTM before they can be installed on according devices. Each developer has to acquire an according set of signing keys in advance.

4.2.5 Symbian®

The Symbian® family of operating systems is available for many years and evolved over time. It turned into an open operating system over time and can still be found on many devices in the field, nowadays mostly related to the low cost price tier. Symbian® has been one of the first available operating systems in the smartphone area.

The access to QoS test related information has always been complex and in many cases not officially supported.

Applications could be installed locally without any need to involve online instances of the vendors.

5 Limitations of Smartphones

To use smartphones as a basis for QoS testing comes with some limitations compared to usual host based systems. These limitations are based on the physical device and on the allowed access to relevant information required for QoS testing.

5.1 Device Limitations

On a physical level, mobile devices come along with limited resources. This covers mainly the areas:

- battery power;
- processing power;
- memory size;
- display size and resolution; and
- speed of the storage media (e.g. like SD cards);
- limitations of the SIM card, e.g. volume caps or bandwidth throttling.

5.2 Operating System Limitations

The characteristics of the smartphone's operating system have a strong influence on the achievable level of automated QoS testing.

Functionality and access to required commands and automation capabilities might be restricted:

- Main hindrances are the missing access to technology information (e.g. network name, network code, cell ID, LAC, etc.).
- Furthermore, some of the OS do not consider any automation capabilities at all. This means that by default any activity is assumed to be user driven (e.g. by handling dialogs on the GUI) and thus leads to the missing possibility to automate testing.
- According to manufacturer restrictions, network related information may be limited strongly. E.g. certain OS provide only very basic information (i.e. if connected via a mobile network or via Wifi) or even hide complete technology related information.

5.3 Approximations due to working on application layer

QoS testing on smartphones is influenced by the level of approximation of the user experience with the device. The more technical data is required, the more capabilities should be enabled on the device.

To be close to the user's experience, all activities done for QoS testing could be considered on application level. As usual, this generates a dependency on the specific application implementation, e.g. for messaging applications or video streaming scenarios.

In host based systems, this dependency is eliminated by mapping the user specific events (i.e. trigger points) to lower and mostly standardized communication protocols. Examples for that are the tracing of packet information conveyed by the TCP/IP messages or the tracing of layer 3 signalling messages.

Since the application's events are triggered before the underlying protocols can become active and reach their own trigger events, there might be a deviation in the timing between the events on different levels. This should be considered especially if the results generated by implementations on different operating systems are benchmarked.

Furthermore, the application developer has not the chance to access lower layer information on smartphones directly. Rooting or jail-breaking might grant additional rights to make use of tracing tools, but beforehand the effects of the additionally installed component should be checked in terms of performance loss. This is recommended to assure that the results generated by this architecture are still comparable with the user experience of an untouched user device.

5.4 Attended, unattended and automated Operations

In most cases, an automated test execution is desirable to generate statistically relevant sample numbers. This means the test execution is performed in cycles without further human interaction. While test automation is a very basic feature, the operation can be done in an unattended or in an attended manner.

In an unattended manner, the automation solution has to be highly error-tolerant and capable of handling all occurring conditions and errors. This requires a high level of complexity of the implementation, but allows to operate the test system without human interaction. Unattended operations might be chosen if the test system itself cannot be reached afterwards or if the services under consideration are very stable so that there are no changes expected in their implementations for a long term. In these cases, test systems can be deployed at dedicated locations and should operate autonomously. Often, remote monitoring and interception is implemented in the testing solution on top to allow a minimum of operational support.

The attended mode of operations makes also use of automation, but has to reserve some human resources for monitoring the test execution and for resolving problematic situations manually. This possibility reduces the level of complexity of the implementation and allows to increase the speed of new implementations. However, the system has to be kept reachable to assure this mode of operation.

5.5 Further Performance Considerations

The generation of QoS parameters on a smartphone device is related to the limited resources of such a device. Therefore, it should be avoided to involve further processes or activities on the device in parallel while executing a test. Every activity which might potentially influence the QoS results of the service under test should be avoided since the radio power, the battery power and the bearer capacities are limited resources and may influence each other.

