



D D 3.3 – Gateway Trials

Arturo Azcorra, Jaime García, Carmen Guerrero, Mario Ibañez and Francisco Valera
Universidad Carlos III de Madrid. Avda. de la Universidad 30. 28911 Leganés, Madrid. Spain
{azcorra;jgr;guerrero;fvalera}@it.uc3m.es, mario.ibanez@uc3m.es

Harold Balemans, Willem van Willigenburg
P.O. Box 1168, 1200 BD Hilversum, Larenseweg 50, 1221 CN Hilversum,
The Netherlands
{hchb;willigenburg}@lucent.com

Vitor Pinto and Vitor Ribeiro
Portugal Telecom Inovação. Rua Eng. Ferreira Pinto Basto. 3810-106 Aveiro. Portugal
{it-v-pinto;vribeiro}@ptinovacao.pt

Pierre Jaffré
France Telecom. 2, avenue Pierre Marzin. 22307 Lannion Cedex - France
pierre.jaffre@francetelecom.com

Identifier:	Deliverable D 3.3
Class:	Report
Version:	8
Version Date:	23/12/2005
Distribution:	Public
Responsible Partner:	UC3M
Filename:	WPD3_0003_V08_D.D33_Gateway_Trial_PU.doc

DOCUMENT INFORMATION

<i>Project ref. No.</i>	IST-6thFP-507295
<i>Project acronym</i>	MUSE
<i>Project full title</i>	Multi-Service Access Everywhere
<i>Security (distribution level)</i>	Public
<i>Contractual delivery date</i>	30/11/2005
<i>Actual delivery date</i>	23/12/2005
<i>Deliverable number</i>	D D3.3
<i>Deliverable name</i>	Gateway trials
<i>Type</i>	Report
<i>Status & version</i>	V8
<i>Number of pages</i>	65
<i>WP / TF contributing</i>	WP D.3
<i>WP / TF responsible</i>	Arturo Azcorra
<i>Main contributors</i>	UC3, LUNL, PTI, FT
<i>Editor(s)</i>	Francisco Valera
<i>EU Project Officer</i>	Pertti Jauhiainen
<i>Keywords</i>	Residential gateway, trials, test-beds, configuration, installation, operator manual
<i>Abstract (for dissemination)</i>	<p>The prototype developed in WPD3 has been completed and this document details the tests and evaluations that have been performed in order to validate the different functionalities that it is presenting within the framework of a fibre to the home scenario. A triple play (video streaming, VoIP and high speed Internet access) testbed has been configured and it will allow trialling a gigabit access to the residential environment with the added value of MUSE QoS model. In addition, this document also details the different tests that have been performed to validate the authentication mechanism proposed for the residential gateway as well as an innovative integrated management procedure based on DSL Forum TR-069 specification guided by OSGi bundles that are capable of being automatically updated when it is required.</p>

DOCUMENT HISTORY

Version	Date	Comments and actions	Status
V0	25/09/2005	TOC	<i>Draft</i>
V1	14/10/2005	Modified TOC according to audioconferences: 1 PROTOTYPE DESCRIPTION 1.1 Introduction (UC3) 1.2 Functionality (FT) 1.3 Prototype architecture (LUNL+UC3) 1.3.1 RGW general architecture (LUNL) 1.3.2 Prototype architecture (UC3) 1.4 Prototype development (LUNL+PTI+UC3) 1.4.1 Introduction (UC3) 1.4.2 RGW prototype development (PTI+UC3) 1.4.3 RGW manual remote management (UC3) 1.4.4 RGW automatic remote management (LUNL+UC3) 1.5 Testbed developments (LUNL+PTI+UC3) 2 RGW TRIALS 2.1 Introduction (UC3) 2.2 Characteristics to be trialled (LUNL+PTI+UC3) 2.2.1 Introduction (UC3) 2.2.2 Network trial (PTI+UC3) 2.2.3 Automatic management trial (LUNL+UC3) 2.3 Testbeds and trials (LUNL+PTI+UC3) 2.3.1 Network trial testbed (PTI+UC3) 2.3.2 Automatic management trial testbed (LUNL+UC3) 3 RGW TRIALS RESULTS 3.1 Introduction (UC3) 3.2 Results(LUNL+PTI+UC3) 3.2.1 Network trial testbed (PTI+UC3) 3.2.2 Automatic management trial testbed (LUNL+UC3) 3.3 Conclusion (UC3)	<i>Draft</i>
V2	2/11/2005	Fist version including draft contributions to serve as input for the rest of the partners	<i>Draft</i>
V3	14/11/2005	Contributions from LUNL, PTI and UC3	<i>Draft</i>
V4	21/11/2005	Contributions from FT and UC3	<i>Draft</i>
V5	1/12/2005	Contributions from UC3	<i>Draft</i>
V6	1/12/2005	Contributions from LUNL and UC3	<i>Draft</i>
V7	3/12/2005	Last edition changes	<i>Draft</i>
V8	16/12/2005	Final version including quality review inputs	<i>Final</i>

TABLE OF CONTENTS

DOCUMENT INFORMATION	2
DOCUMENT HISTORY	3
TABLE OF CONTENTS	4
LIST OF FIGURES	5
LIST OF TABLES	6
ABBREVIATIONS	7
ABBREVIATIONS	7
REFERENCES	9
EXECUTIVE SUMMARY	11
INTRODUCTION	12
1 PROTOTYPE DESCRIPTION	13
1.1 Introduction	13
1.2 Functionality	13
1.3 Prototype architecture	15
1.3.1 <i>RGW general architecture</i>	<i>15</i>
1.3.2 <i>Prototype architecture</i>	<i>17</i>
2 RGW TESTS AND TRIALS	22
2.1 Introduction	22
2.2 Characteristics to be evaluated	22
2.2.1 <i>Introduction</i>	<i>22</i>
2.2.2 <i>Network trial</i>	<i>22</i>
2.2.3 <i>Automatic management trial</i>	<i>26</i>
2.3 Testbeds and trials	26
2.3.1 <i>Network trial testbed</i>	<i>26</i>
2.3.2 <i>Automatic management evaluation testbed</i>	<i>34</i>
3 RGW TRIALS RESULTS	40
3.1 Introduction	40
3.2 Results	40
3.2.1 <i>Network trial testbed</i>	<i>40</i>
3.2.2 <i>Automatic management evaluation testbed</i>	<i>45</i>
3.3 Conclusions	55
ANNEXE A	57
Operator manual	57
<i>Web configuration interface</i>	<i>57</i>

LIST OF FIGURES

Figure 1. RGW functional blocks.....	15
Figure 2. NT2 functional blocks.....	16
Figure 3. Prototype interfaces	16
Figure 4. Functional blocks inside the RGW	18
Figure 5. Process distribution inside the RGW.....	21
Figure 6. Architecture emulated in the trials.....	26
Figure 7. Network trial testbed	28
Figure 8. Complete scenario	29
Figure 9. VoIP scenario.....	32
Figure 10. Video streaming scenario.....	33
Figure 11. Data bulk scenario	34
Figure 12. Automatic management testbed	35
Figure 13. NCB – ACS interface testbed.....	35
Figure 14. NCB-CNC interface and CCF update testbed.....	36
Figure 15. PLB-NCB PLB-CNC communication testbed.....	37
Figure 16. Bundle management testbed	38
Figure 17. Failed authentication attempt and re-authentication attempt by supplicant software running on the RGW	41
Figure 18. Successful authentication	41
Figure 19. Main configuration frame.....	57
Figure 20. Upstream configuration web page	58
Figure 21. Downstream configuration web page.....	60
Figure 22. List active rules interface.....	62
Figure 23. Residential Gateway configuration interface.....	64
Figure 24. Statistics interface	65
Figure 25. Traffic generation interface	66

LIST OF TABLES

Table 1. Interface protocols.....	21
Table 2. Hardware and software involved in the trials.....	30
Table 3. Items to be tested in NCB – CNC interface.....	39
Table 4. Items to be tested in bundle management.....	39
Table 5. Authentication trials.....	40
Table 6. Two Iperf flows.....	42
Table 7. Voice and Iperf marked as Best Effort.....	43
Table 8. Voice and Iperf marked as Real Time.....	43
Table 9. Voice and Iperf marked as Low Latency.....	43
Table 10. Voice, video (low quality) and Iperf marked as Best Effort.....	44
Table 11. Voice, video (low quality) and Iperf marked as Real Time.....	44
Table 12. Voice, video (low quality) and Iperf marked as Low Latency.....	44
Table 13. Voice, video (high quality) and Iperf marked as Best Effort.....	44
Table 14. Voice, video (high quality) and Iperf marked as Real Time.....	45
Table 15. Voice, video (high quality) and Iperf as Low Latency.....	45
Table 16. Results of the NCB – CNC interface test.....	54
Table 17. Items to be tested in bundle management.....	54

ABBREVIATIONS

ACS	<i>AutoConfiguration Server</i>
AF	<i>Application Function</i>
API	<i>Application Program Interface</i>
ANM	<i>Access Network Manager</i>
BMP	<i>Bit Map</i>
B2BUA	<i>Back-to-Back User Agent</i>
CBCA	<i>Click! Boot Configuration Agent</i>
CCF	<i>Click! Configuration File</i>
CHAP	<i>Challenge Handshake Authentication Protocol</i>
CNC	<i>Click! Network Controller</i>
CP	<i>Customer Premises</i>
CPE	<i>Customer Premises Equipment</i>
CSCF	<i>Call/Session Control Function</i>
CSD	<i>Click! Signalling Dispatcher</i>
CSP	<i>Content Service Provider</i>
DHCP	<i>Dynamic Host Configuration Protocol</i>
DNS	<i>Domain Name Server</i>
DSL	<i>Digital Subscriber Line</i>
DTD	<i>Data Type Definition</i>
DVD	<i>Digital Versatile Disc</i>
EAP	<i>Extensible Authentication Protocol</i>
EAPOL	<i>EAP Over Local Area Network</i>
EAP-AKA	<i>EAP- Authentication and Key Agreement</i>
EAP-GTC	<i>EAP-Generic Token Card</i>
EAP-MD5	<i>EAP- Message-Digest algorithm 5</i>
EAP-MSCHAPv2	<i>EAP- MicroSoft Challenge Handshake Authentication Protocol Version 2</i>
EAP-OTP	<i>EAP-One Time Password</i>
EAP-SIM	<i>EAP- Subscriber Identity Module</i>
EAP-TLS	<i>EAP-Transport Layer Security</i>
EAP-TTLS	<i>EAP-Tunnelled Transport Layer Security</i>
FTTH	<i>Fibre to the Home</i>
Ge	<i>Gigabit Ethernet</i>
GIF	<i>Graphics Interchange Format</i>
HDD	<i>Hard Disk Drive</i>
HN	<i>Home Network</i>
HomePNA	<i>Home Phone line Networking Alliance</i>
HTML	<i>HyperText Markup Language</i>
HTTP	<i>HyperText Transfer Protocol</i>
IGMP	<i>Internet Group Management Protocol</i>
IP	<i>Internet Protocol</i>
JPEG	<i>Joint Photographic Experts Group</i>
LAN	<i>Local Area Network</i>
LEAP	<i>Lightweight Extensible Authentication Protocol</i>

MGCP	<i>Media Gateway Control Protocol</i>
MP3	<i>MPEG-1/2 Audio Layer 3</i>
MPEG	<i>Moving Picture Experts Group</i>
NAT	<i>Network Address Translator</i>
NCB	<i>Network Controller Bundle</i>
NCS	<i>Network Controller Servlets</i>
NIC	<i>Network Interface Card</i>
OS	<i>Operating System</i>
OSGi	<i>Open Services Gateway Initiative</i>
PAP	<i>Password Authentication Protocol</i>
PAT	<i>Port Address Translation</i>
PC	<i>Personal Computer</i>
PCI	<i>Peripheral Component Interconnect</i>
PCM	<i>Pulse-code modulation</i>
PEAP	<i>Protected Extensible Authentication Protocol</i>
PERL	<i>Practical Extraction and Report Language</i>
PIM-SSM	<i>Protocol Independent Multicast-Source Specific Multicast</i>
PQ	<i>Priority Queuing</i>
PVR	<i>Personal Video Recorder</i>
QoS	<i>Quality of Service</i>
RGW	<i>Residential Gateway</i>
RTP	<i>Real-time Transport Protocol</i>
RTSP	<i>Real Time Streaming Protocol</i>
SIP	<i>Session Initiation Protocol</i>
SNMP	<i>Simple Network Management Protocol</i>
SP	<i>Signalling Process</i>
SPI	<i>Stateful Packet Inspection</i>
STB	<i>Set Top Box</i>
TFTP	<i>Trivial File Transfer Protocol</i>
UPnP	<i>Universal Plug and Play</i>
USB	<i>Universal Serial Bus</i>
VCR	<i>Video Cassette Recorder</i>
VLAN	<i>Virtual Local Area Network</i>
VoD	<i>Video on Demand</i>
VoIP	<i>Voice over IP</i>
VPN	<i>Virtual Private Network</i>
WAN	<i>Wide Area Network</i>
WLAN	<i>Wireless Local Area Network</i>
XML	<i>eXtensible Markup Language</i>

REFERENCES

- [1] MUSE Deliverable DD3.2. "RGW Solutions". October 2005.
- [2] MUSE Deliverable DA2.4. Network architecture and functional specifications for the multi-service access and edge, December 2005.
- [3] MUSE Deliverable DD4.1 "Design of lab trials and services", June 2005.
- [4] MUSE deliverable TF3.2. Detailed description of residential gateway and advanced features. December 2005.
- [5] MUSE deliverable DTF3.1. Subscriber Interface. December 2004.
- [6] MUSE deliverable DTF1.4. IPv4/IPv6 forwarding access and edge platform and opportunities for IPv6 in access. July 2005.
- [7] DSL Forum, "CPE WAN Management Protocol", TR-069, May 2004. URL: http://www.dslforum.org/aboutdsl/Technical_Reports/TR-069.pdf
- [8] Oscar: <http://oscar.objectweb.org>
- [9] OBR (Oscar Bundle Repository) : <http://oscar-osgi.sourceforge.net>
- [10] MUSE Deliverable DA2.2. Network architecture and functional specifications for the multi-service access and edge, January 2005.
- [11] *QoS Management in Fixed Broadband Residential Gateways*. C. Guerrero, J. García Reinoso, F. Valera, A. Azcorra. 8th International Conference on Management of Multimedia Networks and Services (MMNS 2005). Barcelona, Spain. October 2005
- [12] *Designing a Broadband Residential Gateway using Click! Modular Router*. H. Gascón, D. Díez, J. García, F. Valera, C. Guerrero, A. Azcorra. EUNICE 2005. Colmenarejo, Spain. July 2005
- [13] Open1x: Open source implementation of 802.1X. URL: <http://open1x.sourceforge.net/>
- [14] PERL IO::Interface library description. URL: <http://search.cpan.org/~lds/IO-Interface-0.98/Interface.pm>
- [15] Hostapd: IEEE 802.11 AP, IEEE 802.1X/WPA/WPA2/EAP/RADIUS Authenticator URL: <http://hostap.epitest.fi/hostapd/>
- [16] freeRADIUS: FreeRADIUS open source RADIUS server URL: <http://www.freeradius.org/>
- [17] SER: SIP Express Router. URL: <http://www.iptel.org/ser/>

[18] VLC: VideoLAN Client. URL: <http://www.videolan.org/vlc/>

[19] X-Lite. SIP Software Phone. URL: <http://www.videolan.org/vlc/>

[20] Iperf: The TCP/UDP Bandwidth Measurement Tool. URL:
<http://dast.nlanr.net/Projects/Iperf/>

EXECUTIVE SUMMARY

This document details the different test and trials that have been performed on the residential gateway prototype developed by WPD3 within the framework of the fibre to the home scenario related to SPD and following the architectural guidelines proposed by both SPA and TF3 documents.

The Residential Gateway Prototype has been successfully tested in an scenario where it is capable of authenticating itself towards the access network using 802.1X protocol and is also capable of autoconfigure itself with the proper parameters so as to be able to receive a network access service in a MUSE compliant access network as well as providing network access to the terminals located in the home network.

In order to perform the service trials, a triple play (video streaming, VoIP and high speed Internet access) testbed has been built and allowing the demonstration of different MUSE specific concepts applied over a broadband access line (gigabit). Between these different concepts it is worthwhile to highlight the QoS mechanism defined in MUSE (including 802.1pq frame tagging, CoS queue performance, etc.). The triple play trial shows the performance of the residential gateway when all the different applications and services are set running and different residential gateways are connected to the access network. Different tests are performed using different flow priorities (classes of services) and different audio and video qualities so as to be able to evaluate the QoS schema proposed in MUSE. This triple play testbed has also been used to hold a multicast trial where terminals in the home network were enabled to receive multicast traffic from a videosever.

This document is also describing an innovative approach towards the integrated management of residential gateways including the OSGi platform within the gateway. This platform is responsible of the TR-069 parameter exchange between the residential gateway and the autoconfiguration server but will also allow the automatic update of the different OSGi bundles that are installed in the residential gateway (in order to include some more parameters for example or in order to include new OSGi services such as home sensors management). This new management and configuration approach is compared with a traditional web based configuration procedure that has also been implemented and in fact has been the method used during the network trials in order to configure the different flow rules.

Since the performance of the different tests implies the connection of the residential gateway to an access network together with the deployment of the different services in order to be able to emulate the provider domain this document is also describing the different services and network entities emulators that have also been installed together with the main development of the residential gateway platform. These developments include an access/edge node, different software servers and clients (web, radius, SIP, video server), and different specific hardware that is also part of the trials like wireless SIP phones or a video server.

INTRODUCTION

The focus of this document is set on the prototype that has been implemented by WPD3 within the broadband FTTH scenario specified by SPD. In previous DD3.2 document [1] the functionality and architecture of the prototype was detailed and many technological choices were adopted according to MUSE general framework defined by SPA documents [2].

The prototype trials have been accommodated to several guidelines provided by [3] so as to be able to emphasize different characteristics that were important to evaluate since they were more related to MUSE concepts than other functionalities that although present in the prototype were not such differentiating. All these diverse test and trials are detailed in this document including the specific configuration of the testbeds, the different emulators that have been required in order to complete the whole network environment installed to perform the trials as well as the different set of tests that have been done.

