



IoRL Deliverable D6.1

Scenario Implementation

Editor:	John Cosmas, Brunel University
Deliverable nature:	Report (R) & Prototype (P)
Dissemination level: (Confidentiality)	Public (PU)
Contractual delivery date:	31 st November 2018
Actual delivery date:	10 May 2019
Suggested readers:	Indoor communication experts and providers of indoor communication services
Version:	1.0
Total number of pages:	83
Keywords:	Application Layer Software for 5G Services, Key Performance Indicator Requirements for each service, Bench top Demonstrator and Target Demonstrators in Environments Configurations

Abstract

This deliverable provides a description of the application layer software developed so far in the project for the implementation of the project scenarios and the Key Performance Indicators (KPIs) that each application requires from the IoRL system. Further application layer software may be developed during the rest of the project and will be reported in an updated version of this deliverable. It also provides the initial designs of the light systems that are required for each project scenario. Finally, it describes the equipment, system diagrams and costs of the laboratory test bench systems that will be constructed by each of the technology development partners and the target demonstrator scenario platforms. It explains what sharing of equipment is required between platforms to support each target demonstrator scenario platform.

Disclaimer

This document contains material, which is the copyright of certain IoRL consortium parties, and may not be reproduced or copied without permission.

All IoRL consortium parties have agreed to full publication of this document.

The commercial use of any information contained in this document may require a license from the proprietor of that information.

Neither the IoRL consortium as a whole, nor a certain part of the IoRL consortium warrant that the information contained in this document is capable of use, nor that use of the information is free from risk, accepting no liability for loss or damage suffered by any person using this information.

The EC flag in this document is owned by the European Commission and the 5G PPP logo is owned by the 5G PPP initiative. The use of the EC flag and the 5G PPP logo reflects that IoRL receives funding from the European Commission, integrated in its 5G PPP initiative. Apart from this, the European Commission and the 5G PPP initiative have no responsibility for the content of this document.

The research leading to these results has received funding from the European Union Horizon 2020 Programme under grant agreement number 761992 — IoRL — H2020-ICT-2016-2017/H2020-ICT-2016-2.

Impressum

Full project title: Internet of Radio Light

Short project title: IoRL

Number and title of work-package: WP6

Number and title of task: 6.1

Document title: Scenario Implementation

Editor: John Cosmas, Brunel University

Work-package leader: John Cosmas, Brunel University

Copyright notice

© 2019 Participants in IORL project

Executive summary

The H2020 Internet of Radio Light (IoRL) project develops a 5G compliant indoor networking solution integrating 5G radio with visible light communication (VLC) technology. It utilises the indoor lighting system as access points pervasively located within buildings to deliver increased throughput, reduce interference and electromagnetic field exposure and provide improved location accuracy. IoRL is scheduled to demonstrate its secure, customizable and intelligent indoor network in four different scenarios.

This deliverable provides a description of the application layer software developed for the implementation of the project scenarios and the Key Performance Indicators (KPIs) that each application requires from the IoRL system.

The document also describes the initial designs of the light systems that are required for the project scenarios.

Finally, it describes the equipment, system diagrams and costs of the laboratory test bench systems that will be constructed by each of the technology development partners and the target demonstrator scenario platforms. It explains what sharing of equipment is required between platforms to support each target demonstrator scenario platform.

List of authors

Company	Author	Contribution
Brunel University	John Cosmas, Ben Meunier, Nawar Jawad, Kareem Ali	1.0, 4.0 2.3.3, 2.3.4, 2.3.5, 2.4, 2.5, 3.10 2.7 2.2
Joda	Mathias Lacaud	2.3.1
NCSR	Charilaos Zarakovitis	2.3.2
Arcelik	Sibel Malkos	2.1, 2.3.6
CI3	Silvia de la Orden Rodriguez	2.6
HIT	Yoav Avinoam	3.1 – 3.9
MostlyTek	Moshe Ran	General Review, abbreviation list

Table of Contents

- Executive summary 3
- List of authors..... 4
- Table of Contents 5
- List of figures and tables 7
- Abbreviations 9
- 1 Introduction..... 12
- 2 Applications..... 13
 - 2.1 Ultra-HD TV Streaming application 13
 - 2.1.1 Technical requirements for UHD TV Streaming application13
 - 2.2 Indoor Location Based Data Access, Monitoring & Guiding and Interaction Applications..... 15
 - 2.2.1 Technical requirements for the applications15
 - 2.2.2 Indoor Location Based Data Access, Monitoring & Guiding & Interaction Applications (Key Building Blocks).....15
 - 2.2.3 Location Based Data Access Application.....16
 - 2.2.4 Monitoring & Guiding Application17
 - 2.2.5 Interaction Application.....19
 - 2.3 Streaming Applications..... 23
 - 2.3.1 Multiple-Source Streaming (MSS) over Multi-RATs.....23
 - 2.3.2 Transcoding process of media services under SDN/NFV deployment.....30
 - 2.3.3 360 Degree Streaming.....37
 - 2.3.4 Gaming and Virtual Reality Applications.....44
 - 2.3.5 Bike Game46
 - 2.3.6 UHD TV Streaming Applications.....49
 - 2.4 Virtual Reality Tourism 51
 - 2.4.1 Dystopian London51
 - 2.4.2 Globe theatre52
 - 2.5 Remote Tourism 52
 - 2.6 HoloLens inspection of Tunnels..... 53
 - 2.7 Smart TV Applications..... 54
 - 2.7.1 Indoor Follow-Me TV and Radio.....54
 - 2.7.2 Mockupancy61
 - 2.7.3 Room Multicasting61

- 2.7.4 TV Phone61
- 3 Light System Hardware 62
 - 3.1 Pendant Light Rose 62
 - 3.2 Ceiling Light..... 62
 - 3.3 Accessories 63
 - 3.4 Strip Light..... 64
 - 3.5 Pendant Strip Light and Pendant lights 65
 - 3.6 IP-65 Light 65
 - 3.7 Spot Light 65
 - 3.8 UE Trolley..... 65
 - 3.9 Backpack 65
 - 3.9.1 Considerations.....65
- 4 Platforms 66
 - 4.1 Introduction 66
 - 4.2 Laboratory Benchtop Platforms 67
 - 4.3 Demonstrator Scenario Platforms 73
- References..... 77
- Annex A Installation of the MS-Stream application 78
 - A.1. Browser compatibility..... 78
 - A.2. Using the VNF 78
 - A.3. Bare-metal installation with Docker 80

List of figures and tables

List of figures:

<i>Figure 2-1: Overview of TV application in IoRL system</i>	13
<i>Figure 2-2: IoRL Applications</i>	15
<i>Figure 2-3: Location Applications</i>	16
<i>Figure 2-4: Application fetching data from the database</i>	17
<i>Figure 2-5: Monitoring Application</i>	17
<i>Figure 2-6: Guiding Application</i>	18
<i>Figure 2-7: Code that has been used to find the state of the Wi-Fi and update it into the database</i>	18
<i>Figure 2-8: Location Based Data access Application</i>	19
<i>Figure 2-9: Uploading Image to application</i>	20
<i>Figure 2-10: Uploading Image onto Database</i>	20
<i>Figure 2-11: Creating a Treasure Hunt Game</i>	21
<i>Figure 2-12: A database of all the Treasure Hunt Games</i>	21
<i>Figure 2-13: Creating a Question for the Treasure Hunt Game</i>	21
<i>Figure 2-14: A Database of all the Questions that have been created for a Treasure Hunt Game</i>	22
<i>Figure 2-15: MSS for streaming from two servers</i>	23
<i>Figure 2-16: MSS for multi-path streaming through the IoRL network</i>	23
<i>Figure 2-17: Homepage of the application when logged in</i>	25
<i>Figure 2-18: Two steps of VoD Streaming</i>	26
<i>Figure 2-19: VoD upload form</i>	27
<i>Figure 2-20: Network address form for the video player</i>	27
<i>Figure 2-21: Expected result, the video is played</i>	28
<i>Figure 2-22: The player can be exported using the code at the bottom of the video</i>	28
<i>Figure 2-23: Live Streaming</i>	29
<i>Figure 2-24: Camera form</i>	29
<i>Figure 2-25: Live streaming form</i>	30
<i>Figure 2-26: Illustration of the layout of options of the IoRL transcoding VNF</i>	32
<i>Figure 2-27: Illustration of the IoRL transcoding VNF under the case of same input(s) and different parallel outputs</i>	33
<i>Figure 2-28: Illustration of the IoRL transcoding VNF under the case of same input(s) and duplicate outputs</i>	34
<i>Figure 2-29: Illustration of the topology of the experimental testbed</i>	36
<i>Figure 2-30: Illustration of the video quality (SSIM) over time with no transcoder VNF</i>	36
<i>Figure 2-31: Illustration of the impact of the IoRL transcoder VNF on video quality (SSIM) over time</i>	37
<i>Figure 2-32: Streaming Process</i>	37
<i>Figure 2-33: Mobile Tablet Streaming Process and Software</i>	38
<i>Figure 2-34: Theta V USB Connection</i>	38
<i>Figure 2-35: OBS Software showing the Media Input Selection</i>	39
<i>Figure 2-36: You Tube Live Streaming Events Page</i>	39
<i>Figure 2-37: OBS Stream Key Inputs</i>	40
<i>Figure 2-38: PC Operated VR Streaming Process</i>	41
<i>Figure 2-39: Streamshark Live Streaming Inputs Page</i>	42
<i>Figure 2-40: Streamshark Streamkey settings</i>	43
<i>Figure 2-41: Streamshark Management Page</i>	43
<i>Figure 2-42: Existing VR system overview</i>	44
<i>Figure 2-43: Location sensing protocol 1</i>	45
<i>Figure 2-44: Location sensing protocol 2</i>	45
<i>Figure 2-45: PC IoRL VR</i>	46
<i>Figure 2-46: VR Bike</i>	46
<i>Figure 2-47: Potentiometer on Steering</i>	47
<i>Figure 2-48: Dynamo Speed Sensor</i>	47
<i>Figure 2-49: Arduino Interface Circuit</i>	48
<i>Figure 2-50: Circuit diagram for ADC to Arduino Computer</i>	48
<i>Figure 2-51: UHD TV Streaming Application – Content Display</i>	50
<i>Figure 2-52: UHD TV Streaming Application – Multiple Contents Display</i>	50
<i>Figure 2-53: single central server streaming VR experiences</i>	51

<i>Figure 2-54: Screen Shots of the Dystopian London 3D World</i>	52
<i>Figure 2-55: Screen Shots of the Shakespeare’s Globe Theatre 3D World</i>	52
<i>Figure 2-56: Screen Shots of the 360 Degree Virtual Tourism</i>	53
<i>Figure 2-57: Remote tourism streaming process</i>	53
<i>Figure 2-58: Hololens Inspection Tunnel</i>	54
<i>Figure 2-59: Follow me architecture</i>	55
<i>Figure 2-60: Signalling sequence</i>	56
<i>Figure 2-61: FMA flowchart</i>	58
<i>Figure 2-62: Proxy server flowchart</i>	59
<i>Figure 2-63: FMS GUI1</i>	60
<i>Figure 2-64: FMS GUI2</i>	60
<i>Figure 3-1: Light rose</i>	62
<i>Figure 3-2: Ceiling Light</i>	63
<i>Figure 3-3: Accessory with LEDs</i>	64
<i>Figure 3-4: Accessory without LEDs</i>	64
<i>Figure 4-1: Benchtop Home Scenario</i>	67
<i>Figure 4-2: Benchtop Museum Scenario</i>	67
<i>Figure 4-3: Benchtop Train Station</i>	68
<i>Figure 4-4: Benchtop Supermarket Scenario</i>	68
<i>Figure 4-5: Demonstrator Home Scenario</i>	74
<i>Figure 4-6: Demonstrator Museum Scenario</i>	75
<i>Figure 4-7: Demonstrator Train Station</i>	75
<i>Figure 4-8: Demonstrator Supermarket Scenario</i>	76
<i>Figure A-1: Optimal requirements of the VNF</i>	79
<i>Figure A-2: Homepage of the application</i>	80

List of tables:

<i>Table 1-1: Overview of Performance Requirements of Applications</i>	12
<i>Table 1-2: Performance Requirements of Applications</i>	13
<i>Table 2-1: Data Rate Requirements for video content</i>	14
<i>Table 2-2: Data Rate Requirements for audio content</i>	14
<i>Table 4-1: Cost of Main Components</i>	66
<i>Table 4-2: Cost of Each System Configuration</i>	72
<i>Table 4-3: Cost of Each Laboratory Benchtop Systems</i>	72
<i>Table 4-4: Cost to each Project Component Supplier and Platform Provider</i>	72
<i>Table 4-5: Sharing Lab Benchtop Resources</i>	73
<i>Table 4-7: Light Systems Required for Each Demonstration Platform</i>	76
<i>Table 4-8: Timetable for Scenario Integration</i>	76

Abbreviations

5G	Fifth Generation (mobile/cellular networks)
5G PPP	5G Infrastructure Public Private Partnership
AR	Augmented Reality
CAPEX	Capital Expenditure
DRAN	Distributed Radio Access Network
EMF	Electro-Magnetic Field
FhG IIS	IoRL project partner Fraunhofer Institute for Integrated Circuits IIS
FMA	Follow Me Application
FMS	Follow Me Service
GUI	Graphical User Interface
GW	Gateway
IHIPGW	IoRL Home IP GW
IP	Internet Protocol
IoRL	Internet of Radio Light (project)
IoT	Internet of Things
KPI	Key Performance Indicator
L1	Layer 1
L2	Layer 2
L3	Layer 3
LBDA	Location Based Data Access
LMAG	Location Monitoring & Guiding
M2M	Machine to Machine
mmW	Millimeter Wave
MO	Mock Occupancy
MSS	Multi-Source Streaming
NFV	Network Function Virtualisation

OPEX	Operational Expenditure
OVS	Open Virtual Switch
PC	Personal Computer
QoE	Quality of Experience
QoS	Quality of Service
R&D	Research and Development
RAT	Radio Access Technology
RRLH	Remote Radio Light Head
RSS	Received Signal Strength
RunEL	IoRL project partner RunEL, Israel
SDN	Software Defined Networks
SFY	IoRL project partner Shanghai Feilo Yaming, China
SSIM	Structural Similarity Index
SSSC	Smart Shopping Cart
SWOT	Strengths, Weaknesses, Opportunities, and Threats analysis
TCP	Transmission Control Protocol
TDOA	Time Difference of Arrival
TH	IoRL project partner Tsinghua University, China
UBrunel	IoRL project partner Brunel University, UK
UDP	User Datagram Protocol
UE	User Equipment
UHD TV	Ultra-High Definition TV
ULeic	IoRL project partner University of Leicester, UK
USRP	Universal Software Radio Peripheral
VLC	Visible Light Communication
VNF	Virtual Network Function
VR	Virtual Reality
VoD	Video on Demand

WiFi Wireless Fidelity, is a family of radio technologies that is commonly used for the wireless local area networking (WLAN) of devices which is based around the IEEE 802.11 family of standards.

1 Introduction

The H2020 Internet of Radio Light (IoRL) project develops a 5G compliant indoor networking solution integrating 5G radio with visible light communication (VLC) technology. It utilises the indoor lighting system as access points pervasively located within buildings to deliver increased throughput, reduce interference and electromagnetic field exposure and provide improved location accuracy. IoRL is scheduled to demonstrate its secure, customizable and intelligent indoor network in four different scenarios.

This document reports the results of the activity responsible for implementing the higher layer software applications for the convergent services that will be used in validating and demonstrating the IoRL solution. The applications were selected in agreement with the hosts of the demonstrations.

The following **Table 1-1** summarises the key performance indicators of the IoRL indoor 5G network and the demand for them by the envisioned applications, whilst **Table 1-2** details further the performance requirements of the applications. It is assumed that the bitrate of an UHD TV 4k video (4:2:0 colour, 30 fps, 1% Av Compression) is 30 Mbps and HDTV Video (4:2:0 colour, 30 fps, 2% Av Compression) is 15Mbps. Communication and gaming services have low latency requirements (<10ms). Location Services require high location accuracy (<10cm) but there are some that require very high location accuracy (<1mm). Services that have been designed to be used in public places have high user density (1 devices/m²).

Table 1-1: Overview of Performance Requirements of Applications

No	Application	High Bit Rate (30Mbps)	Low Latency (<10ms)	High Loc Accuracy (<10cm)	Very High Location Accuracy (<1mm)	High User Density 0.1 devices/m ²
1	Indoor Location Based Data Access	√		√		√
2	Indoor Monitoring & Guiding			√		√
3	Indoor Interaction (Treasure Hunt Game & Chatting App)			√		√
4	Multi-Source Streaming over Multi-RATs	√				
5	Adapted Multi Source Streaming over Selected RATs	√				
6	UHD TV Streaming	√				
7	360 Degree Streaming	√	√			
8	Multiplayer Gaming	√	√		√	
9	Virtual Reality Tourism	√	√		√	
10	Indoor Follow-Me TV	√		√		
11	Mockupancy	√				
12	Room Multicasting	√				
13	TV Phone	√	√			

Table 1-2: Performance Requirements of Applications

UC Number	Application	Use case Key Performance Indicators Required																	
		1000x capacity / room (M bits/s)	Latency (ms)	Location sensing (cm)	Energy savings (%)	Battery lifetime (hours)	Typical user data rate (M bits/s)	Ubiquitous coverage (No of rooms in building)	Lowered EMF levels (%)	Mobility (m/s)	Reliability % Success Rate within 1 ms	Range from RRLH (m)	Deployment Time (s)	Security	Service Continuity (%)	QoS/QoE (Absolute Category Rating 1-5)	Jitter (µs)	Survival Time (ms)	Connection Density (devices / m ²)
1	Indoor Location Based Data Access	700	100	10	90	120	30	2	90	1	99.99	10	1-10		99.99	5	1	100	1
2	Indoor Monitoring & Guiding	700	100	10	90	120	1	2	90	1	99.99	10	1-10		99.99	5	1	100	1
3	Indoor Interaction	700	100	10	90	120	5	2	90	1	99.99	10	1-10		99.99	5	1	100	1
4	Multi-Source Streaming over Multi-RATs	700	100	NA	90	120	30	1	90	NA	99.99	10	1-10		99.99	5	1	100	0.1
5	Adapted Multi Source Streaming over Selected RATs	700	100	NA	90	120	30	1	90	NA	99.99	10	1-10		99.99	5	1	100	0.1
6	UHDTV Streaming	700	100	NA	90	120	30	1	90	0	99.99	10	1-10		99.99	5	1	100	0.1
7	360 Degree Streaming	700	10	NA	90	120	30	1	90	0	99.99	10	1-10		99.99	5	1	100	0.25
8	Multiplayer Gaming	700	10	0.02	90	120	30	1	90	0	99.99	10	1-10		99.99	5	1	100	0.1
9	Virtual Reality Tourism	700	10	0.02	90	120	30	1	90	0	99.99	10	1-10		99.99	5	1	100	0.1
10	Indoor Follow-Me TV	700	100	10	90	120	30	2	90	0	99.99	10	1-10		99.99	5	1	100	0.25
11	Mockupancy	700	NA	NA	90	120	30	2	90	1	99.99	10	1-10		99.99	5	1	100	0.1
12	Room Multicasting	700	100	NA	90	120	30	2	90	NA	99.99	10	1-10		99.99	5	1	100	0.1
13	TV Phone	700	10	NA	90	120	30	1	90	NA	99.99	10	1-10		99.99	5	1	100	0.1

2 Applications

2.1 Ultra-HD TV Streaming application

2.1.1 Technical requirements for UHD TV Streaming application

The main requirements for the UHD TV streaming application are:

- the ability to develop an application running on an UHD TV set, or the support of standard web browsers by TV set;
- the support of IP Camera standard protocol (usually RSTP/RTP over UDP/IP or RTMP over TCP/IP) packets for the live streaming with delay constraints from live camera.
- The ability to communicate with IoRL VLC dongle via 100 Mb Ethernet port on TV set.

UHD TV Streaming Application is developed to display high quality video stream such as H.264 or H.265 codec on the TV screen. Delay for a live stream should be minimized for the users. Therefore, TV application should be able to receive UDP packets over TCP/IP protocol for displaying live streaming.

System overview for the UHD TV streaming application is shown in **Figure 2-1**. On the SDN network streaming video from server or IP camera is transmitted to the RRLH controller. Controller transfers packets of the stream to the respective remote radio light head(s). IoRL dongle receives emitted signals by RRLH and transfers to TV set. TV receives the network packets over 100 Mbit Ethernet port on it. TV application handles UDP packets and displays video stream on video player.

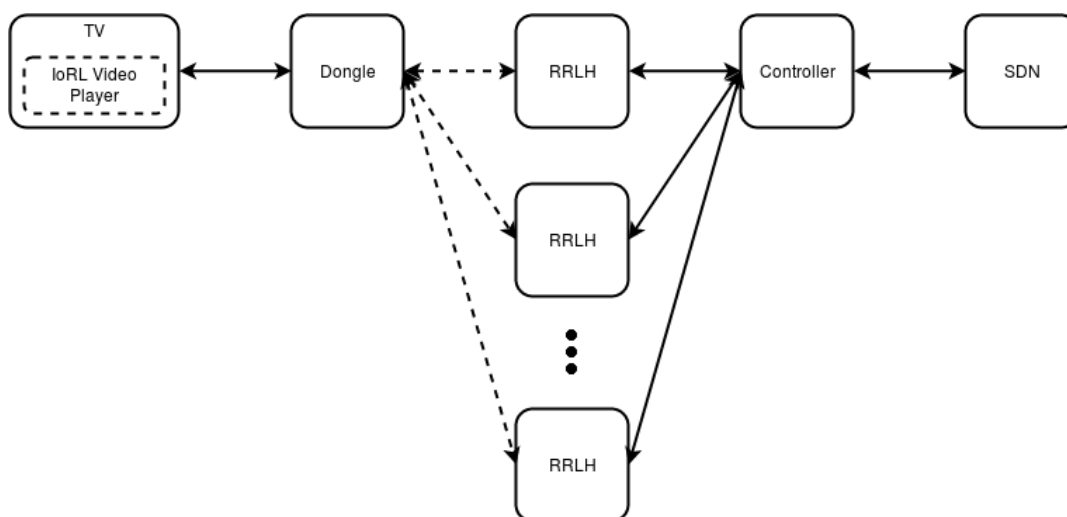


Figure 2-1: Overview of TV application in IoRL system

The content delivered by the server to be displayed on the UHD TV set shall be in MPEG TS format. The supported data rates for audio and video contents that will be used in the application on the TV set are specified in the following tables.

Table 2-1: Data Rate Requirements for video content

Resolution	Compression	Frame Rate (fps)	Local Bandwidth (MB/s)	Network Bandwidth (Mb/s)
UHD 4K(3840 x 2160)	H.265 HEVC	60	3.4	26.8
UHD 4K(3840 x 2160)	H.265 HEVC	50	2.8	22.3
UHD 4K(3840 x 2160)	H.264	60	4.5	36.2
UHD 4K(3840 x 2160)	H.264	50	3.8	30.1
Full HD(1920 x 1080)	H.265 HEVC	60	0.8	6.7
Full HD(1920 x 1080)	H.264	60	1.1	9

Table 2-2: Data Rate Requirements for audio content

Audio Codec	Sample Rate	Channel	Bit Rate
AAC-LC, HEAAC	8KHz ~ 48KHz	Up to 5.1	
DTS	Up to 48KHz	Up to 5.1	< 1.5 Mbps
LPCM	8KHz ~ 48KHz	Up to 5.1	64 Kbps ~ 1.5 Mbps
IMA-ADPCM MS-ADPCM	8KHz ~ 48KHz	Up to 2	384 Kbps
DTS LBR	12KHz, 22KHz 24KHz, 44.1KHz 48KHz	Up to 5.1	Up to 2 Mbps
DTS XLL	Up to 96 KHz	Up to 6	
DRA	8KHz ~ 96KHz	Up to 7.1	< 1533 Kbps
AC3	32KHz,44.1KHz, 48KHz	Up to 5.1	32 Kbps ~ 640 Kbps
EAC3	32KHz,44.1KHz, 48KHz	Up to 5.1	32 Kbps ~ 6 Mbps
MPEG1/2 Layer3	16KHz ~ 48KHz	Up to 2	8Kbps ~ 320 Kbps

2.2 Indoor Location Based Data Access, Monitoring & Guiding and Interaction Applications

2.2.1 Technical requirements for the applications

2.2.1.1 The Location application requirement

- A database that holds information about current exhibits.
- Allows the merchandiser to upload new and update existing Internet pages concerning products.
- Download information from the database.
- Need to be able to update the information in the database by editing, deleting and adding new items.