For example, while executing a data test, there should be no image detection of the built-in camera running in parallel. Furthermore, parallel activities which make use of data bearers should also be avoided during test periods in general. Therefore, the device used for QoS tests should be cleaned up from any applications which might generate interfering traffic. Additionally, the system settings should be adapted in a way to minimize this traffic as well.

One exception from this general requirement applies if the combination of services is the chosen scenario for the services under test. This so called Multi-Service Testing is discussed separately in the present document.

6 Basic Settings for QoS Assessments

6.1 General

The present document mostly re-uses the abstract definitions made in TS 102 250-2 [i.1] to reflect the user's view.

Figure 1 shows the generic approach of communication protocols used in mobile communications. While the user plane conveys all application specific data (the "user data"), the control plane is responsible for controlling and monitoring the resources via signalling protocols. Finally, all protocol information has to be multiplexed for transmission and reception via the physical layer (Layer 1), the so called air interface.

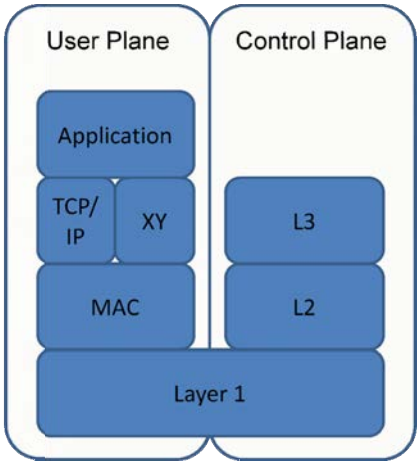

Figure 1: Generic protocol structure in mobile communications

6.2 Trigger points

However, due to the different underlying environments and operating systems, the more detailed information dealing with protocol messages in the trigger point definitions could hardly be re-used. Therefore, a dedicated and OS specific mapping of trigger points is introduced to come as close as possible to the intended view on parameters and their trigger points.

Overall, this necessary mapping leads to the situation that the trigger points cannot be determined as accurate as on host based systems. Some uncertainty in the resulting QoS parameters values is introduced to direct comparisons between smartphones and host based systems.

Only in the cases where the smartphone provides an access to signalling information on layer 3, the QoS parameter definitions made in TS 102 250-2 [i.1] could be re-used.

6.3 Timeouts

The general concept of timeouts is the same as described in TS 102 250-2 [i.1]. One difference could be the way of how outstanding responses are detected.

On smartphones, timeouts are mostly identified on API level by analysing returned error codes of API calls or similar information. The direct detection of missing protocol messages is mostly not possible on layers below the application layer.

7 How to determine QoS parameters in smartphone QoS testing

This clause explains and describes how QoS parameters could be organized to meet smartphone specific requirements and changes. Since many of the QoS parameters defined in TS 102 250-2 [i.1] are considered, the given example is only illustrative.

7.1 Web Browsing HTTP

7.1.1 HTTP Service Non-Accessibility [%]

This smartphone specific QoS parameter refers to the QoS parameter "HTTP Service Non-Accessibility [%]" defined in TS 102 250-2 [i.1].

Remark: QoS on smartphones should re-use existing definitions wherever possible to avoid duplications of content and to avoid undesired dependencies between documents.

7.1.1.1 Trigger Points

Specific trigger points of this parameter are also defined in TS 102 250-2 [i.1]. If no smartphone specific requirements apply, these definitions should be reused. Remark: This information is taken from [i.1], used as a template for the different discussed operating systems.

7.1.1.2 OS specific mappings

According to the OS specific implementations, the generic trigger points could be mapped as follows:

Event from abstract equation	Android OS® (version x)	iOS™ (version x)	Windows Phone 8℠ (version x)	Blackberry OS™ (version x)	Symbian® (version x)
Service access attempt					
Successful attempt					
Unsuccessful attempt					
NOTE: This table format should be adapted to the used table scheme of TS 102 250-2 [i.1] (making assumptions for technology, layer, reference point).					