The document has been structured as follows.

The first section includes a description of the functionalities of the prototype as well as a summary of the prototype architecture introducing the different modules that will be later used in the trials.

The second section defines the test and trials performed on the residential gateway. It first enumerates the different specific characteristics that are wanted to be demonstrated since they are the ones that make the trials to have sense. As the rest of the subsections in the document the characteristics are related first to the network trial and then to the automatic management tests. These two scenarios are then detailed in subsequent subsections both of them with the same structure: first the testbed is introduced and its configuration explained and then the trials are introduced and later explained in detail.

The last section includes all the results of the trials and the tests as well as a final conclusion.

The document is completed with an annex that includes the operator manual that allows any external user to configure and manage the residential gateway using the web configuration interface.

1 PROTOTYPE DESCRIPTION

1.1 Introduction

This section shows a summary of the functionalities available in the prototype also including the architecture of the prototype so that it will be later easier to identify the different modules involved in the different trials, and relate them to the general residential gateway platform.

1.2 Functionality

The prototype developed in WPD3 has been designed with functions as close as possible to the MUSE concepts pushed forward through the documents produced by the Architecture SPA group and the Residential Gateway TF3 group. This work has been achieved in the context of the FTTH scenario considered in SPD subproject.

A first major MUSE concept concerns the management of **end-to-end QoS** focused in MUSE on the IP Multimedia System (IMS) approach, described in WPD3_DD3.2 [1]: the prototype QoS features have been developed under the consideration that the RGW does not need to provide an IMS-based interface in the case where the end user terminals generate and receive SIP signalling: the RGW has only to fill the function of transparent relay of uplink/downlink SIP signalling. Nevertheless the prototype development has also been designed in the scenario where non-IMS terminals may be present at home and where the RGW therefore may easily include an IMS signalling mapping proxy functionality.

Some functional blocks related to QoS like traffic shaping, queuing and scheduling have not yet been completely covered in MUSE (refer for instance to DTF3.1 [5] for some first references) and in general there remain some open issues about MUSE QoS architecture, so that the RGW prototype designed in WPD3 remains open to future architectures about QoS in the access network even although as it will be seen it is supporting some of these QoS functionalities not yet specified in MUSE.

Another major MUSE concept concerns **authentication**, whose principles are described in WPA2_DA2.4 [2] and synthesized in WPD3_DD3.2 [1]. The related recommendations pushed forward, like the use of 802.1X protocol, with an IEEE 802.1X supplicant in the RGW and a IEEE 802.1X authenticator and RADIUS client in the access node, have been implemented in the prototype and trialled. The provided functionality is a basic authentication of the RGW (of the equipment). The major advantage of that technique is to provide a location-independent authentication process to DHCP-based services, even if it does not allow per-user or per-service authentication under full control of the access node.

A third point extensively implemented on the prototype concerns the **remote management** and the related **autoconfiguration process**. The managing communication systems and protocols are described in document DTF3.2 [4] where the standard TR-069 is pushed forward for the remote management, following that way the recommendation of the DSL Forum. That standard has been implemented on the prototype for the exchange between the ACS and the RGW, in the specific case of an OSGi framework implemented in the RGW. In parallel, a RGW manual remote management system through a web interface has been implemented in the prototype.

Recommendations related to **multicast**, developed in [6] have not been the subject of an implementation in the prototype since they are more oriented towards a network node:

- The use of IGMP v3 with PIM-SSM (Protocol Independent Multicast-Source Specific Multicast), of RTSP protocols and of Service Discovery mechanisms for the DVB-IP based video broadcast.
- The definition of application aware edge functionalities for the hybrid multicast solution for the multiparty peer-to-peer applications (conferencing or gaming application).

However, the RGW is capable of supporting multicast delivery to the end terminals so it is possible for the users to join multicast groups and receive this kind of traffic.

Regarding the various **IP-address allocation** schemes and the features related to **IPv6** addressing, presented in DTF3.2 [4] and in WPA2_DA2.2 [10], the RGW is facing the recommendations done related to residential environments where MUSE has specified an IPv4 scenario with no layer 2 direct services support but where IP services are being supported with QoS and where an IP-address allocation based on NAPT has been implemented.

1.3 Prototype architecture

1.3.1 RGW general architecture

The RGW architecture is divided in a number of functional blocks with and interfaces. In this section only an overview is given. For a complete and concise architecture refer to [1].

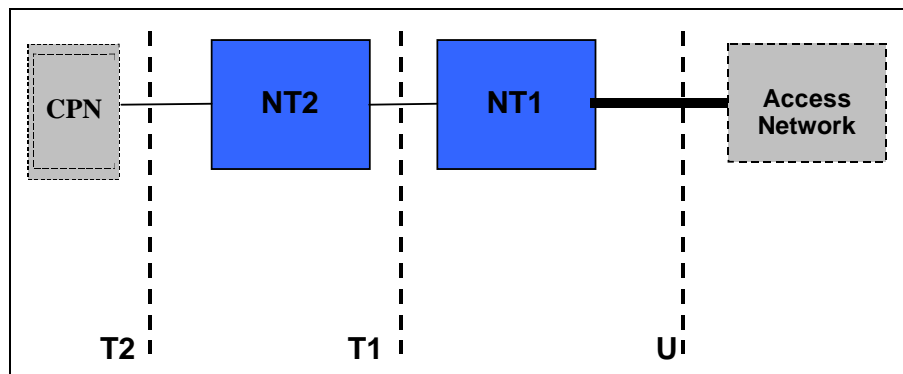


Figure 1. RGW functional blocks

The Residential Gateway is functionally split into a NT1 and a NT2 function.

- **NT1** makes the NT2 access network technology independent.
- **NT2** contains access control functions between the home network and the public network; and it provides connectivity basically on L2 and/or L3 layer including QoS functions, optionally L3 routing. The U interface between the Access Network and the Residential Gateway.
- **CPN** is a LAN.

The NT2 of the prototype is built on a PC platform using:

1. Click! router: for L2/L3 functionality
2. OSGI for service management
3. HTTP server for web based control
4. TR-069 as a protocol for remote management by the Auto Configuration Server

The NT1 is built using a Network Interface Card. The NT1 and NT2 thus are combined in one device.

1.3.1.1 Application of architecture in prototype

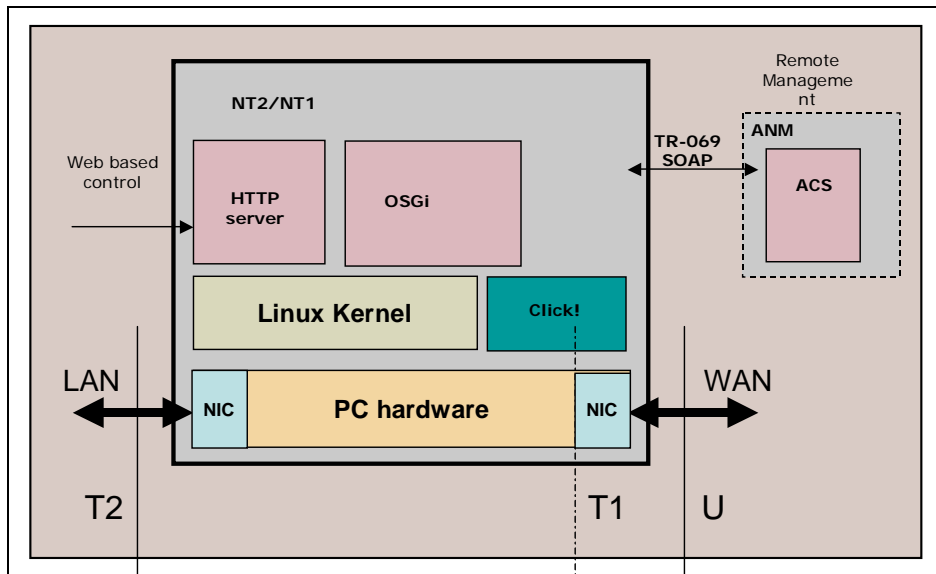


Figure 2. NT2 functional blocks

1.3.1.2 Prototype interfaces

If we would consider the gateway prototype as a black box it would have four interfaces supported by two physical connections (except for the power).

1. Web based control interface (WAN)
2. ACS control interface (on the WAN)
3. One CPN interface
4. One Access Network interface

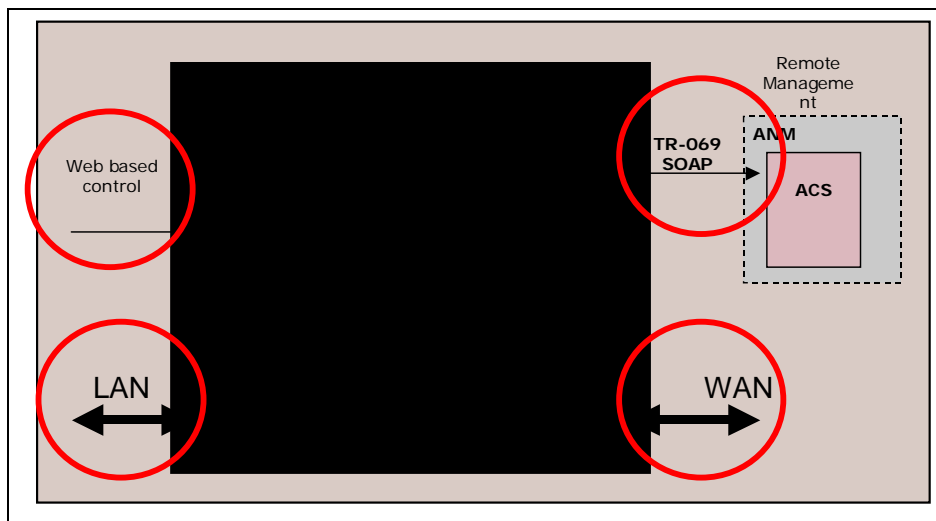


Figure 3. Prototype interfaces

1.3.1.2.1 *Web based Control interface*

This interface allows the user to configure and manage the RGW. For this the user has to access a local webpage on <URL>. The interface is not protected for this prototype and is supported on the WAN side.

1.3.1.2.2 *ACS control interface*

The ACS control interface allows the Auto Configuration Server to manage and control the RGW via the TR-069 protocol (a subset).

It is supported on the WAN side only. The RGW will set up a connection with the ACS and requires IP connectivity in order to do this.

For this prototype the ACS URL is predefined in the RGW as is also the username and password needed to access to ACS .

1.3.1.2.3 *WAN*

This interface is 100BaseT or 1GB Ethernet (optical) .Via this interface (and port 67, default) the RGW issues a DHCP request to obtain the IP address for the WAN.

When this IP address is obtained port 8080 is available for HTTP connections for the Web based control interface (not secured). ICMP echo messages are supported for testing purposes (ping).

The WAN interface allows VLAN 802.1Q/p tags where the p-bits will support the CoS classes defined by MUSE.

1.3.1.2.4 *LAN*

This interface is 100BaseT. It provides a local DHCP server for the Home Network.

The address of this interface and the corresponding DHCP address pool is pre-defined for the prototype.

IP packets (upstream or downstream) may contain QoS marking via the TOS bits. This marking will be translated to/from p-bits on WAN.

The RGW will provide NATP between the WAN and LAN side.

1.3.2 Prototype architecture

This subsection describes the architecture of the prototype already implemented from the point of view of the development environment and this description will be classified using a bottom-up view starting at the data level and then defining the configuration process.

1.3.2.1 Functional blocks at the bottom level

Figure 4 represents the complete picture at the bottom level where all functional blocks and their relationships are depicted. Incoming and outgoing traffic flows are represented and the two separate paths show that these two flows never use the same resources at Click! level. Dotted arrows represent unknown outgoing traffic. Click! level sends these packets to the CSD to treat them and then it sends the packets to the corresponding Signalling Process (SP) to handle it. Finally, the SP returns the packets to Click! level. Dashed arrows are frame copies that Click! sends to the CSD or the IMS due to special characteristics (signalling frames, for example). There is also an OSGi bundle-CSD communication where the Manager can configure Click! on behalf of OSGi and vice versa.

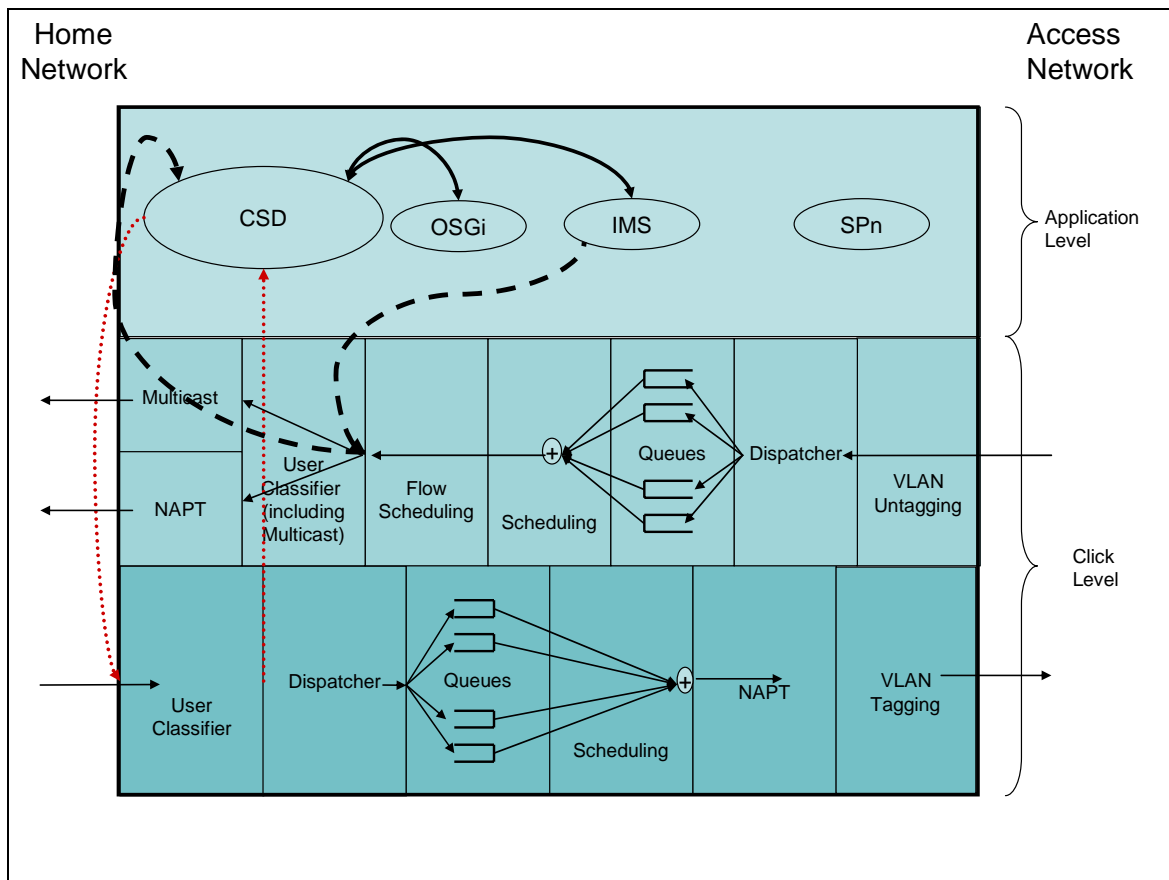


Figure 4. Functional blocks inside the RGW

These are the functional blocks for this level:

- **User Classifier:** the function of this block is to recognize flows depending on the user (administrator) preferences. The user can add, reorder or remove flow definitions and this change will modify the functionality of this block. In the downstream direction, the user can select that a particular flow could be replicated to all in-home interfaces selecting the multicast option.
- **Dispatcher:** based on the p-bits field, this module introduces a packet in one of the four possible queues.

- **Queues:** in Click, the implementation of these queues is based on the invocation of four different queue elements. Each queue represents a different CoS. There are several ways to accomplish the requirements imposed by a specific CoS. For example, a fix size queue can be used to avoid queue delays. This method is explained in [5].
- **Scheduling:** working with two or more queues implies the use of some algorithm to extract a packet from one queue at each time. It is even more complex to elect the right one when priority queues exist. There are many scheduling algorithms to treat this problem: *Priority Queuing, Weighted Fair Queuing, Class-based Weighted Fair Queuing, etc.* There is a Click! element called PrioSched that implements a Priority-like Queuing. New scheduling algorithms implementation is for further study.
- **Policing:** the aim of this functional block is to limit the flows rate for every CoS. When an excess rate is detected, this block can either discard packets or change their CoS tag.
- **VLAN untagging:** removes the 802.1Q/p tags from the packets.
- **Flow scheduling:** detects “important” packets as signal packets and send copies to the CSD. It is useful to detect new flows and their characteristics (CoS for example).
- **NAPT:** for the upstream traffic it detects new sessions and then creates new entries in the NAPT table to change the original source port with the corresponding identification. It also has to detect the end of the session to erase the entry from the NAPT table. For the downstream direction, this box just has to change the destination port and IP address with the matching entry from the NAPT table. Upstream and downstream directions are not separate boxes because they must share the NAPT table.
- **VLAN tagging:** depending on the user/administrator preferences, the outgoing packets will be marked with 802.1Q/p tags and bits to add a priority to the frames. This information (the corresponding p-bit assignment) is pre-configured in the first stage and configured by the network in the final prototype. When a flow is not configured, this block forwards the packet to the CSD. The CSD can then discard the packet or reconfigure the VLAN tagging functional block (to recognize this new flow) and inject the packet again.
- **Multicast:** this block copies all incoming frames to all in-home interfaces.
- **CSD:** this software will be developed in Java to allow an easy and quick portability to other platforms. The CSD will configure and reconfigure the Click! level when OSGi or signalling packets arrives. It is also possible to redirect this kind of packets to the corresponding Signalling Process (SP).
- **IMS:** MUSE plans to adopt SIP, which is used as the prominent signalling protocol in IMS, as its QoS signalling standard. In IMS, a central point has all information related with available resources in the network and, when a new request arrives to the system, it can detect the possibility or not to accomplish that requirement. Because the RGW is the access and home network interconnection point, it must couple both worlds and allow an end-to-end QoS. To accomplish this requirement, the RGW must be configured as any network node. There are no gateway specifications in 3GPP documents so there are no interface definition for the RGW. To develop an RGW prototype with IMS capabilities, a special IMS functional block will be implemented at the application level. When the Click! level detects SIP messages, it will copy this frame to the IMS application. The IMS application will identify the flow and CoS characteristics. With this information, the CSD will reconfigure the Click! level on behalf the IMS block.

