YouTube's DASH implementation analysis

Javier Añorga, Saioa Arrizabalaga, Beatriz Sedano, Maykel Alonso-Arce, and Jaizki Mendizabal

***Abstract*—**As long as *YouTube* is one of the most used services of the World Wide Web, it consumes an enormous quantity of bandwidth along the network. In order to maintain a high efficiency in this bandwidth management requirement, *YouTube* needs to adopt (and renew) highly efficient video streaming techniques. These changes produce a deprecated literature about *YouTube* service traffic characterization. This work reports an analysis of the recent DASH adaptive video streaming technique adopted by *YouTube*. In addition, this article includes the state of art about the literature with regard to the *YouTube* traffic characterization and analyses the DASH implementation of *YouTube,* reporting the relationship between download bandwidth consumption and video quality obtained and its performance.

***Keywords*—**YouTube, DASH, streaming, bandwidth, video quality.

I. Introduction

YouTube is one of the most popular services on internet, being the third most visited web site in the world [1], and its traffic has a great impact over mobile and fixed networks. With this great popularity and bandwidth intensive demand, *YouTube* presents a challenge for Internet Service Providers (ISPs) in order to offer a good quality for the consumed download services by clients. Therefore, the analysis and characterization pattern of this service traffic is important.

YouTube's traffic has been studied and documented, such as in [2] and [3]. *YouTube*'s videos are transported by HTTP over TCP. Consequently, YouTube has not to cope with lost or reordered packets, and the only quality degradation which may be caused by transmission, is a stalling of the video. However, using the Dynamic Adaptive Streaming over HTTP (DASH) technique, *YouTube* is able to switch the video quality based on the link capabilities. The main outcome of this feature is that if on *YouTube*'s player quality parameter is set on "*auto*", *YouTube* can adapt the bitrate of the video based on the client's available bandwidth.

J. Añorga is with Ceit and Tecnun, Parque Tecnológico de San Sebastián, Paseo Mikeletegi, Nº 48, 20009, Donostia - San Sebastián (phone: +34 943 212800 ext. 2983; fax: +34 943 213076; e-mail: jabenito@ceit.es).

S. Arrizabalaga is with Ceit and Tecnun, Parque Tecnológico de San Sebastián, Paseo Mikeletegi, Nº 48, 20009, Donostia - San Sebastián (e-mail: sarrizabalaga@ceit.es).

B. Sedano is with Ceit and Tecnun, Parque Tecnológico de San Sebastián, Paseo Mikeletegi, Nº 48, 20009, Donostia - San Sebastián (e-mail: bsedano@ceit.es).

M. Alonso-Arce is with Ceit and Tecnun, Parque Tecnológico de San Sebastián, Paseo Mikeletegi, Nº 48, 20009, Donostia - San Sebastián (e-mail: maarce@ceit.es).

J. Mendizabal is with Ceit and Tecnun, Parque Tecnológico de San Sebastián, Paseo Mikeletegi, Nº 48, 20009, Donostia - San Sebastián (e-mail: jmendizabal@ceit.es).

The main objective of this work is to find the relationship among the *YouTube*'s downloaded video quality level, the consumed *YouTube*'s video bandwidth and the available download bandwidth from the perspective of an access point to the Internet. This work uses a Home Gateway (HG) as the access point to the Internet with a *YouTube* client inside the Local Area Network (LAN). In addition, the time that *YouTube*'s DASH implementation needs to adapt to available bandwidth fluctuations is characterized.

This paper is organized as follows. Section II describes the state of art about DASH and the adaptive streaming. Section III focuses on the related work about *YouTube* characterization, describing some key points of *YouTube*'s behavior. The test-bed deployment for measurements is depicted in Section IV. Results obtained from the measurements are shown and analyzed in Section V. Finally, conclusions and future lines are exposed in Section VI.