7.1.1.3 Android OS®

This is a short example how the according events could be realized in a programmatic way in an Android® environment (versions 2.x and above):

```
URL myURL = new URL(strURL);

HttpURLConnection conn = (HttpURLConnection) myURL.openConnection();

m_nResponseCode = conn.getResponseCode();   // First server touch
```

8 Test Methodology

In general, the basic principles of QoS testing known from host based systems can be transferred to smartphone devices. This means that the generic usage of services is comparable. However, the limitations in memory, battery power and processing power have to be considered to gain the maximum possible benefit out of this environment: Different operating systems implement some intelligence to detect user activity. This is done to detect when power consuming components like the display are not further used and can be switched off. In addition to this, the speed of the processor might be reduced in parallel which also influences the available performance for testing QoS of services. So the trade-off between power savings and the reduction of calculational power has to be considered in detail per smartphone device.

It is not sufficient to check only the general properties of a specific operating system the device manufacturers can handle different aspects according to their individual specifications.

Another important restriction is related to the memory management: Some operating systems try to keep the available memory permanently filled with the latest used applications. This strategy is implemented to assure short start-up times and to enable a fluid change between applications. On the other hand, if multiple applications are running actively, the operating system might drop activities due to its memory management restrictions. Different operating systems terminate applications or services without notification if there is a resource limitation. This needs to be kept in mind when implementing and using applications for testing QoS purposes.

9 Multi-Service Testing

More and more smartphones come with capabilities that allow them to use multiple services in parallel. Technically, speaking, these devices make use of multiple radio access bearers ("Multi RAB") in parallel. This is either achieved by implementing chipsets which may handle two or more different radio bearers at the same time or by even implementing multiple independent chipsets.

From the user's perspective, this technology allows different actions to take place simultaneously. In many cases, data downloads are executed while the user is having a voice call. Or vice versa, while downloading data via the data bearer (e.g. music files, software updates), an incoming voice call can be handled in parallel. Having this in mind, QoS tests in these scenarios become an interesting and important area.

Up to now, competing actions are mostly related to one service area. E.g. it is possible to receive a SMS during a voice call since both are related to circuit switched services. Or, an existing data context is suspended and reactivated when a SMS is incoming. These limitations are removed with multi-service usage which means that both services can be continued without interruption.

9.1 Basic scenarios for stand-alone devices

To illustrate the basic constellations of services for multi-service testing, this clause introduces different scenarios. These scenarios make use of only one device to state the QoS a user would experience when using such a service combination.

Starting with an explanation of the basic principle, the two scenarios "CS in PS" and "PS in CS" are derived.

9.1.1 Basic principle

To execute multi-service testing, both interfering services have to be defined in advance. One needs to consider that the choice of these services under test determine the main characteristics of the test. It is possible to choose the services both from the same domain (both circuit switched or both packet switched), but the higher requirements with respect to the Quality of Service of each of the services is under consideration if both services are from different domains (one service circuit switched, the other packet switched). The further discussions will handle only "mixed domain scenarios".

Following figure 2, these scenarios require the definition of a number of parameters before test execution:

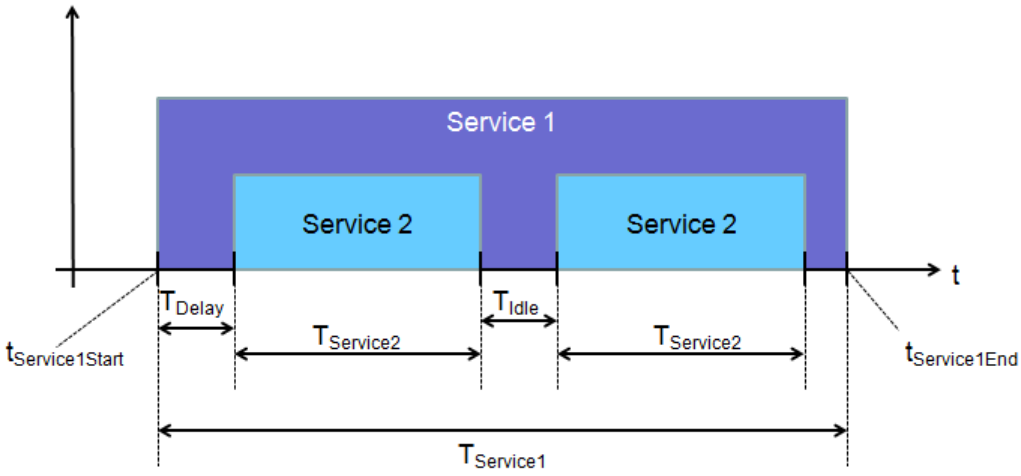

Figure 2: Basic setup of intermitting services

The intermitting characteristics of the services can be described by a set of parameters:

- $T_{Service1}$: Duration of usage of service 1
- T_{Delay}: Time delay before starting service 2
- $T_{Service2}$: Duration of usage of service 2. The duration can be a fixed duration or according to a known statistical distribution. Depending on the chosen service, there might be constraints related to a minimum or maximum duration of usage of service 2
- $N_{Service2}$: Number of service usages of service 2 in parallel to an ongoing usage of service 1
- $T_{IdleService2}$: Idle period between service usage phases of service 2. The duration of the idle periods can be a fixed duration or according to a known statistical distribution. Depending on the chosen constellation, there might be a reasonable minimum or maximum duration of the idle period, e.g. either to turn down all radio resources on the one hand or to keep already established radio resources up and running on the other hand

To avoid ambiguities, the time parameters should use the same time unit (e.g. milliseconds, seconds, minutes).

$T_{Service1}$ has to be chosen in a way that its value is equal or higher than the sum of T_{Delay} , $N_{Service2}$-1 times the T_{Idle} interval and $N_{Service2}$ times the service usage time $T_{Service2}$.

In the generic case with variable idle times and usage times of service 2, the parameter $T_{Service1}$ reads:

$$T_{Service1} \geq T_{Delay} + \sum_{i=1}^{N_{Service2}-1} T_{Idle_i} + \sum_{i=1}^{N_{Service2}} T_{Service2_i}$$

In fact, if statistical processes are involved for the idle periods and the service usage durations of service 2, the parameter $T_{Service1}$ cannot be predefined.

The duration of active phases of service 2 and its inactive phases could be set in relation as well, giving the intermitting ratio $IR_{Service2}$:

$$IR_{Service2} = \frac{\sum_i T_{Service2_i}}{T_1}$$

Therefore, $IR_{Service2}$ defines the duty cycle of service 2.

9.1.2 Characteristics and limitations

Based on the discussed parameters, the outcome of such a multi-service test has always to be mapped against its given characteristics. Changes in the intermitting ratio of service 2 might have a direct influence on the generated QoS parameters.

Furthermore, some more points have to be considered to assure a correct interpretation of results:

If service 2 is a data service using TCP as transport protocol, there might be re-curring slow start effects be visible each time service 2 starts. In detail, if the slow start can be observed depends on the previous service usage in combination with the chosen idle time between service usages.

The combination of an ongoing service 1 with an intermitting service 2 requires an enhancement of preconditions compared to usual QoS testing: Whenever service 2 is about to be used, it should be checked that service 1 is still ongoing as intended. Otherwise the service usage would be a single service testcase and therefore results would be mixed up.

Another consequence resulting out of this constellation is the fact that both services are statistically dependent. Each cut-off of service 1 will result in a reduced number of service usages of service 2. This might have an influence on either the QoS parameters for service 2 or their uncertainty or both.

Figure 3 illustrates possible effects of multi-service testing for the combination of a voice call with an intermitting data session. Mobile devices might experience limitations in the resulting QoS parameters during multi-service usage phases. This is related to the necessity to process voice and data information in parallel. Furthermore, the available radio power has to be split up to two services respectively radio bearers.