1.3.2.2 Functional blocks at the configuration level

Besides the functional blocks dedicated to handle data frames, there are more processes dedicated to start, configure and reconfigure the RGW. Figure 5 depicts all applications regarding RGW configuration both inside the RGW and external elements. To allow a better comprehension of the whole configuration stage, all applications will be described:

- **ACS (AutoConfiguration Server)**. The ACS is located on the ANP side. The ACS has:
 - A DHCP server to configure in the RGW:
 - WAN interface IP address.
 - WAN interface IP submask.
 - DNS IP address.
 - A web browser to configure the RGWs.
- **CBCA (Click! Boot Configuration Agent)**. This is the first application launched in the RGW used to start the autoconfiguration process and the rest of the applications. The CBCA writes in the CCF (Click! Configuration File) the configured parameters. This application exits at the end of the configuration process.
- **CCF (Click! Configuration File)** is not an application but the Click! file uses it to configure the functional blocks explained in 1.3.2.1.
- **CNC (Click! Network Controller)**. After the bootup process, this application is the only one allowed to reconfigure the Click! part. There is a standard API designed to allow different ways to access this functional block: using a NCS, NCB or whatever other block used to configure the RGW remotely. The CNC uses an auxiliary file with an XML format to generate the Click! file.
- **CSD (Click! Signalling Dispatcher)**. When a SP (Signal Process) starts, it must register first with the CSD application. This registration is used by the CSD to create new rules in the Click! part to generate copies that will be sent to the CSD. The CSD will redirect those frames to the corresponding SP.
- **NCB (Network Controller Bundle)**. This OSGi application (bundle) communicates with the ACS to receive and send TR-069 configuration messages. Incoming messages are parsed to construct the proper XML file and then sent to the CNC application. For the outgoing packets the opposite process must be performed.
- **NCS (Network Controller Servlet)**. It is similar than the NCB application but not implemented as an OSGi bundle. The NCS can write in the CCF directly, but in next prototype version it must use an interface with the CNC instead.
- **SPx (Signalling Process)**. These processes can handle different kind of signalling frames, for example: SIP, H.323, etc.

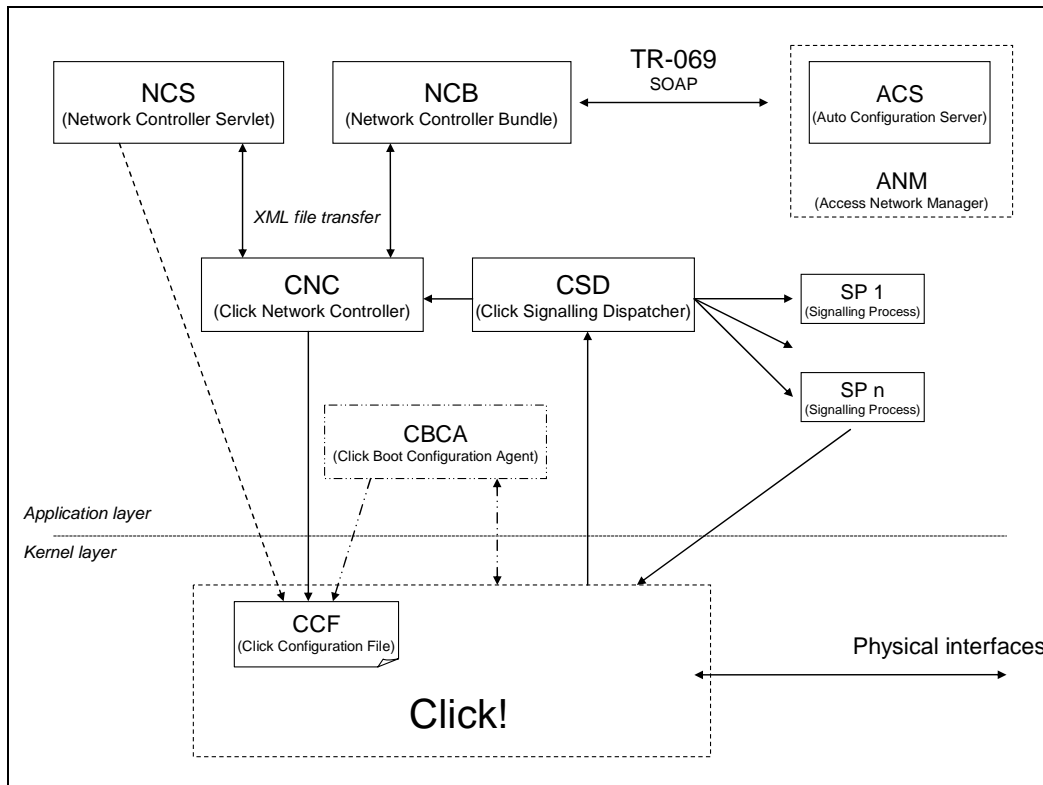


Figure 5. Process distribution inside the RGW

Interface Pairs	Interface
ACS-NCB	TR-069/SOAP
NCB-CNC	XML file
NCS-CNC	XML file
CNC-Click	Click! file
CSD-CNC	Preconfigured by now (no real interace)
Click-CSD	UDP packets (using a known port)
SPx-Click	UDP packets (using a fake device)

Table 1. Interface protocols

2 RGW TESTS AND TRIALS

2.1 Introduction

This section details the different testbeds together with the different tests and trials that have been performed on the RGW.

In order to clearly define the purpose of the tests the first subsection describes the specific characteristics that are going to be trialled and demonstrated (MUSE related characteristics) and the methodology that will be used to test them.

Next section is divided into two subsections corresponding to the two main testbeds that will be evaluated: the network trial testbed and the automatic management testbed. And within each subsection the structure is the same. The first part describes the testbeds that will serve as framework for the trials. The second one details the configuration of the corresponding testbeds. The third part describes the different trials that will be performed on the testbeds. And finally the fourth one details how to perform the different trials.

2.2 Characteristics to be evaluated

2.2.1 Introduction

The most important characteristic of this prototype developed within the framework of WPD3 is that it is a MUSE compliant RGW since it presents different properties that have been adopted or proposed by MUSE architectural design group or MUSE RGW specific taskforce.

This section highlights all the differentiating characteristics available in the RGW prototype with a special focus on those that enable this RGW as MUSE compatible, but also those that are introduced by WPD3 as specific innovation.

The purpose of the trials, apart from validating the correct operation of the RGW, is to verify the proper implementation of MUSE particular characteristic according to the defined functionality.

The description of these characteristics is done in this section following the same order as the description of the different tests done in section 2.3: network trial related characteristics (including both operation relation characteristics as well as manual management characteristics) and automatic management related characteristics.

2.2.2 Network trial

2.2.2.1 RGW network operation

2.2.2.1.1 Authentication characteristics

In MUSE TF1-WPA2 architecture group, namely on [2], several requisites concerning authentication were listed:

- What to authenticate?
 - Basic authentication: only the RGW is authenticated, devices behind it are not authenticated. This type of authentication is considered mandatory.

- Per-service authentication: might have some interest supposing an access line has a profile for gaining access to some services from a plurality of services, it could be interesting to have a more selective authentication of this access line, in terms of the service that is being asked. This type of authentication for now it is not considered mandatory
- Per-user authentication: in situations that involve nomadism, this type of authentication might be of interest, but not considered mandatory for now.

In current WPD3 prototype only basic authentication of the RGW is performed.

- Which authentication method?
 - IEEE802.1X with EAP methods such as: EAP-SIM, EAP-AKA, EAP-TLS, EAP-TTLS, EAP-MD5, etc.
 - PANA with the same EAP methods as IEEE802.1X
 - DHCP option 90

WPD3 RGW prototype authentication is performed through IEEE802.1X with EAP-TLS (in the RGW an IEEE802.1X Supplicant that supports also other EAP methods is implemented).

- Who will authenticate?

As stated on [2], both NAP and NSP can perform the authentication of the RGW. Since this issue is more or less transparent from the point of view of the RGW (from the RGW perspective what is important is that authentication process must be launched on the WAN interface) on WPD3 trials no distinction between the NAP and NSP networks was made. Authentication frames from the RGW are simply sent to the access multiplexer which in turn acts as an IEEE802.1X Authenticator/RADIUS Client.

2.2.2.1.2 *Operation characteristics*

A triple play scenario with applications like (Voice over IP, video streaming and bulk data transfer) is defined to test the main functional blocks developed in the RGW.

The Queue and Scheduling Functional Blocks inside the RGW device are the principal blocks to be tested to assure a complete end-to-end QoS. There are two different blocks one for each direction (downstream and upstream) and both elements must be tested to accomplish the end-to-end behaviour. The main RGW operation characteristics to be tested in the trials are the following:

- **Queues functionality:** the RGW implements four queues, one per CoS with Priority Queues scheduling. This functionality will be tested in several scenarios where different upstream/downstream flows (associated with several applications of video streaming, data bulk traffic and VoIP) will be processed at the corresponding queue. At this point will it be relevant the number of flows treated at each RGW and the corresponding queues built at each RGW. Each flow will be marked with the correct CoS and the RGW will be the responsible of locating and processing each flow in the correct queue. The p-bits marked at each specific flow will be used by the Dispatcher block to locate each packet to the proper queue. Automatic and manual flow (by the web-based management interface) will be defined in order to test the Classifier block. The limitation of the flow rates for each CoS, performed by the Policing block, can be tested by playing with several applications at the same time in extreme situations where the Policing block has to decide whether to discard packets or to change the CoS associated to a certain flow/application. The capacity of VLAN management (tagging and untagging 802.1p/q tags), and in particular the treatment of signalling flows (detecting them and processing in the CSD block) in the RGW will be measured at each scenario.
- **NAPT functionality:** this characteristic will be tested at each trial. In the case where the customer at home start a new upstream flow (i.e. web browsing, media session previously signalled with SIP, etc) and NAPT block detects new sessions and then creates new entries in the NAPT table to change the original source port with the corresponding identification. This block also has to detect the end of the session to erase the entry from the NAPT table. For the downstream direction, this box has just to change the destination port and IP address with the matching entry from the NAPT table. A particular interest is the case of end-to-end signalling (i.e with SIP) and NAPT facilities that are studied in the scenario with the VoIP service. This functionality is tested in all the trials, where several flows are defined and there are preconfigured rules in the NAPT block that will accept or deny the downstream and upstream flows at each RGW.
- **Multicast functionality:** downstream flows, i.e in the case of video streaming application, will be replicated to all in-home interfaces manually selecting the multicast option (allow multicast) in the Classifier block. From the side of a terminal at home that wants to subscribe to a multicast group by using the IGMP protocol, the RGW has to process this multicast signalling, save the state of this subscription and accept the corresponding downstream multicast flow that firstly is received in the WAN interface of the RGW and then is transmitted to all the LAN interfaces at home.
- **Signalling functionality:** how the RGW processes the signalling flows, the overhead of the special treatment of this predefined signalling flows, the marking of them with a specific CoS different from the data flows are the main facilities tested in the RGW in relation with the signalling functionality. The overhead of signalling treatment in the RGW will be measure comparing the delay of processing or not the signalling flow in the CSD and the corresponding SP (Signalling Process). An example of test where the signalling is relevant is the VoIP one where the end user terminal uses the SIP protocol (INVITE, REGISTER, etc.) for signalling the caller and callee conversation. Simultaneously with this VoIP traffic, another end user terminal at home can share the bandwidth with other applications, for example, a data bulk traffic or a simple Internet browsing. These tests will let us show how the RGW treat the signalling flow with higher priority than the data flow at the same time. For example, the data flow is assigned to a real time CoS and the signalling one to a low latency CoS.

As described in different documents in MUSE, four CoS (low latency, real time, elastic and best effort) are defined to create a complete set of possibilities to match with flows, and every CoS has different parameters (jitter, delay and throughput). All these parameters must be tested using different configurations in different scenarios where applications with real time restrictions share the medium with best effort ones.

The selection of the applications used for the trials covers the four CoS defined in MUSE and is based on the premise of providing a “full service” testing with a limited number of applications. It is assumed that the global requirements of each service class are representative by the requirements of the corresponding selected applications.

We have chosen the following applications for the trials:

- **Voice over IP** as an example of the *low latency* CoS application for the signalling flows and *real time* for the data flows with an strict delay requirement. For VoIP a main issue is the delay caused by the network due to the limited performance of heavily loaded IP networks. The main requirements for this application are very low delay and jitter (SIP signalling marked as *low latency* and voice data marked as *real time*).
- **Video streaming** as an example of *real time* CoS application and a possible killer application for broadband networks. In addition, this application might generate considerable amounts of traffic and we have divided the tests in two cases, low quality video and high quality video. Within these RGW trials we will test video quality for both unicast and multicast traffic. The bandwidth and packet-loss are the main parameters that can affect the quality of this application. The results of the tests with video streaming application can be classified in qualitative (subjective human perceived quality) and quantitative ones (objective measurement of relevant parameters).
- **Bulk data** as an example of *elastic* or *best effort* CoS application. It is related with Internet browsing but can use (the iperf application) the two TCP or UDP transport protocol with a more relaxed delay requirement. They can be considered commodity applications and can be provided with guaranteed QoS or with the lower quality best-effort. In general, the requirements of these types of applications are low and will depend on the type of data to be transferred. The bandwidth, delay and packet loss are the main parameters that can affect the quality perceived by the users in these services, but the limits of these parameters are defined by the QoS required in every scenario.

An association between these applications and end-to-end network requirements (throughput, delay, packet loss and jitter) need to be defined on the trials. The results of the trials with respect to QoS will consider how to score and measure the perceived QoS at each application and/or scenario: objective measurements, subjective measurements and the mapping from network quality to perceived quality. The preferred method of scoring on these trials is through objective measurements on the testbeds.

2.2.2.2 Web based network management

The web based network management at the RGW enables an easy and flexible manual configuration mechanism based on a web interface. A web server with servlets/jsp capabilities has been installed in the RGW and through it, it is possible to launch a servlet that allows both configuring and monitoring a lot of parameters and is capable of accessing Click! main configuration file. This manual management interface will be tested and compare the results with the automatic OSGi based trial.

2.2.3 Automatic management trial

The automatic management trial focuses on the application of the TR-069 protocol between an ACS and the RGW. In this phase a simple communication will be tested that shows that the RGW can be configured or parameters be retrieved.

The RGW will use OSGI bundles to create a “client side” of the TR-069 protocol.

Communication between the ACS and the RGW takes place by using SOAP/XML encoded messages.

2.3 Testbeds and trials

2.3.1 Network trial testbed

2.3.1.1 Testbed description

In order to test all the different concepts and characteristics previously described the following scenario will be considered as it is depicted in Figure 6.

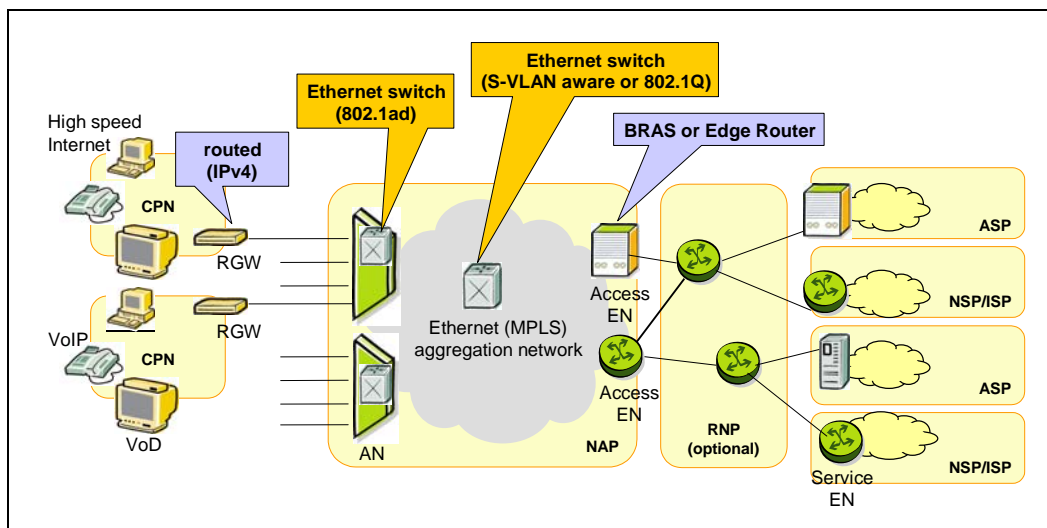


Figure 6. Architecture emulated in the trials

This scenario shows the residential environment on the left part of the picture with two different Customer Networks connected to the Network Access Provider network through different RGW (routed residential gateways), that may be connected to the same or different Access Nodes.

The access node will be 802.1Q aware (like the 802.1ad Ethernet switch in the figure) so that it will be able to understand the VLAN encapsulation coming from the RGW with the corresponding p-bits and it will also be able to reformat the frame according to the VLAN schema used within the Network Access Provider network.

At the other end (left hand side in the picture), the traffic will be received by the different servers that will provide the requested demand.

Since the development done within WPD3 is focused on the RGW itself, the rest of the network is out of the scope of this workpackage. However, in order to be able to properly test the RGW in addition to the validation test performed on a disconnected implementation, it was mandatory to emulate the whole network so that the RGW could in fact be involved into a real triple play scenario with real autoconfiguration on startup performed towards the Access Network, real authentication phase towards the Access Network and definitively real signalling messages exchanged towards the corresponding counter part in the network and real services received from the service provider domain.

Although of course not all the network has been deployed to perform these trials, its functionality has been emulated so that both the residential network environment as well as the servers environment can be included in the trials as if real Access/Edge nodes were providing them the necessary connectivity features so as to link them together through a complete network.

The services scenario that has been chosen to verify the different characteristics and MUSE concepts developed in the prototype is a triple play one with high speed Internet access, voice over IP (SIP-based) and video streaming.