2.2.1.2 The Monitoring & Guiding application requirement

- A database that holds the information.
- The behavior database that records each user's behavior.
- The behavior database should be able to visualize behavior of individual and group of users.

2.2.1.3 The interaction application requirement

- An Internet interface for the Game Master to write the clue and record the correct photo.
- A database that stores photos taken by the game master and the children.
- Visual comparison software to check the matching of the photos taken by the children.
- A server that allows online searching for relevant products' location at the shop.

2.2.2 Indoor Location Based Data Access, Monitoring & Guiding & Interaction Applications (Key Building Blocks)

IoRL demonstrates its technology in four scenarios – (i) at home; (ii) in a museum; (iii) at a train station and in tunnels; and (iv) in a supermarket. The service requirements for these widely different indoor environments would seem to be significantly different but on close analysis there are three main applications – (i) indoor Location Based Data Access (LBDA); (ii) indoor Location Monitoring & Guiding (LMAG); and (iii) Interaction Applications – that are required for each of the four scenarios. Since these scenarios are quite different, they might require different look and feel from the applications, but the applications should essentially perform the same three common functionalities.

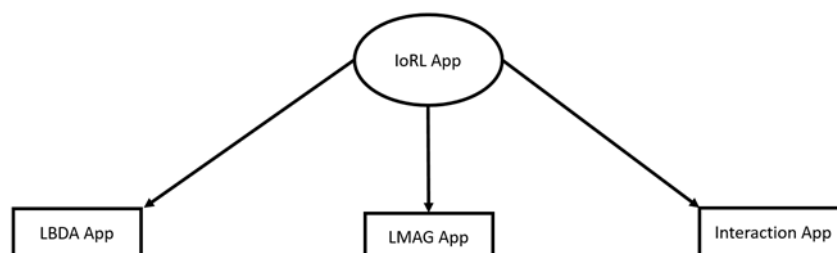


Figure 2-2: IoRL Applications

These novel applications for radio-light communication at the Application Layer and Network Layer are unique to Radio-Light technology. The applications are created and developed to not only meet the scenarios requirements but also to be user friendly.

2.2.3 Location Based Data Access Application

A location-based system would allow for the Museum to deliver data specific to the artefact on display. For a supermarket it would allow data to be delivered regarding the goods on the shelves immediately in front of the shopper. For a train station it would allow data to be delivered to a traveller on entry to the station and on the platform. For a conference it would allow the user to obtain the agenda and proceedings on entry to the conference lobby. For homes it will allow selective control of lights and sensors depending on who is in the room.



Figure 2-3: Location Applications

The Location Scenarios Summary:

- ✓ **Museum:** that detects when people are in the vicinity of a VLC RRLH.
- ✓ **Tunnel:** keeps a record of all the IP addresses of passengers that have bought tickets on line and when they are detected in the railway tunnel environment downloads all information that they might require such as a map with directions to their platform, places where they could relax whilst waiting for their train to arrive etc.
- ✓ **Supermarket:** An Internet page that is associated to VLC light access point that will be downloaded to shoppers' Tablet PCs when they are in the vicinity of the VLC RRLH.
- ✓ **Conference:** Announcement app that allows conference organisers to make announcements to speakers and delegates.

2.2.3.1 Location Based Data Access Application specifics

All the users can be detected in all the scenarios. We are only waiting to test it using the VLC as soon as it is ready. The application is able to view exhibits from the database. The database has been designed using the Django architecture. The application requires the use of a database and an interface that will interact with the VLC access points and the database. Essentially the VLC access points would from now on lookup the data from the Database,

and the application developed would be able to access and edit the information in this database, making it easier to manage and control the application.

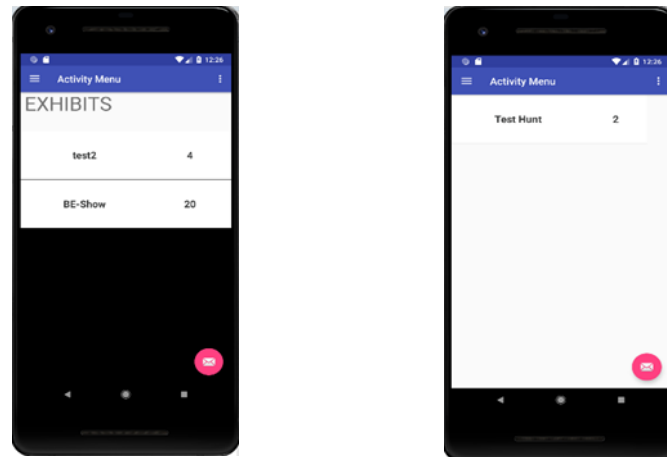


Figure 2-4: Application fetching data from the database

2.2.4 Monitoring & Guiding Application

Indoor location monitoring would allow the museum to record the behaviour of visitors. This in turn will benefit the below scenarios:

- The visitor's path through the museum;
- The amount of time each visitor spent at each exhibit;
- The most visited exhibit by visitors;
- The density of visitors in the museum at any one time.

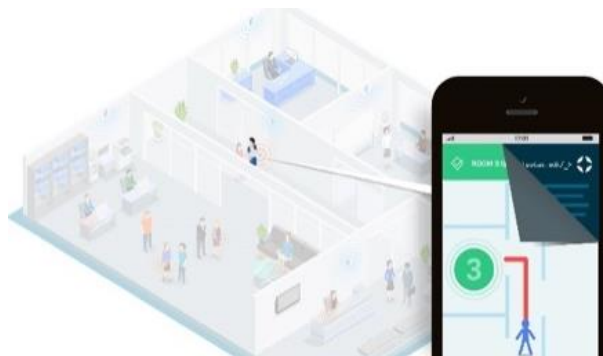


Figure 2-5: Monitoring Application

Indoor location guiding would allow the visitors with a floor plan to navigate their way around easily.

For the train station and tunnel, it would allow the services manager in the control office to monitor and locate the maintenance workers that are performing activities in the train station tunnel and guide the maintenance staff to the specific location of the maintenance activity.

In the supermarket this application would allow the shopper to find its way easily to the products according to its own shopping list.

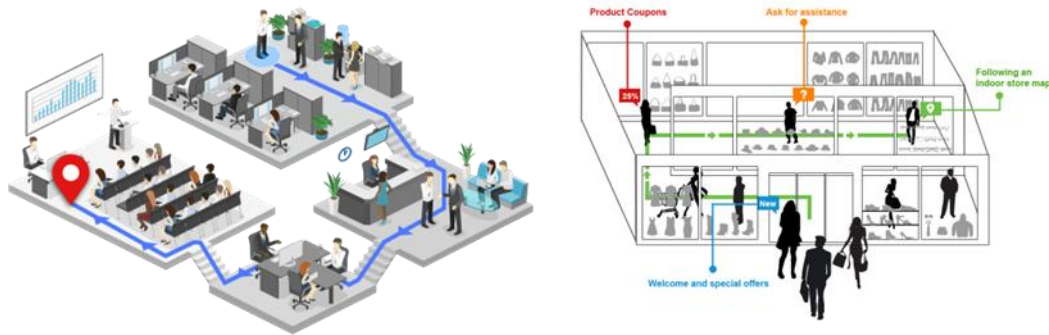


Figure 2-6: Guiding Application

Figure 2-6 above shows an example for indoor location guiding as a person is being guided to its chosen destination. As well as looking at the green line, where the customer is following the map inside the store as his movement is being monitored.

The Monitoring and Guiding Scenarios Summary:

- ✓ **Museum:** An Application with the floor plan on the visitor's electronic device (tablet, phone) that records their behaviour and feeds back the data to the database.
- ✓ **Tunnel:** Control application of map of building coverage area with locations and identities of specified set of users within the building; using the SDN tracking to obtain the location. Additionally, to be able to have a conference call Co-working (e.g. Webex) of various different types of data to maintenance workers.
- ✓ **Supermarket:** A Smart Shopping Cart server that imports a shopping list from a customer's smart phone to guide shopper through store. A Learning curve per customer which allows route calculation and suggests relevant product content (promotions, reviews and feedbacks). SSC can compare using SSC-mobile connection prices with other supermarkets in the area.

2.2.4.1 Monitoring and Guiding Application specifics

The application has been developed that can detect when people are in the vicinity of the access point, the Remote Radio Light Head. This has been simulated using Wi-Fi, where every 1 second an update of the location of the WiFi router IP address is recorded on the database. The code for this is shown in **Figure 2-7**.

```
WifiManager wifiManager = (WifiManager) context.getSystemService(Context.WIFI_SERVICE);
WifiInfo wifiInfo = wifiManager.getConnectionInfo();
if (WifiInfo.getDetailedStateOf(wifiInfo.getSupplicantState()) == NetworkInfo.DetailedState.CONNECTED) {
String ssid = wifiInfo.getSSID();
}
```

Figure 2-7: Code that has been used to find the state of the Wi-Fi and update it into the database

In the next step the monitoring and guiding can be simulated on mini net using different virtual machines to test out the guiding inside the museum. Mini net allows creating a

topology to test the system using Iperf, which is a simple test, as it will show the service performance using (VLC and Wi-Fi).

2.2.5 Interaction Application

Interaction would allow the children to play a game “Treasure Hunt” inside the museum, where it provides the children with clues to locate a picture within the museum. The supermarket would provide the customers with an application through a secure IoRL link “Smart Shopping Cart” (SSC) that allows them to prepare the shopping list and find relevant products. In a conference room, the application would allow the delegates to discuss questions and answers, as well as, rating the speakers and the conference.



Figure 2-8: Location Based Data access Application

Interaction includes three different scenarios, where all can be implemented based on the interactions of each of the user, as shown above. The lady going into a store and she is already receiving information of special offers. Also, looking at the other customers inside the store each one of them is getting updates from the application. Moreover, scanning items inside the supermarket sees information regarding the different products.

The Interaction Scenarios Summary

- ✓ **Museum:** An App that allows children to upload their photos to the database.
- ✓ **Supermarket:** A shopping list app that allows customer to compile a shopping list on his/her smart phone.
- ✓ **Conference:** Chat app allowing delegates to communicate with each other. Rating app that allows delegates to rate speakers. Conference rating app that allows delegates and speakers to rate the conference.

2.2.5.1 Interaction (Treasure Hunt) Application specifics

The treasure hunt game has been created and tested, and is being finalized by adding the image comparison between the reference image stored on the database the game curator and the image uploaded by the game players.

The following figures are the testing for uploading an image to the database (**Figure 2-9**) after capturing it for the treasure hunt game by the game players on their smart phone app.

The image, which has been uploaded on the database by game players can be seen by the game curator as shown in **Figure 2-10**.

However, right now the work in progress is on creating a template with a white border, which uses cross correlation to check if the image uploaded on the database is the correct one or not, using a high cross correlation match or low cross correlation match.

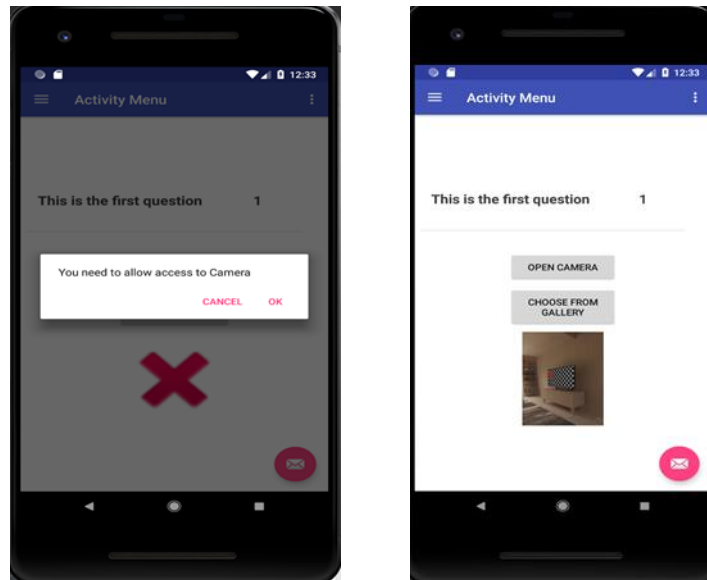


Figure 2-9: Uploading Image to application

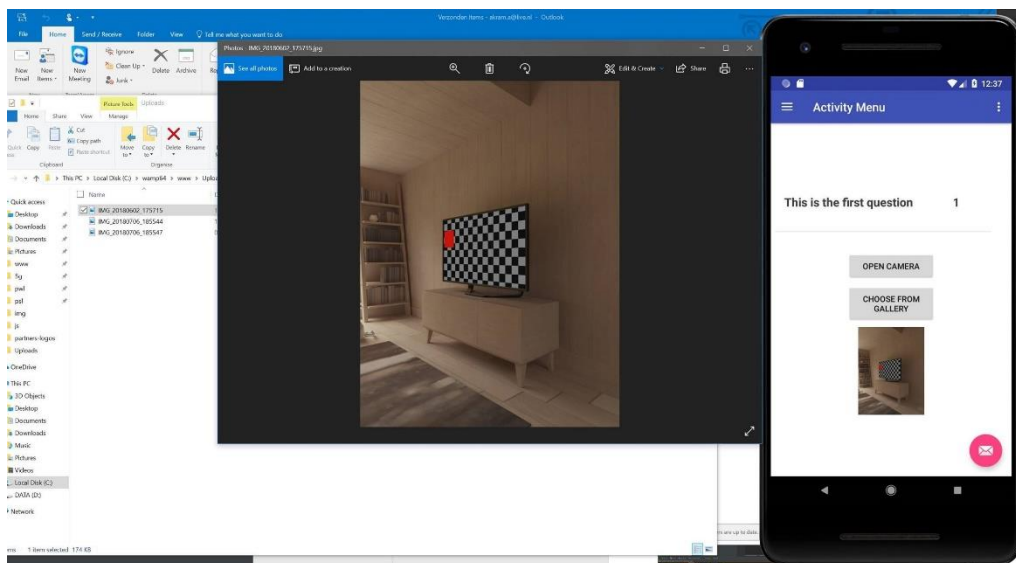


Figure 2-10: Uploading Image onto Database

A database has been created specifically for the treasure hunt game by the game curator. As it allows the manager to upload different treasure hunt games with different images and questions. The curator can create a treasure hunt game as shown in **Figure 2-11** and then view all the games that (s)he has created by viewing the database of all Treasure hunt games as shown in **Figure 2-12**.

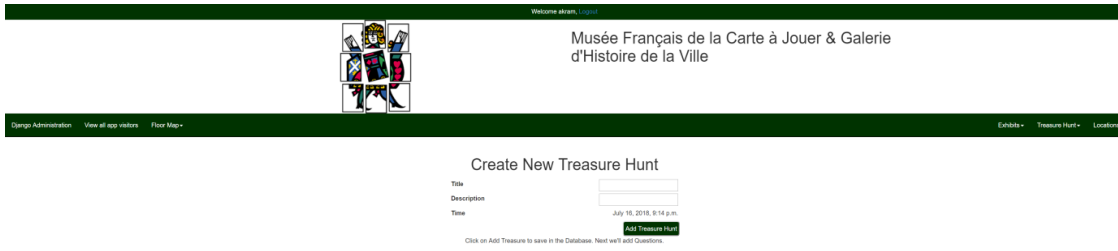


Figure 2-11: Creating a Treasure Hunt Game



Figure 2-12: A database of all the Treasure Hunt Games

Once the curator has created a game (s)he can the create all the questions to help guide the game players to find the right card in the museum as shown in **Figure 2-13** and store and view them onto the game database as shown in **Figure 2-14**.



Figure 2-13: Creating a Question for the Treasure Hunt Game



Figure 2-14: A Database of all the Questions that have been created for a Treasure Hunt Game

2.2.5.2 Interaction (Chatting) Application

The chatting application has been in progress. The chat works as it connects people in the same area at the place but the ratings of the speakers is to be added.

2.3 Streaming Applications

2.3.1 Multiple-Source Streaming (MSS) over Multi-RATs

The UC discussed in this section are the ones defined in the document **IoRL Deliverable 2.1 [IoRLD2.1]**.

2.3.1.1 MSS application in the context of IoRL

In the IoRL architecture the video streaming can be managed in two different ways. Firstly, the video can be streamed from the server to the end user using the IoRL system as a black box providing internet access. This first possibility is essential to allow the compatibility with actual streaming services over the internet, and especially video communication tools (UC 1.4, 3.6, 4.2 and 5.4). Secondly, the video can be streamed to the end user using **MSS**.

MSS adds reliability at the application level for the system by streaming sub-flows of video data from different sources or through different paths. Those sub-flows can be read independently, giving a lower video quality, or can be merged, giving a higher video quality.

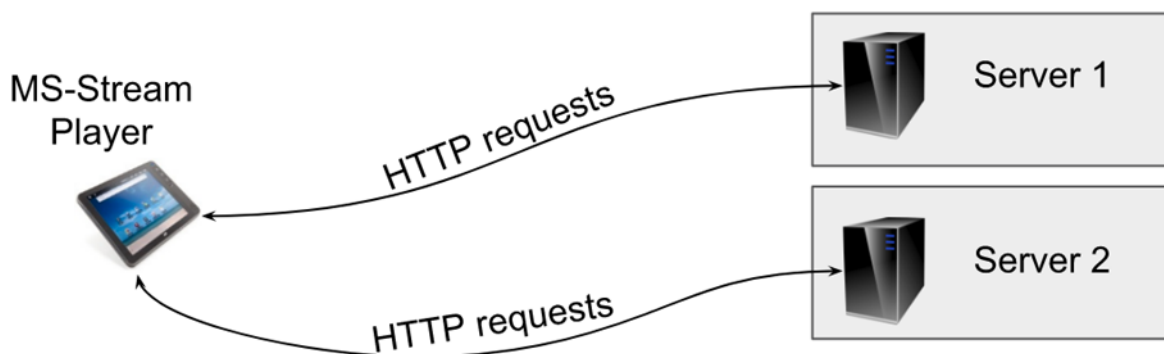


Figure 2-15: MSS for streaming from two servers

In the IoRL system MSS will be used in several use cases to route the video data in high quality through the Radio Light network and in low quality through an alternative network (for example: WiFi) access, providing reliability in case of an interruption in the Radio Light network.

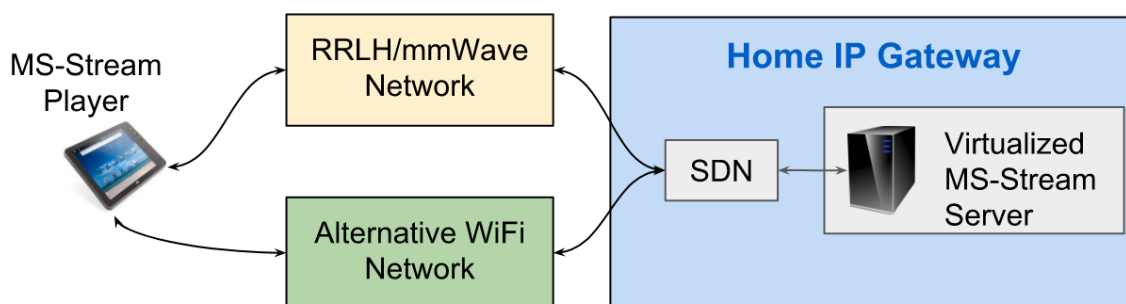


Figure 2-16: MSS for multi-path streaming through the IoRL network

MSS over remote radio light head is composed of three main modules, relying on the HTTP/TCP protocol for communication: the **MSS Server**, the **MSS Player** and the **MSS**

Transcoder. Both the MSS Server and the Transcoder are part of the server side and can be deployed in the Intelligent Home IP Gateway as virtual functions. The MSS Player is to be deployed at the client side, inside of an application running in the UE.

On the client side, the MSS Player is a video player integrated in a web page and accessed through a web browser. This client may also be running in a native application for specific UE. As the algorithm of MSS is defined as client-centric, the MSS Client is responsible for the creation of the HTTP requests sent to the MSS Server through the RRLH Network and the alternative network access (for example: a WiFi network access) for a Multiple-Source Streaming session. The adaptation algorithm is implemented in the MSS client.

On the server side, the MSS Server is an application that can answer client requests for specific video contents. This module is responsible for the creation of video segments adapted to Multiple-Source Streaming. The video segments are created from video data transcoded in numerous different qualities. For that purpose, the Transcoder comes along with the server. This Transcoder is a module that can transcode input video data into Multiple-Source-ready video data in one or several qualities. Those data then can be pushed to the MSS Server and be available for the MSS Player in the UE.

In the UC of IoRL, MSS will be used in two main scenarios: routing video data from a server (UC 1.3, 3.8, 4.5) and routing video data from a live camera (UC 1.5, 2.4, 5.2).

In the first scenario, the full video content is stored on storage servers and can be pre-transcoded into lower qualities. Video can be accessed on the user side through a native or a web application. The client then will start sending HTTP/TCP requests through the Radio Light network and the WiFi network to the MSS Server to request already existing video segments. The video segments will be downloaded and displayed on the MSS client.

In live streaming from a camera scenario, the video data are streamed from the cameras through the SDN of the Home IP Gateway and can be redirected to the Transcoder. At this point, the stream is transcoded into several level of lower qualities and sent to the MSS Server. Then, the live client would be able to get the Multiple-Source-adapted live video data through both the RRLH network and/or the WiFi network, and be able to display it to the end user.

2.3.1.2 MSS application functionalities

2.3.1.2.1 List of features

The MSS application can:

- Transcode video files;
- Transcode live streams from IP camera via RTSP/RTMP;
- Serve video segments for VoD and live streaming;
- Serve MSS video segments from multiple-network;
- Serve 360° video segments.

For that, the application uses the following ports:

- a **web application on port 80;**
- a **RTMP server on port 1935.**

Moreover, the MSS player available in the application can be exported into another web application in one line of HTML using an iframe.

2.3.1.2.2 Architecture of the application

The MS-Stream application is composed of:

- A web server delivering the web application, the video data and exposing an API to encode both live and VoD content;
- A MongoDB database;
- A Nginx RTMP server;
- A Nginx reverse-proxy redirecting the requests to the web server.

2.3.1.2.3 Browser compatibility

The web application is developed with ECMAScript 5 which is supported by all of the modern browsers. The complete list of compatibility can be found at [ECMAS].

The MSS player itself uses the Media Source Extension which is also supported by modern browsers, except on iOS device. The complete list of compatibility can be found at [MSE]. On iOS devices, the MSS player is not used and replaced by a traditional HLS player without multi-source streaming.

2.3.1.2.4 Login

First of all, the user should go to the login page in order to login. The application is configured with one admin user defined during the installation process (see the installation instructions in the Appendix).

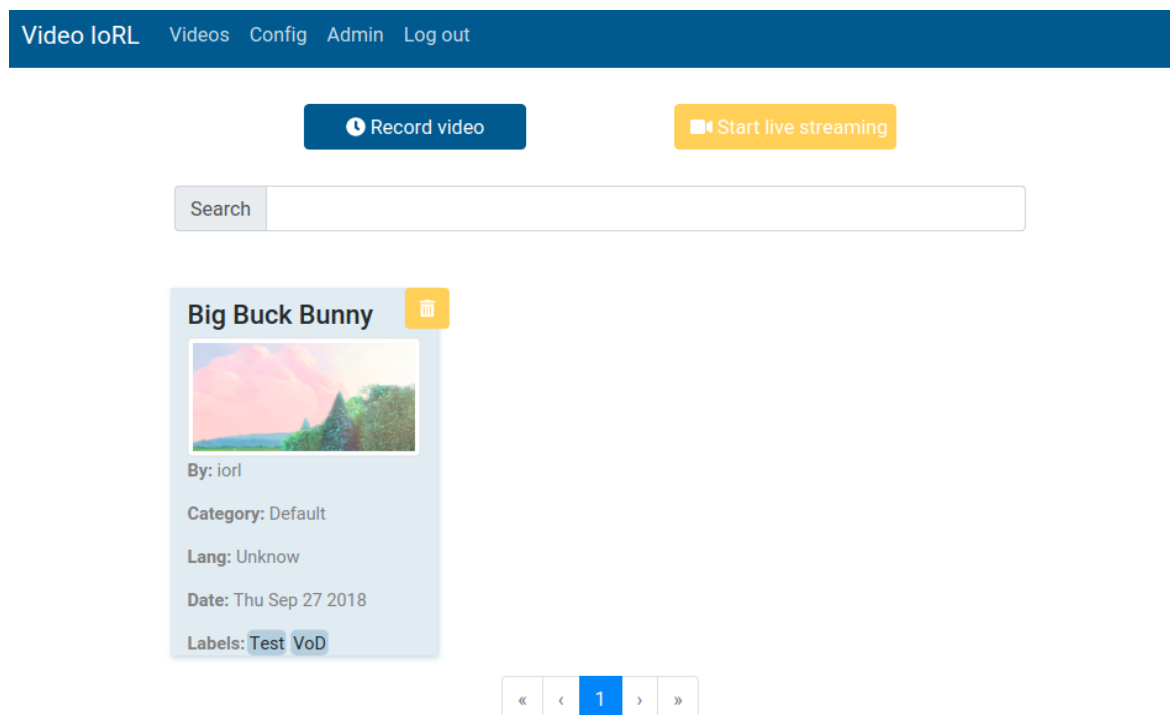


Figure 2-17: Homepage of the application when logged in

2.3.1.2.5 VoD Transcoding and Streaming (UC 1.3, 3.8, 4.5)

The management of VoD is done in two steps. The MSS Transcoder will be used in a first step to transcode the videos into MSS-compatible content. Then, in a second step, a user will be able to watch a video on a user equipment (UE) by sending HTTP requests to the MSS Server through the different networks available.