II. DASH and adaptive streaming

In the past, streaming services were offered over UDP (User Datagram Protocol) transport protocol; however, nowadays, with the increasing bandwidth connection at households and the popularity of World Wide Web, the media content can be efficiently delivered now in larger segments using HTTP (HyperText Transfer Protocol). It is motivated due to two main reasons. Firstly, HTTP is more firewall friendly because most of the firewalls are configured to allow HTTP outgoing connections, and, secondly, with HTTP streaming, the client manages the streaming without having to maintain a session state on the server.

However, basic progressive HTTP based streaming is not suitable for environments which may have a considerable high bandwidth fluctuations. The video stream has to adapt to the varying bandwidth capabilities in order to deliver the user a continuous video stream without any stalls at the best possible quality for the moment. This is achieved by adaptive streaming over HTTP.

There are adaptive HTTP-Streaming based proprietary systems like *Smooth Streaming* (from *Microsoft*), *HTTP Dynamic Streaming* (from *Adobe*) or HTTP Live Streaming (from *Apple*). Each solution reports its advantages and drawbacks depending on the circumstances, as it is detailed in [4]-[6]. DASH [7]-[8], or *MPEG-DASH,* is an emerging ISO/IEC MPEG standard that is an extension of the classic HTTP streaming. Through DASH technology, the video quality level can be delivered according to the current network conditions. Several representations of a video clip are generated based on quality/bit rate level and each representation is divided into fragments (usually from 2 to 10 seconds of length) [9]. In order to avoid rebuffering due to buffer starvation, the video player usually chooses a quality level that has a lower bit rate than the measured available bandwidth. In this sense, the video download rate can be higher than the video playback rate (or at the least the same if a full video buffer is present). Moreover, the use of DASH also results in bandwidth consumption saving as it is reported in [10].

Because of the benefits that HTTP-Streaming based technology implies and due to the fact that *DASH* is a company-independent standard, and it allows saving bandwidth resources, nowadays, *YouTube* and other popular services, such as *Netflix*, have implemented DASH as the preferred streaming technology rather than FLV (Flash Video) streaming.

III. Related Work

There are several studies that have characterized the *YouTube* traffic along the past years. However, until the date when this work was written, there are no so much studies about the *YouTube* behavior with DASH streaming standard [10]. For example, in [3], the *YouTube* service from the viewpoint of traffic generation in the server's application layer is characterized focused on FLV based video clips. An analysis of how content distribution in *YouTube* is realized is done in [11], conducting a measurement study of *YouTube* traffic in a large university campus network.

The article [12], published in 2011, studies the *YouTube* streaming characteristics and operation, and it reports for mobile devices: "In fact, the mobile devices cannot buffer the entire video so the player progressively requests portions according to the evolution of the playback". This reported issue in 2011 presents the same properties as the DASH operation detailed in previous section.

In [3] also two phases of *YouTube* streaming is reported. Frist occurs an initial phase where there is a significant burst of data. This phase is called "burst phase". After this initial burst, the receiving download data rate at *YouTube*'s player is considerably reduced. This second phase is called "throttling phase". In addition, from the analysis of FLV *YouTube* videos, [13] describes that during the initial burst phase, the amount of data sent by the *YouTube* server is related to the transmission rate during the throttling phase.

This work is focused on the analysis of DASH behaviour of *YouTube* video streaming to desktop devices, more precisely using *Google Chrome* as web browser software. This article tries to fill the gap in literature (due to the constant changes in streaming techniques) about *YouTube*'s DASH implementation characterization providing a data report about *YouTube* download bandwidth consumption versus the *YouTube* quality level obtained. In addition, this work reports how *YouTube*'s DASH implementation behaves in terms of time response to available bandwidth fluctuations.

IV. Deployed Test-Bed

In order to carry out this the *YouTube*'s *DASH* analysis, the architecture shown in Fig. 1 is used. A PC is used with *Windows 7* as running operative system and *Google Chrome* as web browser. Client machine is connected to the Internet through the HG. *Client – YouTube's server* download bandwidth measurements are done in the HG. An Additional *Java* application is running on the client machine. This *Java* application is called *Commander* and commands *Google Chrome* to open a *YouTube*'s testing web page served from the HG.