These services will be provided to different residential environments that will also be able to directly interact between each other. All this traffic interchange will be performed within a certain QoS framework that will guarantee the proper treatment of the different flows in the different QoS aware entities so that clients will receive the service without degradation.

The trials over this complete service architecture have been possible through the configuration and installation of the testbed that is depicted in Figure 7.

The two residential environments have been connected to a provider environment where the different servers are located (SIP server, Video Server, Internet Access service, authentication server, etc.). These residential environments consist of different clients for all these services (Web clients and VoD clients on the application layer) but also the capability to be connected between them on a VoIP scenario based on SIP.

The connection to the provider environment has been enabled through a central computer capable of emulating the Access Network functionality and so capable of understanding the framing used both at the server side and at the client side and capable of providing the QoS required by the corresponding traffic.

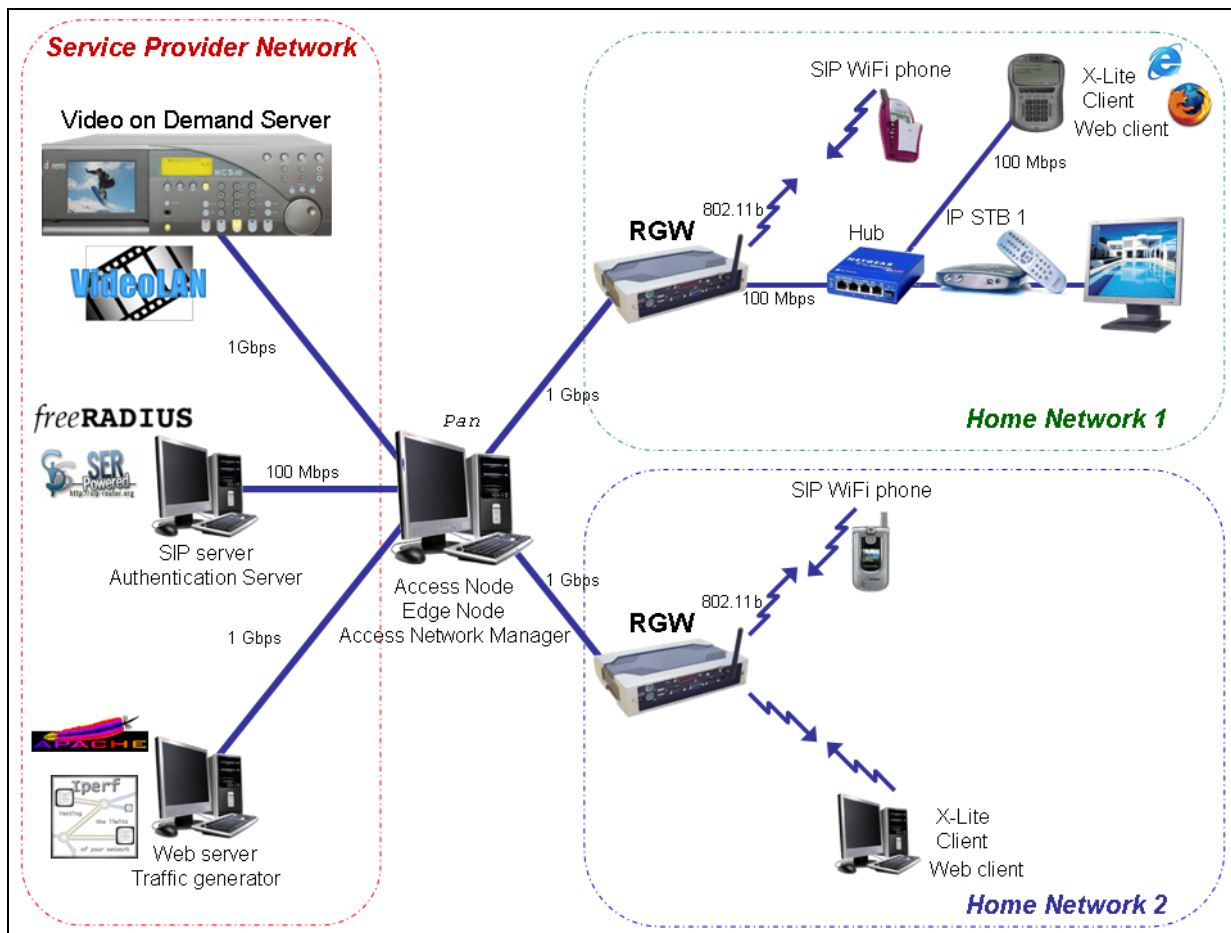


Figure 7. Network trial testbed

2.3.1.2 Testbed configuration

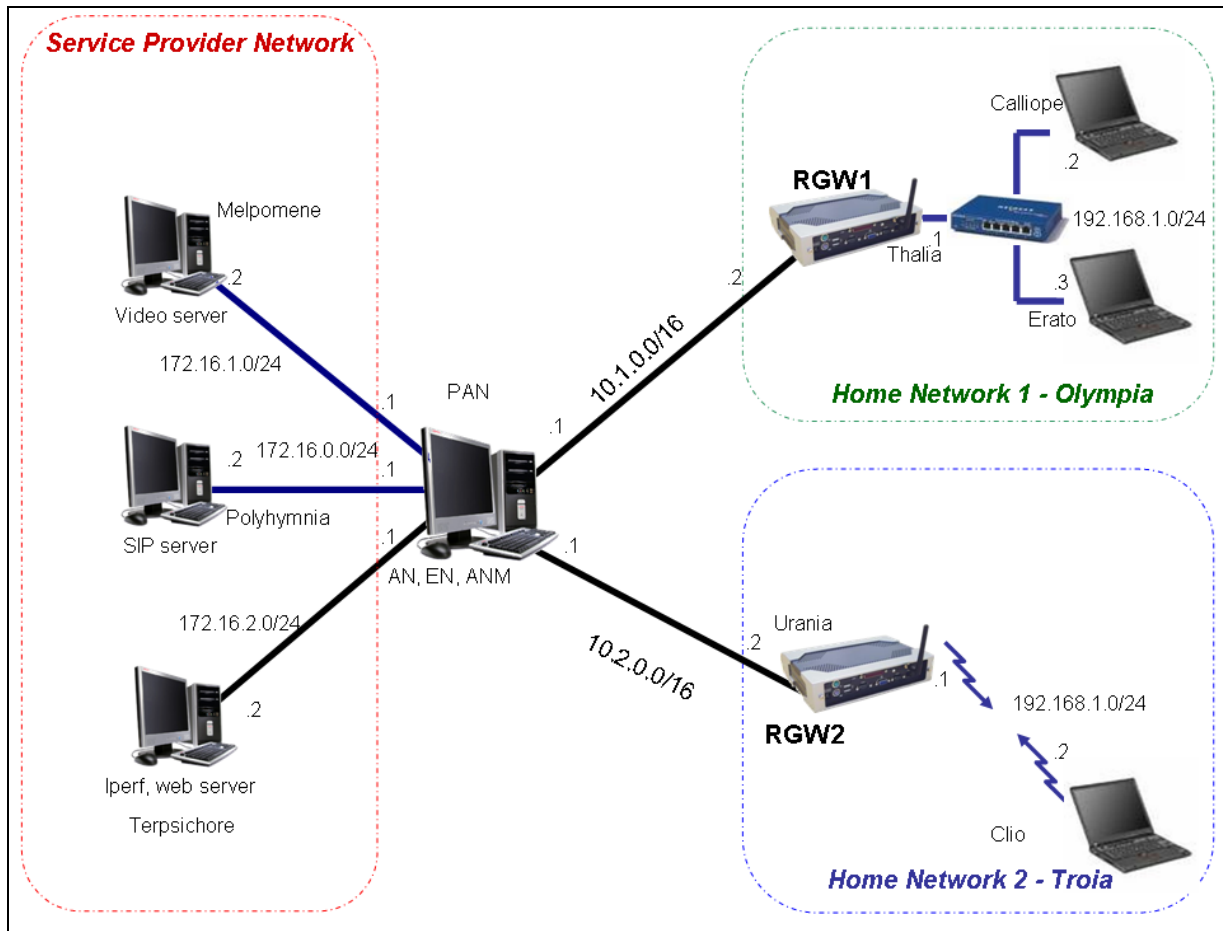


Figure 8. Complete scenario

Figure 8 shows the complete testbed used to develop all the previously described trials. For some trials, a device could be unplugged from the EN or simply powered off just to avoid possible interferences between these devices.

All IP addresses in the Service Provider Network side are statically configured because this part is used just for testing purposes. The dummy device that integrates the AN, EN and ANM is also configured with static IP addresses.

Concerning the authentication trials, hostapd authenticator software, running on the ANM, is configured to act as a RADIUS client of the freeRADIUS server.

At the bootup, a RGW will perform a DHCP Discover to acquire its IP address from a DHCP Server. The ANM runs a DHCP server with the proper configuration to configure the RGWs. At the end of this boot up sequence, the RGW starts its own DHCP server to configure all in-home devices on demand.

Regarding the hardware and the software installed in each device depicted in the complete scenario figure, Table 2 represents the whole specifications list.

	Name	Hardware	Software
Home devices	Calliope Erato	HP Omnibook XE3 Desktop PC 800MHz, 256Mbytes RAM	X-Lite [19], Iperf [20] VLC [18] (client), Iperf
	Clio	Fujitsu-Siemens Amilo M7405	X-Lite, Iperf
RGWs	Thalia	Lex Light 860	Click, Servlet, DHCP server
	Urania	Lex Light 860	Click, Servlet, DHCP server
ANM, AN, EN	Pan	Desktop PC P4 2,4 GHz, 512 Mbytes RAM, 3 Gigabit Ethernet, 2 FastEthernet	Click, DHCP server
Service Provider devices	Melpomene	Desktop PC P4 2,4 GHz, 512 Mbytes RAM, 3 Gigabit Ethernet, 2 FastEthernet	VLC (server), Iperf
	Polyhymnia	Desktop PC 800MHz, 256 Mbytes RAM	SER [17], Iperf
	Terpsichore	Lex Light 823	Iperf

Table 2. Hardware and software involved in the trials

2.3.1.3 Trials description

2.3.1.3.1 Authentication test

As stated in section 1.4.1.2 authentication process is launched by the Start_Gateway that performs other tasks besides the authentication process. For this reason first, trials concerning each one of the separated functions of the Start_Gateway were performed, and then an overall trial of the Start_Gateway script was performed. The Start_Gateway script functions can be roughly grouped in three parts: discovery of the number of interfaces on the RGW, discovery of which one is the WAN interface and the authentication process itself. The purpose of the evaluations of each one of these functions is described on the following sections.

2.3.1.3.1.1 Discovery of the number of ethernet interfaces on the RGW

The objective of this test is to check if all Ethernet compliant interfaces (IEEE802.11 or IEEE802.3) currently available on the RGW are correctly detected by the Start_Gateway script and if the first Click! configuration generated by the script is compliant with the interfaces discovered in the previous step. It should be referred that interfaces that do not have physical connectivity should not be detected nor should appear on the generated Click! configuration file. The test will be made for different sets of interfaces that have physical connectivity at a given time.

2.3.1.3.1.2 *Discovery of the WAN interface:*

The purpose of the test is to test if the script is able to detect which of the discovered interfaces should be interpreted as the WAN interface, and if the second Click! configuration generated is compliant with the interface identified as being the WAN interface. Different tests will be made changing the interface of the RGW that at a given time is connected to the ANM and also changing the content of the authentication frames received by the RGW.

2.3.1.3.1.3 *Authentication process:*

The goal of this test is to check the IEEE802.1X supplicant software (xsupplicant) behaviour under different situations when an EAP-TLS authentication is performed. Evaluations will be performed under failure and success authentication conditions.

2.3.1.3.1.4 *Overall Start_Gateway script trial*

Finally, after the individual test of each block that builds the Start_Gateway script, this last trial has the purpose of checking that the communication between the different blocks of the script is performed correctly and also that the success or failure of an authentication attempt is correctly signalled to the CBCA.

2.3.1.3.2 *Operation trials*

Different operation trials will be performed to accomplish all the previously described characteristics (queues, scheduler, dispatcher, policing, NATP, multicast, signalling) using a video, voice over IP and data bulk traffic scenarios (triple play). These scenarios will be used together using different combinations to simulate many possible combinations but, in this section just single scenarios will be described to avoid possible confusions.

2.3.1.3.2.1 *VoIP scenario*

Figure 9 depicts all the devices and frames direction involved in this scenario. These are the steps needed to perform this test: (1) ANM configures both RGWs at each home to allow traffic flows between all the devices. In particular, the SIP traffic from/to the home is preconfigured in the NATP in order to allow traffic in the corresponding ports. There are different possibilities to perform this action. (2) The SIP phones then register themselves with the SIP server located in the service provider. Once the phones are registered, this phase of registering is not needed in successive calls. The INVITE message also follows this path to the SIP server and finally (3) the data flows directly from the RGWs and the AN.

As is described previously, this simple scenario let us to demonstrate a complete set of functionalities implemented in the RGW like the queues treatment, the two CoS defined for the two main flows, signalling with low latency and data (VoIP) with real time, the NATPT that allow/deny the flows and the overhead of the signalling treated in the RGW that in this case is a simple rely of the SIP messages to the corresponding register and proxy SIP server.

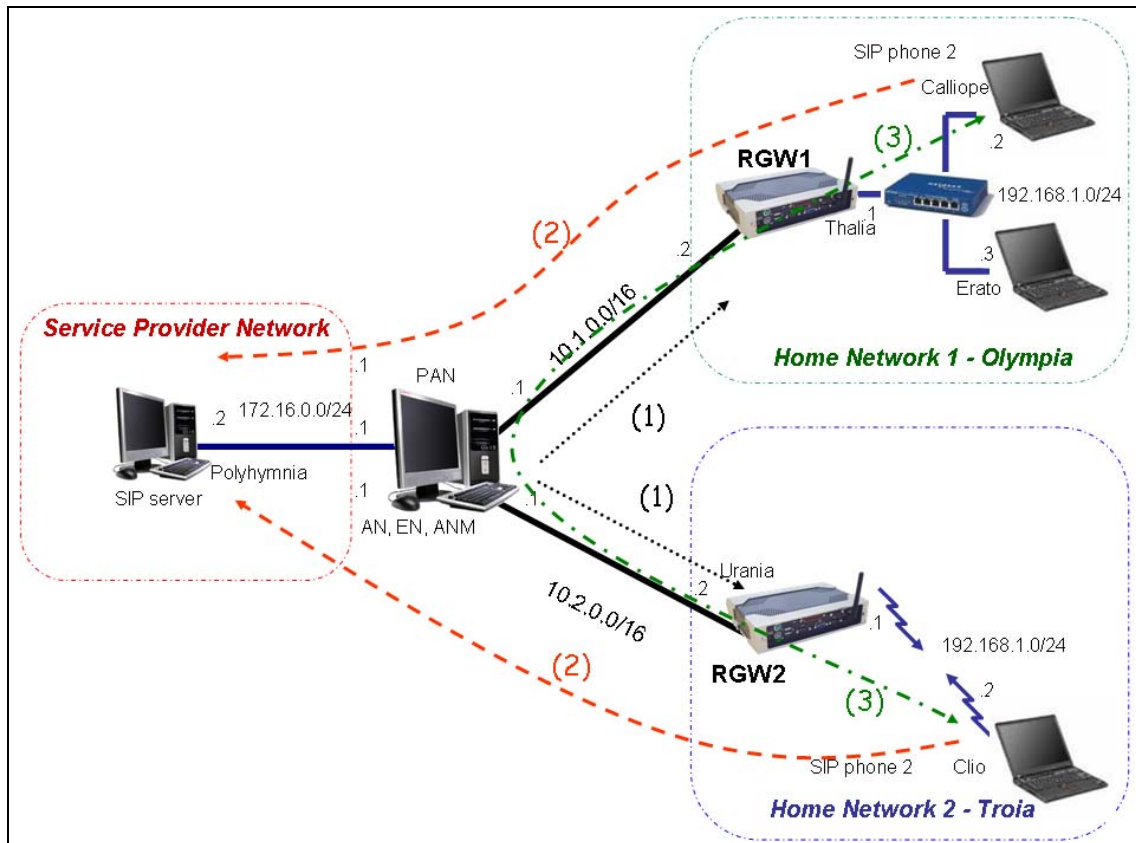


Figure 9. VoIP scenario

2.3.1.3.2.2 Video streaming scenario

Figure 10 describes the video streaming scenario. Clients at home receive a video from the server situated in the service provider by RTP/UDP protocol. The corresponding real time flow is predefined at each RGW in the first step in which the RGW is configured. The trials developed in this scenario are not testing signalling functionalities because the vlc application uses to server the videos does not need any kind of signalling flow to request a specific video to the server, or to respond to a client. However, in this scenario can be implemented trials for testing the following functionalities in the RGW: the queues facilities, the CoS defined for each video flows, the NATPT that allow/deny the flows.