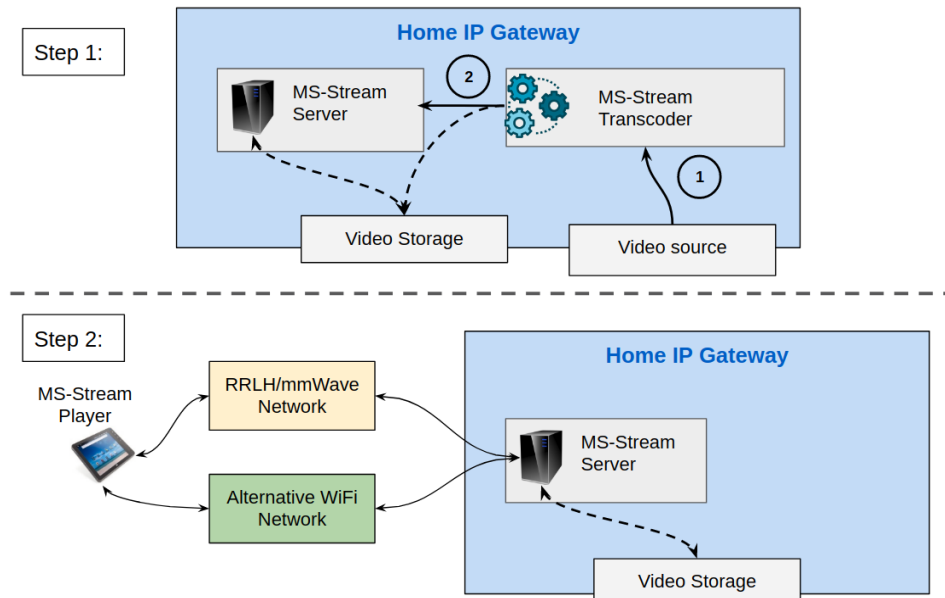


Figure 2-18: Two steps of VoD Streaming

To upload a video, first has to login and then go to the homepage. Two buttons are available on the top: “Record video” for VoD content and “Start Live streaming” for Live contents. Select “Record Video” and then “Upload” and fill out the form.

Video IoRL Videos Config Admin Log out

Video recoding / uploading

Camera Type:
 IP Camera Upload

Title:

File:

Category:

The categories can be configured in the 'Config' page.

Labels:

Date of the event:

Language:

Description:

Add a duration limit

Figure 2-19: VoD upload form

Once done, go back to the homepage. After some time, the video should be encoded and ready to be played. Click on the video to go to the player webpage. In the player page, a small form should be available. Fill out this form with the address of the VNF from several network (for example, ethernet and Wi-Fi) or leave it like it is by default. Click on play to start the video. If the video plays, than the VNF is working.

Video IoRL Videos Config Admin Log out

Big Buck Bunny

Address via Network 1:

Address via Network 2:


Figure 2-20: Network address form for the video player

Video IoRL Videos Config Admin Log out

Big Buck Bunny

Address via Network 1: <http://192.168.1.2>
Address via Network 2: <http://192.168.1.2>

Play



By: iorl

Remove video

Figure 2-21: Expected result, the video is played

At the bottom of the video webpage, a HTML code is given. This code can be copied and pasted to another web site or web application (for example the website of the museum, the web site of the commercial center, etc.) in order to export both the video and the MSS player with the embedded algorithms.

Export

Copy this code and put it in your html page to export a responsive player for this video

Copy the code to clipboard

```
<div style="position:relative;overflow:hidden;padding-top:65.25%;"><iframe id="video-player" style="position:absolute;top:0;left:0;width:100%;height:100%;border:0;" allowfullscreen="true" webkitallowfullscreen="true" mozallowfullscreen="true" frameBorder="0" scrolling="yes" src="http://localhost:10080/embedded/embedded.html?uuid=2d578a98-b1dd-4ecc-b0d2-b32f5b0c51b8"/></div>
```

Figure 2-22: The player can be exported using the code at the bottom of the video

2.3.1.2.6 Live Transcoding and Streaming (UC 1.5, 2.4, 5.2)

For Live Streaming, the MSS Transcoder will be used to transcode the video stream of IP camera into MSS-compatible live content and to save them into the video storage. Then, a user will be able to watch a video in live on a UE by sending HTTP requests to the MSS Server through the different networks available.

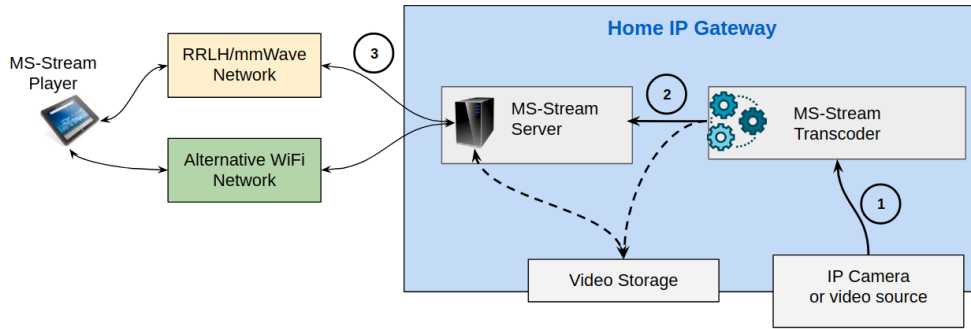


Figure 2-23: Live Streaming

The first step to upload a live stream is to register the address of the camera. This can be done in the “Admin” page.

Figure 2-24: Camera form

Once the camera is registered, go to the homepage and click on “Start live stream” at the top of the page. Then, fill out the form and select the appropriate camera in the list.

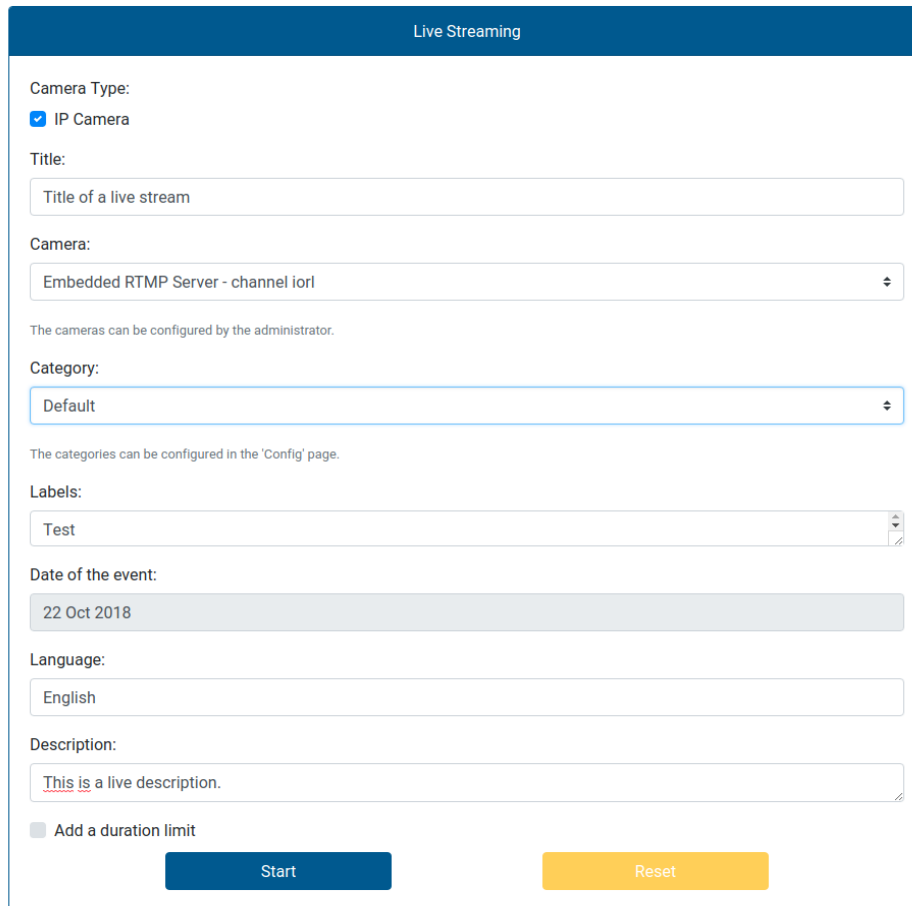


Figure 2-25: Live streaming form

After a few seconds, the video will be available in the homepage. The process to watch the video is the same for VoD content and live content.

2.3.1.2.7 360° Streaming

The MSS application supports 360° video streaming via **equirectangular projection** by using videojs [VJS] and the plug-in videojs-vr [VJS-VR]. To upload a 360° video, do as for a simple video but add a “VR360” in the first position of the labels in the upload form. The 360° videos are encoded with specific bitrate and resolutions in order to provide the best quality of experience for the user.

When the encoding process is done, the video can be watched in a webvr compatible browser [WEBVR] like for standard VoD or live video.

2.3.2 Transcoding process of media services under SDN/NFV deployment

2.3.2.1 The role of the transcoding process in the IoRL home network

The previous Section 3.3.1 described the multiple-source streaming VNF of the IoRL home network. In this section we will approach the process of transcoding and explain how this process impacts the video streaming. The transcoding VNF covers three main tasks of digital media.

First, transcoding at a high level can be used to take already-compressed or encoded video content, decompress and decode it, and, in turn, alter and recompress the video content. For example, it can change the audio and/or video format (codec) from one to another, such as converting from an MPEG2 source (commonly used in broadcast television) to H.264 video and AAC audio (the most popular codecs for streaming), adding watermarks, logos or other graphics to videos, etc.

Second, transcoding allows to change the bitrate of videos such as taking a 4K video input stream at 10 Mbps and converting it into one or more lower-bitrate streams (also known as renditions), HD at 6 Mbps, or other renditions at 3 Mbps, 1.8 Mbps, 1 Mbps, 600 kbps etc.

Third, transcoding can be considered to resize (or trans-size) video frames. For example, change the video size from a resolution of 3840 pixels × 2160 (4K UHD) down to 1920×1080 (1080p) or 1280×720 (720p).

In this regard, the process of transcoding refers to any combination of the above tasks. Recall that video conversion is – typically – computationally intensive and therefore, transcoding usually requires powerful hardware resources, including large storage, fast computing or graphics acceleration capabilities, and so on. In addition, note that transcoding should not be confused with the process of “transmuxing” – also referred to as repackaging, packetising or rewrapping – which requires considerably smaller computational and/or storage overhead than for transcoding. For example, let us consider that we have a video content of H.264/AAC format, and by changing the container this content has been packaged in, we deliver it either as HTTP Live Streaming (HLS), Smooth Streaming, HTTP Dynamic Streaming (HDS) or Dynamic Adaptive Streaming over HTTP (DASH). In such situation, we have “transmuxed” the compressed audio and video by (re)packaging it into different delivery formats but without changing the actual audio or video content, as done in transcoding.

The criticalness of transcoding for the IoRL home network is because such process can enable the video contents to reach more than one end-user in the most effective manner. For example, let us consider that a home user requires to capture webcam audio and video content using browser-based desktop application (e.g. Adobe Flash, which generates 1080p H.264 video and Speex audio content), and that this content has to be delivered to multiple other home users (i.e. viewers). If transcoding process is not utilised, the only way to send this content is to stream it directly from the source user to the destination viewers, which gives rise to two main issues. First, viewers with no sufficient bandwidth will not be able to view the stream because their media players will be buffering constantly as they wait for packets of that 1080p video to arrive. Second, in order to render the Speex audio, it is most likely most viewers to watch the content using Flash Player, which is computationally intensive. Therefore, in such cases, the network will collectively exclude almost all viewers with slower data speeds, tablets, mobile phones and connected TV devices than others, which is not preferable in the IoRL paradigm. Instead, IoRL will use a transcoding VNF to simultaneously create a set of time-aligned video streams, each with a different bitrate and frame size, while converting the Speex audio to AAC audio format. Such set of Internet-friendly streams could then be packaged into several adaptive streaming formats (e.g. HLS), which in turn, can be effectively transferred to any IoRL home user (or any user through the Internet). For example, let us consider an IoRL home user that performs live broadcast using a camera and encoder, or an IP camera. In order this user to attain broadcasting its content to the largest number of online viewers with the best possible quality allowed by their

bandwidth and devices, the IoRL needs to support transcoding with adaptive streaming jointly. That is, to deliver HD H.264/AAC streams to the transcoder (typically located on a server image in the cloud), which, is to create multiple H.264/AAC renditions at different bitrates and resolutions, where these renditions will be packaged into a media server (which can be the same server as the transcoder) into one or more adaptive streaming formats before being delivered to the home users.

In that way, IoRL can provide robust live transcoding software capabilities to power any workflow and take the full control of these workflows by deploying downloadable software onto home or cloud servers. In addition, an IoRL user will be able to fully manage live-streaming contents that can be transcoded and delivered not only locally (i.e. in the home) but also globally to audiences of any size.

2.3.2.2 Development of the transcoding VNF in the SDN/NFV environment of the IoRL home network

Having defined the role of the transcoding process in the IoRL home network, this sub-Section describes the methodology to develop the process by means of VNF using SDN and NFV technologies. By recalling the resource-intensive tasks of transcoding due to decoding, computing and coding of the source streams, the SDN/NFV solution can improve the QoE offered to the destination users compared to legacy approach (i.e. hardware transcoders) by instantiating the process’ tasks in virtualised environments, which can move the whole processing of tasks from the standalone devices of end-users to data centres and distribute the content through faster routes in the transport network (e.g. choose gigabit instead fast-ethernet switches when the streaming demands increase).

The proposed transcoding VNF uses the `flussonic-ffmpeg` package, which can be installed in the OpenStack repository of the SDN/NFV environment using the below command.

```
apt-get -y install flussonic-ffmpeg
```

Note that ffmpeg transcoder can be set as a text string either in the configuration file `/etc/flussonic/flussonic.conf` or in the administrative interface. To activate the transcoder, it is first important to specify all video parameters, second the global options, and third the audio options as, e.g., shown in the layout of options illustrated in

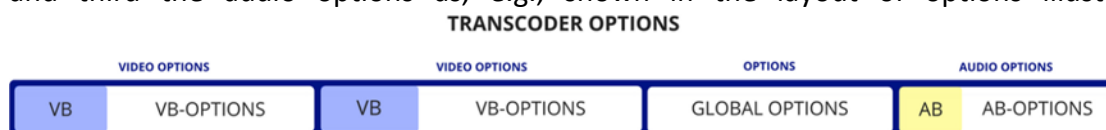


Figure 2-26.

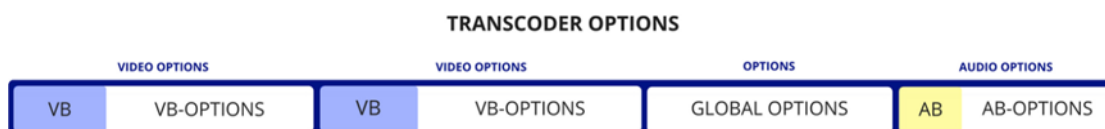


Figure 2-26: Illustration of the layout of options of the IoRL transcoding VNF

Hence, the transcoder for the incoming stream in the `/etc/flussonic/flussonic.conf` configuration file can be included as

```
stream ort {
    url udp://239.0.0.1:5000;
    transcoder vb=2048k size=1280x720 preset=fast ab=128k;
}
```

while any new re-configuration can be considered via the command `/etc/init.d/flussonic reload`.

Furthermore, the IoRL transcoder can process input content to create either different parallel outputs or duplicate outputs.

Case of different parallel outputs: this case is to support multiple outputs created out of the same input(s) content(s) in the same process as illustrated in **Figure 2-27**.

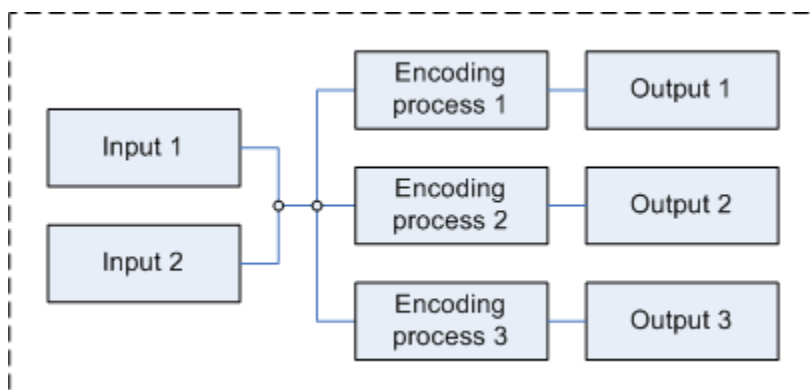


Figure 2-27: Illustration of the IoRL transcoding VNF under the case of same input(s) and different parallel outputs

This can be accomplished by using the below commands.

```
ffmpeg -i input1 -i input2 \
    -acodec ... -vcodec ... output1 \
    -acodec ... -vcodec ... output2 \
    -acodec ... -vcodec ... output3
```

That way the IoRL transcoder can create several different outputs out of the same input(s). For example, to encode a video content in HD, VGA and QVGA resolution, at the same time, the transcoder executes

```
ffmpeg -i input \
    -s 1280x720 -acodec ... -vcodec ... output1 \
    -s 640x480 -acodec ... -vcodec ... output2 \
    -s 320x240 -acodec ... -vcodec ... output3
```

Note that the aforementioned approach considers no filtering options for the outputs meaning that the content cannot split into multiple streams. However, in case of large number of viewers, it is likely to apply filtering to all outputs, which can be attained using the command `-filter_complex` with the `split` filter. For example, to encode a video in HD, VGA and QVGA resolution, at the same time, the transcoder can be configured as below.

```
ffmpeg -i input -filter_complex '[0:v]split=3[out1][out2][out3]' \
    -map '[out1]' -s 1280x720 -acodec ... -vcodec ... output1 \
```

```
-map '[out2]' -s 640x480 -acodec ... -vcodec ... output2 \
-map '[out3]' -s 320x240 -acodec ... -vcodec ... output3
```

The option to apply one filtering instance per output is also available. In that case the transcoder considers filtering, with the different filter(s) applied to each output, using `-filter_complex` with the `split`, but using `split` filter directly to the input, i.e., one filtering instance per each output. For example, to encode a video to three different outputs simultaneously, using three different filters applied to the different outputs (e.g., the `boxblur`, `negate` and `yadif` filter) the transcoder can be configured as below.

```
ffmpeg -i input -filter_complex
'[0:v]split=3[in1][in2][in3];[in1]boxblur[out1];[in2]negate[out2];[in3]yadi
f[out3]' \
-map '[out1]' -acodec ... -vcodec ... output1 \
-map '[out2]' -acodec ... -vcodec ... output2 \
-map '[out3]' -acodec ... -vcodec ... output3
```

Case of duplicate outputs: this case is illustrated in **Figure 2-28** and it is to support the streaming of live audio/video contents, while at the same time, duplicates of these streams are saved into the server to avoid encoding twice, which wastes computing resources of the network.

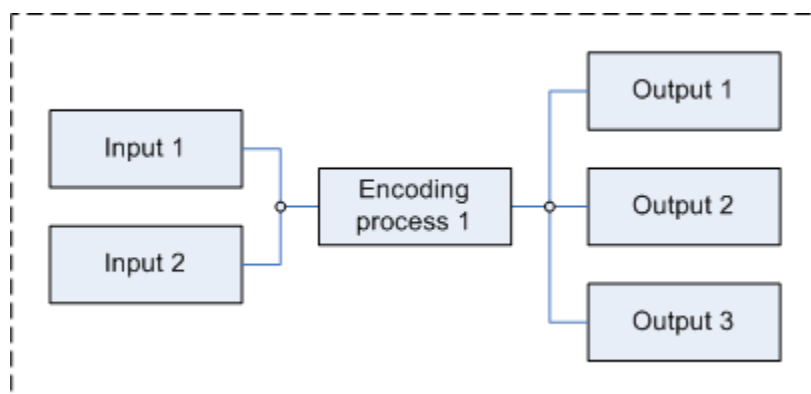


Figure 2-28: Illustration of the IoRL transcoding VNF under the case of same input(s) and duplicate outputs

The below code provides duplication of an MKV video file, which outputs to multiple files with a single instance of `ffmpeg` and a UDP stream.

```
ffmpeg -i input.file -c:v libx264 -c:a mp2 \
-f tee -map 0:v -map 0:a "output.mkv|[f=mpegts]udp://10.0.1.255:1234/"
```

In case specific streams have to be chosen, the `select` option has to be utilised. In that case the video stream is split and scaled to two different sized outputs, and the audio is encoded only once but used by both outputs, otherwise, the transcoder would have to unnecessarily re-encode the same audio multiple times, which wastes resources. To avoid such waste, the `ignore` value for the `onfail - abort` option in the last output can be used to keep the other outputs running even if that last output fails, as shown in the following code example.

```
ffmpeg -i input -filter_complex \
"[0:v]split=2[s0][s1]; \
```

```
[s0]scale=1280:-2[v0]; \
[s1]scale=640:-2[v1]" \
-map "[v0]" -map "[v1]" -map 0:a -c:v libx264 -c:a aac -f tee \
"[select=\ 'v:0,a\']local0.mkv| \
[select=\ 'v:0,a\':f=flv]rtmp://server0/app/instance/playpath| \
[select=\ 'v:1,a\']local1.mkv| \
[select=\ 'v:1,a\':f=flv:onfail=ignore]rtmp://server1/app/instance/playpath"
```

Note that duplicated outputs can further improve transcoding by considering additional processes, so-called piped processes. In such case, one piped process can be used to encode the stream(s), while a second piped process can be used to duplicate the stream(s) to several outputs. It can be achieved using the below code example.

```
ffmpeg -i input1 -i input2 -acodec ... -vcodec ... -f mpegts - | \
    ffmpeg -f mpegts -i - \
        -c copy output1 \
        -c copy output2 \
        -c copy output3 \
```

Outputting and re encoding multiple times in the same transcoding process will typically slow down to the "slowest encoder" in the predefined encoding list. Some encoders (e.g. like libx264) perform encoding "threaded and in the background" so they will effectively allow for parallel encodings. However, note that audio encoding may be serial (not parallel) and become the bottleneck, etc. Hence, in case the transcoder includes encodings that are serial, these encodings will be treated as "real serial" by transcoder and thus ffmpeg may not use all available computing cores, which leads to performance degradation. To avoid the potential of performance degradation, we can (i) either enable multiple serial ffmpeg instances running in parallel, (ii) perform piping from one ffmpeg to another to "do the second encoding", etc., or (iii) use a different – faster - format for the video file (e.g. raw format).

2.3.2.3 Contributions and evaluation of the proposed transcoding process

Having defined the main characteristics of the proposed transcoding process, we continue by highlighting below the key contributions of our development.

Main contributions of the IoRL transcoding process:

- 1) The proposed transcoding process extends from typical ffmpeg setup by using duplicate outputs to increase network performance in terms of saving resources through avoiding re-encoding the streaming videos and audio content;
- 2) The IoRL transcoder utilises piped processes to run multiple ffmpeg instances in parallel that can better utilise the computing resources of the network compared to traditional approach, where the ffmpeg instances run in serial manner;
- 3) We have utilised SDN/NFV technologies to reflect the proposed transcoding process by means of cost-effective VNF instead of bare metal, which also facilitates the dynamic deployment of transcoding together with multi-source streaming VNF;
- 4) We evaluate our transcoding VNF considering triggering events of a congested bottleneck at the access link of the home users, which helps to validate the performance of our development in large-scale system setup.