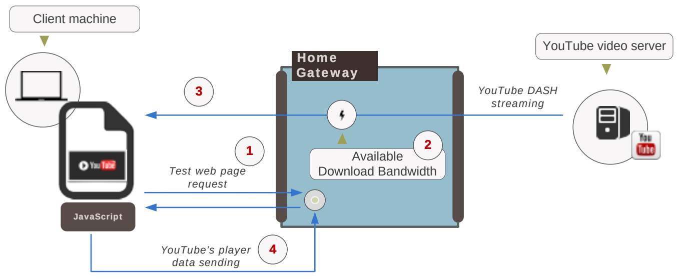


Fig. 1 Test-bed architecture and test steps.

The measurements trend to be as non- invasive as possible. The following steps are done:

1) *Commander* on client machine calls the testing web page from HG with 3 possible arguments:
   - *YouTube*'s video id: the *YouTube*'s video id to invoke at *YouTube*'s video player (which video reproduce from *YouTube*).
   - *YouTube's* video quality: fix the quality of the video. Possible values are: *tiny* (240p), *medium* (360p), *large* (480p), *hd720* (720p), *hd1080* (1080p) and *auto*.
   - HG allowed download bandwidth: the allowed download bandwidth from *Client – YouTube's server* download connection at HG.
2) The requested available download bandwidth is applied at HG.
3) The served web page contains an embedded *YouTube*'s player configured to automatically start to play the *YouTube*'s video id requested at configured quality level.
4) The variables of this *YouTube* embedded player are extracted by *JavaScript* code and sent back to the HG in order to centralize all the measurements.

Once the test finishes a post process of the monitored data is done with *Excel* or *Matlab* software.

V. Testing and Results

This section reports the results obtained from the realization of 4 different tests. The first test (A), details the *YouTube* DASH video streaming phases with no bandwidth limitations at the HG. The next three tests (*B*, *C*, *D*) study the relationship between the average download bandwidth consumption of the requested video and the *YouTube* quality level obtained. These tests are replicated over the same 199 videos, and all of these videos are requested from *YouTube Data Api* [14] with parameters *videoEmbeddable* (only videos that can be embedded in a web page) and *videoSyndicated* (only videos that can be played outside of *youtube.com*) set on *true*. Test *B* tries first to obtain the average download bandwidth consumption of a *YouTube*'s video for a fixed *YouTube*'s quality level. Then, the obtained average bandwidth is used to set an allowed download rate and it is checked whether the corresponding quality level is obtained. Test *C* reports a ratio out of the 199 test videos of which *YouTube* streaming video quality level is obtained restricting the available download bandwidth at the HG.

Finally, test *D* depicts the detailed DASH adaptation pattern for available bandwidth changes, where the time the player needs to notice the change has been measured. In addition, the time delay between the quality request and the video quality change is also quantified.

A. *ZeroTest*

This test reports the *YouTube* DASH implementation behavior with no bandwidth restrictions at HG. The video quality level is set on *auto* and *YouTube* player decides to request *hd1080* because of the high available bandwidth that sees on the link. This test is shown in Fig. 2 illustrating the download video rate and the buffer state.

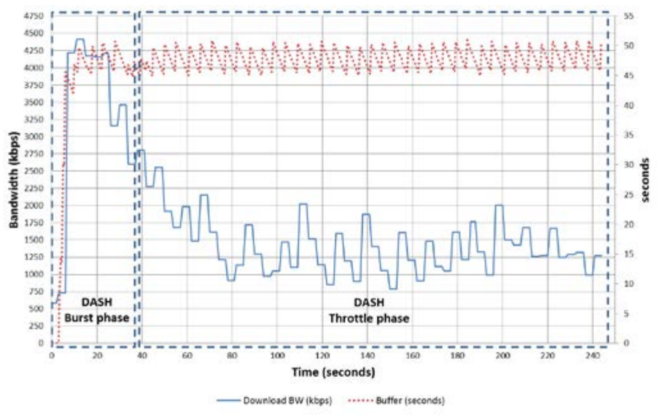


Fig. 2 *ZeroTest* result graph. From approximately second 0 to 35 the first DASH burst phase occurs. After this period, it follows the throttle phase.