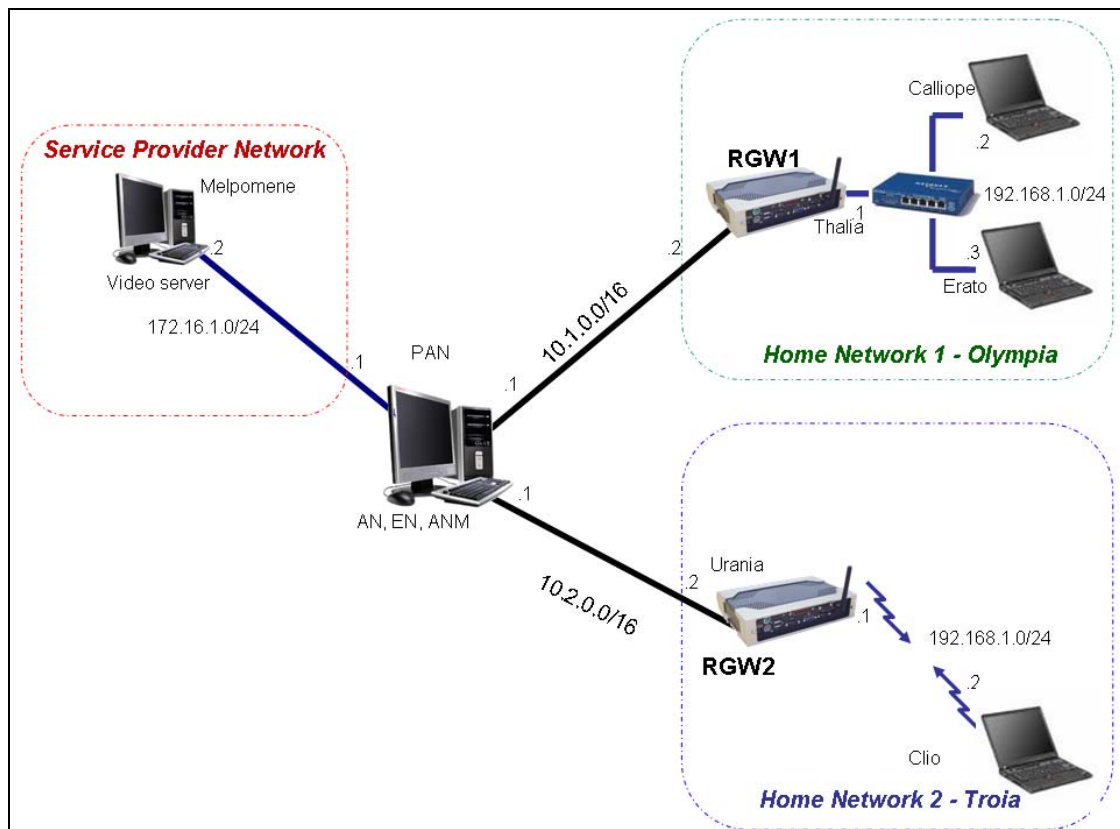


Figure 10. Video streaming scenario

2.3.1.3.2.3 Data bulk scenario

Figure 11 depicts the elements involved in a data bulk scenario. The steps needed to perform this test are (1) each RGWs are configured to support the data traffic at each home with a best effort class or elastic class and (2) the iperf server in the service provider generate traffic with different profiles to several clients at home network 1 and 2. The main functionalities tested in this scenario are: the queues facilities, the CoS defined for each data flows, the NATPT that allow/deny the flows and the generation of the limit traffic supported at each WAN and LAN interfaces at each RGW.

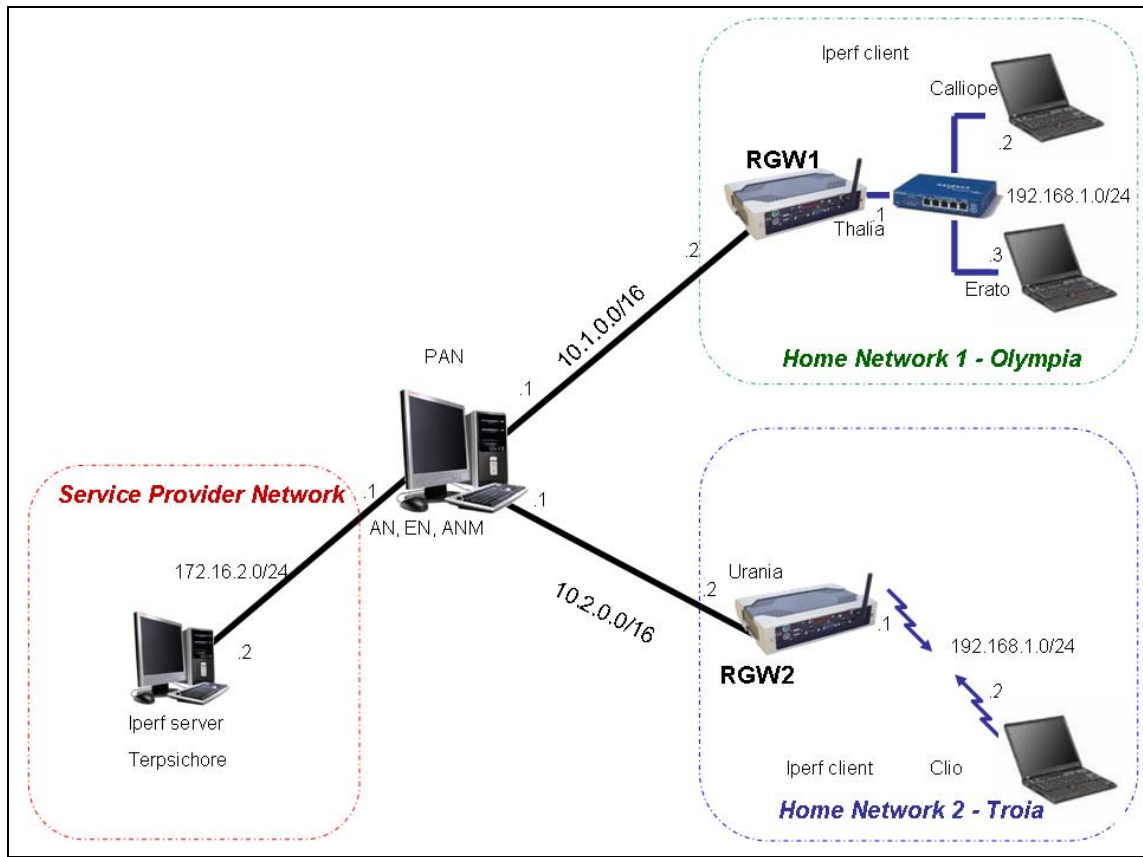


Figure 11. Data bulk scenario

2.3.2 Automatic management evaluation testbed

2.3.2.1 Testbed description

The automatic management has to demonstrate that the ANM can automatically manage the behaviour of the RGW. The elements involved in this management are depicted in Figure 12. This figure shows the relation between the elements needed to achieve the purpose of automatic management.

The scenario includes some devices which are not necessarily completely implemented. As ANM and Click! router. For each case a program for simulate it or a simple file will be necessary.

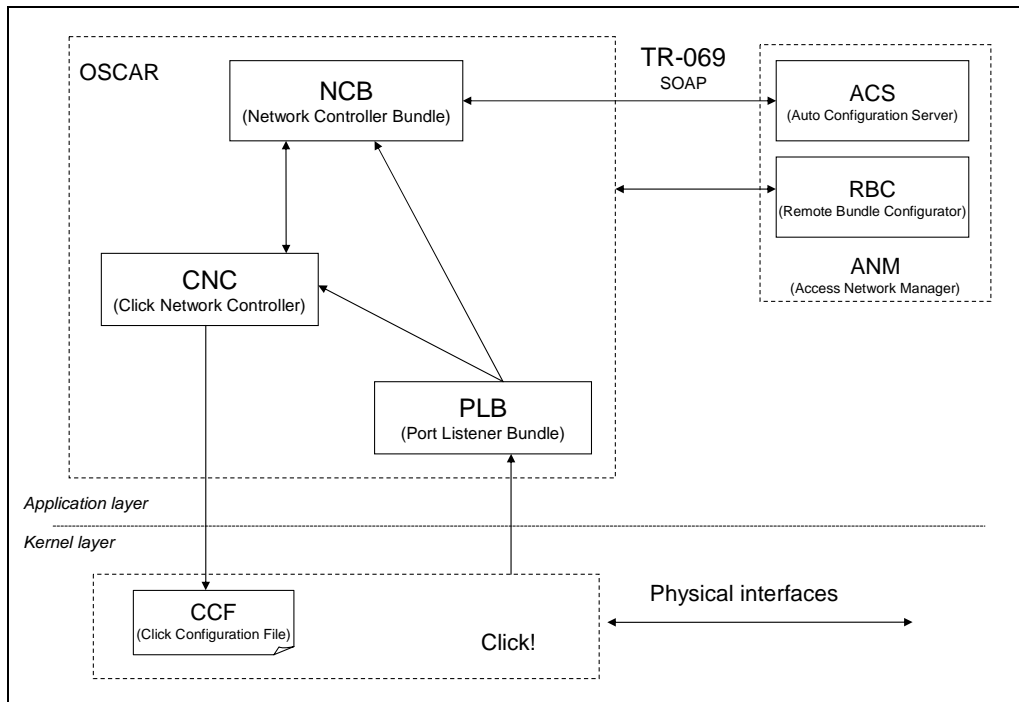


Figure 12. Automatic management testbed

2.3.2.2 Testbed configuration

The functionality described above can be divided in four different testbeds that are related with the different interfaces at the management layer as is depicted in Figure 12. As we can consider these interfaces independent between them, the testbeds are also independent. The different testbeds are described in next subsections.

2.3.2.2.1 NCB-ACS interface

The SW interface for testing the communication between the ACS and the RGW is within the NCB component (OSGI bundle). This interface enables SOAP/XML encoding and decoding and abstracts the actual implementation of the communication with the ACS.

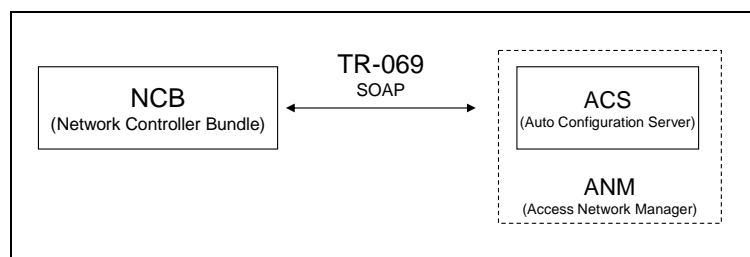


Figure 13. NCB – ACS interface testbed

2.3.2.2.2 NCB-CNC interface and CCF update

The NCB in its role as abstracting the RGW implementation will maintain its own object model representing the TR-069 parameter list. Additionally, it will apply the CNC interface for getting and setting parameter values for each parameter that is under control of the CNC.

The TR-069 object model will reflect the latest state of parameters either provided or retrieved by the ACS.

In this testbed it will be checked that the different parameters defined for the communication between the NCB and CNC works properly. The correction of changes in CCF has also been verified.

For this testbed both bundles (NCB and CNC) has been developed and a dummy CCF have had to be written.

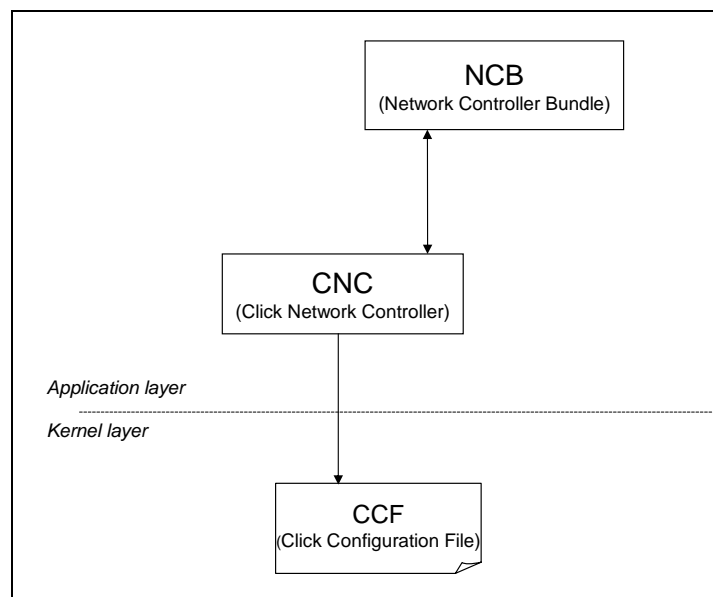


Figure 14. NCB-CNC interface and CCF update testbed

2.3.2.2.3 PLB-NCB PLB-CNC communication

Sometimes a different entity from the OSGi framework may need to modify the behaviour of Click! without using the OSGi framework. That issue has created the need for a bundle that can be listened for any kind of message that Click! wants to send to OSGi or the ACS. This testbed has to check this bundle verifying that can receive a message and communicate it to those bundles which wait for those messages.

For this testbed a PLB bundle has to be created. Also a little program to send messages has to be developed in order to simulate those sendings.

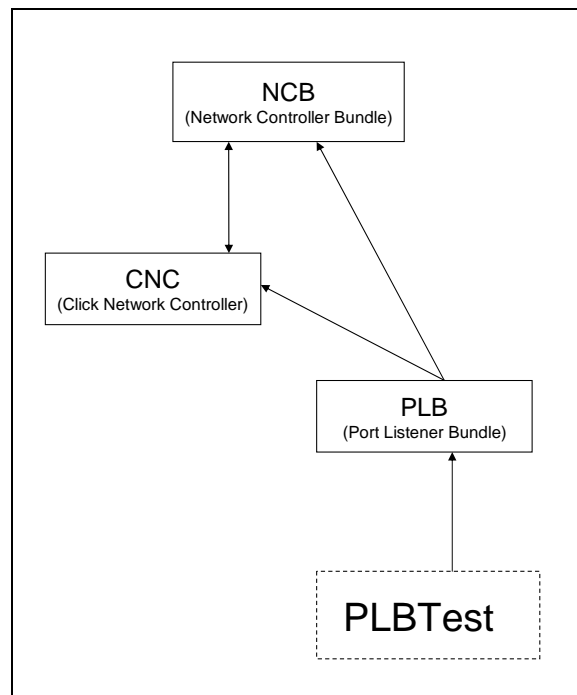


Figure 15. PLB-NCB PLB-CNC communication testbed

2.3.2.2.4 Bundle management

Bundle management consists on providing a tool to change the behaviour of the RGW. This tool consists on changing the bundles at runtime using the possibilities offered by OSGi for those issues.

Therefore this testbed will change some bundles adding more functionality to the RGW. Then, one of the previous testbed has to be checked in order to completely know that the behaviour is updated.

For this testbed new PLB and CNC bundles have been developed as a second version of the first ones.

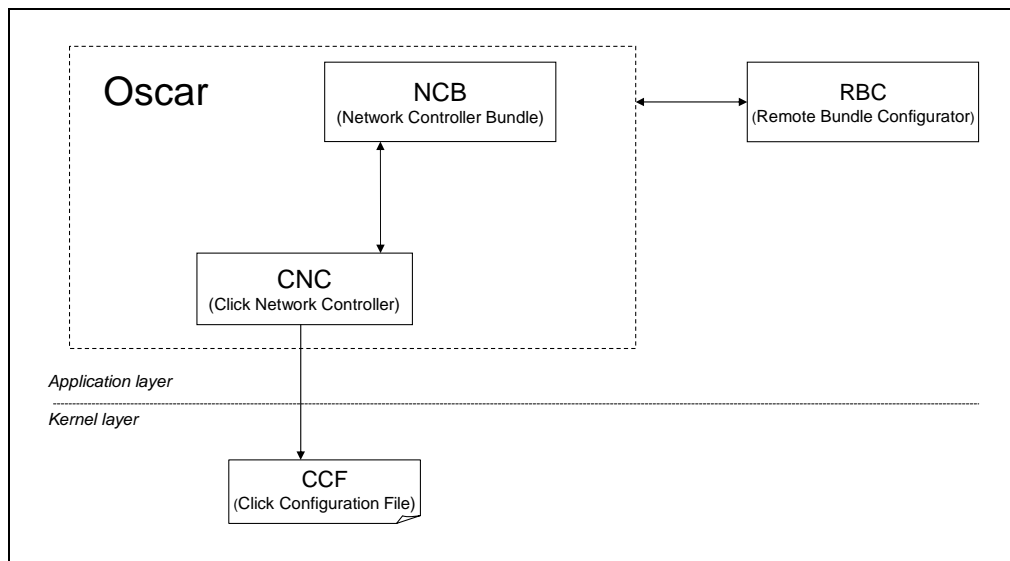


Figure 16. Bundle management testbed

2.3.2.3 Tests description

The functionality described above can be divided in four different tests that are related with the different interfaces at the management layer as it is depicted in Figure 12. As we can consider these interfaces independent between them, four different testbeds have been created and each of one will be explained in next subsections. The functionality of the bundles inside the OSCAR framework is also independent from Click!, because only the CCF is used to communicate between both entities and only will be necessary used the CCF for testing.

2.3.2.3.1 NCB-ACS interface

For this test the ACS and the RGW must be connected (Automatically) and some basic parameter setting and getting will be tested:

1. **RGW Bootstrap**
2. **RGW->inform->ACS**
3. **RGW<-informResponse<-ACS**
4. **RGW->continueSession->ACS**
5. **RGW<-getParameterValues(**
"InternetGatewayDevice.LANDevice.NumberOfEntries",
"InternetGatewayDevice.WANDevice.NumberOfEntries")<-ACS
6. **RGW<-setParameterValues(**
"InternetGatewayDevice.ManagementServer.URL","http://ACSHost:5026/axis/service
s/TR069/ACS",
"InternetGatewayDevice.ManagementServer.Username","RGW"
"InternetGatewayDevice.ManagementServer.Password","TR069")<-ACS

Two PCs are connected via the Intra LAN network. One is running the RGW remote management clients SW the other the ACS SW.

2.3.2.3.2 *NCB-CNC interface and CCF update*

For this test it is needed that the OSCAR framework has been started and the NCB and CNC bundles have also been installed and started. This interface is used when ACS request information from RGW or when ACS wants to modify Click! configuration. Those requests will be translated to NCB and using the interface moved to CNC.

Evaluations have to reflect this interchange of information and are described in the following table:

Identification	Description	Result waited
AM-2-A	NCB request CNC information about one of the possible statistics.	NCB receives data and shows it.
AM-2—B	NCB request CNC information about a LAN interface.	NCB receives data and shows it.
AM-2—C	NCB request CNC information about a WAN interface.	NCB receives data and shows it.
AM-2—D	NCB request CNC information about flows defined.	NCB receives data and shows it.
AM-2—E	NCB send CNC information of a new flow.	CNC change its internal state.
AM-2--F	Test if CCF is updated with new information.	CCF is updated.
AM-2--G	Test if RGW reboot.	RGW is rebooted.

Table 3. Items to be tested in NCB – CNC interface.

2.3.2.3.3 *PLB-NCB PLB-CNC communication*

For this trial it is needed a simple program capable of sending some information to a preconfigured port of the RGW. This program has been programmed to send a message that the PLB understands as a reboot order. This order has to be sent to the NCB in order to notify the ACS that RGW has to reboot and to CNC in order to reboot the machine.