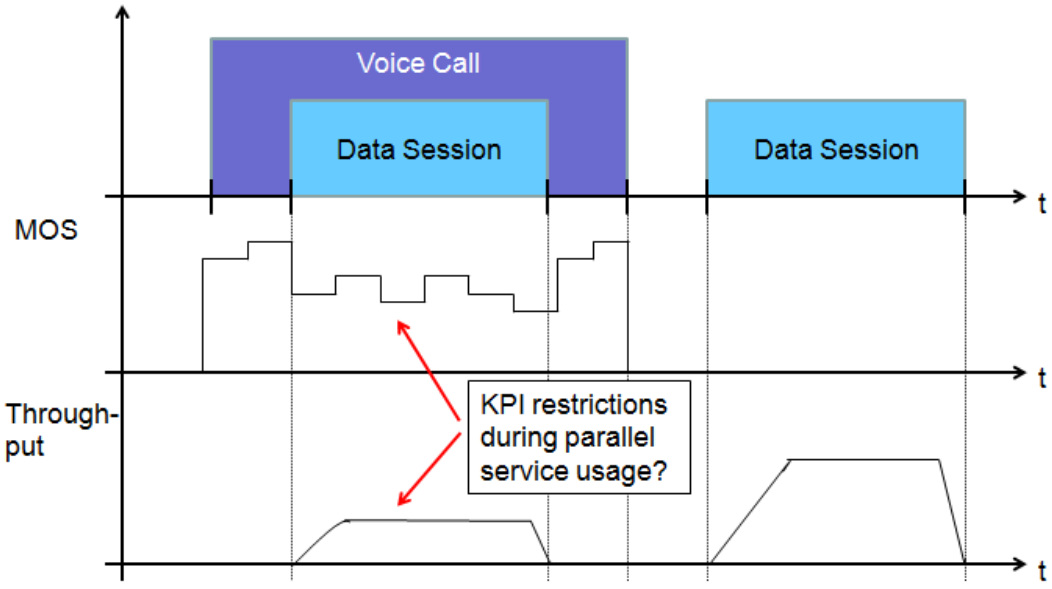

Figure 3: Effects of multi-service scenarios on QoS parameters

Another point to be considered is related to the characteristics of the chosen UE. The behaviour of UEs may differ in their implementation of the always-on functionality: UEs may stay in Cell FACH state, fall back to idle state or make use of features like Network Controlled Fast Dormancy.

9.1.3 Example: Data as a background service ("CS in PS")

This clause describes a typical situation in which an ongoing data session is influenced by voice calls (figure 4).