From second 0 to approximately second 35 occurs the DASH burst phase and the download rate of video reaches 4400 kbps as maximum value. At this phase the *YouTube*'s player buffer is being filled through a high request of video fragments. When the buffer is full, in this case storing about 45 seconds of video playback, a second throttle phase follows. At this stage the amount of received data is significantly reduced (about an average of 1300kbps), maintaining the download rate (and the player buffer state) according to the video playback rate.

B. *avgBW@fixedYTquality(4min)*

This test is intended to report the average download bandwidth consumption of *YouTube*'s videos given a fixed *YouTube*'s requested quality. The test is done over the reproduction of the first four minutes of each one of the 199 videos. For five different quality levels (*small*, *medium*, *large*, *hd720* and *hd1080*) the average bandwidth consumption per video is shown in Fig. 3. It can be appreciated that values show an increasing average bandwidth consumption when requested quality level is increased also.

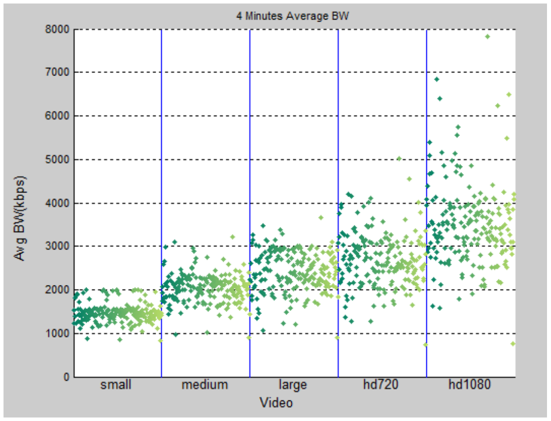


Fig. 3 Average bandwidth consumption (kbps) in first 4 minutes of *YouTube* video playback.

Table I summarizes the measures obtained and depicted in Fig. 3. The average bandwidth consumption per *YouTube*'s quality level of the 199 videos and the standard deviation per *YouTube*'s quality level measurements are shown. It is relevant to remark that the dispersion of the data measured increases with the quality level.

Table I Average bandwidth (kbps) and std. deviation per *YouTube*'s quality level (199 videos). 4 minutes of each video playback.

Quality	**Avg. BW (kbps)**	**Avg. BW std. dev.**
small (240p)	1485.11	225.04
medium (360p)	2061.22	348.46
large (480p)	2446.99	485.66
hd720 (720p)	2737.27	632.22
hd1080 (1080p)	3541.30	951.19

Then, the maximum available download bandwidth is fixed in the HG with the average bandwidth acquired from Table I. The result obtained is that most of the *YouTube*'s videos take a superior quality level than it is expected from Table I, especially for lower intended qualities.

As a second approach, the average bandwidth consumption of the throttle phase has been calculated for the same set of videos. Fig. 4 depicts the results obtained discarding the data of the first two minutes of the previous test. As it is expected, the average bandwidth consumption of throttling phase per quality level also increases when the video quality level increases. Since the throttle phase maintains the player buffer with an approximately constant buffer length, the download bandwidth consumption depends on video playback bit rate.