2.3.2.3.4 *Bundle management*

This test needs that some bundles provided by OSCAR are installed in the RGW. Those bundles provide the functionality to install, uninstall, start, stop and refresh the bundles of the RGW. Trial also needs two sets of bundles each of them of a different version that use different parameters. In that case CNC and PLB bundle have been upgraded to understand a new message that forces the update of the CCF.

As the objective is to test that the RGW can change its behaviour updating bundles next Table shows the different steps follow to achieve the objective.

Identification	Description	Expected results
AM-4-A	The same trial described in 2.3.2.3.3	--
AM-4-B	Uninstall of PLB and CNC	PLB and CNC are uninstalled.
AM-4-C	Install of PLB and CNC	PLB and CNC are installed.
AM-4-D	Start of PLB and CNC	PLB and CNC are started.
AM-4-E	The same trial described in 2.3.2.3.3	--
AM-4-F	Sends to RGW a new messages that have to be captured by PLB	CCF updated.

Table 4. Items to be tested in bundle management.

3 RGW TRIALS RESULTS

3.1 Introduction

This section summarizes the results that have been obtained throughout the trialling and evaluation process following the same sequence order used to describe the tests.

The first subsection is showing the results obtained in the trials performed in order to validate the network functionality offered by the RGW: authentication, VLAN management, per flow QoS, scheduling, multicast traffic, etc. This results are obtained from the triple play testbed described on section 2.3.1 first describing the specific results related with the authentication of the RGW and then the results related to the operation of the RGW where all these traffic related tests have been included.

This subsection is also including all the results of the validation trials performed on the automatic configuration and management mechanism based on TR-069 exchange and supported by OSGi.

Finally, the last subsection concludes will a summary of all the work developed in D3 workpackage embedded in this prototype implementation whose trials and results are being described.

3.2 Results

3.2.1 Network trial testbed

3.2.1.1 Authentication tests

Since the tests performed for the start_gateway process did not involve any kind of measurement, the result of above depicted tests will be described only with the keywords “ok” or “not ok”.

Test	Test Case	Result
Discovery of the number of Ethernet interfaces	a) All the interfaces have physical connection	OK
	b) None of the interfaces have physical connection	OK
	c) Other combinations	OK
Discovery of the WAN interface	a) “Correct” EAP-Request received on different interfaces	OK
	b) EAP-Request with string different of “muse.net”	OK
Authentication process	a) Wrong certificate sent	OK*
	b) Re-authentication attempts	OK*
	c) Correct certificate sent	OK**
Overall start_gateway trial	a) Wrong certificate sent	OK
	b) Correct certificate sent	OK

Table 5. Authentication trials

* See Figure 17. Failed authentication attempt and re-authentication attempt

** See Figure 18. Successful authentication

Time	Source	Destination	Protocol	Info
2.615642	00:13:d4:03:2b:d1	Spanning-tree-(for	EAPOL	Start
2.615772	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Request, Identity [RFC3748]
3.108194	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAP	Response, Identity [RFC3748]
3.110482	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Request, EAP-TLS [RFC2716] [Aboba]
3.115261	00:13:d4:03:2b:d1	3Com_a1:24:e4	TLS	Client Hello
3.146849	3Com_a1:24:e4	00:13:d4:03:2b:d1	TLS	Server Hello, Certificate, Server Key Exchange
3.147084	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAP	Response, EAP-TLS [RFC2716] [Aboba]
3.149368	3Com_a1:24:e4	00:13:d4:03:2b:d1	TLS	Server Hello, Certificate, Server Key Exchange
3.160781	00:13:d4:03:2b:d1	3Com_a1:24:e4	TLS	Certificate, Client Key Exchange, Certificate
3.162254	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Request, EAP-TLS [RFC2716] [Aboba]
3.162938	00:13:d4:03:2b:d1	3Com_a1:24:e4	TLS	Certificate, Client Key Exchange, Certificate
3.168612	3Com_a1:24:e4	00:13:d4:03:2b:d1	TLS	Alert (Level: Fatal, Description: Unknown CA)
3.168759	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAP	Response, EAP-TLS [RFC2716] [Aboba]
6.178970	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Failure
16.609519	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAPOL	Start
26.609325	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAPOL	Start
36.609276	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAPOL	Start
46.609292	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAPOL	Start
56.609388	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAPOL	Start
66.329502	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Request, Identity [RFC3748]
66.608337	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAP	Response, Identity [RFC3748]

Figure 17. Failed authentication attempt and re-authentication attempt by supplicant software running on the RGW

Time	Source	Destination	Protocol	Info
0.000000	00:13:d4:03:2b:d1	Spanning-tree-(for	EAPOL	Start
0.000028	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Request, Identity [RFC3748]
2.016225	00:13:d4:03:2b:d1	Spanning-tree-(for	EAPOL	Logoff
2.016252	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Request, Identity [RFC3748]
2.614751	00:13:d4:03:2b:d1	Spanning-tree-(for	EAPOL	Start
2.614879	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Request, Identity [RFC3748]
3.107287	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAP	Response, Identity [RFC3748]
3.109436	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Request, EAP-TLS [RFC2716] [Aboba]
3.115445	00:13:d4:03:2b:d1	3Com_a1:24:e4	TLS	Client Hello
3.147189	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Request, EAP-TLS [RFC2716] [Aboba]
3.148681	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAP	Response, EAP-TLS [RFC2716] [Aboba]
3.150984	3Com_a1:24:e4	00:13:d4:03:2b:d1	TLS	Server Hello, Certificate, Server Key Exchange,
3.168362	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAP	Response, EAP-TLS [RFC2716] [Aboba]
3.169985	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Request, EAP-TLS [RFC2716] [Aboba]
3.174733	00:13:d4:03:2b:d1	3Com_a1:24:e4	TLS	Certificate, Client Key Exchange, Certificate
3.191519	3Com_a1:24:e4	00:13:d4:03:2b:d1	TLS	Change Cipher Spec, Encrypted Handshake Message
3.200990	00:13:d4:03:2b:d1	3Com_a1:24:e4	EAP	Response, EAP-TLS [RFC2716] [Aboba]
3.203811	3Com_a1:24:e4	00:13:d4:03:2b:d1	EAP	Success

Figure 18. Successful authentication

3.2.1.2 Operation trials

3.2.1.2.1 General aspects

For these trials, three different kinds of flows were used to test a triple-play scenario where video and audio applications are represented with video streaming (using the VLC free application for both server and client sides) and VoIP (using SER as the SIP server and X-Lite as the user clients). To simulate constant user data (FTP data for example) Iperf is ideal to generate raw frames in the Server Provider side and collect a lot of statistics in the client side.

Meanwhile the video and data applications were configured using different rates, VoIP was tested using just one codec generating traffic at 120 kbps. This is due to the low rate involved in this kind of transmission and it is not necessary to test with other qualities. To test the video scenario, two different sources were used with different video and audio codecs: low quality where both video and audio are transmitted at 2 Mbps (DivX for video and MP3 for audio) and high quality using 5,2 Mbps (MPEG2 for video and AC3 for audio).

It is important to notice that all experiments were executed during 30 seconds reinstalling all the devices at the beginning and gathering the results at the end of this period. Due to this fact, the relevance of the queues size (10000 frames for each queue) is not so important because for a very long experiment the results could differ for a given value. The aim of these tests is just to probe the feasibility and performance of the QoS system standardised in MUSE and developed for this prototype and not to obtain the best values for the queues length for a given performance.

Iperf was executed from different servers depending on the required QoS. It is always invoked to generate 100 Mbps but the interface could not generate that maximum rate (this maximum rate depends on the network card, the Linux driver, etc. but it was observed that it is around 95 Mbps).

An important value obtained in these tests is the maximum number of packets that have ever been in a queue at once. This parameter is called `highwater_length` in the Queue Click element and will be represented in the results.

3.2.1.2.2 Results

- Two Iperf flows: one marked as Low Latency and the other as Real Time. The Iperf marked as Real Time is invoked first and it can be observed in the server that almost all traffic is received but, as soon as the Iperf with the Low Latency priority is launched, the Real Time traffic received goes down and all the Low Latency traffic sent is received in the server. When the Low Latency traffic finalise, the rate of the Real Time one goes up. The main result of this test is to confirm that traffic with a higher priority shapes the output of a lower one. As can be observed in Table 6 the Iperf traffic marked as Real Time has a high amount of lost packets in the receiver side due to the drops occurred in the Real Time queue at the RGW. The value of the highwater parameter helps us to understand this better and it is clear that the Real Time queue was full for some time period during the test. Another important parameter is the jitter in the Real Time traffic and it demonstrates that the Priority Queuing algorithm used in the Scheduler block is not the optimum because it penalises too much other priorities than the higher one.

Iperf as Low Latency						Iperf as Real Time					
Interval	Transferred	Mean Rate	Jitter	Lost/Sent pkts	Highwater	Interval	Transferred	Mean Rate	Jitter	Lost/Sent pkts	Highwater
0-30.6 s	340 Mbytes	93.1 Mbps	0.176 ms	501/254953 (0.2%)	83	0-30 s	43.4 Mbytes	10.8 Mbps	6.538 ms	220978/253475 (87%)	10000

Table 6. Two Iperf flows

- VoIP. Two different flows were tested: the SIP signalling and the data transfer. Both RGWs and ENs are configured to treat this traffic as low latency meanwhile data is marked as real time. The registration process is usually instantaneous and the delay could be considered negligible. In the data (voice) transmission no packets were dropped in the RGWs and a very small and assumable delay was appreciated.
- Voice and Iperf. The goal of this test is to observe the performance of the voice in a high load scenario when both signalling and data voice is marked with the highest priority. The results of this test were clear: both registration and data (voice) transmissions are perfect with no delay even when the high load traffic was marked with Low Latency quality of service.
 - Iperf marked as Best Effort

VoIP			Iperf					
Signalling	Data	Highwater	Interval	Transferred	Mean Rate	Jitter	Lost/Sent pkts.	Highwater
OK	OK	6	0.0 – 30.6 s	335 Mbytes	91.9 Mbps	0,143 ms	4699/255870 (1.8%)	536

Table 7. Voice and Iperf marked as Best Effort.

- Iperf marked as Real Time

VoIP			Iperf					
Signalling	Data	Highwater	Interval	Transferred	Mean Rate	Jitter	Lost/Sent pkts.	Highwater
OK	OK	3	0.0 – 30.0 s	336 Mbytes	93.7 Mbps	0,191 ms	2489/253823 (0.98%)	20

Table 8. Voice and Iperf marked as Real Time

- Iperf marked as Low Latency

VoIP			Iperf					
Signalling	Data	Highwater	Interval	Transferred	Mean Rate	Jitter	Lost/Sent pkts.	Highwater
OK	OK	59*	0.0 – 30.0 s	339 Mbytes	94,4 Mbps	0,177 ms	421/224668 (0.19%)	20

Table 9. Voice and Iperf marked as Low Latency

- Voice, video (low quality) and Iperf. This test tries to represent a complete triple-play scenario where voice is marked with the highest priority, video uses the Real Time one and Iperf simulates a high load traffic using the lowest priority in one scenario, the same than the video in the second scenario and the highest in the last one. With these three different scenarios we want to test the behaviour of both the voice and the video transmission depending on the high load priority traffic. The results confirm that voice has a too low rate to be affected by other traffic and video with low quality is unaffected too due to this fact.

- o Iperf marked as Best Effort

VoIP			Video		Iperf					
Signalling	Data	Highwater	Data	Highwater	Interval	Transferred	Mean Rate	Jitter	Lost/Sent pkts	Highwater
OK	OK	3	OK	12	0-30.6 s	339 Mbytes	93 Mbps	0.499 ms	2996/255866 (0.78%)	13

Table 10. Voice, video (low quality) and Iperf marked as Best Effort

- o Iperf marked as Real Time

VoIP			Video		Iperf					
Signalling	Data	Highwater	Data	Highwater	Interval	Transferred	Mean Rate	Jitter	Lost/Sent pkts	Highwater
OK	OK	4	OK	55*	0-30 s	331 Mbytes	92.6 Mbps	0.189 ms	1184/249351 (0.47%)	55*

Table 11. Voice, video (low quality) and Iperf marked as Real Time

- o Iperf marked as Low Latency

VoIP			Video		Iperf					
Signalling	Data	Highwater	Data	Highwater	Interval	Transferred	Mean Rate	Jitter	Lost/Sent pkts	Highwater
OK	OK	8*	OK	13	0-30.5 s	340 Mbytes	93.3 Mbps	0.181 ms	528/254886 (0.2%)	8*

Table 12. Voice, video (low quality) and Iperf marked as Low Latency

- Voice, video (high quality) and Iperf. This is another triple-play scenario using a high quality video. This time, when the high load is marked as Low Latency, the video reception is too bad (neither the video nor the audio is received during the high load transmission). As soon as the Iperf transmission ends, the video restarts its reception in the client side (sometimes it needs 2 or 3 seconds to resynchronise). This is another consequence of the Priority Scheduling algorithm and this must be definitely changed for the next prototype version.

- o Iperf marked as Best Effort

VoIP			Video		Iperf					
Signalling	Data	Highwater	Data	Highwater	Interval	Transferred	Mean Rate	Jitter	Lost/Sent pkts	Highwater
OK	OK	5	OK	50	0-32 s	336 Mbytes	88.2 Mbps	0.126 ms	4157/255885 (1.6%)	9178

Table 13. Voice, video (high quality) and Iperf marked as Best Effort

- o Iperf marked as Real Time

VoIP			Video		Iperf					
Signalling	Data	Highwater	Data	Highwater	Interval	Transferred	Mean Rate	Jitter	Lost/Sent pkts	Highwater
OK	OK	4	OK	1886*	0-30 s	314 Mbytes	87.9 Mbps	0.143 ms	3567/238965 (1.5%)	1886*

Table 14. Voice, video (high quality) and Iperf marked as Real Time

- o Iperf marked as Low Latency

VoIP			Video		Iperf					
Signalling	Data	Highwater	Data	Highwater	Interval	Transferred	Mean Rate	Jitter	Lost/Sent pkts	Highwater
OK	OK	81*	KO**	3250	0-31s	335 Mbytes	90.8 Mbps	0.149 ms	4303/255306 (1.7%)	81*

Table 15. Voice, video (high quality) and Iperf as Low Latency

* Queue shared between two different flows

** The video stopped as soon as Iperf started to send traffic and at the end of the 30 seconds interval, the video could resynchronise 2 or 3 seconds later.

- Multicast trials: in this trial a multicast client sends an IGMP report to the network and this message is redirected to the Access Network by the RGW so the RGW starts receiving the multicast frames replicating them to all home interfaces. The trial finalise when all clients in the home receives the multicast frames (detecting this fact using the Ethereal application).

3.2.2 Automatic management evaluation testbed

3.2.2.1 NCB-ACS interface

3.2.2.1.1 Disclaimer

The material presented in this document is the output of the first successful client-server communication with the TR-069 protocol. Please note that this represents work in progress and is subject to frequent changes. Furthermore, a number of issues are still to be resolved, and full compliance to the standard has not been achieved yet.

3.2.2.1.2 Server output

TR069ServerRPC: The Remote Procedure Call (RPC) handler that converts the SOAP/XML messages into procedure calls to the TR069Server module.

TR069Server: Executes the RPCs.

```

===== TR069 Server Handler version 0.2.99 =====
-----
[TR069ServerRPC] Handling inform call to server...
[TR069Server] Device manufacturer = Lucent BL
[TR069Server] Device manufacturer OUI = NL
[TR069Server] Device product class = pc
[TR069Server] Device serial no. = 1234567890
[TR069Server] Event code = 0 BOOTSTRAP
[TR069Server] Event key =
[TR069Server] CurrentTime = Tue Nov 29 10:13:21 CET 2005
[TR069ServerRPC] Returned from inform call to server
[TR069ServerRPC] Handling continueSession call to server...
[TR069ServerRPC] Continuing session with 'GetParameterValues'
[TR069ServerRPC] Handling continueGetParameterValues call to server...
[TR069ServerRPC] Continuing session with 'GetParameterValues'
[TR069ServerRPC] Handling getParameterValuesResponse call to server...
[TR069Server] Param[0]: InternetGatewayDevice.LANDeviceNumberOfEntries = 1
[TR069Server] Param[1]: InternetGatewayDevice.WANDeviceNumberOfEntries = 1
[TR069ServerRPC] Returned from getParameterValuesResponse call to server
[TR069ServerRPC] Continuing session with 'SetParameterValues'
[TR069ServerRPC] Handling continueSetParameterValues call to server...
[TR069ServerRPC] Continuing session with 'SetParameterValues'
[TR069ServerRPC] Handling setParameterValuesResponse call to server...
[TR069Server] SetParameterValues return with status 0
[TR069ServerRPC] Returned from setParameterValuesResponse call to server
[TR069ServerRPC] No more requests in queue.
[TR069ServerRPC] Handling transferComplete call to server...
[TR069Server] Command key = Bootstrapping
[TR069Server] Fault code = 0
[TR069Server] Fault description = No error
[TR069Server] Start time = Tue Nov 29 10:13:21 CET 2005
[TR069Server] Completion time = Tue Nov 29 10:13:53 CET 2005
[TR069ServerRPC] Returned from transferComplete call to server

```

3.2.2.1.3 Messages

Note: The setting up of a TCP session that lasts the entire TR-069 session is still to be implemented. Each message is now sent in a new TCP session.