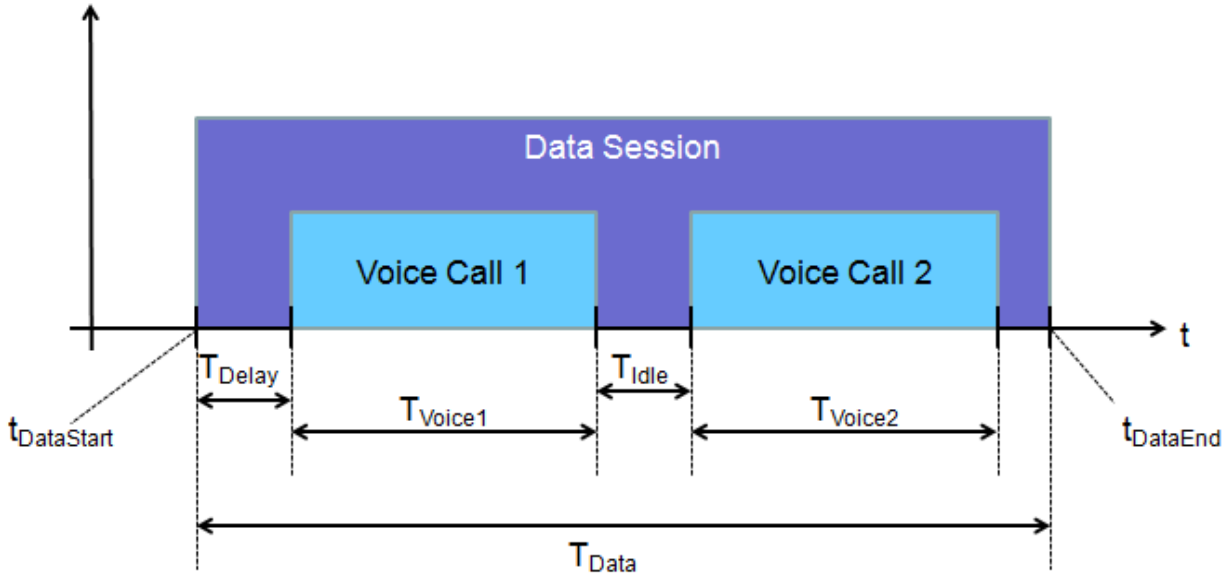

Figure 4: Data as background service, voice as intermitting service

In detail, it is recommended to make a distinction of scenarios for incoming and outgoing calls since radio bearers are used differently in both cases.

The same applies to the direction of voice transfer, meaning the uplink and downlink transmission may have different influence on the stated QoS parameters. Therefore, it is recommended to keep track of the kind of voice transmission used in each phase.

9.1.4 Example: Voice as a background service ("PS in CS")

A similar scenario exists if an ongoing voice call is accompanied by data sessions like depicted in figure 5. Service 1 is the voice call, service 2 is given by some intermitting data sessions.

Here, the same parameters might be used to describe the timely structure of such a test case (figure 5).

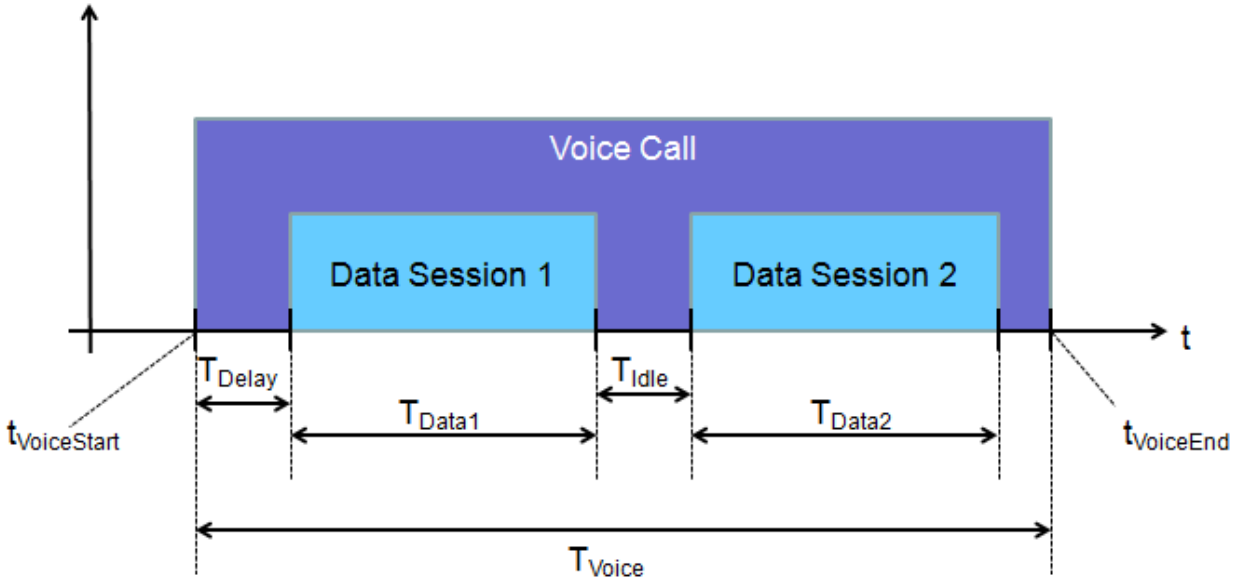

Figure 5: Voice as background service, data as intermitting service

9.1.5 Multi-service QoS parameter constellations

Since both services to be used in multi-service tests could in general be related to both transmission directions, the resulting parameters need to reflect this individually. E.g. a CS in PS constellation of voice and data would then generate QoS parameters with this granularity, always shown according to the nomenclature of "service 2 in service 1":

- Voice DL in Data Session DL
- Voice DL in Data Session UL
- Voice UL in Data Session DL
- Voice UL in Data Session UL

Figure 6 depicts these combinations of voice and data services.