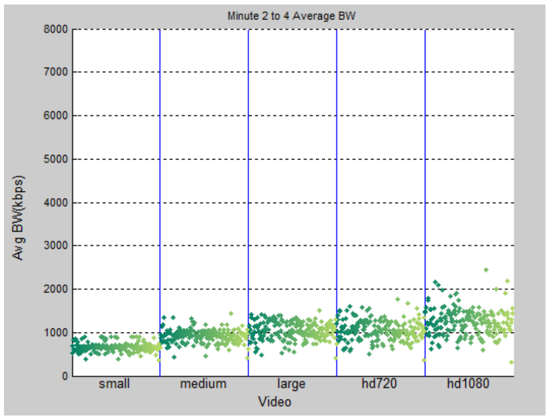


Fig. 4 Average bandwidth consumption (kbps) in minute 2 to 4 of *YouTube* video playback (throttle phase).

Table II summarizes the measurements obtained and depicted in Fig. 4. The average bandwidth consumption per *YouTube*'s quality level of the 199 videos and the standard deviation per *YouTube*'s quality level measurements are shown. Again the dispersion of the average bandwidth consumed per video increases with the quality level, although this dispersion is sensibly lower. The average bandwidth consumption at throttle phase is nearly reduced by a percentage of 60%.

Table II Average bandwidth (kbps) and std. deviation per *YouTube*'s quality level (199 videos). 2 minutes of each video playback.

Quality	**Avg. BW (kbps)**	**Avg. BW std. dev.**	**% of BW reduction**
small (240p)	668.38	103.17	54.99
medium (360p)	922.04	157.14	55.27
large (480p)	1052.77	211.05	56.98
hd720 (720p)	1065.22	240.72	61.08
hd1080 (1080p)	1252.65	318.61	64.63

Again, the same proof is done. If the maximum available download bandwidth is fixed with the average bandwidth obtained in Table II the result obtained this time is that most of the *YouTube*'s videos trend to take an inferior quality level than it is expected from Table II, this time especially if the high definition qualities (*hd720* and *hd1080*) are tested.

C. *YTQuality@fixedAvailBW(4min)*

This test is intended to report the quality level that the *YouTube*'s DASH implementation finally choses when a fixed available downstream bandwidth is applied at HG and *auto* quality if configured. As the previous case, the test is done over the first four minutes of video playback.

Fig. 5 it shows the percentage of the tested videos that take a determined *YouTube*'s quality level with a fixed available download bandwidth (*tiny* quality stands for 144p resolution). The figure shows, once more, the increasing bandwidth consumption with the quality level. Whereas the *small* and *medium* qualities present high percentage of videos concentered around a determined value of available bandwidth, *hd720* presents the most entropy in finding a corresponding available bandwidth value.

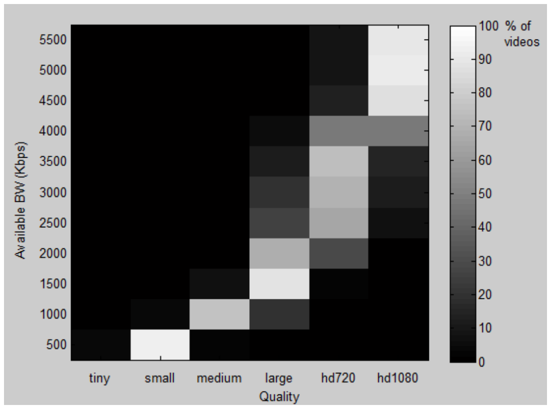


Fig. 5 Percentage of *YouTube*'s tested videos that finally take a determined quality (Xaxis) with a fixed available download bandwidth (Yaxis).

In Fig. 6 it is shown the percentage of *YouTube*'s tested videos that take a determined quality level or superior given a fixed allowed download bandwidth. For example, for the case of *hd720*, it is needed to set an available download bandwidth of 4000kbps to estimate with around 80% of probability that the video quality level is going to be set as *hd720* when the *auto* quality is selected.