INFORM message from client to server

```

POST /axis/services/TR069ACS HTTP/1.1
Host: localhost:5026
Connection: keep-alive
Content-Length: 1572
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.2.1
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
X-Forwarded-For: 135.85.69.46, 192.11.224.124
Via: 1.0 nll100849-proxy1 (NetCache NetApp/6.0.1P3D4), 1.1 ge100418-proxy2 (NetCache NetApp/6.0.1P2D1)

<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  instance">
    <soapenv:Body>
      <ns1:Inform soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org">
        <DeviceId href="#id0"/>
        <Event href="#id1"/>
        <MaxEnvelopes xsi:type="xsd:unsignedInt">1</MaxEnvelopes>
        <CurrentTime xsi:type="xsd:dateTime">2005-11-29T09:08:33.881Z</CurrentTime>

```

```

    <RetryCount xsi:type="xsd:unsignedInt">2</RetryCount>
    <ParameterList href="#id2"/>
  </ns1:Inform>
  <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:DeviceIdStruct" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="urn:dslforum-org:cwmp-1-0">
  <Manufacturer>Lucent BL</Manufacturer>
  <OUI>NL</OUI>
  <ProductClass>pc</ProductClass>
  <SerialNumber>1234567890</SerialNumber>
</multiRef>
  <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns3:EventStruct"
xmlns:ns3="urn:dslforum-org:cwmp-1-0"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
  <EventCode>0 BOOTSTRAP</EventCode>
  <CommandKey xsi:type="ns3:CommandKeyType"></CommandKey>
</multiRef>
  <multiRef id="id2" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns4:ParameterValueList" xmlns:ns4="urn:dslforum-org:cwmp-1-0"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
</soapenv:Body>
</soapenv:Envelope>

```

INFORM response from server to client

```

HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/xml;charset=utf-8
Transfer-Encoding: chunked
Date: Tue, 29 Nov 2005 09:11:14 GMT

1e3
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <soapenv:Body>
    <ns1:InformResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org">
      <MaxServerEnvelopes xsi:type="xsd:unsignedInt">1</MaxServerEnvelopes>
    </ns1:InformResponse>
  </soapenv:Body>
</soapenv:Envelope>0

```

“Empty” HTTP POST to server to continue the session

Note: the message should be really empty (no SOAP body); to be implemented.

```

POST /axis/services/TR069ACS HTTP/1.1
Host: localhost:5026
Connection: keep-alive
Content-Length: 395
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.2.1
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""

X-Forwarded-For: 135.85.69.46, 192.11.224.124
Via: 1.0 nl100849-proxy1 (NetCache NetApp/6.0.1P3D4), 1.1 ge100418-proxy2 (NetCache
NetApp/6.0.1P2D1)

<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <soapenv:Body>
    <ns1:ContinueSession
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org"/>
  </soapenv:Body>
</soapenv:Envelope>

```

The server returns a string with the next command so the client knows what RPC to call (in this case "GetParameterValues").

Note: this is non-standard behaviour to deal with the RPC based handling of AXIS; to be implemented.

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/xml;charset=utf-8
Transfer-Encoding: chunked
Date: Tue, 29 Nov 2005 09:11:19 GMT

1f3
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  instance">
    <soapenv:Body>
      <ns1:ContinueSessionResponse
  soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org">
        <NextCommand xsi:type="xsd:string">GetParameterValues</NextCommand>
      </ns1:ContinueSessionResponse>
    </soapenv:Body>
  </soapenv:Envelope>
0
```

The client calls continueGetParameterValues RPC and expects a GetParameterValues request from the server.

Note: this is non-standard behaviour.

```
POST /axis/services/TR069ACS HTTP/1.1
Host: localhost:5026
Connection: keep-alive
Content-Length: 406
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.2.1
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
X-Forwarded-For: 135.85.69.46, 192.11.224.124
Via: 1.0 nl100849-proxy1 (NetCache NetApp/6.0.1P3D4), 1.1 ge100418-proxy2 (NetCache
NetApp/6.0.1P2D1)

<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  instance">
    <soapenv:Body>
      <ns1:ContinueGetParameterValues
  soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org"/>
    </soapenv:Body>
  </soapenv:Envelope>
```

The server sends the GetParameterValues request to the client; parameters requested are:

- InternetGatewayDevice.LANDeviceNumberOfEntries
- InternetGatewayDevice.WANDeviceNumberOfEntries

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/xml;charset=utf-8
Transfer-Encoding: chunked
Date: Tue, 29 Nov 2005 09:11:24 GMT

350
<?xml version="1.0" encoding="utf-8"?>
```

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <soapenv:Body>
    <ns1:ContinueGetParameterValuesResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org">
      <ParameterNames href="#id0"/>
    </ns1:ContinueGetParameterValuesResponse>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:ParameterNames" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="urn:dslforum-org:cwmp-1-0">
      <string>InternetGatewayDevice.LANDeviceNumberOfEntries</string>
      <string>InternetGatewayDevice.WANDeviceNumberOfEntries</string>
    </multiRef>
  </soapenv:Body>
</soapenv:Envelope>

```

The client responds with a GetParameterValuesResponse. The values returned are:

- InternetGatewayDevice.LANDeviceNumberOfEntries = 1
- InternetGatewayDevice.WANDeviceNumberOfEntries = 1

```

POST /axis/services/TR069ACS HTTP/1.1
Host: localhost:5026
Connection: keep-alive
Content-Length: 1477
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.2.1
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
X-Forwarded-For: 135.85.69.46, 192.11.224.124
Via: 1.0 nl100849-proxy1 (NetCache NetApp/6.0.1P3D4), 1.1 ge100418-proxy2 (NetCache
NetApp/6.0.1P2D1)

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <soapenv:Body>
    <ns1:GetParameterValuesResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org">
      <ParameterList href="#id0"/>
    </ns1:GetParameterValuesResponse>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:ParameterValueList" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="urn:dslforum-org:cwmp-1-0">
      <ParameterValueStruct href="#id1"/>
      <ParameterValueStruct href="#id2"/>
    </multiRef>
    <multiRef id="id2" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns3:ParameterValueStruct" xmlns:ns3="urn:dslforum-org:cwmp-1-0"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
      <Name>InternetGatewayDevice.WANDeviceNumberOfEntries</Name>
      <Value xsi:type="xsd:anySimpleType">1</Value>
    </multiRef>
    <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns4:ParameterValueStruct" xmlns:ns4="urn:dslforum-org:cwmp-1-0"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
      <Name>InternetGatewayDevice.LANDeviceNumberOfEntries</Name>
      <Value xsi:type="xsd:anySimpleType">1</Value>
    </multiRef>
  </soapenv:Body>
</soapenv:Envelope>

```

The server responds to this message with a string indicating the next request to come (in this case "SetParameterValues")

Note: this is non-standard behaviour

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/xml; charset=utf-8
Transfer-Encoding: chunked
Date: Tue, 29 Nov 2005 09:11:30 GMT

209
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
    <soapenv:Body>
      <ns1:GetParameterValuesResponseResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org">
        <NextCommand xsi:type="xsd:string">SetParameterValues</NextCommand>
      </ns1:GetParameterValuesResponseResponse>
    </soapenv:Body>
  </soapenv:Envelope>
0
```

The client calls continueSetParameterValues RPC and expects a SetParameterValues request from the server

Note: this is non-standard behaviour

```
POST /axis/services/TR069ACS HTTP/1.1
Host: localhost:5026
Connection: keep-alive
Content-Length: 406
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.2.1

Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
X-Forwarded-For: 135.85.69.46, 192.11.224.124
Via: 1.0 nl100849-proxy1 (NetCache NetApp/6.0.1P3D4), 1.1 ge100418-proxy2 (NetCache
NetApp/6.0.1P2D1)

<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
    <soapenv:Body>
      <ns1:ContinueSetParameterValues
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org"/>
    </soapenv:Body>
  </soapenv:Envelope>
```

The server sends the SetParameterValues request to the client; the parameters to set are:

- InternetGatewayDevice.ManagementServer.URL
(to "http://localhost:8080/axis/services/TR069ACS")
- InternetGatewayDevice.ManagementServer.Username (to "RGW")
- InternetGatewayDevice.ManagementServer.Password (to "TR069")

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/xml; charset=utf-8
Transfer-Encoding: chunked
Date: Tue, 29 Nov 2005 09:11:35 GMT

7c4
<?xml version="1.0" encoding="utf-8"?>
```

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <soapenv:Body>
    <ns1:ContinueSetParameterValuesResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org">
      <ParameterList href="#id0"/>
      <ParameterKey xsi:type="xsd:string">Bootstrapping</ParameterKey>
    </ns1:ContinueSetParameterValuesResponse>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns2:ParameterValueList" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns2="urn:dslforum-org:cwmp-1-0">
      <ParameterValueStruct href="#id1"/>
      <ParameterValueStruct href="#id2"/>
      <ParameterValueStruct href="#id3"/>
    </multiRef>
    <multiRef id="id2" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns3:ParameterValueStruct" xmlns:ns3="urn:dslforum-org:cwmp-1-0"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
      <Name>InternetGatewayDevice.ManagementServer.Username</Name>
      <Value xsi:type="xsd:anySimpleType">RGW</Value>
    </multiRef>
    <multiRef id="id1" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns4:ParameterValueStruct" xmlns:ns4="urn:dslforum-org:cwmp-1-0"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
      <Name>InternetGatewayDevice.ManagementServer.URL</Name>
      <Value
xsi:type="xsd:anySimpleType">http://localhost:8080/axis/services/TR069ACS</Value>
    </multiRef>
    <multiRef id="id3" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns5:ParameterValueStruct" xmlns:ns5="urn:dslforum-org:cwmp-1-0"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
      <Name>InternetGatewayDevice.ManagementServer.Password</Name>
      <Value xsi:type="xsd:anySimpleType">TR069</Value>
    </multiRef>
  </soapenv:Body>
</soapenv:Envelope>

```

The client responds with a SetParameterValuesResponse. The return value indicates the status (here: 0, which means the parameters have been set as requested).

```

POST /axis/services/TR069ACS HTTP/1.1
Host: localhost:5026
Connection: keep-alive
Content-Length: 650
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.2.1
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
X-Forwarded-For: 135.85.69.46, 192.11.224.124
Via: 1.0 nl100849-proxy1 (NetCache NetApp/6.0.1P3D4), 1.1 ge100418-proxy2 (NetCache
NetApp/6.0.1P2D1)

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
  <soapenv:Body>
    <ns1:SetParameterValuesResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org">
      <Status href="#id0"/>
    </ns1:SetParameterValuesResponse>
    <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="xsd:int"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">0</multiRef>
  </soapenv:Body>
</soapenv:Envelope>

```

The server has no more requests to send at this stage and should send back an empty HTTP Response.

Note: the message should be really empty (no SOAP body); to be implemented; here a "TransferComplete" string is returned to request the end of the session.

```

HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/xml;charset=utf-8
Transfer-Encoding: chunked
Date: Tue, 29 Nov 2005 09:11:40 GMT

207
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  instance">
    <soapenv:Body>
      <ns1:SetParameterValuesResponseResponse
  soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org">
        <NextCommand xsi:type="xsd:string">TransferComplete</NextCommand>
      </ns1:SetParameterValuesResponseResponse>
    </soapenv:Body>
  </soapenv:Envelope>
0

```

The client calls the TransferComplete RPC to end the session.

```

POST /axis/services/TR069ACS HTTP/1.1
Host: localhost:5026
Connection: keep-alive
Content-Length: 1184
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.2.1
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
X-Forwarded-For: 135.85.69.46, 192.11.224.124
Via: 1.0 nl100849-proxy1 (NetCache NetApp/6.0.1P3D4), 1.1 ge100418-proxy2 (NetCache
NetApp/6.0.1P2D1)

<?xml version="1.0" encoding="UTF-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  instance">
    <soapenv:Body>
      <ns1:TransferComplete
  soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org">
        <CommandKey xsi:type="ns2:CommandKeyType" xmlns:ns2="urn:dslforum-org:cwmp-1-
  0">Bootstrapping</CommandKey>
        <FaultStruct href="#id0"/>
        <StartTime xsi:type="xsd:dateTime">2005-11-29T09:08:33.881Z</StartTime>
        <CompleteTime xsi:type="xsd:dateTime">2005-11-29T09:09:06.806Z</CompleteTime>
      </ns1:TransferComplete>
      <multiRef id="id0" soapenc:root="0"
  soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="ns3:FaultStruct"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:ns3="urn:dslforum-org:cwmp-1-
  0">
        <FaultCode href="#id1"/>
        <FaultString>No error</FaultString>
      </multiRef>
      <multiRef id="id1" soapenc:root="0"
  soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:type="xsd:int"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">0</multiRef>
    </soapenv:Body>
  </soapenv:Envelope>

```

The server sends a TransferCompleteResponse message, which contains no other information.

```

HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/xml; charset=utf-8
Transfer-Encoding: chunked
Date: Tue, 29 Nov 2005 09:11:45 GMT

194
<?xml version="1.0" encoding="utf-8"?>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
    <soapenv:Body>
      <ns1:TransferCompleteResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://wsdl.tr069.cwmp.lucent.org"/>
    </soapenv:Body>
  </soapenv:Envelope>
0

```

3.2.2.2 NCB-CNC interface and CCF update

The results of the test are described in next table:

Identification	Results
AM-2-A	OK
AM-2-B	OK
AM-2-C	OK
AM-2-D	OK
AM-2-E	OK
AM-2-F	OK
AM-2-G	OK

Table 16. Results of the NCB – CNC interface test

3.2.2.3 PLB-NCB PLB-CNC communication

This test that was as simple as sending a message to a preconfigured port, producing the expected results.

3.2.2.4 Bundle management

The results of the test are described in next table:

Identification	Results
AM-4-A	OK
AM-4-B	OK
AM-4-C	OK
AM-4-D	OK
AM-4-E	OK
AM-4-F	OK

Table 17. Items to be tested in bundle management

3.3 Conclusions

This document has shown the different tests and trials that have been performed to the RGW prototype developed in WPD3. These specific tests have been applied on the RGW in order to demonstrate the application of different MUSE concepts such as the authentication test, the automatic management mechanism or the QoS performance tests. Apart from them, a complete triple play scenario has been trialled in order to complement the tests and in order to evaluate the network performance operation of the prototype. This section is summarizing the most important conclusions.

A mechanism for RGW authentication has been successfully tested (802.1X) and integrated within the prototype complete automatic bootup sequence which is also including the required procedures in order to obtain different parameters required by a MUSE compliant RGW (gateway, DNS and ANM IP addresses, number and name of the interfaces, id of the WAN interface, etc.). After running this process, it has been verified that the RGW is capable of automatically obtaining a network connection as well as automatically provide a network connection to the terminals located in the home network.

The trials to test the network part of the RGW were designed to evaluate the operation of the gateway in accordance with MUSE QoS functionality, when multiple flows with different QoS marks arrive to the device and so different prioritization has to be applied on their treatment.

This trial has been done over a gigabit access where three different flows are being received by the RGW at the same time (triple play): video streaming, VoIP and Iperf application data or bulk data (actually two different trials depending on the quality of the video stream). The main result of all these tests is a qualitative result supported by numeric values that confirm the appreciated parameters (video and audio quality). Due to the QoS treatment done in the different queues according to MUSE specifications, in the trial it was not possible to appreciate from the clients located in the home network terminals any distortion neither in the video nor in the audio (VoIP) reception when these priorities are properly assigned (VoIP as Low Latency, Video as Real Time and background traffic as Best Effort). Even if the video streaming is transmitted with a high video and sound quality (DVD quality) the reception of both video and VoIP is correct.

According to MUSE specifications the priority scheduling algorithm is the one used for queue management. As consequence it could be tested that when the background traffic priority is increased (to the highest one for example), the video is not properly received at the home device anymore in case. The VoIP reception however, is always good due to the low rate transmission it requires, no matter how much traffic is sent in the other flows, because in the case of flows with the same priority the traffic is served in a Round Robin fashion (it should be necessary too many flows to collapse the VoIP performance).

Another important conclusion (it was already known in advance, but the laboratory trials have confirmed it) is the notorious problems that some applications are presenting when they are executed behind a NAPT box. In these tests both the VoIP and the video applications experienced these problems in the signalling phase when SIP in the VoIP scenario and RTSP in the video one started to negotiate the transport ports. Special developments were needed in order to overcome these problems.

All the triple play tests and trial were guided by a web based configuration application that has a clean interface that accelerates the creation and destruction of flow rules, the queue length modification and the statistics visualisation. It also enables the administrator with a 'history' functionality that allows restoring a previous saved configuration, cleaning it completely or even making a hard reset restarting the RGW with the original configuration.

Apart from this trial, a specific test on Automatic Autoconfiguration of the gateway by an autoconfiguration server has been shown in the context of an OSGi enabled gateway providing independency of the operative system to manage services and applications. The usage of an OSGi bundle for the client side of the TR-069 interface has proven to be flexible. OSGi has also been used to provide an easily maintainable interface from the TR-069 client towards the software boxing the actual implementation of the gateway control.

The gateway was able to contact the ACS and to send all parameter values specified in TR-098. The ACS was able to initiate a session by addressing a HTTP port on the gateway. When the gateway received a GET on this port it would set up a session with the ACS.

In the SOAP/XML encoding some hurdles with respect to the compliance to the TR-069 standard are taken by a workaround.

It can be concluded that a proof of concept has been given for a TR-069 remote configured gateway while using OSGi as a flexible platform for implementation the TR-069 client side of the protocol.

In addition to this Automatic Autoconfiguration capability, OSGi has also been proved to be a good solution to remotely manage the behaviour of the RGW Autoconfiguration functionality. It has been tested that it is possible for the operator to automatically update the bundles that are installed in the RGW and so it is also possible for the operator to include new functionalities whenever it is needed very easily.

ANNEXE A

Operator manual

This annexe describes the operator manual so as to be able to configure the different options of the RGW using the web interface to directly access the RGW.

Web configuration interface

Interface specification and functionality

The RGW manual configuration interface is based on a web environment with an HTML/Javascript client front-end and the business logic based on different servlets which are responsible for the communication with the RGW internals (modification of Click! configuration by rewriting the Click! configuration files and reinstalling the Click! kernel module).

This configuration interface was already introduced in [1] but different changes and updates have been introduced in order to allow a better configuration of the RGW.

Figure 19 shows the different options available in the interface that will be developed in this short operator manual (this web interface will be available both from the WAN side and from the LAN side). All these options will always be shown in the upper part of the window so that it is easy and fast to change from one option to another.