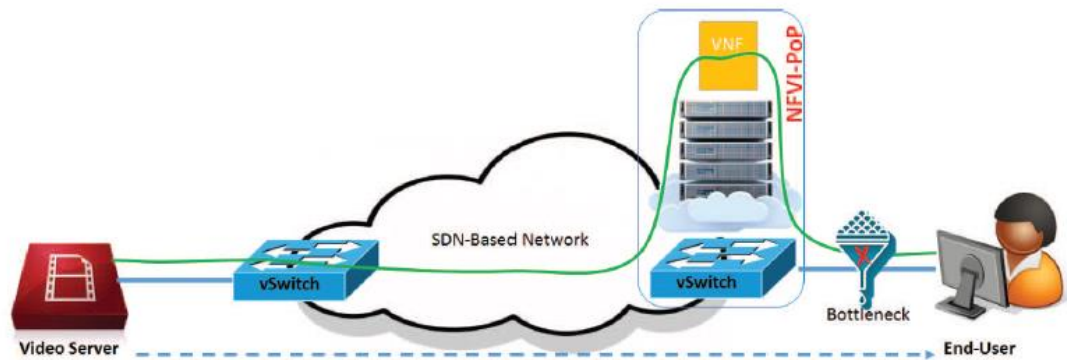


Figure 2-29: Illustration of the topology of the experimental testbed

To evaluate our transcoder, we consider the experimental topology illustrated in **Figure 2-29**, where at the ingress and egress points of the terrestrial network domain there exist two SDN-compatible Open Virtual Switches (OVS) managed by the OpenDaylight SDN controller.

At the egress point of the domain, an Opstack cloud platform has been installed to support the instantiation and orchestration of the transcoding VNF, based on the SDN/NFV environment of the IoRL home network as described in **IoRL deliverable D3.1 [IoRLD3.1]**. The experiment focuses on sending video contents from the server to the user and evaluating the structural similarity index (SSIM) over time, which accesses the network traffic with respect to the video quality offered to the user. In **Figure 2-30**, we consider network overload by setting up each video requested by the user to be approximately of 512 kbps quality, which exceeds the total capacity of the access link and results to severe quality degradation.

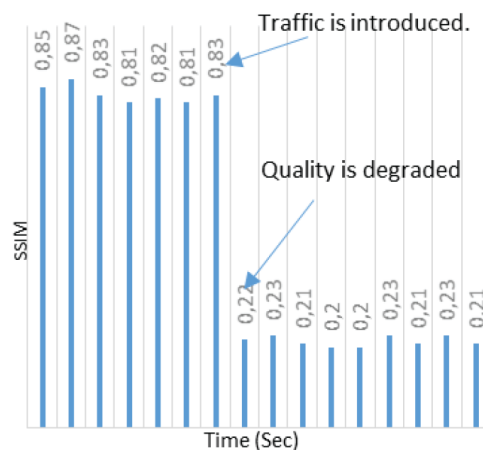


Figure 2-30: Illustration of the video quality (SSIM) over time with no transcoder VNF

However, when the orchestrator instantiates the transcoder VNF, we observe in **Figure 2-31** seamless improvement of the media service delivery.

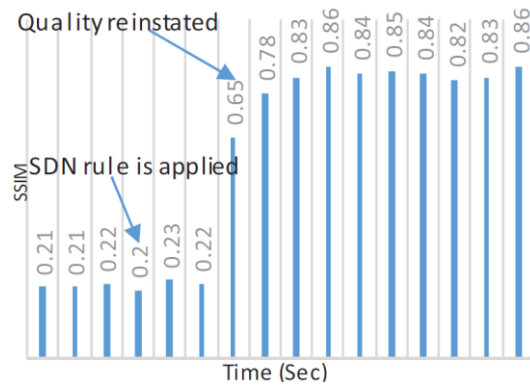


Figure 2-31: Illustration of the impact of the IoRL transcoder VNF on video quality (SSIM) over time

More specifically, our VNF performs real time transcoding of the media service from video quality of 512 kbps down to 256 kbps. In that way, the video quality can be reinstated seamlessly (i.e. without requiring any interruption) because the total traffic has been reduced, which allows the user to be served efficiently through the available capacity of the access link.

It should be finally noted that the presented study with results refer to the preliminary development of the IoRL transcoding VNF, which will be fully presented in the next IoRL deliverable D3.2 including more details on the applicability of the NFV and SDN technology towards providing agile video adaptation solutions and further evaluations based on more inclusive experimental testbed.

2.3.3 360 Degree Streaming

360 degree streaming can refer to 360 media being displayed in real time to mobiles, tablets, pc’s and VR headsets.

This is managed using a 360 degree camera to upload media to an online platform. This data can then be accessed by other devices and played with minimal latency.

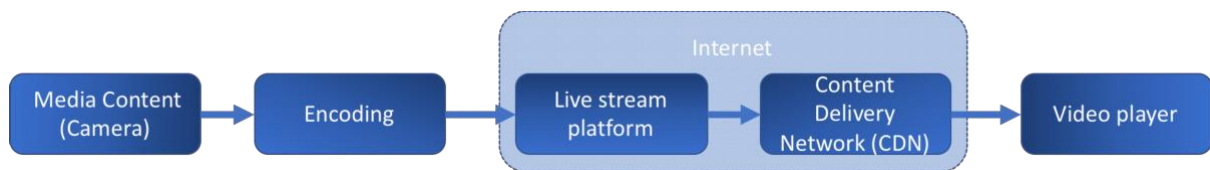


Figure 2-32: Streaming Process

2.3.3.1 Mobile

Enabling streaming from a Theta V camera to a tablet, PC, mobile and thus a mobile VR experience.

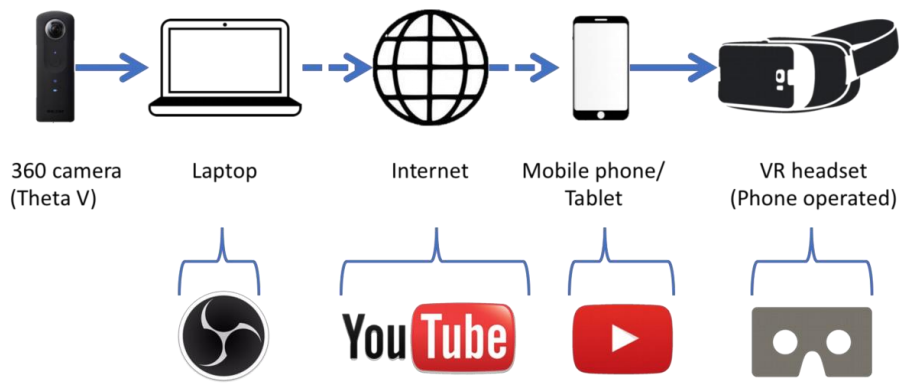


Figure 2-33: Mobile Tablet Streaming Process and Software

Hardware

Theta V camera

- Enables 360 degree video and imaging
- Features live streaming up to 4k resolution

Softwares

OBS (Open Broadcast software)

- o A free and open source encoding software for Live streaming and video recording. Outputs RTMP format.

YouTube Uploader

- o YouTube has existing features that allow for uploading 360 degree live streams

YouTube App

- o The mobile app enables mobile VR experiences given its in built VR Play mode

Theta Live Streaming driver

- o Driver required to live stream, supposedly only required on Windows.
- o Available with instructions in section '2.3.2 Theta V Windows Drivers' [THETA]

Chrome

- o Chrome is the required browser to live stream from Youtube

2.3.3.1.1 Uplink process

Firstly ensure you have downloaded all relevant software's and drivers mentioned previously.



Figure 2-34: Theta V USB Connection

1. Connect Theta V via USB cable
 - Turn Theta V on
 - Cycle between modes for Live Video symbol as shown in **Figure 2-35**

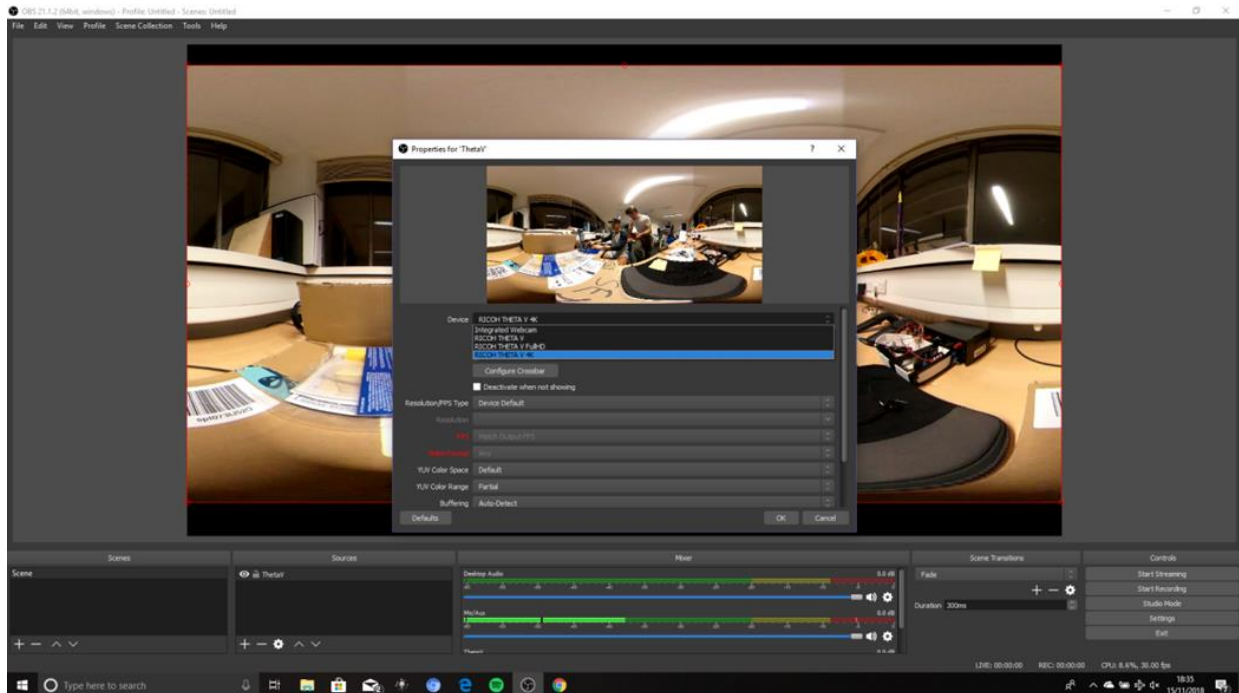


Figure 2-35: OBS Software showing the Media Input Selection

2. Open OBS software
 - Select (+) in sources as shown and select 'Video Capture Device'
 - In the following box select 'create new' and enter a name for the source (ThetaV)
 - In the next popup select the correct input source from the 'Device' dropdown selection.
 - *Ensure the red boundary of the image fills the black background. Any media outside the main black boundary will be cropped.*

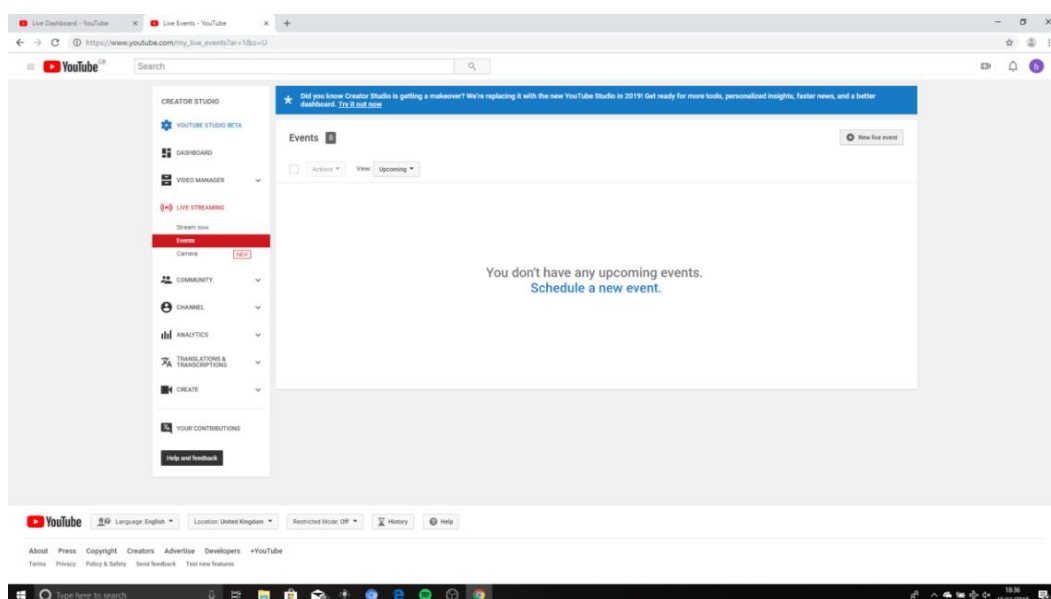


Figure 2-36: You Tube Live Streaming Events Page

3. YouTube

- Login or register to a YouTube Account
- Reach the 'Creator Studio' page
- Under 'Live Streaming' in the side menu Select 'Events'
- Select 'New Live Event'
- Enter a 'Title' and time to start. Ensure 'Type' is set to Custom
- Enter Advanced settings
- Scroll down and select 360 video – This live stream is 360
- Create the event

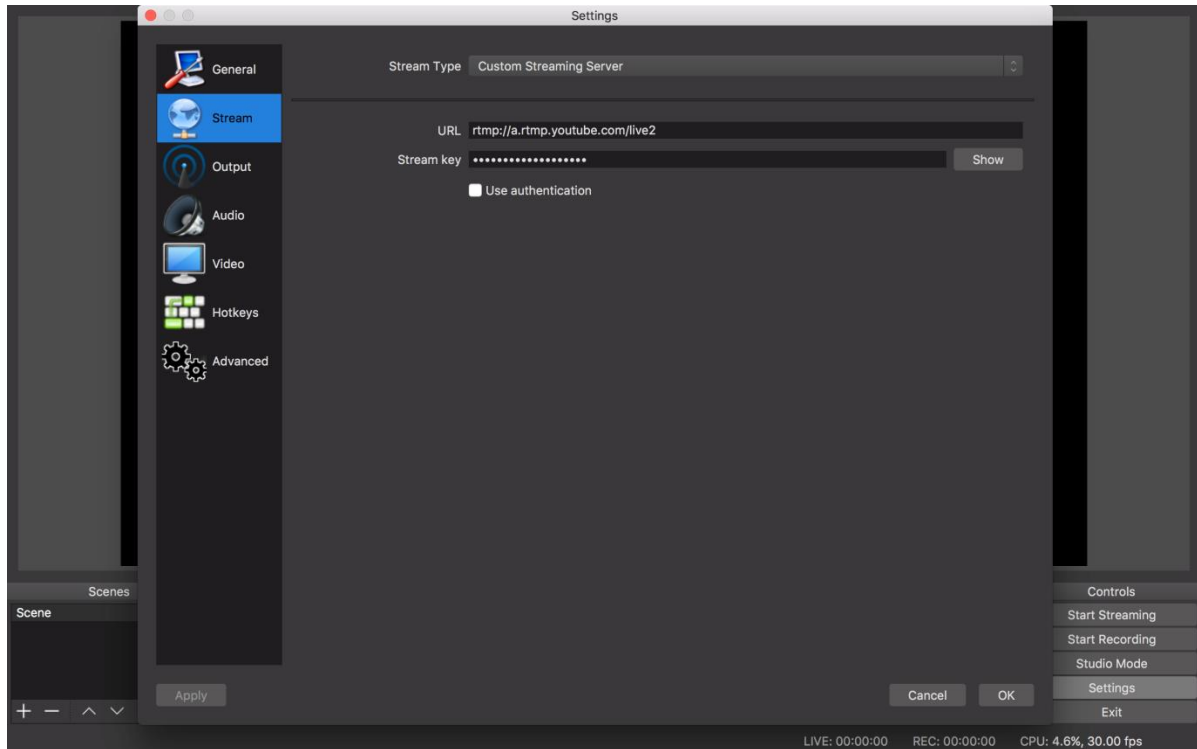


Figure 2-37: OBS Stream Key Inputs

4. Stream-key connection

- Select either 'Single-use stream key' for first time use or 'Reusable' to save stream settings for future use.
- Copy and paste the 'Stream name' and 'Primary server URL' into the OBS Settings (bottom right corner)-> select 'Stream' menu -> type as Custom Streaming server.
- Stream key = The Stream Name
- URL = Primary Server URL
- In Step 4. Select the Hyperlink to the live control room and begin the stream.

2.3.3.1.2 Downlink process

Download the YouTube App and search for the title of the streamed video.

To experience the 360 live video in VR mode select the cardboard mode (👓) and place in a mobile VR headset.

2.3.3.1.3 Where IoRL fits in

The key components to this application are latency and quality. Low data rates lead to slow throughput and increased latency. This causes the viewers experience to be hindered especially in scenarios where user involvement or if feedback were to be applied.

Quality relating to image and audio resolution: In 360 video there is a high amount of data to process. Image quality often suffers in slow networks.

The IoRL system would allow for less compression, meaning possibly no encoder necessary, and faster uplink and downlink. This would greatly reduce the latency between users.

2.3.3.1.4 Results

The latency of this process was tested under the following conditions:

Uplink Laptop: 8.27Mbps down & 5.09Mbps up

Downlink Mobile: 10.15Mbps down & 5.31Mbps up

This provided a latency of approximately 32 seconds

2.3.3.2 VR Headset

2.3.3.2.1 Overview

Streaming to a PC operated VR Headset poses new challenges. VR is fairly new technology and as such there are no existing applications or packages that allow live streaming from a personal device. Some services however do exist for companies, events and specific functions. For this application it was possible to stream from a personal device (Theta V) using an online streaming platform with VR compatibility.

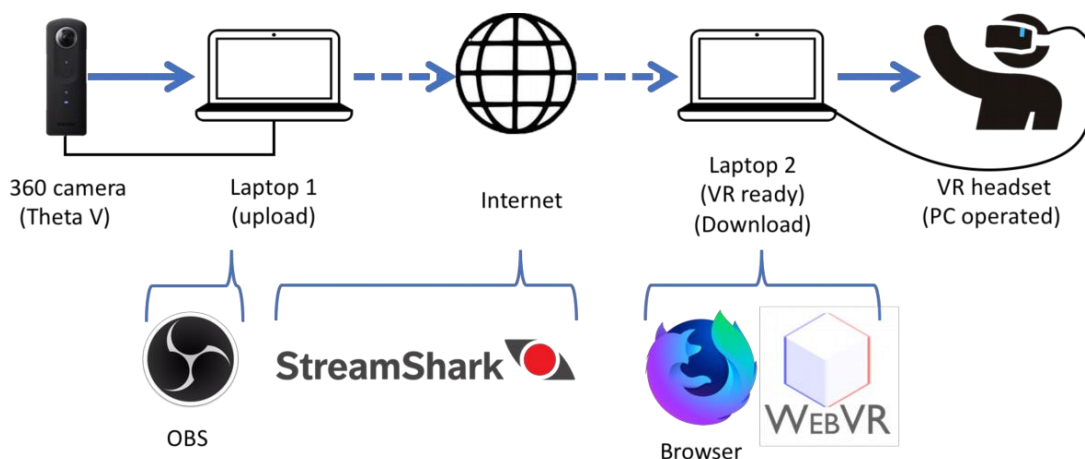


Figure 2-38: PC Operated VR Streaming Process

Hardware

Theta V camera

- Enables 360 degree video and imaging
- Features live streaming up to 4k resolution

VR headset (HTC Vive)

- Provides Immersive environment

- Highest demanding media service currently available, ideal showcase for IoRL 5G

Software

OBS (Open Broadcast software)

Streamshark

- an online streaming platform which utilises VideoStitch Vahana VR software to enable 360 live streaming to VR headsets such as the HTC Vive, Oculus Rift and Google Daydream
- Account needed to share content

Firefox nightly

- One of the first Browsers to integrate WebVR compatibility allowing to view the internet in VR
- Suggested by Streamshark to enable viewing the provided online player in VR

WebVR

- An online service aiming to provide the internet in a whole new way
- Suggested by Streamshark to enable viewing the provided online player in VR

2.3.3.2.2 Uplink Process

Repeat steps 1 and 2 from section 2.3.3.1.1

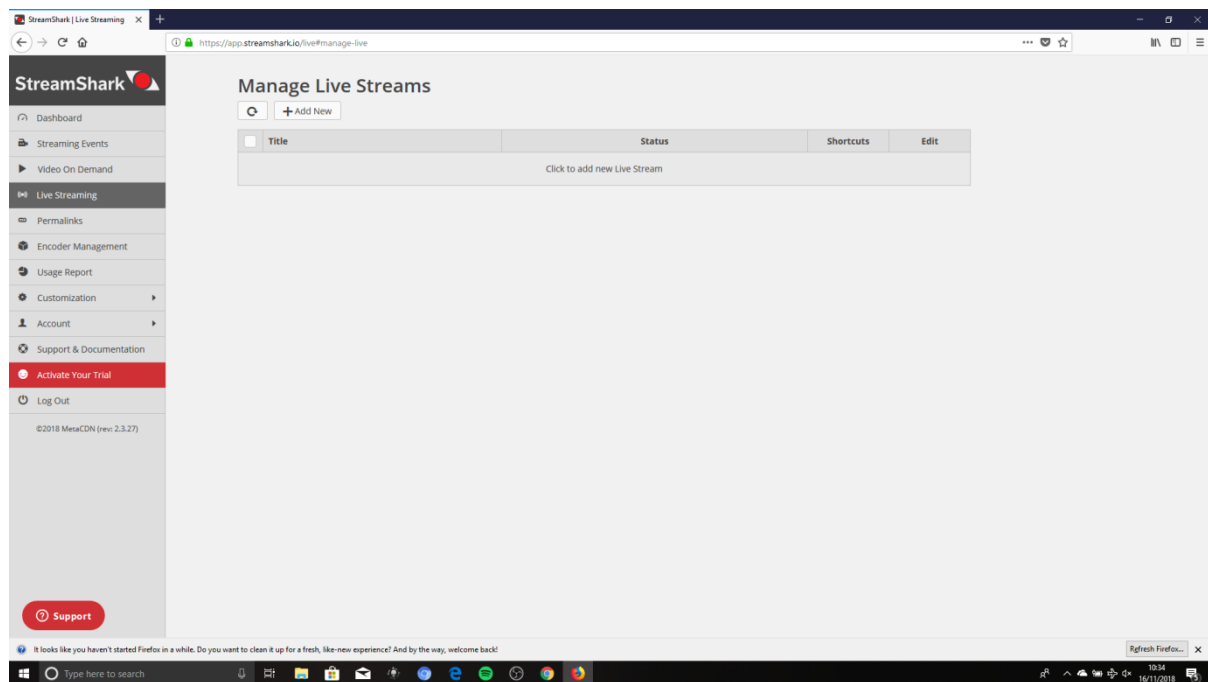


Figure 2-39: Streamshark Live Streaming Inputs Page

2.3.3.2.3 Streamshark

- Login to Streamshark account
- Head to 'Live Streaming' in the main menu
- Select '+Add New'
- Include general settings such as stream name, title, password etc
- Streamshark will then process the stream which can take up to 40 minutes