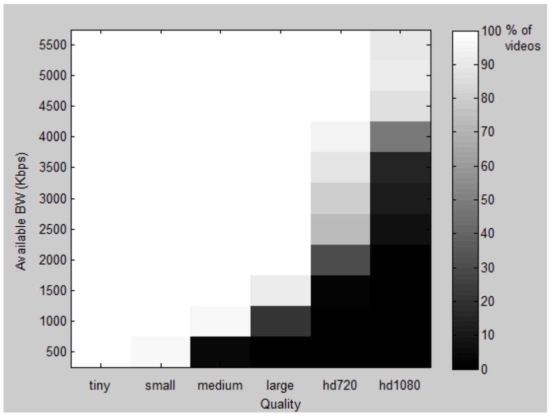


Fig. 6 Percentage of *YouTube*'s tested videos that finally take a determined quality or superior (Xaxis) with a fixed available download bandwidth (Yaxis).

Fig. 6 can be used to obtain an estimation of the probability of watch *YouTube*'s videos at a determined quality level or superior with an allowed maximum download rate. It provides valuable information for bandwidth management in the HG settings.

Table III Summary of results.

Allowed BW change (seconds)	Quality requested (seconds)	Player quality request (seconds)	Video quality change (seconds)	Delay for quality request (seconds)	Delay quality change (seconds)	Player buffer state (seconds)
0	*small* (240p)	0	Not applicable	Not applicable	Not applicable	Not applicable
35,83	*medium* (360p)	58,05	104,46	22,22	46,42	45,23
120,83	*large* (480p)	137,70	244,47	16,86	106,78	105,58
262,82	*hd720* (720p)	275,82	464,41	13,00	188,59	187,84
493,83	*hd1080* (1080p)	532,77	619,51	38,94	86,74	85,97

D. *YTQuality@BWScale*

This last test is intended to report the time that *YouTube* player lasts to show a requested player quality operating in *auto* mode. A monotonically increasing scale of maximum available download bandwidth steps at HG is used for this test, which have been selected based on the knowledge acquired in the previous tests. This test is applied over a selected *YouTube*'s video as an example, and it is depicted in Fig. 7. The figure shows the download bandwidth consumption of the *YouTube*'s player in kbps, the available download bandwidth set in the HG in kbps, *YouTube*'s player buffer in seconds and the marks representing the instant of *YouTube* quality request (triangle marks) and effective quality change in the played video (diamond marks).

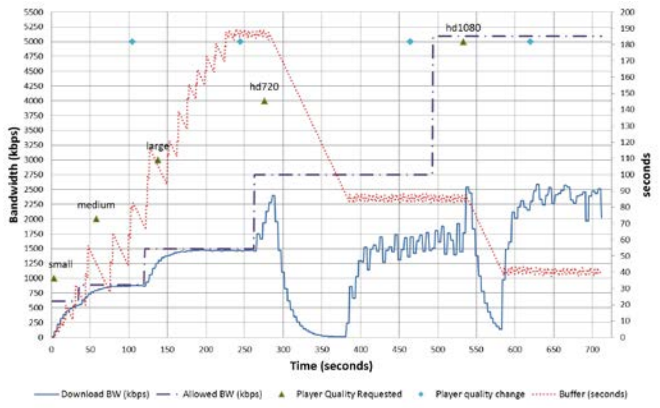


Fig. 7 Monotonically increasing scale of maximum available download bandwidth test.

Results obtained in Fig. 7 are summarized in Table III. The table shows that the time delay for the player to detect network bandwidth fluctuations goes in the range from 13 to 40 seconds. Then, the player requests the new quality but this is not immediately shown to the user. In fact, the time delay for the quality change measured in the test and shown in the table (column *delay quality change*) is approximately the same as the amount of seconds of video that embedded player's video stores. Due to this fact, when the request is made with a high amount of video stored in buffer the expected quality is shown with a relative high delay of time (approximately 3 minutes for *hd720* quality request at the depicted test).

The use of a large buffer involves a lack of response in showing the quality requested with regard to a decision of change the quality requested by the player due to an available bandwidth fluctuation. However, a large buffer prevent video stalling and rebuffering events, and it also allows the user to instantly seek and play any part of the video that is inside the buffer range without the stalling of the rebuffering event. This presents a trade-off analysis in order to obtain an optimum quality of experience by the end user.