Figure 19. Main configuration frame

Upstream flow definition interface

In order to configure the different rules (multifield classifier capable of specifying rules related to frame marking for QoS purposes, firewalling purposes, etc.) for the flows that will traverse the RGW in the upstream direction, frames going from the LAN side towards the WAN side, the following web page is used:

Upstream Flows

Source IP: Dest IP: Mac Src:

Src If: Protocol: Src Port: Dest Port:

DSCP: List Order: Action:

Rules:

1 If eth0 Mac Src: FF:AB:AC:34:AC:FA IP Src: 192.117.139.1/32 IP Dst: 192.117.139.2/32 Proto: 123 SPort: 123 DPort: 123 DSCP: 10 Action: Mark as Low Latency

2 IP Src: 126.123.145.1/32 Action: Best Effort

3 IP Src: 192.168.1.10/32 Proto: TCP Action: Mark as Elastic

Figure 20. Upstream configuration web page

The following sections are included in this configuration page:

- **‘Flow definition’ area:** in this part the different parameters that will be used to characterise every upstream flow are presented. The last parameter defines the action that will be provided to the frame in case the flow matches the specified parameters. The different parameters managed by this interface are:
 - Source IP: X.X.X.X text field
 - It is also possible to define a complete range through the addition of an IP prefix alter the address (‘/XX’).
 - Destination IP: X.X.X.X text field
 - It is also possible to define a complete range through the addition of an IP prefix alter the address (‘/XX’).
 - Source MAC: XX:XX:XX:XX:XX:XX text field
 - There is no ‘destination MAC’ field since it will always be the one of the RGW.
 - Source interface: single choice menu
 - The possible values can be: Eth1, Eth2, Wireless, etc. (by default, this field will be empty).
 - Protocol: X (number) text field
 - The administrator may specify any valid protocol number. The application will also understand some common protocol names and will translate them into numbers before transferring the rule into click: ‘tcp’, ‘TCP’, ‘udp’, ‘UDP’, ‘icmp’, ‘ICMP’, ‘igmp’ and ‘IGMP’.
 - Source port: X (number) text field
 - Destination port: X (number) text field
 - DSCP: X (number) text field
 - Between 0 and 2^4-1

- List order:
 - This is the place within the current list of rules where the rule that is being defined will be inserted (in case there are 10 rules, for example, a the number 5 is specified here it implies that this rule will be 5th one in the list, the previously 5th rule now will be the 6th one and so on).
 - The default value for this field is 1 (the new rules are always inserted at the beginning of the list unless other value is specified).
 - As it will be seen later, in the 'current rules' area the rules will also show their precedence order and it will be editable (in case a new number is written in the precedence field the whole rule list will be reordered).
- Action: `single choice menu`
 - Possible upstream actions (in case the rule matches the outgoing flow):
 - 'Send to CSD': the packet will be sent to the CSD (this is typically used for signalling packets) so that they can be properly treated at the application layer.
 - 'Send to localhost': the packet will be sent to the RGW kernel, allowing packets going to the RGW to reach its destination (for example packets going to the web server enabled so as to configure the RGW).
 - 'Mark as Low Latency': the packet will be sent to the Low Latency queue.
 - 'Mark as Real Time': the packet will be sent to the Real Time queue.
 - 'Mark as Elastic': the packet will be sent to the Elastic queue.
 - 'Mark as Best Effort': the packet will be sent to the Best Effort queue. This will be the default option.
 - 'Discard': the packet will be discarded (firewall rule).
 - 'Signalling': the packet will be sent to the highest priority queue.
 - 'Local': this traffic is intended to be forwarded through another local interface (will not be forwarded to the WAN).
- Notes:
 - In case a field is left empty, when the rule is processed any value in that field of the flow will be accepted as valid (this is why it is so important the precedence field that will establish in which order are the rules processed).
 - The application checks for validity of the fields (valid IP address, valid Ethernet address, valid protocol or DCSP number, etc.) before it is included into Click! so it avoid any kind of RGW misconfiguration.
- **'Buttons' area:** this area has four buttons to add or to remove a rule (or a selected set of rules), to show the currently activated rules at the 'current rules area' or to Update Click, a button that will start the reconfiguration sequence of Click.
 - **Add Rule:** when this button is pressed, the servlet gets the request and checks all the given parameters. If all they are correct, a new formatted rule


```
(order;interface;eth_src;eth_dest.;ip_src;ip_dst;protocol;port_src;port_dst;dscp;action;)
```

is created and added at the right place in the flows configuration file.

- **Delete Rules:** when this button is pressed, the servlet processes the request, which contains the rules as parameters. The servlet searches for these rules at the flows configuration file and deletes them.
- **Update Click:** this button makes the servlet start the auto-configuration sequence installing the ruled into Click.
- **'Current rules' area:** in this area the rules that have been added are shown (some administration rules may remain hidden in this area, since it is not possible to remove them). They will appear numbered and in the order that they will be applied (first the number 1) and with a choice selection box (it will allow to select multiple rules so as to remove them altogether).
 - It is important to note that the rules displayed may still be uninstalled (inactive). These rules are the ones that have been added using the Add Rule button and will became active as soon as the Update Click! button is pressed. If the operator is interested in the rules that are already installed into the RGW the List Active Rules interface should be accessed.

Downstream flow definition interface

In order to configure the different rules for the flows that will traverse the RGW in the downstream direction, for frames coming from the WAN side, the following web page is used:

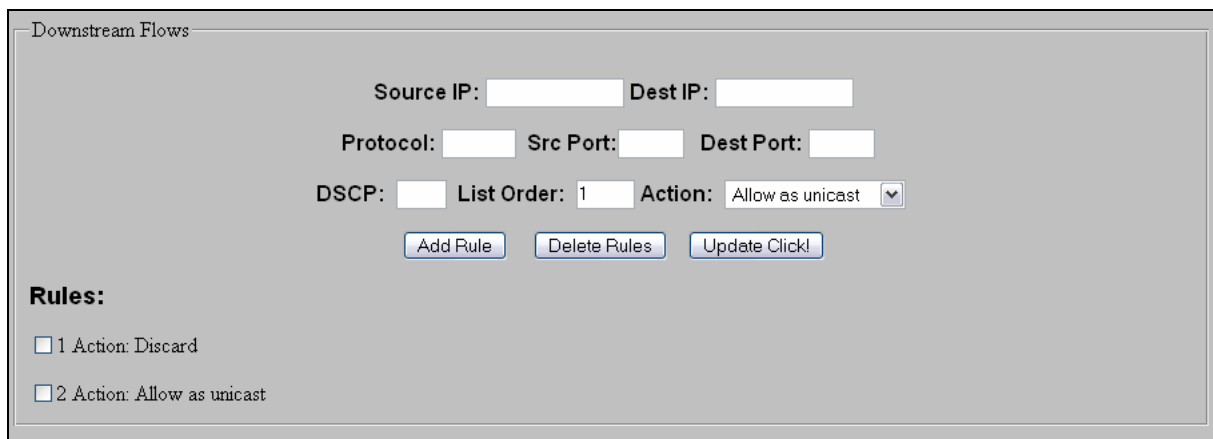


Figure 21. Downstream configuration web page

The following sections are included in this configuration page:

- **'Flow definition' area:** in this part the different parameters that will be used to characterise every downstream flow are presented. The last parameter defines the action that will be performed with the frame in case the flow matches the specified parameters. The different parameters managed by this interface are:
 - Source IP: text field
 - It is also possible to define a complete range through the addition of an IP prefix alter the address ('/XX').
 - Destination IP: text field
 - It is also possible to define a complete range through the addition of an IP prefix alter the address ('/XX').

- Protocol: X (number) text field
 - The administrator may specify any valid protocol number. The application will also understand some common protocol names and will translate them into numbers before transferring the rule into click: 'tcp', 'TCP', 'udp', 'UDP', 'icmp', 'ICMP', 'igmp' and 'IGMP'.
- Source port: X (number) text field
- Destination port: X (number) text field
- DSCP: X (number) text field
 - Between 0 and 2^4-1
- List order:
 - This is the place within the current list of rules where the rule that is being defined will be inserted (in case there are 10 rules, for example, a the number 5 is specified here it implies that this rule will be 5th one in the list, the previously 5th rule now will be the 6th one and so on).
 - The default value for this field is 1 (the new rules are always inserted at the beginning of the list unless other value is specified).
 - As it will be later seen, in the 'current rules' area the rules will also show their precedence order and it will be editable (in case a new number is written in the precedence field the whole rule list will be reordered).
- Action: single choice menu
 - Possible downstream actions (in case the rule matches the incoming flow):
 - 'Send to CSD': the packet will be sent to the CSD (this is typically used for signalling packets) so that they can be properly treated at the application layer.
 - 'Send to localhost': the packet will be sent to the RGW kernel, allowing packets going to the RGW to reach its destination (for example packets going to the web server enabled so as to configure the RGW).
 - 'Allow as unicast': it will be sent to the queue indicated by the p-bits. In case it is not marked it will be sent to the best effort queue.
 - 'Allow as multicast': it will be sent to the queue indicated by the p-bits. In case it is not marked it will be sent to the best effort queue. Once the packet has left the queue it will be resent by all LAN interfaces so that it will be able to reach the multicast client.
 - 'Discard': the packet will be discarded (firewall rule).
 - 'Signalling': the packet will be sent to the highest priority queue.
- Notes:
 - In case a field is left empty, when the rule is processed any value in that field of the flow will be accepted as valid (this is why the precedence field, that will establish in which order are the rules processed, is included as an optional field for any added rule).
 - The application checks for validity of the fields (valid IP address, valid Ethernet address, valid protocol or DCSP number, etc.) before it is included into Click! so it avoid any kind of RGW misconfiguration.

- **'Buttons' area:** this area has the same buttons and functionality as the downstream button area.
- **'Current rules' area:** this area has the same functionality as the upstream button area.

List rules interface

This informational web page shows the complete list of rules that are configured in the RGW, including:

- **Preconfigured rules:** the rules that have been preconfigured and are not shown in the flow definition interface (since they cannot be deleted), like the rules to allow the web configuration of the RGW.
- **Automatically configured rules:** the ones that have been automatically generated by some signalling protocol (SIP establishing the QoS for a certain flow, IGMP allowing incoming multicast traffic, etc.).
- **User defined rules:** the rules configured using the flow definition interface (both for the upstream and downstream traffic).

An example of this interface is shown in Figure 22.

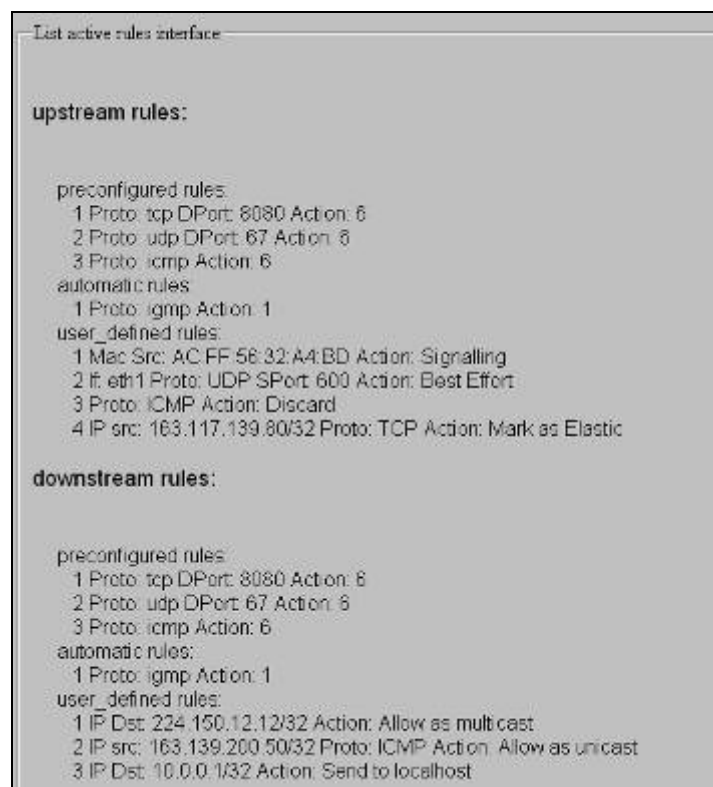


Figure 22. List active rules interface

Configuration interface

This web interface shows information related to the configuration of the network parameters of the RGW (some of them are purely informational and automatically read from the RGW already configured parameters and some other may be also changed).

The different parts of this interface are the following:

- Interface list (automatically read from the RGW):
 - Name: eth0, eth1, etc. text field (not editable)
 - IP: X.X.X.X text field (not editable)
 - Subnetmask: X.X.X.X text field (not editable)
 - MAC: XX:XX:XX:XX:XX:XX text field (not editable)
 - Prefix: /XX text field (not editable)
 - Type of interface: LAN/WAN
- ACS (Autoconfiguration Server): X.X.X.X text field
- DNS server: X.X.X.X text field

The second part of this interface allows the operator to configure the size of the different queues associated to every class of service (both in the upstream and the downstream direction).

In order to do this, the operator must choose in the single choice boxes the class of service and the flow direction and then indicate the number of packets of the queue (from 1 to 30000). The `Update` button will install these values into the RGW. The sizes of the different queues can be seen pressing the `View` button.

Finally, this interface is also able to manage the configuration history base of the RGW so that the operator is able to save and restore Click! configurations.

- Every time the button 'Save' is pressed, the RGW actual configuration is stored into the 'history' directory (using an automatically generated name based, composed by the date and the time).
- A list of stored configurations will be shown, each one with its corresponding selection box (single choice selection). Clicking into the hyperlinked date of the configuration, the user will be able to see the content of the XML configuration file.
- There are different possible options to be executed over the selected configuration:
 - `Delete`: this will remove all the files associated with this configuration and update the list of configurations.
 - `Soft reset`: this will restart Click! with the selected configuration.
 - `Hard reset`: this will restart the whole machine with the selected configuration (this will perform all the DHCP procedure, etc.).
 - Another option, 'Clear configuration', will install an empty configuration file on the RGW (as if it were started for the very first time).



Figure 23. Residential Gateway configuration interface

Statistics interface

This web page shows different statistics retrieved from the RGW. It is not possible to edit any field since all of them are just informational. The following statistics are displayed:

Upstream and Downstream Packets Statistics		
Packet Class	Upstream	Downstream
Total	- [Kbytes]	- [Kbytes]
Low Latency	- [Kbytes] (-%) - packets lost (-%)	- [Kbytes] (-%) - packets lost (-%)
Real Time	- [Kbytes] (-%) - packets lost (-%)	- [Kbytes] (-%) - packets lost (-%)
Elastic	- [Kbytes] (-%) - packets lost (-%)	- [Kbytes] (-%) - packets lost (-%)
Best Effort	- [Kbytes] (-%) - packets lost (-%)	- [Kbytes] (-%) - packets lost (-%)
Signalling	- [Kbytes] (-%)	- [Kbytes] (-%)
To CSD	- [Kbytes] (-%)	- [Kbytes] (-%)
Discarded at Firewall	- [Kbytes] (-%)	- [Kbytes] (-%)
Non Matching	- [Kbytes] (-%)	- [Kbytes] (-%)
TCP Protocol	- [Kbytes] (-%)	- [Kbytes] (-%)
UDP Protocol	- [Kbytes] (-%)	- [Kbytes] (-%)

Updated at: Fri Oct 28 12:15:06 CEST 2005

Figure 24. Statistics interface

- Total number of packets and bytes going upstream and downstream.
- Number of packets and bytes going upstream and downstream per class of service (including the percentage out of the total number displayed above and the number of packets discarded in the queues associated with every CoS).
- Total number of signalling packets and bytes going upstream and downstream.
- Total number of packets and bytes going towards the CSD.
- Total number of packets and bytes coming from the CSD.
- Total number of packets and bytes discarded by the firewall.
- Total number of packets and bytes discarded because they do not match any rule (non matching packets).
- Total number of TCP and UDP packets going upstream and downstream.
- Elapsed time since the last change in the configuration file.

It is also important to note that for the four different classes of service, the number of lost packets and its total percentage is shown (all these numbers related to the queue associated to the corresponding CoS).

Traffic generation interface

This interface allows the operator to generate frames from the RGW with the possibility of specifying different parameters of this traffic flow. Although it is not a RGW formalized functionality it has been provided since it is considered quite a useful tool both for testing and trialling purposes.

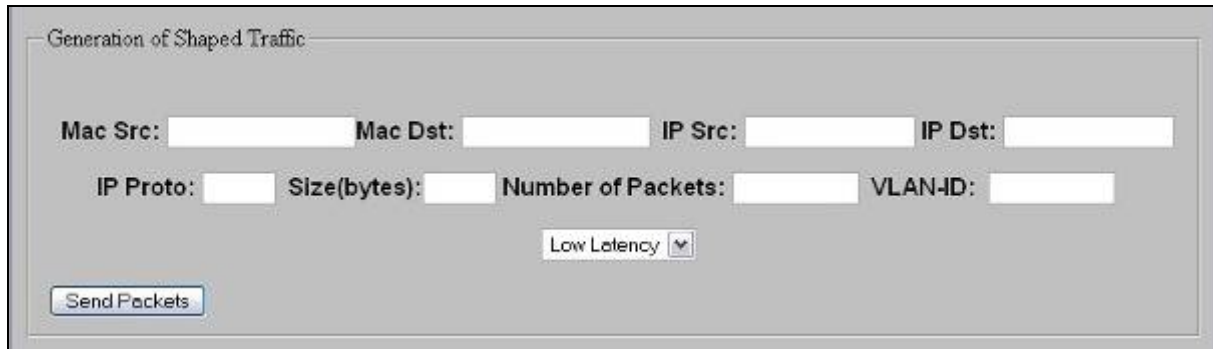


Figure 25. Traffic generation interface

- Source MAC: XX:XX:XX:XX:XX:XX text field
- Destination MAC: XX:XX:XX:XX:XX:XX text field
- Source IP: X.X.X.X text field
- Destination IP: X.X.X.X text field
- Protocol: X text field
- Datagram size (bytes) X text field
 - The data field will be automatically filled to as to fit the specified size.
 - No IP fragmentation is allowed so this field must be a number between 20 and 1482 (or 1486 in case no VLAN is used).
- Number of packets X text field
- VLAN-ID: X text field
 - In case no VLAN number is specified a normal Ethernet frame will be built.
 - Depending on this field, the frame will be sent towards the WAN side (in case a VLAN number is included) or towards the LAN side (in case no VLAN number is included).
- CoS:
 - This option allows the selection of the p-bits that will be used in the frame in case the VLAN tagging is used.
- 'Send Packets' button: once this button is pressed the RGW will begin to send the specified number of packets.