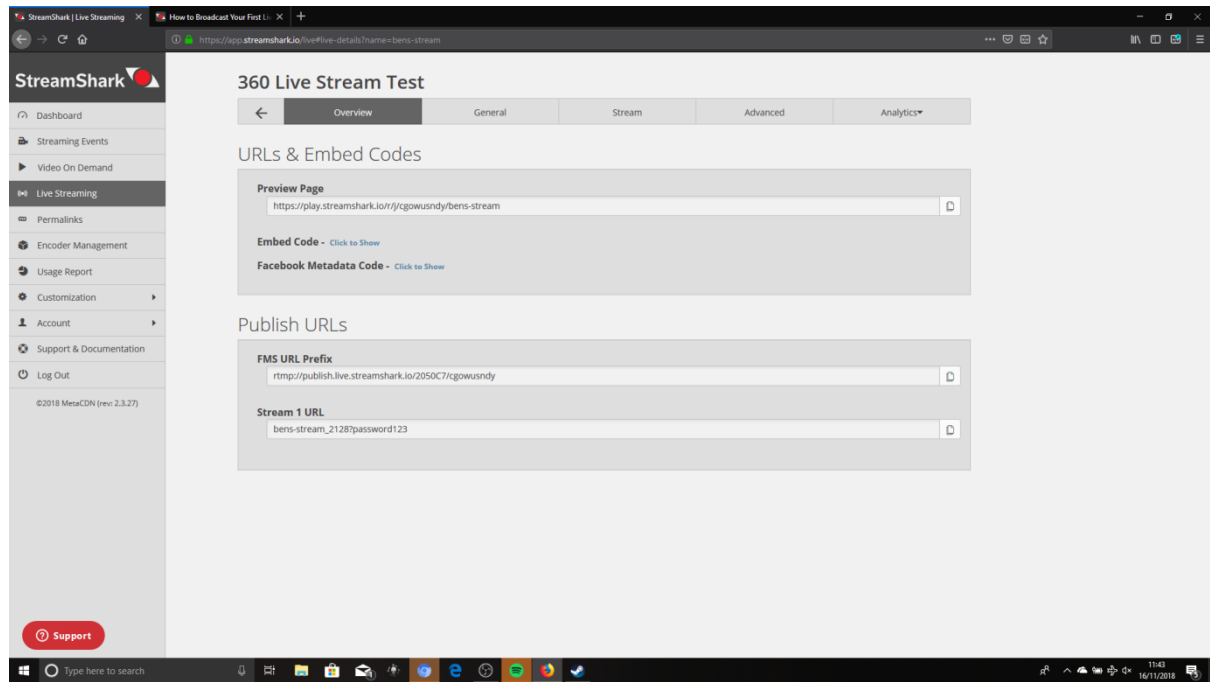


Figure 2-40: Streamshark Streamkey settings

2.3.3.2.4 Stream-key Setup

- Within Streamshark ‘Overview’ note the Publish URLs
- In OBS settings- stream – custom settings:
 - o URL = StreamSharks ‘FMS URL Prefix’
 - o Stream Key = Streamsharks ‘Stream 1 URL’
- Apply the changes and begin streaming

2.3.3.2.5 Downlink Process

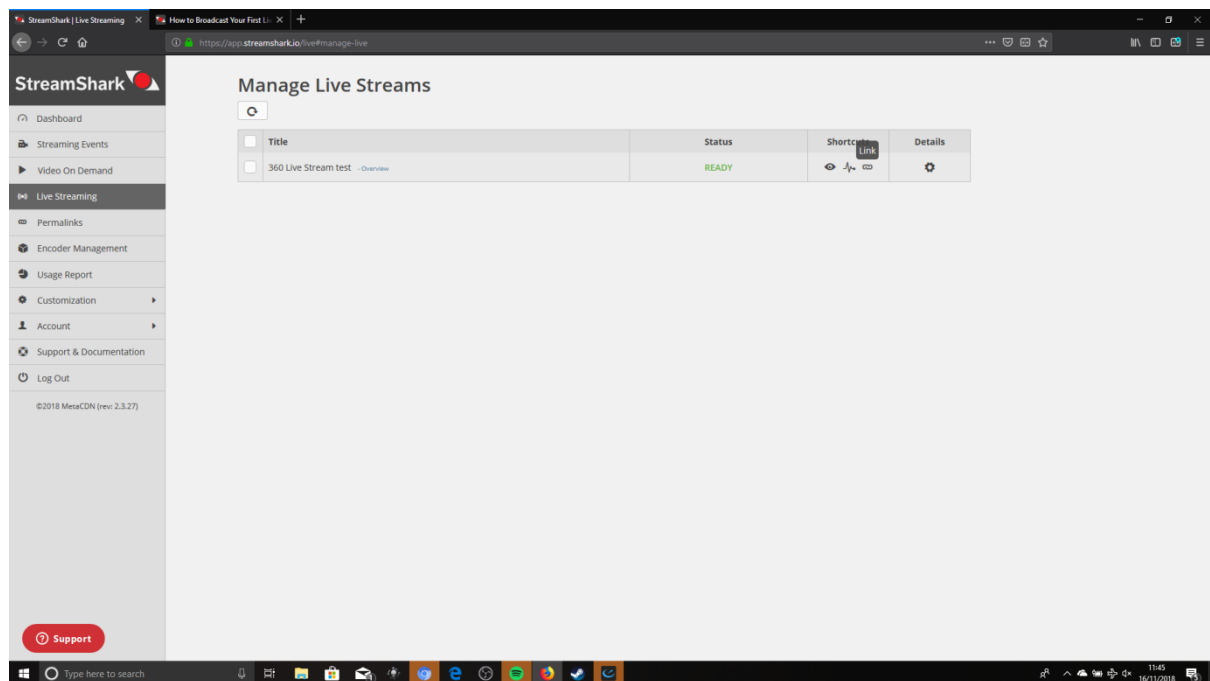


Figure 2-41: Streamshark Management Page

Firefox nightly

- Using Firefox Nightly with WebVR installed, copy the URL of the Streamshark stream provided under 'link'
- While running Steam VR, select WebVR mode

2.3.3.2.6 Results

This use case was tested under the following conditions:

Laptop 1 (Wi-fi) – 8.27Mbps down & 5.09Mbps up

Laptop 2 VR (Wired) – 25.24Mbps down & 10.97Mbps up

With a maximum total induced latency of approximately 31 seconds

2.3.4 Gaming and Virtual Reality Applications

2.3.4.1 Current 6DOF process

6 Degrees of Freedom (6DOF) is available on PC, standalone and modified mobile systems such as the NOLO [NOLO]. With the exception of standalone systems the processes for delivering 6DOF are generally similar. External trackers, such as 'Lighthouses' for the HTC Vive, use various methods to detect headset and controller node movements. This raw positional data, coupled with sensory data from sensors within the headset (accelerometers, gyroscopes and magnetometers), is delivered to the PC VR application. The PC application processes the raw data into useful motion that can be used to understand the users physical position. The PC will output the corresponding media output (audio and visual) to the headsets display via HDMI and USB cable.



Figure 2-42: Existing VR system overview

2.3.4.2 IoRL Location sensing overview

Stage 1 – Storing location sensing data

Storing the location data requires an initial protocol in which TDOA and RSS data from both mmWave and VLC technology respectively is collected by the controller. The collection of data from each RRLH, along with the controller and User Equipment (UE) addresses is delivered to the IoRL Software Defined Network (SDN). Before being stored in an SDN database the TDOA and RSS raw data must be processed by the relevant algorithms to determine the users' physical position. These location estimations will likely be averaged to determine the best approximations.

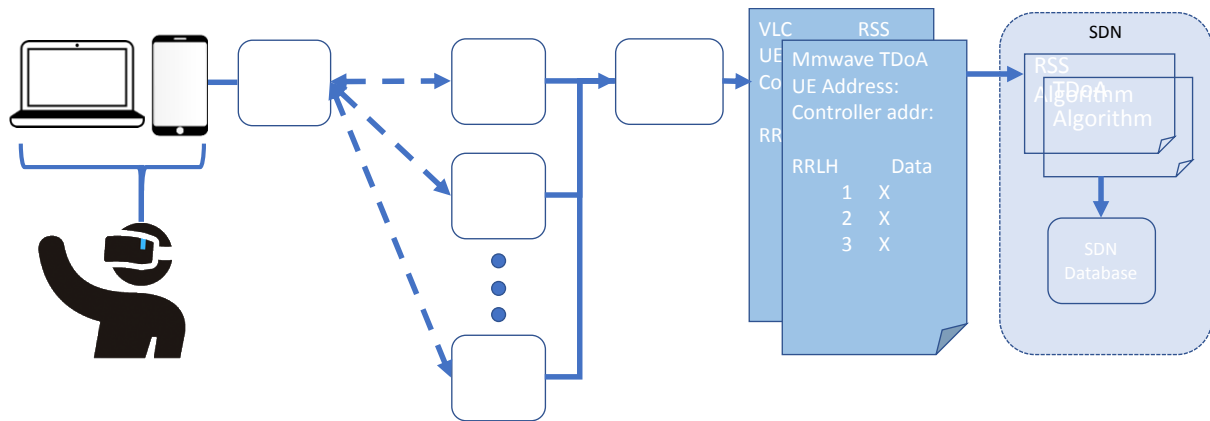


Figure 2-43: Location sensing protocol 1

Stage 2 – Accessing location sensing data

With the location of the UE (VR headset) known the VR application must retrieve this data, with minimal delay, to determine the users physical movements and thus their position in the virtual environment. This requires a second protocol within the VR application. The procedure involves the mobile or PC application sending out a request from the device dongle to the RRLHs. The location data is then outputted from the SDN database back to the application. This is illustrated in **Figure 2-44: Location sensing protocol 2** below.

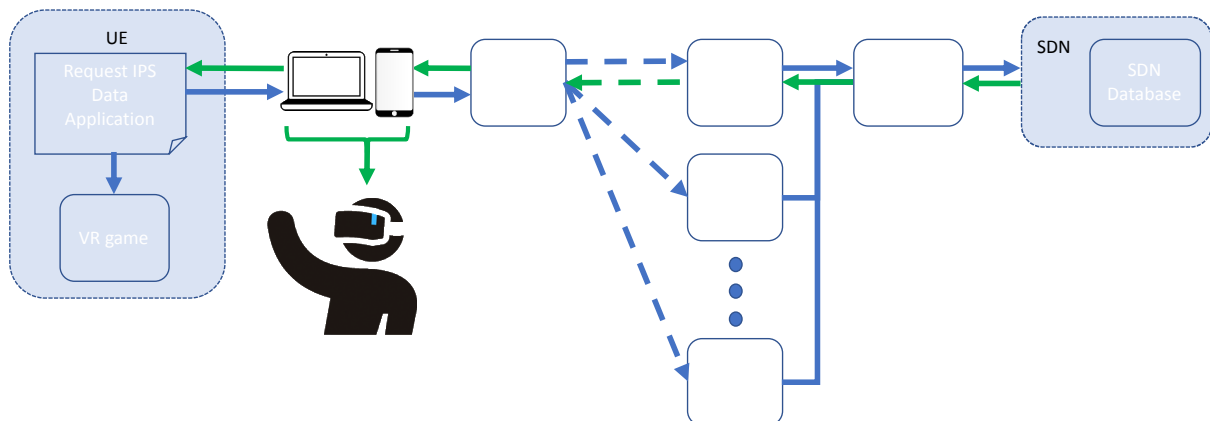


Figure 2-44: Location sensing protocol 2

2.3.4.3 PC VR operation

For PC VR systems the existing external tracking system from (**Figure 2-42: Existing VR system overview**) can be substituted for the IoRL location sensing data. This data must be suitably interfaced to work with the PC VR application in order to translate the movements correctly. In doing so it would be possible to provide 6 Degrees of freedom to PC games without the need for visible external trackers. The RRLH’s would act as multiple discrete external trackers.

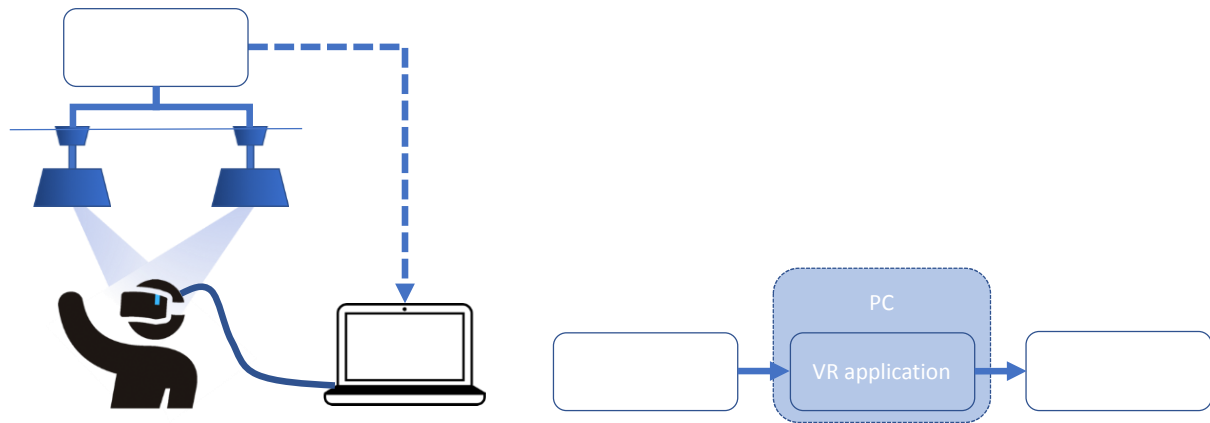


Figure 2-45: PC IoRL VR

2.3.4.4 Mobile VR operation

By providing location sensing data to a mobile device it should be possible to enable 6DOF for mobile VR experiences. The location data can be requested by the mobile application and once received, processed correctly to determine the users’ movements and thus the appropriate media output.

2.3.5 Bike Game

2.3.5.1 Bike Game overview

The bike game allows users to maneuver and power a static exercise bike while wearing a VR headset gives a VR experience which simulates that they are cycling in motion. The objective of this application is to develop multiplayer ‘races’ where users can compete with friends. IoRL technology intends to reduce latency between users and increase the general VR quality.

2.3.5.2 Key hardware components

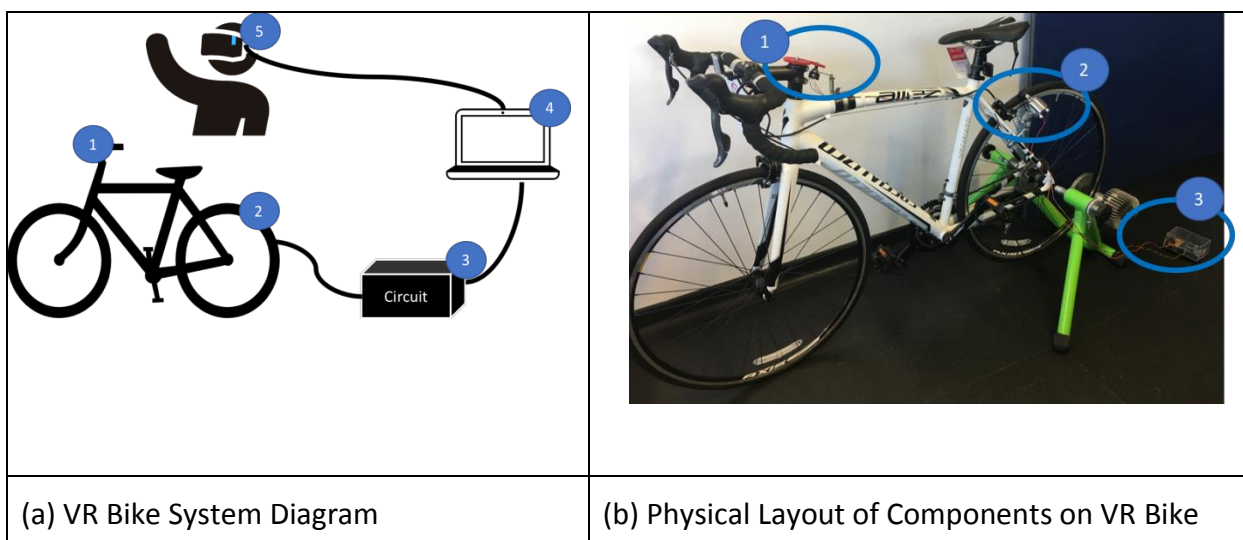


Figure 2-46: VR Bike

1. Potentiometer (Steering)

The potentiometer is mounted such that it rotates by turning the handlebars. This is managed by a joining of gears connected to the handlebar shaft and the central beam of the bike. The potentiometer's position can then convey in 0-5V the position of the steering wheel from 90 degrees clockwise to 90 degrees anticlockwise. Approximately 2.5V being 0 degrees, with the wheel in a straight position.

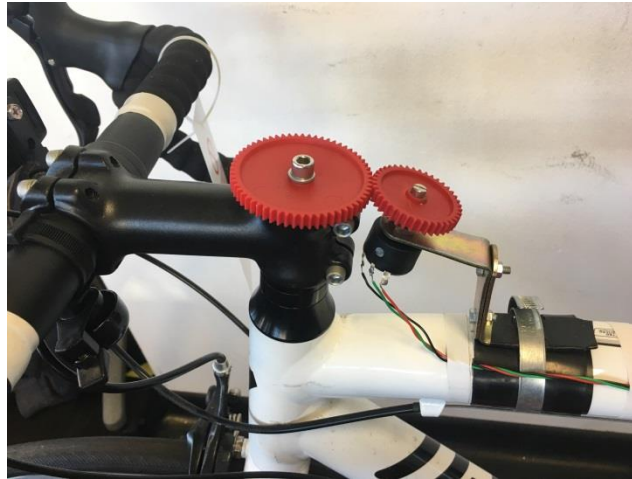


Figure 2-47: Potentiometer on Steering

2. Dynamo (Speed)

The Dynamo is used to generate a voltage that varies given the speed of the user. This is mounted to the back wheel in order to provide speed data. This is also handy for braking given that the back wheel is stopped when manual brakes are applied.



Figure 2-48: Dynamo Speed Sensor

3. Circuit

The circuit feeds the potentiometer and dynamo connections to the Arduino board. The circuit also reduces the analogue -60V -> 60V signal produced from the Arduino into a single 0 – 5V signal. Using a simple rectifier, capacitor and potential divider circuit as shown below.

The Arduino is programmed to read the data inputs, map them to suitable values and output a single package of data with steering position and output speed.

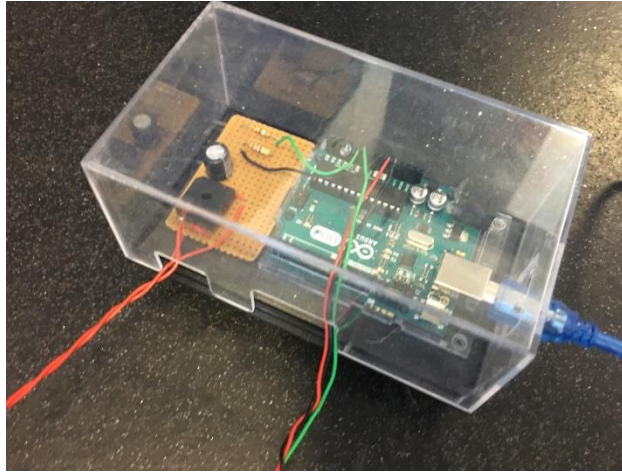


Figure 2-49: Arduino Interface Circuit

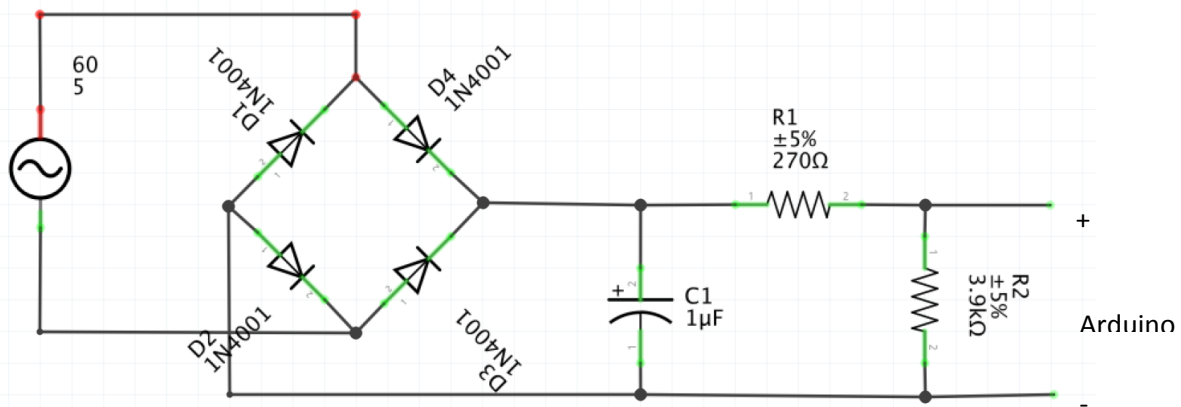


Figure 2-50: Circuit diagram for ADC to Arduino Computer

A VR ready computer is required to handle the processing and graphics demanded by VR systems. The computer also runs the game program using unity software.

Headset

In order to use VR a VR system is required. A HTC Vive was used for this application.

Key software

Unity 3D

- Powerful physics engine software used to develop games and 3D virtual environments

Stream VR

- Required to run a HTC Vive system

2.3.5.3 Program overview and using the program

Scripts within the unity software allow for data to be read from the bikes Arduino via USB serial port. When extracting the data into controllers for the physics engine the data package

is separated. This is then used to individually manipulate the rotation of the handlebars and the turning of the back wheel.

Unity comes prepared with a pre-existing 'Stream VR Plugin' this allows instant access to VR within 3D virtual environments.

2.3.5.3.1 Multiplayer

Unity 3D contains network managers that provide the means to enable multiplayer gaming. This has been added to the VR bike game, allowing for multiple users to play together. This internet connectivity allows for this application to be enhanced by the IoRL project.

2.3.5.3.2 Game Start up

1. Connect the programmed Arduino to the VR computer
2. Ensure in the 'Bike move script' that the correct computer port is selected to read data from
3. To run both the multiplayer and non-multiplayer versions of the VR Bike game simply open the respective Unity Files and build the game for the appropriate platform.

The VR Bike project can be found on the IoRL code repository on GitHub [IoRLGR].

2.3.5.4 Future improvements/plans

Several improvements are being considered within the bike game to benefit ease of play and user comfort.

Firstly, the Serial Port communications between the Arduino and Unity scripts should be terminated between each event of game play. This will reduce the data reading errors that occur preventing the virtual bike from responding to user inputs.

Various maps with a diverse range of scenes and surroundings could be implemented. A greater focus on realistic environments could improve player immersion while highlighting the data streaming benefits of the IoRL project.

Calibration to map the virtual bikes position to the physical bike would save tedious setup time.

2.3.6 UHD TV Streaming Applications

2.3.6.1 Content Display

In the first step, the development of an application for content display delivered via Viavi Solutions dongle to UHD TV set will be targeted. The functionality of the developed application as target demo scenario are clarified as follows. The software on TV set runs on embedded Linux platform. The native and web applications are supported running on this platform. A specific TV application is needed to display video stream received over IoRL network. The TV application consists of a video player integrated in the webpage with the HTML5 video object. The player receives videos from streaming server or IP cameras. TV will receive streaming video as network packets over UDP protocol. Network packets will be provided from the IoRL dongle via Ethernet connection. Dongle receives signals from light headsets which are connected to SDN.

Figure 2-51 illustrates the proposed application for UHD TV demo scenario of content display.



Figure 2-51: UHD TV Streaming Application – Content Display

2.3.6.2 Multiple Contents Display

In the second stage, the development of an application for support of multiple contents display on TV set will be investigated. If the constraints of TV set which needed to display two different IP streams on TV set at the same time are met on TV software, the application will be developed as final demo. The multiple contents which are displayed on TV set will be delivered by Viavi Solutions dongle with the different header tag as UDP packets. When the applications running on TV set are identified the streams according to header IDs.



Figure 2-52: UHD TV Streaming Application – Multiple Contents Display

2.4 Virtual Reality Tourism

Being able to explore real and fictional environments is becoming an ever growing application of VR. Such examples include Google Earth VR, VR Museum Galleries and Real estate tours. Within IoRL these applications could be stored in a central server which allows for streaming to numerous user devices.