VI. Conclusion

This article has exposed the state of the art about DASH standard and adaptive video streaming and also has made a review of the related work on *YouTube*'s traffic analysis. From this review can be concluded that changes in technology, and more precisely in streaming techniques, occurs frequently, deprecating the analysis and reports of a determined service made by researches along the recent years. Due to this constantly changing world, this work tries to fill the gap about the characterization of the recent use of DASH implementation as the preferred streaming technique by *YouTube*.

This article has reported the results obtained from the analysis of the *YouTube*'s DASH implementation download traffic patterns. The relationship between the average download rate of *YouTube*'s video streaming and the quality that *YouTube*'s player operating in *auto* mode finally decides to request from server. In addition, the *YouTube*'s DASH quality adaptation performance with regard to a bandwidth fluctuation test is also reported. From this last analysis, it is concluded that the use of a large video buffer involves a lack of response in showing the quality requested with regard to available bandwidth fluctuations. However, a large video buffer prevent video stalling and rebuffering events and also allows the end user to instantly seek and play any part of the video that is inside the buffer range without the stalling of the rebuffering event.

Future lines of this work involve to expand the bandwidth fluctuation test analyzing the buffer filling and varying some network conditions, such as the latency to the server. Also a comparison analysis between *YouTube*'s video streaming to PCs and mobile devices could be done.

References

[1] Alexa Corporation. *The top 500 sites on the web.* Available: http://www.alexa.com/topsites. Accessed 2015.

[2] A. Rao, A. Legout, Y. Lim, D. Towsley, C. Barakat and W. Dabbous, "Network characteristics of video streaming traffic," *in Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies*, pp. 25, 2011.

[3] P. Ameigeiras, J. J. Ramos-Munoz, J. Navarro-Ortiz and J. M. Lopez-Soler, "Analysis and modelling of YouTube traffic," *Transactions on Emerging Telecommunications Technologies*, vol. 23, pp. 360-377, 2012.

[4] C. Müller, S. Lederer and C. Timmerer, "An evaluation of dynamic adaptive streaming over http in vehicular environments," *in Proceedings of the 4th Workshop on Mobile Video*, pp. 37-42, 2012.

[5] S. Akhshabi, A. C. Begen and C. Dovrolis, "An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http," *in Proceedings of the second annual ACM conference on Multimedia systems*, pp. 157-168, 2011.

[6] S. Akhshabi, S. Narayanaswamy, A. C. Begen and C. Dovrolis, "An experimental evaluation of rate-adaptive video players over HTTP," *Signal Process Image Commun*, vol. 27, pp. 271-287, 4, 2012.

[7] C. Timmerer and C. Müller, "HTTP streaming of MPEG media," *Streaming Day*, 2010.

[8] I. Sodagar, "The mpeg-dash standard for multimedia streaming over the internet," *IEEE Multimedia*, pp. 62-67, 2011.

[9] S. Lederer, C. Müller and C. Timmerer, "Dynamic adaptive streaming over http dataset," *in Proceedings of the 3rd Multimedia Systems Conference*, pp. 89-94, 2012.

[10] D. K. Krishnappa, D. Bhat and M. Zink, "DASHing YouTube: An analysis of using DASH in YouTube video service," *in Local Computer Networks (LCN), 2013 IEEE 38th Conference on*, pp. 407-415, 2013.

[11] M. Zink, K. Suh, Y. Gu and J. Kurose, "Characteristics of YouTube network traffic at a campus network–measurements, models, and implications," *Computer Networks*, vol. 53, pp. 501-514, 2009.

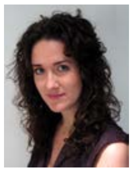
[12] A. Finamore, M. Mellia, M. M. Munafò, R. Torres and S. G. Rao, "Youtube everywhere: Impact of device and infrastructure synergies on user experience," *in Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pp. 345-360, 2011.