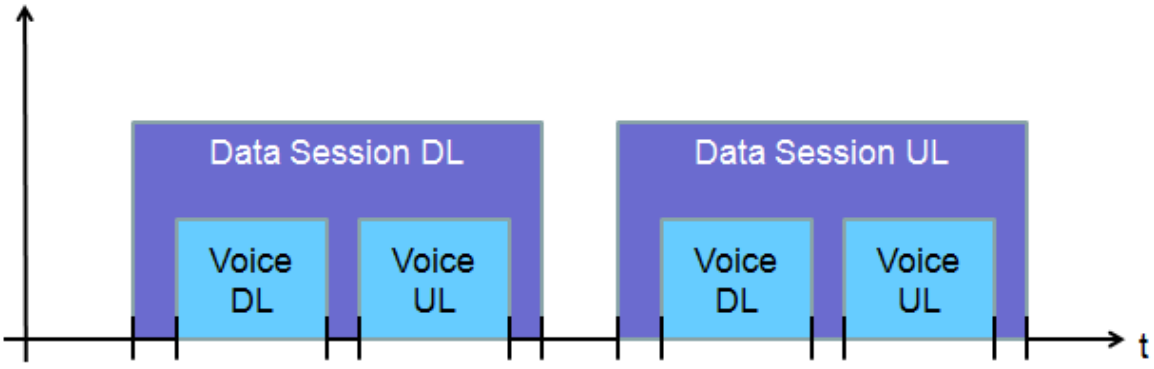

Figure 6: Service constellations considering the transmission direction of the content

Especially the last scenario has implications on the sharing of available radio resources in uplink direction and on sharing the available transmission power. The overlapping of both effects may double the limitations of multi-service testing.

In detail, all parameters should carry the following quadruple of parameters in addition to the parameters specified in TS 102 250-2 [i.1]:

Parameter Name (Service 1, Service 2, Direction Service 1, Direction Service 2)

9.2 Extended scenarios for device pairs

Having discussed multi-service scenarios which try to reflect the user's experience, there might be extended scenarios where the comparison of single-service scenarios in a direct comparison with multi-service scenarios is of interest. This test constellation is working as a differential test.

In these cases, there should be a combination of tests in parallel:

1) a single-service test generating QoS parameters which is used as a reference test

2) a multi-service test which generates the according QoS parameters

After test execution, the generated QoS parameters out of both test parts can be set in relation. E.g. if the voice quality in the single-service scenario shows low MOS values, the multi-service scenario is not expected to show better results.

To allow this comparison, both tests should be executed at the same date and place in the same radio environment (i.e. load situation, signal to noise ratio, interference situation, etc.) using the same type of device. With these results, the QoS experience of users can be assessed via a direct comparison.

9.3 Download and Upload Throughput Tests in parallel

In many cases, throughput tests related to data services are dedicated to a simple scheme where download and upload QoS parameters are determined in a sequential order.

This shows two idealizing effects with respect to the simultaneous usage of both transport bearers:

- The available bandwidth does not need to be shared between different services or applications.
- The return channel (e.g. uplink channel while executing a download test) is more or less kept idle and just has to convey the generated acknowledgement messages. They will reach the peer entity quite fast since queuing effects are avoided in general.

Executing download and upload throughput tests in parallel, this idealized environment has to be left and a contrary working mode takes place: The bandwidth of both transport bearers (downlink, uplink) have each to be shared for the forward transmission of bulk data and the reverse transmission of feedback data like acknowledgements.

On top, the usage level of both bearers influences the overall bandwidth as soon as feedback messages are delayed due to queuing effects.

Overall, the simultaneous use of downlink and uplink bearers generate a complex scenario with strong dependencies between the single unidirectional scenarios. This has to be considered when designing, executing and assessing according test campaigns.

History

Document history		
V1.1.1	October 2013	Publication