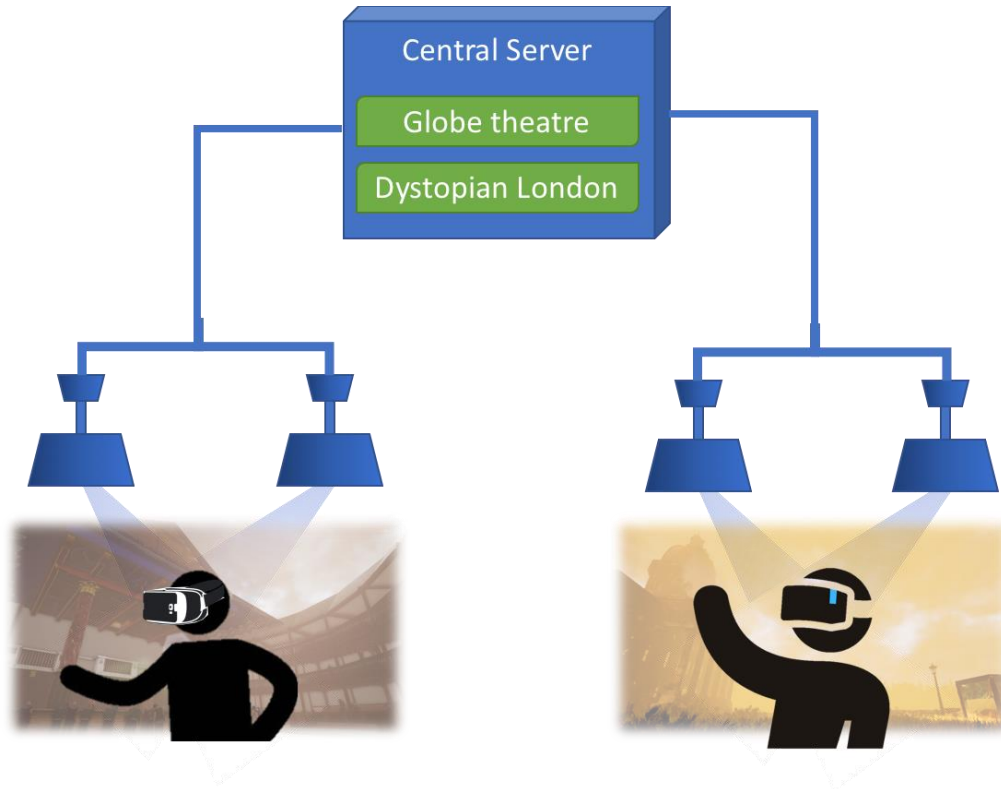


Figure 2-53: single central server streaming VR experiences

Figure 2-53 Illustrates how using similar streaming processes to those detailed in **section 2.3.3 360 Degree Streaming** a central server would store VR experience applications. These could be requested by and streamed to multiple users via the IoRL system.

2.4.1 Dystopian London

The dystopian London experience sets users in derelict and abandoned London landmarks, the Shard and Trafalgar square. Users are able to move around these eerie environments using VR controllers.

The project focused on how central London would look post-apocalypse, transporting users into a fictional futuristic atmosphere. VR was used to provide users with a greater sense of immersion and reality. The software's used were 3Ds Max, Photoshop and Unreal Engine.

- 3Ds Max was used to model the environment from scratch (buildings, roads etc)
- Photoshop was used to create textures for the models.
- Unreal Engine was used to add the VR motion, lighting, rendering, adding foliage and tress.

To run the file:

1. Install Unreal Engine (5.0 or later)

2. Steam VR must be up and running prior to opening the unreal file.
3. For the controller motion to work both the controllers must be turned on so that the software can recognize the left and the right controllers.

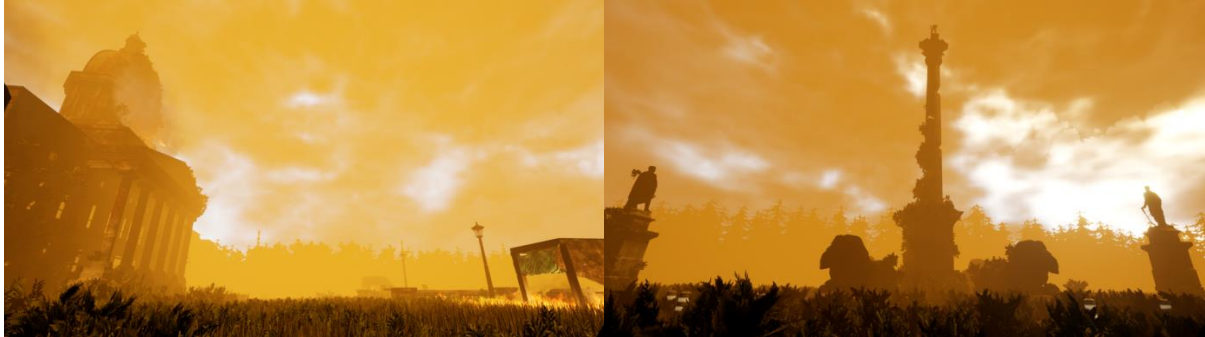


Figure 2-54: Screen Shots of the Dystopian London 3D World

2.4.2 Globe theatre

This VR experience sets the user within the standing area of the Shakespeare Globe theatre during a live performance among other viewers. By transporting the viewer into the past we are able to provide an experience of how one may feel during this period in history.

To start the experience:

1. Ensure Unreal Engine (5.0 or later) is installed onto your device
2. Have Steam VR setup (If using HTC Vive) and running.
3. Open and run the Globe theatre file



Figure 2-55: Screen Shots of the Shakespeare's Globe Theatre 3D World

2.5 Remote Tourism

This user experience takes users on a tour of existing London Landmarks via 360 degree video. The user is unable to traverse freely within the environment given that the environment is not virtual.

Videos used in this use case could vary from museum tours, educational videos to sporting events and festivals.

The virtual reality tourism files (Dystopian London, Globe theatre, London tour) have been published on the IoRL YouTube channel [IoRLYC].



Figure 2-56: Screen Shots of the 360 Degree Virtual Tourism

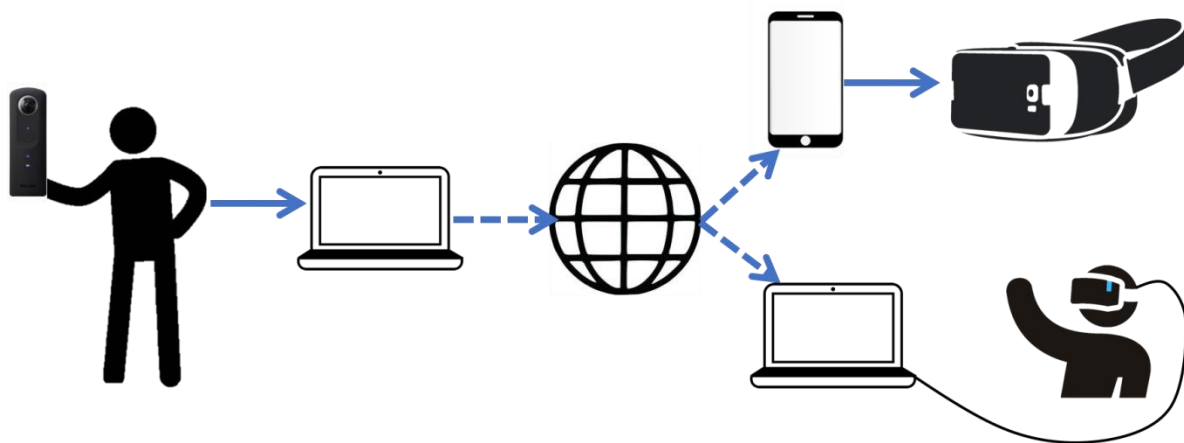


Figure 2-57: Remote tourism streaming process

Initially the IoRL project intends to stream pre-recorded 360 data from a server to a VR headset.

Figure 2-57 demonstrates the proposed plan to live stream a remote tourism experience using the processes detailed in **section 2.3.3 360 Degree Streaming**.

2.6 HoloLens inspection of Tunnels

In this use case, focused on Augmented Reality, the real world remains central to the experience (in this case, the tunnel), enhanced by virtual details. It helps maintenance workers to visualize and measure different parameters such: temperature, humidity and concentration of pollution gasses (NO₂, CO, SO₂) inside the tunnel in real time. This extra information provided by the AR will help to the safety and quality of the maintenance tasks.

To visualize this information the user will carry on a Microsoft HoloLens connected to the IoRL technology. Moreover, the user could visualize tunnel plans, electrical schemes and spatial information correlated with reality i.e. electrical panels, emergency exits, fixing points, etc. In addition to these uses, accuracy location of IoRL system can be tested inside the tunnel and the path to the exit can be shown in the HoloLens. This use case has high importance to test the technology due to the specific conditions of critical infrastructure as a tunnel.

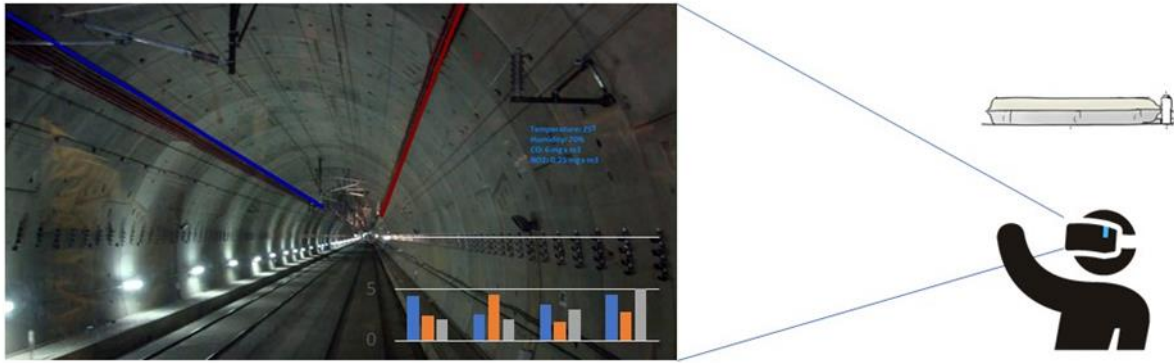


Figure 2-58: Hololens Inspection Tunnel

2.7 Smart TV Applications

2.7.1 Indoor Follow-Me TV and Radio

The aim is to offer service to users within the IoRL coverage area that exploits the huge bandwidth and high data rate provided by home eNodeB due to relying on 5G access and routing technologies to improve the QoE for UEs. Follow me service (FMS) enables users to enjoy watching a variety of videos by their demand from various sources on the different TV screens within the IoRL coverage area.

2.7.1.1 Follow Me Service architecture

FMS architecture is shown in **(Figure 2-59)**, where it is built on top of IoRL architecture with necessary additions to realize the service. It consists of end user application installed on UE smart phone, SDN Follow Me Application (FMA) interacting with IoRL entities (Location server and SDN controller) and proxy/cache server. The main focus of FMA is to manage and coordinate the proxy server according to the location information stored on the location server. Proxy server and location server are fully accessible and controllable by the FMA within IoRL IHIPGW enabling efficient service operation to optimally forward the data to the correct TV destination within the IoRL coverage area.

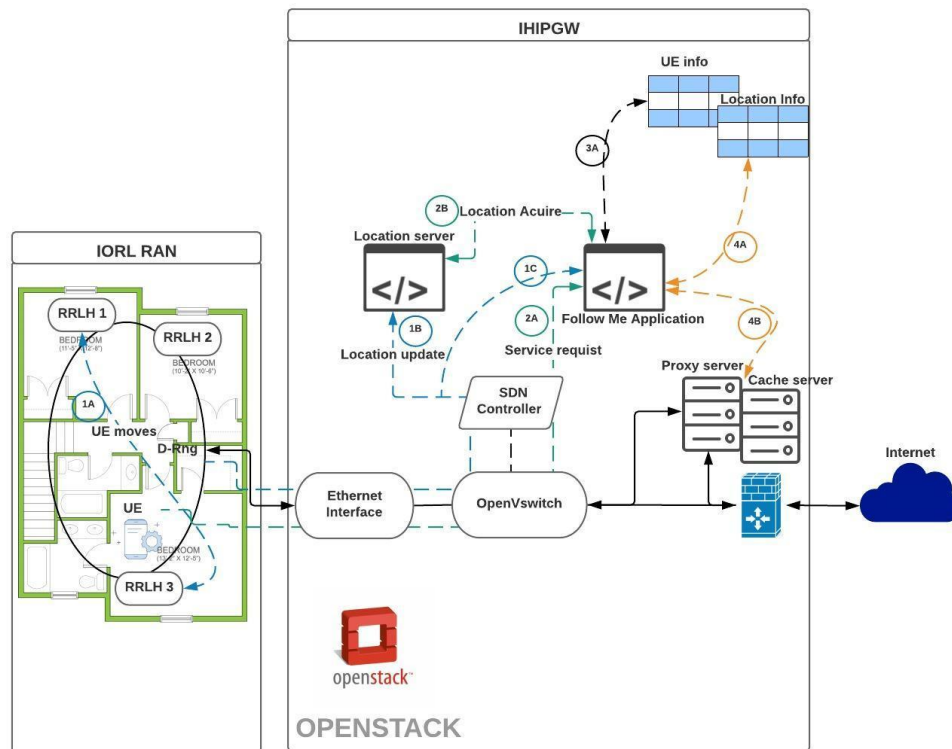


Figure 2-59: Follow me architecture

FMA interactions with location server and proxy server is transparent to the UEs registered for this service, it only requires the UE to initialize the service by requesting a video content to be displayed on specific TV set or multicast the video to all TV sets. FMA configures the proxy server to establish a UDP broadcast session to the end devices. The proxy server intercepts the TCP connection establishments originated by the UE and splits the end-to-end TCP connection into an upstream, downstream TCP sessions and downstream UDP session.

One of the TCP Proxy features to maximize throughput is to establish a connection to the external server asynchronously, download and buffer the downstream data at the proxy and generate ACKs towards the server. It provides improved QoE to the users especially when it starts to send the downstream data using a UDP protocol, which is considered to be faster than TCP, and exploits very wide bandwidth offered by the use of 5G access technology namely VLC and mmWave.

This advancement in the download is the enabler for providing a reasonable throughput gain, and would only become a problem if the service offers inter handover capability, which it does not, because it is unlikely for a user to request such service for TV when moving in and out of IoRL coverage area.

2.7.1.2 Signalling sequence

Follow me service relies on UE location to maintain data flow destined to the correct destination. In IoRL architecture there is a location detection mechanism which uses special algorithms to calculate user location depending on mmWave Time Difference of Arrival TDOA, and VLC Received Signal Strength RSS to locate the user with up to 3 cm² accuracy. Each user location will be stored and constantly updated in a dedicated location server.

The connection establishment and down streaming of the video are based on location updates of the UE, as shown in **Figure 2-60**.

The request and response procedure shown in **Figure 2-60**, requires a client to request a video content, which is consequently requested by proxy from a content server (if it has not already been cached earlier). The proxy splits the session into two TCP sessions UE – Proxy, Proxy – Server.

FMA monitors FMS clients' locations and updates the proxy server when necessary to make the required change to ensure correct delivery to the correct destination using UDP broadcast sessions. SDN intelligence plays an important role in improving FMS performance, where based on the network metrics feedback, RYU¹ controller configures the OVS to select the path of the downstream either via VLC/mmWave, or WiFi link.

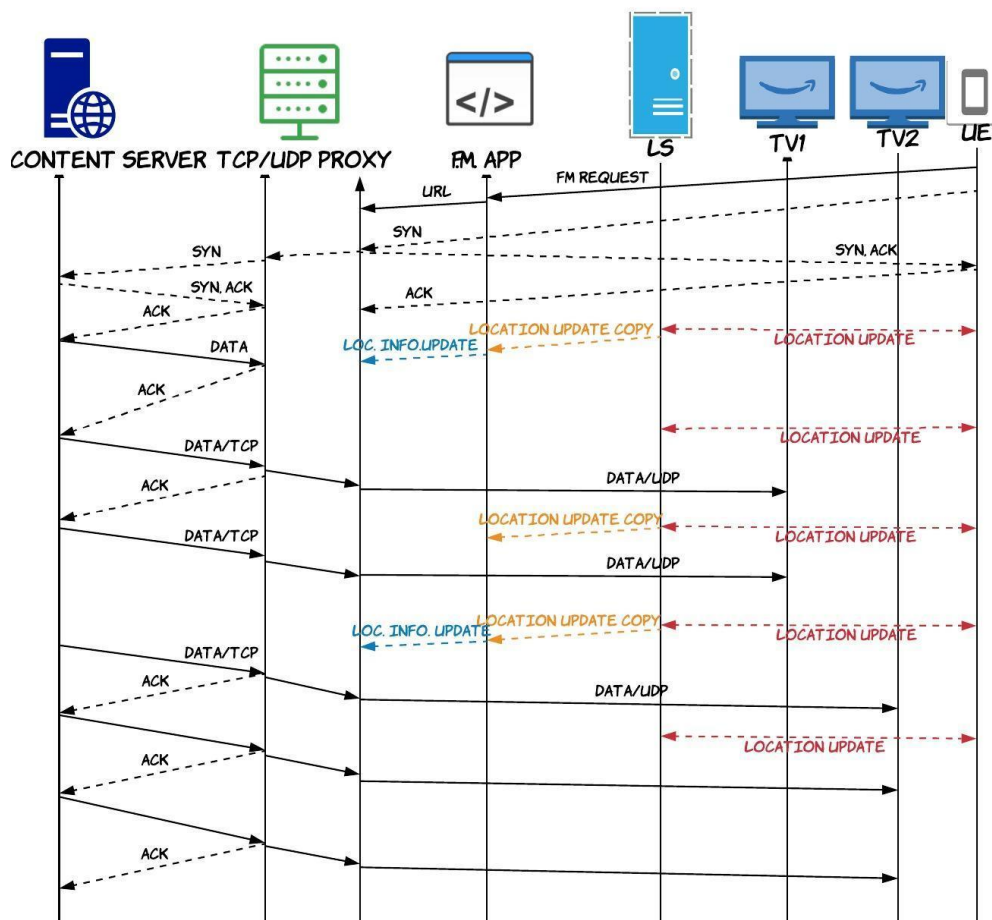


Figure 2-60: Signalling sequence

¹ Ryu is a network OS that integrates with OpenStack and supports OpenFlow. It provides a logically centralized controller and well-defined API that makes it easy for operators to create new network management and control applications

2.7.1.3 Follow Me Service (FMS) operation

Follow Me Service (FMS) relies on three different components to provide its services to IoRL clients. Namely, proxy server, Follow Me SDN Application (FMA), and Follow me end user application. It also exploits the available bandwidth offered by IoRL.

Thanks to 5G access technology mmWave, and VLC communication, FMS utilises the connection less capability of UDP to send the video to multiple destinations without the risk of losing datagrams due to congestion.

2.7.1.3.1 FMA operation procedure

As shown in flowchart (**Figure 2-61**) and (**Figure 2-59**), FMA is registered as a listener for two types of incoming messages, follow me request messages, and location update messages.

- 1- When a Location update message is sent to location server by UE, the FMA receives a copy of that message, and look the UE up in its table. If it is found then it goes to point 3, otherwise, the message is ignored (i.e. the UE is not registered with FMS).
- 2- When the Follow me request message is received by FMA, the URL request is forwarded to the proxy server, and UE is looked up within FMA tables to check if it is a new client or if it is an existing client.
 - a. In case of new client, FMA sends location acquire message to location server, and stores the UE location information from the returned response message.
 - b. Otherwise, it is an existing client, so go to point 3.
- 3- FMA compares the UE location information, with the client's previously stored location information.
 - a. Two comparisons take place in this step: the first comparison is the client's current coordinates with predefined room coordinates to extract TV IP address and TV port (each room match TV IP address, and port number), and the second comparison is to compare the extracted IP address and port number with the previous TV IP address and TV port for the same client.
 - b. New client has no previous TV IP address, TV port number, therefore, the current information will be used by the proxy to start sending the video to its first location.
- 4- If the client is in the same room, then no message will be sent to proxy, otherwise, follow me procedure is triggered.
 - a. Store current TV IP port number that corresponds to client's current location.
 - b. Send a new update message to the proxy server with (UE IP address, TV IP address, and TV port number).
 - c. If the UE is not in FMS enabled area, and not selected (stream to all TVs), then the update message will be equal to (0, 0, UE IP address).

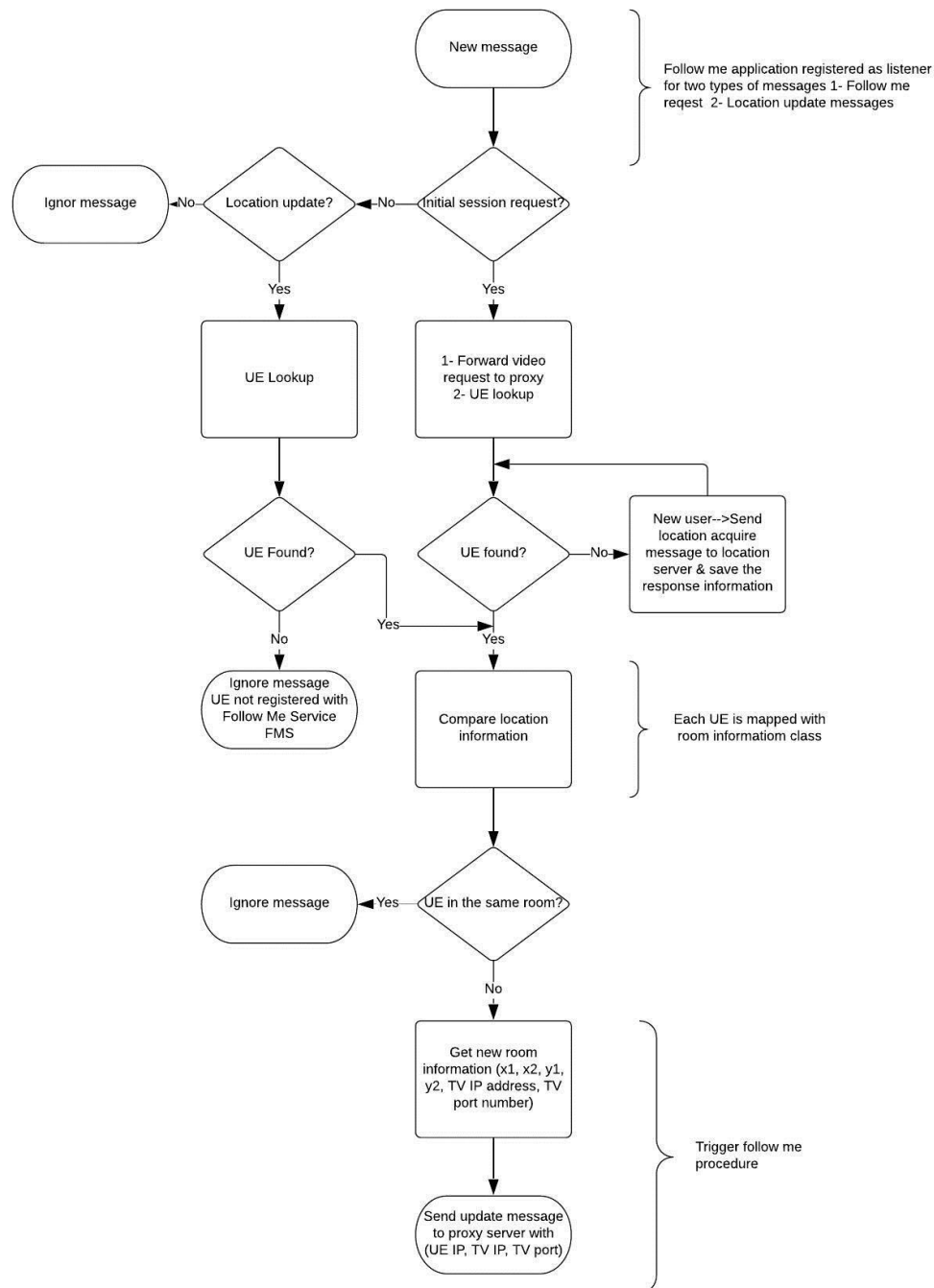


Figure 2-61: FMA flowchart

2.7.1.3.2 Proxy server operation procedure

Proxy operation sequence is shown in flowchart in **(Figure 2-62)**. When a new message is received by proxy, it would either be a client initial session request for a video or an update message from FMA.

- 1- The proxy checks to see if the video has been cached on the local caching server, otherwise it requests the video content from the external server.
- 2- If the received message is an update message, then it includes the required information for the downstream (TV IP address, TV port number, and UE IP address).

- 3- However, if the UE is not in one of the FMS rooms, or (streaming to multiple TVs) option is not selected, then the update message would be equal to (0, 0, UE IP address), triggering the proxy server application to pause and buffer the stream, until a new destination information is received.
- 4- If the UE has selected to stream the video and it is within the premises of one of the FMS enabled rooms, then the proxy uses the (TV IP address, and port number) which is extracted by the FMA based on UE location and down streams the video content accordingly.