[13] S. Alcock and R. Nelson, "Application flow control in YouTube video streams," *ACM SIGCOMM Computer Communication Review*, vol. 41, pp. 24-30, 2011.

[14] YouTube. *YouTube Data API*. Available: https://developers.google.com/youtube/v3/. Accessed 2015.

Javier Añorga was born in Logroño, Spain, in 1987. He received his MSc degree in Telecommunications Engineering from Tecnun (School of Engineering at San Sebastián), University of Navarra, Spain, in 2011.

He joined the CEIT Research Centre in San Sebastián in 2011, and he is currently working on his PhD in CEIT’s Electronics and Communications Department. His professional research activity is in the field of communication protocols, QoS, QoE, Residential Gateways, Embedded Systems and Information Technology. Currently he is also lecturer at the Engineering School of the University of Navarra (Tecnun) in San Sebastián.



Saioa Arrizabalaga was born in Azkoitia in 1979. She received her MS degree in Telecommunication Engineering from the Faculty of Engineering in Bilbao (UPV-EHU) in 2003 and obtained her PhD degree from the University of Navarra in 2009.

She has been involved in several projects regarding remote monitoring using embedded systems, Internet access sharing, Residential Gateways or QoS management in Multi-Dwelling Units. She has published a book and several articles in international journals and conferences. Currently she is also lecturer of the Computer Architecture subject at the Engineering School of the University of Navarra (Tecnun) in San Sebastián.

Beatriz Sedano was born in Eibar in 1978. She received her degree in Telecommunication Technical Engineering in the Engineering School from Santander (University of Cantabria) in 2000 and in Telecommunication Engineering in the same university in 2003. Afterwards she received her PhD. In Engineering (about ultra-wideband microwave oscillators) in 2009 in the Engineering School of the University of Navarra (TECNUN).

She has been involved in research projects related to monitoring using embedded systems: RF domotic devices for intelligent housings and a system based on GSM-R for railway communications. Currently she works in a research project related to a wireless communication system for bioengineering, and she is lecturer of the Microwave subject and its laboratory in the Engineering School of the University of Navarra in San Sebastian. She has published several articles in international journals and conferences.

Maykel Alonso-Arce was born in Vitoria-Gasteiz (Spain) in 1985. He received his Technical Telecommunications Engineering degree, majoring in Communications Systems, in 2007 and his Telecommunications Engineering degree in 2009 from the Faculty of Engineering Mondragon University (MGEP), Spain.

During his studies he worked for Fagor Electronics porting uC-Linux to a prototype board as an intern of the Faculty of Engineering - Mondragon University (MGEP). He also studied one semester at the Aalborg University (AAU), Denmark, through an ERASMUS scholarship. In 2009 he joined the Electronics & Communication Department at CEIT, and started his PhD studies, through the Iñaki Goenaga (FCT-IG) Technology Centers Foundation grant. At present, his research interest field is focused on the “Design of embedded communications systems inside human bodies”.

Jaizki Mendizabal is a lecturer at Tecnun, the Technological Campus of University of Navarra, San Sebastián, Spain, and a researcher in the Electronics and Communications Department at CEIT. He was born in Zarautz and received his MSc and PhD degrees in Electrical Engineering from Tecnun (University of Navarra, San Sebastian, Spain) in 2000 and 2006 respectively.

He joined Fraunhofer Institut fr Integrierte Schaltungen, Erlangen (Germany) from 2000 to 2002 and SANYO Electric Ltd, in Gifu (Japan) from 2005 to 2006 as RF-IC designer. He obtained his PhD in the field of monolithic RF design for GNSS systems. He currently works in CEIT where his research interests include RFICs and analogue safety systems for the railway industry. He has participated in more than 8 research projects, has directed 2 doctoral theses, is author or co-author of some 21 scientific and technical publications in national and international journals and conferences and is the author of the book GPS and Galileo Dual RF Front-end receiver and Design, Fabrication, Test published by McGraw-Hill.