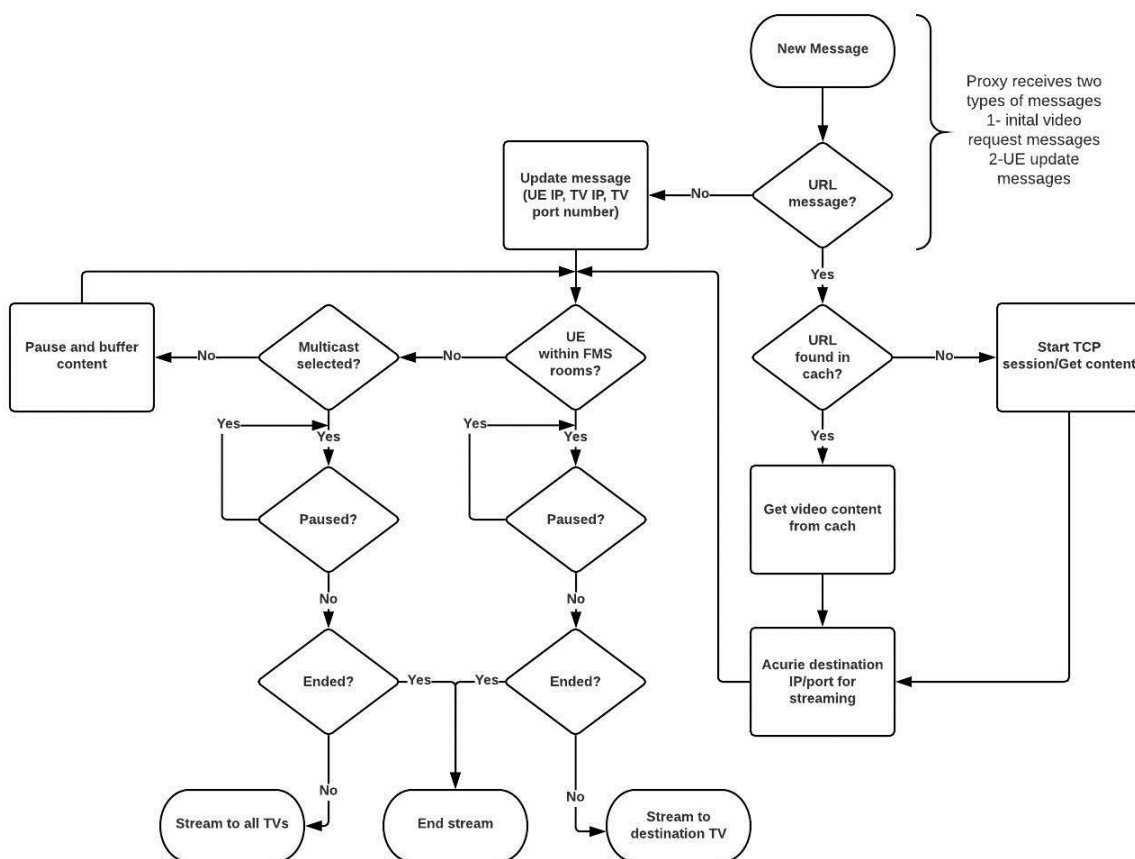


Figure 2-62: Proxy server flowchart

2.7.1.4 Application GUI

When the user turns on his Follow Me app, the first screen appears as shown in Figure 2-63. The user will select FMS and will be registered as FMS client within the FMA.

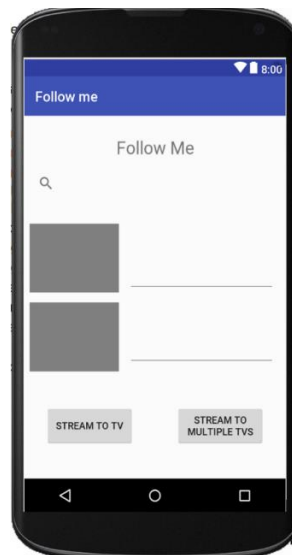


Figure 2-63: FMS GUI1

The client will have the ability to search for a video within the search bar, and the results will be shown below the search bar. After selecting a video, he will have the option to select Stream to TV or Stream to all TVs.

Client selections will be processed by the FMA along with the UE location information. This information will be used to stream the traffic flow to the correct TV screen/screens, and will automatically be updated as explained earlier to provide the FMS within IoRL coverage area.

Also, when the user requests streaming to all TVs, FMA will make the proxy server to stream the traffic to all TV addressees stored within the IoRL coverage area. **Figure 2-64** shows the second screen of the end user application, which enables the client to pause, end or return to the first page to select another video upon clients' preferences.



Figure 2-64: FMS GUI2

2.7.2 Mockupancy

The Mock Occupancy (MO) service enables the clients to control the home lighting system with great flexibility, allowing for full home lighting status view and control via simple Graphical User Interface (GUI) of the end user application.

2.7.2.1 Mock Occupancy architecture and operation

The MO coexist with the other available IoRL services, and has its own components to perform the required purpose, it consists of the front end, and backend user application as well as an SDN application.

The user views the entire home lighting system on his smartphone. Turning on/off one of the lights triggers signalling towards the backend of the user application. The SDN application listens to any change in the backend and sends packets towards L3/L2 processing with predefined format indicating the required state of the light as well as the ID of that light. This is achieved by exploiting the system knowledge about the light IDs and the global view of the SDN network over the entire deployment scenario.

2.7.3 Room Multicasting

Room multicasting service is also a video streaming service enabling the users to stream videos to group of registered users in a specific location regardless of their actual distance from the stream requester user.

2.7.3.1 Room multicasting architecture and operation

The architecture of the Room Multicasting service is like the FMS, but it uses the available system information differently. For example, in a conference venue with multiple halls, users can register themselves as listeners in a specific hall by using their end user application, thereby receiving a copy of any video being requested to be streamed in that specific hall, this enables the presenter to stream any video to all his audience no matter how large his hall is. Also, any of the audience can start receiving any other video by simply registering to another hall while they are walking into this new location. The SDN application modifies the flow rule of the OVS to add new destination for each newly registered user within the predefined hall premises.

2.7.4 TV Phone

This service enables users to use the nearest TV set to make a video call if they request to do so; However, it requires a TV set with a camera and microphone. As all the other proposed services it utilizes the UE location information to identify the nearest TV set to use it as mentioned earlier.

The user can use the end user application to make video call, SDN application forwards the uplink to the required destination, while forwarding the downlink to the nearest TV set based on the updated user location.

3 Light System Hardware

3.1 Pendant Light Rose

The pendant light rose is in advance design stage and these are the implementation process notes:

- a. The Light rose RRLH will be installed in the Home scenario and will be fitted with a pendant light and a strip light supplied by SFY.
- b. Internal components of the light rose RRLH:
 - i) mmWave
 - ii) VLC module
 - iii) 2 patch antennas
 - iv) Electricity to light fixture
- c. This is the updated design of the light rose (**Figure 3-1**):



Figure 3-1: Light rose

3.2 Ceiling Light

The Ceiling light is in its configuration stage, and these are the implementation process notes:

- 1) The ceiling light is part of the home scenario and can be fitted to a ceiling or a wall.
- 2) The ceiling light is not based in an SFY existing light and is designed from scratch.
- 3) Internal components:
 - i) mmWave
 - ii) VLC module
 - iii) 2 patch antennas
 - iv) LEDs panel light
 - v) Light diffuser
- 4) This is the updated design of the ceiling light (**Figure 3-2**):



Figure 3-2: Ceiling Light

3.3 Accessories

The accessories are two RRLH designs that will fulfill where the IORL project will be unable to replace the lights or interfere with the current light system and is divided into 2 accessories with LEDs(self-standing) and without LEDs(connects to a preinstalled LED light system).

- 1) The accessories will be used in the tunnel scenario, at the station where it is impossible to replace the light strips at the platform.
- 2) The accessories do not use SFY light fixtures.
- 3) Internal components of accessory with LEDs:
 - i) mmWave
 - ii) VLC module
 - iii) 2 patch antennas
 - iv) LEDs panel light
 - v) Light diffuser
- 4) Internal components of accessory without LEDs:
 - i) mmWave
 - ii) VLC module
 - iii) 2 patch antennas
- 5) This is the updated design of the accessory without LEDs (**Figure 3-3**):



Figure 3-3: Accessory with LEDs

6) This is the updated design of the accessory without LEDs (**Figure 3-4**):

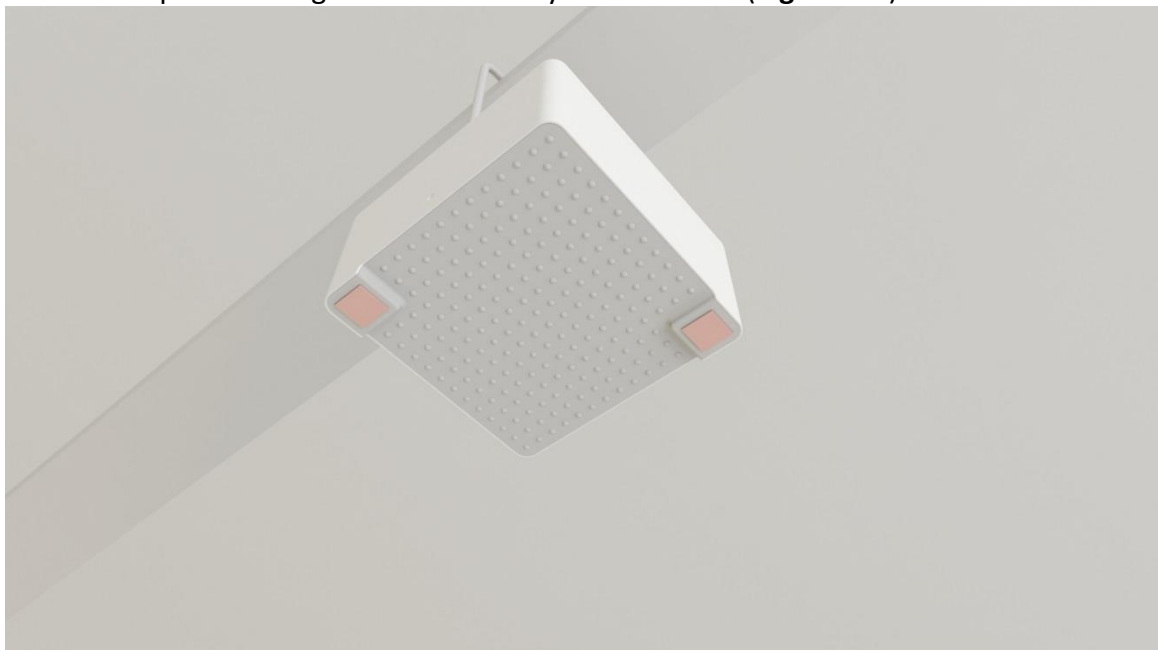


Figure 3-4: Accessory without LEDs

3.4 Strip Light

The Strip light from the Tunnel scenario that are in the platform today uses fluorescent lights, our intention is to replace these with retrofit LED tubes and connect it to the accessories we have designed, this will perform as a hybrid RRLH constructed from the LED tubes from the current light system in the platform and the RRLH accessories.

3.5 Pendant Strip Light and Pendant lights

The proposed pendant light solution works both as a pendant light and pendant strip light. The pendant strip light will be chosen from the SFY collection and will be connected to the Light rose RRLH for the home scenario.

3.6 IP-65 Light

The IP-65 light are lights inside the train tunnel in the tunnel scenario, the design process for this light is still in discussion as the tunnel scenario leaders are yet to finalise scenario location. The design is intended to be finished by June 2019.

3.7 Spot Light

The spot light is part of the museum scenario, and to be connected to a rail system in the museum, design will be done by June 2019.

3.8 UE Trolley

The UE Trolley will be for the museum scenario (and others according to testimonial demands) and will be a portable station to connect to the RRLH, will be done by September 2019.

3.9 Backpack

The User Equipment Backpack aims to provide a portable method of transporting the large prototype receiver designed by Viavi Solutions. Due to the size of this current model, which is expected to remain the size of a small desktop by the end of the project, a backpack provides a sensible weight-bearing solution.

This will be suitable for scenarios such as the train tunnels where a portable wheeled kiosk would not be appropriate on the train tracks.

3.9.1 Considerations

Dimensions – The current dimensions provided are ‘the size of a small desktop’. This should be possible to further minimise by eliminating unnecessary casings and component housings.

Heat dissipation – Given the assumption that components would be packed closely in a small and non-breathable bag heat sinks, fans and perhaps an open style backpack would allow for greater cooling.

Power – Computers require large amounts of power, often requiring mains or adapted voltages. For this use, powerful batteries will be required to maintain portability of the device.

4 Platforms

4.1 Introduction

The IoRL project's main component costs are presented in **Table 4-1**, where the partner component suppliers are identified as either a IoRL project partner or external supplier.

Table 4-1: Cost of Main Components

Equipment Costings	Cost (EUR)	Provider
IHIPGW	7000	External Supplier
DRAN	10000	RunEL
RRLH Controller	10000	RunEL
mmWave RRLH	3500	FhG IIS
VLC RRLH	500	TH
mmWave UE	3500	FhG IIS
VLC UE	500	OLED COM
Light System	100	SFY
10MHz Clock	200	External Supplier
USRP	5039	External Supplier
Mini Server + PCIE 10G high speed ethernet	5000	External Supplier
UE Laptop + Graphics Driver	1300	External Supplier

The strategy of the IoRL project is to build laboratory benchtop systems on technology partner sites to the point where they are fully working and then ship the working systems to their demonstration scenario sites, scale and install them appropriately. The technology partner sites are:

- Home: Viavi Solutions/Leicester University
- Museum: ISEP
- Train Station: Brunel University
- Supermarket: Tsinghua University

The demonstrator scenario sites are:

- Home: Building Research Establishment, Watford
- Museum: Musée de la Carte à Jouer, Paris
- Train Station: Nuevos Ministerios railway station tunnel and platform, Madrid
- Supermarket: Leadcom

4.2 Laboratory Benchtop Platforms

The laboratory benchtop systems for the individual Home, Museum, Train Station and Supermarket are shown in **Figure 4-1**, **Figure 4-3**, **Figure 4-5**, **Figure 4-7** respectively, and uses the Single Room VLC-mmWave configuration. The costs of the Single Room VLC-mmWave configuration that will be used for the individual Home, Museum, Train Station and Supermarket laboratory benchtop systems and the Two Rooms VLC-mmWave configuration that will be used for the Home, Museum, Train Station and Supermarket Demonstrators are presented in **Table 4-2**. The costs to each project component supplier and platform provider is presented in **Table 4-4**.

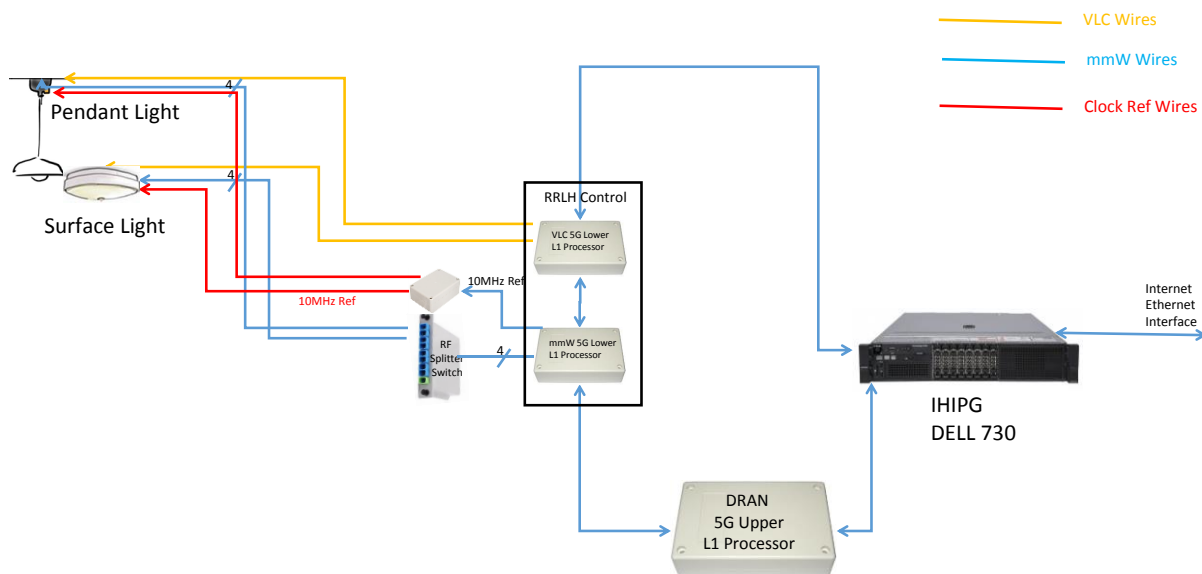


Figure 4-1: Benchtop Home Scenario

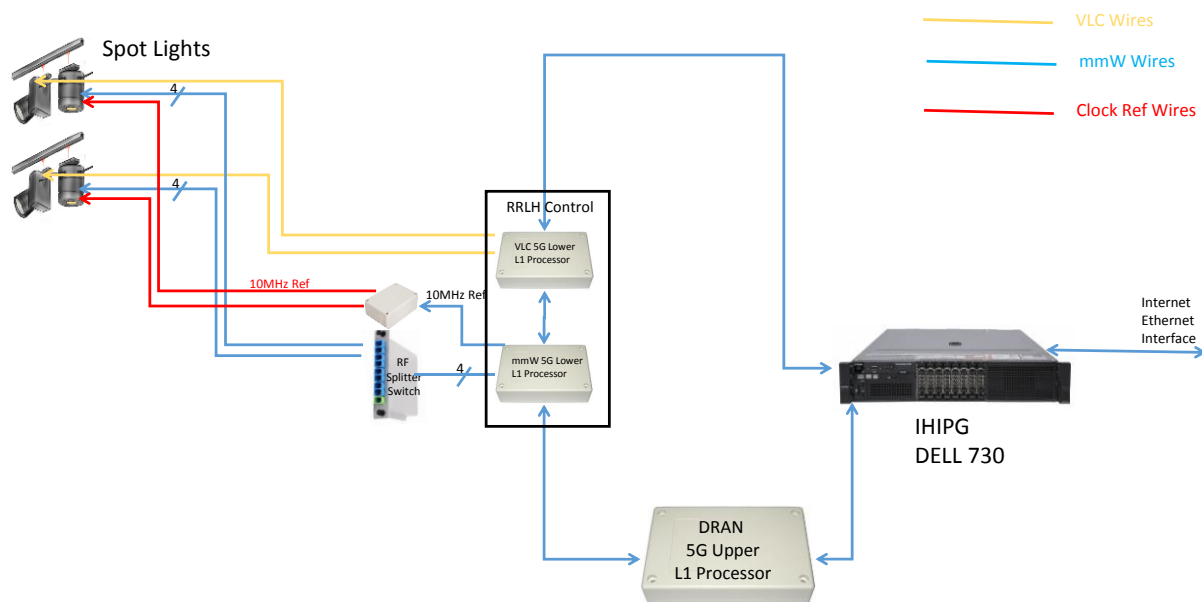


Figure 4-2: Benchtop Museum Scenario

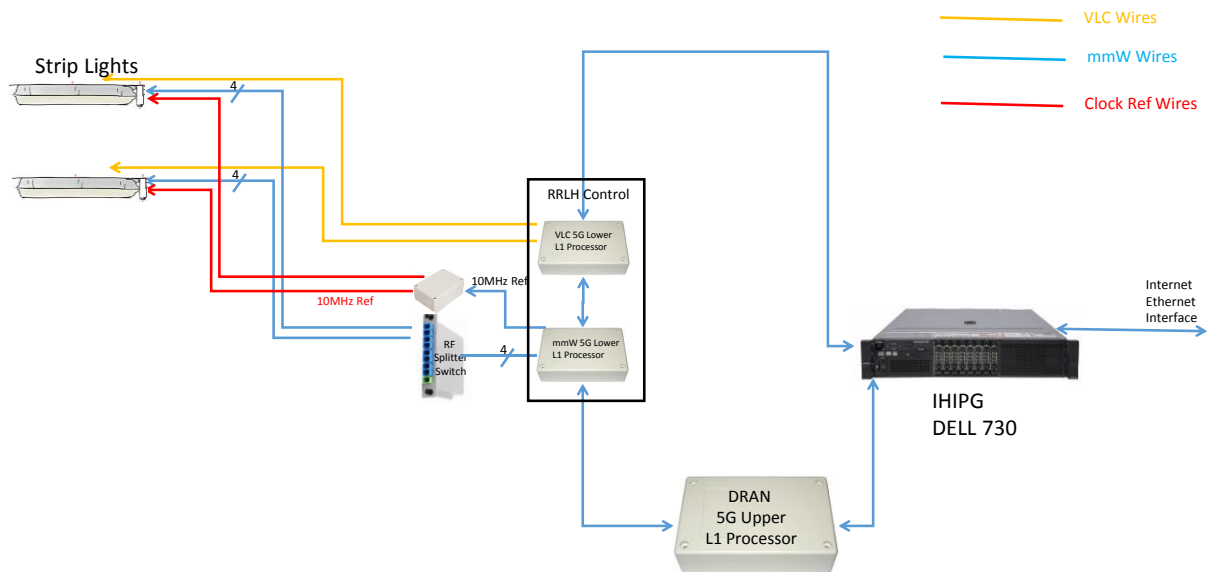


Figure 4-3: Benchtop Train Station

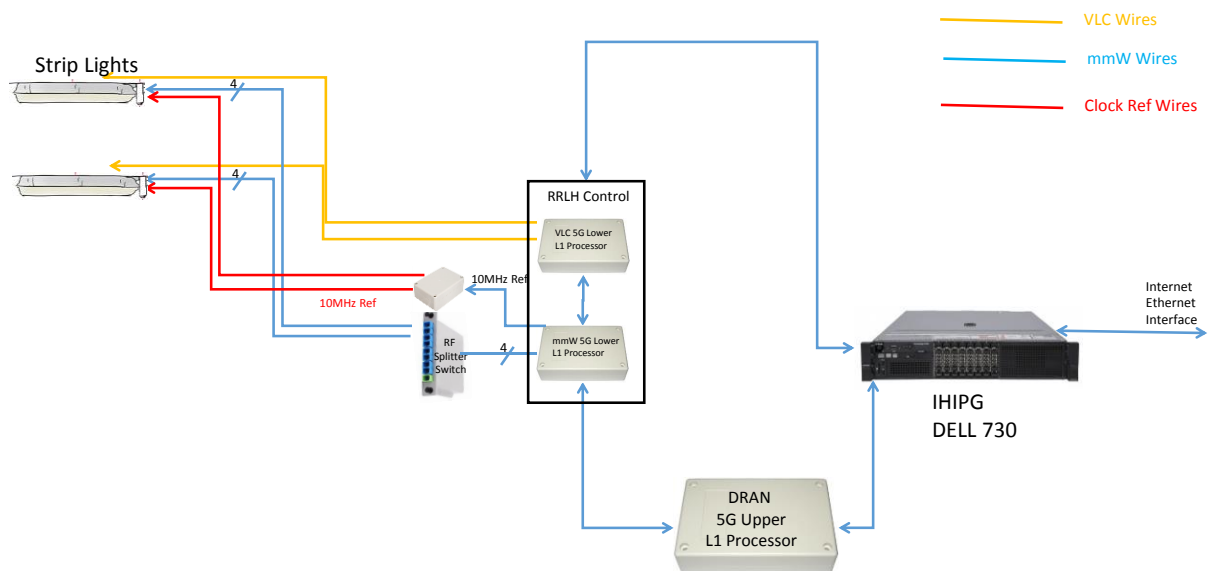


Figure 4-4: Benchtop Supermarket Scenario

In tables **Table 4-2** and **Table 4-3** and **Table 4-4** we present the results of a detailed costing exercise that the IoRL project performed. As a result, the project will seek some budget adjustments between partners in the frame of a Grant Agreement amendment to support the implementation of the systems accordingly.

Table 4-2: Cost of Each System Configuration

	External Supplier	RunEL	RunEL	FhG IIS	TH	SFY	External Supplier	FhG IIS	OLED COM	External Supplier	External Supplier	External Supplier	
System Configuration	IHIPGW	DRAN	RRLH Controller	mmW RRLH	VLC RRLH	Light System	10GHz Clock	mmW UE	VLC UE	USRP	Mini Server	UE Tablet	Total EUR
Single Room VLC-mmWave	1	1	1	2	2	2	1	1	1	1	1	1	50739
Two Rooms VLC-mmWave	1	1	2	8	8	8	2	1	1	1	1	1	85539

Table 4-3: Cost of Each Laboratory Benchtop Systems

Scenario	System Configuration	Location	System Cost	Total Cost in EUR
Home	Single Room VLC-mmWave	Watford	50739	50739
Museum	Single Room VLC-mmWave	Paris	50739	50739
Train Station	Single Room VLC-mmWave	Madrid	50739	50739
Supermarket	Single Room VLC-mmWave	Beijing	50739	50739
Total				202956

Table 4-4: Cost to each Project Component Supplier and Platform Provider

System Configuration	Project Component Supplier						Project Platform Provider				Total in EUR	
	RunEL DRAN	RunEL RRLH Control	FhG IIS	TH	OLED COM	SFY	ISEP	UBrunel	ULeic	Leadpcom		
Single Room VLC-mmWave	10000	10000	10500	1000	500	200			18539			
Single Room VLC-mmWave	10000	10000	10500	1000	500	200	18539					
Single Room VLC-mmWave	10000	10000	10500	1000	500	200		18539				
Single Room VLC-mmWave	10000	10000	10500	1000	500	200					18539	
Total in EUR	40000	40000	42000	4000	2000	800	128800	18539	18539	18539	18539	74156

4.3 Demonstrator Scenario Platforms

Sharing of hardware resources is required to construct the platforms for the demonstrator scenarios.

- Home: Viavi Solutions/Leicester University
- Museum: ISEP
- Train Station: Brunel University
- Supermarket: Tsinghua University

The demonstrator scenario sites are:

- Home: Building Research Establishment, Watford (**Figure 4-5**)
- Museum: Musée de la Carte à Jouer, Paris (**Figure 4-6**)
- Train Station: Nuevos Miniserios railway station tunnel and platform, Madrid (**Figure 4-7**)
- Supermarket: Leadpcom (**Figure 4-8**)

The first column in **Table 4-5** shows the main technology components of the Lab Benchtop systems.

The second column in **Table 4-5** shows that sharing of just 4 RRLHs between two lab benchtop platforms allows the construction of a system that supports a single room coverage area of 4 RRLHs enabling location estimation experiments to be performed. These components are small and compact enough to be easily transported between partner sites by air transport in hand-luggage.

The third column in **Table 4-5** shows that sharing of just 8 RRLHs between four lab benchtop platforms and two RRLH Controllers between two lab benchtop platforms allows the construction of a system that supports a two room coverage areas of 4 RRLHs each enabling intra handover experiments to be performed. These components are small and compact enough to be easily transported between partner sites by air transport in hand-luggage.

Table 4-5: Sharing Lab Benchtop Resources

	Lab Benchtop Configuration	Sharing Between 2 Lab Benchtops	Sharing Between 4 Lab Benchtops
IHIPG – Dell 730	1	1	1
5G Upper L1 Processor DRAN	1	1	1
RRLH Controller (5G Lower L1 mmWave Processor 5G Lower L1 VLC Processor 5G mmWave Splitter-Switch)	1	1	2
mmW and VLC RRLH (mmW: 4 antennas e.g. 2 dual pole TDD)	2	4	8
User Equipment (1 mmW TDD but not dual pole)	1	2	4

First each of the demonstrator platforms will be constructed and tested without the shared single RRLH Controller and six RRLHs and then the shared components will be transported to

the demonstration site and installed. This requires that the six shared mmWave and VLC modules be integrated into their respective light systems as itemized in **Table 4-6** and the timetable for scenario integration is given in **Table 4-7**. The schedule of the supermarket scenario is not considered here, as the Chinese partners will receive funding until the end of April 2021.

Since the conference scenario is similar to the supermarket and train station scenarios, these platforms could be used in the BMSB 2020 conference, which is scheduled for June 2020, and thus can be covered only if the project is extended.

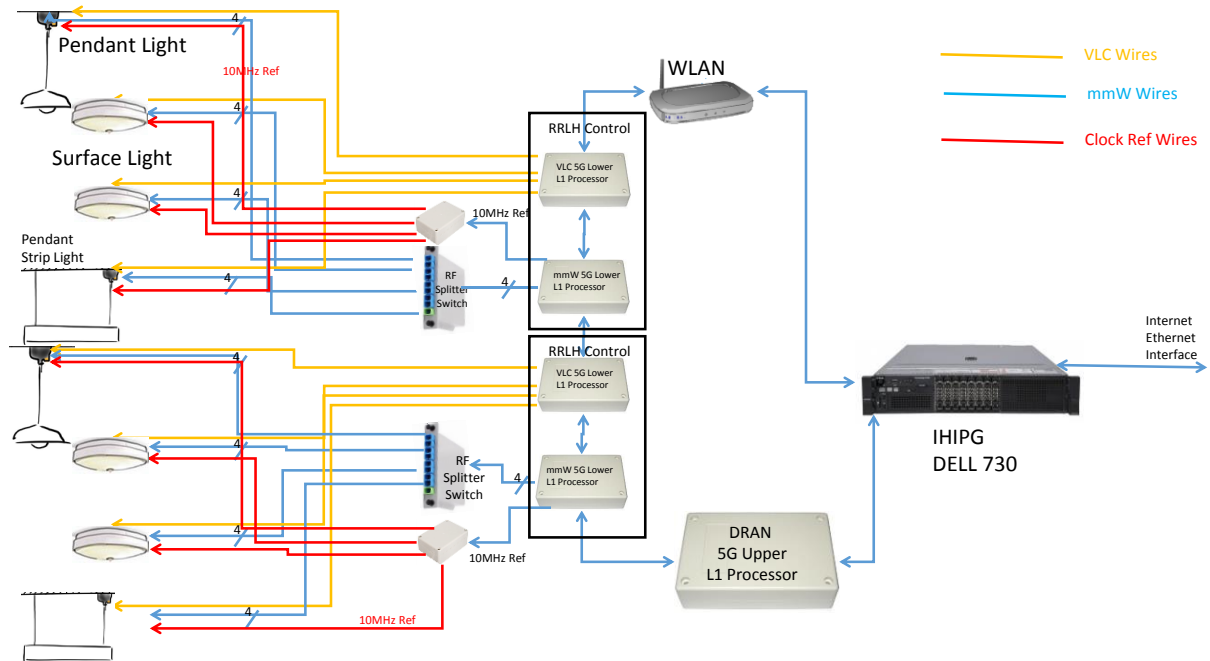


Figure 4-5: Demonstrator Home Scenario

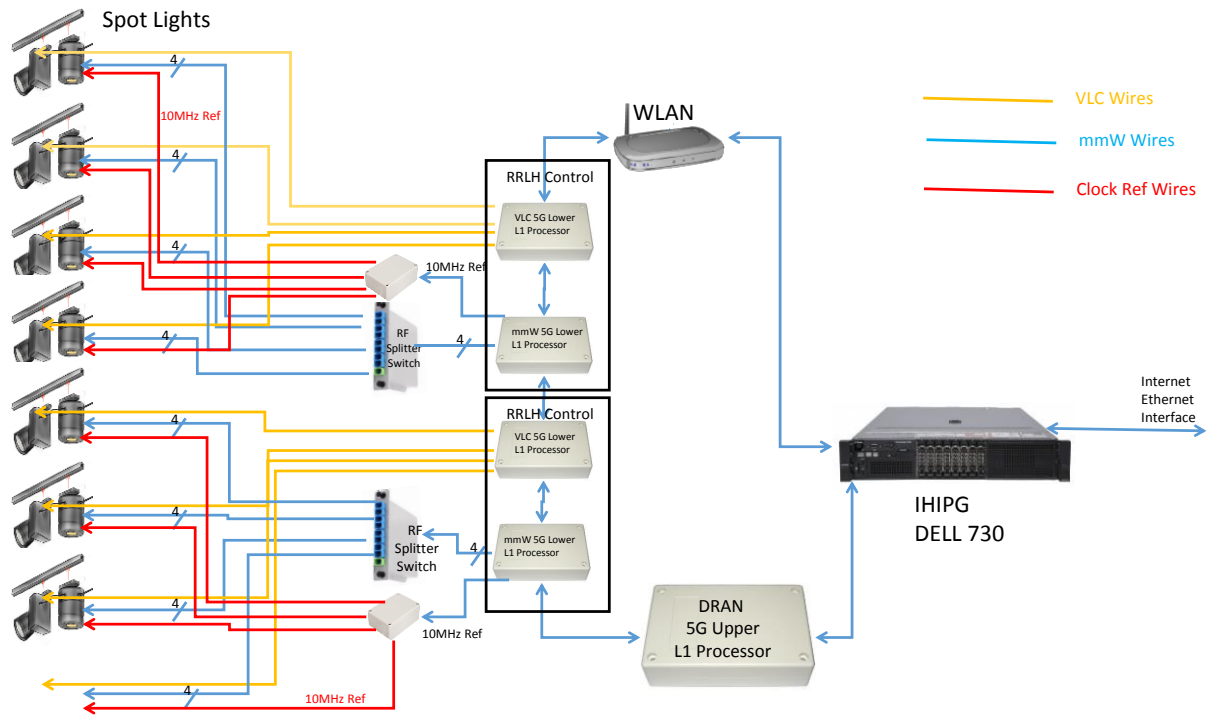


Figure 4-6: Demonstrator Museum Scenario

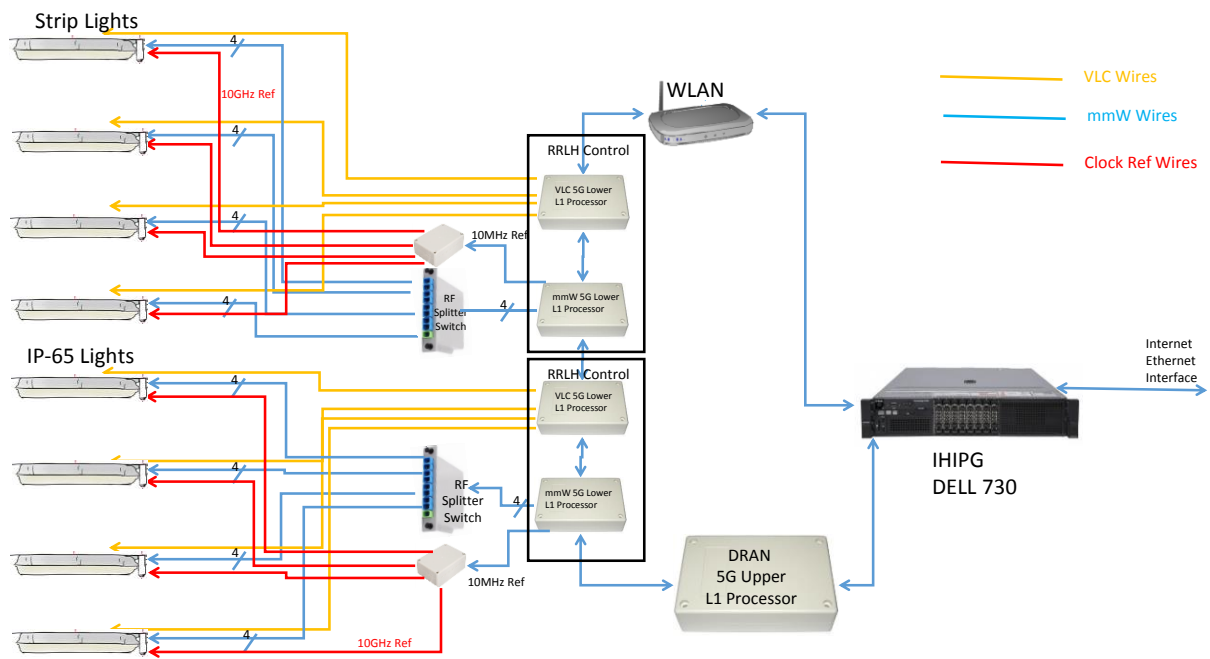


Figure 4-7: Demonstrator Train Station

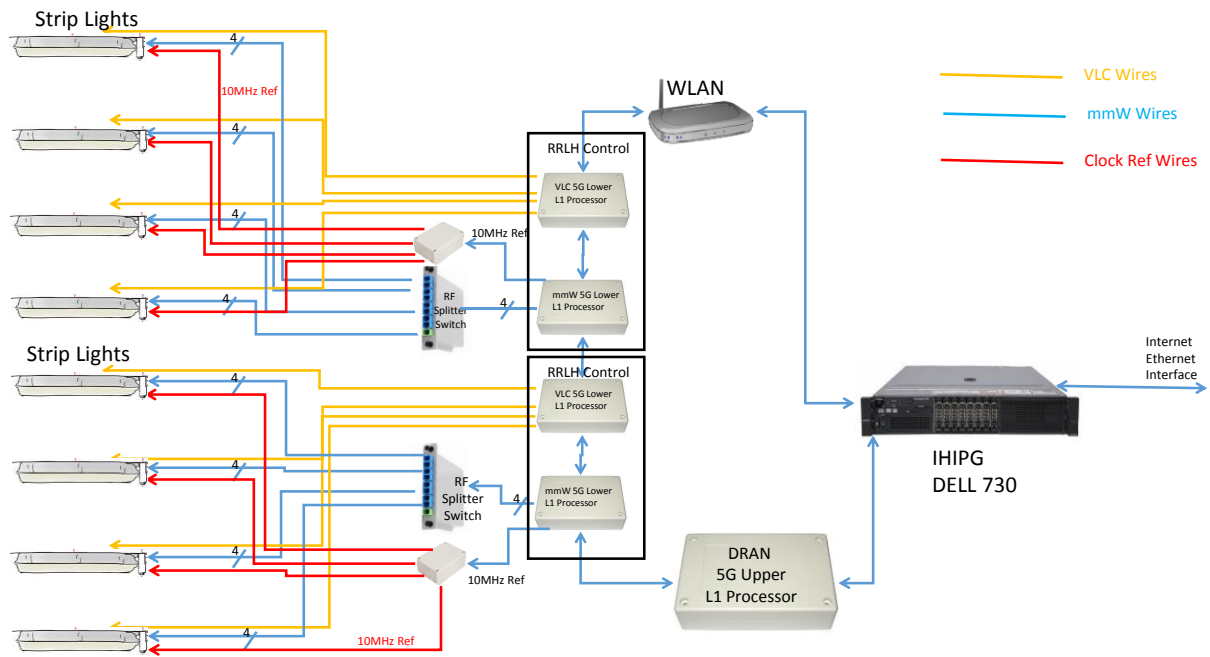


Figure 4-8: Demonstrator Supermarket Scenario

Table 4-6: Light Systems Required for Each Demonstration Platform

	Home	Museum	Train Station Tunnel	Supermarket	Conference
Spot Lights		8			
Strip Lights			4	8	8
IP-65			4		
Pendant Strip lights	2				
Pendant lights	2				
Surface Lights	4				

Table 4-7: Timetable for Scenario Integration

Platform	Apr, May, Jun 2019	July, Aug, Sept 2019	Oct, Nov, Dec 2019	Jan, Feb, Mar 2020	Apr, May, Jun 2020
Portable Lab Demonstrator	First Version Complete for EU CNC 2019				
BRE Home		Cabling	field trial (4 lights in kitchen & 4 in sitting room)		
Train Station			Cabling	field trial (4 lights on platform & 4 in evacuation tunnel)	
Museum				Cabling	field trial (4 lights around one exhibit & 4 lights around a second exhibit)

References

- [ECMAS] <https://caniuse.com/#feat=es5>
- [DOCK] <https://www.docker.com/>
- [DOCKC] <https://docs.docker.com/compose/>
- [IoRLD2.1] IoRL Deliverable D2.1 – Definition and Description of the IoRL Use Cases and Derivation of User Requirements, <https://doi.org/10.5281/zenodo.2577990>
- [IoRLD3.1] IoRL Deliverable D3.1 – SDN/NFV environment in the Home Network, <https://doi.org/10.5281/zenodo.2579869>
- [IoRLGR] IoRL GitHub repository at <https://github.com/H2020-5G-IoRLproject/H2020-IoRL-code>
- [IoRLYC] IoRL YouTube channel at <https://www.youtube.com/channel/UCx7iquE3GHoGEgjduswZ7iw>
- [MSE] <https://caniuse.com/#feat=mediasource>
- [NOLO] <https://www.nolovr.com/index>
- THETA] <http://theta360.guide/community-document/live-streaming.html>
- [VJS] <https://videojs.com/>
- [VJS-VR] <https://github.com/videojs/videojs-vr>
- [WEBVR] <https://webvr.info/developers/>

Annex A Installation of the MS-Stream application

A.1. Browser compatibility

The web application is developed with ECMAScript 5 which is supported by all of the modern browsers. The complete list of compatibility can be found at [\[ECMAS\]](#).

The MS-Stream player itself uses the Media Source Extension which is also supported by modern browsers, except on iOS devices. The complete list of compatibility can be found at [\[MSE\]](#). On iOS devices the MS-Stream player is not used and replaced by a traditional HLS player without multi-source streaming.

A.2. Using the VNF

The VNF is delivered as a qcow2 image disk running an Ubuntu server guest with embedded docker containers. The docker containers should start automatically. The default login credentials of the application running inside of the VNF are:

<p style="text-align: center;">login: iorl password: msstream-iorl</p>
--

The VNF can be run using qemu and for example the following configuration:

```
qemu-system-x86_64 -enable-kvm -cpu host -m 2048 -hda msstream-vnf.qcow2 -net user,hostfwd=tcp::10080-:80,hostfwd=tcp::10022-:22 -net nic
```

This first version of the VNF contains a webserver with a database and a video transcoder as an all-in-one VNF.

As the video storage is inside of the VNF, so the disk space needed will increase when a new video is uploaded and transcoded.

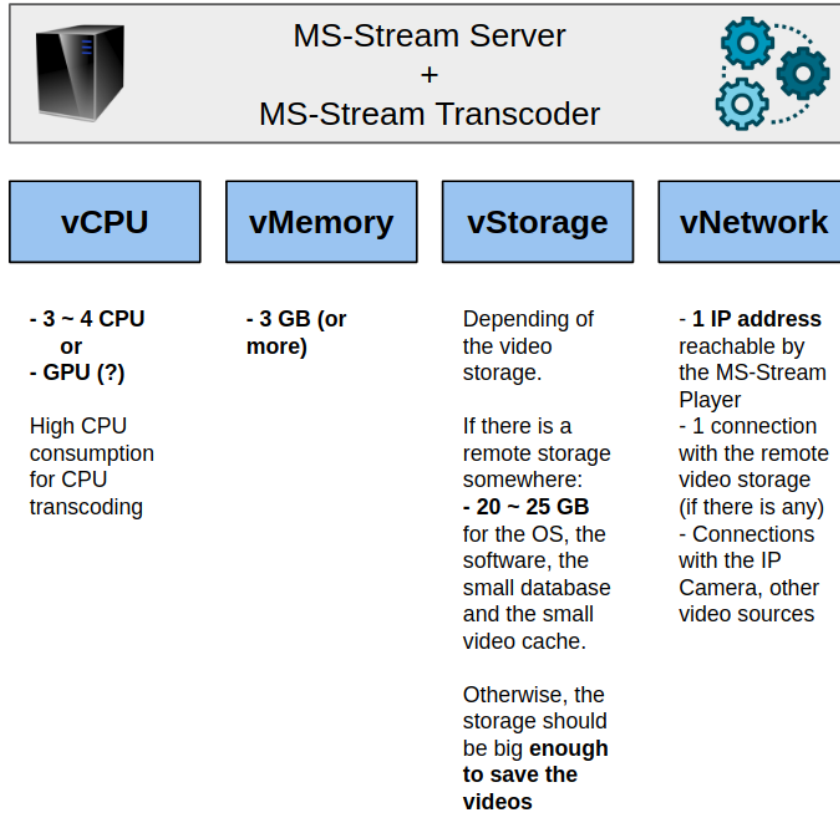


Figure A-1: Optimal requirements of the VNF

If needed, the login/password of the guest OS are:

login: msstream-iorl
password: msstream-iorl

To test if the VNF is up and running, the user can open a browser and go to: ***http://<VNF_address>/*** and should see a website like the one in **Figure A-2**.

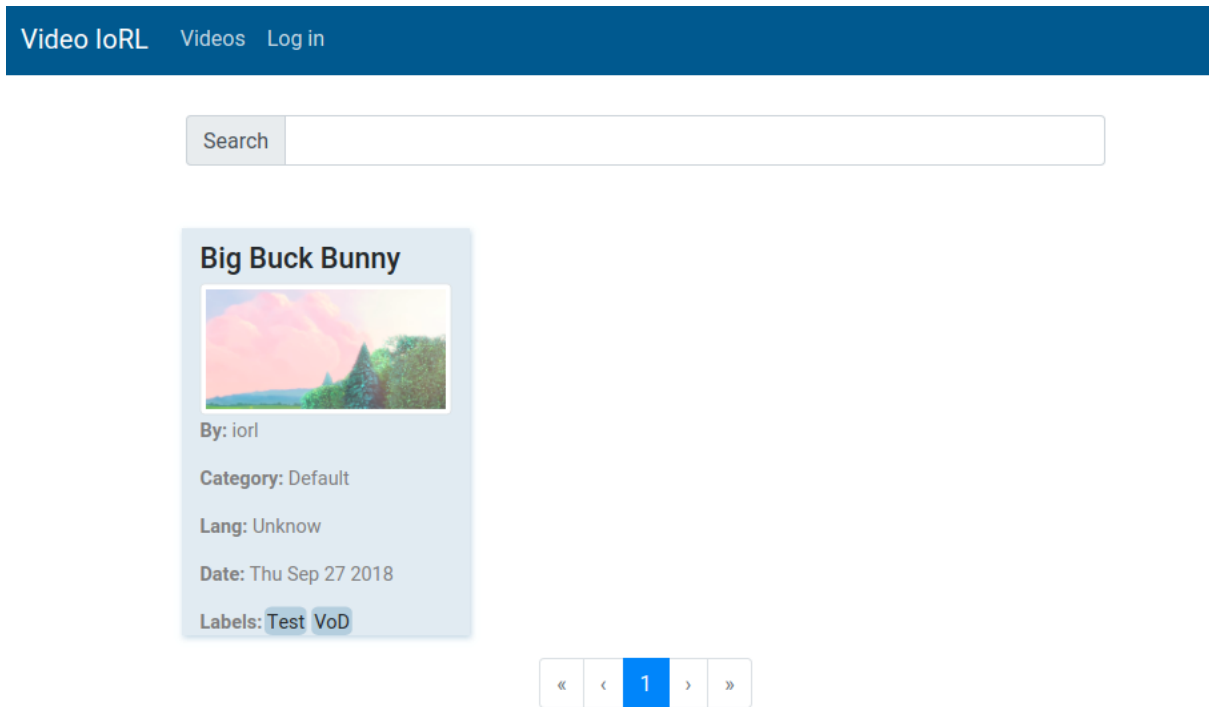


Figure A-2: Homepage of the application

By clicking on the video and then on the “Play” button without modifying the network address the simple video streaming can be tested.

A.3. Bare-metal installation with Docker

The MS-Stream application can be deployed using Docker, and specific Docker images built by Joadá.

To start with the computer needs to be connected to the Internet and both Docker **[DOCK]** and Docker-Compose **[DOCKC]** should be installed on the server.

First of all, pull the required Docker image from the official registry:

```
docker pull msstream/videoiorl
docker pull msstream/rtmpserver
docker pull mongo
docker pull nginx:alpine
```

Then, the following 3 files need to be created inside the directory: **docker-compose.yml**, **nginx.conf** and **root_infos.env**.

docker-compose.yml:

```
proxy:
```



```
image: nginx:alpine
restart: always
log_opt:
  max-size: 10M
ports:
  - "80:80"
  - "443:443"
links:
  - videoiorl
volumes:
  - ./nginx.conf:/etc/nginx/nginx.conf:ro

videoiorl:
image: msstream/videoiorl
expose:
  - "3000"
volumes:
  - ./static_vid/./static_vid/
links:
  - mongodb
env_file:
  - root_infos.env
environment:
  - DATABASE=mongodb
log_opt:
  max-size: 10M
restart: always

mongodb:
image: mongo
container_name: mongoiorl
expose:
  - "27017"
```

```
restart: always
log_opt:
  max-size: 10M
volumes:
  - ./data:/data/db

rtmpserver:
image: msstream/rtmpserver
ports:
  - "1935:1935"
restart: always
log_opt:
  max-size: 10M
```

nginx.conf: this file contains the rules of the redirections. It can be modified to add new services and/or HTTPS support.

```
events {}

http {
  server {
    listen 80 default_server;
    client_max_body_size 0;
    location / {
      proxy_pass http://videoiorl:3000;
      proxy_set_header Host $http_host;
      proxy_set_header X-Real-IP $remote_addr;
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
      proxy_set_header X-Forwarded-Proto $scheme;

      proxy_http_version 1.1;
      proxy_set_header Connection "";
    }
  }
}
```

```
}  
}  
}
```

root_infos.env: this file contains the credentials for the admin user.

```
VIDEOIORL_ROOT_USERNAME=iorl  
VIDEOIORL_ROOT_PASSWORD=msstream-iorl  
VIDEOIORL_ROOT_MAIL=IoRL-Contact@5g-ppp.eu
```

When the files are created the application can be started as follows:

```
docker-compose up -d
```